Universidad Autónoma
de Madrid

Escuela Politécnica Superior
Departamento de Ingeniería Informática

Contact recommendation in social networks: algorithmic models, diversity and network evolution

A dissertation written by
Javier Sanz-Cruzado Puig
under the supervision of
Pablo Castells Azpilicueta

Madrid, January 2021

PhD thesis title: Contact recommendation in social networks: algorithmic models, diversity and network evolution

Author: **Javier Sanz-Cruzado Puig**

Affiliation: Departamento de Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid, Spain

Supervisor: **Pablo Castells Azpilicueta**
Universidad Autónoma de Madrid, Spain

Date: January 2021

Committee: **Álvaro Barreiro**
Universidade da Coruña, Spain

Simone Santini
Universidad Autónoma de Madrid, Spain

Ido Guy
eBay, Israel

Nicola Barbieri
Tumblr, USA

Enrique Amigó
Universidad Nacional de Educación a Distancia, Spain

Abstract

Online social networks like Twitter, Facebook and LinkedIn are used daily by hundreds of millions of people to connect with other users around the world and share information with then. The massive scale of these platforms has led to the development of automated tools to prevent users from being overwhelmed by the vast amount of information they have access to in this sites. Recommender systems appear as a family of these tools, designed to make individual suggestions of items or people that user might be interested in according to their past personal preferences.

This thesis focuses on the study of a particular problem in the confluence between online social networks and recommender systems: the problem of finding users in the network with whom the people wants to connect – the problem known as contact recommendation. We explore this problem from three different perspectives.

We first aim to identify the factors leading to the development of effective contact recommendation approaches, targeting the density of the network. For this, we explore the relation between contact recommendation in social networks and text information retrieval. Considering a collaborative filtering perspective, we explore the utility of adapting search-based models for their use in three different aspects of contact recommendation: as direct recommenders, as similarity measures and as samplers and features in learning to rank. Thorough experiments over Twitter and Facebook samples show their effectiveness for the three roles when compared to the best state of the art approaches.

We also explore the potential of contact recommendation algorithms to drive the evolution of social networks towards desirable properties of the network as a whole – instead of aggregating the isolated gains of each user. We investigate the definition of novel diversity metrics that quantify the effects of people recommendation approaches over the structure of the network, with a particular focus on notions of structural diversity and weak ties. Over samples from Twitter, we find that recommending weak ties leads to an increase on the novelty and diversity of the information that reaches the users in the network, with potential implications in the mitigation of filter bubbles.

Finally, we analyze the recommendation task as an interactive cyclic process, where information is constantly exchanged between the users and the system. We develop a simple stochastic approach, based on the classical user-based k nearest neighbors collaborative filtering algorithm, that deals with the uncertainty of the available data for selecting the optimal neighbors of the user we want to generate recommendations for. We then explore its utility for dealing with cold starts situations over a wide variety of datasets from different recommendation domains – including contact recommendation as a particularly compelling one.

Keywords: recommendation, social networks, evaluation, diversity, novelty, multi-armed bandits, axiomatic analysis, information diffusion, weak tie

Resumen

Las redes sociales online como Twitter, Facebook y LinkedIn son utilizadas diariamente por cientos de millones de personas para establecer conexiones con usuarios alrededor del mundo y compartir información con ellos. La enorme escala de estas plataformas ha dado lugar al desarrollo de herramientas automáticas para evitar que los usuarios se vean desbordados por la cantidad de información a la que tienen acceso. Los sistemas de recomendación son una familia de estas herramientas, diseñada para sugerir a cada usuario aquellos ítems o personas en los que puedan estar interesados en función de sus preferencias personales pasadas.

Esta tesis se centra en el estudio de uno de los problemas particulares que surgen de la confluencia entre las redes sociales online y los sistemas de recomendación: el problema de encontrar a aquellos usuarios de la red con los que la gente quiera conectar – un problema conocido como recomendación de contactos. Exploramos este problema desde tres perspectivas diferentes.

En primer lugar, queremos identificar los factores que llevan al desarrollo de soluciones efectivas para la tarea de recomendar contactos, tomando como objetivo incrementar la densidad de la red. Para ello, exploramos la relación existente entre la recomendación de personas en redes sociales y la recuperación de información basada en texto. Considerando en todo momento una perspectiva basada en filtrado colaborativo, exploramos la utilidad de adaptar modelos de búsqueda para su uso en tres aspectos diferentes de la recomendación de contactos: como recomendadores directos, como medidas de similitud en estrategias de vecinos próximos y como características y algoritmos de muestreo en técnicas de aprendizaje de rankings. Mediante varios experimentos sobre muestras de Twitter y Facebook observamos la efectividad de estos modelos en los tres papeles, en comparación con los mejores algoritmos del estado del arte.

También exploramos el potencial que los algoritmos de recomendación de contactos tienen para conducir la evolución de las redes sociales hacia propiedades deseables para la red en su conjunto – en lugar de agregar las ganancias individuales de cada usuario. Investigamos la definición de métricas novedosas de diversidad que cuantifican los efectos de recomendar personas sobre la estructura de la red, con un interés particular en el concepto de la diversidad estructural y de los enlaces débiles. Sobre muestras de Twitter, observamos que recomendar enlaces débiles da lugar a una mejora en la novedad y la diversidad de la información que llega a los diferentes usuarios de la red, con posibles implicaciones en la eliminación de filtros burbuja.

Finalmente, analizamos la tarea de recomendación como un proceso cíclico e interactivo, en el que se produce un intercambio continuo de información entre los usuarios y el sistema. Desarrollamos una propuesta estocástica simple para esta tarea, basada en el algoritmo clásico de k vecinos próximos orientado a usuario, que tiene en cuenta la incertidumbre de los datos del sistema a la hora de seleccionar los mejores vecinos del usuario al que le queremos proporcionar recommendaciones. Exploramos la efectividad de este algoritmo para tratar situaciones de arranque en frío sobre una amplia variedad de conjuntos de datos, procedentes de diferentes dominios de recomendación – incluyendo, como caso particular e interesante la recomendación de contactos.

Palabras clave: recomendación, redes sociales, evaluación, diversidad, novedad, bandidos multi-brazo, análisis axiomático, difusión de información, enlace débil

Acknowledgements

Since I started this journey so long ago, I have had the pleasure to coincide with many wonderful people whom I would like to dedicate them a few lines.

First of all, I would like to thank my supervisor, Pablo Castells, for offering me the opportunity to work with him in the Information Retrieval Group for all these years. It seems a long time now when I entered his office to ask him to supervise my BSc thesis, but, since then, his support has been invaluable. A good supervisor is one of the keys to the success of a PhD thesis, and I cannot imagine a better one. Without him, this thesis would not have been the same. I hope to keep working with him and continue learning from him for a long time.

Next, I would like to thank all the people with whom I have had the pleasure to live in the B-408 office. Here, I specially want to thank two people: Rocío Cañamares and Pablo Sánchez, who have shared the way towards their PhDs with me. Thank you both for all the conversations, meals and good moments that we have shared. I could not, however, forget the rest of the people I have happened to meet there: Ignacio Fernández, Sofía Marina Pepa, Daniel Valcarce, Walter Anelli...: all of you have contributed towards finishing this thesis. That is also true for the rest of professors of the Information Retrieval Group, from whom I have learned a lot: Alejandro Bellogín, Ivan Cantador and Fernando Díez.

I would also like to thank Iadh Ounis and Craig Macdonald for their supervision during my internship at the University of Glasgow. I believe that I am a much better researcher thanks to how much I learnt from them. I also want to thank all the members of the Terrier Team who welcomed me there and made me feel like home. Thanks to them, I feel a bit Glasgewian at heart. From all of them, special thanks to all the PhD students and postdocs who worked with me at the Fiii office: Ting Su, Xi Wang, Graham McDonald, Zaiqiao Meng and Amir Jadidinejad. I hope to see you all soon and have some drinks and ... maybe play some tennis?

No veo esta tesis solo como el final de mis estudios de doctorado, sino como el final de mis estudios universitarios (al menos de momento). Por ello, querría dedicar también unas palabras a todas aquellas personas que me han acompañado desde hace ya más de 10 años. En primer lugar, quiero dar las gracias a todos aquellos que han trabajado conmigo, pasando el día entero en los laboratorios de la Escuela para realizar esas eternas prácticas (espero no dejar fuera a nadie – si fuera así y lees esto, gracias a ti también): Carlos, Euler, Rus, Laura, Dani, Rafa, Óscar, Mónica, Santos, Fede, y, sobre todo, Rubén, quien no solamente tuvo que soportarme durante el instituto, sino que, además, fue mi inseparable compañero de laboratorio durante la mayor parte de la carrera. No todo fue trabajo, así que también he de dar las gracias a los "gilís", con los que tantos fines de semana he pasado jugando al Smash, a Dungeons and Dragons y a miles de cosas más: Relinque, Víctor, Republicano, Álex, Guinot..., gracias.

Fuera del ámbito universitario, quiero dar también las gracias a todos aquellos amigos que han estado ahí, apoyándome, durante todos estos años. María, Mónica, Nadia, Ana, Dani, Arabia, Tatiana, Jose... sin vuestra amistad, llegar hasta donde estoy habría sido infinitamente más complicado. Espero seguir disfrutándola durante muchísimos años.

Finalmente, pero no por ello menos importante (más bien todo lo contrario), quiero dar las gracias a mi familia, y, en concreto, a mis padres y a mi hermano. Sé que siempre habéis estado allí cuando lo he necesitado y siempre lo estaréis. No podré nunca agraceder lo que habéis hecho por mí. Esta tesis va dedicada, sobre todo, a vosotros.

Javier Sanz-Cruzado
December 2020

Contents

List of Figures

List of Tables

Acronyms

Notation	Description
AEW	average edge weakness.
API	application programming interface.
ARSL	average reciprocal shortest path length.
ASL	average shortest path length.
AUC	area under the ROC curve.
BFS	breadth-first search.
BIR	Binary Independent Retrieval.
CB	content-based.
CCC	complementary clustering coefficient.
CEGC	community edge Gini complement.
CF	collaborative filtering.
CLNC	candidate length normalization constraint.
CLUB	CLUstering of Bandits.
DCG	discounted cumulative gain.
DFR	divergence from randomness.
DGC	degree Gini complement.
EBM25	extreme BM25.
EHR	external hashtag rate.
ERR	expected reciprocal rank.
ERRIA	intent-aware expected reciprocal rank.
EW-CLNC	edge weight - candidate length normalization constraint.
EWC	edge weight constraint.
FOAF	friends of a friend.
HGC	hashtag Gini complement.
HITS	Hypertext Induced Topic Selection.
HSD	honest significant differences.
HUGC	hashtag user Gini complement.
IB kNN	item-based k nearest neighbors.
ICEGC	inter-community edge Gini complement.
idf	inverse document frequency.
IDGC	in-degree Gini complement.
ILD	intra-list dissimilarity.
iMF	implicit matrix factorization.
InterPMF	interactive probabilistic matrix factorization.
IR	information retrieval.
kNN	k nearest neighbors.
LHN	Leicht-Holme-Newman.
LNC	length normalization constraint.
LTN	long tail novelty.

Notation	Description
MAB	multi-armed bandits.
MAE	mean average error.
MAP	mean average precision.
MC	modularity complement.
MCN	most common neighbors.
ML	machine learning.
MMR	maximum marginal relevance.
MPD	mean prediction distance.
MRR	mean reciprocal rank.
NDC	neighbor discrimination constraint.
nDCG	normalized discounted cumulative gain.
ODGC	out-degree Gini complement.
PGC	predicted Gini complement.
PMF	probabilistic matrix factorization.
PRP	probability ranking principle.
QL	query likelihood.
QLD	query likelihood with Dirichlet smoothing.
QLJM	query likelihood with Jelinek - Mercer smoothing.
QLL	query likelihood with Laplace smoothing.
RAE	reciprocal average eccentricity.
RD	reciprocal diameter.
RL	reinforcement learning.
RMSE	rooted mean square error.
ROC	receiver operating characteristic.
RSJ	Robertson - Spärck-Jones formula.
SALSA	Stochastic Approach for Link Structure Analysis.
TDC	term discrimination constraint.
tf	term frequency.
TF-LNC	term frequency - length normalization constraint.
TFC	term frequency constraint.
UB kNN	user-based k nearest neighbors.
UCB	upper confidence bound.
VSM	vector space model.

1 Introduction

1.1 Motivation

Online social networks represent the epitome of the transformation of the World Wide Web towards a more participatory service, driven by user-generated contents and communication. Platforms like Facebook, Twitter, LinkedIn, Instagram or Tik Tok assist millions of people in contacting and establishing meaningful connections with users located anywhere on Earth or in sharing photos, interests or feelings with them. Conceived in the late 1990s, with sites like SixDegrees or Friendster (boyd and Ellison, 2007), the adoption of these services by the general public has been massive, with several of them even appearing among the most visited web sites.[1] Their importance has been constantly growing, making a great impact on modern society and lifestyle with – both positive and negative – effects on several aspects like privacy, health (O'Keeffe et al., 2011), politics (Hong and Kim, 2016) and how people relate to each other.

If, as of 2013, around 73% of the U.S.A. adult population already admitted using at least one social networking site, and 42% of them having user accounts on several of them,[2] nowadays active users in Twitter, LinkedIn and Facebook are counted in the range of hundreds of millions around the world. Also, every second, thousands of user-generated contents are published on each of these platforms (according to recent statistics,[3] more than 9,000 tweets and 1,000 photos are posted, respectively, on Twitter and Instagram in that period of time). A consequence of dealing with this data magnitude is that users might suffer from *social overload* (Guy, 2015), which appears as the combination of two phenomena: on the one hand, the amount of data the average citizen can access on any social networking site has become impossible to process manually – a problem known as information overload; on the other, users might participate in such a vast amount of interactions at the same time, that they are not able to cope with them – or, in other words, they suffer from interaction overload. The social overload problem raises the need for automatic tools able to curate, summarize or filter the contents of the platform, so they become accessible to the users.

Recommender systems (Adomavicius and Tuzhilin, 2005, Ricci et al., 2015) represent a family of tools for alleviating these problems, designed for assisting users in filtering the available information and discovering products, content or people that bring them value. With their inclusion in a wide spectrum of applications, their popularity has risen in the last two decades. Today, audiovisual streaming services (Netflix, HBO, Spotify) provide suggestions about movies to watch or songs to listen to, e-commerce platforms like Amazon or eBay recommend products to buy , academic repositories (Mendeley, Google Scholar) recommend scientific articles to read. Online social networks represent yet another application domain for these systems.

The confluence between online social networks and recommender systems has given rise to new and interesting challenges and perspectives (Guy, 2015). The availability of social connections between users introduces potentially useful links between recommendation, social network analysis and network science which can be exploited to improve item recommendations (Chaney et al., 2015, Fan et al., 2019, Tang et al., 2013). Another highly interesting case emerges when, instead of suggesting items, we recommend people in the network with whom users might want to engage (Guy, 2018, Hannon et al., 2010). This problem is known as contact recommendation.

This particular recommendation perspective adds to the availability of links between users the distinctive property that users and items are no longer separate spaces – both of them are extracted from the same set. People recommendation has grown into a fundamental service for online social network platforms, and has

[1] Alexa's top 50 sites: https://www.alexa.com/topsites (Accessed 19th December 2020)

[2] Social Media Update 2013: https://www.pewresearch.org/internet/2013/12/30/social-media-update-2013/ (Accessed 19th December November 2020)

[3] Internet Live Stats (1 second): https://www.internetlivestats.com/one-second/ (Accessed 19th December 2020)

received attention from both academia (Backstrom and Leskovec, 2011, Hannon et al., 2010) and industry (Goel et al., 2015, Gupta et al., 2013, Guy, 2015, Satuluri et al., 2020). The most prominent social media platforms started to feature contact recommendation services by the second half of the 2000s decade: among others, Twitter has its "Who to follow" service, and Facebook and LinkedIn have each a version of "People you may know". The research in this thesis focuses on this problem from different perspectives.

The first perspective considers the development of effective algorithms that seek to maximize the density growth of the network – in other words, the amount of links they cause to appear in the network. It is the primary take on contact recommendation in the literature, and has inspired the creation of a wide variety of approaches from diverse fields – including network science (Liben-Nowell and Kleinberg, 2007, Lü and Zhou, 2011, Martínez et al., 2017), machine learning (Hasan et al., 2006, Lichtenwalter et al., 2010), recommender systems (Hannon et al., 2010) and, to a lesser extent, text information retrieval (Hannon et al., 2010).

Although the relation between text information retrieval (Baeza-Yates and Ribeiro-Neto, 2011) – oriented to the creation of search engines – and recommender systems has been largely studied (Adomavicius and Tuzhilin, 2005, Bellogín et al., 2013, Parapar et al., 2013, Wang et al., 2008a,b), new and interesting relations between both tasks can be defined when users and items are the same, allowing the creation of novel approaches based on the adaptation of search-oriented models. What is more: since these models follow established principles (Fang et al., 2004, 2011) in their design, the study of this relation can provide useful information about the mechanisms driving the evolution of social networks – which, in turn, might lead to the development of even better recommendation approaches.

Contact recommendation has come to play a substantial role in the growth of social networks: a considerable fraction of the new links in social networking sites are created as a result of automatic recommendation (Aiello and Barbieri, 2017, Goel et al., 2015, Guy, 2018). Different algorithms may hence lead to the appearance of very different sets of ties and, therefore, different network structures. As networks are not about isolated people, but about their connections and interactions, these new edges might affect not only the close environment of users, but the entirety of the network. For example, these structural changes might lead to positive effects, like a reduction of the polarization of the users (Musco et al., 2018), but also negative ones like the creation of filter bubbles (Pariser, 2011) or the perpetuation of discriminatory phenomena, like the glass ceiling effect (Stoica et al., 2018). Consequently, comprehending how different algorithms can modify the properties of the network, and the consequences these changes have for the different actors involved (not only users, but also owners, shareholders or workers) is fundamental.

This represents the second perspective we consider in the thesis. Specifically, we focus on the impact contact recommendation has over several (past and novel) notions of structural diversity (Burt, 1995, De Meo et al., 2014, Easley and Kleinberg, 2010, Granovetter, 1973, Ugander et al., 2012) – which we observe as a potential positive characteristic of social networks. We also analyze the consequences that recommending contacts has on the novelty and diversity of the information reaching the users in the network when we use recommendations to enhance these structural diversity properties.

Finally, not only may recommender systems influence the behaviour of the users and cause changes in the network – the opposite is also true: in order to keep their suggestions relevant, contact recommendation strategies modify their behaviour according to the new connections that appear in the social platform. Recommendation has a cyclic nature, where most of the input the system receives comes from the reaction of the users to the recommendations. This cyclic perspective has been named interactive recommendation, and has recently attracted the interest of the community (Gentile et al., 2014, Li et al., 2010, 2016, Zhao et al., 2013), linking to the machine learning paradigm known as *reinforcement learning* (Sutton and Barto, 2018). As this represents a realistic take on recommendation, understanding its mechanisms and developing novel approaches for the task is the third and final perspective we study in this thesis.

1.2 Research goals

The general aim of the present thesis is to provide a thorough understanding of the contact recommendation task and its effects on the evolution of social networks in terms of its density (accuracy) and diversity. We focus our study on the following research goals.

- **RG1: Review and compare previous approximations to the contact recommendation task.** Literature on people recommendation and link prediction is social networks is very extensive. However, as far as we are concerned, a comparison between a wide variety of approaches under a unified top-N recommendation setting is still missing. Therefore, we aim to conduct experiments under a unified setting, and provide useful information about what makes a recommendation algorithm good (or bad) for the task.
- **RG2: Explore the adaptation of text information retrieval models to the contact recommendation task.** Following this research goal, our research seeks to make contributions at two different levels: first, at a theoretical level, we aim to provide a better understanding of the contact recommendation task, through the development of connections between the information retrieval and recommendation concepts and techniques; second, at a practical level, we seek to open new ways for the development of effective contact recommendation approaches, by importing information retrieval techniques.
- **RG3: Study the effect of contact recommendation on the evolution of social networks.** As a first step, we aim to understand how contact recommendation approaches modify the structural properties of networks (such as distances between users, redundancy of the links, degrees). Then, we want to determine the potential benefits these changes have for the users in the social network. In our work, we focus on the effect that structural diversity produced by recommended links may have on the diversity of the information flow in the network and the mitigation of filter bubbles.
- **RG4: Develop new interactive recommendation approaches based on reinforcement learning.** We want to explore the adaptation of reinforcement learning to create new effective approaches for cyclic recommendation. For this, we particularly focus on a family of reinforcement learning techniques known as multi-armed bandits.

1.3 Contributions

The research conducted in this thesis has resulted in several contributions, which we summarize next.

- A framework for adapting text information retrieval principles and approaches to the contact recommendation task. This framework is based on the relation between the fundamental spaces in the search task (queries, documents and terms) and the unique space in collaborative filtering for contact recommendation: the users in the network. This framework allows the construction of new people recommendation methods as well as experimental procedures for evaluating them.
- A collection of effective contact recommendation algorithms based on information retrieval (IR) techniques. We differentiate three groups of methods: a) direct adaptations of IR models, b) k nearest neighbors (collaborative filtering) approaches using IR models to select the neighbors of the target and candidate users, and c) learning to rank methods.
- Several principles for the theoretical analysis of friends of friends contact recommendation analysis, based on the fundamental properties (axioms) behind the design of information retrieval models.
- A set of metrics, adapted from social network analysis, that allow us to quantitatively measure how the structure of social networks changes due to the action of contact recommendation algorithms in relevant ways. We first study the effect in reducing distances between user pairs in the network, and, second the presence of weak ties in the structure, understood as a) non-redundant connections and b) links between different communities of users.
- A global greedy algorithm for enhancing properties of the whole system (as structural properties in social networks) by reordering the initial outcome of a recommendation method. This strategy is designed to jointly optimize the value of the global property and the accuracy of the new recommendation rankings. When reordering an individual recommendation, our strategy does not only consider the suggestions made to its recipient, but also the recommendations for the rest of the users in the system.
- Empirical evidence of the effect that recommending weak ties has on information diffusion in networks. In particular, we find that enhancing the structural diversity of the network has benefits on the novelty and diversity of the information that arrives to the people in the network.

- An interactive recommendation approach that uses a multi-armed bandit strategy known as Thompson sampling to select the best neighbors that represent the tastes of the target user. The resulting algorithm is a stochastic reformulation of the user-based memory-based collaborative filtering algorithm that deals with the uncertainty of the data when selecting the best neighbors.
- Two novel social network datasets, downloaded from the Twitter social network, that allow us to evaluate contact recommendation approaches. Each sample contains two different directed networks: an explicit (static) network of stable following relations between users, and an implicit (dynamic) network, built from the interactions between users (mentions and retweets) on Twitter.

1.4 Structure of the thesis

We summarize next the structure of this document. The thesis is divided in four different parts and 11 chapters:

- **Chapter 1 (Introduction)** presents the motivation for the realization of this thesis, along with the main goals of our research, our main contributions, the structure of the document and the publications related to the thesis.
- **Part I – Preliminaries:** The first content block of the thesis reviews the basic notions and previous work in the different fields we explore through the thesis. It is divided in three chapters:
  - *Chapter 2 (Related work)* explores the background of this thesis and summarizes the basic concepts of the fields related to our research: recommender systems, information retrieval, social network analysis and multi-armed bandits.
  - *Chapter 3 (Contact recommendation)* focuses on the contact recommendation problem and summarizes its algorithmic solutions and challenges.
  - *Chapter 4 (Preliminary experiments)* describes the methodology and data we use through the thesis, as well as an initial experiment comparing state of the art contact recommendation methods.
- **Part II – Information retrieval models for contact recommendation:** The second block explores the adaptation of information retrieval algorithms and techniques for recommending contacts in social networks. It is divided in three chapters.
  - *Chapter 5 (Adaptation of IR models)* introduces the basic framework for applying information retrieval techniques to contact recommendation and explores the use of classical models as contact recommendation algorithms.
  - *Chapter 6 (Axiomatic analysis)* studies the properties that make IR models effective approaches for recommending people from an IR perspective.
  - *Chapter 7 (Advanced adaptations of IR models)* shows two additional uses of IR models in contact recommendation: as similarities in a kNN collaborative filtering scheme and as features in learning to rank approaches.
- **Part III – Novelty and diversity in contact recommendation:** The third block explores additional dimensions for the evaluation and design of contact recommendation approaches, complementing their accuracy.
  - *Chapter 8 (Impact on network structure)* researches the definition of a) metrics for contact recommendation and link prediction based on concepts of novelty and diversity in recommender systems and IR and b) metrics targeting the effects of contact recommendation in the topological properties of networks.
  - *Chapter 9 (Effects on information diffusion)* analyzes the effects that contact recommendation approaches have on the spread of information on social networks, focusing on the correlation between enhancing the structural diversity and the novelty and diversity of the information reaching the users.
- **Part IV – Interactive recommendation:** The fourth block explores the recommendation task from a cyclic perspective, where the system changes over time.

  - *Chapter 10 (Multi-armed bandits in recommendation)* investigates the use of multi-armed bandits as collaborative filtering approaches, and proposes a novel neighbor-based approach for addressing the exploration-exploitation problem in the recommendation cycle.

- **Chapter 11 (Conclusions)** summarizes the findings in this thesis, and proposes several research lines for the continuation of this work.

In addition to the main contents of this thesis, we include several appendices with additional information complementing our main findings:

- **Appendix A (Introducción)** shows the Spanish translation of Chapter 1.
- **Appendix B (Conclusiones y trabajo futuro)** shows the Spanish translation of Chapter 11.
- **Appendix C (Parameter selection)** shows the optimal hyperparameter selection for the different experiments conducted in this thesis.
- **Appendix D (Statistical significance)** reports statistical significance tests for the different accuracy comparisons in the thesis.
- **Appendix E (Specialized global reranking)** provides further details about the implementation of the reranking strategies first described in Chapter 9.

1.5 Publications related to the thesis

The research carried out in this thesis has given rise to a several publications in workshops, conferences, journals and books related to the Information Retrieval area (Sanz-Cruzado and Castells, 2018a, 2019a,b, Sanz-Cruzado et al., 2018a,b, 2019, 2020a,b) that we summarize here.

Publications related to Part I

In the following publication, we provide an overview of the contact recommendation task, including a comprehensive list of algorithms and practical challenges. We also include a comparative of representative state of the art algorithms in terms of accuracy, novelty and diversity metrics.

- **Sanz-Cruzado and Castells (2018a)**: Javier Sanz-Cruzado and Pablo Castells. Contact Recommendations in Social Networks. In Shlomo Berkovsky, Iván Cantador and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 519-569. World Scientific Publishing, Singapore, 2018.

Publications related to Part II

The following contributions explore the adaptation of information retrieval models and techniques to contact recommendation in social networks. In Sanz-Cruzado and Castells (2019a), Sanz-Cruzado et al. (2018a, 2020a) we adapt the basic IR models for suggesting people in networks, as we explain in Chapter 5. We analyze the effectiveness of such models in terms of an axiomatic analysis in Sanz-Cruzado et al. (2020b). Finally, the remaining contents of Sanz-Cruzado et al. (2020a) explore the use of IR models as similarities in a nearest neighbors scheme and as features in learning to rank approaches.

- **Sanz-Cruzado et al. (2018a):** Javier Sanz-Cruzado, Sofía M. Pepa and Pablo Castells. Recommending Contacts in Social Networks Using Information Retrieval Models. In *Proceedings of the 5th Spanish Conference on Information Retrieval (CERI 2018)*, Zaragoza, Spain, June 2018. ACM.
- **Sanz-Cruzado and Castells (2019a):** Javier Sanz-Cruzado and Pablo Castells. Information Retrieval Models for Contact Recommendation in Social Networks. In *Proceedings of the 41st European Conference on Information Retrieval (ECIR 2019)*, number 11437 in LNCS, pages 148–163, Cologne, Germany, April 2019. Springer.

- **Sanz-Cruzado et al. (2020a):** Javier Sanz-Cruzado, Pablo Castells, Craig Macdonald and Iadh Ounis. Effective Contact Recommendation in Social Networks by Adaptation of Information Retrieval Models. *Information Processing and Management*, 57(5),Article 102285, September 2020.
- **Sanz-Cruzado et al. (2020b):** Javier Sanz-Cruzado, Craig Macdonald, Iadh Ounis and Pablo Castells. Axiomatic Analysis of Contact Recommendation Methods in Social Networks: an IR perspective. In *Proceedings of the 42nd European Conference on Information Retrieval (ECIR 2020)*, number 12035 in LNCS, pages 175–190, Online, April 2020. Springer.

Publications related to Part III

The subsequent papers analyze contact recommendation properties beyond the accuracy. Sanz-Cruzado and Castells (2019b), Sanz-Cruzado et al. (2018b) introduce several novelty, diversity and structural metrics and provide a first overview by comparing their values for a selection of contact recommendation algorithms, similarly to how we do in Chapter 8. In (Sanz-Cruzado and Castells, 2018b) we explore the effect that recommending links for enhancing structural diversity metrics has on the properties of information diffusion.

- **Sanz-Cruzado et al. (2018b):** Javier Sanz-Cruzado, Sofía M. Pepa and Pablo Castells. Structural Novelty and Diversity in Link Prediction. In *Proceedings of the 9th International Workshop on Modelling Social Media (MSM 2018), in Companion of the The Web Conference 2018 (WWW 2018)*, pages 1347–1351, Lyon, France, April 2018. IW3C2.
- **Sanz-Cruzado and Castells (2018b):** Javier Sanz-Cruzado and Pablo Castells. Enhancing structural diversity in social networks by recommending weak ties. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018)*, pages 233–241, Vancouver, Canada, October 2018. ACM.
- **Sanz-Cruzado and Castells (2019b):** Javier Sanz-Cruzado and Pablo Castells. Beyond Accuracy in Link Prediction. In *Proceedings of the 3rd Workshop on Social Media Personalization and Search (SoMePeaS 2019) co-located with the 41st European Conference on Information Retrieval (ECIR 2019)*, number 1245 in Communications in Computer and Information Science, pages 79–94, Cologne, Germany, April 2019. Springer.

Publications related to Part IV

The following paper (Sanz-Cruzado et al., 2019) focuses oninteractive recommendation. Specifically, we propose a novel approach combining nearest-neighbor collaborative filtering and multi-armed bandits.

- **Sanz-Cruzado et al. (2019)**: Javier Sanz-Cruzado, Pablo Castells and Esther López. A Simple Multi-armed Nearest-neighbor Bandit for Interactive Recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, pages 358–362, Copenhagen, Denmark, September 2019. ACM.

Part I

Preliminaries

"If I have seen further, it is by standing on the shoulders of giants."
Isaac Newton

Contact recommendation is a multi-disciplinary task, lying on the intersection between different fields, disciplines and problems. Consequently, the research we present in this thesis has ties to many of them, and having a basic comprehension of them appears as fundamental for understanding it. This part is intended to provide an introduction to such fields and contextualize our research.

In Chapter 2, we start by introducing the reader to the recommender systems field. We describe the recommendation task and provide a review of the different families of algorithms and their strengths and weaknesses. We then analyze the problem of evaluating these approaches, with a special focus on three properties beyond accuracy: scalability, novelty and diversity. Afterwards, we shift our focus to text search in information retrieval: we examine how it works, its relation to the recommendation problem and two research lines particularly interesting for our research: axiomatic thinking and learning to rank.

Afterwards, we analyze the possibilities that social network analysis offers for the study of social networks, such as structural properties, community detection or information diffusion. Finally, we introduce the reader to reinforcement learning, with a special interest on multi-armed bandits for the recommendation task.

In Chapter 3, we focus on the particular task of recommending contacts in social networks, providing a comprehensive review of algorithms, relations to other fields and current practical challenges which have to be considered in the development of such approaches. Finally, in Chapter 4 we introduce the setup of the experiments we carry through the rest of the thesis (describing such aspects as data collection and treatment, and hyperparameter selection) and we provide an initial comparison of several state of the art contact recommendation approaches in terms of accuracy.

2
Related work

The research we carry throughout this thesis has connections with multiple fields and disciplines. In this chapter, we examine those fields, establishing initial relations between them and our research. We introduce four different (although related) fields: first, as a basic ground concept for the thesis, we define the recommendation task as well as several of its challenges; next, we explain the fundamentals of textual information retrieval and its relation to recommender systems, in addition to two particular research lines in this field: axiomatic thinking and learning to rank; afterwards, we present the basic notions and concepts from graph theory and social network analysis that we will need in the rest of the thesis; and last, we provide an introduction to reinforcement learning with a special focus on multi-armed bandits and their application in the design of recommender systems. As this chapter is only intended to act as an introduction to these areas, we provide just so many details as needed about algorithms, metrics and techniques.Details on specific aspects will be expanded when needed in the following chapters.

2.1 Recommender systems

Traditionally, individuals have relied on several sources of new information or opportunities they might find useful or be interested in. A first source of discovery is word of mouth: people very often rely on close relatives or friends to get suggestions about books, movies or even people to befriend, date, or solve a problem (e.g. doctors, plumbers, etc.). Marketing campaigns in television, magazines, bus stops, etc. are another important but less personal means of discovery. Furthermore, newspapers and specialized magazines include reviews, written by experts on different matters. All the previous examples illustrate different cases of recommendation.

Recommender systems appear as an alternative to these long-established sources, as software tools that automatically suggest their users some items or people they might be interested in (Adomavicius and Tuzhilin, 2005, Berkovsky et al., 2018, Ricci et al., 2015). Their conception, research and development dates back to the 1990s (Goldberg et al., 1992, Resnick and Varian, 1997), where they were mainly oriented to e-commerce. Since then, recommendation technologies have largely expanded and grown, and they are now available in multiple domains, such as online shopping (Amazon, eBay, Walmart), news (Google News), academic publications (Mendeley), music and video streaming (Netflix, Spotify, Youtube), app stores (Google Play, Steam) and social networks (Twitter, Facebook, LinkedIn). The development of recommender systems is often multidisciplinary, involving different fields such as machine learning, information retrieval, human computer interaction, statistics or marketing.

The general functioning of recommender systems is the following: given a target user u (the receiver of the recommendation), the system observes her history, i.e. the set of interactions the user has had with items in the system. These interactions can be explicit (e.g. the five star convention in Amazon, Booking or Google Play, or the binary like/dislike feedback in platforms like Youtube, Spotify or Facebook), or implicit (the number of times the user interacts with – plays, clicks, buys – a certain item or the total amount of time the user has spent consuming some product). This user-item feedback can be represented as a matrix, where there are as many rows and columns as users and items, respectively. When a user interacts with an item, a numerical value is assigned to the corresponding cell – an abstraction (a simplification) of the data describing such interaction, which we refer to as a rating. This matrix is illustrated in Figure 2.1. Since users generally only interact with a small fraction of the items, this matrix is usually highly sparse: most of its cells are empty.

Taking the rating matrix as starting point, recommender systems identify the items which have not been interacted (yet) by the user, which we denote as the set of candidate items. Then, for each item in that set, the system generates a recommendation score $f_u(i)$ representing how important the system believes that the item i may be for the target user u, and then ranks the items in decreasing score order. We illustrate this procedure

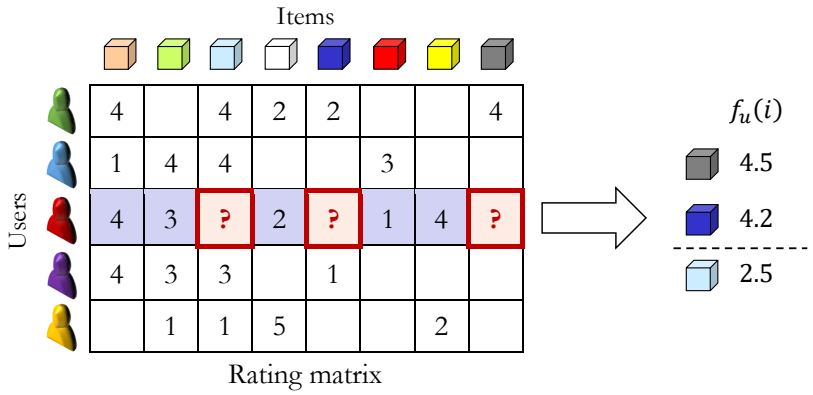


Figure 2.1: The recommendation task.

in Figure 2.1, where we want to recommend items to the red user. As the pool of candidate items in the system is usually very large (for example, Spotify has, as of May 2020, more than 60 million songs and almost 2 million podcasts[1]), users would be overwhelmed if they were shown all the possible items. Therefore, pruning the ranking and only showing a reduced number of items is a common practice in the recommender systems field (Cremonesi et al., 2010). In Figure 2.1, this is illustrated by a dotted line separating the items the user sees from the ones that the system hides from her.

Beyond the interactions between the users and the items, it is possible for recommender systems to exploit many other information sources. Items usually have some characteristics or features that describe them and can help match items to user tastes, like genres for book or movies or music composers in the case of songs. On some applications, users also provide some demographic information(such as their age, gender, location, job, etc.) which can be exploited by recommenders. More recently, other elements such as the connections between users in the system have been exploited for the recommendation. The use of these additional data sources is further discussed in the following section.

2.1.1 Algorithms

Since their conception in the earlier 1990s (Goldberg et al., 1992), the methods which have been developed for the delivery of relevant suggestions to users are counted by hundreds or even thousands. Selecting an appropriate algorithmic solution, according to the specific characteristics of each system, is fundamental for the success of the recommendations. As an initial way to determine which approaches might be more interesting for the systems we want to develop, we group them according to different criteria, such as the elements we want to recommend or what kind of data we use for generating the recommendations. In Figure 2.2, we provide three different taxonomies for recommender systems, which we describe next.

A first criteria divides recommendation algorithms according to whether their outcome depend or not on the target user receiving the recommendation. If the algorithm provides specific recommendations for each user according to their specific needs or tastes, we talk about *personalized* recommendation. Algorithms like matrix factorization (Hu et al., 2008, Koren et al., 2009) are examples of personalized recommenders. On the other hand, *not personalized* methods do not differentiate between the different users in the platform. Expert opinions, as well as recommenders that use the global set of ratings in the system to suggest the most popular items (Cremonesi et al., 2010) are examples of not personalized approaches.

A second criteria relies on the nature of the items to be recommended. We talk about *domain independent recommendation* when the recommendation algorithms can be applied to multiple domains. Apart from that, we can talk about news recommendation (Li et al., 2010), music recommendation (Jannach et al., 2018), venue recommendation (Wang et al., 2019b), movie recommendation (Sefati et al., 2018) or, more important for this thesis, people recommendation (Guy, 2015, 2018, Pizzato et al., 2010, Sanz-Cruzado and Castells, 2018a, Terveen and McDonald, 2005).

[1]Spotify statistics: https://newsroom.spotify.com/company-info/ (Accessed 19th December 2020)

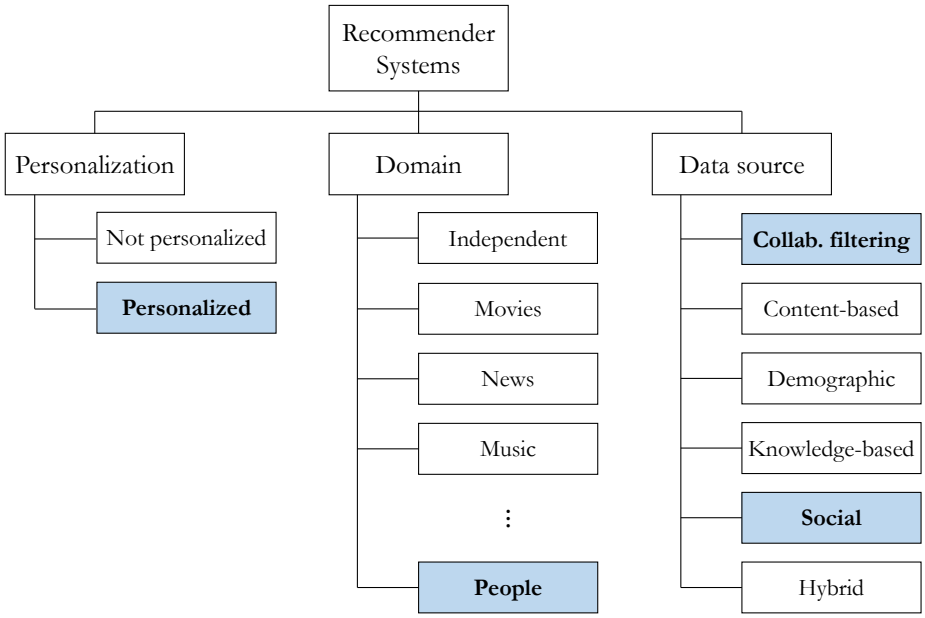


Figure 2.2: Recommender systems taxonomy. Blue boxes with bold text indicate the categories explored in this thesis.

Finally, the third (and most established) criteria to specify a classification of the recommenders depends on the nature of the data the algorithms take as input: just the ratings, item features, social information, etc. Using the classification provided by Ricci et al. (2015), we can differentiate six groups of approaches: collaborative filtering, content-based, demographic, knowledge-based, social and hybrid recommenders. We next provide a detailed description of these families, along with their advantages and limitations, which are summarized in Table 2.1.

Collaborative filtering

Collaborative filtering (Goldberg et al., 1992) approaches use interactions between users and items as their input data. No further information is used to predict the interest of the target user on the different items in the system. Their principal idea is that the ratings provided by other users help the target user to discover new items (i.e. other users unknowingly collaborate to help her discover new items) based on her personal preferences: let's suppose that a user, Susan, has enjoyed watching "Titanic" and "The Lion King", and has registered positive ratings for them in the system. Three other users in the system like both films, and the three of them also like "The Godfather". As their tastes appear to be similar to Susan ones, it seems likely that Susan might also enjoy "The Godfather".

Depending on how the ratings are used to generate the recommendations, collaborative filtering can be divided in two different categories: memory-based and model-based algorithms (Adomavicius and Tuzhilin, 2005):

Memory-based approaches, also known as neighborhood-based, straightforwardly use the ratings in the user-item matrix to generate the recommendations (Ning et al., 2015). They are based on the notion of neighborhood: these algorithms limit the set of interactions they consider by selecting a subset of the users (or items) which have similar ratings to the target user (or the candidate items), and use their preferences to determine which items the target might like.

Model-based approaches, on the other hand, learn the behaviour of the user in the system from its ratings (they create a model representing that behaviour) and use the acquired knowledge to recommend items. A particularly well-known and effective subset of these methods is the so-called matrix factorization family (Koren et al., 2009). These recommenders learn an approximate representation of the rating matrix as the product of two matrices: a user matrix and an item matrix. The columns of such matrices represent (respectively) users and items in a low dimension latent space. For the items, these factors might vaguely correspond to properties (for example, difficulty or age target of a video game) – but it is rarely possible to give them a meaning –, and,

Table 2.1: Advantages and disadvantages of recommender systems.

Category	Advantages	Limitations
Collaborative filtering	• No need for side-information (only ratings) • Domain independent	• User cold start • Item cold start • Data sparsity
Content-based	• Works with sparse matrix. • No item cold start.	• User cold start. • Overspecialization. • Difficult to retrieve relevant information about items
Demographic	• No user cold start	• Item cold start • Privacy and security concerns
Knowledge-based	• No need for ratings • No need to create user profiles	• Need for explicit user needs • Difficult to retrieve relevant information about items
Social	• Reduces user cold start • Greater coverage	• Noisy connections • Users with few connections • Network sparsity
Hybrid	• See the mixed approaches	• See the mixed approaches

for the users, they represent how interested the user is in those features. Lately, model-based approaches have also included some algorithms based on deep neural networks (He et al., 2017).

Advantages and limitations

As some of the most popular and effective recommendation approaches, collaborative filtering methods show a series of advantages, which we list next:

- **Domain-independent:** As these methods only use the (explicit or implicit) interactions between users and items in the system to generate the recommendation scores, they can be applied to different recommendation domains with little to no modification. Their effectiveness over different domains, is, however, not ensured.
- **Novelty and diversity:** Since the recommendations are not only inferred from the target user preferences, but other users participate in the process, it is possible for collaborative filtering approaches to recommend items outside the area of knowledge of the target user (Burke, 2007).

However, they also have drawbacks and limitations (Adomavicius and Tuzhilin, 2005):

- **Cold start:** This problem takes its name from the automobile sector. When a car engine is started in cold weather, the oil is cold, making more difficult for the engine to ignite . As the engine starts running, the temperature of both the engine and the oil increases, making it easier for the engine to burn the oil and reach higher revolutions (therefore, allowing the vehicle to run more easily).

  Analogously, something similar occurs in recommender systems: when the system receives a new user or a new item, they are "cold". In the case of a new user, recommenders do not have enough information about her tastes until she has provided some ratings to the system – thus making difficult to provide her good recommendations. In the case of the items, they are invisible to the system – and, therefore, cannot be recommended. We refer to the user case as *user cold start* and to the item one as *item cold start*.

  In the case of the users, as they start providing more and more ratings to items in the system, then, it becomes easier for the system to infer their preferences and provide relevant recommendations – the user becomes "warmer". In the case of items, becoming warmer implies receiving multiple ratings. Once this happens, the visibility of the item increases, and the system can recommend it to users who might be interested on it.

  Collaborative filtering methods are strictly based on the ratings provided by the users, so both cold start variants represent a serious problem for them. In order to prevent (or at least mitigate) this, it is possible to

apply some active learning techniques (Elahi et al., 2016). For instance, Twitter asks users to follow several popular users to determine their tastes during the registration process. Many other applications have similar mechanisms.

- **Data sparsity:** Another important practical concern for collaborative filtering recommender systems is the sparsity of the rating matrix. Companies like Netflix, Amazon and Spotify are used by millions of users daily, and are constantly adding new items to their systems, making impossible for the users to consume all of them in their lifetime. From the vast amount of items in the system, only a tiny fraction of them is typically consumed by each user.

  In order to provide an example of the scale of this, we use information provided by Spotify[2]: as of September 2020, Spotify had 320 million monthly active users and above 60 million songs. Supposing that a user spends 4 hours a day listening to music on that platform anda song is around 3 minutes long it would take him 750,000 days (more than 2,000 years!) to listen to all the content available in Spotify by that date, without ever repeating a song. Therefore, it is easy to see that the rating matrix for this platform must be very sparse.

  The sparsity of the rating matrix is a challenge for collaborative filtering, since the algorithm might not have enough data to provide reliable recommendations to the users. For example, in memory-based methods, there might not be enough overlapping ratings between users to find appropriate neighbors for the target user.

Content-based recommendation

Algorithms within this category consider that users are prone to enjoying similar items to those they consumed (and liked) in the past (de Gemmis et al., 2015). For example, if a user has provided positive ratings to all "Spiderman" and "X-men" films, he might enjoy other comic-book based American super-hero movies such as "Superman","The Incredibles" or "The Avengers". These approaches exploit the similarities between items, but, instead of using ratings, they take specific properties and features of the items to compute such similarities. For example, a movie might be represented by information about its genre, actors or user-generated tags. Content-based approaches compute their scores by comparing the candidate items with those items that the target user has interacted with. For example, using the ratings in the system, some content-based methods create a profile for each user indicating how interested the user is on each featureand, then, the recommendation scores are computed by measuring how similar are the user and item profiles (Adomavicius and Tuzhilin, 2005). Content-based approaches take inspiration in disciplines like information retrieval (Adomavicius and Tuzhilin, 2005, de Gemmis et al., 2015) – with methods as Rocchio (1971) or the vector space model (Salton et al., 1975) – or machine learning, with nearest-neighbors or Naive-Bayes (Bishop, 2006).

Advantages and limitations

Personalized content-based recommenders have several advantages and disadvantages (de Gemmis et al., 2015). We first list their advantages:

- **No item cold start:** In content-based recommendation, ratings are only used for determining the importance the rated items have for the users. Then, candidate items are recommended according to their similarity to those rated elements. Therefore, if a new item arrives in the system, as long as it has feature data, it might be recommended. Thus, these methods do not suffer from the item cold start problem.
- **Work on sparse matrices:** For the same reason as above, once we have enough ratings to determine the tastes of the users, we can exploit the item features to provide recommendations – making the recommender not to depend so heavily as collaborative filtering approaches do on the amount of ratings in the system matrix.

Beyond those advantages, these systems do still suffer from some limitations (Adomavicius and Tuzhilin, 2005):

- **User cold start:** When the system receives a new user to provide recommendations to, the system does not have any information about the features that the user might be interested in. Therefore, it is not possible to build the user profile and generate useful recommendations.

[2]Spotify statistics: https://newsroom.spotify.com/company-info/ (Accessed 19th December 2020)

- **Limited content analysis:** These approaches are limited to the set of features associated to the items. This fact causes several problems.

  First, if there are not items in the system matching the features in the user profile, the system cannot provide recommendations. These might occur when the appearance of some properties in the system is scarce.

  In addition to this, the set of features of the items has to be either manually introduced (which might not be feasible in platforms with thousands to millions of items) or by using automatic feature extraction techniques (such as information retrieval in the case of text). However, depending on the domain, this might also be difficult.

  Finally, this limitation to the known features might cause yet another problem: two items sharing the same exact features are indistinguishable by the recommender, and both of them would be given the same recommendation score (Adomavicius and Tuzhilin, 2005). For example, if two movies (one good and one bad movie) are identified by a similar description, the system is not able to tell which one is the good one without further information.

- **Overspecialization:** Content-based approaches always recommends similar items to the ones that the user already knows. Because of this, items with features that the user does not know cannot be recommended. For instance, considering a target user who has only expressed her interest on super-hero movies a content-based recommender would continuously recommend this kind of movies. Unless the user discovers other genres such as romantic comedies or terror movies on her own, or the recommendation algorithms finds some mixed genre film, the recommendation approaches would not be able to recommend movies outside this category. Therefore, content-based approaches do not encourage the discovery of new surprising elements, thus limiting the utility of the recommenders.

  Lack of diversity in recommender systems, as occurs with the overspecialization in content-based approaches, might lead to the creation of filter bubbles (Pariser, 2011). The filter bubble phenomenon occurs when, due to the personalization of recommendation or search algorithms, the user is only receiving information according to his tastes or ideology, becoming isolated from other points of view.

Demographic recommendation

Demographic algorithms represents the counterpart to content-based approaches: as the latter exploit information about items, demographic recommenders use information about users in the system – such as their age, location, occupation, income, sex, etc. – instead to produce relevant recommendations. They are based on the idea that users with a similar demographic condition might have similar influences and therefore enjoy similar items. For example, university students specializing on Scottish history might be all interested in movies and TV series set on different moments of the history of Scotland, such as "Braveheart", "Outlander" or "Mary, Queen of Scots".

An example of demographic recommendation is the approach by Al-Shamri (2016), who proposes an algorithm based on memory-based collaborative filtering where the neighbors of the target user, instead of being computed in terms of the ratings provided by him, are computed considering user profiles built with demographic information.

Advantages and limitations

Demographic recommendation approaches have the following set of advantages:

- **No user cold start:** Analogously to how content-based recommenders eliminate the item cold start problem using item features, demographic recommendation eliminates the user cold start problem, since they use their demographic profile to determine their preferences (and not their ratings).
- **Novelty and diversity:** Similarly to collaborative filtering, in this family of algorithms, recommendations are produced according to other people's ratings, thus allowing these methods to recommend novel items to the target users (and may not suffer so much from the overspecialization problem).

However, they have problems that limit the utility of the approaches:

- **Item cold start:** In order to recommend items to users, it is necessary that users with similar demographic properties have previously rated the items. Therefore, similarly to collaborative filtering, new items with zero ratings are invisible to the system.
- **Too general recommendations:** If the number of features to describe a user is limited, the personalization of these approaches would be very coarse: for instance, in Madrid, Spain, there are around 230,000 university students. If we only use occupation and location as demographic properties, the personalization of the demographic methods would be rather low as a consequence of the huge size of the group.
- **Privacy and security concerns:** Demographic recommendation algorithms rely on data that users provide among themselves about their age, location, income, etc. However, users might be reluctant to give to the system such information, as it might be wrongfully used to obtain information about themselves which they are not willing to provide. Therefore, acquiring the demographic information about the users might be a challenging task.

Knowledge-based recommendation

Knowledge-based methods take explicit user requirements and exploit them to generate recommendations using information about the underlying domain of the recommendation (Burke, 2000, Ricci et al., 2015). For example, if we wanted to find a restaurant which is similar to our favorite one, a knowledge-based recommender would use information we introduce about the cuisine, the average price or the location to provide recommendations. Or, if we want to buy a computer, we might specify some features about it like the speed of the processor or the amount of random-access memory to receive a list of recommended computers which satisfy our expectations.

Differently from other recommendation techniques, these systems do not depend on the ratings provided by their users, but on a set of explicit specifications introduced by the user about what they are looking for (Burke, 2000). Depending on how these explicit user needs are matched with the features of the items to provide the recommendations, we can differentiate two types of knowledge-based approaches: case-based and constraint-based algorithms. *Case-based methods* use the available information to recommend to the users those items that have the most similar items to the provided specifications (Lorenzi and Ricci, 2003) whereas *constraint-based methods* use an underlying knowledge base containing specific rules for matching the needs of the users with the features of the candidate items in order to provide a valuable solution for the user (Felfernig and Burke, 2008).

Advantages and limitations

Knowledge-based approaches have several advantages, mainly based on the fact that they do not depend on the ratings provided by the users, just on the explicit needs of the user:

- **No cold start:** Since these systems do not need ratings, new users might be served recommendations as long as they introduce their needs into the system, and new items might be recommended as they have the proper features.
- **No user profiles:** Differently from the other families, these methods do not require storing any information about the users, making them more memory efficient than other methods such as content-based approaches.

However, there are concerns related to the information that the system has about the different items:

- **Maintenance of the knowledge-base:** These methods need high quality information about the different features of the items to work properly. This information is not always easy to retrieve and maintain (Lorenzi and Ricci, 2003), similarly to how it is difficult to obtain features for content-based recommendation.
- **Need for explicit queries:** As these recommenders require an active query from the user to generate the recommendations, this might lead to several problems. For instance, if the user does not introduce enough requirements, their output might be noisy. However, if the user introduces too much information or contradictory requirements, there might be no items for him.

Social recommendation

Also known as community-based recommenders (Ricci et al., 2015), social recommendation algorithms exploit the information about relations between different users in the system to suggest new items to consume (Guy, 2015, Tang et al., 2013). Using an (underlying) social network, these approaches automatize the fact that people usually rely on close friends to provide good recommendations (Sinha and Swearingen, 2001). We can motivate social recommendation with an example: let's suppose that Peter is looking for a new book. Then, he meets his friends, and one of them mentions that he has enjoyed "The Three Body Problem", and recommends the book to him. Then, other friends in the group mention that they have also liked the novel. In the end, as many of his friends have enjoyed it, then, Peter decides that he might give an opportunity to the book. The idea behind social recommenders is to model this by using the connections between users in a social network.

These family of recommenders has taken advantage of the fast growth and development that social networks have experienced since the beginning of the 20th century. It is based on two phenomena: social influence (Anagnostopoulos et al., 2008), that establishes that the actions of one individual might cause his friends to perform the same actions (for example, if the user reads "The Three-Body Problem", his friends might become curious about the book and also read it), and the homophily principle (McPherson et al., 2001), which states that people tend to connect to other people with similar characteristics – people with similar beliefs, jobs, people within the same communities, or, more important for recommender systems, people with the same tastes.

Most social recommendation approaches are based on basic collaborative filtering ones (Tang et al., 2013). One of the first works in social recommendation, the TidalTrust algorithm proposed by Golbeck (2006) adapts neighborhood-based collaborative filtering, but, instead of selecting users according to how similar their ratings are, finds the neighborhoods depending on trust values given by the users to their friends in a social network. Konstas et al. (2009) proposed something similar, but taking random walks centered on the target users to select the neighbors. Ma et al. (2008) proposed the SoRec method, an approach based on probabilistic matrix factorization (Salakhutdinov and Mnih, 2007) that computes recommendations based on co-factorizing the user-item rating matrix and the social network adjacency matrix. Jamali and Ester (2010) estimate the trust between pairs of users according to their connections in a social network, and integrates that estimations in a matrix factorization scheme. Finally, in the last few years, social recommendation methods based on deep neural networks have started to emerge, like GraphRec (Fan et al., 2019).

Advantages and limitations

Similarly to other categories, we can identify several advantages and disadvantages of social recommendation (Tang et al., 2013). The main advantages are achieved thanks to the social influence phenomenon: social recommenders can exploit propagation mechanics through the network for (a) reducing the number of cold start users in the network and (b) increase the coverage of the recommendation, with relation to other families as collaborative filtering. Beyond those advantages, there are some limitations and concerns about the use of social connections to increase the effectiveness of the recommendation:

- **Noisy connections:** In social networks such as Twitter or Facebook, the cost of creating a new connection to other users is very small. Although in those platforms, that number is somehow limited, it still leads the users to connect with multiple kinds of people, ranging from family and good friends to work mates or even old school friends with whom they do not talk anymore. In general, not all the connections are valid for generating good recommendations – indeed, some of the people in the network might even provide negative signals —, but filtering the useful ones is not a simple task.
- **Users with few or no connections:** Beyond having many connections, the opposite problem might also happen: there might be users in the network without connections (or with only one or two connections). For these users, the methods do not have enough information to compute relevant recommendations.
- **Network sparsity and degree distribution:** The previous problems might be magnified due to the structure of real networks. Real networks are typically very sparse, and the distribution of the number of neighbors is typically similar to a power-law distribution, where few users concentrate most of the connections and most users have only a few friends in the network (Newman, 2018).

Hybrid recommendation

Hybrid recommenders are devised as means to overcome the limitations of the previous families of recommender systems by combining methods or techniques that have as input different sources of data (such as ratings, item features, social connections, etc.). To set up an example, collaborative filtering approaches suffer from the cold start problem when they receive new items and new users. If we manage to use information about the candidate item features as in content-based methods, we might be able to recommend items which have just arrived to the platform (thus solving the new item problem).

According to how different data sources are combined, we can establish seven different groups of hybrid recommenders (Burke, 2007). We should note that these techniques might also be used to combine different methods/techniques that take as input the same data source (for example, it is possible to create ensembles of collaborative filtering approaches). However, we only consider that a recommendation algorithm is hybrid if it combines different sources. We provide next a brief description of each hybridization technique:

- **Weighted:** These approaches take different recommendation approaches as components of the hybrid recommender. Then, each component provides scores for the different candidate items. Then, given an item, the recommender computes the definitive score by combining the weights in a static manner (for example, by a linear weighted combination).
- **Mixed:** In this case, we take again different recommendation methods as components. Then, each recommender provides its own recommendation ranking, and all the rankings are then combined into a unique ranking, or shown side-by-side.
- **Switching:** This group of recommenders takes again different recommendation algorithms as components, but, instead of combining them, depending on the context, it chooses one among them to provide the recommendation (for example, based on a confidence value on the approach).
- **Feature combination:** A unique algorithm receives input data from different sources. The algorithm integrates those features to generate a recommendation.
- **Feature augmentation:** In this case, we have a recommendation algorithm that receives information features from one data source. As we want to add new data sources as inputs, we use other recommenders which take those source as inputs, take their outcomes, and use them as another feature for our recommender.
- **Cascade:** These hybrids establish a hierarchical relationship between different recommenders, so the recommendations made by a recommender are just refined by others.
- **Meta-level:** A meta-level hybrid approach is a recommender that takes as input the model learned by other method.

In this thesis, we focus on personalized approaches for contact recommendation. Contact recommendation appears as a special case of social recommendation where we recommend people to other people. As we shall see in the following chapter, in this special case, collaborative filtering and pure social recommendation are nearly indistinguishable, and the set of items is just extracted from the set of users of the social platform. We highlight the related categories in Figure 2.2.

2.1.2 Evaluation

The development of recommender systems is tied to their utility, to their ability to satisfy some goals. For example, we might consider a recommender system for an e-commerce platform. From the point of view of the owners of the business, it is intended to increase the revenue of the platform. Or, for the customers, the system is only useful if the recommended items are of their interest. Deploying different algorithms might lead to important differences on the performance of the system on satisfying such goals and many others. Therefore, it is essential to measure and determine the utility and quality of the different algorithms, or, in other words, it is necessary to evaluate them (Bellogín and Said, 2018, Gunawardana and Shani, 2015, Herlocker et al., 2004). Evaluating a recommender system involves running several test over real or simulated data, measuring the properties we want to check. Depending on how it is done, we might differentiate three types of evaluation: offline evaluation, user studies and online evaluation (Gunawardana and Shani, 2015), which we describe next.

Offline evaluation

Offline evaluation techniques perform tests over pre-collected data from real systems. Considering that the users provide ratings to the items similarly to how they did when the data was collected, offline tests simulate their behaviour (Gunawardana and Shani, 2015). Since they do not require interaction with real users, offline experiments are considered to be the cheapest and easiest to apply for evaluating recommender systems. However, the collected set of ratings might present biases which can distort the experimental results, such as biases caused by the sparsity of the datasets (Bellogín et al., 2017) or the skewness of the rating distribution (Cañamares, 2019, Cañamares and Castells, 2017, 2018). In addition to these biases, others might have been introduced by the underlying recommender system used during the period of time the data was collected (Fleder and Hosanagar, 2007).

Despite their limitations, offline experiments are still a widely used and useful approach for comparing the performance of different recommendation approaches. In academic research, they are the predominant evaluation methods, since they come as an affordable way of evaluating the recommenders and it is not common for researchers to have access to the real platform and their users. In industry, they provide an early-stage filter for selecting a reduced subset of algorithms which might be aligned with the main objectives of the system, without risking the users' trust (Bellogín et al., 2017). In this thesis, we focus on this type of evaluation, conducting experiments over data samples extracted from Twitter and Facebook.

The common procedure for offline experiments is the following: first, the pre-collected data is divided in several sets: tipically, a training and a test set. Multiple methods can be used for splitting the data. Campos et al. (2014) describe a framework for applying different partitioning strategies in offline experiments, such as randomly splitting the ratings in two sets, taking all the ratings before a given date as training and the rest of them as test, or fixing, for each user, a number of ratings for the test set, and taking the rest of them as training. Once the partition is done, the system takes the training set to create a model from the provided interactions (for instance, the latent factors for users and items in matrix factorization or the neighborhoods in memory-based collaborative filtering), and generates recommendation scores for the user-item pairs which are not present in the training set. Finally, these scores are used along the interactions in the test set to evaluate the different systems.

The evaluation depends on one or several metrics that quantify desirable properties of the algorithms. Some interesting properties of recommendation algorithms include their scalability, privacy, robustness, novelty or diversity, but, by far, the most well-known and studied evaluation dimension is accuracy, i.e. how similar the outcome of the algorithms is to the real behaviour of the users (Gunawardana and Shani, 2015). The reasons behind this are not surprising: if a recommender consistently fails at delivering good recommendations to the users, it is likely that they abandon the system, rendering it useless. Depending on how the accuracy of the system is measured, we differentiate two families of metrics: rating-based and ranking-based metrics.

Rating-based metrics assume that the goal of the recommendation algorithms is to accurately predict the values that users would assign to the item in the system. That way, if the ratings are close to the real ones, we can sort the candidate items by the system prediction to provide a good recommendation ranking to the user. Rating-based metrics focus then on measuring the difference between the predicted ratings and the ones in test. The most well known metrics for evaluating this differences are the mean average error (MAE) of the system and the rooted mean square error (RMSE). The first one can be defined as:

$$\text{MAE} = \frac{1}{|R_{test}|} \sum_{(u,i) \in R_{test}} |f_u(i) - r_u(i)| \tag{2.1}$$

where R_{test} is the set of user-item interactions in the test set, $f_u(i)$ is the recommendation score for the user u and the item i, and $r_u(i)$ is the real rating value for that pair, and the second as:

$$\text{RMSE} = \sqrt{\frac{1}{|R_{test}|} \sum_{(u,i) \in R_{test}} (f_u(i) - r_u(i))^2} \tag{2.2}$$

These accuracy metrics are the most dominant in the recommender systems literature, particularly in their early years (and they were further popularized thanks to contests such as the Netflix prize[3]). However, they

[3]Netflix prize: https://www.netflixprize.com/index.html (Accessed 19th December 2020)

have several flaws that should be taken into account. The main drawback of these metrics is that they do not distinguish between errors in the top items predicted for a system (the ones shown to the users) and the rest of them. Large errors on the first set of items might be balanced by a great prediction accuracy on the items that are not shown to the user. In addition to this, not all recommenders can be evaluated using these metrics, since they might provide recommendation scores outside the range of values that users are able to introduce in the system (Bellogín and Said, 2018).

In the last years, the attention on accuracy metrics has shifted towards *ranking metrics* (Bellogín and Said, 2018, Cañamares et al., 2020, Cremonesi et al., 2010). This family of measures brings into play the notion of relevance: an item is relevant to a user if satisfies one of his necessities. In the recommender systems field, we can say that a recommended item accomplishes that goal if the user likes it. This can mean several things: the user might have provided a high rating to the item, has written a positive review about it or simply, it has consumed it (for example, it has played a song that has been recommended to him in Spotify). Beyond the concept of relevance, ranking-based metrics also acknowledge that recommendation is usually presented to the user in the form of a ranking of fixed size (only a reduced number of items is shown to the user), by measuring the corresponding properties only over that set of top-ranked items.

Most of these metrics come from the information retrieval field, where relevance is a central notion (Baeza-Yates and Ribeiro-Neto, 2011). Some of these metrics, like precision or recall, are just focused on measuring the amount of relevant items in the recommendation ranking, whereas others such as mean average precision (MAP) or normalized discounted cumulative gain (nDCG) give more importance to the position of the relevant items in the recommendation ranking. Beyond information retrieval metrics, it is also possible to apply ranking-based machine learning evaluation techniques such as visualizing the receiver operating characteristic (ROC) or computing the area under it (AUC). As in the experiments we conduct throughout this thesis we mainly focus on ranking-based metrics, we will provide further details about some of them in Chapter 4.

Beyond the previously detailed options – how the split is done and which metrics we should use – there are many other decisions which have to be considered when applying offline experiments, which might change the outcome of the recommendation. We refer to the work by Cañamares et al. (2020) for a comprehensive examination of those options, such as whether to include in the recommendation ratings those user-item pairs which do not appear in the test set, or choosing a proper statistical test to determine if the differences between algorithms are due to random variation.

User studies

User studies involve the recruitment of a group of users for evaluating the system. These users are asked to perform different tasks over the final system, and their behaviour is then recorded to measure the different properties of the system we want to evaluate (Gunawardana and Shani, 2015). As an advantage, these studies allow a direct observation of how the users interact with the system, and measuring how the recommender systems influence them. It also allows the users to provide direct feedback in the form of comments during and after the experiments.

However, carrying user studies is not cheap: in order to enable significant results, it is necessary to recruit a large number of users. This carries another disadvantage: as the number of users grow, the time needed to conduct the evaluation also increases. In addition, the people selected for the experiment might present several biases which are not present on the real users of the platform: for example, volunteers are usually interested in the systems they try, and might have a greater expertise on the tested applications than others. Finally, these people are aware that they are participating on an experiment, so their behaviour might not fully align correspond to how they would use the system on their own.

Online evaluation

The last group of evaluation techniques takes advantage of the interactions of users with the real system to measure the performance of the recommenders. Several techniques can be used here. The most typical techniques are A/B tests, where a fraction of the traffic in the system is redirected to a different recommendation engine, and then, the interactions of the users with both systems (the one in production and the one we want to test) are recorded and compared (Gunawardana and Shani, 2015), but they are not the only ones. For instance, an-

other technique for comparing several recommenders in online production is ranking interleaving, where two (Chapelle et al., 2012, Hofmann et al., 2013) or many (Schuth et al., 2014) approaches are combined in a single ranking, and, depending on which items are clicked, the system collects evidence about the performance of each recommender.

Online evaluation techniques have the advantage of providing the most realistic results, since they are collected from the real system and real users. However, performing such experiments might be risky: providing irrelevant or bad results from the alternative recommendation solutions might lead to a reduction on the trust of the users in the system, and discourage users from using the real one. Also, it is not common for researchers to have access to the real recommendation platforms, making it impossible for them to conduct this kind of tests.

2.1.3 Beyond accuracy

Even though the accuracy of the recommendation is usually the main concern behind the development of new recommendation algorithms, it is not by far the only important measure of utility which can be used to evaluate and/or desing these systems. Properties such as the robustness, privacy or explainability of recommender systems have also been targeted on their design and evaluation (Gunawardana and Shani, 2015). In this thesis, we explore three of such properties: the scalability, novelty and diversity of the recommendations.

Scalability

As we have already commented several times through this section, the goal of recommender systems is to help people identifying items they might be interested in, from a collection from thousands to millions of items. However, as the number of items (and also the number of users) in the system grows, this task becomes more and more complicated: when this occurs, recommenders become slower or they require more computation power or space requirements to provide the recommendations in a reasonable time. Depending on how scalable the system is, just by multiplying the number of items in the system by 10, a system producing recommendations in a matter of seconds could even take days to generate them. This is a particularly critical problem in systems where the recommendations have to be provided in real time: users cannot be kept waiting for several minutes before the recommendation is delivered. Because of this, it is not uncommon to find cases where a trade-off between accuracy and speed is considered in the development of the algorithms (Pilászy et al., 2010, Yu et al., 2016).

There are many ways to measure the scalability of the recommendation approaches (Gunawardana and Shani, 2015). Theoretically, it involves determining the complexity of the algorithms (Cormen et al., 2009) in terms of both time and memory consumption (for example, as Hu et al. (2008) do). Empirically, there are different means to measure this: in offline experiments, it is possible to measure the speed and resource consumption as the number of ratings in the system grow (Yu et al., 2016) or as the hyperparameters of the algorithm vary (Pilászy et al., 2010); in online evaluation, we can measure the time needed for making a recommendation or the number of recommendations a system can provide per second.

Although we consider the scalability of algorithms throughout all the thesis (and we report, at least, the complexity of the developed recommenders), we particularly focus on this in Chapter 5, where we compare different contact recommendation approaches both theoretically and empirically, and in Chapter 10, where we analyze the cost of several interactive recommendation approaches.

Novelty and diversity

In addition to technical properties of the recommendation approaches, such as their scalability or robustness, it is also interesting to consider other utility dimensions at a more core and conceptual level of the recommendation problem. Two of these properties, novelty and diversity (Castells et al., 2015, Hurley and Zhang, 2011) have been earning attention since the beginning of the 2000s (Smyth and McClave, 2001).

The *novelty* of a system can be defined as how different the past and present experiences of the users with the system are (Castells et al., 2015). In recommender systems, the past experience is related to the knowledge the users already have about the items in the system (either thanks to external influence or earlier interactions with

the system), and the present one refers to the provided recommendations. Providing the users with novel recommendations is particularly important in the most common recommendation scenario, where users want to discover new items to consume: even when a recommendation algorithm provides accurate recommendations, if it keeps recommending items the users already know, they might become bored, and abandon the system because none of the recommendations are really useful for them.

The novelty of a system is measured from two perspectives. The first perspective is user-independent and considers how unpopular the recommendations provided by the system are. As an example in the social network domain, even when they do not follow them, it is more likely that an average user in the system knows about popular people such as Barack Obama or Katy Perry on Twitter than about other non-famous people like the author of this thesis. The second perspective measures how different the recommended items are from the ones the user already knew. As the information the user really knows is not commonly known – with the exception of some datasets like CM100k[4] (Cañamares, 2019) – novelty measures typically assume that the user knows what he has rated in the system.

The second perspective, *diversity*, ignores past experiences of the users. It focuses on studying how different the parts of the same experience are (Castells et al., 2015). In recommender systems, this means assessing the differences between items in the same recommendation list. Providing diverse recommendations can lead to different objectives, such as reducing the redundancy of the recommendations (Smyth and McClave, 2001, Vargas et al., 2011), dealing with uncertainty about what the user might like (Castells et al., 2015) or preventing the creation of the so-called filter bubbles (Pariser, 2011), a phenomenon that occurs when the personalization of the sources of information of a user leads him to be completely isolated from anything that he might not like.

Again, two different perspectives can be applied for measuring the diversity of a system. The first, local perspective, takes the point of view of the user. It measures how different the items in a recommendation list are among them. The second, is a global, bussiness perspective, which studies to which extent each item in the system has been recommended, analyzing how visible the different items in the system are for the users (Fleder and Hosanagar, 2007, Vargas and Castells, 2014). In addition to these perspectives, other diversity concepts from related fields such as information retrieval (Chapelle et al., 2011, Clarke et al., 2008, Zhai et al., 2003) – where diversity is understood as a means to deal with uncertainty – have also been adapted to recommendation (Vargas et al., 2011).

In Chapter 8, we explore the adaptation of these two concepts for the link prediction (Liben-Nowell and Kleinberg, 2007) and contact recommendation tasks. We adapt different metrics and explore the different meanings they could have for the task of suggesting people to follow or befriend in social networks. We also explore alternative notions of diversity, based on the changes that contact recommendation produces in the network structure.

2.2 Information retrieval

Information retrieval (IR) is an area of computer science oriented to providing easy access to information in unstructured (or semistructured) and massive spaces (Sanderson and Croft, 2012). Following the definition provided by Baeza-Yates and Ribeiro-Neto (2011):

> "Information retrieval deals with the representation, storage, organization of, and access to information items such as documents, Web pages, online catalogs, structured and semi-structured records and multimedia objects. The representation and organization of the information items should be such as to provide the users with easy access to information of their interest"

Since the 1950s, methods, theories, technologies and solutions for different IR problems such as text and multimedia search, text classification, automatic summarization or topic detection have been in research and development. However, it has only been from the 1990s, with the creation of the Web, that these technologies have been brought into the spotlight (Sanderson and Croft, 2012): the huge size of the Web (as of December 2020, the indexed fraction of the Web has been estimated to contain around 55 billion web pages[5]) makes

[4] CM100k dataset: http://ir.ii.uam.es/cm100k (Accessed 19th December 2020)

[5] Worldwide Web size: https://www.worldwidewebsize.com/ (Accessed 19th December 2020)

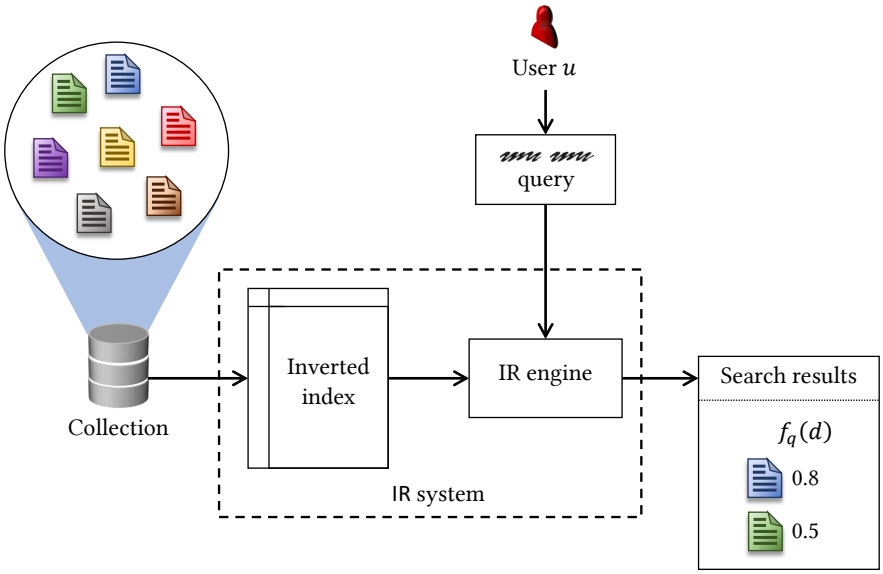


Figure 2.3: Arquitecture of a text IR system

unfeasible for the users to search and process all the information available. Therefore, platforms like Google, Bing or Yandex have become the most used web applications, issuing, as of 2019, more than 70,000 queries per second.[6] In this thesis, we focus on the study of the relation between contact recommendation and the most classic and paradigmatic area of information retrieval: text search (or, as we shall use equivalently to the rest of the document, text IR).

2.2.1 Text IR

Given a query issued by a user, a text IR system seeks to obtain the most relevant documents for the information need expressed by the query, based on an analysis of the words in the documents and the words in the query. We differentiate three different spaces for the IR task: the queries issued by the users, the document collection and the term space. Both queries and documents can be represented in the term space. For example, queries and documents can be modelled as vectors taking the possible terms as dimensions, and the frequency of those terms in them as the coordinate values (Salton et al., 1975).

Search engines are the technologies applied to solve the text IR problem. Most search engines have a similar architecture to the one illustrated in Figure 2.3. First, in order to build the IR system, we have to determine which documents will define the retrieval space (i.e. what documents can be retrieved by our system). This selection strongly depends on the application domain of the system. If we want to build a system useful for Web search, we need to recover a large fraction of the Web using crawlers, but, if we want to create a system to identify relevant documents about quantum mechanics, we just need to collect scientific articles related to the topic from journals or conferences.

After the documents have been collected, they are processed so the IR system is able to address the queries in an efficient an effective manner. The most common way to do this consists on indexing the collection (Baeza-Yates and Ribeiro-Neto, 2011). A typical structure for that is the inverted index. An inverted index is a structure with two differentiated parts. The first one, the dictionary, lists the terms in the vocabulary of the collection (i.e. the whole set of terms in all the documents). The second part includes, for each term, an associated posting list. The posting list includes, for each document in the collection that contains the term, information like its frequency in the document or the positions where the term appears (Büttcher et al., 2010). We illustrate an example of inverted index in Figure 2.4. This inverted index is where the IR system stores most of the data it needs.

[6] Google search statistics from HubSpot: https://blog.hubspot.com/marketing/google-search-statistics (Accessed 19th December 2020)

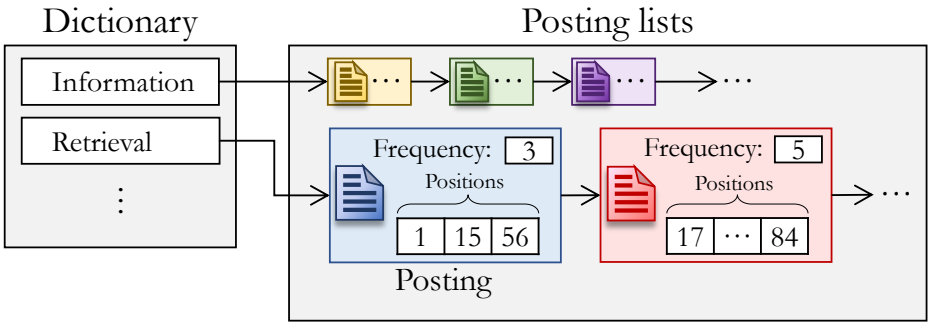


Figure 2.4: Inverted index example

Then, using the inverted index, the IR engine processes the queries that the users issue and, for each of them, provides scores to the documents according to a retrieval model. A retrieval model is a formal representation of the matching process between a document and a query (Croft et al., 2010). A model defines both a logical representation of both queries and documents (typically modelling the statistical properties of text) and a ranking function over that representation which is used to generate the scores. Multiple retrieval models have been designed and developed: from the classical vector space model (Salton et al., 1975) to more recent techniques such as neural retrieval algorithms (Mitra and Craswell, 2018), a wide variety of methods including probabilistic models (Robertson, 1977, Robertson and Zaragoza, 2009), language models (Lafferty and Zhai, 2001, Lavrenko and Croft, 2001, Ponte and Croft, 1998) or divergence from randomness (Amati, 2003, 2006, Amati and Van Rijsbergen, 2002, Amati et al., 2007, 2011) approaches have been proposed. Once the scores have been computed, the IR engine uses them to sort the documents, and, finally, shows a small fraction of the ranking to the user (for instance, Google or Bing just show 10 results to the user on the first page of results).

2.2.2 Relation between textual IR and recommendation

Connections between recommendation and textual IR date back to the earliest recommender systems and their relation to the information filtering task (Belkin and Croft, 1992). The development of content-based methods has been greatly influenced by information retrieval. By defining both user profiles and items in terms of side information such as tags, genres or textual descriptions, IR models like the vector space model (Salton et al., 1975) have been successfully applied to recommendation (Adomavicius and Tuzhilin, 2005). Recently, this connection has also led into the development of collaborative filtering algorithms (Bellogín et al., 2013, Parapar et al., 2013, Valcarce, 2015, Valcarce et al., 2017, Wang et al., 2008a,b).

A particularly representative and relevant proposal for this thesis was developed by Bellogín et al. (2013), allowing the adaptation of any IR term weighting scheme to create a collaborative filtering algorithm. To this end, the approach represents both users and items in a common space, where users play the role of queries and items the role of the documents to be retrieved. Depending on the definition of that common space, two different algorithms were proposed. In the first one, both users and items are represented in the user space. Each item is represented by its ratings (the users who rated the item) and users are represented by their similarities to other users. In the second one, users and items are represented in the item space: users are represented by their ratings, and items are represented by their similarities. Earlier on, Wang et al. (2008a,b), and later Parapar et al. (2013), explored similar developments upon different – more specific – IR-inspired probabilistic models. Our work pursues a similar goal to this research direction, but taking a step further: if Bellogín et al., Wang et al. and Parapar et al. folded three spaces (terms, documents, queries) into two (users, items), we fold them into just one, as we explain in Chapter 5.

Another relevant study for our work is the proposal by Valcarce et al. (2017), who explored the use of query likelihood models (Ponte and Croft, 1998) with different smoothing techniques as similarities for neighborhood-based collaborative filtering recommendations. Using datasets from different domains, they tested their approach and found that, indeed, using those similarities improves both the accuracy and the diversity of the recommendations when compared to others like cosine similarity. In Chapter 7 we adapt this

work to the task of recommending people to people, and we expand it by testing other IR weighting schemes as similarities for both user-based and item-based neighborhood-based methods.

Finally, there are already some established connections between the specific tasks we analyze in this thesis, link prediction and contact recommendation , and that of text information retrieval. For example, some prediction methods like the Jaccard index (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007) or cosine similarity (Lü and Zhou, 2011) are based on IR (Salton and McGill, 1983). More recently, Hannon et al. (2010) adapted the vector space model (Salton et al., 1975) to recommend users on Twitter, based on both content-based and collaborative filtering algorithms. Our work seeks to extend, generalize and systematize their point of view to adapt any state of the art IR model to contact recommendation.

2.2.3 Axiomatic thinking in IR

As the development of IR models advanced, it became necessary to determine and understand which properties of the models are key to providing effective results. This has been the objective of the research line known as axiomatic thinking, which, as developed by Fang et al. (2004), has permitted to guide the development of both sound and effective IR models by explaining, diagnosing and improving them. Axiomatic analysis seeks to provide a set of heuristics (known as axioms) that an IR system must (at least) follow to provide effective results.

In their seminal work, Fang et al. (2004) propose a group of heuristics addressing basic properties of classic models, such as the frequency of the query terms in the retrieved documents, the relative discrimination between query terms, or how a model deals with long documents. They also analyze the impact of such properties on the effectiveness of state of the art models such as BM25 (Robertson and Zaragoza, 2009), query likelihood (Ponte and Croft, 1998) and the pivoted length normalization vector space model (Singhal et al., 1998), and find that, by applying minor modifications to them to adhere to the different heuristics, the models improve their retrieval performance.

Since then, the original axioms have been refined and expanded (Fang et al., 2011, Shi et al., 2005). Other additional properties of information retrieval models have been studied, such as the semantic relations between queries and documents (Fang and Zhai, 2006), term proximity (Tao and Zhai, 2007), as well as techniques like pseudo-relevance feedback (Clinchant and Gaussier, 2013). In the last few years, axiomatic analysis has been applied on neural IR models: Câmara and Hauff (2020), Rennings et al. (2019) propose a way to check if neural models fulfil the different axioms empirically, while Rosset et al. (2019) apply the axioms for guiding the training procedure of neural models. Beyond text IR, axiomatic analysis has also expanded to other related areas such as recommender systems, where Valcarce et al. (2017, 2018) explore the benefits of penalizing users who rate lots of items when selecting neighbors in user-based k nearest neighbors (kNN).

In Chapter 6 we explore the importance of IR axioms for achieving good accuracy performance in contact recommendation. Taking the framework we introduce in Chapter 5 as a basis, we map the set of axioms proposed by Fang et al. (2011) and empirically analyze how valid and meaningful each heuristic is for this task.

2.2.4 Learning to rank

Learning to rank techniques appear in the confluence between information retrieval and supervised machine learning. Although we can describe, in a broad sense, learning to rank methods as those approaches that apply machine learning to solve ranking problems, it is common to use a more narrow definition, where learning to rank methods are machine learning methods which learn the optimal way of combining features extracted from query-document pairs through discriminative training to solve such problems (Hauff, 2019, Liu, 2007):

The strength of learning to rank techniques comes from their proficiency at combining multiple sources of evidence in a single model, represented as the set of features of the system. For a query q, each document d is associated with a feature vector $x = \Phi(q, d) \in \mathbb{R}^F$, where F is the number of features, that reflects the relevance of the document for the query (Liu, 2007). Multiple families of features have been used in a learning to rank scenario. We provide here a classification of such features in terms of how they depend on the query or the document:

- **Query dependent:** The value of these features depends on both the query and the document. Typically, these features involve the use of the scores provided by unsupervised models such as BM25 or query like-

lihood applied over different parts of the document, such as the content, title, snippets, etc. (Macdonald et al., 2013b).

- **Query independent:** These features vary from document to document, but not when we change the query. Examples of these features include the PageRank of a web document (Brin and Page, 1998), the length of the document or its spamminess.
- **Query features:** These features do not depend on the document: for a given query, all the feature vectors contain the same value of the feature. Examples of these features include the query length, predictors of the query performance or the query type (e.g. presence of entities, navigational vs. informational query).

Then, using these features, the methods are trained in a supervised discriminative manner: taking as a basis a training set, containing labelled examples (where labels might represent relevance degree of the individual documents, comparisons between pairs of documents or a total order relation between documents), learning to rank algorithms automatically learn their parameters by minimizing the differences between the outcome of the approach and the optimal outcome, represented by the ground truth provided by the labels of such examples.

Differently from other machine learning problems, such as regression or classification, which try to estimate the labels of the training data, learning to rank techniques aim to get the relative order of the set of documents with the highest predicted labels at the top of the ranking. This matches with the user expectations, who want the most relevant documents to appear in the first positions of the ranking, as it is rewarded by IR evaluation metrics such as MAP, MRR in the case of binary labels, nDCG (Järvelin and Kekäläinen, 2002), ERR (Chapelle et al., 2009) when we define several grades of relevance.

Depending on their output, we can classify learning to rank techniques in three categories: pointwise, pairwise and listwise methods: (Liu, 2007):

- **Pointwise learning to rank:** Pointwise methods represent a direct application of machine learning approaches such as regression or classification for the ranking task: the idea behind them is to learn the relevance degrees of each one of the documents, separately. The ranking is then built by sorting the documents according to those estimations. Despite being the simplest algorithms, they have their limitations: these methods do not consider interdependency between documents, and the final ranking is invisible to the system (the ranking in only built when the system has already estimated the scores for all the documents). For instance, this might lead the algorithm to spend lots of time minimizing the error of unimportant documents, which, at the end, will not be shown to the user. Many regression and classification functions have been proposed to use as pointwise learning to rank approaches, such as polynomial regression functions (Fuhr, 1989) or support vector machines (Nallapati, 2004).
- **Pairwise learning to rank:** This family receives as input pairs of documents, and estimates the relative differences between them. As an advantage over pointwise methods, this family considers the relation between documents. However, they only consider the relative order of pairs, which can be inconsistent and, looking only at that relative difference, it might be difficult for the system to derive the position of the documents in the final ranking. A list of pairwise approaches includes algorithms like RankNet (Burges, 2010) or FRank (Tsai et al., 2007).
- **Listwise learning to rank:** The last family of learning to rank methods directly addresses the task by providing full document rankings for the queries. These approaches do it by directly optimizing a ranking IR metric such as nDCG (Järvelin and Kekäläinen, 2002) or ERR (Chapelle et al., 2009), or by minimizing the differences between the document permutation provided by the method and the optimal one provided by the ground truth. AdaRank (Xu and Li, 2007) or ListNet (Cao et al., 2007) are examples of listwise algorithms.

As they are closer to the task they attempt to solve, Liu (2007) finds that pairwise and listwise learning to rank methods achieve better accuracy results for textual search than pointwise regression and classification methods.

In Chapter 7, we study how the methodology used for the development and deployment of learning to rank search engines in IR can be adapted for the task of recommending people in social networks. We then develop contact recommendation ensembles (using the equivalent of query-dependent features in contact recommendation) based on the LambdaMART model (Burges, 2010, Wu et al., 2008), a hybrid learning to rank algorithm combining pairwise loss with a listwise component optimizing IR metrics (tipically nDCG).

2.2.5 Novelty and diversity in information retrieval

The novelty and diversity concepts have also been addressed from different perspectives than the recommender systems ones. A relevant example for the work we develop in this thesis is the text search perspective (Clarke et al., 2008). Under an IR point of view, novelty and diversity are related to two common problems of search engines: the redundancy of the search results and the ambiguity and underspecification of the queries introduced by the users in the system.

The presence of duplicates (or near duplicates) in the document collection helps us illustrate the first problem. If we only use the text of the documents to generate the ranking scores, very similar documents should appear in close ranking positions (if they are duplicates, they might even receive the same score). Although two highly similar documents might be considered equally relevant for the user by the search engine, the truth is that the one that appears in a later position in the ranking provides no novel information to the target user, thus reducing the utility of the search. Consequently, we can improve the utility of the system if we reduce such redundancy.

For the second case, users usually write short and ambiguous queries. For example, if a search engine receives the query "Pluto", the query might be referred to several things: the dwarf planet in the Solar System, the Roman god of the underworld, Mickey Mouse's dog or even a manga authored by Naoki Urasawa, among other meanings. If the ranking only covers one of such meanings, it might not provide information to satisfy the information need of the user. If we try to cover as many aspects as possible, we might be able to provide at least one relevant search result for the user.

The study of novelty and diversity in IR has been addressed from two perspectives. The first perspective considers the evaluation of search engines. In this case, the evaluation has been commonly addressed in terms of the different aspects (also known as intents or subtopics) of the query (for example, in the case of the "Pluto" query, each of the possible meanings). A first family of metrics is defined upon the subtopic retrieval task, i.e. given a query, determining its different aspects. Zhai et al. (2003) propose several metrics for this task which have been used as diversity metrics in IR, such as subtopic recall or subtopic precision. Agrawal et al. (2009) propose another group of metrics, known as intent-aware metrics, which generalize IR metrics targeting the accuracy of the search task to consider the different intents of the queries. One of the most used intent-aware metric is ERRIA (Chapelle et al., 2011), but others based on nDCG or MAP have also been considered. Finally, other metrics generalize IR metrics, but penalize the appearance of documents covering the same aspects of the query in close positions in the ranking, such as α-nDCG (Clarke et al., 2008).

The second perspective addresses the enhancement of the novelty and diversity of search rankings. Several greedy approaches have been proposed for jointly optimizing the relevance of the search ranking documents and its novelty and diversity. Such methods reorder (rerank) the original ranking provided by a search engine in different manners. We differentiate two families of proposals, depending on whether they use explicit aspects of the document or not. Implicit methods like maximum marginal relevance (MMR) (Carbonell and Goldstein, 1998) assume that two documents cover similar topics and intents if their texts are similar. These approaches define a document similarity measure (like the cosine, Dice or Jaccard similarities) and address reducing the similarity between the elements in the rank. Explicit methods directly exploit the different aspects of a query to maximize their coverage in the search ranking. Notable explicit approaches include the explicit query aspect diversification method (xQuaD) devised by Santos et al. (2010a,b) and the IA-Select approach (Agrawal et al., 2009).

In recent years, such metrics and diversification techniques have been adapted for the recommender systems field considering side-information such as the genres of movies as the intents or aspects of the recommended items (Vargas et al., 2011, 2012). In Chapter 8 we follow the work by Vargas et al. (2011), by adapting several IR diversity metrics to the contact recommendation problem. Then, in Chapter 9, we propose a new algorithm to enhance global metrics for contact recommendation methods.

2.3 Social networks

A social network is a representation of some pattern of contact or interaction among a group of people or several groups of people (Newman, 2003). The nature of the network gives a meaning to these patterns of interaction,

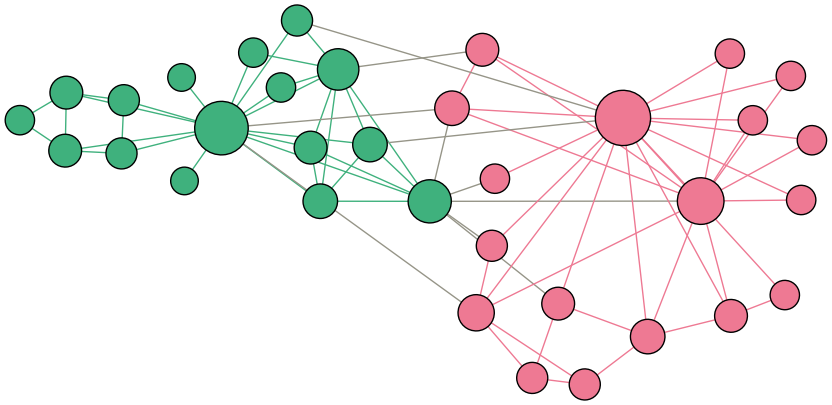


Figure 2.5: Zachary (1977) karate club network.

which can represent, among others, friendship, love or professional relationships, professional interactions such as collaboration on scientific articles or acting on the same movies, or paths of disease contagion.

The earliest documents introducing notions of social network analysis were introduced in the final years of the 19th century by Tönnies (1887) and Durkheim (1893). However, the concept of social network was not properly introduced until the 1930s, when Moreno (1934) established the concept of sociogram: a representation of the relations between people in the form of a network. In his work, Moreno also iniciated the sociometry field – the mathematical study of psychological properties of populations, focusing in their structure – which would later develop into social network analysis. Since then, social networks have been the object of study of many different fields and disciplines, including sociology, psychology, biology, physics, statistics or computer science (Newman, 2018). Social network analysis has multiple applications, including (but not limited to) the analysis of the spread of diseases over a population, the identification of important people in the network, the prediction of social dynamics or even planning marketing campaigns.

The development of social network analysis has been notably impulsed by the creation of online social media platforms in the late 1990s. Facebook, Twitter, LinkedIn or Instagram are used daily by hundreds of millions of people worldwide, who transfer social information to those online platforms. The availability of that huge amount of information has open new opportunities for the study and exploitation of social data, from both a business and a research perspective – similarly to how the appearance of the Web gave new life to information retrieval.

Social networks can be mathematically modelled using graphs. A graph is composed by two elements. First, its fundamental elements are known as vertices (also known as nodes or actors). Each person in the network is represented as a vertex in the graph. Second, the edges: lines joining pairs of vertices. The existence of an edge (which might also receive the name of link or tie) in a network indicates that there is some kind of relation or interaction between the two nodes it joins. Figure 2.5 shows the graphical representation of one of the most well-known examples of social network, modelling the friendship connections between members of a karate club in the 1970s, as identified by Zachary (1977).

Social networks can be categorized according to different criteria, with the most common ones depending on the properties of the links between nodes. The most well-known and used criteria for categorizing social networks depends on the reciprocity of the interactions represented by the edges:

- **Directed or asymmetric networks:** In this family of networks, the interactions between networks do not have to be reciprocated. Citation networks (where one user is related to another if the first has cited the second in a publication) or follow networks such as Twitter or Instagram represent examples of directed networks. In this type of networks, the relations are represented as directed edges in the graph model.
- **Undirected or symmetric networks:** This family represents relations which have to be reciprocated, such as love or friendship relations. Examples of undirected networks include collaboration networks (where there is a link between users if they have worked with each other) or friendship networks as the ones in Facebook or LinkedIn. In the graph representation of the network, the edges are represented as undirected.

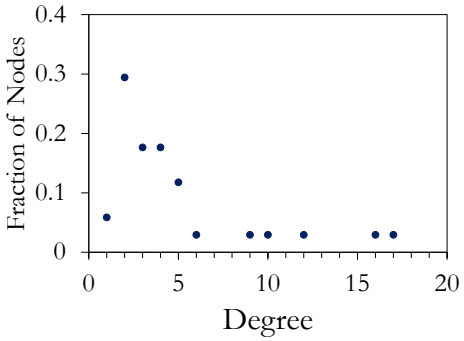


Figure 2.6: Degree distribution for the Zachary (1977) karate club network in Figure 2.5

A second classification depends on whether the edges have weights associated or not. Weights represent a quantitative property associated to the connection between two people (for instance, how many times two users have interacted to each other or the physical distance between them). Networks where the links are associated a weight are known as **weighted networks** whereas networks without edge weights are referred to as **unweighted networks**.

The network science and social network analysis fields are very broad, covering many problems and disciplines. Consequently, in this section, we only provide an overview of the main research lines related to the work we present in this thesis. We start by introducing the basic properties of nodes, vertices and social network as a whole; afterwards, we give some insights on the community detection and information diffusion problems; we then focus on the creation of new links and evolution of network and, finally, we explore the concept of diversity in social network analysis.

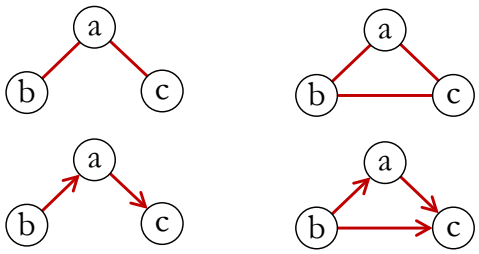
2.3.1 Structural properties of social networks

One of the main tools of social network analysis since its origins has been the study of the structure of the networks, which leads to a better understanding of communication patterns between users, determining how the network evolves and grows, etc. Myriads of concepts have been explored to determine the characteristics of a network, but, in this section, we summarize some of the most important, and we provide some insights on their value for real-world networks, such as collaboration networks or the underlying networks in social media platforms as Twitter or Facebook.

Degree

One of the simplest (though more important and widely studied) structural properties that we can measure in social networks is the degree of the nodes. The degree of a vertex can be defined as the number of edges which have the vertex as an endpoint. In case we only allow a single edge between a pair of users, this is equivalent to the number of neighbors (the nodes which can be reached by traversing a single link). In directed networks, we differentiate two values: the out-degree of the node (the number of edges in the network which have the node as origin) and the in-degree (the number of incoming edges). In order to study the network as a whole, it is common to study the degree distribution of the network, where the possible degrees values are shown against the proportion of nodes in the network with that degree value. Figure 2.6 shows, as an example, the degree distribution of the the network in Figure 2.5.

In real-world networks, the degree distribution is usually very skewed, with most of the nodes having low degree values, whereas a few nodes have high degrees (Newman, 2018). Those high degree values are known as hubs, and represent nodes with connections to a large fraction of the network. In directed networks, similar trends are shown for both the in-degree and out-degree distributions. These degree distributions are related to the preferential attachment phenomenon in social networks: when the network evolves and new links appear, nodes with high degree are more likely to receive new connections that nodes with low degree (Barabási and Albert, 1999).



(a) Open (non-transitive) triad (b) Closed (transitive) triad

Figure 2.7: Examples of open and close triads. Upper row shows examples for undirected networks and lower row for directed ones.

Paths

A path between two nodes u, v can be defined as the sequence of links that a walker has to traverse to reach node v starting from u. The most interesting subset of those paths comprises the minimum distance paths, also known as shortest paths or geodesic paths (Newman, 2018): the minimal sequences of links which can be defined to reach v from u. The number of links in those paths is known as the distance between u and v (or the path length). When distances are averaged over the different pairs of nodes in the networks, they represent one of the most relevant properties of networks: the average distance between nodes in the network.

This measure is central to the concept of small world networks, i.e. networks where the distance between pairs of nodes is very small when compared to the size of the network (Watts and Strogatz, 1998). This effect was first observed by Stanley Milgram (1967), who devised the following experiment: he gave letters to several people, and asked these people to deliver them to a certain person. If the holder of the letter knew its final recipient, he should send it directly to him. Otherwise, the person had to find an acquaintance who might know who the receiver was. That person would then repeat the process. From the letters that arrived, Milgram found that, in average, the letters had followed six steps before they reached their destination.

Online social networks are also examples of small world networks: in 2012, the average shortest path length in Twitter was around 4.05 when the network had 175 million users (Myers et al., 2014), and the average distance between two users in the Facebook network in 2011 – when Facebook had around 721 million active users – was just 4.3 steps (Ugander et al., 2011). In fact, when the networks grows in number of nodes and links, the distance between nodes decreases: Leskovec et al. (2007) study this phenomenon and find that the diameter of the networks (the maximum distance between nodes) decreases over time.

Beyond the average distance, there are many other measures centered around the concept of distance and shortest paths: the eccentricity of a node shows the distance between the node and the farthest vertex in the network (Dankelmann et al., 2004, Ilić, 2012), the betweenness of an edge computes the number of shortest paths traversing that link or the closeness of a node measures its mean distance of a node to other nodes (Newman, 2018).

Triadic closure

Another important property of social networks is related to the concept of triadic closure. A triad is just a sequence of three nodes which form a path of length 2, i.e. if we consider three nodes, u, v and w, they form a triad if u follows v and v follows w. Such triad is transitive (or closed) if there is a link between the origin of the triad and its ending. We illustrate examples in Figure 2.7.

As real-world networks typically show a large fraction of transitive triads (Newman, 2018), triadic closure represents the basis for many link prediction and contact recommendation methods, as we see in Chapter 3. Network metrics have also been devised to measure this, such as the clustering coefficient, which measures the proportion of closed triads in the network (Newman, 2018). A local definition of this metric just considers the proportion of transitive triads among the paths of length two which have a node as their middle point (Watts and Strogatz, 1998).

Components

Finally, another interesting property of social networks is the presence of connected components. A connected component is a subset of the nodes of a network for which there is, at least, a path from any node in the set to any other. In undirected networks, connected components represent clusters of nodes without links to nodes in other components. In directed networks, we differentiate two kinds of connected components: weakly connected components, which represent the set of connected components the network would have if it was undirected – i.e. groups of nodes without connections to/from other groups –, and strongly connected components, considering the direction of the edges. Strongly connected components are fully contained inside the weakly connected ones.

Related to these components, another interesting phenomenon occurs: in real-world networks there is often a connected component containing more than half of the nodes in the network (known as the giant component of the graph). For example, in 2012, the largest weakly connected component in Twitter contained the 92.9% of the users in the network, whereas the largest strongly connected one included the 68.7% of the users (Myers et al., 2014). In 2011, the giant component of the Facebook network was formed by the 99% of the people in the network (Ugander et al., 2011).

2.3.2 Communities

A common phenomena that has been observed in several types of networks is their community structure (Fortunato, 2010, Newman, 2018): as, following the homophily principle (McPherson et al., 2001), people connect with people with similar characteristics (tastes, geographical location, etc.), they tend to gather into groups or communities where the vertices are tightly connected among them, but only a few links travel outside each group. This grouping naturally appears in social networks, where such groups might represent families, sports teams, companies, etc. An example of community partition can be observed in the Zachary karate club network in Figure 2.5. In his work, Zachary (1977) studied the relations between people in a karate club. Due to differences between the karate instructor and the administrator of the club, the club was eventually separated into two groups, which we show in the figure using colors: each color represents one of the resulting clubs. If we observe in such structure, most of the links appear between people in the same group: only few edges go from one club to another, following the previous definition.

The analysis of the community structure of a network can provide interesting insights about the formation of networks and its structure (Fortunato, 2010): for instance, it can be used to determine how a company is organized. This structure can also be used to simplify the graphical representation of networks (Newman, 2018) and it can even be exploited for improving the accuracy of link prediction and contact recommendation (Valverde-Rebaza and de Andrade Lopes, 2013, Wu et al., 2017). In addition to these uses, the problem of finding community structure in social networks is an interesting problem on itself, and it constitutes one of the areas of social network analysis, called community detection (Newman, 2018).

One of the most common and effective family of community detection algorithms maximizes the modularity of the network (Newman and Girvan, 2004). Given a community partition, the modularity measures the number of links inside communities when compared to the expected number of links inside of those clusters in a configuration network (Newman, 2018) where the degree of the nodes is kept, but the edges are distributed randomly. It is known that a partition of the network which maximizes its modularity exists, but, unfortunately, the problem is NP-complete (Brandes et al., 2008), so no exact method can be proposed.

Nonetheless, several heuristic proposals have been devised to deal with this optimization problem. Clauset et al. (2004) introduce the FastGreedy algorithm, which starts by assigning a different community to each node, and then, for each pair of communities, computes the modularity increment that merging them would produce. Then, it merges the pair of communities producing the maximum increment. The Leading Eigenvector approach (Newman, 2006) successively divides communities into two subgroups. For the corresponding community, it computes a modularity matrix, and later, finds the eigenvector associated to the largest eigenvalue of such matrix. The nodes corresponding to the positive coordinates of such eigenvector form a community, and the rest of nodes the other community. Another algorithm, known as Louvain (Blondel et al., 2008), improves the modularity of the network in several phases: first, starting with all nodes in different communities, moves nodes from one community to another only if they improve the modularity of the network. Once the modu-

larity is no longer improved, the algorithm condenses the network into another one which takes the detected communities as nodes and repeats the whole process until no further modularity increase is achieved.

Beyond modularity maximization approaches, other heuristics have been applied for the detection of community structures in networks. Newman and Girvan (2004) apply a method based on the betweenness of the edges, i.e. to the number of shortest paths that traverses each edge. Considering that there are few links between communities, they expect them to have a high betweenness score, so they build a divisive algorithm in which they iteratively remove the edges with the largest betweenness values. Raghavan et al. (2007) devise a label propagation method in which, iteratively, each node is associated to the community of the majority of his neighbors. Finally, Rosvall and Bergstrom (2008) consider concepts from information theory to devise an algorithm for finding the community structure of a network. In their algorithm, Infomap, they identify communities by minimizing the number of bits needed to represent a random walk in the network.

Comparing with an artificial ground truth (Lancichinetti et al., 2008), Yang et al. (2016) evaluate eight different community detection methods in terms of their effectiveness and efficiency. They find that the Louvain algorithm (Blondel et al., 2008) is not only the algorithm providing a better community partition for several network configurations (regarding the fraction of links traversing communities and the size of the networks), but, is also notably fast, finding communities in $O(|\mathcal{U}| \log |\mathcal{U}|)$ time, where $\mathcal{U}$ represents the set of users in the social network.

In our work, we consider the community structure of the network in the design of novel metrics for contact recommendation algorithms, targeting the potential effect of such approaches on the structural properties of the network. As an effective method (Yang et al., 2016), we consider the Louvain algorithm (Blondel et al., 2008) as our preferred community detection algorithm in our study of such properties.

2.3.3 Diversity in social networks

The notion of diversity in social networks has been a profuse and broad field of study. Many different definitions have been proposed for it – often interrelated – and this diversity has been associated to multiple advantages not only for the individual users, but for the network as a whole. In this section, we overview some of these definitions, as well as the advantages conferred to them. In particular, we bring attention to the notion of structural diversity: the notion of diversity related to the connections between different users in the networks (Burt, 1995, De Meo et al., 2014, Granovetter, 1973, Ugander et al., 2012).

The most relevant and important concept for defining the structural diversity of a network is the notion of weak tie (Aral, 2016, De Meo et al., 2014, Easley and Kleinberg, 2010, Granovetter, 1973). From a sociological point of view, we can define the strength of a link as "*a (probably linear) combination of the amount of time* [spent on the relationship],*the emotional intensity, the intimacy (mutual confiding), and the reciprocal services which characterize the tie*" (Granovetter, 1973). Strong links represent close and intimate relations – those we have with friends or family – whereas weak links represent ties with more casual acquaintances, such as people we meet at conferences, shopkeepers at the local market, etc.

Granovetter (1973) hypothesizes that weak ties in the network are a source of more novel information and resources than strong ties and, they thus play a major role in the diffusion of information. For instance, an observation Granovetter does is that people often rely on loosely related people (weakly linked to the person) to find new jobs. His hypothesis maintains that the novelty of the information and resources in a network comes from a subset of the (weak) edges that he calls bridges. He proposes two different definitions of bridge: global bridges, which are ties that connect strongly connected components of the network, and local bridges, which are related to the concept of redundancy: an edge is a local bridge if the endpoint users do not have any common neighbors.

These definitions are quite restrictive in actual networks, such as Facebook or Twitter ones, due to phenomena like the giant component one. Therefore, taking into account that Granovetter (1973) also states that these bridges are only a fraction of the weak ties in the network, several studies have proposed some relaxed definitions of weak links. Starting from the definition of global bridge, De Meo et al. (2014) consider as weak connections those which travel across pairs of communities in the network. As communities are restricted to connected components, global bridges also fall under this definition. Meanwhile, the triadic closure principle makes difficult to find local bridges, so, instead of using a binary definition of link strength – considering that a link is strong if the endpoints have at least one common neighbor and weak otherwise –, Easley and Kleinberg

(2010) and Zhao et al. (2010) propose an alternative definition, which measures the strength of a link as the proportion of common neighbors between the endpoints of the edge – the so-called embeddedness of the tie.

Another definition for the structural diversity of the network related to the notion of redundancy of the edges is the one by Burt (1995). Burt studies the diversity of networks from an economic point of view. He states that an economic player with an optimally structured network of contacts enjoys greater benefits since he receives better information, such as new funding opportunities, early information about new items entering the market, etc. Burt recognizes that increasing the size of the neighborhood is the most common way to have access to more information, but, unless this growth considers the diversity of the users, it might have negative effects for the user. As a concept of structural diversity, he proposes structural holes, or non-redundant connections.

In terms of the network connections, a structural hole is defined as the absence of two conditions in a connection: structural cohesion and structural equivalence. Under the structural cohesion condition, two contacts are considered as redundant if they are connected to each other by a strong relationship. In this case, since they exchange information between them, they are likely to provide the same information. The structural equivalence criterion establishes that two contacts are redundant if they have the same contacts (or lead you to the same clusters of people), since they do receive the information from the same sources. Burt (1995) does not consider these properties as absolute definitions, but allows different degrees (for example, two users might be connected to the same cluster of users, but have connections to others).

A user-centric notion of structural diversity is considered by Ugander et al. (2012), who ponder the number of structurally cohesive groups of neighbors of the user (without direct connections between them) as a measure of the diversity of his environment. Using this definition, they show that users with highly diverse environments are more likely to join social media platforms such as Facebook. Also, they refine their notion of diversity, analyzing the number of k-cores (connected groups of neighbors of a user with all the nodes having degree greater than k) or k-braces (connected groups of neighbors of a user with the nodes sharing, at least, k neighbors). Under those definitions of diversity, they observe that greater diversity leads to improvements on the user engagement in the social media platform.

Eagle et al. (2010) take as a measure of structural diversity how equally people communicate to individuals in other geographical communities. Using data extracted from a communications register in the United Kingdom, for each user in the network, they find the entropy of the volume of calls he has with (a) his contacts and (b) the different geographic communities as diversity metrics, and establish relations with the income of those geographical regions. They find that the economic wealth of the region correlates with the diversity of the connections of the users. This observation is reinforced when the previous diversity is combined with the concept of structural hole proposed by Burt (1995).

Other studies propose the use of side information to determine how diverse a social network (or a part of the network is). This is the case of the study by Barefoot et al. (2005), who categorize the relationships of some users attending to the nature of the relationships (family, friends, work colleagues, neighbors, etc.) and the frequency of interaction (daily, weekly, monthly, rarely, never). Using the information collected in a study conducted in Copenhagen, they analyze how this diversity correlates with the risks of suffering and dying from ischemic heart diseases. The study concludes that keeping connections to a diverse set of people reduces such risks.

Finally, Bakshy et al. (2015) explore the diversity of the information reaching the different users in the network. They categorize a set of users according to their political ideology (conservative or liberal), and measure the proportion of the information they receive which includes news and posts from the opposite view. They conclude that the level of exposure to contents from other political ideologies depends on the second degree connections of the users in the network (i.e. on friends of friends).

In this thesis, we explore the effect that contact recommendation algorithms have on the structural properties of the social network, with a special focus on structural diversity. From the different definitions provided here, we concentrate our efforts on studying those definitions related to the concept of weak tie (De Meo et al., 2014, Easley and Kleinberg, 2010, Granovetter, 1973, Zhao et al., 2010). We propose several metrics in Chapter 8 and, later, in Chapter 9 we analyze the effects of enhancing these properties on the diversity of information that travels through the social network.

2.3.4 Information diffusion

The whole concept of social network relies on the interactions between people. Although there are different ways for people to interact to each other, it is hard to think about a more important, meaningful and representative interaction than information exchange. Communication represents the way we gain new knowledge, establish new relations to people and it is even a way to entertain ourselves. For that reason, most online social networks are built upon the concept of sharing information. Let's take Twitter as an example: every user in the network is able to publish a short message (known as tweet), which reaches all her followers. The followers, at the same time, can share that tweet with more people, and so forth, leading to the creation of cascades and flows of information spread through the network.

The process by which pieces of information (knowledge) are spread and reach individuals through interactions is known as information diffusion (Zafarani et al., 2014) and represents an important field of study in social network analysis. We differentiate four elements in the information diffusion process:

- **Information piece:** This element represents the message, the knowledge that travels through the network.
- **Sender:** The sender represents the creator of the information piece: a person (or a group of people) who start the diffusion process by introducing the information in the system.
- **Receiver:** This defines the person or group of people who have access to the information piece during the diffusion process.
- **Medium:** It represents how the information travels between users. For instance, in social networks, the links between individuals would represent the medium through which the messages spread.

The study of the diffusion process has many practical applications. Sociologists have studied how new products or ideas (or, more generally, innovations) spread through populations (Rogers, 1962, Zafarani et al., 2014). Such studies are also useful for marketing companies, since they can use them for the detection of influential users that maximize the spread of their products (Guille et al., 2013, Kempe et al., 2003). Another important and interesting application of diffusion is the detection and stoppage of infectious diseases like HIV or influenza (Easley and Kleinberg, 2010, Hethcote, 2000, Newman, 2018). Finally, the spread of information can also provide some insights on the creation of new connections in the social network (Farajtabar et al., 2017).

The simplest way to understand a diffusion process for a single information piece is considering that a vertex of the network has two possible states: activated, if the node has received the information and it is willing to propagate it, or not activated otherwise. Then, we can understand the diffusion process as the successive activation of nodes over time (Guille et al., 2013, Zafarani et al., 2014). Although there are more complex processes (for example, epidemic models might consider the inactivation and reactivation of nodes), this simple perspective is enough to describe the work we present in this thesis. Under this definition, two factors affect the way information spreads through the networks: the diffusion model (or protocol) and the structure of the social network.

The diffusion model is an abstract and simplified formulation for explaining how information travels through networks. Plenty of diffusion protocols have been proposed. We overview here some of the simplest and most widely known ones. First, in the independent cascade model (Goldenberg et al., 2001), a vertex u can activate one of her neighbors, v according to some probability, p_{uv}, which only depends on both nodes. A simpler model, known as the linear threshold model (Kempe et al., 2003) establishes that a node is activated only when a given fraction of his neighbors are activated. Finally, Doerr et al. (2011) models the rumor spread through networks using three protocols originally devised for the exchange of information through databases (Demers et al., 1987): the push model, in which an activated node randomly activates some of its neighbors, the pull model, in which a node selects a set of neighbors and activates itself only if one of these neighbors are already activated, and the push-pull protocol, combining both approaches.

The structure of the network itself plays a major role in the diffusion of information. Most works have focused on the impact of strong and weak ties (under different possible meanings) on the speed and coverage (the proportion of users receiving the information) of the spread bits. The work by Goldenberg et al. (2001) shows that under the independence cascade model, the influence of ties in the diffusion speed between tightly connected clusters of users is at least as powerful as that the influence of strong ties, although the speed depends on other factors such as the size of those clusters. On that line, using another weak tie definition, based on the

number of interactions between users, Bakshy et al. (2012) check that weak links are a source of novel information and are more influential than strong ties for making Facebook users reshare information. Zhao et al. (2010) consider that favoring not redundant ties on the network for spreading information does just have positive subtle effects on the speed over using any type of link without distinction, but the presence of such links has a great impact on the coverage of the diffusion. De Meo et al. (2014) observe that removing links between communities slows the diffusion process in Facebook networks, and, finally, Centola (2015), considering a threshold model, shows that community structure is desirable for spreading information as far as possible, as long as there are enough connections between the clusters. Beyond weak and strong ties, Doerr et al. (2011), using the push-pull protocol, show that preferential attachment networks (Barabási and Albert, 1999) are better for spreading rumors than random networks (Erdös and Rényi, 1959) thanks to the existence of connections between hubs and low-degree users.

We pay special attention to information diffusion in Chapter 9, where we analyze the effects that contact recommendation might have on the properties of the information flow. Particularly, we study potential effects on the speed of the diffusion, the novelty of the pieces reaching the users, and their diversity.

2.3.5 Link prediction

A particularly compelling problem in network science for this thesis is link prediction. Link prediction seeks to accurately predict the formation of new links in a social network (Liben-Nowell and Kleinberg, 2007) taking as input its current structure. In social networks, the utility of link prediction is manifold: understanding the mechanisms that drive the growth of the network, detecting missing and spurious links (Guimerà and Sales-Pardo, 2009, Hasan et al., 2006), predict missing labels of nodes in networks (Lü and Zhou, 2011) and even for recommending people in social networks (Backstrom and Leskovec, 2011, Barbieri et al., 2014).

Several approaches have been defined for solving the link prediction task. Liben-Nowell and Kleinberg (2007), Lü and Zhou (2011), Martínez et al. (2017) provide an extensive collection of algorithms, both unsupervised and supervised. Unsupervised techniques infer scores for each possible link considering the topology of the network. Several methods consider the common neighbors between the endpoints of the new possible links as the main predictor, as it is the case of the Jaccard similarity (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007) or Adamic-Adar (Adamic and Adar, 2003). Others, like Katz (Katz, 1953, Liben-Nowell and Kleinberg, 2007) and the local path index (Lü and Zhou, 2011, Lü et al., 2009) take advantage of the links between pairs of nodes. Other algorithms rank the links according to the probability that a random walker starting from one user reaches another, as personalized PageRank (White and Smyth, 2003) or SimRank (Jeh and Widom, 2002) estimate. Finally, more complex methods based on probabilistic models (Clauset et al., 2008, Guimerà and Sales-Pardo, 2009) or matrix factorization (Menon and Elkan, 2011) have also been devised. Supervised approaches, on the other hand, consider the link prediction problem as a classification problem with two classes: the presence or absence of the link in the network.

In this work, we consider the use of some link prediction approaches for contact recommendation in social networks. In Chapter 3, we first delve into the differences between contact recommendation and link prediction, and, afterwards, we describe a battery of algorithms (including some methods originally designed for link prediction).

2.4 Reinforcement learning

The natural learning process of human beings or animals involve their continuous interaction with their surroundings, other life forms or themselves. These interactions, from the simplest ones like touching a surface to the most complex like the development of scientific experiments in a laboratory, have consequences – either positive or negative – that constitute a fundamental source of knowledge or experience for fulfilling some goals. As a simple example, we might consider a puzzle designed to help an infant learn about different shapes. In order to solve that puzzle, the child has to match a set of pieces with holes shaped like them. When he first tries to solve the puzzle, it is quite probable that his first match is incorrect, and the piece does not fit in the selected gap. Then, the child has to follow a trial and error process for finding the correct relation. The process is then repeated for the subsequent shapes. Then, the second time he attempts to complete the puzzle, as he has some

previous knowledge, he is able to finish it faster. After several repetitions, the toddler is able to introduce each shape in its corresponding position on his first attempt – showing that he has learnt how the puzzle works.

Reinforcement learning (RL) represents a machine learning paradigm mimicking such natural learning process for solving problems (Sutton and Barto, 2018). Under this paradigm, an agent has to perform a series of actions to achieve a goal, which usually involves maximizing a numerical signal. However, the agent does not know beforehand which actions it has to take: it must acquire the knowledge about what it has to do by interacting with its environment and observing how the environment reacts to those interactions. These systems are characterized by two challenges:

The first challenge is the exploration-exploitation dilemma (Sutton and Barto, 2018): in order to maximize its earnings, the agent has to use (exploit) the acquired information to perform those actions it thinks to be more beneficial at long term. However, as the agent only has incomplete knowledge, it is possible that other actions might be more beneficial than those it is applying. Therefore, the agent needs also to discover (explore) them. The second involves the difference between short term and long term benefits: in reinforcement learning, it is assumed that current actions have an impact not only on the immediate feedback of the environment, but also on the feedback for future actions. For example, in chess, sacrificing a pawn or a bishop might be understood as a negative movement (it might seem that having less pieces to use is bad for the player), but, if this single loss leads to capturing the enemy's queen (the most powerful piece in the game) then, the long term benefits surpass the losses. Reinforcement learning systems have to consider this kind of trade-offs.

A reinforcement learning system is characterized by several elements which describe its inner workings:

- **Agent:** The agent is the learner and the decision maker in the reinforcement learning problem. It has one or several explicit goals that it attempts to satisfy. It can (at least partially) sense the environment, and it can even affect it by performing actions. Each time step t, the agent takes an action $a_t \in \mathcal{A}$ to interact with the environment – with $\mathcal{A}$ representing the set of all possible actions.
- **Environment:** It represents what an agent interacts with (everything external to the agent) and provides information to it. The environment might have several states $\mathcal{S}$ which affect the set of actions an agent can undertake. For instance, in chess, an agent cannot move a piece that has been caught, or cannot move another piece whose movement leaves the king unprotected.
- **Policy:** The policy is the strategy followed by an agent to fulfil its goals. Mathematically, given the state $s \in \mathcal{S}$ of the environment, a policy can just be defined as a (more or less complex) mapping $\pi : \mathcal{A}(s) \rightarrow \mathbb{R}$ between the actions an agent can choose to do in the state, $\mathcal{A}(s)$ and the probability the agent has of selecting that action. Policies are usually stochastic to deal with the uncertainty.
- **Reward signal:** The reward is a numerical value $r \in \mathbb{R}$ that encodes the feedback the environment provides the agent as a consequence of its actions. The goal of a reinforcement learning algorithm is to maximize the total reward that the agent receives over time. Along with the changes in the environment state, the rewards represents the main influence for modifying the policy's behaviour.
- **Value function:** Whereas the rewards received by the agent show the immediate profit (or backslash) of performing an action, the value function v provides an estimation about how good the different actions are in the long run. Starting from a state s, the value function is defined as the total amount an agent can expect to obtain in the future using the current policy. The correct estimation of the value function is fundamental for the success of a reinforcement learning algorithm.
- **Environment model:** This is a simplification of the environment that allows the inference of its behaviour over time. For instance, given an action $a \in \mathcal{A}$ and a state $s \in \mathcal{S}$, the model can be used for estimating the next state and reward. Models are commonly used for planning the future course of action of the learning agent. Not all reinforcement learning algorithms need a model to work.

Considering all these elements, Figure 2.8 illustrates the basic reinforcement learning process: given the current state of the environment, s_t, the agent uses the policy π to select an action $a_t \in \mathcal{A}(s_t) = \{a_{t,1}, \cdots, a_{t,n_t}\}$, where n_t is the number of possible actions in state s_t. The policy chooses this action considering the estimations of the value function ($v_{j,t}$ for $1 \leq j \leq n_t$) for the different possible actions. Then, the agent executes the corresponding action (the $a_{t,1}$ action highlighted in yellow), which causes a reaction on the environment. The environment, as a response, returns a) the reward for the action, r_{t+1} and b) the state for the next iteration, s_{t+1}. The reward is then used to update the policy, and the process starts again for iteration $t + 1$.

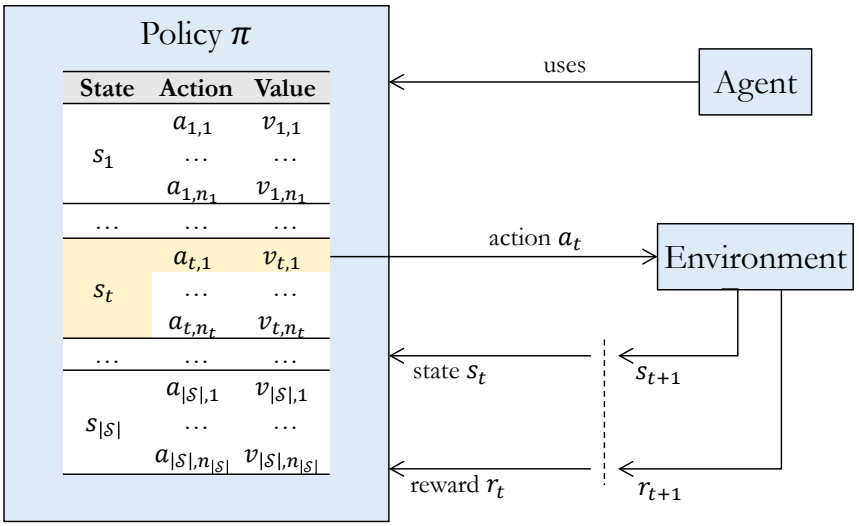


Figure 2.8: The reinforcement learning task

Reinforcement learning has attracted the attention of many fields since their origins in the 1950s. Techniques like multi-armed bandits (Lattimore and Szepesvári, 2020) or Markov decision processes (Sutton and Barto, 2018) have many uses, ranging from the development of marketing campaigns, the creation of automatic bots for playing games (Silver et al., 2017, Vinyals et al., 2019) or, more interestingly for our work, information retrieval (Glowacka, 2019) and recommender systems (Li et al., 2010, 2016, McInerney et al., 2018). In our work, we consider the use of a family of techniques for contact recommendation: multi-armed bandits. We explore them next.

2.4.1 Multi-armed bandits

Multi-armed bandits represent a family of state-less reinforcement learning techniques, on which the agent has to maximize some numerical signal by selecting one out of a fixed set of actions. This family takes its name from slot machines, which represent a paradigmatic application example for these techniques. Slot machines operated by a single lever (arm) are known as "one-armed bandits" (because they "steal" your money when you use them). If we have one with many levers and we have to earn some money, we have to choose the sequence of pulls that allows us to maximize our earnings.

Formally, a multi-armed bandit (Lattimore and Szepesvári, 2020) is a reinforcement learning algorithm that, on a discrete time step, has to select among k predefined actions $\mathcal{A} = \{a_1, ..., a_k\}$. The selection of an action $a \in \mathcal{A}$ yields a reward $r(a)$, and the goal of the approach is to maximize the obtained reward over time. For that, it estimates the value of the arms, considering the rewards obtained when the arm was pulled (i.e. when the action a was selected and performed). Several algorithms have been devised to solve this task, such as ε-greedy (Sutton and Barto, 2018), upper confidence bound bandits (Auer et al., 2002) or Thompson sampling (Chapelle and Li, 2011).

In the purest and most classical view of multi-armed bandits, the value of the different arms is just estimated by considering the previous rewards obtained for the different actions of the bandit. For instance, in the previous slot machine example, after playing some rounds, we have a first (although incomplete) impression on which lever produces the best return. However, it is possible to integrate additional knowledge about the different arms. For example, we might observe that one of the levers has been recently painted, leading us to believe that it has been more used by people, and, therefore, it might give us more benefits. This additional information is known as the context, and bandits which are able to consider it are known as contextual bandits (Li et al., 2010).

Multi-armed bandit techniques have many applications (Lattimore and Szepesvári, 2020): automatic A/B testing for determining which of two (or more) decisions is better for a website, learning the shortest path between two points in a network, dynamic pricing of products, allocation of scarce resources or optimizing the

amount of time needed to travel to work. IR and recommender systems have also benefited from the use of multi-armed bandits (Glowacka, 2019, Li et al., 2010, Zhao et al., 2013).

2.4.2 Reinforcement learning in recommender systems

Recommender systems have a cyclic nature, in which the system provides recommendations to its users, and the users return them feedback which can be used to improve the accuracy of the system. This feedback loop that recommender systems are exposed to can be modelled as a reinforcement learning problem, where the system plays the role of the agent, and the users the role of the environment. Because of this, the use of reinforcement learning techniques for recommendation has been increasingly attracting attention for a couple of decades (Cañamares et al., 2019, Li et al., 2010, 2016, Shani et al., 2005).

The foremost use of reinforcement learning techniques in recommendation is the development of new and effective algorithmic solutions. Although some of the proposed methods use RL approaches like Markov decision processes for recommendation (Shani et al., 2005, Xin et al., 2020), the most prolific group of techniques for the recommender systems field, is, by far, multi-armed bandits (Bresler et al., 2014, Kawale et al., 2015, Li et al., 2010, 2016, Zhao et al., 2013). In most of the work developed so far, the candidate items of the recommendation are modelled as the actions to take (as the arms). Although algorithms designed this way can only be used to recommend a single item at a time, it is possible to overcome this drawback by using multiple play bandits (Louëdec et al., 2015), which allow the selection of several arms at once. In addition to them, some other works have addressed the development of ensembles, by assigning a different recommendation algorithm to each arm (Cañamares et al., 2019). Focusing on the approaches which take the candidate items as arms, we differentiate two research lines.

The first research line considers the adaptation of previously existing collaborative filtering recommendation algorithms to a reinforcement learning setting. These recommenders merge collaborative filtering approaches with contextual bandit schemes. A first work in this line is the interactive collaborative filtering method by Zhao et al. (2013), who combine three multi-armed bandit algorithms (ε-greedy (Sutton and Barto, 2018), UCB (Auer et al., 2002) and Thompson sampling (Chapelle and Li, 2011)) with a probabilistic matrix factorization formulation (Salakhutdinov and Mnih, 2007). Instead of modelling the reward distribution, these approaches model the latent factors of the users in the matrix factorization scheme according to a stochastic bandit strategy. A similar method builds both the user and item factors in the probabilistic matrix factorization method using Thompson sampling (Kawale et al., 2015). Instead of building a simple model, it learns several factorizations at the same time, and, after collecting feedback, selects the most promising ones. An extension of this algorithm that considers dependency between the bandit arms is proposed by Wang et al. (2019a). Finally, Bresler et al. (2014) build a recommendation technique by combining the ε_t-greedy bandit algorithm (Auer et al., 2002) with a user-based nearest-neighbor strategy (Ning et al., 2015).

The second line considers the use of item properties or features for building linear contextual bandits. The first work in this research line is the LinUCB algorithm devised by Li et al. (2010), who determine the value of an arm (item) as the inner product of an unknown coefficient vector and a set of item features (represented as a vector). Following this line, Gentile et al. (2014) propose an approach, named CLUB (CLUster of Bandits) that divides the set of users in the system in several clusters according to their tastes, and uses the same estimation of the unknown coefficient vector for all users in the same cluster. This way, the algorithm can overcome user cold start situations by using the knowledge of the users in the same group. Finally, Li et al. (2016) further extend this scheme in their COFIBA policy (COllaborative FIltering BAndits) by also dividing items in clusters, so each item might have associated its own user partition.

In Chapter 10 we introduce a novel recommendation algorithm using multi-armed bandits. This method differs from previous work in how we model the task: we base our approach on a simple user-based kNN scheme (Ning et al., 2015), and, instead of modelling items as arms, we model the potential neighbors of the target user.

3
Contact recommendation

The contact recommendation task represents the main object of study of this thesis. To comprehend the scope and challenges of this specific recommendation task, we define, formalize and overview it in this chapter: we analyze the relation this problem has to other areas, the different algorithmic solutions which have been proposed in both academy and industry, and current challenges in the development and evaluation of the task. The present chapter provides a context for our research, and, as such, we also relate the contents introduced here to the rest of the thesis.

The work introduced here was partially published in

- **Sanz-Cruzado and Castells (2018a)**: Javier Sanz-Cruzado and Pablo Castells. Contact Recommendations in Social Networks. In Shlomo Berkovsky, Iván Cantador and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 519-569. World Scientific Publishing, Singapore, 2018.

3.1 The contact recommendation task

Contact recommendation functionalities in social networks represent a means to help people to enhance their connectivity in social media platforms. To achieve this goal, contact recommendation aims to identify three types of relations: those which a) already exist, b) would naturally form in the future or c) may form as a consequence of the recommendation. In the first two cases, these functionalities assist the users in transferring their real social environment to the online platform, whereas the last one is related to the discovery of new people (or, at least, people the user had forgotten about) the user might wish to connect with, like people with similar tastes or content creators like artists or photographers in networks like Instagram).

Also named as link recommendation (Li et al., 2017), contact recommendation is a particular case of recommendation with some salient and distinctive properties. Whereas in most domains users and items belong to different spaces, this task does not make that distinction, since both belong to the same set: the users in the social network. A second difference that arises is the meaning and representation of the rating matrix. In general item recommendation, we can represent the rating matrix as a bipartite graph, where there are only links between the users and the interacted items (elements in the same group are not connected). On the contrary, the rating matrix in the contact recommendation task represents the interactions between the users in the network, i.e. the adjacency matrix of the social network graph, which is no longer bipartite. Such interactions can be either explicit and stable social relationships (i.e. friendship, following, etc.) or implicit and punctual exchanges such as direct messaging, or indirect interactions on user created contents, such as forwarding such contents or mentioning other users in them (Guy, 2018). We illustrate this task in Figure 3.1.

This variation on the definition of the rating matrix modifies the categorization of contact recommendation approaches in terms of their input source, as the frontiers between some of the families vanish: as the ratings here are equivalent to the links between people in the network, collaborative filtering methods for this task are also social recommendation algorithms; and, depending on how content-based approaches are defined, we might just confound them with demographic methods (since we use profile information – either from the target or the candidate users).

3.2 Relation to other fields

The contact recommendation area has connections to many areas. If we look at this task from a broad perspective, we can locate it in the intersection between recommender systems and social network analysis. But, if we give it more fine-grained observation, we can establish connections with more specific concepts, such as social

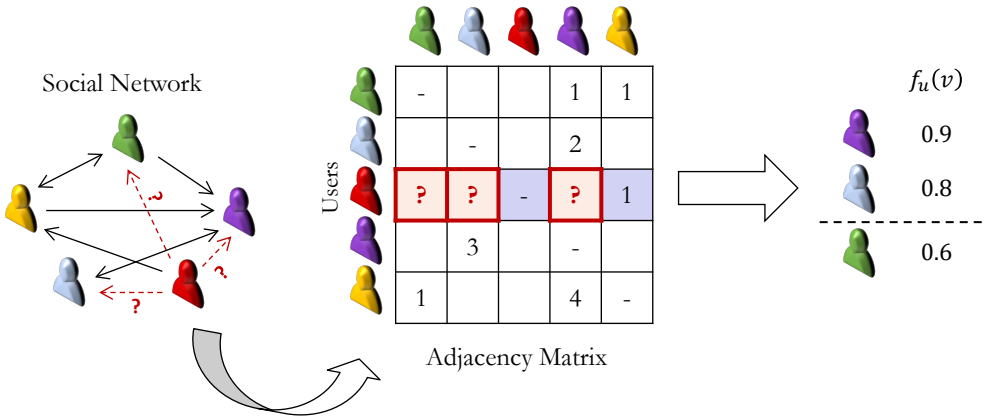


Figure 3.1: The contact recommendation task

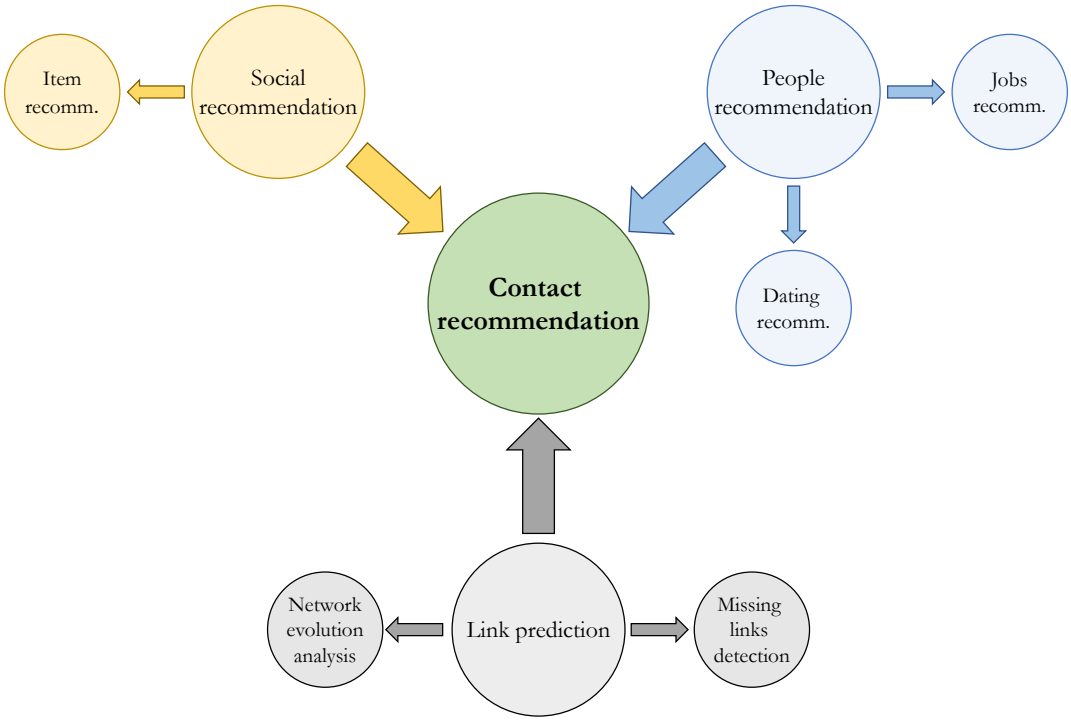


Figure 3.2: Relation between contact recommendation and other areas

recommendation, people recommendation and link prediction. In this section, we discuss these three areas and their relation to link recommendation. As a summary of our discussion, we illustrate the relation between the different areas in Figure 3.2, and we show a comparative of the main properties of each of these areas in Table 3.1.

3.2.1 People recommendation

We denote as people recommendation or social matching (Terveen and McDonald, 2005) the subset of recommender systems which have as an objective recommending people of interest. It is easy to see that contact recommendation represents a subset of this task – it is commonly referred to as people recommendation in social media (Guy, 2018). However, it is important to notice that people recommendation functionalities are not restricted to social network environments: beyond social networks, people recommendation has become the main functionality of some platforms, such as those for online dating (Diaz et al., 2010, Koprinska and Yacef, 2015, Neve and Palomares, 2019, Pizzato et al., 2010, 2013), job (Malinowski et al., 2006) or expert finding (McDonald and Ackerman, 2000).

Table 3.1: Specific properties of the related tasks

Task	Suggests	Assumes social network	Primary task type
Social recommendation	Items or people	Yes	Ranking
People recommendation	People	No	Ranking
Link prediction	Network links	Yes	Classification
Contact recommendation	People	Yes	Ranking

Online dating and job recommendation approaches can be categorized as reciprocal recommenders (Koprinska and Yacef, 2015, Pizzato et al., 2013): a subset of people recommendation approaches that require both the target and the candidate users to accept the recommendation in order to consider it successful. This is the same case we have in some contact recommendation problems – for example, when we recommend people in symmetric social networks as Facebook, both users have to accept the link – but these two cases show other properties not present when we suggest links in networks: first, a single user should not be recommended to a lot of users (a single user does not commonly date thousands of people or do hundreds of jobs) and, second, after a good recommendation, people might abandon the system forever (as the need the user wanted to satisfy is now covered).

3.2.2 Social recommendation

Similarly to how people can be recommended without a social network, we can also consider connections between users as an input for recommending other items such as movies, songs or user authored contents in social media (Guy, 2015, Tang et al., 2013). Independently to what we recommend (either people or items), as we introduce in Section 2.1.1, we refer to these systems that only use the connections in a social network and ratings as social recommenders. Sometimes, the connection between social item and user recommendation approaches is really close, since some approaches, like SoRec (Ma et al., 2008) can be used to provide both kinds of recommendations at the same time. However, not all contact recommendation algorithms can be considered social recommenders: in order to find interesting connections, we can also apply other families of approaches for the task such as content-based (Hannon et al., 2010) or demographic-based approaches (Quercia and Capra, 2009). Nonetheless, as we state before, all collaborative filtering link recommendations do belong to the social recommendation family.

3.2.3 Link prediction

Finally, link prediction represents a classic and tightly related problem to contact recommendation (Liben-Nowell and Kleinberg, 2007), and it is often considered as its precursor. The most clear difference between both tasks is that link prediction can be applied over other types of networks which are not necessarily social, such as biological networks (Cannistraci et al., 2013, Sulaimany et al., 2017) or transport networks (Zhu and Xia, 2016). Beyond this fact, differences between both tasks are subtle. Link prediction has been mainly oriented to find unobserved links that already exist or will form in the future (Liben-Nowell and Kleinberg, 2007), but there is nothing that prevents us from using the same techniques to identify users who might be useful for the people in the network (Barbieri et al., 2014) – in fact, many contact recommendation methods are just adaptations of link prediction methods –, so this distinction would feel artificial, and does not really change the problem nor how we design solutions for it.

Taking a broader perspective, we obtain more meaningful (although subtle) differences between both tasks. Contact recommendation is generally defined as a "local" ranking problem, where as link prediction is devised as a binary classification task. On the one hand, in the link prediction task, an algorithm would produce scores in a global scope, determining to what extent the links are likely or not to appear in the network. These scores provide a global ranking of links, which can be processed in several ways (establishing a threshold for differentiating the classification of the links, defining a number of links to predict, etc.)

On the other hand, contact recommendation produces a set of user rankings (containing the recommended people for each different user in the network). We can straightforwardly transform a global ranking into small independent rankings for each user, so any link prediction method can be applied to recommend contacts.

Table 3.2: Notation summary

Notation	Meaning
$\mathcal{G}$	Graph structure of the network
$\mathcal{U}$	Set of users in the network
E	Set of edges in the network
$\mathcal{U}_*^2$	Set of pairs of different users in the network
$\Gamma(u)$	Neighborhood of user $u \in \mathcal{U}$
$\Gamma_{in}(u)$	Incoming neighborhood of user $u \in \mathcal{U}$
$\Gamma_{out}(u)$	Outgoing neighborhood of user $u \in \mathcal{U}$
$\Gamma_{und}(u)$	Undirected neighborhood of user $u \in \mathcal{U}$
$w(u, v)$	Weight of the link $(u, v) \in E$
$w_{in}(u, v)$	Weight of v as in-neighbor of u
$w_{out}(u, v)$	Weight of v as out-neighbor of u
$w_{und}(u, v)$	Weight of v as both in-neighbor and out-neighbor of u
A	Adjacency matrix of the graph
$\hat{\Gamma}_{out}(u)$	Set of users recommended to u
$f_u(v)$	Ranking function for recommendation

However, this local perspective, where we independently produce recommendations to to each user, allows us to apply some transformations to the ranking function that are not appropriate for the global task. For instance, we can remove elements that act as constants because they only depend on the target user of the recommendation. Consequently, although we can see link prediction approaches as a subset of contact recommendation when we apply them over social networks, the opposite is not true.

3.3 Preliminaries and notation

In this section, we introduce the basic notation we use throughout all this thesis. First, considering that the structure of a social network can be defined as a graph, we denote it by $\mathcal{G} = \langle \mathcal{U}, E \rangle$, where $\mathcal{U}$ represents the set of users in the network and $E \subset \mathcal{U}_*^2$ expresses the relations that exist between them (friendship, interactions or whatever kind of relation that the social network is representing), where $\mathcal{U}_*^2 = \{(u, v) \in \mathcal{U}^2 | u \neq v\}$ is the set of pairs formed by different users.

Then, for each person in the network, $u \in \mathcal{U}$, we denote her neighborhood, i.e. the set of users u has connections with, by $\Gamma(u) \subset \mathcal{U}$. In networks where the direction of the edges is not considered, such as Facebook, this definition of neighborhood is complete. However, in directed ones, three different neighbors can be considered for each user, depending on which link orientation is considered: the incoming neighborhood $\Gamma_{in}(u) = \{v \in \mathcal{U} | (v, u) \in E\}$ takes as neighbors all the users who create links towards u; the outgoing neigborhood $\Gamma_{out}(u) = \{v \in \mathcal{U} | (u, v) \in E\}$ represents the set of users towards whom u has created links; finally, if both neighborhoods are combined, we obtain the undirected neighborhood $\Gamma_{und}(u) = \Gamma_{in}(u) \cup \Gamma_{out}(u)$, i.e. the neighborhood that u would have if the network was not directed.

When weighted networks are considered, there is an additional function we have to define: the weight function $w : \mathcal{U}_*^2 \to \mathbb{R}$, which returns the weight of an edge if $(u, v) \in E$ and 0 otherwise. In unweighted networks, we can also have this function if we consider that the weights are binary: $w(u, v) = 1$ if $(u, v) \in E$, and 0 otherwise. Following the same reasoning that has lead us to define three different neighborhoods for directed networks, we can similarly define three different weight functions, using the previous one as a basis. Taking the natural weight function $w(u, v)$ as it is leads to the weight v has for user u as a user in u's outgoing neighbor, which we denote by $w_{out}(u, v)$. We also denote the weight v has for user u as a user in u's incoming neighbor as $w_{in}(u, v) = w(v, u)$. Finally, we define the undirected weight, $w_{und}(u, v)$ as the sum of both: $w_{und}(u, v) = w(u, v) + w(v, u)$.

Once we have established the previous definitions and notations, we can also define the adjacency matrix of the network. This matrix, $A \in \mathbb{R}^{|\mathcal{U}| \times |\mathcal{U}|}$ represents the relations between the users in the network. Each cell, A_{uv}, represents the connection between nodes u and v, taking value $w(u, v)$ if the link (u, v) exists in the

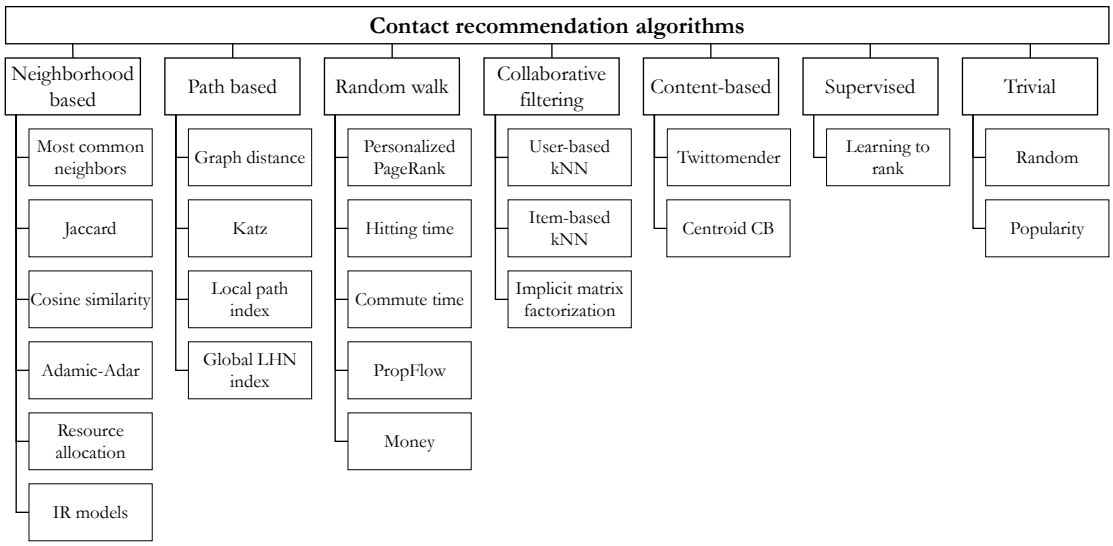


Figure 3.3: Taxonomy of contact recommendation algorithms

network ($(u, v) \in E$), or 0 otherwise. In undirected networks, this matrix is symmetric (i.e. $A_{uv} = A_{vu}$ for all $u, v \in \mathcal{U}$).

Using the previous notation, we can define the contact recommendation task. Given a target user $u \in \mathcal{U}$, contact recommendation's goal is to find a subset of users $\hat{\Gamma}_{out}(u)$ with two properties: first, that user u has no links towards them ($\hat{\Gamma}_{out}(u) \subset \mathcal{U} \setminus (\Gamma_{out}(u) \cup \{u\})$); second, that user u might be interested in creating new connections with them. This is typically addressed as a ranking problem, in which we find a limited number of users $n = |\hat{\Gamma}_{out}(u)|$ according to a ranking function $f_u : \mathcal{U} \setminus \Gamma_{out} \to \mathbb{R}$ which induces a (decreasing) order over the candidate users. The notations included in this section are summarized in Table 3.2.

3.4 Algorithms

From the point of view of recommender systems, we can understand contact recommendation as a standard problem, where we take the users as both users and items, and the adjacency matrix of the social network as the rating matrix. Considering this, it is possible to straightforwardly apply multiple recommendation approaches devised for other domains to suggest people in networks such as Twitter or Facebook. However, due to the particularities of social networks, this task has been undertaken by a wide selection of algorithms, originated in diverse fields, such as social network analysis (with tools like link prediction), machine learning (treating the problem as a classification one) or recommender systems. In this section, we review some of the most notable and relevant proposals. As a summary, Figure 3.3 categorizes the different algorithms we use in this thesis (either we explore them in detail in this section or not).

3.4.1 Trivial methods

Besides complex and heavily personalized recommendation, it is useful to have non-personalized, simple recommendation approaches to be used as rock-bottom, sanity check algorithms in an algorithmic comparison. In contact recommendation, it is simple to adapt some of these methods from general, domain-independent item recommendation scenarios. In particular, we explore two: random and popularity-based recommendation.

Random

Independently from any relevance signal or side information, the random recommendation algorithm sorts the candidate users by selecting as their recommendation score, a number uniformly chosen between 0 and 1.

$$f_u(v) = \text{random}(0, 1) \tag{3.1}$$

Random recommendation as it is does not make sense in production: as no evidence is taken to generate the recommendation, the accuracy of this algorithm is very low. However, it is useful to consider this approach for offline evaluation experiments. It serves to check for inconsistencies in the algorithms, as an algorithm performing below random is a symptom of an error or anomaly, and also provides a reference on how difficult the recommendation task is: to some extent, the accuracy of the recommendation is proportional to the density of the network, and it provides a lower bound for the accuracy of the datasets.

Popularity-based recommendation

Popularity-based recommendation ranks the candidate users by their degree in the social network. Applying it might also be understood as recommending contacts by the prior probability that a random user in the network follows the candidate user. Mathematically, we can define its ranking function as

$$f_u(v) = |\Gamma(v)| \tag{3.2}$$

Due to its simplicity, we might expect this approach to achieve poor performance for the recommendation task. However, it is widely used to recommend items in a wide range of applications, such as news sites (most read news), online stores (best-selling items) or social network platforms (i.e. trending topics in Twitter), and recent studies have shown that, although suboptimal, recommending the most popular items in a system provides a decent performance (Cañamares, 2019, Cremonesi et al., 2010). The reason for this is that, on average, users are most likely to consume those items that most people have previously consumed (Amatriain, 2013).

In contact recommendation, popularity-based recommendation is linked to the preferential attachment phenomenon (Barabási and Albert, 1999). This phenomenon establishes that nodes create new social relationships with other users with a probability proportional to their degree. Taking this into account, it is possible to define a link prediction algorithm that indicates that a link between a pair of nodes is more likely to appear when both users are highly connected (Liben-Nowell and Kleinberg, 2007, Newman, 2001). This is formulated as

$$f_u(v) = |\Gamma(u)||\Gamma(v)| \tag{3.3}$$

but, as the first term of the multiplication is the same for all the candidate items for u (i.e. $f_u(v) \propto |\Gamma(v)|$), we can remove it, to find the same definition as in equation (3.2).

3.4.2 Friends of friends methods

Also known as neighborhood-based methods (Sanz-Cruzado and Castells, 2018a) or local similarity indexes (Lü and Zhou, 2011), contact recommendation approaches within this category consider that the best strategy to recommend users in social networks is to focus on the close environment of the users involved in the recommendation. In the physical world, friends, relatives or coworkers sometimes introduce you to other people in their environment (for instance, their spouses, friends, business-related people, etc.) with whom you might be interested to befriend or be connected to in some manner. Friends of friends methods try to model that behavior, using the common neighbors between the target and candidate users as intermediaries for the recommendation. As this common users provide the only information source for this algorithms, friends of friends methods are only able to suggest people at distance 2 from the target user. Friends of friends recommenders are also strongly tied to the concept of triadic closure: if the network is undirected, a correctly predicted link using these methods creates new transitive triads. We review here the most representative approaches within this category.

Most common neighbors (MCN)

The most common neighbors (MCN) approach is the simplest friends of friends algorithms. It considers that the more friends you share with the candidate user, the more likely is that you end up creating a connection with him (Newman, 2001). Taking the joint probability that a neighborhood of the target user is a common neighborhood of the candidate user as a ranking function, we obtain the following formulation:

$$f_u(v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\mathcal{U}|(|\mathcal{U}| - 1)} \tag{3.4}$$

which can be simplified to counting the number of common neighbors between the target and the candidate users:

$$f_u(v) = |\Gamma(u) \cap \Gamma(v)| \tag{3.5}$$

In this simple method, we have the first glimpse of a problem that appears several times in the thesis: the neighborhood orientation selection. Equation (3.5) is completely defined when we work with undirected networks. However, in directed networks, where we do not have a single definition for the neighborhood of a user it is not: several neighborhoods can be selected for the target and candidate users.

As an illustrating example, we take here the work by Golder et al. (2009), who define several variants of this algorithm just by changing the directionality selected for the neighbors of the target and candidate user: first, they explore the possibility of recommending people at directed distance 2, i.e. taking the outgoing neighbourhood of the target user and the incoming one of the candidate user: $f_u(v) = |\Gamma_{out}(u) \cap \Gamma_{in}(v)|$. In addition, following the principle of homophily (McPherson et al., 2001), they also propose recommending people sharing interests with the target user (i.e. sharing followees, $f_u(v) = |\Gamma_{out}(u) \cap \Gamma_{out}(v)|$) or sharing audiences with the target user (i.e. sharing followers, $f_u(v) = |\Gamma_{in}(u) \cap \Gamma_{in}(v)|$. This problem is common to all the different friends of friends methods, and, as such, we discuss it later.

Jaccard similarity

As the probability of sharing neighbors with other people in the network increases with their degree, the MCN algorithm tends to favor highly connected and popular users. In networks with a skewed degree distribution, this might lead to MCN providing similar recommendations to popularity-based approaches. To mitigate this effect, several methods have been proposed, with the Jaccard similarity (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007) being the most popular of them. This approach takes as ranking function the probability that, if you randomly select a neighbor user of either the target or candidate user, it belongs to the neighborhoods of both users:

$$f_u(v) = \frac{|\Gamma(u) \cap \Gamma(v)|}{|\Gamma(u) \cup \Gamma(v)|} \tag{3.6}$$

Cosine similarity

Another method devised to prevent MCN from recommending high degree users is the cosine similarity (Salton et al., 1975), also known by some as Salton similarity (Lü and Zhou, 2011). This algorithm represents both target and candidate users in a $|\mathcal{U}|$-dimensional space, where each dimension represents a single user in the network. Defining the t-th coordinate of each vector as the weight of the link between t and the target/candidate user, $\vec{u}_t = w(u,t)$, this approach computes its similarity proportional to the cosine of both vectors:

$$f_u(v) = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|_2 \|\vec{v}\|_2} \propto \frac{\vec{u} \cdot \vec{v}}{\|\vec{v}\|_2} = \frac{w(u,t) \cdot w(v,t)}{\sqrt{\sum_{t \in \Gamma(v)} (w(v,t))^2}} \tag{3.7}$$

In addition to this approach and the Jaccard index, other algorithms such as the Sørensen coefficient (Lü and Zhou, 2011, Sørensen, 1948), the local Leicht-Holme-Newman (LHN) index (Leicht et al., 2006, Lü and Zhou, 2011) or the Hub Promoted and Hub Depressed indexes (Ravasz et al., 2002, Zhou et al., 2009) have been proposed for this. Due to their similarity with the these two approaches, we omit them here.

Adamic-Adar

Similarly to MCN, the Adamic-Adar algorithm (Adamic and Adar, 2003, Liben-Nowell and Kleinberg, 2007) recommends users with high overlap over the neighborhoods of the target and candidate users. Differently from that method, this algorithm gives more importance to those common friends with lower degree, as they are more "unique" to both relationship circles than popular people who have a lower discriminative power in that comparison. For example, if a user in Twitter is only followed by the target and candidate user, it seems more likely that they know each other than if they followed some highly popular account such as Barack Obama or Lady Gaga. The algorithm uses the following ranking function:

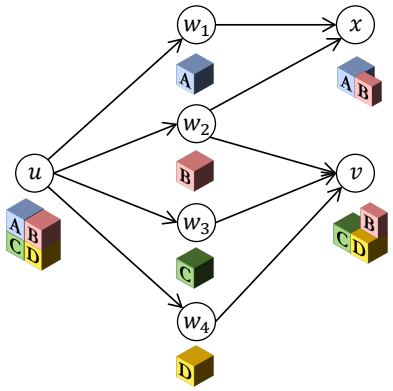


Figure 3.4: Resource allocation process

$$f_u(v) = \sum_{t \in \Gamma(u) \cap \Gamma(v)} \frac{1}{\log |\Gamma(t)|} \tag{3.8}$$

which just counts the common users neighbors, weighting them by the inverse of their popularity, dampened by a logarithm. Note that, in directed networks, similarly to how we can select different possibilities for the neighborhoods of the target and candidate users, we might also select three different possibilities for weighting the common neighbors ($\Gamma_{in}, \Gamma_{out}$ and Γ_{und}).

Resource allocation

This method is based on the so-called resource allocation physical processes, which involve the distribution of finite resources through social networks (Zhou et al., 2009). We illustrate this process in Figure 3.4. It works as follows: let u be a person with a finite amount of a resource, who wants to spread it through the network. As this person cannot copy the resource, the way she distributes it is by dividing it into equal parts, and giving each one to a neighbor, who repeats the process. After this two steps, the resource allocation method (Zhou et al., 2009) computes the fraction of the initial resource that reaches the candidate user. Then, this contact recommendation approach ranks the candidate users by the amount of resource that each candidate receives from u:

$$f_u(v) = \frac{1}{|\Gamma(u)|} \sum_{t \in \Gamma(u) \cap \Gamma(v)} \frac{1}{|\Gamma(t)|} \tag{3.9}$$

As the $1/|\Gamma(u)|$ term is the same for all the candidate users, we can omit it for implementation. This approach is quite similar to Adamic-Adar, but applying a heavier penalization to the number of common neighbors between the target and candidate user: thus, popular common neighbors will have even less importance for the resulting recommendation.

Other friends of friends algorithms

Besides the previously defined approaches, there are many other ways we can use the set of common neighbors between the target and the candidate users to provide relevant recommendations. For instance, link prediction proposes methods such as the individual attraction index or local Naive Bayes (Martínez et al., 2017). More interesting for our research is the adaptation of information retrieval models for the contact recommendation task (Hannon et al., 2010, Sanz-Cruzado and Castells, 2019a, Sanz-Cruzado et al., 2020a). We have not included descriptions of such algorithms here, because we explore the problem of adapting them in depth in Chapter 5. As a preview, if we only use the links in the network to generate the recommendations, we can use such IR models to define multiple friends of friends approaches.

3.4.3 Path-based methods

Taking a global perspective of the network allows algorithms to take advantage of several graph structural properties that are invisible at the local level, such as the paths between different nodes in the network. Here, we show a selection of the most relevant methods which use information about paths to improve the recommendations.

Graph distance

Following the definition of small world networks (Watts and Strogatz, 1998), where individuals are separated from others by a small number of steps in the network, and the fact that, in real networks, the distance between pairs of users tends to be reduced (Leskovec et al., 2007), this algorithm just takes the inverse of the length of the shortest path between the target and candidate users to generate the ranking function:

$$f_u(v) = -\delta(u, v) \tag{3.10}$$

where $\delta(u, v)$ denotes the shortest-path distance between nodes u and v.

Katz

First designed as a method for computing the status of a node in a social graph (Katz, 1953), and later adapted as a link prediction approach (Liben-Nowell and Kleinberg, 2007), this algorithm takes advantage of all the possible paths between the target and the candidate users. It seeks to identify the most densely connected and close people to the target user, by applying a weighted sum of the number of paths between him and the possible candidates. The importance of each path decreases exponentially as the length of the path increases, to give more importance to short paths. A general formulation for this approach is:

$$f_u(v) = \sum_{l=1}^{\infty} \beta^l \left|\text{paths}_l(u, v)\right| \tag{3.11}$$

where β represents the dampening factor for the path lengths (taking values in the $(0, 1)$ interval to ensure the convergence of the infinite sum), and $\text{paths}_l(u, v)$ represents the set of all the paths of length l between users u and v.

We can provide an exact formula for this, considering that the n-th power of the adjacency matrix of the network, A^n, contains, for each pair of nodes, the number of paths of length n between them. Using this, we can compute the sum as a geometric series, leading the following equation:

$$f_u(v) = \sum_{l=1}^{\infty} \beta^l A^l_{uv} = \left((I - \beta A)^{-1} - I\right)_{uv} \tag{3.12}$$

where I represents the unit matrix, such that $I_{uv} = 1$ if and only if $u = v$, and 0 otherwise. The inversion of the matrix can be computed by numerical networks, but it can also be approximated by adding some of the first terms in the sum (as the geometric series tends to zero very fast).

Local Path Index

Starting from the general formulation behind Katz (the one shown in equation (3.11)), it is possible to define another method that provides a trade-off between the accuracy and the complexity of the approach: the local path index approach (Lü et al., 2009). This method just ignores the paths between users and their neighbors (as, in contact recommendation, we are not recommending people we already know), and limits the length of the paths. In the original formulation (Lü et al., 2009), the limit for this was distance 3, but later, it was redefined to allow longer paths (Lü and Zhou, 2011). The expression for this index is the following:

$$f_u(v) = \sum_{l=2}^{n} \beta^l \left|\text{paths}_l(u, v)\right| \tag{3.13}$$

where n is the maximum length of the paths.

Global Leicht-Holme-Newman index

Similarly to Katz, the global Leicht-Holme-Newman (LHN) algorithm was originally created as a similarity measure between different vertices in networks (Leicht et al., 2006, Lü and Zhou, 2011). It considers that two vertices in a network are similar if their immediate neighbors in the network are similar. In order to find the formulation of this approach, we take as a basis the formulation of the Katz method in equation (3.12), but we give each term A^l_{uv} its own weight C^l_{uv}. This weight is defined as the inverse of the expected number of paths of length l between u and v in a configuration model (Newman, 2018) – a random graph created given the degrees of the different nodes:

$$\frac{1}{C^l_{uv}} = \mathbb{E}\left[A^l_{uv}\right] = \frac{|\Gamma(u)||\Gamma(v)|}{2|E|}\lambda_1^{l-1} \tag{3.14}$$

where λ_1 is the principal (greater) eigenvalue of the adjacency matrix. This leads, then, to the following expression for this algorithm:

$$f_u(v) = \frac{2|E|\lambda_1}{|\Gamma(u)||\Gamma(v)|}\left[\left(I - \frac{\phi}{\lambda_1}A\right)^{-1}\right]_{uv} \tag{3.15}$$

where $\phi \in (0, 1)$ is a free parameter controlling the decay of the similarity as the length of the path increases (the lower ϕ the fastest it decays).

3.4.4 Random walk methods

This family of contact recommendation methods considers that social interactions can be modeled as random walks. A random walk is a process that describes a succession of steps over the network by selecting the links to traverse at random (Newman, 2018). Establishing probabilities for transitioning from one node to another, these approaches use the probability that a random walker is at a certain node at a random point in time to generate the recommendation scores.

Personalized PageRank

PageRank (Brin and Page, 1998) is a random walk algorithm designed to estimate the importance of the nodes in a network according to its link structure. Given a node u in the network, its PageRank score represents the stationary probability that a "walker" who traverses the network randomly choosing the links to follow travels through that node.

The importance of a node in a network under this algorithm depends on three different factors: first, the number of incoming nodes; second, the importance of those nodes; and third, the number of outgoing edges from those nodes. As the node receives more and more connections from other users in the network, its importance increases. From each of these incoming edges, we receive an importance score, proportional to the importance of the link origin. And, finally, the received importance is decreased proportionally to the number of outgoing edges of that node. Considering all of this, we can recursively define the PageRank of a node as:

$$f_u(v) = \frac{r}{|\mathcal{U}|} + (1-r)\sum_{w\in\Gamma_{in}(v)}\frac{f_u(w)}{|\Gamma_{out}(w)|} + \frac{1-r}{|\mathcal{U}|}\sum_{w:|\Gamma_{out}(w)|=0} f_u(w) \tag{3.16}$$

where r is a parameter that establishes the probability that the random walker teleports to other node in the network randomly (ignoring the link structure). The rightmost term of the equation helps the algorithm handle the network sinks, so $\sum_w f_u(w) = 1$. The values of this approach are computed by repeatedly applying the expression above over all the users in the network until the values converge.

The resulting recommendation method is not personalized (we can easily observe that none of its terms depend on who the target user is). However, several ways to personalize the PageRank algorithm have been devised. From all of them, the most common (and the one we consider in this thesis) is the method proposed by White and Smyth (2003). The way to make this algorithm personalized consists in modifying the teleport probability. In the basic random walk method, every user can teleport to any node in the network with uniform probability. In the personalized version, instead of this, the walk is restarted by teleporting to the target user.

That way, the importance of a node does not only depend on the structure of the network, but also to how close the node is to the target user (as the walker is constantly returning to that node). The resulting formula is then:

$$f_u(v) = r\delta_{uv} + (1-r)\sum_{w\in\Gamma_{in}(v)}\frac{f_u(w)}{|\Gamma_{out}(w)|} + (1-r)\delta_{uv}\sum_{w:|\Gamma_{out}(w)|=0} f_u(w) \tag{3.17}$$

where δ_{xy} is the Kronecker delta function (taking $\delta_{xy} = 1$ if $x = y$ or $\delta_{xy} = 0$ otherwise). In link prediction, this personalized version of PageRank has received the name of rooted PageRank (Liben-Nowell and Kleinberg, 2007), as the target user acts as the "root" of the random walk.

Hitting time

The hitting time from a node u to a node v is the expected number of steps (if we consider that each step of a random walker takes one time unit, the expected time) required from a random walker starting from u to reach node v, H_{uv}. It is also known as mean first passage time, and defined as:

$$H_{uv} = \sum_{t=0}^{\infty} t\left(p_{u\to v}(t)\right) \tag{3.18}$$

where $p_{u\to v}(t)$ represents the probability of reaching v starting from node u in time equal to t. This hitting time function can be used to recommend people according to how easy they are to reach from the target user, using its inverse value:

$$f_u(v) = -H_{uv} \tag{3.19}$$

Depending on how the random walker traverses the network, there are several options for defining the hitting time algorithm. The most general approach, starting from the transition matrix of the random walk T – a matrix T which has as coordinate T_{uv} the probability of travelling from node u to node v –, defines the hitting time matrix for the random walker by the following equation (Meyer Jr., 1975):

$$H = \left[I - B^{+} + JB^{+}_{dg}\right]\Pi^{-1} \tag{3.20}$$

where $B = I - T$, B^{+} is the pseudoinverse matrix of B (Erdelyi, 1967), B_{dg} represents the matrix B with 0s out of the main diagonal, J is a matrix with all its entries equal to 1, and Π is a diagonal matrix whose entries represent the stationary probability of each node in the random walk. For example, we can take the same random walk as the one we use for PageRank, by defining its transition matrix,[1] and taking $\Pi_{uu} = p(u)$ where $p(u)$ is the PageRank value for node u. Further details can be found in (Meyer Jr., 1975).

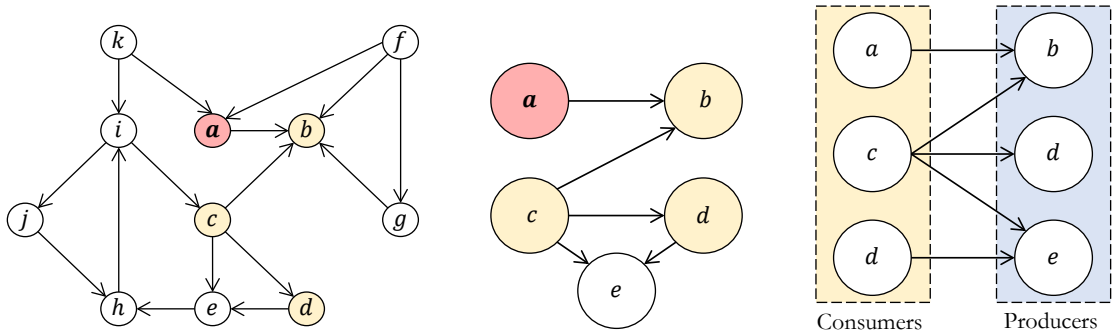
As long as the random walker can travel between each pair of nodes in the network, the previous algorithm for computing the hitting time can be applied, independently from the nature of the network (directed or undirected, weighted or unweighted, etc.). But this is not the only method which can be applied: in the particular case where we have a connected and undirected network, we can find an alternative definition considering that the random walker only traverses the network links. This method (Fouss et al., 2007) involves the pseudoinverse L^{+} of the Laplacian matrix $L = D - A$, where D represents a diagonal matrix with $D_{uu} = |\Gamma(u)|$. Using this approach, the hitting time is defined as:

$$H(u,v) = \sum_{t\in\mathcal{U}}\left(L^{+}_{ut} - L^{+}_{uv} - L^{+}_{vt} + L^{+}_{vv}\right)|\Gamma(t)| \tag{3.21}$$

Related to the mean first passage time, we can also compute another property of the random walks, known as the **commute time**, which is defined as the expected time for the random walker to travel from u to v, and returning back to u (Liben-Nowell and Kleinberg, 2007). If we denote the commute time as C_{uv} the ranking function is defined as:

$$f_u(v) = -C_{uv} = -\left(H_{uv} + H_{vu}\right) \tag{3.22}$$

[1] It is well known that the PageRank transition matrix (Langville and Meyer, 2006) can be defined as $T = rJ + (1-r)\overline{A}$, where $\overline{A}$ represents the adjacency matrix of the graph, where the rows have been normalized so its values sum 1, and, for each sink node, the probability of travelling to other node in the network is uniform.



(a) Original (full-sized) network (b) Reduced network (c) Bipartite network

Figure 3.5: Example of the bipartite network for the Money algorithm. Figure (a) shows the full graph and Figure (b) shows the reduced network with the users in the circle of trust for user a and their outgoing neighbors. In both figures, the target user, a, is highlighted in red, and the users in the circle of trust, b, c and d, highlighted in yellow. Finally, Figure (c) shows the definitive bipartite graph. Node b is not included in the consumer set since it does not have any outgoing links.

PropFlow

The PropFlow algorithm (Lichtenwalter et al., 2010) considers limited random walks that start on a node u and finish on another one v, or after a fixed number of steps l. The walk uses link weights as transition probabilities – or constant weights when working with unweighted networks, and terminates when it reaches v, it revisits a node (including the origin of the walk) or gives l steps. Then, the ranking function is defined as the probability of reaching the node v from u in l steps or less. These probabilities can be computed using a simple breadth-first search (BFS) on the network starting at u. Taking $f_u(u) = 1$, the procedure updates the probabilities of each one of the neighbour nodes the visited one in the BFS, x as

$$f_u(v) \leftarrow f_u(v) + f_u(x)\frac{w(x,v)}{\sum_{t\in\Gamma(x)} w(x,t)} \tag{3.23}$$

The PropFlow algorithm is thus similar to personalized PageRank, but, as it only has to compute a breadth first search with maximum depth l, it is cheaper to compute. Also, the depth restriction is intended to mitigate the potential noise that can appear in random walks when we travel to far regions of the network.

Money

Reported to be one of the main components of the Twitter contact recommendation system, "Who to follow" (Goel et al., 2015, Gupta et al., 2013), the Money algorithm exploits the link structure of the network using the random walk-based method known as SALSA (Stochastic Approach for Link Structure Analysis, Lempel and Moran (2001)).

This algorithm starts by building reduced bipartite networks for each user in the network to deal with the massive scale of online social networks and limit the influence of spam users in the recommendations. The set of sources of the bipartite graph, also known as the set of consumers or hubs, is formed by the so-called "circle of trust" of the network. For each target user of the recommendation, its circle of trust is selected by running an "egocentric" random walk algorithm, similar to personalized PageRank (Bahmani et al., 2010, Goel et al., 2015, Gupta et al., 2013). The circle of trust is comprised of the target user and the top k nodes in the network for that algorithm. Then, the set of destinations (known as the set of producers or authorities) includes every user in the network followed by the users in the circle of trust. We illustrate the construction process for this consumer-producer graph in Figure 3.5.

An interesting observation about this network is that, if the selected size k of the circle of trust is greater than (or equal to) the number of nodes in the social network, all users have the same bipartite network: the hub set would contain all the users in the network, and the authorities set would be comprised by all the nodes with in-degree greater than zero. Also, as only the consumer-producer graph is used for generating the recommendations, this algorithm can only recommend the users in the circle of trust or their followees.

Once this reduced network is built, a personalized version of the SALSA algorithm is run over it. The basic formulation of the SALSA algorithm defines two different scores for each node: a score as a hub, and another one as an authority in the reduced network. The transitions between nodes are defined using double steps: for computing the hub scores, the random walker takes a link forward (from the consumer set to the producer set) and another one backwards (back to the consumer set); for the authorities score, the opposite traversal of links is applied. At each step, if the random walker has d neighbors towards which he can transition, a uniform fraction of the score, $1/d$ is propagated to each of those nodes – because of this, the algorithm receives its name: as the money is a finite resource, it must be divided if anyone wants to give it to many people. Formally, the scores are computed as:

$$a(v) = \sum_{t \in \Gamma_{in}(v)} \frac{h(t)}{|\Gamma_{out}(t)|} = \sum_{t \in \Gamma_{in}(v)} \sum_{x \in \Gamma_{out}(t)} \frac{a(x)}{|\Gamma_{out}(t)||\Gamma_{in}(x)|} \tag{3.24}$$

$$h(v) = \sum_{t \in \Gamma_{out}(v)} \frac{a(t)}{|\Gamma_{in}(t)|} = \sum_{t \in \Gamma_{out}(v)} \sum_{x \in \Gamma_{in}(t)} \frac{h(x)}{|\Gamma_{in}(t)||\Gamma_{out}(x)|} \tag{3.25}$$

where $a(v)$ is the authority score and $h(v)$ is the hubs score. However, the initial formulation of the algorithm introduced in equation 3.24 has two main drawbacks: first, it is only personalized when the circles of trust are different for each user; second, it can be proved that the authorities and hubs scores are proportional to the in-degree and out-degree of the nodes, respectively – making them similar to the popularity-based recommendation (Lempel and Moran, 2001). Consequently, in the Money algorithm, a personalized version of the method, where a user-centered teleport vector is added to the hubs equation, is used. This approach is defined as:

$$f_u^a(v) = a_u(v) = \sum_{t \in \Gamma_{in}(v)} \frac{h_u(t)}{|\Gamma_{out}(t)|} \tag{3.26}$$

$$f_u^h(v) = h_u(v) = \alpha \delta_{uv} + (1 - \alpha) \sum_{t \in \Gamma_{out}(v)} \frac{a_u(t)}{|\Gamma_{in}(t)|} \tag{3.27}$$

Under this variant, the hub scores represent the similarity between the nodes in the circle of trust and the target user, and the authorities values provide a measure of the relevance of a producer to u (Goel et al., 2015, Gupta et al., 2013). Therefore, although the most sensible option for generating recommendations is the authority score, f_u^a, we can also take the hubs score f_u^h (Gupta et al., 2013) attending to the homophily principle (McPherson et al., 2001). In our experiments in this thesis, we observe that identifying which option is best for the recommendation might depend on the network we are working with.

Other random walk approaches

In addition to the methods introduced above, several other methods can be defined over random walks in the network, but are not explored in this thesis. We summarize here some of the most notorious approaches.

Similarly to how SALSA is applied over the consumer-producer graph in the Money algorithm, it is also possible to apply other random walk approaches over it (Goel et al., 2015, Gupta et al., 2013). In particular, Goel et al. (2015) proposes the use of another approach, Love, which, instead of SALSA, applies a personalized version of the HITS algorithm (Hypertext Induced Topic Selection, Kleinberg (1999)). However, this approach produces worst accuracy results than Money.

Another method, SimRank (Jeh and Widom, 2002), recommends similar users to the target one, following the intuition that two users are similar if they are followed by similar users. Due to the recursive nature of that similarity, we can interpret this algorithm in terms of random walks as the expected number of steps two random walks which traversed the edges backwards (one starting from the target user and another one starting from the candidate user) would need to meet at the same node.

Finally, Backstrom and Leskovec (2011) propose a method that combines supervised learning with random walks. The algorithm takes a weighted version of the personalized version, where the probability of travelling through a link is proportional to its weight. The weights for the random walk are learned by a supervised algorithm using demographic information, structural properties or link formation data.

3.4.5 Neighborhood-based collaborative filtering

Contact recommendation in social networks can also be defined as a classical recommendation problem. For this, users play both the role of the users and items in the recommendation, and the adjacency matrix of the network, A represents the rating matrix of the system. Considering this, it is straightforward to apply collaborative filtering methods for suggesting people to people. Although a brief description of the collaborative filtering approaches is provided in section 2.1, we do not provide there any formulation for the different methods we can use. Therefore, we now delve into them. We differentiate two families of methods: memory-based collaborative filtering and model-based approaches. We explore the first ones in this section, and the model-based ones in Section 3.4.6.

Memory-based algorithms, also known as the nearest neighbors (kNN) approaches, are based on the principle that similar users prefer similar items, and similar items are preferred by similar users (Ning et al., 2015). Taking the top k most similar users (or items) to the target user (or candidate items), these methods compute the recommendation scores as a linear combination of the ratings of those users to the candidate items (or the ratings of the target user for those items). Depending on which similarity we study, we differentiate two groups.

First, when we choose the set of closest users to the target user u, we talk about *user-based kNN* approaches. Denoting that set of users $N(u)$ as the neighborhood of the target user u – we note here that, for these algorithms the concept of neighbor is not related to the connections of node u in the network, but to their similarity –, user-based methods compute the score as a linear combination over the ratings of those users for the candidate items:

$$f_u(i) = \frac{1}{C} \sum_{t \in N(u)} \text{sim(u,t)} \cdot r_t(i) \tag{3.28}$$

where C is a normalization value, $\text{sim}(u,t)$ represents how similar users u and t are, and $r_t(i)$ is the rating user t gives to item i. In contact recommendation, since the "rating" of a user t for an "item" (which is, again, a user) v is A_{tv}, we can redefine this as:

$$f_u(v) = \frac{1}{C} \sum_{t \in N(u)} \text{sim}(u,t) A_{tv} = \frac{1}{C} \sum_{t \in N(u)} \text{sim}(u,t) w(t,v) \tag{3.29}$$

Second, when the similarity between the items in the system, we talk about *item-based kNN*. In this case, the neighborhood we take comprises the set of most similar items to the candidate item of the recommendation. A weighted linear combination of the ratings that the target user has provided for those neighbors is used for computing the recommendation score. Following the same mapping we use for the user-based variant, we get the following formulation:

$$f_u(v) = \frac{1}{C} \sum_{t \in N(v)} \text{sim}(v,t) A_{ut} = \frac{1}{C} \sum_{t \in N(v)} \text{sim}(v,t) w(u,t) \tag{3.30}$$

Several configuration options can be defined for both memory-based approaches, where the most important are the weight normalization and similarity selection. For more information about additional options, we refer to the work by Ning et al. (2015). When we choose a weight normalization score for a kNN approach, we have several possibilities, including not applying any normalization (or, equivalently, taking $C = 1$). If we decide to apply one, it is common to choose the sum of the absolute values of the similarities of the neighbors (i.e. $C = \sum_{t \in N(u)} |\text{sim}(u,t)|$), to deal with similarities which can take negative values.

About the similarities, there are many options that we can use here (Ning et al., 2015). In collaborative filtering, these similarities are defined in terms of the interactions between users and items. That way, user similarities depend on the ratings the users provide to the items, and item similarities, on the other hand, depend on the ratings that users have provided to the items to compare. A simple and effective similarity is the cosine similarity, which for user-based kNN, is defined as:

$$\text{sim}(u,t) = \frac{\sum_{\Gamma_{out}(u) \cap \Gamma_{out}(t)} w(u,x) w(t,x)}{\sqrt{\sum_{x' \in \Gamma_{out}(u)} \left(w(u,x')\right)^2} \sqrt{\sum_{x' \in \Gamma_{out}(t)} \left(w(t,x')\right)^2}} \tag{3.31}$$

For the item-based kNN, we would just inverse the direction of the links (we would take $\Gamma_{in}(u), \Gamma_{in}(t)$ instead of $\Gamma_{out}(u), \Gamma_{out}(t)$ and $w(x,u), w(x,t)$ instead of $w(u,x), w(t,x)$. We explore further possibilities for these similarities in Chapter 7, where we consider IR-based similarities for both user-based and item-based kNN schemes.

Besides user-based and item-based kNN, other neighborhood-based collaborative filtering algorithms have been proposed for people recommendation in social networks. An example is the SocialCollab algorithm (Cai et al., 2010), which combines both user-based and item-based schemes to deal with the reciprocity of relations in social networks.

3.4.6 Model-based collaborative filtering: matrix factorization

Matrix factorization approaches are model-based collaborative filtering approaches first devised in the 2000s, which factorize the user-item rating matrix with respect to a space of latent factors (Koren et al., 2009). Typically, the matrix is approximated as a product of two matrices: a matrix for the users, $X \in \mathbb{R}^{|\mathcal{U}| \times k}$ and another one for the items $Y \in \mathbb{R}^{|\mathcal{I}| \times k}$. Each row of these matrices represents a user or an item in a joint space of dimension k, smaller than the number of items or users in the system.

In the case of the items, each latent variable vaguely represents one of their characteristics. For example, in contact recommendation, the latent features might represent the age or political affiliation of the candidate users. In the case of the target users, the value of each variable measures to what extent the user is interested in candidates with high values on the given factor. In real applications, a real meaning for the item latent factors is rarely found, but the previous examples can be taken as an intuition about how these approaches work. Several matrix factorization approaches have been proposed for general recommendation (He et al., 2017, Koren et al., 2009, Salakhutdinov and Mnih, 2007, 2008, Sarwar et al., 2000).

Matrix factorization for implicit feedback

A particularly compelling approach for this thesis is the matrix factorization method devised by Hu et al. (2008) for dealing with systems that use implicit feedback to generate the recommendations, i.e. for systems where the user-item interactions might not be rating values entered by the users in the system, but records of interactions of the users with the items instead (for example, how many times a user has accessed the page of a game in sites like Steam, how many hours has played that game, etc.). As this approach does not consider explicit ratings, differently from other algorithms (Salakhutdinov and Mnih, 2007), its goal is not to predict the future ratings, but to predict whether the user will interact with the item or not.

To solve this problem, the authors find two values for each user-item pairs, depending on the implicit feedback: a preference value p_{ui}, which indicates if the user has ever interacted with the item in the past, and another one c_{ui} that represents the confidence level on that preference value. The confidence level is just a monotonous function, that grows as the value of the implicit feedback grows. In contact recommendation, following the original definition, we can define these two values as:

$$p_{uv} = \mathbb{1}_E(u,v) \tag{3.32}$$

$$c_{uv} = 1 + \alpha \cdot w(u,v) \tag{3.33}$$

where $\alpha > 0$ is a free parameter that establishes the importance of the weight, and $\mathbb{1}_E$ is the indicator function, that takes value 1 if $(u,v) \in E$, or 0 otherwise.

For contact recommendation, this approach factorizes the binarized adjacency matrix of the network using two separate matrices: a target user latent matrix $X \in \mathbb{R}^{k \times |\mathcal{U}|}$ and a candidate user latent matrix $Y \in \mathbb{R}^{k \times |\mathcal{U}|}$. Once these matrices are found, the recommendation score is found as the inner product of the u-th column of matrix X and the v-th column of the Y matrix:

$$f_u(v) = x_u^T \cdot y_v \tag{3.34}$$

In order to find the latent factors of the users and items, the algorithm devised by Hu et al. (2008) minimizes the following objective function:

$$\min_{x*,y*} \sum_{u,v} c_{uv} \left(p_{uv} - x_u^T y_v\right)^2 + \lambda \left(\sum_u \|x_u\|^2 + \sum_v \|y_v\|^2 \right) \tag{3.35}$$

where the $\lambda \left(\sum_u \|x_u\|^2 + \sum_v \|y_v\|^2 \right)$ term regularizes the model to prevent overfitting and λ is a free parameter, greater than 0. Through the rest of the thesis, we shall refer to this method as implicit matrix factorization (iMF).

Specific algorithms for contact recommendation and link prediction

In addition to the collaborative filtering approaches targeting the general recommendation task, their effectiveness and popularity has led to the development of matrix factorization techniques specifically designed to target the link prediction and contact recommendation tasks.

One of the most notable algorithms is the proposal by Menon and Elkan (2011), who proposed factorizing the adjacency matrix of the network as a function of the product of two matrices:

$$A \approx L(W^T \Lambda W) \tag{3.36}$$

where $W \in \mathbb{R}^{k \times |\mathcal{U}|}$ is a matrix containing the latent vectors for each user in the network (each column w_u represents a user u), $\Lambda \in \mathbb{R}^{k \times k}$ is a square matrix for handling directed networks (the identity matrix if we are considering undirected ones), and $L(\cdot)$ is a link function. The recommendation ranking score is then produced as:

$$f_u(v) = L(w_u^T \Lambda w_v) \tag{3.37}$$

To find such factorization, Menon and Elkan (2011) propose a method that aims at maximizing the area under the ROC curve (Fawcett, 2006).

Other matrix factorization approaches have been proposed to recommend items rather than users using social data (Jamali and Ester, 2010, Ma et al., 2008), but they can also be used to recommend contacts. For instance, the SoRec (Ma et al., 2008) algorithm simultaneously factorizes the network adjacency matrix and a user-item rating matrix as the products of two matrices. The approximation of the adjacency matrix can be then used to produce people to people recommendation in networks.

3.4.7 Content-based methods

We have thus far explored contact recommendation approaches that only deal with the link structure of the network to produce the recommendations. However, social networks provide plenty of extra information which can be exploited to produce recommendations. For instance, we can use the contents published by the users in the platforms, such as the tweets on Twitter or the posts in Tumblr (Hannon et al., 2010), information about the location of the users (Valverde-Rebaza et al., 2018) or the time they have spent together (Quercia and Capra, 2009). Similarly to how collaborative filtering methods devised for general item recommendation can be straightforwardly applied for people to people recommendation, we can state the same about content-based recommenders such as the Rocchio algorithm (Adomavicius and Tuzhilin, 2005, Rocchio, 1971).

Specific approaches have been designed to take advantage of this specific task. An algorithm that belongs to this family is the content-based variant of the **Twittomender** system which Hannon et al. (2010) propose for recommending users in Twitter. This algorithm represents each user in the network as the concatenation of a set of user-generated contents. Then, taking the IR vector space model (Salton and McGill, 1983) as a source of inspiration, the algorithm creates, for each user, a tf-idf vector, where each coordinate represents a different word. Then, it computes the cosine similarity between target user vector and the candidate user ones as the scoring function:

$$f_u(v) = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|_2 \|\vec{v}\|_2} \tag{3.38}$$

Hannon et al. (2010) propose four ways to select the set of contents of each user u: taking only the contents authored by u, contents authored by the incoming neighbors of u, by the outgoing ones and the union of all the previous contents. The coordinates of the $\vec{u}$ vector are defined using a tf-idf scheme:

$$u_t = \text{tf-idf}(t, u) = \text{tf}(t, u) \cdot \text{idf}(t) \tag{3.39}$$

where $\text{tf}(t, u)$ is a monotonically increasing function of the frequency of term t in the set of (concatenated) contents of the user, $\text{C}(u)$, and $\text{idf}(t)$ measures how specific is the term for the use – reducing its value when the term appears in more documents.

A similar algorithm, named as **centroid CB** (Sanz-Cruzado and Castells, 2018a,b) can be defined if, instead of first concatenating the users, and later computing the tf-idf vectors of those documents, we apply the opposite order: we first find the tf-idf vectors of each of the user authored contents, and then we find a centroid for each user in the network by aggregating the vectors of the corresponding contents representing each user:

$$\vec{u} = \sum_{d \in C(u)} \vec{d} \tag{3.40}$$

where

$$d_t = \text{tf-idf}(t, d) = \text{tf}(t, d) \cdot \text{idf}(t) \tag{3.41}$$

In this case, the scoring function is no different from the one we define in equation (3.38).

In both algorithms, we can apply any valid instantiation of the tf-idf scheme (Baeza-Yates and Ribeiro-Neto, 2011). For example:

$$\text{tf}(t, d) = 1 + \log_2 \text{freq}(t, d) \tag{3.42}$$

$$\text{idf}(t) = \log_2 \left(1 + \frac{|D|}{|\{d \in D | \text{freq}(t, d) > 0\}| + 1} \right) \tag{3.43}$$

where D is the set of text documents and $\text{freq}(t, d)$ is the frequency of the term in the document. In Twittomender, D is the set of users and $\text{freq}(t, d)$ is just the sum of the frequencies of term t in the contents representing user d. In centroid CB, D represents the set of user-generated contents, and $\text{freq}(t, d)$ just counts the number of appearances of the term t in the content d.

In the case of the Twittomender system, there is also a collaborative filtering variant that adapts the vector space model, using neighbor ids instead of contents. We explore this collaborative filtering algorithm in depth in Chapter 5 along with other approaches based on information retrieval models (Sanz-Cruzado et al., 2020a).

3.4.8 Supervised methods

All the previous link prediction and contact recommendation approaches have been designed as unsupervised methods. However, another perspective, known as supervised link prediction, takes techniques from supervised machine learning (ML) for carrying this task. Under this perspective, several methods such as Naive Bayes (Hasan et al., 2006, Lichtenwalter et al., 2010), support vector machines, multilayer perceptrons (Hasan et al., 2006) or random forests (Cukierski et al., 2011, Lichtenwalter et al., 2010). These approaches, instead of ranking the links, address the problem as a classification problem with two classes: the presence and absence of the link in the future. This particular set of algorithms has several specific issues which have to be addressed:

- **Pattern ranking:** The goal of a classifier is to determine which is the most likely class for an example (a pattern). In link prediction, finding a pattern for each one of the missing links in the network, this would be equivalent to estimating how likely a link is going to be present in the network or not.

  ML classifiers can be defined as algorithms, that, given a pattern, rank the different classes, according to how likely the pattern belongs to them. The recommendation task nature is quite the opposite: given a class (in contact recommendation, the "existing link") one, it ranks the links according to their membership likelihood.

  Nonetheless, transforming the classification task to the recommendation task is straightforward if the classifiers produce class scores. It suffices to sort the different links by their score for the class that indicates that the link will appear in the network.

- **Class imbalance:** In the most typical scenario, users in a network only establish connections with a small fraction of the people in a social network (Dunbar, 1993). This is reinforced by the upper limits that some networks establish for the number of relations created by a single user, to prevent some undesirable behaviours, such as link creations by spam bots (Myers et al., 2014), and, consequently, in networks with millions of users, this leads to very sparse networks. In supervised approaches, this might lead to an extreme imbalance problem, where a simple way of producing successful classifiers is just to consider that no new links are going to be added to the network. To overcome this problem, several proposals have been

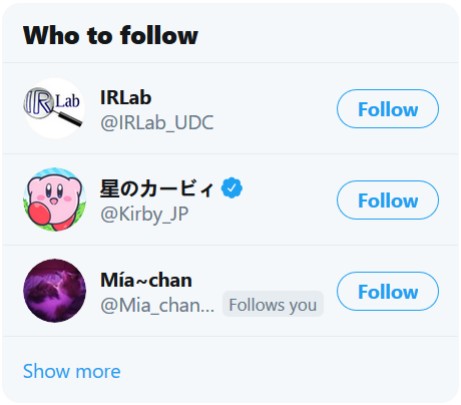


Figure 3.6: Example of contact recommendation in Twitter

done, such as undersampling the set of examples (Lichtenwalter et al., 2010), using approaches specifically tuned for imbalanced datasets (Hasan et al., 2006), or creating new artificial patterns of the minority class (Chawla et al., 2002, Lichtenwalter et al., 2010).

- **Feature selection:** A key aspect for obtaining good supervised approaches is the selection of a reduced and informative set of features for the classifier. Many possibilities can be explored, such as network analysis metrics (Hasan et al., 2006), features related to the contents published by the user in social networks (e.g. number of co-authored papers, keyword count, or number of common keywords in a citation network (Hasan et al., 2006)), or the output of unsupervised methods (Cukierski et al., 2011, Lichtenwalter et al., 2010).

3.4.9 Further algorithms

Some other algorithms have been proposed for contact recommendation. A group of approaches consider that the structure of the social network follows an underlying probabilistic model, and fit it as close as possible to the structure and properties of the network. Then, this model is used to predict links by computing the probability that the link is added to the network according to such model. Examples of this algorithms include the hierarchical structure model (Clauset et al., 2008) that assumes that social networks have an underlying hierarchical structure, or the stochastic block model (Guimerà and Sales-Pardo, 2009), which considers that nodes are divided into several partitions, and the probability of a link only depends on those groups. Another family of algorithms includes hybrid recommendation methods. An example within this family is the work by Barbieri et al. (2014), who propose a stochastic Bayesian model combining information about the social network structure with evidence about the interest of the users on different topics. Finally, late link prediction literature has focused on the development of deep learning techniques for building node embeddings (Grover and Leskovec, 2016, Meng et al., 2019).

3.5 Contact recommendation in industry

Since the late 2000s, the main social networking platforms such as Facebook, LinkedIn or Twitter have provided contact recommendation services on their platforms. An example of contact recommendation in Twitter can be observed in Figure 3.6. Although public knowledge about their algorithms and specifications is limited, the different companies have disclosed some internal details about them.

Facebook: The "People You May Know" algorithm that Facebook uses[2] has been identified as a hybrid algorithm, that uses many signals to provide the recommendation. The most important signal is the existence of common friends between the target and candidate users, but other information such as belonging to common

[2] About Facebook PYMK: https://www.facebook.com/help/336320879782850?locale=en_US (Accessed 19th December 2020)

groups or affiliations (using profile information such as school, university or work) or being tagged in the same photos or contents is also used to recommend people in Facebook.

LinkedIn: The contact recommendation algorithm in LinkedIn (also known as "People You May Know"[3]) uses commonalities between users to suggest new connections. These commonalities include shared contacts, similar profile information and experiences, working at the same companies or industries or attending the same schools.

Tumblr: This platform recommends new blogs (the equivalent of users in this network) by mixing the topics that the user prefers and the network structure (Aiello and Barbieri, 2017). When they first sign in, the platform asks the users about their preferences for a set of predetermined topics. This information is used to create a topical profile for each user. This profile is used to deal with the cold start problem until their number of contacts grows. Then, it gives more importance to recommending new blogs using the triad closure principle (i.e. friends of friends algorithms).

Twitter: Differently from the rest of social networking sites, Twitter has disclosed some inner details of their "Who to Follow" system (Goel et al., 2015, Gupta et al., 2013). The algorithm originally stated to be applied for recommending people in this network is the Money approach we introduce in Section 3.4.4. In addition to this, Twitter has also reported experiments with Love, personalized PageRank and most common neighbors approaches, variants of the cosine similarity and the closure algorithm, which recommends followers of the target user who lay at distance two of him. In recent years, Twitter has changed its recommendation algorithm to an undisclosed engagement prediction model (Satuluri et al., 2020) using several features.

3.6 Practical aspects

When a contact recommendation system is built, there are some aspects that we have to consider, such as how they should be evaluated, the directionality of the edges, the scalability of the different algorithms or how these approaches might affect the network and reinforce some undesirable properties. In this section, we provide an overlook of these practical aspects, which we cover through this thesis.

3.6.1 Edge orientation

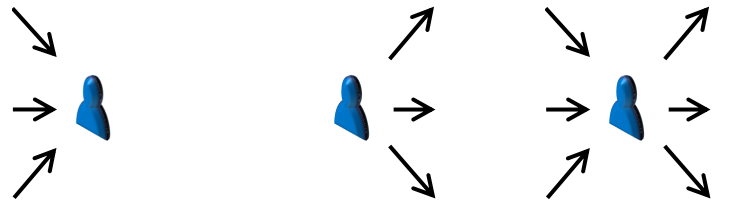
In their beginning, the development of link prediction and contact recommendation approaches was oriented to undirected networks. However, the appearance and success of asymmetrical online social networks such as Twitter, Tumblr or Instagram motivates bringing more attention to directed structures. In many cases, the adaptation of people recommenders from the undirected cases is straightforward, and does not require any modification – just operating on the asymmetric adjacency matrix. This is the case of approaches like Katz or Local path index. In other cases, the algorithm establishes which are the orientations of the edges that we should consider, as it happens with personalized PageRank.

However, some methods, like the friends of friends algorithms, admit different configurations depending on the definition of neighborhood that we use. As we illustrate in Figure 3.7, in directed networks, for each user u, we have three different definitions for the neighborhood: incoming, $\Gamma_{in}(u) = \{v \in \mathcal{U} | (v,u) \in E\}$, outgoing, $\Gamma_{out}(u) = \{v \in \mathcal{U} | (u,v) \in E\}$ and undirected, $\Gamma_{und}(u) = \{v \in \mathcal{U} | (u,v) \in E \vee (v,u) \in E\}$. Choosing one orientation or another in those algorithms might lead to different performances of the algorithms.

In previous works, Golder et al. (2009) propose the use of different neighborhood selections for the target and candidate users in the most common neighbors algorithm. Later, Hannon et al. (2010) represented the users in the network selecting a just one of these orientations for both the target and candidate users in their tf-idf algorithm.

As this is still an open question, we explore it in depth in Chapter 5, where we compare the effectiveness of variants of friends of friends approaches for which we only vary the orientation selection for the target and candidate users.

[3] About LinkedIn PYMK: https://www.linkedin.com/help/linkedin/answer/29/people-you-may-know-featureoverview (Accessed 19th December 2020)



(a) Incoming neighborhood Γ_{in} (b) Outgoing neighborhood Γ_{out} (c) Undirected neighborhood Γ_{und}

Figure 3.7: Possible neighborhood orientations.

3.6.2 Scalability

The massive scale of the current social media platforms, with hundreds of millions of monthly active users (Myers et al., 2014) make producing people to people recommendations a difficult challenge in terms of both time and memory resources. When we talk about contact recommendation, the costs are usually dependent on the amount of users in the social network. For instance, if we want to consider any missing link in the network as a potential contact, we need to compute $|\mathcal{U}|(|\mathcal{U}|-1)-|E|$ values to recommend users to everyone in the network. As social networks are very sparse (Hill and Dunbar, 2003, Myers et al., 2014), the cost of computing all those scores is, at the least, quadratic on the number of users. Even for a single user, we would need to compute billions of scores. Because of this, it is not uncommon to observe several mechanisms oriented to dealing with this problem.

The own nature of several algorithms makes them avoid this bottleneck. This is the case of the friends of friends approaches, such as MCN, Jaccard or Adamic-Adar, who only consider candidate users that share, at least, one neighbor with the target user of the recommendation. In an efficient implementation (we see how we can address this in Chapter 5), the cost of computing all these scores can be reduced to be quadratic on the average number of neighbors of the users in the network (a very small number in comparison to the number of users). Following the notion of small world networks (Milgram, 1967, Watts and Strogatz, 1998), where all the users are close to each other, other algorithms use their parameter selection to deal with this problem, just by limiting the distance between the target user and the candidates. This is the case of the PropFlow algorithm (Lichtenwalter et al., 2010) or the local path index (Lü et al., 2009). For instance, Lü et al. (2009) only take connections up to distance 3 in the definition of their local path index algorithm. In addition to this, other methods define more sophisticated ways to deal with the scalability issues as the Twitter "Who to Follow" system, where Goel et al. (2015) define a reduced consumer-producer graph using random walks to later apply a costly algorithm over the reduced network.

Scalability has not only affected the development of new link prediction and contact recommendation approaches, but also how they are evaluated – particularly, how evaluation experiments are carried in offline settings. Due to the high computational cost of some algorithms, computing some metrics such as AUC for all the $|\mathcal{U}|(|\mathcal{U}|-1)-|E|$ in large networks is highly challenging – we would not only need to compute all those values, but also sort them, which takes $O(N \log N)$ cost, being N the number of evaluated links, and afterwards, compute the metric, which can be found in linear time. Several works have addressed this problem by subsampling the test set for the experiments (Grover and Leskovec, 2016, Meng et al., 2019): instead of considering all possible pairs, they only generate scores for the new links in the test set, and a randomly selected set of negative examples, and evaluate over them.

3.6.3 Effects on the networks

Similarly to how the accuracy has been the main target of general recommendation, the same has occurred with the research and development of link prediction and contact recommendation techniques. In this context, maximizing the accuracy means either maximizing the amount of recommended links that were already present but unobserved in the social network (Zhou et al., 2009) or are accepted by the target users because they find them useful (Chen et al., 2009). However, the formation of a link has effects in their surroundings, so contact recommendation might lead to substantial changes on the properties of the network. For example, different

network structures change how rumors are propagated trough networks (Doerr et al., 2011, 2012). Thus, in the last few years, several works have studied the effects these algorithms might have on the different properties of the network.

Among the first works in this line, Daly et al. (2010) bring attention to the effects recommendations have on the local and global structural properties of the network, such as the degree distribution, its skewness, node betweenness and triadic closure on the IBM SocialBlue network. Later, Su et al. (2016) analyze how the Twitter "Who to Follow" service affects the growth of Twitter, finding a popularity reinforcement effect and an increase on the triadic closure. Aiello and Barbieri (2017) consider the topological impact of the people to people recommendation functionalities of Flickr and Tumblr at a local ego-network level, targeting the popularity of the linked nodes, the creation of different communities of users, the variation of the diameter. Afterwards they take the overlap between recommendations as a diversity measure. Huang et al. (2013) speculate on the interest of providing structurally diverse recommendations. They carry a study on the LinkedIn "People You May Know" and find that, using simple topological diversity metrics such as the number of recommended components and triangles, despite losing some accuracy, increasing those metrics lead to a better user engagement. Taking a different perspective, Stoica et al. (2018) study the potential reinforcement on the glass ceiling effect in social networks as a consequence of recommendation, and suggest a way to counter it.

Beyond evaluating the effects of recommendation, other works use contact recommendation to enhance several properties of the network. As examples, Parotsidis et al. (2016) optimize the recommendation ranking to minimize the expected path length between the target user and the rest of the network, whereas Corò et al. (2019) maximize the number of active nodes in diffusion processes.

All the studies above contextualize the research we present in part III. In that part, in Chapter 8 we first explore the definition of new metrics for studying the effects contact recommendation approaches might have on the structure of the network, providing a wider view than the works by Daly et al. (2010) or Su et al. (2016). Later, in Chapter 9, in the line of Parotsidis et al. (2016) we propose a method for enhancing the structural diversity of rankings, and we later explore which effect this enhancement has on the information diffusion on networks.

4
Preliminary experiments

Highlights

- *We use a Facebook undirected network, and two Twitter datasets, each with two networks: a dynamic weighted interaction network and an unweighted follows network.*
- *We split the data using temporal partitions when possible, randomly when it is not.*
- *Collaborative filtering methods – matrix factorization and nearest-neighbors algorithms – are very effective for contact recommendation.*

As a first look to how contact recommendation in social network works, we run some preliminary experiments, comparing the effectiveness of several state of the art approaches (first introduced in Chapter 3) in terms of accuracy metrics. As the methodology we apply in our experiments through this thesis are similar, we first introduce the general experimental setup of our offline experiments: data collection and cleaning, training / test partitioning, evaluation methodology, etc. Then, we provide the experimental comparison of contact recommendation approaches over Facebook and Twitter networks.

The contents of this chapter were partially published in

- **Sanz-Cruzado and Castells (2018a)**: Javier Sanz-Cruzado and Pablo Castells. Contact Recommendations in Social Networks. In Shlomo Berkovsky, Iván Cantador and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 519-569. World Scientific Publishing, Singapore, 2018.

4.1 Data

In this thesis, we run our experiments over a diverse set of networks with different properties (directed, undirected, weighted, unweighted, etc.). These networks are samples extracted from two of the largest social networking sites: Facebook and Twitter. We first provide a brief description of both platforms, and then, we proceed to describe the collection procedures and the properties of the collected data.

4.1.1 Platform description

We first provide a brief description of the social media platforms we extract the datasets from, as well as the commonly used terms in those platforms, since we shall use them in the remaining chapters of these thesis.

Facebook

Facebook, funded[1] in 2004, is the most active and well-known social media platform on the Web, with up to 1.82 billion active daily users and 2.74 billion active monthly users as of September 2020[2]. The main goal of Facebook is to help users transfer their actual social relations (friendship, family, colleagues, etc.) to the online world, and keep in touch with them. This is reflected on the underlying social network, an undirected network where both endpoints of a link have to accept its creation before it is added to the network, and on the fact that neighbors are referred to as "friends" in the platform.

Beyond the creation of such connections, Facebook offers other functionalities, that allow people to interact to each other or share information. As other means to establish connections, Facebook allows users to

[1] Facebook: https://www.facebook.com (Accessed 19th December 2020)

[2] Facebook Q3 2020 Report: https://investor.fb.com/investor-news/press-release-details/2020/Facebook-Reports-Third-Quarter-2020-Results/default.aspx (Accessed 19th December 2020)

Figure 4.1: Example of a tweet with a hashtag.

Figure 4.2: Example of a retweet

gather into groups (with people sharing similar interests) or following individual pages. In addition, users can generate new contents and share text, photos, videos, etc. with their friends. The contents created by a user in the platform, known as posts, are then shown in the news feed of her friends – which shows a selection of recent publications of the friends of the user, the groups she belongs to or the pages she follows. Users can interact with these contents by commenting them, reacting to them (for example, liking or disliking them) or sharing the post with their friends.

In this thesis, we just use friendship information collected from the Facebook social graph. We take a single dataset, collected by McAuley and Leskovec (2012), which contains a reduced sample of the network in 2012. The Facebook network is a representative of undirected and unweighted networks in our experiments.

Twitter

Twitter[3] is a representative of microblogging platforms. Released in 2006, it has grown up to have 185 million daily active users as of September 2020[4]. As a social media platform, the creation of connections between different users is important, but the creation and consumption of user-generated contents is (at the very least) as vital for its success (Myers et al., 2014). User-generated contents come in the form of short messages (known in this platform as tweets) with up to 280 characters. These messages can contain multiple types of content, ranging from text to multimedia contents (videos, pictures, etc) or URLs and can be categorized using tags, which, in this system, are referred to as hashtags, since they all start with the '#' character. We show an example of a tweet with a hashtag in Figure 4.1.

As a site focused on the creation and consumption of these tweets, the set of users and their connections in the platform can be described using an asymmetric social networks where users follow others. Then, each time a user publishes a tweet, it appears in her incomming neighbors' (her follower's) timelines – a collection of the

[3]Twitter: https://twitter.com/ (Accessed 19th December 2020)

[4]Twitter Q3 2020 Shareholders letter: https://s22.q4cdn.com/826641620/files/doc_financials/2020/q3/Q3-2020-Shareholder-Letter.pdf (Accessed 19th December 2020)

Figure 4.3: Example of a mention within a tweet.

tweets published by the followees (the outgoing neighbors) of the user, sorted by an algorithm which combines the recency of the tweets with their relevance for the user. The platform also allows interactions between users. Most of such interactions make use of the tweets, as it is the case of retweets, mentions and replies. Retweets are the way people forward tweets in Twitter. A retweet appears as a new tweet in the user's profile (and her followers' timelines), but the authorship of the content is assigned to the original creator. We can see an example in Figure 4.2. A mention, as its name indicates, is a reference to other user in the network inside a tweet. People can mention each other by adding the '@' character before the nickname of the user in the platform (as shown in Figure 4.3). Finally, a reply is just an answer to the contents of other tweet. In addition to such interactions, people can also send each other direct private messages as well as like their tweets or answer polls embedded in the tweets' content.

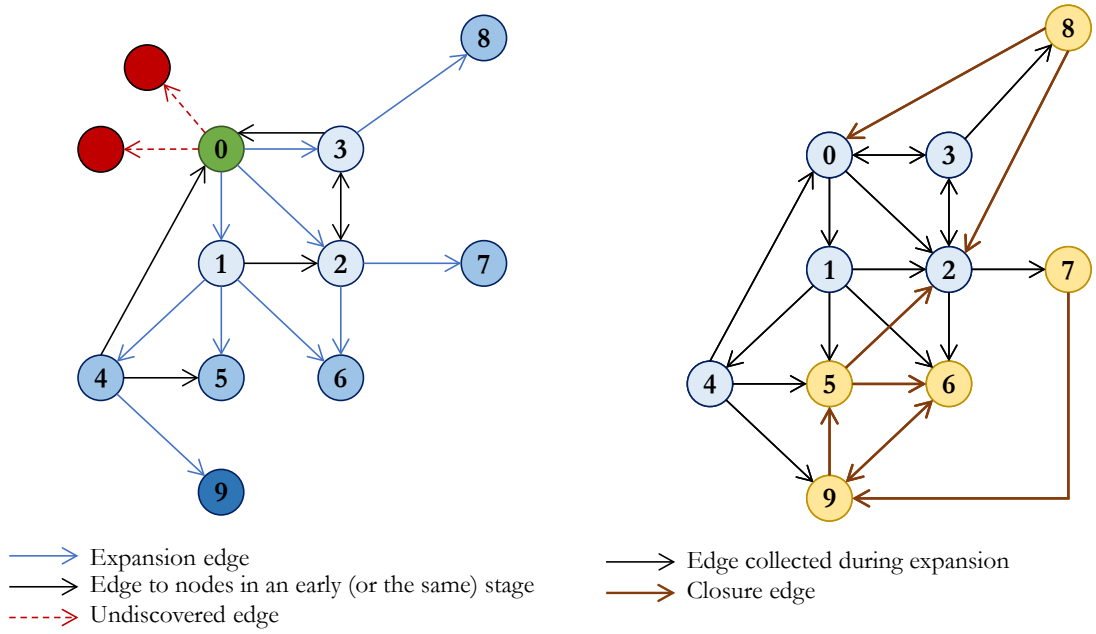
In this thesis, we collect two different Twitter datasets, each with two social networks: first, an explicit follows network, where there is a link between two nodes u and v if u follows v in the Twitter network, representing directed and unweighted networks and second, an implicit interaction network, where the (u, v) link exists if u has retweeted or mentioned v as a directed and weighted network (where weights are just the frequency of interaction). Tweets are used as information for content-based approaches, and hashtags as topical information about those tweets.

4.1.2 Data collection

First, we identify the mechanisms we have used to obtain the data from the two social networks. The Facebook data we use in our experiments is taken from the Stanford Large Network Dataset collection.[5] In particular, we use the ego-Facebook dataset. For obtaining such network, McAuley and Leskovec (2012) collected the ego networks of ten different users, i.e. a sample of the network containing one of those users, all his friends and all the connections among them. The resulting network is just the union of all these ego networks.

For our Twitter data, we use two types of networks: dynamic, implicit networks, induced by the interactions between users (i.e. $(u, v) \in E$ if u retweeted or mentioned v) and explicit networks, formed by static follow links (i.e. $(u, v) \in E$ if u follows v). To retrieve them, we download data from the platform using the Twitter API[6]. The Twitter API provides access to most of the Twitter functionalities: retrieving information from user timelines, interactions and following relations between users, and also sending private messages, post-

[5] Stanford Large Network Dataset: https://snap.stanford.edu/data (Accessed 19th December 2020)
[6] Twitter API: https://developer.twitter.com/en (Accessed 19th December 2020)



(a) Expansion phase (b) Closure phase

Figure 4.4: Example of the sampling procedure for 10 users.

ing new tweets, etc. However, the usage limits of the API make difficult to collect a large amount of information (Wang et al., 2015).

In order to retrieve the greatest amount of information we can from the API, we start by collecting the interactions network. Once that network is collected, we just retrieve the follows network by finding the explicit relations between the crawled users. Many ways have been proposed for sampling information from large social networks (Das et al., 2008, Leskovec and Faloutsos, 2006, Papagelis et al., 2013), but, as an efficient method that we can apply for collecting as much information as possible from the Twitter API, we use a snowball sampling approach (Goodman, 1961): we first start with a single seed, and we collect the interactions (mentions and retweets) of this user in a selection of tweets of the user. This interactions are taken as outgoing edges to be traversed. The set of users retrieved with those interactions form the first level of the sample. The algorithm repeats the algorithm for each node in this first level to retrieve users in the second level, and so on until a fixed number of users are reached. At that point, we run over the nodes in the crawling frontier and we retrieve any outgoing edges going from them to the sampled users.

We illustrate this process in Figure 4.4: Figure 4.4a shows how the selection of nodes is done: the green node represents the single seed, the blue ones the nodes we sample and the red nodes represent unreachable nodes (nodes the seed user has interacted with, but not in the retrieved set of tweets). Depending on the intensity of the color, we identify several sampling levels: light blue nodes represent the first level, intermediate blue nodes represent the second level and dark blue the third and last level. As we only want 10 users, the procedure ends after visiting the 4th user. Then, in Figure 4.4b, we identify the frontier of the crawler (highlighted in yellow) and we identify the links from them to the rest of users in the sample.

Using the previous procedure, we have built two different datasets: one containing all tweets posted by a set of around 10,000 users from June 16^{th} to July 16^{th} 2015, and one containing the last 200 tweets (the maximum number of tweets that can be collected in a single Twitter API call) posted by 10,000 users as of August 2^{nd} 2015. We denote the first dataset as *"1 month"* whereas the second is denoted as *"200 tweets"*.

4.1.3 Data partitioning and cleaning

As we run offline experiments, it is common under such methodology, to use a partition of the data we collect (Cañamares et al., 2020, Gunawardana and Shani, 2015): at least, we need two separate sets: a training set to use as input for the recommendation approaches and a test set that acts as a ground truth for evaluating the recommendation algorithms. In addition, we can also have a validation set for tuning the hyperparameters of

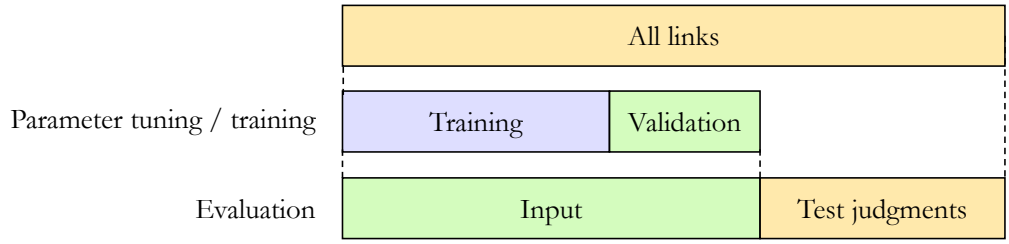


Figure 4.5: The random and temporal network partitioning approaches.

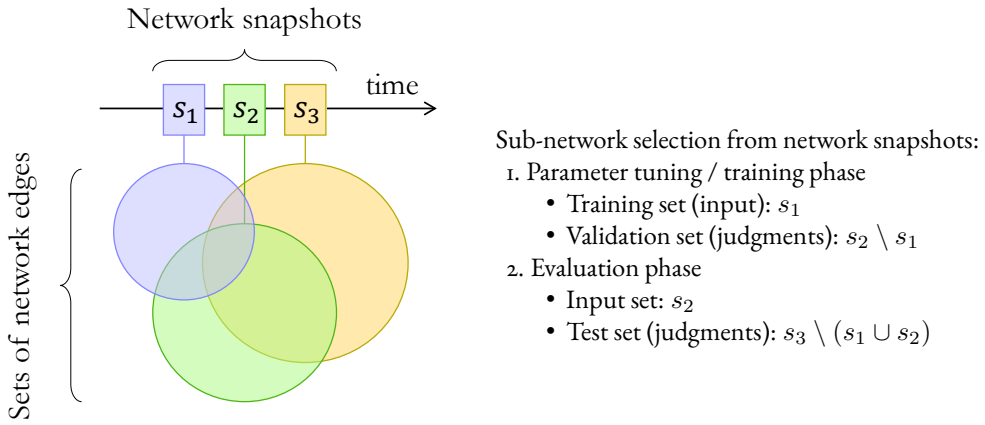


Figure 4.6: The snapshot-based network partitioning approach. The procedure is based on network growth over time, but also considers that some edges may disappear (i.e. we do not necessarily have $s_1 \subset s_2 \subset s_3$).

the algorithms. Therefore, in our experiments, given a network dataset (or data source) we obtain three different subnetworks (disjoint sets of edges): a *training*, a *validation* and a *test* network. We use them as follows: first, we use the training set as input and the edges in the validation set as relevance judgments for configuring the hyperparameters of the different algorithms. Then, each algorithm (with its optimal configuration) takes the union of the training and validation networks as input (from now on, we will refer to this union as the *input* set). The test graph is held out from the algorithms, and it is used for evaluation purposes.

To obtain these disjoint networks, we take three different approaches: random split, temporal split and temporal snapshots:

- **Random split:**Taken a single network as starting point, we select the edges for each subnetwork uniformly at random, based on the desired percentage of the edges we want on each set.
- **Temporal split:** We start from a single network and we split the set of edges in three different sets, ensuring that the timestamps of the training set precede the validation set, and that the latter also precedes the test set. If a link appears in more than one set (for example, when a two users interact with each other several times across a split point), it is removed from the most recent one, to avoid "contaminating" the test with training data.
- **Temporal snapshots:** Instead of dividing a network, we obtain three different snapshots of the network at three consecutive time points. The first snapshot is used as the training graph; the new links in the second snapshot (i.e. the links that appear in the second snapshot but not in the first one) are taken as validation; and the new edges in the third download (i.e. the third network minus the second and the first) are used as test data.

The random and temporal splits are illustrated in Figure 4.5 whereas the temporal snapshots method is shown in 4.6. We identify next which method is used for each one of the networks we explore. Table 4.1 summarizes the number of users and edges on each set for the different datasets and subgraphs after applying the split.

Table 4.1: Dataset statistics.

	Twitter 1-month		Twitter 200-tweets		Facebook
	Interactions	Follows	Interactions	Follows	
Directed	Yes	Yes	Yes	Yes	No
Weighted	Yes	No	Yes	No	No
Split type	Temporal	Snapshots	Temporal	Snapshots	Random
# Users	9,528	9,770	9,985	9,964	4,039
# Training edges	116,332	630,504	104,866	427,568	56,466
# Validation edges	33,867	46,628	29,131	46,760	14,100
# Input edges	170,425	645,022	137,850	475,730	70,566
# Test edges	54,335	81,110	21,598	98,519	17,643

For the Facebook network we do not have any temporal information about the creation of the links, so we split the graphs randomly: we take 60% of the links as training set, 20% as validation and the remaining 20% as test.

In the interaction networks of both Twitter datasets, we apply a temporal split, since it better represents a real setting. In these graphs, we take as weight the frequency of interaction between every pair of users in the corresponding split (for example, if Peter and Jack interact three times in the period of time that corresponds to the training set and twice in the period for the test set, we just add the (Peter, Jack) link to the training set with $w(\text{Peter}, \text{Jack}) = 3$). For the 1-month dataset, we take as split points July 2th 0:00:00 GMT and July 9th 2015 0:00:00 GMT – we take two weeks for training, one for validation and the last one for testing. In the 200-tweets dataset, the selected split dates are July 24th 17:15:26 GMT and July 29th 2015 1:08:07 GMT – chosen so we can have $60\%, 20\%$ and 20% interactions in the training, validation and test sets, respectively. After removing the repeated edges and those crossing the network partition, the size of the validation and test subsets might decrease from the desired fraction, but the difference is not large (as it can be checked in Table 4.1).

Finally, we apply to the follows networks the snapshot approach, since Twitter does not provide information about when the follow links are created. Therefore, the follows network has been downloaded three times. For the 1-month dataset, the first download contains all the follows relations present in the Twitter network as of October 9th 2015. The first snapshot for the 200-tweets networks contains the links between users before October 20th. The validation snapshot is obtained for each network four months after the first one, and the test one two years after the second.

4.1.4 Description of the datasets

In addition to the raw numbers of nodes and edges present on each network, it is interesting to analyze other topological properties of each network. We provide some comment on them here. The first element we analyze is the degree distributions of the networks, illustrated in Figure 4.7. All the plots are in log-log scale, and include a (exponential) trend line, represented by a dotted line. In the different directed networks we explore, we do not only show their degree distribution – considering the degree of the nodes as the sum of their in-degree and out-degree – but we also plot the distributions for the in-degree and out-degree.

We observe that, for all five networks, they show the usual trend in real-world networks: highly skewed distributions where few nodes have high degree and most of them have low values for this characteristic. However, we can observe differences on the skewness of the degree distributions from one network to another. For instance, follows networks are more flattened than interaction ones, and the same occurs with the Facebook friendship graph. This might just be an effect of the density of the network (as follow networks are 4 or 5 times more dense than interaction ones), or it might reflect that users are likely to create some connections after they join the network, but not necessarily interact with such acquaintances. Then, comparing the 1-month and 200-tweets datasets by observing the undirected degree distribution, we can conclude that the 1-month dataset shows more skewed networks than the 200-tweets ones, thanks to the limitation of 200 tweets in the capture of the networks.

Table 4.2: Network properties of the input network.

	Twitter 1-month		Twitter 200-tweets		Facebook
	Interactions	Follows	Interactions	Follows	
Density	0.0018	0.0067	0.0013	0.0048	0.0087
Average degree	17.887	66.021	13.806	49.054	35, 107
Clustering coefficient	0.0562	0.2252	0.0939	0.1782	0.4160
Average shortest path length	3.5376	3.1074	4.6912	3.2796	4.0164
Diameter	13	10	19	10	11
Strongly Connected components	1, 701	1, 676	1, 318	1, 055	2
Giant component size (%)	81.84	82.60	86.66	88.98	99.95
# Tweets	1, 558, 518	2, 369, 596	1, 449, 267	1, 964, 585	–

Beyond the degree distribution, we show other properties of the networks. We report them in Table 4.2. First, we show the degree distribution and density of the different datasets. As it can be inferred from the number of edges, we observe that follows networks are much more dense than the interaction ones, but, the densest graph in our comparison is the undirected Facebook network (even when we report a smaller number of edges than any other network, the number of nodes is also smaller, and each undirected edge counts as two edges in an undirected graph). Despite this, the follows networks are the ones with the higher average degree. The increase in density has notable effects on the clustering coefficient of the networks: again, the 1-month and 200-tweets interactions networks have a lower clustering coefficient than their follow counterparts. Nonetheless, the Facebook network takes advantage of its construction (it is just the union of several ego-networks) to achieve a very large clustering coefficient value (almost doubling the next graph).

In terms of the average shortest path length, we find values similar to the ones reported by Myers et al. (2014) and Ugander et al. (2011): in the range of 3 to 5 steps. A bit larger is the diameter of the network, specially on the Twitter 200-tweets networks, where the maximum distance between connected nodes is close to 20 steps. The closest diameter is, again found in the follows networks. In terms of connected components, we omit the number of weakly connected components for the Twitter networks since there is only one. Surprisingly, there are more than one thousand strongly connected components for each of them, but most of them are comprised by one or two nodes: as it can be observed, for all networks, at least 81% of the nodes are located inside the giant component of the network. In the case of the undirected Facebook network, there are two connected components, but the giant component contains all but a couple of nodes.

Finally, in addition to the structural properties, we need to report another property of the datasets. In order to use content-based methods like Twittomender or centroid CB in our comparison, we need to define a reasonable set of features representing the different users in the network. Following the work by Hannon et al. (2010), we use the tweets obtained during the collection of the dataset. For the training networks, we only take the set of tweets before the split date. For the follows networks, since they were collected after the interaction graph, we use the whole set of tweets. In the case of the Facebook network we do not have access to any kind of side information or user-generated contents of the users, so we do not use content-based approaches for this network.

4.2 Experimental setup

As a preliminary exploration of contact recommendation, we provide a comprehensive comparison of state of the art approaches. Using an offline methodology, we compare the accuracy of these algorithms over the Twitter and Facebook networks described in Section 4.1. For that, we first provide, in this section, a thorough description of the chosen evaluation procedure and considerations we take not only on this first set of experiments, but in the ones we conduct in the following chapters.

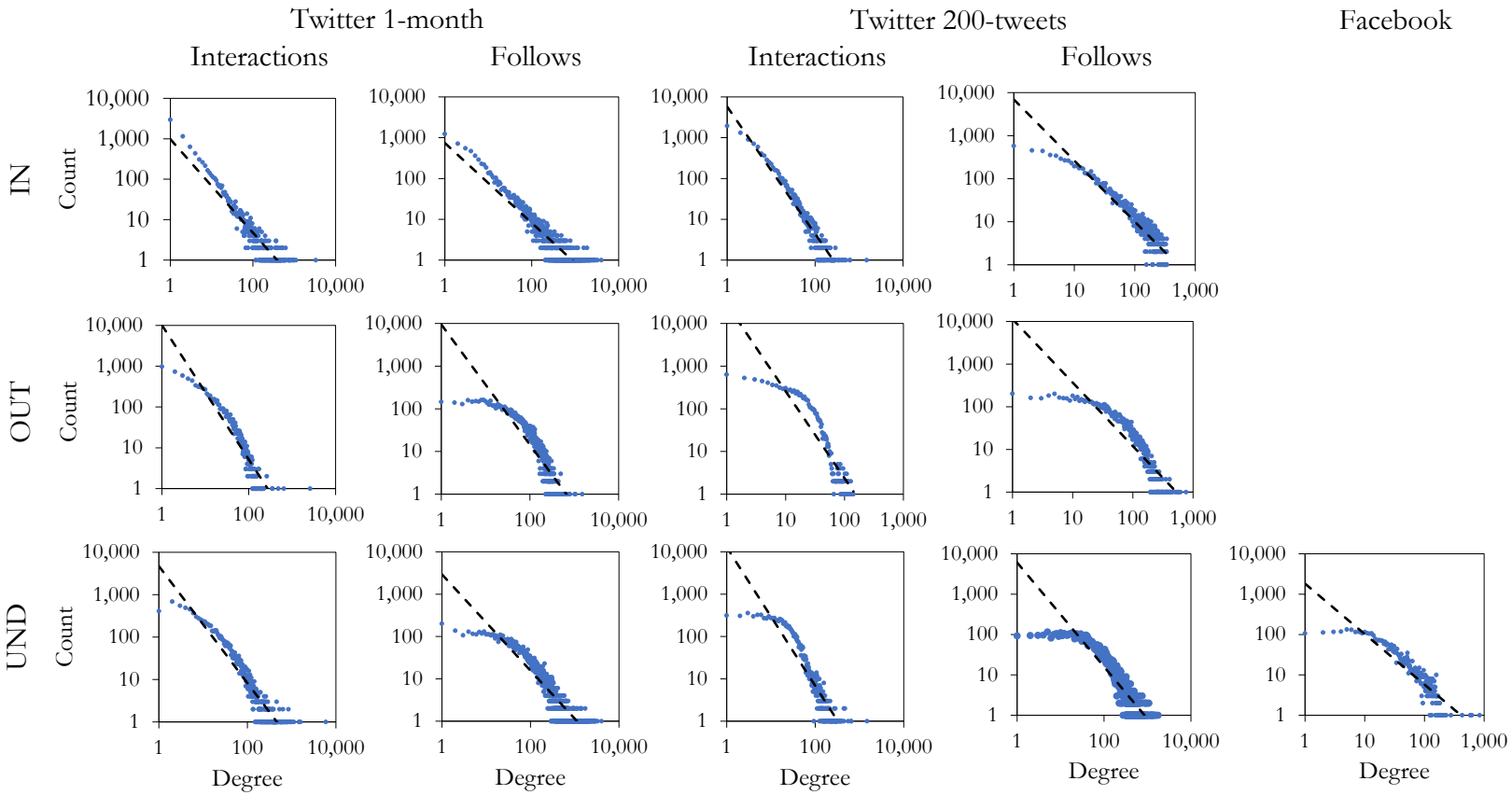


Figure 4.7: Degree distribution for the different datasets

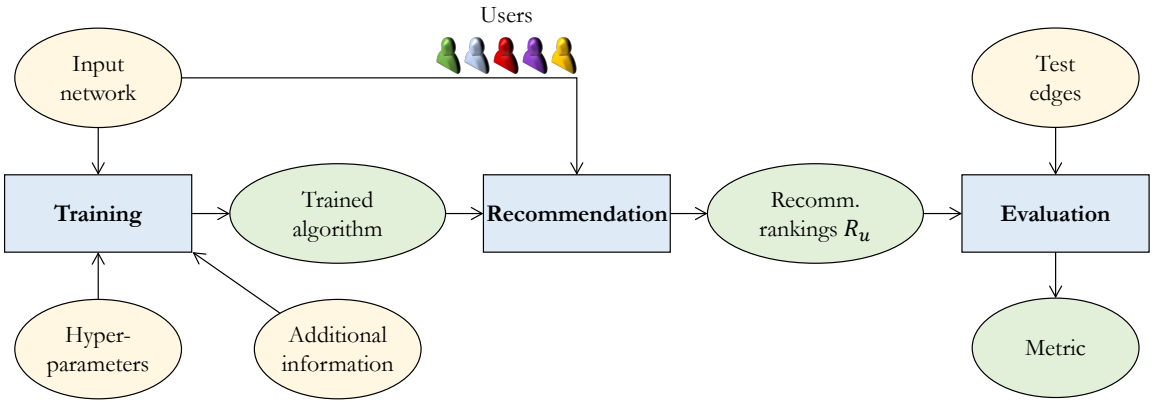


Figure 4.8: Evaluation procedure.

4.2.1 Experimental procedure

As we evaluate and compare the effectiveness of several recommendation approaches in an offline manner, we use the data splits introduced in Section 4.1 to perform our experiments. As we state before, we take the training and validation links to select an appropriate set of hyperparameters for each algorithm, and, using that selection along the input and test networks, we generate the definitive recommendations and we evaluate them for our comparison. In both steps (regardless of the pair of subgraphs we use), the procedure we follow is equivalent, and it is divided in three steps, we describe next: training, recommendation and evaluation. We illustrate it in Figure 4.8.

In the training phase, we prepare (train) the algorithm so it is capable of producing recommendations. We do it by providing to the system all the necessary data as input. All contact recommendation algorithms receive a network as input (the training network for the hyperparameter selection step or the input network otherwise), but some others need other information, such as hyperparameters or, in the case of content-based approaches, the set of tweets published by the different people in the network. The system then collects all the information it needs and, in some cases, learns a model to be able to suggest new users to befriend (for example, this is the case of matrix factorization algorithms, which learn the latent factors for the target and candidate users).

Once the algorithm has been trained, we enter the recommendation phase. In this step, a recommendation ranking is generated for every user in the network. This ranking sorts the candidate users in descending order according to the ranking function of the recommender. There are three kinds of users who are excluded from the recommendation: first, the target user already knows himself, so we thereby prevent him from appearing in the ranking; second, if there is a link from the target to the candidate user, we also exclude that candidate from being recommended (as it is not possible to create a new link between both users); finally, for directed networks, we also remove from the recommendation the people in the incoming neighborhood of the target user. We do this for avoiding the trivialization of the task: if we recommended reciprocal links, this recommendation approach would represent a trivial hard-to-beat baseline, given the high reciprocation ratio on Twitter. Furthermore, in Twitter, users are already notified when another user retweets, mentions or follows them, so recommending them a reciprocal link would be redundant for them, and barely add value for the users. We also remove such reciprocal connections from the test set to avoid distortions in the evaluation.

Then, in the evaluation phase, we measure the quality of the recommendations. We use several metrics for that, depending on the properties we analyze. In the experiments in this chapter, we measure the accuracy of the recommendations, by comparing the suggested candidate users with the links appearing in the test set. We describe the different accuracy metrics we consider in this thesis in the following section. Other properties beyond accuracy, such as the novelty and the diversity of the recommendations (Castells et al., 2015), can also be evaluated in this phase, but we do not cover them in this chapter: we define them and compare algorithms according to them in Chapter 8. In the end, after all metrics have been computed, we apply two tests to check whether the differences between algorithms are significant or not: first, a simple paired t-test over each pair of algorithms and then, to deal with multiple comparisons, a Tukey honest significant differences (HSD) test (Sakai, 2018, Tukey, 1949). Specific details about how these test work are provided in Appendix D.

Given how our experiments are designed, we consider that the task of the evaluated systems is to predict the future actions of users in the network (follow or interact with other people). Inasmuch as we can assume that these actions are beneficial for them, our experiments provide a fair proxy for determining how useful the recommendations could be if they were delivered as suggestions to the users – a common perspective in the evaluation practice of these systems (Cañamares et al., 2020, Gunawardana and Shani, 2015, Sanz-Cruzado and Castells, 2018a).

4.2.2 Accuracy metrics

We first define the different accuracy metrics that we employ in the different experiments we carry. Most of the metrics have their origin in the information retrieval field and are based on relevance judgments (Baeza-Yates and Ribeiro-Neto, 2011). In the contact recommendation task, we shall consider a link relevant if it appears in the network in the future (i.e. if it appears in the test network in the case of offline experiments) and not relevant if it does not. We should note that this observation does not prevent irrelevant links to become relevant in the future.

As a general procedure, for each user, a ranking of candidate users is produced, sorted by the ranking function of the recommendation approach. The metric is used to evaluate each individual ranking, and, finally, average over the different users. Since the full ranking is not shown to the target users of the recommendation, it is common to evaluate those metrics at cutoff k, i.e. as if the recommendation algorithm only returned k users in the ranking. We focus on ranking-based metrics, since they provide a natural evaluation perspective for the recommendation task. Under this perspective, we consider four different metrics:

Precision

The precision represents the simplest accuracy metric. It just measures the proportion of the recommended links which are relevant (Baeza-Yates and Ribeiro-Neto, 2011). If we denote as R_u the recommendation ranking for $u \in \mathcal{U}$, the precision is defined as:

$$P(u) = \frac{|\text{relevant}(u) \cap R_u|}{|R_u|} = \frac{|\{v \in R_u | (u,v) \in E_{test}\}|}{|R_u|} \tag{4.1}$$

where $\text{relevant}(u) = \{v \in \mathcal{U} | (u,v) \in E_{test}\}$ represents the positive relevance judgments for user u (i.e. the links that have u as starting point in the test set, E_{test}).

Recall

The recall provides a complementary metric for the precision, by determining the proportion of the relevant candidates which have been retrieved (Baeza-Yates and Ribeiro-Neto, 2011). Even when the ranking has irrelevant users, this metric can achieve a maximum value (something impossible for the precision). However, it requires knowing the whole set of relevant candidate users – so it can only be used in offline experiments:

$$R(u) = \frac{|\text{relevant}(u) \cap R_u|}{|\text{relevant}(u)|} = \frac{|\{v \in R_u | (u,v) \in E_{test}\}|}{|\{v \in \mathcal{U} | (u,v) \in E_{test}\}|} \tag{4.2}$$

Normalized discounted cumulative gain (nDCG)

The previous metrics consider that a relevant candidate user on any position of the ranking is equally valuable. However, it is not uncommon for latter elements in the list to go unnoticed. Therefore, after applying a recommendation approach, we want the most relevant items to be on top of the ranking. The nDCG metric (Järvelin and Kekäläinen, 2002) allows this, as well as considering several degrees of relevance in the judgments (we can take several grades of importance of the nodes instead of just binary values as it happened in precision and recall). The nDCG metric is defined as

$$\text{nDCG}(u) = \frac{DCG(u)}{IDGC(u)} \tag{4.3}$$

where the discounted cumulative gain (DCG) for the user u, $\mathrm{DCG}(u)$, is

$$\mathrm{DCG}(u) = \sum_{k=1}^{|R_u|} \frac{g_u(v_k)}{\log_2(1+k)} \tag{4.4}$$

with $g_u(v_k)$ representing the grade of relevance of the k-th candidate user (v_k) in the recommendation ranking for u. $\mathrm{IDCG}(u)$ acts as a normalization term, so nDCG takes values in the $[0, 1]$ interval and it represents the DCG value for the best possible recommendation ranking for u:

$$\mathrm{IDGC}(u) = \max_{R \in \sigma_u} \mathrm{DCG}(u) \tag{4.5}$$

where σ_u represents all the possible permutations of the set of candidate users for user u.

Mean average precision (MAP)

Another way to represent the accuracy of a recommender is the so-called precision-recall curve: a plot of the precision of a recommendation as a function of the recall. The average precision just represents the area under that curve, and it appears as another metric that considers the importance of recommending relevant people on the top positions of the ranking. It is defined as:

$$\mathrm{AP}(u) = \frac{\sum_{k=1}^{|R_u|} P@k(u) \cdot \mathbb{1}_{\mathrm{relevant}(u)}(v_k)}{\mathrm{relevant}(u)} \tag{4.6}$$

where $P@k(u)$ represents the value of precision applied over the top k users in the ranking. When this metric is averaged over all the users, it is known as mean average precision (MAP).

4.2.3 Algorithms

Finally, we report the algorithms we use in our preliminary experiments. We choose a comprehensive group of algorithms from the set of approaches described in Chapter 3. We summarize our selection here:

- **Friends of friends (FOAF):** most common neighbors (Liben-Nowell and Kleinberg, 2007), the Jaccard similarity (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007), cosine similarity (Lü et al., 2009, Salton et al., 1975), Adamic-Adar (Adamic and Adar, 2003, Liben-Nowell and Kleinberg, 2007) and resource allocation (Zhou et al., 2009).
- **Path-based:** We consider four different methods: the Katz algorithm (Katz, 1953, Liben-Nowell and Kleinberg, 2007), the local path index (Lü and Zhou, 2011, Lü et al., 2009), distance-based recommendation (Liben-Nowell and Kleinberg, 2007) and the global Leicht-Holme-Newman index (Leicht et al., 2006, Lü and Zhou, 2011).
- **Random walks:** As representative of the random walk family, we take the Money algorithm (Goel et al., 2015, Gupta et al., 2013), personalized (or rooted) PageRank (Liben-Nowell and Kleinberg, 2007, White and Smyth, 2003), the PropFlow algorithm (Lichtenwalter et al., 2010) and hitting and commute times (Liben-Nowell and Kleinberg, 2007). For the Money algorithm, as we work with small networks, we consider that all users belong to the circles of trust.
- **Collaborative filtering (CF):** We take three different methods: the matrix factorization algorithm for implicit feedback proposed by Hu et al. (2008), user-based and item-based nearest-neighbor collaborative filtering recommenders with cosine similarity (Ning et al., 2015).
- **Content-based (CB):** As content-based approaches, we consider the Twittomender algorithm (Hannon et al., 2010) as well as the centroid-based method introduced in Section 3.4.7.

For the different approaches, hyperparameters are selected using a simple grid search optimizing nDCG@10 on the validation set. For the friends of friends methods, that grid search includes the different orientation selections for the neighbors of the target and candidate users. Table C.2 in Appendix C shows, for each algorithm, its optimal configuration.

Table 4.3: Effectiveness of several state of the art contact recommendation approaches. The cell color goes from red (lower) to blue (higher values) for each metric/dataset, with the top value both underlined and highlighted in bold. Results for the statistical tests are shown in Figures D.1 to D.5 in Appendix D.

		Twitter 1-month						Twitter 200-tweets						Facebook		
		Interactions			Follows			Interactions			Follows					
	Algorithm	P	R	nDCG	P	R	nDCG	P	R	nDCG	P	R	nDCG	P	R	nDCG
CF	iMF	**0.0834**	**0.1415**	**0.1388**	**0.1025**	**0.1353**	**0.1461**	0.0515	**0.1350**	**0.1030**	**0.1013**	**0.1059**	**0.1328**	0.2911	0.3337	0.4249
	UB kNN	0.0810	0.1318	0.1367	0.0962	0.1271	0.1413	0.0475	0.1202	0.0954	0.0952	0.0980	0.1273	0.3245	0.4391	0.5135
	IB kNN	0.0736	0.1114	0.1172	0.0896	0.1131	0.1296	0.0360	0.0859	0.0724	0.0916	0.0912	0.1207	0.3058	0.3928	0.4542
FOAF	Adamic-Adar	0.0671	0.0921	0.0981	0.0815	0.1037	0.1175	**0.0529**	0.1230	0.0997	0.0865	0.0860	0.1140	0.3576	0.5043	0.5746
	MCN	0.0633	0.0850	0.0918	0.0833	0.1026	0.1189	0.0502	0.1151	0.0948	0.0846	0.0832	0.1110	0.3512	0.4913	0.5585
	Resource Allocation	0.0605	0.0833	0.0880	0.0754	0.0904	0.1039	0.0477	0.1119	0.0913	0.0847	0.0871	0.1117	**0.3680**	**0.5182**	**0.5922**
	Cosine	0.0234	0.0399	0.0393	0.0383	0.0387	0.0497	0.0251	0.0588	0.0480	0.0581	0.0600	0.0768	0.3311	0.4253	0.4943
	Jaccard	0.0225	0.0278	0.0317	0.0420	0.0399	0.0543	0.0296	0.0679	0.0571	0.0645	0.0647	0.0848	0.3271	0.4245	0.4913
Random walk	Money	0.0772	0.1325	0.1315	0.0750	0.1025	0.1104	0.0476	0.1180	0.0932	0.0840	0.0909	0.1131	0.3604	0.5115	0.5867
	Pers. PageRank	0.0598	0.1076	0.0996	0.0710	0.0931	0.0965	0.0331	0.0845	0.0630	0.0677	0.0743	0.0843	0.3607	0.5153	0.5891
	PropFlow	0.0581	0.1025	0.0973	0.0680	0.0918	0.0926	0.0313	0.0811	0.0632	0.0658	0.0719	0.0824	0.3465	0.4985	0.5700
	Hitting time	0.0234	0.0582	0.0512	0.0323	0.0422	0.0462	0.0204	0.0480	0.0412	0.0287	0.0299	0.0359	0.0249	0.0590	0.0671
	Commute time	0.0234	0.0581	0.0512	0.0323	0.0422	0.0462	0.0203	0.0480	0.0411	0.0287	0.0299	0.0359	0.0269	0.0627	0.0742
Path-based	Local Path Index	0.0701	0.1096	0.1160	0.0756	0.0972	0.1086	0.0477	0.1062	0.0887	0.0706	0.0706	0.0925	0.2975	0.3813	0.4569
	Katz	0.0136	0.0225	0.0220	0.0070	0.0078	0.0090	0.0075	0.0223	0.0173	0.0090	0.0093	0.0129	0.0364	0.0444	0.0532
	Distance	0.0090	0.0203	0.0152	0.0054	0.0093	0.0069	0.0131	0.0317	0.0235	0.0096	0.0130	0.0138	0.0318	0.0548	0.0464
	Global LHN Index	0.0016	0.0033	0.0027	0.0013	0.0026	0.0017	0.0078	0.0166	0.0121	0.0062	0.0105	0.0078	0.0648	0.1076	0.0891
CB	Centroid CB	0.0237	0.0300	0.0318	0.0376	0.0345	0.0471	0.0357	0.0745	0.0640	0.0563	0.0562	0.0736	-	-	-
	Twittomender	0.0012	0.0017	0.0018	0.0009	0.0005	0.0010	0.0013	0.0029	0.0024	0.0015	0.0014	0.0017	-	-	-
	Popularity	0.0313	0.0548	0.0572	0.0325	0.0412	0.0449	0.0225	0.0505	0.0422	0.0343	0.0279	0.0397	0.0283	0.0511	0.0522
	Random	0.0007	0.0009	0.0008	0.0012	0.0011	0.0014	0.0005	0.0007	0.0007	0.0014	0.0009	0.0016	0.0027	0.0028	0.0032

4.3 Results

Table 4.3 shows the experimental results for the best version of each algorithm in the different datasets. In this preliminary study, we compare the different algorithms in terms of three different accuracy metrics: precision, recall and nDCG at cutoff 10. Metrics are computed over those users who have (at least) a new outgoing edge in the test set. This is a standard option in recommender systems evaluation, as the rest of users would not change anything here. In general, the three accuracy metrics correlate to each other, so, in this section we just focus on commenting the results for nDCG@10.

We observe in such table that there are differences between datasets which can be attributed to the different nature and properties of such networks. Some algorithms show a similar behaviour over the different Twitter datasets. Classic collaborative filtering provides outstanding contact recommenders, followed by friends of friends link prediction approaches, such as Adamic-Adar or most common neighbors. In contrast, over the Facebook network, personalized Random walks and friends of friends methods stand out over the rest. This difference can be explained over the differences in terms of clustering coefficient between datasets (as shown in Table 4.2). As a large value of the metric indicates that most pairs of triplets are transitive (i.e. if there is a link from u to v and a link from v to w, it is likely that the (u, w) edge exists in the network), recommending people at distance 2 from the target user is more likely to work on networks with a high clustering coefficient. This is the case of the friends of friends algorithms, and, although personalized random walks can travel further, they commonly stay in the close environment of the target user (specially in networks with high clustering).

Across all the datasets, we observe some commonalities. The most notable one is the good performance of friends of friends approaches, with Adamic-Adar standing out among them. To a lesser extent, other alternatives such as resource allocation (which manages to be the best algorithm at Facebook) and MCN also achieve high accuracy values. Exceptions to this are the cosine similarity and Jaccard coefficient, which are quite suboptimal in all networks, always a step behind the rest of friends of friends methods.

On the other hand, path-based approaches are far from the top algorithms. Katz, the global LHN index and the distance-based recommendation method always appear among the four worst algorithms in the comparison (ignoring the random recommendation). The only exception to this is the local path index, which stands as a mid packer algorithm in the different datasets. About random walk approaches, Money, personalized PageRank and PropFlow seem to provide very good results in the Facebook network whereas they are far from optimal in Twitter. Among them, Money, the algorithm devised for the "Who to follow" service at Twitter seems to stand out above the rest. The other two approaches, hitting and commute time, achieve low values, close to those of popularity-based recommendation. Finally, on content-based methods, Twittomender is the worst of them, with a performance similar to random recommendation, whereas centroid CB, although having a greater performance, it is still far from mid packers such as local path index or personalized Pagerank.

4.4 Conclusions

In this chapter, we have introduced the data and the experimental setup we use in the following parts of the thesis, along with a first experiment, showing a comprehensive comparison of state of the art approaches. Overall, we can say that state of the art collaborative filtering methods such as matrix factorization (Hu et al., 2008) or kNN (Ning et al., 2015) represent very effective algorithms when we adapt them to the contact recommendation problem, specially on directed networks. Also, some of the most classic link prediction approaches based on friends of friends, like Adamic-Adar (Adamic and Adar, 2003, Liben-Nowell and Kleinberg, 2007) are also effective, achieving high accuracy values consistently over the different datasets.

Part II

Information retrieval models for contact recommendation

"The truth of things is the chief nutriment of superior intellects."
Leonardo Da Vinci

In this part, we investigate the connection between the contact recommendation and the text information retrieval (IR) tasks, by investigating the adaptation of IR models (classical and supervised) for recommending people in social networks using only the structure of these networks.

We first explore the adaptation of unsupervised IR models as direct standalone recommender systems. We report thorough experiments over Twitter and Facebook data where we observe that IR models, particularly BM25, are competitive when compared to state-of-the-art contact recommendation approaches. We further find that IR models have additional advantages in computational efficiency,and allow for fast incremental updates of recommendations as the network grows.

To provide explanations about the IR models performance, inspired by new advances in the axiomatic theory of IR, we study the existing IR axioms for the contact recommendation task. Our theoretical analysis and empirical findings show that while the classical axioms related to term frequencies and term discrimination seem to have a positive impact on the recommendation effectiveness, those related to length normalization tend to be not desirable for the task.

Finally, seeking additional effectiveness enhancements, we further explore the use of IR models as neighbor selection methods, in place of common similarity measures, in user-based and item-based memory-based collaborative filtering. On top of this, we research the application of learning to rank approaches borrowed from text IR to achieve additional improvements. Our research shows that the IR models are effective in three different roles: as direct contact recommenders, as neighbor selectors in collaborative filtering and as samplers and features in learning to rank.

5 Adaptation of IR models

Highlights

- *IR models can be adapted as standalone algorithms to effectively recommend contacts in social networks by mapping queries, documents and terms to the space of users in the graph.*
- *IR models, particularly BM25, are competitive when compared to state-of-the-art approaches.*
- *IR models are more efficient than best approaches and allow fast incremental updates of the recommenders as the network grows.*

The particular properties of contact recommendation approaches have motivated the creation of a wide variety of methods. These algorithms have their origin and inspiration in a wide variety of fields such as network science and link prediction (Liben-Nowell and Kleinberg, 2007, Lü and Zhou, 2011), supervised machine learning (Hasan et al., 2006, Lichtenwalter et al., 2010), recommender systems (Hannon et al., 2010, Sanz-Cruzado and Castells, 2018a) or, to a lesser extent, text information retrieval (Hannon et al., 2010). Following this last line of research, our work expands it to investigate the relation between contact recommendation in social networks and the text search IR task.

For this purpose, we establish associations between the basic elements on each of the tasks, and we then adapt classic IR models to the task of suggesting people to befriend or to interact with in social networks. We explore the adaptation of well-known models: the vector space model (Salton et al., 1975, Singhal et al., 1998), BM25 (Robertson and Zaragoza, 2009), query likelihood (Ponte and Croft, 1998) and divergence from randomness approaches (Amati, 2003, 2006, Amati and Van Rijsbergen, 2002, Amati et al., 2007, 2011). We empirically compare the effectiveness of the resulting algorithms to state of the art approaches, and we find that IR are competitive approaches when compared with the best alternatives. Moreover, we find additional advantages in terms of computational efficiency when producing recommendations from scratch as well as when incrementally updating the algorithms as the new users and links appear in the network.

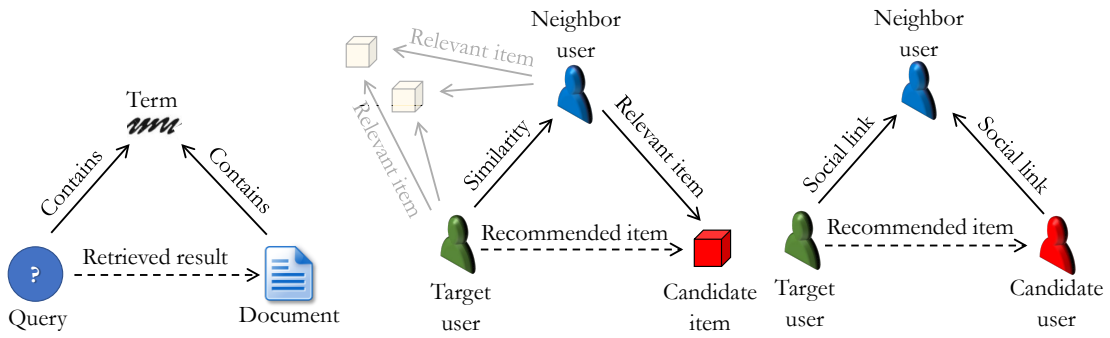
The work we present throughout this chapter has been partially published in three research articles:

- **Sanz-Cruzado et al. (2020a):** Javier Sanz-Cruzado, Pablo Castells, Craig Macdonald and Iadh Ounis. Effective Contact Recommendation in Social Networks by Adaptation of Information Retrieval Models. *Information Processing and Management*, 57(5),Article 102285, September 2020.
- **Sanz-Cruzado and Castells (2019a):** Javier Sanz-Cruzado and Pablo Castells. Information Retrieval Models for Contact Recommendation in Social Networks. In *Proceedings of the 41st European Conference on Information Retrieval (ECIR 2019)*, number 11437 in LNCS, pages 148–163, Cologne, Germany, April 2019. Springer.
- **Sanz-Cruzado et al. (2018a):** Javier Sanz-Cruzado, Sofía M. Pepa, and Pablo Castells. Recommending Contacts in Social Networks Using Information Retrieval Models. In *Proceedings of the 5th Spanish Conference onInformation Retrieval (CERI 2018)*, Zaragoza, Spain, June 2018. ACM.

5.1 Research questions

In this chapter, we address the following research questions:

- **RQ1:** How can we adapt IR models to be applied as contact recommendation algorithms?
- **RQ2:** Are they competitive compared to the state of the art algorithms?
- **RQ3:** For friends of friends approaches, which is the optimal orientation to select the target user's neighborhood? And for the candidate user?



(a) Three spaces in text retrieval (tripartite graph)

(b) Two spaces in item recommendation (bipartite graph)

(c) Single space in contact recommendation (unipartite graph)

Figure 5.1: Text IR elements (a) vs. item recommendation elements (b) vs. contact recommendation elements (c).

- **RQ4:** How do IR models compare to other state of the art approaches in terms of efficiency (execution time and memory consumption)?

5.2 IR framework for contact recommendation

Recommendation and text search have been traditionally considered as different tasks, and, as such, they have been separately addressed. However, relations between both problems can be established by considering, as many researchers do, that recommender systems can be described as retrieval systems where the explicit query is replaced by records of user activity (Bellogín et al., 2013). Taking this perspective, our work adapts models from the text IR task to the contact recommendation one.

A first step for this consists on establishing meaningful equivalences between the different elements in the IR task (queries, documents and terms) with the different spaces in the contact recommendation task (users and relations between them). As described in Section 2.2.2, the adaptation of information retrieval models to the recommendation task has been approached to formulate content-based recommendation algorithms (Adomavicius and Tuzhilin, 2005) or collaborative filtering approaches in a more general setting (with items like movies, songs, news, etc.). In those previous collaborative filtering approaches, the three IR spaces were commonly folded into two: the user and item sets (Bellogín et al., 2013, Parapar et al., 2013, Wang et al., 2008a,b).

However, when we seek to suggest people to other people in social networks, the difference between user and item spaces vanishes: they are both the same space. Therefore, to adapt IR models to our task, we fold the three spaces into a single dimension: the space of users in the social networks. Then, as we illustrate in 5.1, users play three different roles in our adaptation: the role of the queries, documents and terms in the information retrieval task. We provide some more detail on this mapping next.

First, we naturally associate the documents in the search space to the candidate users in the recommendation as they play the same role: they are the elements that the system needs to retrieve to satisfy a user need. In the IR task, that user need is explicitly expressed in the form of a query. Although such an explicit need does not exist in most recommender systems, it is usually assumed that users have an implicit one: in the case of contact recommendation, the need of creating new bonds with other people. Recommendation approaches address this implicit need by using the past activity of the user to generate predictions – which, in contact recommendation, is represented by the set of existing links to and/or from the target user. As this past activity plays an equivalent role to the query keywords in text search, we can then say that the target user plays the role of the query in our adaptation.

Finally, we need to establish the equivalences with the term representation of the documents and queries. In prior adaptations of IR models for recommendation, this equivalence was the main difficulty: since user and items are different objects, a representation that suits one might not be good for the other (Bellogín et al., 2013). For instance, in content-based recommendation, it is easy to find a representation of the items: the features representing the properties of the items. However, when we want to apply such a representation for the users, we have to aggregate the features of the items they have like in some manner (for example, using centroids

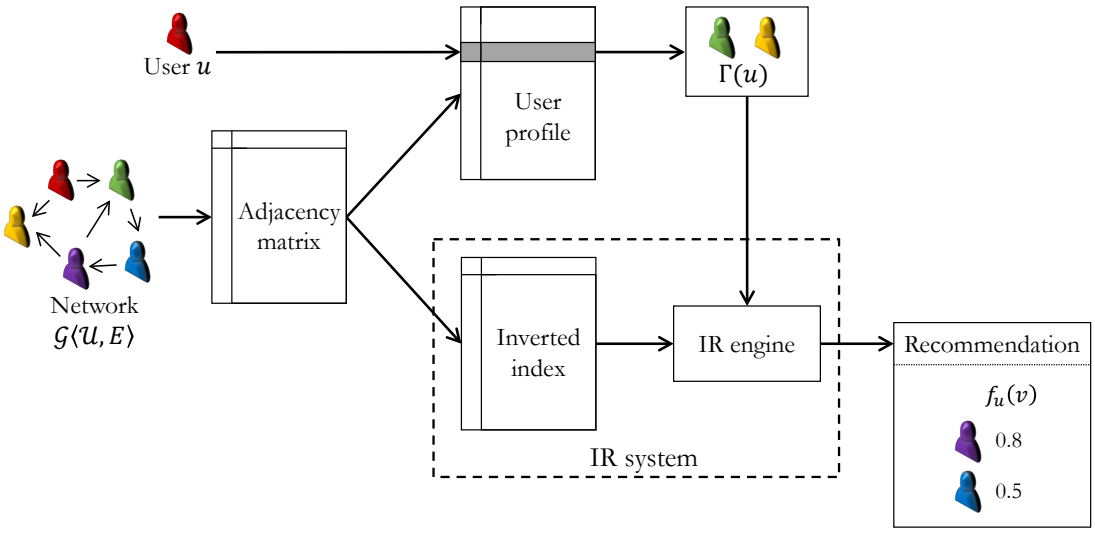


Figure 5.2: Framework for using IR models for contact recommendation.

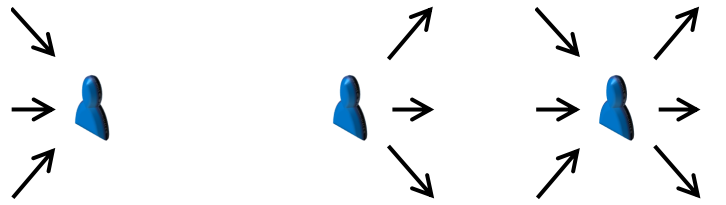
(Adomavicius and Tuzhilin, 2005)). This problem is also present in the collaborative filtering adaptation of IR models: Bellogín et al. (2013) proposed two different approaches: first, representing both users and items in the user space (representing items by their ratings and users by their similarities with other users) and, second, representing them in the item space (representing users by their ratings and items by their similarities).

In contact recommendation, finding a suitable representation for the terms becomes easier, since both users and items are the same thing. So, any term representation that works for the target users can be automatically applied for representing the "items" (i.e. the candidate users), without the need of applying any transformation. For instance, we can define content-based approaches by using texts associated to the users, such as messages or documents posted (or liked) by the users (Hannon et al., 2010) or demographic information. On the other hand, if we take the users as the term space and we match the term-document relationship with the connections between users, we obtain collaborative filtering algorithms, which represents our current focus.

We illustrate the proposed collaborative filtering adaptation framework in Figure 5.2. In that framework, a social network is encoded as a weighted adjacency matrix A, where $A_{uv} = w(u, v)$. Using the links available in the network, then, we build two elements: first, an inverted index which allows the fast retrieval of the candidate users, and, second and last, a structure that provides direct access to the profile of each user. This profile represents the neighborhood of the target users, i.e. the query representation. The inverted index takes as keys the network users (playing the role of terms), and the posting lists store the set of candidate users to whose neighborhood representation (as "documents") the "key" users belong to.

Using both structures, any text IR method can be applied as a contact recommendation algorithm. IR models are commonly defined as functions over the set of common terms between the query and the document. Consequently, when used as contact recommendation approaches and accordingly to the taxonomy we introduce in Chapter 3, they are part of the friends of friends family, with the "term" users being the common neighbors between the target and candidate users.

We want to note that, even after we have defined the connections between them, there are still differences between the text search and contact recommendation tasks. First, their nature is different: in text retrieval, the content of the documents is often assumed to be static; if it is changed, the document update can be processed as a new document. In contact recommendation, the equivalent of the words in documents are the social connections of candidate users, which, in contrast, evolve over time – a condition needed for the recommendation task to make sense: taking the structure of the network as a basis, we are predicting how the network will grow in the future. If that structure does not change, it is not possible to find useful recommendations. In the text IR analogy, the contact recommendation task can be seen somehow as predicting the new words that will be added to a document in the future. Another difference is that, if a link is present in the current network, it would not be useful to recommend it, in the general scenario where users may find little use in being recommended people who they already know and they are connected to. This sets an important difference with text retrieval, where



(a) Incoming neighborhood Γ_{in} (b) Outgoing neighborhood Γ_{out} (c) Undirected neighborhood Γ_{und}

Figure 5.3: Possible neighborhood orientations.

the equivalent constraint would imply that we should not return documents that are known to be relevant to the query.

Additional details and options remain open for adapting specific IR models to the contact recommendation task. In section 5.3 we address many of them, but, before that, we pay specific attention to one of these aspects, concerning the direction of social links in the reinterpretation of these text search models.

Neighborhood orientation

In directed social networks such as Twitter or Instagram, we can use three different definitions of user neighborhood, as described in Section 3.3: the incoming neighborhood $\Gamma_{in}(u)$, the outgoing neighborhood $\Gamma_{out}(u)$ and the union of both (the undirected neighborhood) $\Gamma_{und}(u) = \Gamma_{in}(u) \cup \Gamma_{out}(u)$, as shown in Figure 5.3. Any of the three options might be used in our adaptation of IR models. In addition, since the inverted index and the user profiles are created independently, we can even take a different orientation choice for the target and candidate users: since we still use the same elements to represent (the equivalent of) both queries and documents, it is possible to work smoothly with different neighborhood choices for targets and candidates.

The identification of the neighborhoods which best characterize the candidate and target users in the contact recommendation task represents an interesting problem by itself, (Golder et al., 2009, Hannon et al., 2010), and it concerns multiple state-of-the-art recommendation approaches besides IR adaptations: the performance of all the friends of friends approaches in Section 3.4.2, such as Adamic-Adar (Adamic and Adar, 2003) or Jaccard similarity (Jaccard, 1901, Salton and McGill, 1983) can also vary depending on what orientation we choose for the neighbors of the users involved. Therefore, in our experiments in Section 5.4, we shall analyze this issue in detail.

5.3 Adaptation of specific IR models

As examples of the general unification framework we proposed in the previous section, we detail the adaptation of several IR models to the contact recommendation task. According to the categorization of these methods in IR, we divide them in several families: probability ranking principle models (Robertson, 1977, Robertson and Zaragoza, 2009), language models (Lafferty and Zhai, 2001, Lavrenko and Croft, 2001, Ponte and Croft, 1998), divergence from randomness models (Amati, 2003, Amati and Van Rijsbergen, 2002) and vector space models (Salton et al., 1975).

For all our adaptations, we denote the neighborhood of the target user u as $\Gamma^q(u)$ (as the set of neighbors of the target user plays the role of the query) and the neighborhood of the candidate user v as $\Gamma^d(v)$ (as the candidate users act as the documents in the IR task). Table 5.1 provides a summary of the relations between the IR and contact recommendation elements which we explore in this section, as well as the notation we use for the elements in both tasks. Further details about those notations are provided in this section.

5.3.1 Probability ranking principle models

We start by exploring models based on the probability ranking principle (PRP) introduced by Robertson (1977). This principle establishes that, given a query, documents should be ranked according to their probability of

Table 5.1: Relation between the IR and contact recommendation elements.

Information retrieval	Contact recommendation
Document collection, D	Set of users, $\mathcal{U}$
Vocabulary, $\mathcal{V}$	Set of users, $\mathcal{U}$
Query, q	Target user's neighborhood, $\Gamma^q(u)$
Document, d	Candidate user's neighborhood, $\Gamma^d(u)$
Term $t \in q/d$	Neighbor user, $t \in \Gamma^q(u)/\Gamma^d(v)$
Documents containing a term, D_t	User's inverse neighborhood, $\Gamma^d_{inv}(t)$
Frequency of a term, freq(t, d)	Weight of a link, $w^d(v, t)$
Document length, $\|d\|$	Length of the user, len$^l(v)$

relevance. We study two models from this family, BIR and BM25 (Robertson and Zaragoza, 2009) and propose a new one, extreme BM25, based on the second.

Binary Independence Retrieval (BIR)

The Binary Independent Retrieval (BIR) model (Robertson and Zaragoza, 2009) is the simplest representative of IR models based on the probability ranking principle (Robertson, 1977). Assuming that the ocurrence of terms in a document follows a (multiple) Bernoulli distribution, this model estimates the probability of relevance of a document d for a query q as:

$$p(r|d, q) \propto \sum_{t \in d \cap q} \text{RSJ}(t) \tag{5.1}$$

where r represents the relevance of the document, and RSJ is the Robertson - Spärck-Jones formula (Robertson and Zaragoza, 2009):

$$\text{RSJ}(t) = \log \frac{|R_t|(|D| - |D_t| - |R| - |R_t|)}{(|R| - |R_t|)(|D| - |D_t|)} \tag{5.2}$$

In equation (5.2), R represents the set of relevant documents for the query, R_t is the set of relevant documents containing the term t, D is the document collection, and D_t is the set of documents containing t. Since the set R of the relevant documents for the query is not known, an approximation can be taken by considering that typically only a tiny fraction of the documents of the collection are relevant:

$$\text{RSJ}(t) = \log \frac{|D| - |D_t| + 0.5}{|D_t| + 0.5} \tag{5.3}$$

Then, to adapt this model for contact recommendation, as described in Section 5.2, we equate queries and documents to target and candidate users, respectively, and we also equate the term-document relationship to edges in the social network. Under this mapping, $|D|$ is the number of users in the network, $|\mathcal{U}|$ and D_t is the set of users that t is a neighbor of (i.e. the set of neighbors u would have if the orientation of the links was reversed). We shall call these set of users the inverse neighborhood of user t, and denote it as $\Gamma^d_{inv}(t) = \{u \in \mathcal{U} | t \in \Gamma^d(u)\}$. Using that notation, the adapted BIR equation becomes

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \text{RSJ}(w) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \log \frac{|\mathcal{U}| - |\Gamma^d_{inv}(t)| + 0.5}{|\Gamma^d_{inv}| + 0.5} \tag{5.4}$$

BM25

Starting from the same principle as BIR, but modelling term ocurrence in documents as a Poisson distribution (instead of a Bernoulli), we find one of the best-known and most effective probabilistic IR models, BM25 (Robertson and Zaragoza, 2009). The ranking function for this method is defined as

$$P(r|d, q) \propto \sum_{t \in d \cap q} \frac{(k+1)\text{freq}(t, d)}{k\left(1 - b + b\frac{|d|}{\text{avg}_{x \in D}|x|}\right) + \text{freq}(t, d)} \text{RSJ}(t) \tag{5.5}$$

where $\text{freq}(t, d)$ denotes the frequency of t in d, $|d|$ is the document length, $\text{RSJ}(w)$ is defined in Equation (5.3), and $k = [0, \infty)$ and $b \in [0, 1]$ are free parameters controlling the effect of term frequencies and the influence of the document length, respectively.

The text retrieval space can be mapped into the social network just as we describe before, now taking, in addition, edge weights as equivalent of term frequency. In directed networks, similarly to how the neighborhoods for the target and candidate users are selected, it is necessary to select the weight of incoming or outgoing edges as the equivalent of frequency. We link this decision to the edge orientation selected for candidate users, as follows:

$$\text{freq}(t, v) = w^d(t, v) = \begin{cases} w_{in}(t, v) = w(t, v) & \text{if } \Gamma^d := \Gamma_{in} \\ w_{out}(t, v) = w(v, t) & \text{if } \Gamma^d := \Gamma_{out} \\ w_{und}(t, v) = w(t, v) + w(v, t) & \text{otherwise} \end{cases} \tag{5.6}$$

The last element we have to map is document length, which can be now defined as the sum of edge weights of the candidate user. In unweighted graphs, this is equivalent to the degree of the node. On the other hand, in directed networks, different choices can be considered. The BM25 formulation for text retrieval considers different options in defining the document length, such as number of unique terms or sum of frequencies (Robertson and Zaragoza, 2009). We find similarly worthwhile to decouple the orientation choice for document length from the term representation of the candidate users. We reflect this by defining the length of a candidate user as

$$\text{len}^l(v) = \sum_{t \in \Gamma^l(v)} w^l(t, v) \tag{5.7}$$

where $\Gamma^l(v)$ represents the candidate's neighborhood in a specific orientation choice for document length. Based on this, the adaptation of BM25 becomes:

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{(k+1) w^d(t, v)}{k \left(1 - b + b \frac{\text{len}^l(v)}{\text{avg}_{x \in \mathcal{U}} \text{len}^l(x)}\right) + w^d(t, v)} \text{RSJ}(t) \tag{5.8}$$

Extreme BM25

In addition to the previous probabilistic models, we introduce a new one, which we denote by extreme BM25 (EBM25). This algorithm is defined as the BM25 algorithm when the parameter k tends to ∞. Its equation is the following:

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{w^d(t, v)}{1 - b + b \frac{\text{len}^l(v)}{\text{avg}_{x \in \mathcal{U}} \text{len}^l(x)}} \text{RSJ}(t) \tag{5.9}$$

The main difference between BM25 and Extreme BM25 is the disappearance of the smoothing of the numerator. This approach is useful for analyzing the importance of this element on the contact recommendation task, as we see in Chapter 6.

5.3.2 Language models

Another family of probabilistic IR approaches is the family of language models (Lafferty and Zhai, 2001, Lavrenko and Croft, 2001, Ponte and Croft, 1998). A language model is a probability distribution over the linguistic units, such as words, sentences or whole documents. Given a text, its language model can be used to generate new text by picking a term at random with probability of obtaining each word equal to the one assigned by the language model.

One of the most widely used and successful language modelling methods in IR is the so-called query likelihood (QL) model (Ponte and Croft, 1998). This model ranks documents depending on whether the language

model for the document is likely to generate the text from the query. We can formulate this intuition as

$$f_q(d) = p(q|d) \propto \sum_{t \in q} \text{freq}(t, q) \log_2(p(t|d)) \tag{5.10}$$

where $p(t|d)$ (the probability that term t is generated by the document language model) can be estimated by maximum likelihood. Using our mapping, we can easily adapt this model to the contact recommendation task. The ranking function defined in equation (5.10) translates to

$$f_u(v) = p(u|v) \propto \sum_{t \in \Gamma^q(u)} w^q(t, u) \log_2 p(t|v) \tag{5.11}$$

for recommending people to people in networks.

To prevent the whole score of a document from vanishing because a single query term does not appear in it, it is common to apply a smoothing technique for estimating $p(t|d)$. These smoothing techniques combine the probability $p(t|d)$ with the probability of randomly obtaining term t in the collection, $p_c(t)$, which is defined as:

$$p_c(t) = \frac{\sum_{d \in D_t} \text{freq}(t, d)}{\sum_{d \in D} \text{len}(d)} \tag{5.12}$$

In contact recommendation, this can be easily translated using our framework, into:

$$p_c(t) = \frac{\sum_{v \in \mathcal{U}} w^d(t, v)}{\sum_{v \in \mathcal{U}} \text{len}^d(v)} = \frac{\text{len}^d_{inv}(t)}{\sum_{v \in \mathcal{U}} \text{len}^d(v)} \tag{5.13}$$

where $\text{len}^d_{inv}(t)$ represents the length of user t if the orientation edges were inverted:

$$\text{len}^d_{inv}(t) = \sum_{x \in \Gamma^d_{inv}(t)} w^d(t, x) \tag{5.14}$$

There are several ways to combine this probabilities. In this thesis, we explore three different smoothing techniques for query likelihood models, which have been previously applied in adaptations of this model to recommender systems (Bellogín et al., 2013, Valcarce, 2015, Valcarce et al., 2017): the Jelinek-Mercer (Jelinek and Mercer, 1980), the Dirichlet (MacKay and Peto, 1995) and the Laplace (Hazimeh and Zhai, 2015, Valcarce et al., 2017) smoothings. For each smoothing technique, we can formulate a different query likelihood approach:

Query likelihood with Jelinek-Mercer smoothing (QLJM)

The Jelinek-Mercer smoothing combines the maximum likelihood estimation and the probability $p_c(t)$ by linearly interpolating both probabilities. The amount of smoothing is then controlled by a parameter $\lambda \in [0, 1]$. Values closer to 0 give less importance to the smoothing (with $\lambda = 0$ meaning that no smoothing is applied). After applying this smoothing technique, QLJM for contact recommendation is reformulated as:

$$f_u(v) = \sum_{t \in \Gamma^q(u)} w^q(t, u) \log \left((1 - \lambda) \cdot \frac{w^d(t, v)}{\text{len}^d(v)} + \lambda \cdot p_c(t) \right) \tag{5.15}$$

It is possible to reformulate equation (5.15) to an equivalent expression which has efficiency advantages when using inverted indexes. This reformulation will be specially relevant in Chapter 6, and it is shown in equation (5.16).

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t, u) \log \left(\frac{\lambda}{1 - \lambda} \cdot \frac{w^d(t, v)}{\text{len}^d(v)} \cdot \frac{\sum_{x \in \mathcal{U}} \text{len}^d(x)}{\text{len}^d_{inv}(t)} \right) \tag{5.16}$$

Query likelihood with Dirichlet smoothing (QLD)

The Dirichlet smoothing technique (MacKay and Peto, 1995) is derived from a maximum likelihood estimation of $p(t|d)$ considering that the prior $p(t)$ follows a Dirichlet distribution. This has as a result the following formulation for the query likelihood method:

$$f_u(v) = \sum_{t \in \Gamma^q(u)} w^q(t,u) \log \left(\frac{w^d(t,v) + \mu \cdot p_c(t)}{\text{len}^d(v) + \mu} \right) \tag{5.17}$$

where $\mu > 0$ is a hyperparameter used to manage the importance of the smoothing.

Similarly to how we modified the QLJM approach, we can also rewrite this equation as the sum of the neighbor users shared between the target and candidate users. The resulting equation is:

$$\begin{aligned} f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} & \left[w^q(t,u) \log \left(\frac{w^d(t,v)}{\mu} \cdot \frac{\sum_{x \in \mathcal{U}} \text{len}^d(x)}{\text{len}^d_{inv}(t)} \right) \right] \\ & - \text{len}^q(u) \log \left(\frac{\text{len}^d(v)}{\mu} \right) \end{aligned} \tag{5.18}$$

Query likelihood with Laplace smoothing (QLL)

Also known as additive smoothing, this technique does not take into account the prior probability of appearance of a term in the collection. Instead, it adds a small amount (known as a pseudo-count) $\gamma > 0$ to each of the weights $w^d(t,v)$. That way, we define the new formulation as:

$$f_u(v) = \sum_{t \in \Gamma^q(u)} w^q(t,u) \log \left(\frac{w^d(t,v) + \gamma}{\text{len}^d(v) + \gamma |\mathcal{U}|} \right) \tag{5.19}$$

And, similarly to how we have done with the Dirichlet and Jelinek-Mercer smoothings, equation (5.19) can be rewritten as:

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \log \left(\frac{w^d(t,v) + \gamma}{\gamma} \right) - \text{len}^q(u) \log \left(\text{len}^d(v) + \gamma |\mathcal{U}| \right) \tag{5.20}$$

5.3.3 Divergence from randomness models

The divergence from randomness (DFR) models are a another family of probabilistic IR models, which have their foundations on the idea that, given a term t and a document d, the more the distribution of the term in the document diverges from the distribution of the term in the whole collection, the more informative will the term be for that document (Amati, 2003) and, consequently, the more important that term is for describing the document. For instance, if the term "quantum" appears only a few times in the collection, but in a document, it is the most used term, then, it is reasonable to think that "quantum" is one of the words that better describes that document. The DFR family of algorithms is very wide, and we focus here on five popular DFR algorithms: PL2, DFRee, DFReeKLIM, DLH and DPH, all of them available in the Terrier IR Platform (Macdonald et al., 2012, Ounis et al., 2006).

PL2

PL2 (Amati, 2003, Amati and Van Rijsbergen, 2002) is one of the best-known DFR models. This model is characterized by three different elements. First, the divergence between the distribution of the terms in each individual document and the collection is measured using a Poisson distribution. Second, a Laplace aftereffect estimation is applied as a first normalization of the model and, last, the term frequency is normalized using the

so-called Normalisation 2 (Amati, 2003, Amati and Van Rijsbergen, 2002). In terms of queries and documents, the formula for this algorithm is the following:

$$f_q(d) = \sum_{t \in q} \frac{\text{freq}(t,q)}{\hat{F}(t,d)+1} \left[\hat{F}(t,d) \log_2 \frac{|D| \cdot \hat{F}(t,d)}{\sum_{x \in D} \text{freq}(t,x)} + \frac{\log_2(2\pi \hat{F}(t,d)}{2} + \right. \\ \left. + \left(\frac{\sum_{x \in D} \text{freq}(t,x)}{|D|} + \frac{1}{12\hat{F}(t,d)} - \hat{F}(t,d) \right) \log_2 e \right] \tag{5.21}$$

where $\hat{F}(t,d)$ is the frequency of the term in the document after applying Normalization 2, and it is formulated as

$$\hat{F}(t,d) = \text{freq}(t,d) \cdot \log_2 \left(1 + c \cdot \frac{\text{avg}_{x \in D} |x|}{|v|} \right) \tag{5.22}$$

As a contact recommendation algorithm, equation (5.21) leads to the following formulation:

$$f_u(v) = \sum_{t \in \Gamma^q(u)} \frac{w^q(t,u)}{\hat{w}^d(t,v)+1} \left[\hat{w}^d(t,v) \log_2 \frac{|\mathcal{U}| \cdot \hat{w}^d(t,v)}{\text{len}^d_{inv}(t)} + \frac{\log_2(2\pi \hat{w}^d(t,v))}{2} + \right. \\ \left. + \left(\frac{\text{len}^d_{inv}(t)}{|\mathcal{U}|} + \frac{1}{12\hat{w}^d(t,v)} - \hat{w}^d(t,v) \right) \log_2 e \right] \tag{5.23}$$

where $\hat{w}^d(v,t)$ is the equivalent to the normalized term frequency defined in equation (5.22), which, after the mapping, has the following formulation:

$$\hat{w}^d(v,t) = w^d(t,v) \log_2 \left(1 + c \cdot \frac{\text{avg}_{x \in \mathcal{U}} \text{len}^d(x)}{\text{len}^d(v)} \right) \tag{5.24}$$

DFRee

DFRee stands for DFR free from parameters. This model applies concepts from information theory (Shannon, 1948) to measure the divergence between the document and the collection. Specifically, this model takes as a measure of divergence the number of extra bits we would need to code the document if an extra appearance of the term was added to it. As there are two possible ways of sampling the distribution (considering only the document, or the complete collection), this method averages both information measures (i.e. computes their inner product), as follows:

$$f_q(d) = \sum_{t \in q} \left[\text{freq}(t,d) \log \frac{\hat{p}(d,t)}{p(t)} + (\text{freq}(t,d)+1) \log_2 \frac{\hat{p}^+(t,d)}{p(t)} + \right. \\ \left. + \frac{1}{2} \log \frac{\hat{p}^+(t,d)}{\hat{p}(t,d)} \right] \text{freq}(t,q) \cdot \text{freq}(t,d) \log \frac{\hat{p}^+(t,d)}{\hat{p}(t,d)} \tag{5.25}$$

where $p(t)$ represents the prior probability of the term in the collection, $\hat{p}(d,t)$ is the relative frequency of the term in the document and $\hat{p}^+(d,t)$ represents the relative frequency of the term after we add one more occurrence to the document. When adapted to contact recommendation, the formulation for this model is the following:

$$f_u(v) = \sum_{t \in \Gamma^q(u)} \left[w^d(t,v) \log \frac{\hat{p}(t,v)}{p(t)} + (w^d(t,v)+1) \log_2 \frac{\hat{p}^+(t,v)}{p(t)} + \right. \\ \left. + \frac{1}{2} \log \frac{\hat{p}^+(t,v)}{\hat{p}(t,v)} \right] w^q(t,u) w^d(t,d) \log \frac{\hat{p}^+(t,v)}{\hat{p}(t,v)} \tag{5.26}$$

where the probabilities $p(t)$, $\hat{p}(v,t)$ and $\hat{p}^+(v,t)$ have the following formulation:

$$p(t) = \frac{\text{len}^d_{inv}(t)}{\sum_{x \in \mathcal{U}} \text{len}^d(x)} \tag{5.27}$$

$$\hat{p}(t,v) = \frac{w^d(t,v)}{\text{len}^d(v)} \tag{5.28}$$

$$\hat{p}^+(t,v) = \frac{w^d(t,v)+1}{\text{len}^d(v)+1} \tag{5.29}$$

DFReeKLIM

A model specifically proposed to deal with short documents such as tweets, the DFReeKLIM model (parameter-free DFR using a Kullback-Leibler based product of Information Measures) is another DFR approach without hyperparameters which uses information theory concepts to compute the divergence (Amati et al., 2011). This model is computed as the inner product of two information measures: the additional cost, in bits, of coding the document this term belongs to when considering the probability $\hat{p}(t,d)$, and the cost of adding this term to the document with respect to the optimal encoding of the document. The formulation is then the following:

$$f_q(d) = |d| \sum_{t \in q} \text{freq}(t,d) \left[\log \frac{\hat{p}(t,d)}{p(t)} \log \frac{\hat{p}^+(t,d)}{\hat{p}(t,d)} \right] \tag{5.30}$$

which, mapped to contact recommendation, is:

$$f_u(v) = \text{len}^d(v) \sum_{t \in \Gamma^q(u)} w^q(t,u) \hat{p}(t,v) \left[\log \frac{\hat{p}(t,v)}{p(t)} \log \frac{\hat{p}^+(t,v)}{\hat{p}(t,v)} \right] \tag{5.31}$$

where $p(t)$, $\hat{p}(t,v)$ and $p^+(t,v)$ are described in equations (5.27), (5.28) and (5.29) respectively.

DPH and DLH

The DLH (Amati, 2006) and DPH (Amati et al., 2007) are both parameter-free DFR models that use a hypergeometric distribution as the divergence measure. Their only difference is the normalization scheme they use: DLH uses the Laplace normalization and DPH the Popper normalization. We show next the formulations. For the DLH approach, its equation is:

$$f_q(d) = \sum_{t \in q} \text{freq}(t,q) \frac{\text{freq}(t,d) \log \frac{\hat{p}(t,d)}{p(t)} + 0.5 \log \left(2\pi \cdot \text{freq}(t,d)(1-\hat{p}(t,d))\right)}{\text{freq}(t,d)+1} \tag{5.32}$$

where $p(t)$ is the prior probability of term t in the collection and $\hat{p}(t,d)$ is the relative frequency of t in the document d, as we defined for DFRee. The score for the DPH method is computed as:

$$\begin{aligned} f_q(d) = \sum_{t \in q} \left(1 - \frac{\text{freq}(t,d)}{\text{len}(v)}\right)^2 \left[\text{freq}(t,d) \log \left(\frac{\text{freq}(t,d) \sum_{x \in D} |x|}{|d|} \right) + \right. \\ \left. + \frac{1}{2} \log \left(2\pi \cdot \text{freq}(t,d)\right) \left(1 - \frac{\text{freq}(t,d)}{|d|}\right) \right] \frac{\text{freq}(t,q)}{\text{freq}(t,d)+1} \end{aligned} \tag{5.33}$$

We can use this approaches as contact recommendation networks by applying the mapping in Table 5.1. The DLH model is redefined as

$$f_u(v) = \sum_{t \in \Gamma^q(u)} w^q(t,v) \frac{w^d(t,v) \log \frac{\hat{p}(t,v)}{p(t)} + 0.5 \log \left(2\pi w^d(t,v)(1-\hat{p}(t,v))\right)}{w^d(t,v)+1} \tag{5.34}$$

where $p(t)$ and $p(t,v)$ are defined in equations (5.27) and (5.28) and the DPH is formulated as

$$f_u(v) = \sum_{t \in \Gamma^q(u)} \left(1 - \frac{w^d(t,v)}{\text{len}^d(v)}\right)^2 \left[w^d(t,v) \log\left(\frac{w^d(t,v) \sum_{v' \in \mathcal{U}} \text{len}^d(v')}{\text{len}^d(v)}\right) + \right. \\ \left. + \frac{1}{2} \log\left(2\pi w^d(t,v)\right)\left(1 - \frac{w^d(t,v)}{\text{len}^d(v)}\right)\right] \frac{w^q(t,u)}{w^d(t,v)+1} \tag{5.35}$$

5.3.4 Vector space model

The vector space model (VSM) is one of the first and best-known models in IR (Salton et al., 1975). This model represents both documents and queries as $|\mathcal{V}|$-dimensional vectors, where $\mathcal{V}$ represents the set of terms in the collection. Using this model, the ranking function is often the cosine similarity of the query and document vectors. It can be formulated as:

$$f_q(d) = \sum_{t \in q \cap d} \frac{q_t d_t}{\sqrt{\sum_{t \in d} d_t^2}} \tag{5.36}$$

where q_t and d_t represent the coordinate of the query and document vectors for term t. This coordinate is defined as the product of two separate terms: the term frequency (tf), which is a function of the number of appearances of the term in the document, and the inverse document frequency (idf), a term that gives more value to the more discriminative terms (those which appear in just a few documents). There are different possibilities for both functions. After applying the mapping, the VSM equation is

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{u_t v_t}{\sqrt{\sum_{t \in \Gamma^d(v)} v_t^2}} \tag{5.37}$$

where u_t is the coordinate for the target user and v_t the coordinate for the candidate user. Among the different possibilities there are for defining tf and idf, in this thesis we use the following simple and common tf-idf definition (Baeza-Yates and Ribeiro-Neto, 2011):

$$u_t = \text{tf-idf}(u,t) = (1 + \log_2 w^q(u,t)) \cdot \log_2\left(1 + \frac{|\mathcal{U}|}{1 + |\Gamma^q_{inv}(t)|}\right) \tag{5.38}$$

$$v_t = \text{tf-idf}(v,t) = (1 + \log_2 w^d(u,t)) \cdot \log_2\left(1 + \frac{|\mathcal{U}|}{1 + |\Gamma^d_{inv}(t)|}\right) \tag{5.39}$$

where we add 1 to the denominators in the idf expressions to avoid division by zero when a user has no neighbors (this might happen in partial directed network samples, differently from text collections, where each term is supposed to appear at least once in the collection).

Pivoted length normalization VSM

In addition to the classic VSM, we include a generally more effective variant of this model, proposed by Singhal et al. (1998) for the TREC-7 conference. In this approach, the length normalization applied when we are computing the cosine similarity between two vectors is exchanged by another length normalization, similar to the one for BM25 (Robertson and Zaragoza, 2009). For the search task, the equation for this model is defined as

$$f_q(d) = \sum_{t \in q \cap d} \text{freq}(t,u) \cdot \frac{1 + \log \text{freq}(t,d)}{1 - s + s\frac{|d|}{\text{avg}_{x \in D}|x|}} \cdot \log \frac{|D|+1}{|D_t|} \tag{5.40}$$

which, after we apply our mapping, becomes

$$f_u(v) = \sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \cdot \frac{1 + \log w^d(t,v)}{1 - s + s\frac{\text{len}^d(v)}{\text{avg}_{x \in \mathcal{U}} \text{len}^d(x)}} \cdot \log \frac{|\mathcal{U}|+1}{|\Gamma^d_{inv}(t)|} \tag{5.41}$$

for the contact recommendation task.

Tables 5.2, 5.3, 5.4 and 5.5 show a summary of the different formulations of the IR models introduced in this section.

Table 5.2: Probability ranking principle (PRP) models.

Algorithm	Equation $f_u(v)$
BIR	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \log \frac{\lvert\mathcal{U}\rvert - \lvert\Gamma^d_{inv}(t)\rvert + 0.5}{\lvert\Gamma^d_{inv}\rvert + 0.5}$
BM25	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{(k+1) \cdot w^d(t,v) \cdot \mathrm{RSJ}(t)}{k\left(1 - b + b\frac{\mathrm{len}^l(v)}{\mathrm{avg}_{x \in \mathcal{U}} \mathrm{len}^l(x)}\right) + w^d(t,v)}$
Extreme BM25	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{w^d(t,v) \cdot \mathrm{RSJ}(t)}{1 - b + b\frac{\mathrm{len}^l(v)}{\mathrm{avg}_{x \in \mathcal{U}} \mathrm{len}^l(x)}}$

Table 5.3: Language models - query likelihood (QL) models

Algorithm	Equation $f_u(v)$
QLD	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \left[w^q(t,u) \log\left(\frac{w^d(t,v)}{\mu} \cdot \frac{\sum_{x \in \mathcal{U}} \mathrm{len}^d(x)}{\mathrm{len}^d_{inv}(t)}\right)\right] - \mathrm{len}^q(u) \log\left(\frac{\mathrm{len}^d(v)}{\mu}\right)$
QLJM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \log\left(\frac{\lambda}{1-\lambda} \cdot \frac{w^d(t,v)}{\mathrm{len}^d(v)} \cdot \frac{\sum_{x \in \mathcal{U}} \mathrm{len}^d(x)}{\mathrm{len}^d_{inv}(t)}\right)$
QLL	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \log\left(\frac{w^d(t,v) + \gamma}{\gamma}\right) - \mathrm{len}^q(u) \log\left(\mathrm{len}^d(v) + \gamma\lvert\mathcal{U}\rvert\right)$

Table 5.4: Divergence from randomness (DFR) models

Algorithm	Equation $f_u(v)$
PL2	$\sum_{t \in \Gamma^q(u)} \frac{w^q(t,u)}{\hat{w}^d(t,v)+1}\left[\hat{w}^d(t,v) \log_2 \frac{\lvert\mathcal{U}\rvert \cdot \hat{w}^d(t,v)}{\mathrm{len}^d_{inv}(t)} + \frac{\log_2(2\pi\hat{w}^d(t,v))}{2} + \right.$ $\left. + \left(\frac{\mathrm{len}^d_{inv}(t)}{\lvert\mathcal{U}\rvert} + \frac{1}{12\hat{w}^d(t,v)} - \hat{w}^d(t,v)\right)\log_2 e\right]$
DFRee	$\sum_{t \in \Gamma^q(u)} \left[w^d(t,v) \log \frac{\hat{p}(t,v)}{p(t)} + (w^d(t,v)+1)\log_2 \frac{\hat{p}^+(t,v)}{p(t)} + \right.$ $\left. + \frac{1}{2}\log \frac{\hat{p}^+(t,v)}{\hat{p}(t,v)}\right] w^q(t,u) w^d(t,d) \log \frac{\hat{p}^+(t,v)}{\hat{p}(t,v)}$
DFReeKLIM	$\sum_{t \in \Gamma^q(u)} w^q(t,u)\hat{p}(t,v)\left[\log \frac{\hat{p}(t,v)}{p(t)} \log \frac{\hat{p}^+(t,v)}{\hat{p}(t,v)}\right] \mathrm{len}^d(v)$
DLH	$\sum_{t \in \Gamma^q(u)} w^q(t,v) \frac{w^d(t,v) \log \frac{\hat{p}(t,v)}{p(t)} + 0.5 \log\left(2\pi w^d(t,v)(1-\hat{p}(t,v))\right)}{w^d(t,v)+1}$
DPH	$\sum_{t \in \Gamma^q(u)} \left(1 - \frac{w^d(t,v)}{\mathrm{len}^d(v)}\right)^2 \left[w^d(t,v) \log\left(\frac{w^d(t,v) \sum_{v' \in \mathcal{U}} \mathrm{len}^d(v')}{\mathrm{len}^d(v)}\right) + \right.$ $\left. + \frac{1}{2}\log\left(2\pi w^d(t,v)\right)\left(1 - \frac{w^d(t,v)}{\mathrm{len}^d(v)}\right)\right] \frac{w^q(t,u)}{w^d(t,v)+1}$

5.4 Experiments

To analyze the performance of the adapted IR models and compare them with other baselines, we conduct several offline experiments using Facebook and Twitter social networks, and, in addition to a comparison between algorithms, we examine the effect of different edge orientations for the target and candidate users. In the following, we describe the data, as well as our experimental approach and setup.

Table 5.5: Vector space model (VSM) models.

Algorithm	Equation $f_u(v)$
VSM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{u_t v_t}{\sqrt{\sum_{t \in \Gamma^d(v)} v_t^2}}$
Pivoted normalization VSM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{w^q(t,u) \cdot \left(1 + \log w^d(t,v)\right)}{1 - s + s \frac{\text{len}^d(v)}{\text{avg}_{x \in \mathcal{U}} \text{len}^d(x)}} \cdot \log \frac{\lvert\mathcal{U}\rvert + 1}{\lvert\Gamma^d_{inv}(t)\rvert}$

5.4.1 Experimental setup

Data and procedure

In our experiments, we use the same four Twitter networks and the Facebook dataset defined in Section 4, and we follow the same evaluation procedure described in Section 4.2: for each of the datasets, we have taken three subnetworks: the training network, the validation network and the test networks. We tune the hyperparameters for each method by taking the training set as input and the edges in the validation set as relevance judgements, and then, for evaluation, we take the union of the edges in the training and validation graphs as the input network for the algorithms, and the test set as relevance judgments for evaluation. In order to obtain the different subnetworks, we apply the same procedure as in Section 4.2. Also, to avoid trivializing the recommendation task on directed networks, reciprocal links to those included in the training networks are excluded from both the test and validation networks, as well as from the systems' output.

Recommendation algorithms

We assess the proposed IR model adaptation by comparing them to a selection of the most effective and representative algorithms in the link prediction and contact recommendation literature. Our selection includes, as friends of friends approaches, Adamic-Adar (Adamic and Adar, 2003), most common neighbors (Liben-Nowell and Kleinberg, 2007, Newman, 2001), Jaccard (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007) and cosine similarity (Lü and Zhou, 2011). As random walk representatives, we include personalized PageRank (White and Smyth, 2003) and the Money algorithm (Goel et al., 2015, Gupta et al., 2013) where, for simplicity, we include all the users in the circle of trust. Representing collaborative filtering algorithms, we include item-based and user-based kNN with cosine similarity (Ning et al., 2015) and implicit matrix factorization (Hu et al., 2008). Finally, we also include random and most-popular recommendation as sanity-check baselines.

Using the training and validation sets, we select the hyperparameters of the different algorithms (both edge orientation and parameter settings) by grid search, optimizing nDCG@10. (Järvelin and Kekäläinen, 2002). For those that can take advantage of edge weights (i.e. the adapted IR models and link prediction algorithms), we select the best option. For IR models, we show the parameter grid in Table C.3 and the optimal settings in Table C.4 in Appendix C. As the rest of approaches keep the optimal parameters for our experiments in Chapter 4, the reader can observe their optimal parameters in Table C.2.

5.4.2 General Results

In our experiments, we measure the behaviour of the IR models described in Section 5.3 when compared to the rest of the algorithms, in terms of nDCG and MAP at cutoff 10. We show the results in Table 5.6. To check whether the differences are significant, we use two-tailed paired t-tests at $p < 0.05$, and we complement them with Tukey honest significant differences (HSD) tests (Tukey, 1949) at $p < 0.05$ to deal with the fact that we are doing multiple comparisons, as suggested in the literature (Carterette, 2012, Sakai, 2018). We report the full outcome of these tests in Section D.2.2 of Appendix D. We observe that, in general, as expected, the Tukey HSD test is more conservative – the more conservative the higher the number of systems we compare. However, for our experiments, both test broadly agree on the most important conclusions. For that reason, when we make observations about statistical significance, unless otherwise specified, we do them for both tests.

A first observation of our results shows that IR models are quite effective for our task: for all five datasets, at least one model appears among the top 5 algorithms in the comparison, in terms of both nDCG@10 and

Table 5.6: Effectiveness of the IR model adaptations and baselines. Cell color goes from red (lower) to blue (higher values) for each metric/dataset, with the top value highlighted both underlined and bold. The differences between BM25 (the best IR model) and iMF (the best baseline) are always statistically significant under a simple two-tailed paired t-test at $p < 0.05$, excepting MAP@10 in the 200-tweets interactions network. In that network, the difference between both algorithms in terms of nDCG@10 is not significant under a more conservative Tukey HSD test. Full statistical significance tests are reported in figures in Appendix D.

		Twitter 1-month				Twitter 200-tweets				Facebook	
		Interactions		Follows		Interactions		Follows			
	Algorithm	nDCG	MAP	nDCG	MAP	nDCG	MAP	nDCG	MAP	nDCG	MAP
IR models	BM25	0.1042	0.0440	0.1177	0.0479	**0.1097**	**0.0583**	0.1159	0.0405	0.5731	0.3686
	BIR	0.0983	0.0399	0.1173	0.0475	0.1004	0.0522	0.1140	0.0389	0.5720	0.3670
	PL2	0.0962	0.0390	0.1184	0.0467	0.0983	0.0507	0.1166	0.0405	0.5712	0.3670
	DFRee	0.0849	0.0335	0.1112	0.0424	0.0976	0.0511	0.1164	0.0408	0.5601	0.3548
	Extreme BM25	0.0848	0.0326	0.1167	0.0472	0.1034	0.0543	0.1132	0.0390	0.5519	0.3480
	QLD	0.0767	0.0311	0.0985	0.0370	0.0937	0.0500	0.1152	0.0411	0.5799	0.3732
	QLJM	0.0882	0.0350	0.1154	0.0456	0.0694	0.0373	0.1161	0.0412	0.5664	0.3616
	QLL	0.0855	0.0338	0.1098	0.0410	0.0939	0.0497	0.1160	0.0399	0.5599	0.3564
	DPH	0.0806	0.0317	0.1063	0.0399	0.0976	0.0514	0.1159	0.0409	0.5568	0.3514
	DFReeKLIM	0.0709	0.0273	0.0927	0.0336	0.0953	0.0506	0.1138	0.0402	0.5565	0.3519
	DLH	0.0643	0.0246	0.0901	0.0325	0.0901	0.0480	0.1122	0.0396	0.5566	0.3526
	Pivoted Norm. VSM	0.0512	0.0194	0.0858	0.0295	0.0812	0.0436	0.0621	0.0198	0.5521	0.3481
	VSM	0.0292	0.0121	0.0503	0.0174	0.0425	0.0223	0.0787	0.0275	0.5237	0.3224
CF	iMF	**0.1388**	0.0663	**0.1462**	0.0590	0.1035	0.0558	**0.1329**	**0.0465**	0.5210	0.3207
	User-based kNN	0.1367	**0.0669**	0.1413	**0.0594**	0.0954	0.0510	0.1297	0.0462	0.5457	0.3491
	Item-based kNN	0.1174	0.0533	0.1296	0.0529	0.0724	0.0389	0.1205	0.0413	0.4542	0.2650
Contact rec.	Money	0.1315	0.0638	0.1129	0.0465	0.0932	0.0492	0.1131	0.0401	0.5867	0.3809
	Adamic-Adar	0.0981	0.0398	0.1175	0.0467	0.0997	0.0513	0.1140	0.0388	0.5746	0.3695
	MCN	0.0918	0.0368	0.1189	0.0467	0.0948	0.0493	0.1110	0.0373	0.5585	0.3546
	Pers. PageRank	0.0800	0.0315	0.0965	0.0362	0.0630	0.0317	0.0843	0.0276	**0.5891**	**0.3826**
	Jaccard	0.0317	0.0117	0.0543	0.0176	0.0571	0.0307	0.0848	0.0287	0.4913	0.2967
	Cosine	0.0393	0.0186	0.0497	0.0168	0.0480	0.0243	0.0768	0.0265	0.4943	0.2981
	Popularity	0.0572	0.0291	0.0449	0.0161	0.0422	0.0212	0.0397	0.0098	0.0523	0.0234
	Random	0.0014	0.0005	0.0011	0.0002	0.0003	0.0001	0.0018	0.0003	0.0030	0.0009

MAP@10. Among the different IR models, two of them stand out above the rest in terms of accuracy and robustness: BM25 and PL2.

Indeed, BM25 always appears in the top 6 of our comparison in terms of both metrics, even achieving the top accuracy for the interactions graph in the 200-tweets dataset. The only exception to this occurs in the comparison for the 200-tweets follows dataset. However, in that dataset, the difference with the fourth algorithm in the comparison, PL2, is not significant (as shown in Figure D.9 in Appendix D). One of the reasons to achieve such performance is likely the ability of BM25 to take advantage of the interaction frequency (the weights of the edges) better than any other algorithms, since, excepting VSM and cosine similarity, BM25 is the only algorithm that produces worse results when a non-binary representation is used.

The second approach, PL2, is the other consistent algorithm in the comparison, consistently appearing among the top 7 algorithms. To a lesser extent, BIR also provides competitive results in the comparison. On the contrary, the classical VSM model achieves the worst results for an IR model in our experiments, obtaining values close to the not personalized popularity-based recommendation. The rest of IR methods stand as mid-packers in the different studied datasets.

It is worth to notice that classical collaborative filtering algorithms represent a hard baseline to beat in the directed Twitter networks. This is particularly true for the implicit matrix factorization algorithm, which achieves top nDCG values in three of the Twitter graphs, and it is second-to-best in the remaining one. In terms of MAP@10, this algorithm is the best algorithm in the 200-tweets follows network, whereas it is reaches the second position for the remaining directed networks. However, in the Facebook network, the algorithm is far from optimal, even achieving worse results than the vector space model. This shows that iMF is unable to capture the undirected nature of the network when generating the recommendations as effectively as it does for the

directed Twitter networks. We can almost state the same for the user-based kNN approach, which obtains top performance values in the Twitter 1-month dataset and the Twitter 200-tweets follows network. However, in addition to the Facebook network, its performance also decays on the Twitter 200-tweets interactions one.

For the rest of algorithms, MCN, Adamic-Adar and item-based kNN behave as mid-packers in the different networks. Jaccard and cosine similarity perform poorly in comparison, similarly to the vector space models. As these two models are the ones that penalize the popularity of the candidate users the most, we observe that, in general, avoiding the popularity bias in the contact recommendation task does not lead us to good results. We can relate this to the so-called preferential attachment effect in social networks (Barabási and Albert, 1999), in which users with high degree are more likely to receive links by other users than users with low degree. We delve further into this in Chapter 6.

Overall, we observe that information retrieval models (and, particularly, BM25 and PL2) are highly competitive contact recommendation methods in different types of networks. However, BM25 is not the top algorithm, since iMF has a slight advantage in effectiveness. Based on previous experiences using iMF in different domains, we note that iMF works better with higher density of the dataset. This might explain the advantage of this model over IR models in the follows networks – a well as its better performance when we compare the 1-month and 200-tweets datasets. On the other hand, iMF seems to be sensitive to high network clustering degrees, as it occurs in the Facebook dataset, where this algorithm performs rather poorly. As we see later in Table 7.1 in Chapter 7, over 99% of the edges in the test network are only at distance 2 of the target user in this network (i.e. they are edges closing triads), in according to the high clustering coefficient reported in Table 4.1. While all the IR models can only recommend users at such distance, iMF can recommend very distant people in the network. In this dataset, it is unlikely to hit a link in the test set at distance greater than 2 from the target user. Other approaches like random walks can also travel long distances but their personalized teleportation parameters control how far they pick the recommendations. Depending on the (automatically tuned) parameter setting, random walks can gravitate as near the target user as it is most optimal. In networks such as Facebook where the clustering coefficient is high, this explains why these algorithms like personalized PageRank or Money work so well in this network.

As random walk algorithms and kNN obtain decent results in some datasets, but are quite suboptimal in the others, we can therefore say that BM25 is a decent second best in recommendation accuracy after matrix factorization. Hence, the IR models prove to be quite an effective option for contact recommendation. We find however, that BM25 has a great advantage over iMF in terms of computational cost and simplicity, which we explore in Section 5.5.

5.4.3 Neighborhood orientation

For the directed Twitter networks, we can also examine which neighbor orientation works best in the neighborhood-based and IR algorithms – whether users are better represented by their followers, their followees or both. To this end, we do the following: first, for each algorithm, we take the optimal hyperparameters shown in Table C.4. Then, we train nine different versions of the recommendation algorithms, selecting each possible link orientation for the target and candidate users, and we evaluate them over the test set using nDCG@10.

Figure 5.4 shows a detailed comparison of all combinations for this setting in the four directed graphs. The outer labels on the axis show the neighborhood orientation for the target user, and the inner ones for the candidate user. We can see that the undirected neighborhood Γ_{und} is consistently the most effective selection for the target users, whereas the incoming neighborhood Γ_{in} works best for the candidate users.

5.5 Complexity analysis

The success of a recommendation approach depends on many factors beyond the accuracy of the recommender. Some of these aspects are the computational cost and simplicity of the methods. When deployed in a commercial application, with thousands to millions of users and items, it might be critical for the system to generate the recommendations in real time. In this section, we analyze two aspects of the recommendation approaches we compare in section 5.4: a) the time needed for those algorithms to generate recommendations from scratch and b) the time needed for updating or retraining the algorithms when a new user or link is added to the network.

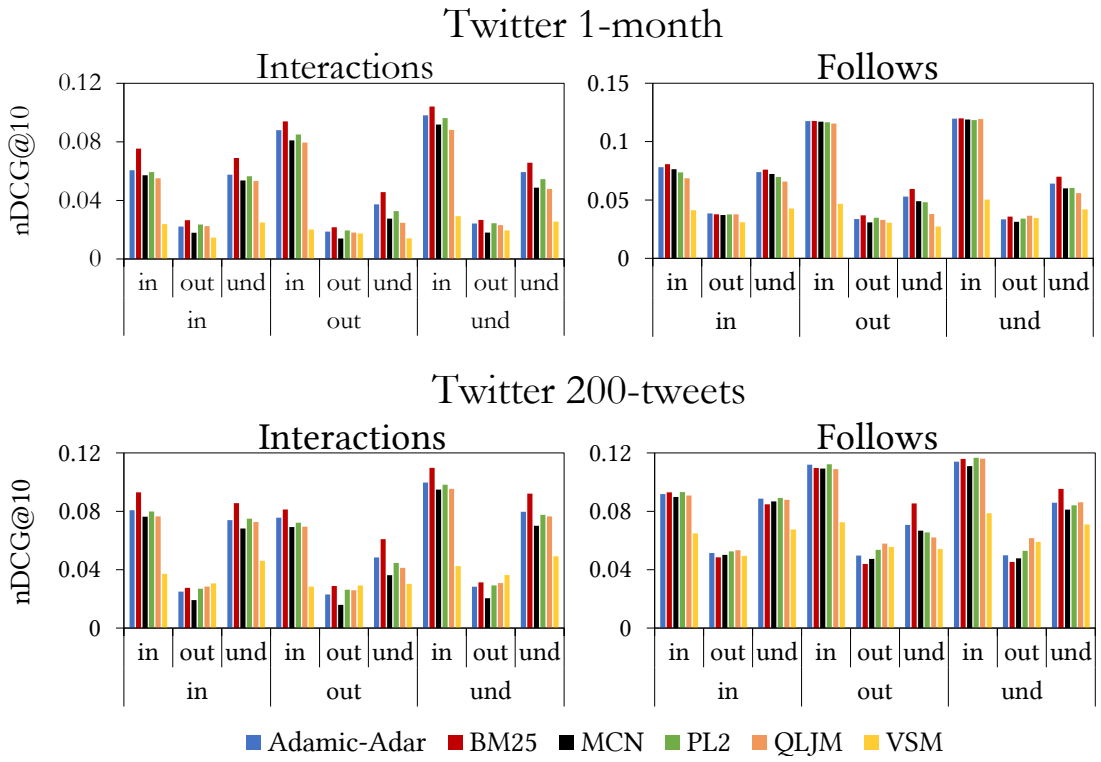


Figure 5.4: nDCG@10 values for the different possible choices for Γ^q and Γ^d on a selection of the most effective algorithms in the comparative included in Table 5.6.

We start by examining the theoretical analytical cost and then, we run a small test to observe the empirical differences between algorithms. We mostly focus on three algorithms: BM25 (as an effective representative of both IR models and friends of friends methods), user-based kNN and the implicit matrix factorization approach we use in our experiments (Hu et al., 2008).

5.5.1 Theoretical analysis

We report in Table 5.7 the complexity analysis for the algorithms tested in Section 5.4 grouped by families. We differentiate two complexity times: first, the theoretical time for training the algorithms from scratch, and, second, the theoretical time for generating the recommendation scores. We can see that IR models, along with other friends of friends approaches like MCN, Adamic-Adar and Jaccard, are the fastest recommendation methods, whereas implicit matrix factorization is among the costliest.

Friends of friends approaches (including IR models) obtain such a low cost since they can take advantage of the index-based optimizations developed for information retrieval tasks. For example, algorithms devised to obtain fast query responses such as the "term-at-a-time" or "document-at-a-time" approaches (Büttcher et al., 2010) can be applied to compute recommendation scores. As an example, we illustrate in Figure 5.5 the "term-at-a-time" approach: in that algorithm, we just run over the "posting lists" of the target user neighbors (that play here the role of the query terms), aggregating the partial scores for each candidate (in the figure, $f_u(v,t)$ just represents the score of the candidate user v if she only had t as common neighbor with the target user). As we run over the "posting lists" of all the neighbors of the target user, the resulting average complexity for these methods is $m_2 = \text{avg}_{u \in \mathcal{U}} |\Gamma(u)|^2$

In addition to this cost, for most algorithms (excluding MCN), we have to compute additional values, such as the length of the neighborhoods, or idf values which can be done in $O(|E|)$. These structures are the only ones stored in memory so, beyond the network we only need to store $O(|\mathcal{U}|)$ values. Finally, a new user or link appears in the network, we can easily update these values in $O(1)$ time without accuracy loss.

These algorithms can be exploited twice in user-based and item-based nearest neighbors to compute a) the similarities between users and b) the recommendation scores. First, we can use the fast query response algorithms to compute the similarities in the same way as we do for the recommendation scores in friends of friends

Table 5.7: Running time complexity of the different algorithms, grouped by families. We show the complexity for both the full training and the recommendation scores computation for all the users (excluding the additional $\log N$ for rank sorting). The variable m denotes the average degree of the network, m_2 the average of the squares of the degrees and c is the number of iterations for computing the personalized PageRank and Money scores and the latent factors for iMF; and k represents the number of latent factors in iMF and the number of neighbors in kNN.

Algorithms	Training	Recommendation
IR models	$O(\lvert E\rvert)$	$O(\lvert\mathcal{U}\rvert m_2)$
Jaccard, Adamic-Adar	$O(\lvert E\rvert)$	$O(\lvert\mathcal{U}\rvert m_2)$
MCN	-	$O(\lvert\mathcal{U}\rvert m_2)$
Random walks	$O\left(c\lvert\mathcal{U}\rvert^2 + c\lvert\mathcal{U}\rvert\lvert E\rvert\right)$	$O(\lvert\mathcal{U}\rvert)$
User/Item-based kNN	$O(\lvert E\rvert)$	$O\left(\lvert\mathcal{U}\rvert\left(m_2\log(k) + km\right)\right)$
Matrix factorization	$O\left(c\left(k^2\lvert E\rvert + k^3\lvert\mathcal{U}\rvert\right)\right)$	$O\left(\lvert\mathcal{U}\rvert^2 k\right)$
Popularity	$O(\lvert E\rvert)$	$O(\lvert\mathcal{U}\rvert)$
Random	-	$O\left(\lvert\mathcal{U}\rvert^2\right)$

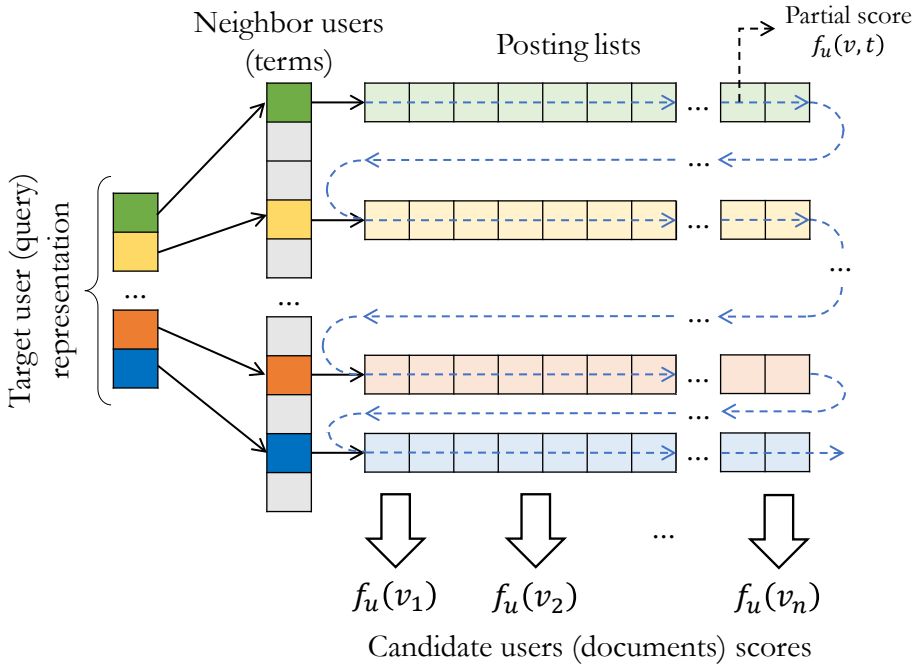
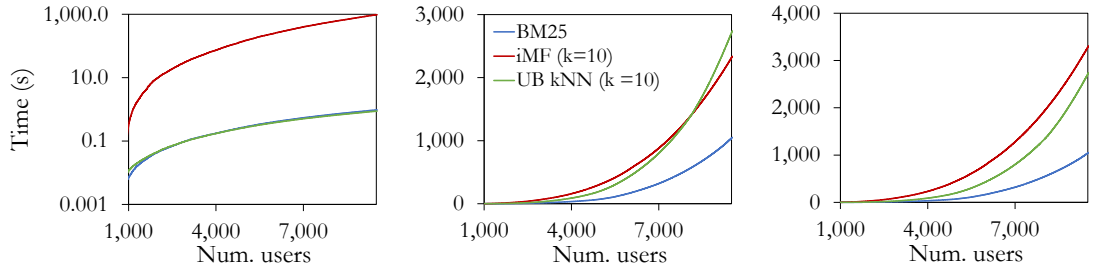


Figure 5.5: Term-at-a-time fast query response algorithm (Büttcher et al., 2010)

approaches (we can observe that, for instance, cosine similarity can be applied both as a similarity and a recommendation algorithm – we explore this in more detail in Chapter 7). Considering this, and using structures like ranking heaps, we can obtain the neighbors for all the users in $O\left(\lvert E\rvert + m_2 \log k\right)$ where k is the size of the neighborhood. Then, we can use those k most similar users (instead of the actual neighbors in the network) as the target user representation to apply, again, that algorithm to generate the final recommendation scores, with $O(mk)$ average complexity, with $m = \text{avg}_{u\in\mathcal{U}}\,\lvert\Gamma(u)\rvert$.

In Table 5.7 we distribute the cost as follows: in training time we just store values such as the length of the neighbors, whereas, in recommendation time, we compute the similarities, neighbors and final recommendation scores. Although this increases the recommendation time, it has advantages over both incremental updates and memory consumption: adding the overhead of computing similarities and neighborhoods to recommendation time, we can update without loss the necessary values for the algorithm in $O(1)$ time, and we only need to store $O(\lvert\mathcal{U}\rvert)$ values (user lengths, idf values or norms depending on the chosen similarity).

Implicit matrix factorization, on its side, has a higher recommendation cost than the other two approaches (for each user, its $O(\lvert\mathcal{U}\rvert k)$), since, in general, $\lvert\mathcal{U}\rvert$ is much larger than the average degree (for example, in the Twitter 1-month interaction training network, there are $9,528$ users, whereas the average degree is smaller than



a) Update time b) Recommendation time c) Total time

Figure 5.6: Time comparison between BM25, user-based kNN and implicit matrix factorization. a) shows the time needed for updating the recommenders, b) the time needed for generating the recommendation and c) the total time.

20). In terms of training, it is very expensive to compute the latent factor matrices ($O\left(c(k^2|E| + k^3|\mathcal{U}|\right)$ where k is the number of latent factors and c is the number of alternate least squares iterations needed for their convergence. In terms of memory spending, iMF needs $2k^2|\mathcal{U}|$ decimal values, in comparison with the $O(|\mathcal{U}|)$ values needed by BM25 or user-based kNN.

Moreover, matrix factorization is not particularly flexible for incremental updates. In order to obtain lossless updates, we would need to retrain the algorithm from the beginning each time we receive new users or links in the networks. There are other solutions for incremental update, like the one by Yu et al. (2016), by which new like can be added in $O(mk^2 + k^3)$ time. However, such solutions do not work as exact retraining, and they come at the expense of incremental accuracy losses in the updated model.

5.5.2 Empirical observation

In order to observe how this theoretical analysis translates to reality, we carry out an experiment where we test the running times for BM25, iMF and user-based kNN. Taking the 1-month interaction network, we randomly sample 10% of the users in the training network, and we take all the links between them. Taking this network as the starting point of the algorithm, we train the algorithm and run the recommendation algorithms. Then, iteratively, we add one of the remaining 90% and we add all its edges pointing to or from the users in the growing network. Then, we update the recommenders and generate the recommendations. We compute separately the time taken to update the recommender, and the time spent in generating the recommendations.

For the experiment, we use the best version of BM25 for that dataset, using the "term-at-a-time" algorithm for computing the recommendation scores as fast as possible, against implicit matrix factorization and user-based kNN configured with $k = 10$ (respectively, factors and neighbors). In implicit matrix factorization, we apply the incremental update with accuracy losses proposed by Yu et al. (2016).

We show the results of our experiment in Figure 5.6. In that figure, x axis shows the number of users in the network, while y axis shows the cumulative update time for Figure 5.6a, the cumulative recommendation time for Figure 5.6b and the total time (the sum of the two previous times) in Figure 5.6c. We observe that, for all three plots, BM25 achieves the smaller times, showing that, as expected, BM25 is the fastest algorithm by far. The only apparent exception is the update time, where it has a tie with user-based kNN, as expected, since they have to apply the same operations in the update.

On the other hand, iMF achieves the worst results for the update times, even with the two applied optimizations: the fast incremental update and the fact that we only use 10 latent factors. If we incremented that number of latent factors (as our parameter selection indicates) we would expect a cubic growth of the time needed to update the recommender as a function of k. In terms of recommendation time, it is close (and even faster for large number of users in the network) to user-based kNN, thanks to all the additional computations user-based kNN must do in order to find the neighbors If we incremented the number of latent factors for iMF or the number of nearest neighbors for user-based kNN, we would expect here a linear growth for both algorithms. Finally, even when the overall recommendation time is larger than the update time for all the algorithms, when we aggregate the update and recommendation times, iMF is still the slowest. The difference with user-based

kNN is not large for $k = 10$, but, due to the cost of updating iMF, we expect that difference to largely grow as k grows in favor of user-based kNN.

5.6 Conclusions

Even though they are considered to be separate fields (and research on them has been mostly developed as if they were), text-based search and recommendation are closely related tasks. Prior research has address this relation on the general perspective of adapting IR techniques to item recommendation (Adomavicius and Tuzhilin, 2005, Bellogín et al., 2013, Parapar et al., 2013, Wang et al., 2008a,b). In the work presented in this chapter, we particularize this adaptation to the recommendation of people in social networks. We summarize next the findings of this chapter:

- Information retrieval models can be applied as friends of friends contact recommendation algorithms by mapping the three spaces in the IR task (queries, documents and terms) to a unique space in the people recommendation one: the space of users in the social network.
- IR models provide effective solutions for the contact recommendation task. In particular, the BM25 model (Robertson and Zaragoza, 2009) achieves competitive accuracy results when compared to the best state of the art approaches.
- In terms of efficiency, friends of friends methods (and information retrieval models as a particularly interesting case) are able to run and update several orders of magnitude faster than most contact recommendation approaches by taking advantage of techniques devised for search engines.
- For IR and friends of friends methods, target users are best characterized by their undirected neighbors, whereas the best representation for the candidate users takes their incoming ones.

Compared to alternative solutions, the translation of new and principled IR models and their adaptation to recommend people in social networks can add new and deeper insights to how we understand and solve the contact recommendation task, allowing us to import the theory and foundations upon which the IR models were developed, as well as other techniques from the text search field. In this chapter, we have made a first step in that direction, by adapting the models, but many other techniques could be translated to the contact recommendation field, such as relevance feedback, learning to rank or axiomatic thinking. We explore some of them in Chapters 6 and 7. Reciprocally, this adaptation can contribute to expand the meaning, interpretation and usefulness of the IR models in different tasks, giving them higher levels of abstraction.

6
Axiomatic analysis

Highlights

- *Our mapping preserves the original properties of IR models when adapted to the contact recommendation task.*
- *Term frequencies and term discrimination fundamental IR axioms seem to have a positive impact on the accuracy of friends of friends contact recommendation approaches.*
- *Length normalization axioms interfere with the preferential attachment phenomenon and seem to deteriorate the accuracy of friends of friends contact recommendation approaches.*

The experimental results we report in Chapter 5 show that the adaptation of IR models to recommend new users to connect in social networks is not only possible, but also effective (specially in the case of the BM25 model) and efficient. So, in this chapter, taking the adaption of such approaches as a basis, we wanted to go a step further in our analysis of the relationship between contact recommendation and text search, by adapting techniques from axiomatic thinking in IR to contact recommendation.

In text IR, axiomatic analysis has provided several heuristics (known as axioms) which can be used for developing new and effective approaches, as well as for diagnosing and explaining why an existing (or novel) system provides (or not) good search results. As a first step to providing such tools for the contact recommendation task, we study the validity and importance of the fundamental text search axioms proposed by Fang et al. (2004, 2011). We first give a meaning to those properties under the contact recommendation setting and then we analyze whether our mapping preserves the constraints when we adapt the models to recommendation approaches. Finally, we do some experiments to identify whether satisfying the axioms has a positive impact on the accuracy of the algorithms or not.

The work introduced in this chapter was partially published in the following article:

- **Sanz-Cruzado et al. (2020b):** Javier Sanz-Cruzado, Craig Macdonald, Iadh Ounis, and Pablo Castells. Axiomatic Analysis of Contact Recommendation Methods in Social Networks: an IR perspective. In *Proceedings of the 42nd European Conference on Information Retrieval* (ECIR 2020), number 12035 in LNCS,pages 175–190. Springer, April 2020.

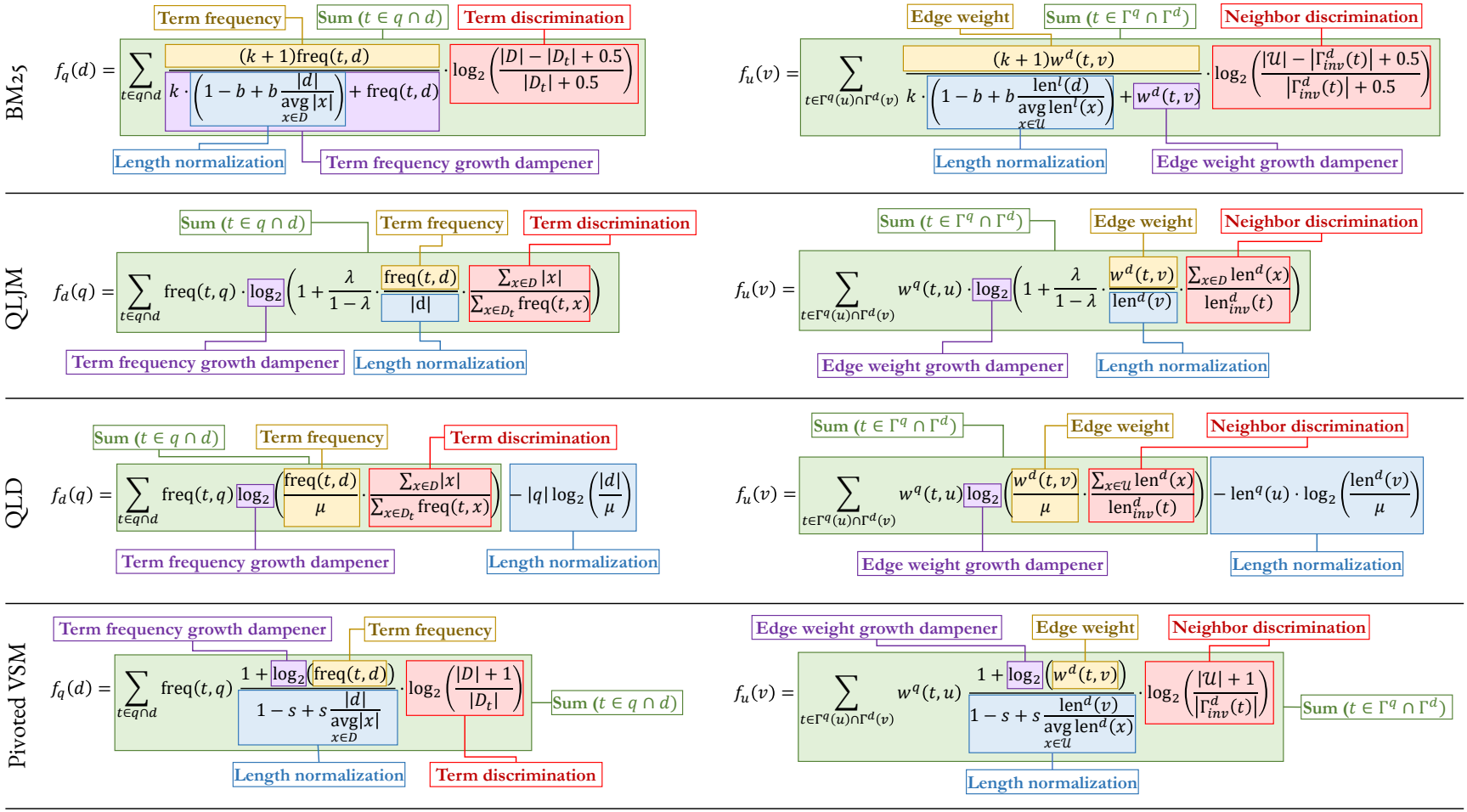
6.1 Research questions

The contents of this chapter address the following research questions:

- **RQ1:** Does the mapping in Chapter 5 preserve the fundamental properties of the IR models when they are translated to the recommendation task? If it does not, how do these properties change?
- **RQ2:** From the studied set of axioms, which of them are related to an improvement of the accuracy of contact recommendation approaches?

6.2 Fundamental IR axioms

As we state in Section 2.2.3, axiomatic thinking in IR has as its main objective to provide a set of valid heuristics (known as axioms) that any effective information retrieval model should adhere to in order to provide good search results. Although following all of these axioms does not ensure good performance, following these properties often improves the accuracy of the systems (Fang et al., 2004, 2011).



(a) Text search

(b) Contact recommendation

Figure 6.1: Identification of the different elements of IR models for (a) the text search task and (b) the contact recommendation task.

The first set of IR axioms (what we know as the fundamental or basic IR axioms) was proposed and later expanded by Fang et al. (2004, 2011), who focused on the common elements that most IR models share. We illustrate these common elements in Figure 6.1(a), where we highlight the different parts of four IR models: BM25 (Robertson and Zaragoza, 2009), QLJM (Ponte and Croft, 1998, Zhai and Lafferty, 2004), QLD (Ponte and Croft, 1998, Zhai and Lafferty, 2004) and the pivoted length normalization VSM (Singhal et al., 1998). Elements in different models sharing the same color play an equivalent role in all of them.

By observing that figure, the first thing we notice is that all the models can be expressed as a sum of the terms shared between the query and the document (as highlighted in green). Also, a document increases the score as the frequency of a query term increases (as it is shown in yellow), but this growth is dampened so the growth of this score is sublinear (as the purple boxes illustrate). In addition, the more discriminative terms (those appearing only in a few documents in the collection) are given a greater importance (in red), and, finally, the scores are normalized by the length of the document to retrieve (as indicated in blue).

These five structural elements are the seed for the definition of the fundamental set of axioms (Fang et al., 2004, 2011), which we study in this chapter for the contact recommendation task. As it is illustrated in Figure 6.1(b), the different elements in the formulation of IR models for search task have their equivalences with their formulations for contact recommendation. Considering the mapping introduced in Chapter 5, in contact recommendation, instead of expressing the model as the sum over the common terms between the query and the document, we do it as the sum over the common neighbors between the target and candidate users. Both elements related to term frequency are translated to elements related with the equivalent in our mapping: the weight of the edges. Instead of considering the discriminative power of the terms, we consider the discriminative power of the neighbors. Finally, the length normalization term stays the same, by defining it as the sum of the weights of the edges between the candidate user and her neighbors.

Considering all the previous observations, as a first step towards understanding how important IR axioms are for providing good people recommendations in social networks, we enunciate the different fundamental axioms, adapt them to contact recommendation and explore their possible meaning in this field. Following the original categorization, we divide the fundamental axioms in four categories: term frequency, term discrimination, length normalization constraints and constraints combining term frequency and length normalization.

6.2.1 Term frequency constraints (TFCs)

Known as the term frequency constraints (TFCs), the first family of axioms analyzes what is the role of the frequency of the query terms present in the retrieved documents. As the term frequencies are represented by edge weights in our framework, for our reformulation of the axioms, we rename this heuristics as "edge weight constraints" (EWCs). There are three different axioms in this family.

First term frequency constraint (TFC1)

The first constraint, TFC1, establishes that, if the only different between two documents is the frequency of a query term, then, the document with the higher term frequency should be ranked in a higher position than the other. Formally (Fang et al., 2011):

TFC1: Let $q = \{t\}$ be a query with a single term, t, and d_1, d_2 two documents such that $|d_1| = |d_2|$ and $\text{freq}(t, d_1) > \text{freq}(t, d_2)$. Then, we should observe that $f_q(d_1) > f_q(d_2)$.

The intuition behind this axiom is naturally translated to contact recommendation by considering the "common friends" principle in social relationships. All things being equal, a user is more likely to connect to people who have stronger bonds to her common friends. This can be formalized for contact recommendation as:

EWC1: If the target user u has a single neighbor $\Gamma^q(u) = \{t\}$, and we have two different candidate users v_1, v_2 such that $\text{len}^l(v_1) = \text{len}^l(v_2)$, and $w^d(t, v_1) > w^d(t, v_2)$, then we should have $f_u(v_1) > f_u(v_2)$.

We illustrate an example of a graph where the base conditions for the EWC1 are met in Figure 6.2. As we can observe, in that example, the only difference between both candidate users v_1, v_2 is the weight of the

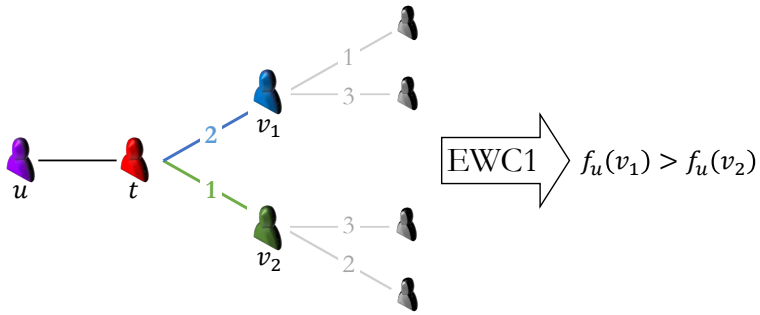


Figure 6.2: EWC1 example. User u (in purple) represents the target user, whereas users v_1 (in blue) and v_2 (in green) represent the candidate users. Grey users represent other neighbors of v_1 and v_2. We highlight in blue the weights and edges that impact on this axiom for v_1 and in green for v_2. Both candidate users have the same length ($\text{len}(v_i) = 6$). Since other neighbors of the intermediate user t (in red) do not influence this, we omit them.

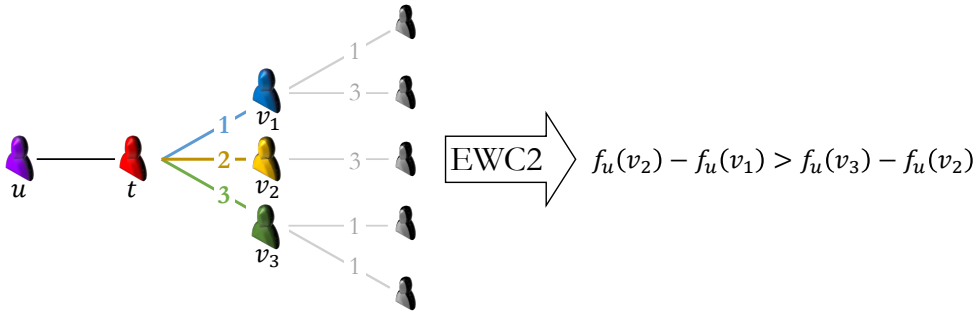


Figure 6.3: EWC2 example. User u (in purple) represents the target user, whereas users v_1 (in blue), v_2 (in yellow) and v_3 (in green) represent the candidate users. Grey users represent other neighbors of v_1,v_2 and v_3. We highlight in blue the weights and edges that impact on this axiom for v_1, in yellow those for v_2, and in green those for v_3. All candidate users have the same length ($\text{len}(v_i) = 5$). Since other neighbors of the intermediate user t (in red) do not influence this, we omit them.

common neighbor for each of them. Following Figure 6.1, this axiom would be related to the term frequency term highlighted in yellow.

Second term frequency constraint (TFC2)

The second term frequency constraint (TFC2) establishes that the ranking score increment produced by increasing the term frequency should decrease with the frequency (i.e. the growth of ranking scores should be dampened on term frequency, as in a diminishing returns pattern). Formally (Fang et al., 2011):

TFC2: Let $q = \{t\}$ be a query with a single term, t, and d_1, d_2, d_3 three documents such that $|d_1| = |d_2| = |d_3|$. If $\text{freq}(t, d_3) = \text{freq}(t, d_2) + 1$ and $\text{freq}(t, d_2) = \text{freq}(t, d_1) + 1$ then, $f_q(d_2) - f_q(d_1) > f_q(d_3) - f_q(d_2)$.

In contact recommendation, this has also a direct meaning: the difference between the scores of two candidate contacts should decrease with the weights of the common friends they have with the target user. Formally, we express this constraint as:

EWC2: For a target user u with a single neighbor $\Gamma^q(u) = \{t\}$, and three candidate users v_1, v_2, v_3 such that $\text{len}^l(v_1) = \text{len}^l(v_2) = \text{len}^l(v_3)$, and $w^d(t, v_3) = w^d(t, v_2) + 1$ and $w^d(t, v_2) = w^d(t, v_1) + 1$, then $f_u(v_2) - f_u(v_1) > f_u(v_3) - f_u(v_2)$.

An example of initial conditions of the constraint is shown in Figure 6.3, where we have three candidate users which have, as their only difference, the weight $w^d(t, v_i)$, and $w^d(t, v_{i+1}) - w^d(t, v_i) = 1$. In Figure 6.1,

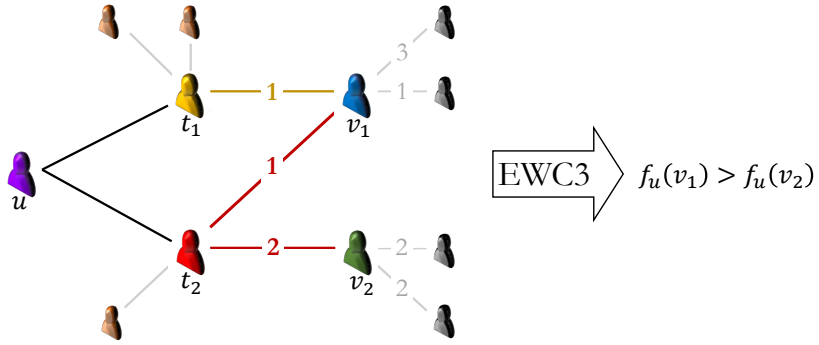


Figure 6.4: EWC3 example. User u (in purple) represents the target user, whereas users v_1 (in blue) and v_2 (in green) represent the candidate users. Grey users represent other neighbors of v_1 and v_2, and orange users represent the neighbors the common users t_1 (in yellow) and t_2 (in red). Both common users have the same number of neighbors (4), and both candidate users have the same length ($\text{len}(v_i) = 6$). We highlight in yellow the weights and edges which are influenced by t_1 and in red those of t_2.

the term frequency growth dampeners highlighted in purple illustrate example elements of IR models oriented to satisfy this axiom.

Third term frequency constraint (TFC3)

The third and final term frequency axiom (TFC3) reflects the following property: being the total occurrences of query terms equal between two documents, and being the discriminative power of all the terms also equal, the document that covers more distinct terms should attain a higher score. If we define $\text{td}(t)$ as a measure of how informative the terms appearing in both the query and the document are (similarly to the term discrimination component of various models, such as idf in VSM or RSJ in BM25), we can formally express this as (Fang et al., 2011):

TFC3: Let t_1, t_2 two terms of the query q ($\{t_1, t_2\} \subset q$) such that $\text{td}(t_1) = \text{td}(t_2)$. Given two documents d_1, d_2 such that $|d_1| = |d_2|$, if $\text{freq}(t_2, d_2) = \text{freq}(t_1, d_1) + \text{freq}(t_2, d_1)$, $t_1 \notin d_2$ and $\{t_1, t_2\} \subset d_1$, then, $f_q(d_1) > f_q(d_2)$.

In the people recommendation space, this translates to the triadic closure principle (Newman, 2001, 2018): given that all other things (length, users' discriminative power, sum of the edge weights of the common users) are equal, the more common friends a candidate contact has with the target user, the higher the chance that a new link between them exists. Formally:

EWC3: Let $\{t_1, t_2\} \subset \Gamma^q(u)$ be two neighbors of target user u, with $\text{td}(t_1) = \text{td}(t_2)$. Given two candidate users v_1, v_2 with $\text{len}^l(v_1) = \text{len}^l(v_2)$, if $w^d(t_2, v_2) = w^d(t_1, v_1) + w^d(t_2, v_1)$, $t_1 \notin \Gamma^d(v_2)$, and $\{t_1, t_2\} \subset \Gamma^d(v_1)$, then $f_u(v_1) > f_u(v_2)$.

where $\text{td}(t)$ represents the discriminative power of user t, similarly to how this element works in text search.

The fact that IR models can be expressed as the sum of common terms between the query and the document (as it is highlighted in green in Figure 6.1), along the term frequency dampener (in purple) are what make possible for IR models to satisfy this constraint. Figure 6.4 illustrates a case example where the conditions of the axiom would be satisfied if the algorithms adhere to it. As we can observe in that figure, we show two candidate users with the same length, v_1 and v_2, and two neighbors of the target user u: t_1 and t_2, which have both the same number of neighbors. Then, the first candidate user shares both neighbors with u, whereas the second one only shares t_2, with weight $w^d(t_2, v_2)$ equal to the sum of the weights of both users in v_1's neighborhood.

Relation between the TFCs

An interesting observation about the three TFC axioms is that they are not independent: if we take $\Gamma^q(u) = \{t\}$ and we fix the values for $\text{td}(t)$ and $\text{len}^l(v)$, we could rewrite $f_u(v)$ as a function of the term frequency in the

document, $f_u(w^d(v,t))$. If $f_u(w^d(v,t))$ is a positive function ($f(x) > 0 \forall x$), it is easy to see that EWC1 $\Leftrightarrow$ $f_u(w^d(v,t))$ is an increasing function, EWC2 $\Leftrightarrow$ $f_u(w^d(v,t))$ is strictly concave and EWC3 $\Leftrightarrow$ $f_u(w^d(v,t))$ is subadditive ($f(u+v) < f(u) + f(v)$).

Given a function g that is both positive and concave, then, g is increasing and subadditive. Therefore, for such functions (as is the case for the ranking functions of most IR models), adhering to the second edge weight constraint EWC2 would imply that the first and third constraints are true (EWC2 $\Rightarrow$ EWC1 $\wedge$ EWC3). However, if EWC2 is not satisfied, either EWC1 or EWC3 could still be satisfied.

6.2.2 Term discrimination constraint (TDC)

The term discrimination constraint (TDC) formalizes the intuition that penalizing popular words appearing in a large fraction of the documents (such as stopwords) and assigning higher weights to more discriminative query terms should improve the quality of the search results. For instance, if we look for the query "The prince of Persia" on a Web search engine, the terms "the" and "of" do not help filtering the results, since they appear in almost every webpage written in English. On the other hand, "prince" and "Persia" provide a much more useful identification of the set of documents we are looking for. Formally, this axiom is defined as (Shi et al., 2005):

TDC: Let q be a query with two terms, $q = \{t_1, t_2\}$. Given two $\{t_1, t_2\} \subset \Gamma^q(u)$ be two neighbors of target user u, with $\text{td}(t_1) = \text{td}(t_2)$. Given two documents d_1, d_2 such that $\text{freq}(t_1, d_1) = \text{freq}(t_2, d_2)$ and $\text{freq}(t_2, d_1) = \text{freq}(t_1, d_2)$, if $\text{freq}(t_1, d_1) > \text{freq}(t_2, d_1)$ and $\text{td}(t_1) > \text{td}(t_2)$, then $f_q(d_1) > f_q(d_2)$.

Differently from the rest of the axioms, whose formal definitions are extracted from Fang et al. (2011), we adapt the version of this heuristic proposed by Shi et al. (2005), since it has a more direct adaptation for the contact recommendation field. Fang et al. (2011) version is more general, but its meaning does not make full sense for recommending people.

The principles behind this constraint also make sense in contact recommendation: sharing a very popular and highly connected friend (e.g. two people following Katy Perry or Barack Obama on Twitter) might be a weak signal to infer a new relation. A less social common friend, however, may suggest that the two people may have more interests in common, or that they belong to the same circle of friends. This idea has been taken as the basis of contact recommendation algorithms such as Adamic-Adar (Adamic and Adar, 2003) or resource allocation (Zhou et al., 2009).

As in contact recommendation, the elements that discriminate the candidate users are the common neighbors, we rename the axiom as the "neighbor discrimination constraint" (NDC), which we formally define as:

NDC: Let u be the target user, with $\Gamma^q(u) = \{t_1, t_2\}$. Given two candidate users v_1, v_2 where $\text{len}^l(v_1) = \text{len}^l(v_2)$, and $w^d(t_1, v_1) = w^d(t_2, v_2)$ and $w^d(t_2, v_1) = w^d(t_1, v_2)$, if $w^d(t_1, v_1) > w^d(t_2, v_1)$ and $\text{td}(t_1) > \text{td}(t_2)$, then $f_u(v_1) > f_u(v_2)$.

We provide an example of the conditions of these axiom in Figure 6.5. In that figure, we can observe two candidate users v_1, v_2 with the same length that share the same two common neighbors t_1, t_2 with the target user u. The distribution of weights is similar (both candidate users share a neighbor with weight equal to 1 and another one equal to 2), but t_2 has a smaller discriminative power, since it has one more neighbor than t_1.

The term discrimination elements highlighted in red in Figure 6.1 illustrate some examples of elements in IR models which may be used to adhere to this property.

6.2.3 Length normalization constraints (LNCs)

The third family of IR axioms, the length normalization constraints LNC studies how ranking functions should consider using the length of the documents to improve the search results. In the mapping defined in Chapter 5, the document length is translated to the sum of the edge weights between the candidate user and v its neighbors: $\text{len}^l(v)$. All these axioms are related to the elements highlighted in blue in Figure 6.1.

As the LNC axioms only study the length of the documents, and only the candidate users play the role of the documents in our framework, we rename this axioms as "candidate length normalization constraints" (CLNCs) for the people recommendation task. There are two different length normalization axioms.

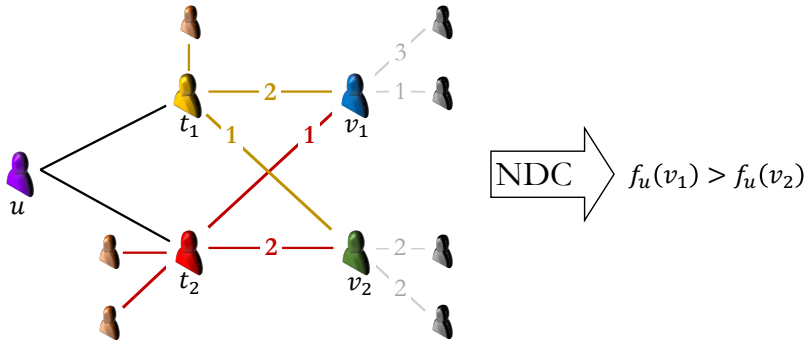


Figure 6.5: NDC example. User u (in purple) represents the target user, whereas users v_1 (in blue) and v_2 (in green) represent the candidate users. Grey users represent other neighbors of v_1 and v_2, and orange users represent the neighbors the common users t_1 (in yellow) and t_2 (in red). Both candidate users have the same length ($\text{len}^l(v_i) = 7$), and t_2 has one more neighbor than t_1. It is also observed that $w(t_1, v_1) = w(t_2, v_2) = 2$ and $w(t_1, v_2) = w(t_2, v_1) = 1$. We highlight in yellow the edges that t_1 shares with her neighbors, and in red the links between t_2 and her contacts.

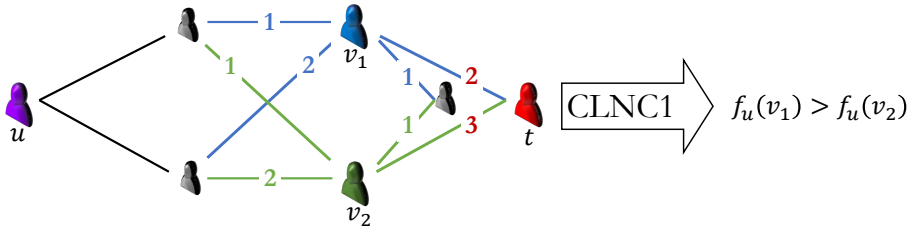


Figure 6.6: CLNC1 example. User u (in purple) represents the target user, whereas users v_1 (in blue) and v_2 (in green) represent the candidate users. We represent in red a neighbor t of both candidate users (but not a neighbor of the target user u), which has different weight for v_1 than for v_2. Grey users represent the rest of neighbors of v_1 and v_2 (including the ones shared with the target user). We highlight in blue the edges and weights that contribute to the length of user v_1 and in green those which contribute to the length of v_2. Weights in red represent the differences between $\text{len}(v_1)$ and $\text{len}(v_2)$. We omit other neighbors of u and the rest of the users, since they are not relevant for the example.

First lenght normalization constraint (LNC1)

The first length normalization axiom (LNC1) states that, if two documents have the same number of occurrences for all the terms in the query, we should choose the shorter one, since it contains the least amount of query-unrelated information. Formally, this is translated to (Fang et al., 2011):

LNC1: Let q be a query, and d_1, d_2 two documents such that $\text{freq}(t, d_2) > \text{freq}(t, d_1)$ for a term $t \notin q$, but $\text{freq}(x, d_1) = \text{freq}(x, d_2)$ for any other term $x \neq t$ (including the query terms). Then, $f_q(d_1) > f_q(d_2)$.

In the contact recommendation domain, this axiom implies penalizing popular, highly connected candidate users which have many neighbors not shared with the target user. We hence reformulate the axiom as:

CLNC1: Given a target user u and two candidate users v_1, v_2, if $w^d(t, v_2) > w^d(t, v_1)$ for some user $t \notin \Gamma^q(u)$, but $w^d(x, v_1) = w^d(x, v_2)$ for any other user $x \neq t$, then $f_u(v_1) > f_u(v_2)$.

We show an example of this constraint conditions in Figure 6.6, where we show two candidate users, who share the same weights between them and the rest of users, except for one, t (causing v_2 to have a larger length), who is not a neighbor of the target user.

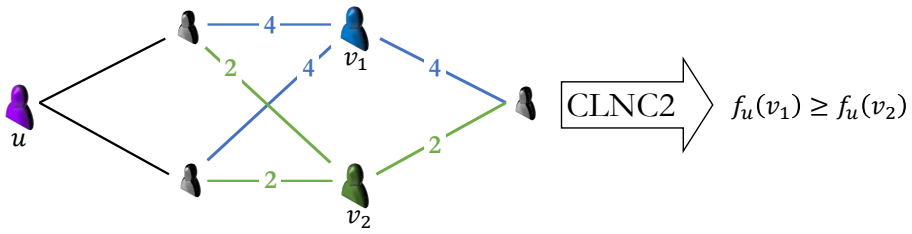


Figure 6.7: CLNC2 example. User u (in purple) represents the target user, whereas users v_1 (in blue) and v_2 (in green) represent the candidate users. Grey users represent the rest of neighbors of v_1 and v_2 (including the ones shared with the target user). We highlight in blue the edges and weights that contribute to the length of user v_1 and in green those which contribute to the length of v_2. For all users $t \in \mathcal{U}$, $w(t, v_1) = 2 \cdot w(t, v_2)$. We omit other neighbors of u and the rest of the users, since they are not relevant for the example.

Second length normalization constraint (LNC2)

The second length normalization constraint (LNC2) aims to avoid over-penalizing long documents. It states that, if we concatenate a document to itself multiple times, the resulting document should obtain a greater or equal score than the original one. Formally (Fang et al., 2011):

LNC2: If two documents d_1, d_2 are such that $\text{freq}(x, d_1) = k \cdot \text{freq}(x, d_2)$ for all terms x and some constant $k > 1$, and $\text{freq}(t, d_1) > 0$ for some term t that belongs to the query q ($t \in q$), then, $f_q(d_1) \geq f_q(d_2)$.

In contact recommendation, this means that, if we multiply all the edge weights of a candidate user by a positive number (greater than 1), the score for the candidate user should not decrease. This is translated as:

CLNC2: If two candidate users v_1, v_2 are such that $w^d(x, v_1) = k \cdot w^d(x, v_2)$ for all users x and some constant $k > 1$, and $w^d(t, v_1) > 0$ for some neighbor $t \in \Gamma^q(u)$ of the target user u, then we have $f_u(v_1) \geq f_u(v_2)$.

In the particular case where, in directed networks, $\Gamma^d := \Gamma_{und}$, multiplying by k the weights of the edges, in this context, means multiplying both the weights for the incoming and outgoing edges by k (not adding new edges or allowing different coefficients for the incoming or outgoing edges). We illustrate this on Figure 6.7, where, for two candidates users, v_1 and v_2, we duplicate the value of the weights for v_1 with respect to v_2.

6.2.4 Term frequency - length normalization constraint (TF-LNC)

The last heuristic (TF-LNC) aims to provide a balance between query term frequency in documents and length normalization. The axiom states that if we add more occurrences of a query term to a document, its score should increase, as a larger fraction of the document contains relevant terms. It is formally formulated as (Fang et al., 2011):

TF-LNC: Given a query q with a single term t, $q = \{t\}$, if two documents d_1 and d_2 are such that $\text{freq}(t, d_1) > \text{freq}(t, d_2)$ and $|d_1| = |d_2| + \text{freq}(t, d_1) - \text{freq}(t, d_2)$, then, $f_q(d_1) > f_q(d_2)$.

For contact recommendation, where we rename this axiom as the "edge weight - candidate length normalization constraint" (EW-CLNC), the intuition is similar: if the weight of the link between two users v and t increases, then v's score as a candidate for target users having t in their neighborhood should also increase. This axiom is then expressed as follows:

EW-CLNC: Given a target user u with a single neighbor $\Gamma^q(u) = \{t\}$, if two candidates v_1 and v_2 are such that $w(t, v_1) > w(t, v_2)$ and $\text{len}(v_1) = \text{len}(v_2) + w(t, v_1) - w(t, v_2)$, then $f_u(v_1) > f_u(v_2)$

Similarly to the rest of the axioms, Figure 6.8 shows an example of the EW-CLNC constraint initial conditions. We show two candidate users v_1, v_2 which differ on the weight of the common user t with the target user u.

Finally, to summarize the relation between the IR and contact recommendation heuristics introduced in this section, we map both families of constraints in Table 6.1.

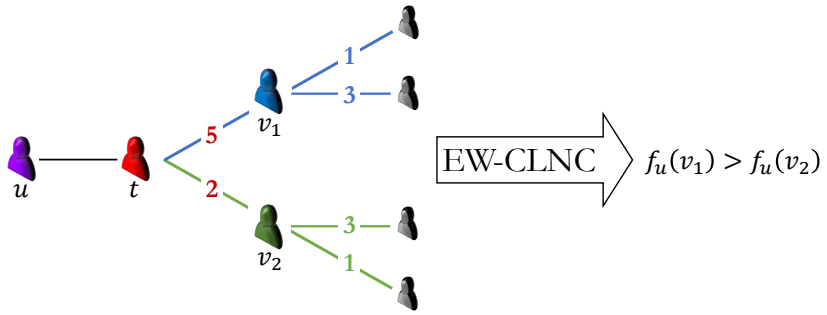


Figure 6.8: EW-CLNC example. User u (in purple) represents the target user, whereas users v_1 (in blue) and v_2 (in green) represent the candidate users. Grey users represent other neighbors of v_1 and v_2. We highlight in blue the weights and edges that impact on the length of v_1 and in green those of v_2. We also highlight in red the intermediate user between the target and candidate users, t, and the weights $w(t, v_i)$ (noticing that the weight for v_1 is greater than the one for v_2). The sum of weights of other users for v_1 and v_2 is the same ($\text{len}(v_i) - w(t, v_i) = 4$). Since other neighbors of the intermediate user t (in red) do not influence this, we omit them.

Table 6.1: Relation between the IR and contact recommendation heuristics

Text IR	Contact recommendation
TFCs	EWCs
TDC	NDC
LNCs	CLNCs
TF-LNC	EW-CLNC

6.3 Theoretical analysis

A first step in the analysis of IR axioms in the contact recommendation task consists of identifying the set of algorithms for which the different axioms are applicable, and, then, for those, determining which constraints they satisfy (or under which conditions they satisfy them). In this section, we provide an overview of different contact recommendation methods, and their relation with the axioms.

To study this, we simplify the algorithmic taxonomy introduced in Chapter 3, and divide the approaches into two different groups: friends of friends methods, which only recommend people at network distance 2 from the target user, and methods which might recommend users at a greater distance. The first group includes the whole set of IR models introduced in Chapter 5, as well as the neighborhood-based methods introduced in Section 3.4.2: most common neighbors (Liben-Nowell and Kleinberg, 2007, Newman, 2001), Adamic-Adar (Adamic and Adar, 2003, Liben-Nowell and Kleinberg, 2007), preferential attachment (Zhou et al., 2009), the Jaccard similarity (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007) and the cosine similarity (Salton et al., 1975). The second group includes the rest of the approaches, including matrix factorization (Hu et al., 2008, Koren et al., 2009), random walk-based methods (Goel et al., 2015, White and Smyth, 2003) or kNN (Adomavicius and Tuzhilin, 2005, Ning et al., 2015).

The set of constraints we propose is only applicable to the first group of algorithms. The contact recommendation heuristics proposed here are based on the idea that the weighting functions depend on the common users between target and candidate users. This is similar to how the fundamental axioms for the search task consider that the functions use the terms shared between the query and the document. What is more: with that consideration, we can we define a tripartite graph with the three IR spaces (as the one in Figure 6.9) where the retrievable documents (those who share at least one term with the query) appear at distance 2 from the query.

For that reason, in this chapter we will focus on the algorithms in this friends of friends family. We envision, as future work, an extension of this set of axioms to cover the rest of contact recommendation approaches, for example, by adapting the concept of query expansion or pseudo-relevance feedback (Ruthven and Lalmas, 2003), as the formal analysis from Clinchant and Gaussier (2013) suggests (in our mapping, adding new terms to the query would be equivalent to allowing distances greater than 2).

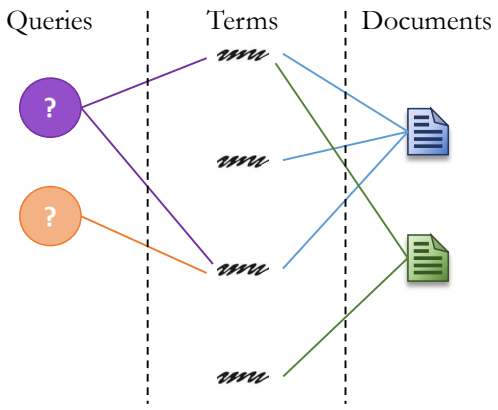


Figure 6.9: Tripartite graph of the IR task

6.3.1 IR models

We start by analyzing the IR models proposed in Chapter 5. In the adaptation of these models we describe the components of the ranking functions (frequency/weight, discriminative power functions, document/user length) keep their basic properties on which the formal analysis by Fang et al. (2004, 2011) relies, as shown in Figure 6.1. Consequently, the adapted methods satisfy the same constraints in the social network as they do in the text IR space. Also, if the axioms are only satisfied under certain conditions, we can find the new ones by adapting them to the contact recommendation task, using our framework. Then, models like PL2 (Amati, 2003, Amati and Van Rijsbergen, 2002), the pivoted length normalization vector space model (Singhal et al., 1998) or query likelihood approaches with Dirichlet (Zhai and Lafferty, 2004) or Jelinek-Mercer smoothing (Ponte and Croft, 1998) keep their original properties in this new space. We refer to the original paper (Fang et al., 2011) to see which properties they need to follow to satisfy the axioms.

The only point of difference caused by our mapping is related to the possibility of choosing different neighborhoods for defining the candidate users ($\Gamma^d(v)$) and their length ($\Gamma^l(v)$). It we choose the same neighborhood, nothing changes: the conditions for adhering to the axioms are the same. However, if we choose a different one, things change. As we only allow this difference for the BM25 model (and the extreme BM25 variant), we study how the different axioms change.

BM25

As the only difference between the original BM25 (Robertson and Zaragoza, 2009) and our version (Sanz-Cruzado and Castells, 2019a) is the definition of the candidate length, using the original proofs provided by Fang et al. (2011), it is straightforward to check that all the edge weight constraints and the neighborhood discrimination one are satisfied under the same conditions than they are for text IR: NDC is unconditionally true, whereas the EWCs depend on the following condition:

$$C_1 : |\Gamma^d_{inv}(t)| < |\mathcal{U}|/2 \tag{6.1}$$

This condition is very likely to be true in contact recommendation. For outgoing links, social network platforms limit the number of friendship relations than a single user can create. For incoming, as an example, as of 2020, Twitter has more than 300 million users, and the most followed user has just 108 million followers.

Differences arise when the constraints involving length normalization (both CLNCs and EW-CLNC) are studied. If we choose the same orientation for the user length and the neighborhood selection for the candidate user, the mapping maintains the properties. Therefore, the condition for satisfying it is the same than in the original model, C_1 (Fang et al., 2011). However, if the orientation for the length is changed, we can show that, for the CLNC1, BM25 satisfies the axiom if both conditions C_1 and C_2 are true, or both are false, where:

$$C_2 : \left(\Gamma^l := \Gamma^d\right) \vee \left(\text{len}^l(v_2) > \text{len}^l(v_1)\right) \tag{6.2}$$

Proof. Let t be a user in the network such that, for two candidate users, $w^d(t, v_2) > w^d(t, v_1)$, and, if u is the target user, $t \notin \Gamma^q(u)$. If, for any other user $x \neq t$, $w^d(x, v_1) = w^d(x, v_2)$, we need to prove that $f_u(v_1) > f_u(v_2)$.

A first observation to be made is that, as excepting the weights for t, the weights of the rest of people are the same for both candidate users, $\Gamma^q(u) \cap \Gamma^d(v_1) = \Gamma^q(u) \cap \Gamma^d(v_2)$. To simplify the proof, we assume that this intersection has only one neighbor: user x. Then, we can rewrite equation (5.8) as:

$$f_u(v_i) = \frac{C(x)}{k\left(1 - b + b\frac{\text{len}^l(v_i)}{\text{avg}_{v' \in \mathcal{U}} \text{len}^l(v')}\right) + w^d(x, v_i)} \text{RSJ}(x)$$

where $C(x) = (k+1)w^d(x, v_i)$ can be treated as constant for this proof, since $w^d(x, v_i)$ is the same for both candidate users. Starting from this formula, we can observe that

$$f_u(v_1) - f_u(v_2) > 0 \Leftrightarrow \text{RSJ}(x)(\text{len}^l(v_2) - \text{len}^l(v_1)) > 0$$

i.e. if the algorithm satisfies the condition on the right, then the algorithm satisfies the constraint. We study two different cases: when C_1 is satisfied and when it is not.

One the one hand, if the C_1 condition is true, then, $\text{RSJ}(x) > 0$. Consequently, the axiom will only be satisfied if $\text{len}^l(v_2) > \text{len}^l(v_1)$. From all the cases where this happens, we differentiate the case where $\Gamma^l(v) \coloneqq \Gamma^d(v)$, as, if this occurs, the axiom would be satisfied. The reason is that, for this particular case,

$$\text{len}^l(v_2) = \text{len}^d(v_2) = \text{len}^d(v_1) + w^d(t, v_2) - w^d(t, v_1) > \text{len}^d(v_1) = \text{len}^l(v_1)$$

by the initial properties of the proposed heuristic. With this, we can formulate condition C_2 as shown in equation (6.2). On the other hand, if condition C_1 is false, then, $\text{len}^l(v_2)$ must be smaller than $\text{len}^l(v_1)$ which only happens if condition C_2 is not met.

If we took more than one term in the query, then, we would need the sum of the RSJs of the common neighbors to be positive if C_2 is satisfied, or negative otherwise. Following Fang et al. (2011), we will still denote this as satisfying condition C_1.

□

In addition to the CLNC1 constraint, the possibility of changing the neighborhoods for the length and the description of the candidate user also affects the EW-CLNC axiom. In this case, we define a new condition we denote as C_3. BM25 will satisfy the constraint if conditions C_1 and C_3 are met, or none of them are. C_3 is defined as:

$$C_3 : \left(\Gamma^l \coloneqq \Gamma^d\right) \vee \left(\Gamma^l \coloneqq \Gamma_{und}\right) \vee \left(\frac{1-b}{b} \underset{v' \in \mathcal{U}}{\text{avg}} \text{len}^l(v') > w^d(t, v_2) - \text{len}^l(v_2)\right) \tag{6.3}$$

Proof. Given a target user u with a single neighbor t, $\Gamma^q(u) = \{t\}$, and two candidate users v_1 and v_2 such that $w^d(t, v_1) > w^d(t, v_2)$ and $\text{len}^l(v_1) = \text{len}^l(v_2) + \Delta w$ where $\Delta w = w^d(t, v_1) - w^d(t, v2)$, we have to check under which conditions $f_u(v_1) > f_u(v_2)$ for BM25. Using the initial properties, we can observe that

$$f_u(v_1) - f_u(v_2) > 0 \Leftrightarrow \text{RSJ}(t) \cdot \Delta w \left[(1-b)k + kb\frac{\text{len}^l(v_2) - w^d(t, v_2)}{\text{avg}_{x \in \mathcal{U}} \text{len}^l(x)}\right] > 0$$

so the axiom is kept by the BM25 approach if the right term is true. As $k, b, \Delta w > 0$ and for all $x \in \mathcal{U}$, $\text{len}^l(x) \geq 0$, we study again, two cases, depending on whether we have that $\text{RSJ}(t) > 0$ or not.

First, we study the case where the C_1 condition is met. Then, we observe the following:

$$\begin{aligned} f_u(v_1) - f_u(v_2) > 0 &\Leftrightarrow \Delta w \left[(1-b)k + kb\frac{\text{len}^l(v_2) - w^d(t, v_2)}{\text{avg}_{x \in \mathcal{U}} \text{len}^l(x)}\right] > 0 \\ &\Leftrightarrow (1-b) + b\frac{\text{len}^l(v_2) - w^d(t, v_2)}{\text{avg}_{x \in \mathcal{U}} \text{len}^l(x)} > 0 \\ &\Leftrightarrow \frac{1-b}{b} \text{avg}_{x \in \mathcal{U}} \text{len}^l(x) > w^d(t, v_2) - \text{len}^l(v_2) \end{aligned}$$

i.e. the algorithm fulfils the constraint if

$$\frac{1-b}{b}\text{avg}_{x\in\mathcal{U}}\text{len}^l(x) > w^d(t,v_2) - \text{len}^l(v_2)$$

Similarly to how we did for CLNC1, we express two particular cases of this result. As $b > 0$ and $\text{avg}_{x\in\mathcal{U}}\,\text{len}^l(x) > 0$, this expression is always true wen the orientation chosen for the neighborhood description is the same than the one for the length ($\Gamma^l(v) \coloneqq \Gamma^d(v)$). In this case, as, by definition, for all users t' and v' in the network $\text{len}^d(v') > w^d(t',v')$, then,

$$w^d(t,v_2) = w^l(t,v_2) < \text{len}^l(v_2)$$

so the above expression is true. The other particular case arises when we choose the undirected neighborhood of v for computing the user length ($\Gamma^l(v) \coloneqq \Gamma_{und}(v)$). In this other case, we have that:

$$\text{len}^d(v_2) = \text{len}_{und}(v_2) \geq w_{und}(t,v_2) \geq w^d(t,v_2)$$

Using the three conditions, we can define C_3 as in equation (6.3). Finally, if C_1 is false, and $\text{RSJ}(t) < 0$, we would need C_3 to be false if we want the algorithm to satisfy the axiom.

□

Differently from the other two, the remaining length normalization axiom, CLNC2, is not affected by the different orientation selections as the others. Indeed, the conditions for this axiom remain the same as in its original domain, i.e. the axiom is met by BM25 if condition C_1 is true (Fang et al., 2011).

Proof. Given a target user u with a single neighbor t, $\Gamma^q(u) = \{t\}$, given two candidate users v_1, v_2 such that $w^d(t,v_1) = n \cdot w^d(t,v_2)$ (with $n > 1$) then, we have to check whether $f_u(v_1) > f_u(v_2)$ and under which conditions. To simplify the proof, we assume that the only neighbor in $\Gamma^q(u)$ also belongs to $\Gamma^d(v_2)$ (otherwise, $f_u(v_1) = f_u(v_2) = 0$ and the axiom would be true without any other condition).

Then, we can prove that

$$f_u(v_1) - f_u(v_2) \geq 0 \Leftrightarrow \text{RSJ}(t)\left[(n-1)\cdot k\cdot(1-b) + kb\frac{n\cdot\text{len}^l(v_2) - \text{len}^l(v_1)}{\text{avg}_{x\in\mathcal{U}}\,\text{len}^l(x)}\right] \geq 0$$

Restricting to the right condition, and assuming that $\text{RSJ}(t) > 0$, as $n > 1, k > 0, b \in [0,1]$ and the average length is positive, we check whether $n \cdot \text{len}^l(v_2) \geq \text{len}^l(v_1)$. We distinguish four cases:

- Case 1: $\Gamma^l(v) \coloneqq \Gamma^d(v)$

  In this first case, $w^d(x,v_1) = n \cdot w^d(x,v_2)$ for all the users in the network, so

  $$\text{len}^l(v_1) = \text{len}^d(x,v_1) = n\cdot\text{len}^d(x,v_2) = n\cdot\text{len}^l(v_2)$$

  Therefore, the axiom is true for this algorithm.

- Case 2: $\Gamma^l(v) \coloneqq \Gamma_{und}(v) \wedge \Gamma^d(v) \coloneqq \Gamma^l(v)$

  In this second case,

  $$w^l(x,v_1) = w_{und}(x,v_1) = w_{und}(x,v_2) + (n-1)w^d(x,v_2) \leq n\cdot w_{und}(x,v_2)$$

  Then, $\text{len}^l(x,v_1) = \text{len}_{und}(v_1) \leq n\cdot\text{len}_{und}(v_2) = n\cdot\text{len}^l(v_2)$

- Case 3: $\left(\Gamma^l(v) \coloneqq \Gamma_{in}(v) \wedge \Gamma^d(v) \coloneqq \Gamma_{out}(v)\right) \vee \left(\Gamma^l(v) \coloneqq \Gamma_{out}(v) \wedge \Gamma^d(v) \coloneqq \Gamma_{in}(v)\right)$

  In this third case, it is easy to see that the weights $w^l(x,v_1) = w^l(x,v_2)$, and, then, $\text{len}^l(v_1) = \text{len}^l(v_2) < n\cdot\text{len}^l(v_2)$.

- Case 4: $\Gamma^l(v) \coloneqq \Gamma_{und}(v) \wedge \Gamma^d(v) \coloneqq \Gamma_{und}(v)$:

  For this case, as $w_{und}(t,v_1) = n\cdot w_{in}(t,v_2) + n\cdot w_{out}(t,v_2)$, then, $w^l(t,v_1) = n\cdot w^l(t,v_2)$ and $\text{len}^l(v_1) = n\cdot\text{len}^l(v_2)$.

Table 6.2: Constraint analysis results for BM25. By the equivalence notation e.g. $C_1 \equiv C_2$ we mean that conditions C_1 and C_2 can only be either both true or both false.

	TFC/EWC				LNC/CULNC		
	1	2	3	TDC/NDC	1	2	TF-LNC/EW-CULNC
Text IR	C_1	C_1	C_1	Yes	C_1	C_1	C_1
Contact rec.	C_1	C_1	C_1	Yes	$C_1 \equiv C_2$	C_1	$C_1 \equiv C_3$

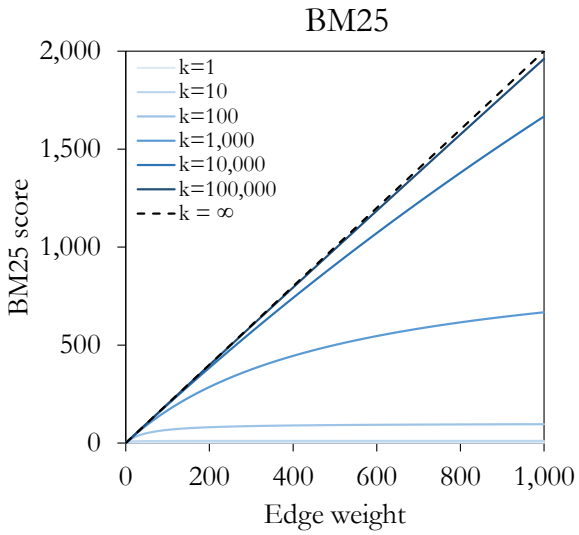


Figure 6.10: Example growth of the BM25 weight as a function of the edge weight for a single common neighbor between the target and the candidate users. Darker plots represent bigger values of the k parameter. For all the points, we fix the values of RSJ and user length.

If RSJ < 0, we would need $n \cdot \text{len}^l(v_2) < \text{len}^l(v_1)$ which, under the circumstances defined by the axiom, is impossible as we have proved. Then, the algorithm adheres to the axiom if condition C_1 is satisfied.

□

We summarize in Table 6.2 the conditions under which the BM25 adheres to the proposed constraints for both the contact recommendation and the text IR task.

Extreme BM25

In addition to BM25 (Robertson and Zaragoza, 2009), we also provide a study of the extreme BM25 (EBM25) method introduced in Section 5.3.1. By definition, this algorithm is a variant of BM25 where we make the k parameter tend to infinity. When we compare it with BM25, we notice that all the axioms are met under the same conditions than the original algorithm, except for EWC2 and EWC3.

In the BM25 model, the k parameter is the element that establishes how $f_u(v)$ grows as a function of the weight of a user in the intersection of the neighborhoods of the target and candidate users. The greater the value of k, the more the growth function approximates a linear function, as we can see in Figure 6.10. When k tends to infinity, the growth becomes linear, causing extreme BM25 ranking function not to be either strictly concave or strictly subadditive. We show next a formal proof of this.

Proof. We have to prove that extreme BM25 does not satisfy the second and third edge weight constraints. First, we do it for EWC2. If we have a target user u with a single neighbor t, $\Gamma^q(u) = \{t\}$ and three candidate users v_1, v_2, v_3 such that $\text{len}^l(v_i) = L$ for $i = 1, 2, 3$, and $w^d(t, v_3) = w^d(t, v_2) + 1$ and $w^d(t, v_2) = w^d(t, v_1) + 1$, we have to check that $f_u(v_2) - f_u(v_1) \leq f_u(v_3) - f_u(v_2)$. As L is a constant, we can rewrite $f_u(v_i)$ as:

$$f_u(v_i) = \frac{w^d(t, v_i) \cdot \text{RSJ}(t)}{C}$$

Table 6.3: Constraint satisfaction for different contact recommendation algorithms.

Algorithm	EWC1	EWC2	EWC3	NDC	CLNC1	CLNC2	EW-CLNC
BM25	Cond.	Cond.	Cond.	Yes	Cond.	Cond.	Cond.
EBM25	Cond.	No	No	Yes	Cond.	Cond.	Cond.
Pivoted VSM	Yes	Yes	Yes	Yes	Yes	Cond.	Cond.
PL2	Cond.	Cond.	Cond.	Cond.	Cond.	Cond.	Cond.
QLD	Yes	Yes	Yes	Yes	Yes	Cond.	Yes
QLJM	Yes	Yes	Yes	Yes	Yes	Yes	Yes
MCN	No	No	Yes	No	No	Yes	No
Adamic-Adar	No	No	Yes	No	No	Yes	No

where $C = 1 - b + bL/\operatorname{avg}_{x \in \mathcal{U}} \operatorname{len}^l(x)$ is a constant for the three candidate users. Then, it is enough to see that:

$$f_u(v_{i+1}) - f_u(v_i) = \frac{\text{RSJ(t)}\left(w^d(t, v_{i+1}) - w^d(t, v_i)\right)}{C} = \frac{\text{RSJ}(t)}{C}$$

This shows that $f_u(v_3) - f_u(v_2) = f_u(v_2) - f_u(v_1)$, and, consequently, the algorithm does not meet the constraint.

Now, we study EWC3. Given a target user u with two neighbors, t_1, t_2, such that $\text{RSJ}(t_1) = \text{RSJ}(t_2)$ and two candidate users v_1, v_2 such that $\operatorname{len}^l(v_1) = \operatorname{len}^l(v_2)$. If $w^d(t_2, v_2) = w^d(t_1, v_1) + w^d(t_2, v_1)$, with $t_1 \notin \Gamma^d(v_2)$ and $\{t_1, t_2\} \subset \Gamma^d(v_1)$, we want to see that $f_u(v_1) \leq f_u(v_2)$. As both RSJ values and the both user lengths are the same, it is straightforward that:

$$f_u(v_2) - f_u(v_1) = \frac{\text{RSJ}(t_1)}{C}\left[w^d(t_2, v_2) - \Big(w^d(t_1, v_1) + w^d(t_2, v_1)\Big)\right]$$

where C is the same constant as the one defined for EWC2 By the initial conditions set by the axiom, the result of this operation is 0, and, therefore, $f_u(v_1) = f_u(v_2)$, so the extreme BM25 algorithm does not adhere to this axiom.

□

Other algorithms

Beyond the IR model, other approaches such as Adamic-Adar (Adamic and Adar, 2003) or MCN (Newman, 2001) do operate at distance 2. In the particular case of these methods, they consider neither weights nor any means of normalization; only EWC3 and CLNC2 are applicable here. Both methods adhere to EWC3, since, under the conditions established by the heuristic, both algorithms are proportional to the number of common neighbors between the candidate and the target user. They also meet CLNC2, since, if we multiply all the weights of the links by a positive value, the scores for both functions remain the same (consequently satisfying the axiom).

We summarize our previous analysis in Table 6.3, where we identify, for all the methods analyzed in this section, whether they satisfy (fully or conditionally) or not the different axioms. Then, in the next section, we empirically analyze whether adhering to the axioms leads to an improvement of the performance of the algorithms.

6.4 Empirical analysis

As a mechanism to validate whether the proposed axioms are useful to predict, explain or diagnose why an IR system is working well or not, prior work on axiomatic thinking Fang et al. (2004, 2011) has studied to which extent satisfying a set of axioms is correlated with the accuracy of the systems. In our work, we take a similar perspective to undertake an empirical analysis of the fundamental IR axioms for the contact recommendation setting, focusing on friends of friends algorithms.

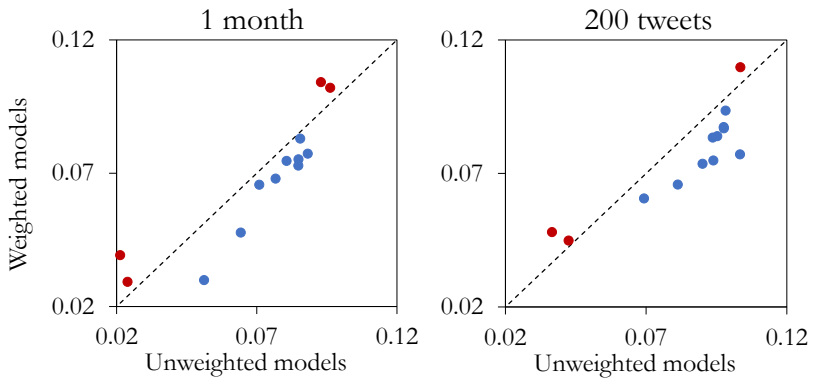


Figure 6.11: For the Twitter interaction datasets, nDCG@10 comparison between the weighted (y axis) and unweighted (x axis) versions of different contact recommendation algorithms. In both graphs, red dots represent those elements such that the value of nDCG@10 is greater for the y axis than for the x axis.

6.4.1 Experimental Setup

Data: For the experiments we describe in this section, we use the five networks described in Chapter 4: the follow and interaction networks for the Twitter 1-month and Twitter 200-tweets datasets and the Facebook friendship network released in the Stanford Large Network Dataset collection (McAuley and Leskovec, 2012). We apply the same split described in Chapter 4 for evaluation purposes.

Algorithms: In this chapter, we focus on friends of friends contact recommendation approaches, which suggest people at distance 2 from the target user. As representative IR models, we include the whole set of algorithms introduced in Section 5.3: BIR, BM25 and extreme BM25 (Robertson and Zaragoza, 2009) as probabilistic models based on the probability ranking principle (Robertson, 1977); query likelihood (Ponte and Croft, 1998) with Jelinek-Mercer (Jelinek and Mercer, 1980), Dirichlet (MacKay and Peto, 1995) and Laplace (Valcarce et al., 2017) smoothing as language models; PL2 (Amati, 2003, Amati and Van Rijsbergen, 2002), DFRee, DFReeKLIM (Amati et al., 2011), DPH (Amati, 2006) and DLH (Amati et al., 2007) as divergence from randomness models; and, as representatives of the vector space model, the simple cosine similarity VSM (Salton et al., 1975) and the pivoted length normalization VSM (Singhal et al., 1998).

In addition, we include the adaptations of neighborhood-based link prediction approaches: Adamic-Adar (Adamic and Adar, 2003), Jaccard similarity (Jaccard, 1901), most common neighbors (Liben-Nowell and Kleinberg, 2007) and cosine similarity (Salton et al., 1975).

We use the same hyperparameters we selected for the experiments in Chapter 5. We include the full parameter grid for the different algorithms and the optimal parameters in Tables C.3 and C.4 in Appendix C.

6.4.2 Experiments & Results

Edge Weight Constraints (EWCs)

We start by analyzing the edge weight constraints. Since we only have weights for the Twitter interaction graphs (edge weights for Facebook and the Twitter follow networks are binary), we focus on them, using interaction frequency as a natural basis for edge weighting.

The first natural question that arises when we study this group of axioms is whether the weights are useful or not for providing good recommendations. As this is equivalent to test the importance of the first of these axioms for the contact recommendation task, we compare the two options (binarized vs. not binarized weights) in all algorithms which make use of weights: cosine similarity between users and all the IR models except BIR. Taking the optimal configurations for each algorithm, we compare them in terms of nDCG@10. We show the results in Figure 6.11, where each dot represents a different approach. In the x axis, we represent the nDCG@10 for the unweighted variants, whereas the y axis shows nDCG@10 for the weighted ones. We can see that using weights results in an inferior performance in all algorithms, except for BM25, the simple VSM and cosine similarity without tf-idf. From them, only BM25 achieves a good performance. These observations suggest that EWC1

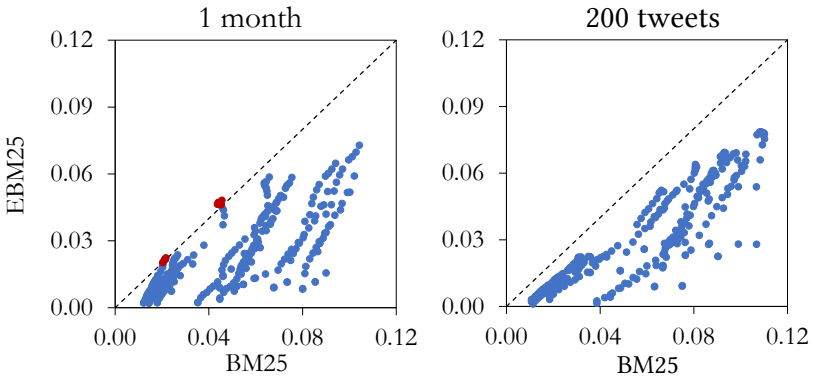


Figure 6.12: For the Twitter interaction datasets, nDCG@10 comparison between weighted variants of BM25 (x axis) and EBM25 (y axis). In both graphs, red dots represent those elements such that the value of nDCG@10 is greater for the y axis than for the x axis.

Table 6.4: Average AUC values for the most common neighbors algorithm for the different datasets, using $\Gamma^q := \Gamma_{und}$ and $\Gamma^d := \Gamma_{in}$ in the directed networks.

Twitter 1-month		Twitter 200-tweets		Facebook
Interactions	Follows	Interactions	Follows	
0.7545	0.8327	0.7064	0.7951	0.9218

does not appear to be a reliable heuristic for contact recommendation in networks (or, at least, not with this edge weight definition).

However, once we see that the weight is important for a method (and, therefore, EWC1 is important), does satisfying the rest of edge weight constraints provide more accurate recommendations? To check that, we use a similar approach to how Fang et al. (2004, 2011) verified the value of the axioms: in their work, starting from well-known IR models, such as BM25, which do not satisfy a constraint or satisfy it conditionally, they modify them to meet such constraints. Then, if the accuracy of the system improves, the axiom is deemed relevant for the corresponding task. In our experiments, we do the opposite: starting from a model that satisfies an axiom at least, conditionally, we modify it so the constraint is never met, and see if this results in a decrease of its effectiveness. Therefore, we compare an algorithm that satisfies all three constraints with another one that does not satisfy EWC2 and EWC3: we compare BM25 vs. extreme BM25. Fixing the best value for the k parameter for the BM25 model, we compare different configurations for BM25 and extreme BM25.

Results for this experiment are shown in Figure 6.12, where every dot in the plot corresponds to a different model configuration, the x axis represents nDCG@10 values for BM25 variants, and the y axis those of the EBM25 model. It can be observed that EBM25 does not improve over BM25 for almost every configuration (most dots are below the $x = y$ plane), thus showing that, as long as the weights are important for the model (and EWC1 is important), both EWC2 and EWC3 are relevant.

As explained in Section 6.2.1, the EWC3 axiom can be satisfied independently of EWC1 and EWC2, so, to finish our analysis of the edge weight constraints, we check its importance. We do it by addressing the following question: for any friends of friends algorithm, such as Adamic-Adar (Adamic and Adar, 2003), most common neighbors (Newman, 2001) or the IR models, is it beneficial to reward the number of common users between the target and the candidate users? To analyze this, we compare the MCN algorithm (satisfying the constraint) with a binarized version of MCN which returns all people at distance 2 regardless of the common neighbor count (i.e. we give all neighbors at distance two the same recommendation score). Restricting the test set to people at distance 2, Table 6.4 shows the AUC (Fawcett, 2006) of the MCN algorithm, averaged over users on each network. Under these conditions, the binarized version would have an AUC value of 0.5. Hence, since we attain AUC values over 0.7, we show that the number of common neighbors seems to be a strong signal for achieving good accuracy with friends of friends approaches (and, consequently, EWC3 seems to be important on its own as a contact recommendation heuristic).

Table 6.5: IR models without neighbor discrimination

Algorithm	$f_u(v)$ (without term discrimination)
BM25	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{(k+1) w^d(t,v)}{k\left(1 - b + b \cdot \text{len}^l(v) / \text{avg}_{x \in \mathcal{U}} \text{len}^l(x)\right) + w^d(t,v)}$
EBM25	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{w^d(t,v)}{1 - b + b \cdot \text{len}^l(v) / \text{avg}_{x \in \mathcal{U}} \text{len}^l(x)}$
QLD	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \left[w^q(t,u) \log\left(\frac{w^d(t,v)}{\mu}\right)\right] - \text{len}^q(u) \log\left(\frac{\text{len}^d(v)}{\mu}\right)$
QLJM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \log\left(\frac{\lambda}{1-\lambda} \cdot \frac{w^d(t,v)}{\text{len}^d(v)}\right)$
Pivoted VSM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{w^q(t,u)(1 + \log w^d(t,v))}{1 - s + s \cdot \text{len}^d(v) / \text{avg}_{x \in \mathcal{U}} \text{len}^d(x)}$

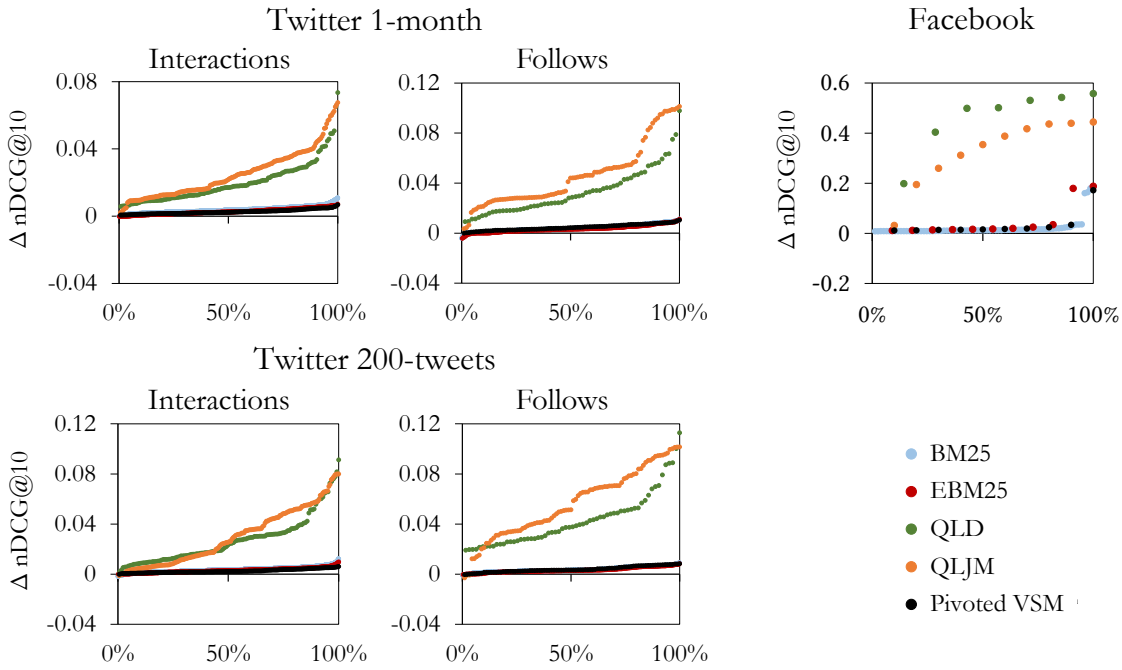


Figure 6.13: Difference in nDCG with and without term discrimination for different configurations of IR-based algorithms, sorted by difference value. Each dot represents a different configuration of the corresponding algorithm. A positive value indicates that the variant with term discrimination is more effective.

Neighbor Discrimination Constraint (NDC)

As explained in Section 6.2.2, satisfying this constraint limits the importance of highly popular common neighbors. This is typically addressed in IR by including a term discrimination element (such as the Robertson - Spärck-Jones formula in BM25/EBM25 or the $p_c(t)$ term in query likelihood approaches). Therefore, to check whether satisfying this axiom benefits the effectiveness of the contact recommendation methods, we compare – in terms of nDCG@10 – five IR models (BM25, EBM25, QLD, QLJM and pivoted length normalization VSM) with variants of them that lack term discrimination. We show the formulations for these variants in Table 6.5.

Figure 6.13 illustrates the difference between the original versions of each algorithm and their variants without neighbor discrimination. Each point in the plot represents the difference in the nDCG@10 value for a given configuration of the an algorithm. We sort the configurations by difference, so the x axis represents the percentage of the studied configurations of the algorithm with a smaller (or more negative) difference. We observe that in an overwhelming majority of points, the original versions beats the variant without neighbor discrimination,

Table 6.6: IR models without length normalization

Algorithm	$f_u(v)$ (without length normalization)
BM25	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \frac{(k+1)w^d(t,v)}{k + w^d(t,v)} \text{RSJ}(t)$
EBM25	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^d(t,v) \text{RSJ}(t)$
QLD	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} \left[w^q(t,u) \log \left(\frac{w^d(t,v)}{\mu} \cdot \frac{\sum_{x \in \mathcal{U}} \text{len}^d(x)}{\text{len}^d_{inv}(t)} \right) \right]$
QLJM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \log \left(\frac{\lambda}{1-\lambda} \cdot w^d(t,v) \cdot \frac{\sum_{x \in \mathcal{U}} \text{len}^d(x)}{\text{len}^d_{inv}(t)} \right)$
Pivoted VSM	$\sum_{t \in \Gamma^q(u) \cap \Gamma^d(v)} w^q(t,u) \cdot (1 + \log w^d(t,v)) \cdot \log \frac{\lvert\mathcal{U}\rvert + 1}{\lvert\Gamma^d_{inv}(t)\rvert}$

showing that NDC appears to be a key heuristic for providing good contact recommendations. This confirms the hypothesis in many recommendation approaches such as Adamic-Adar (Adamic and Adar, 2003) or resource allocation (Zhou et al., 2009) that favoring common users with low degree seems to be a good idea when recommending people in social networks.

Length Normalization Constraints (CLNCs & EW-CLNC)

Last, we study the effect of normalizing by the length of the candidate user. Similarly to the previous section, we remove the length normalization element from the BM25, EBM25, QLD, QLJM and the pivoted length normalization VSM, and compare these new versions with the original adaptations of the IR models. We show the equations for these new approaches in Table 6.6. Again, we compare them in terms of nDCG@10. We plot the differences in accuracy for different configurations of each of these algorithms in Figure 6.14. In that figure, we observe an opposite trend to what we expected: instead of achieving a worst performance, the algorithms without normalization do improve the results. Therefore, the different length normalization constraints do not seem to be useful for contact recommendation.

These observations are consistent with the preferential attachment phenomenon in social networks (Barabási and Albert, 1999). This phenomenon shows that, in social networks, high-degree users are more likely to receive new links than long-tail degree users. We check this in Figure 6.15. In that figure, we compare the performance of the recommendation approaches listed in Section 6.4.1 with the average in-degree, out-degree and (undirected) degree of the recommended people. We only use unweighted versions of the algorithms since the behaviour was similar for the weighted versions. We observe that, for all the datasets, both the in-degree and the degree are clearly correlated with the nDCG@10 values, as suggested by the preferential attachment phenomenon. The behaviour is not so clear in the case of the out-degree: for this measure, the correlation is nearer to 0, and even negative in the case of the 200-tweets interaction dataset. This explains the few cases where the original algorithm is slightly better than the variant without normalization: all of them normalize by the sum of the weights of the outgoing links of the candidate users.

6.5 Conclusions

In this chapter, we have deepened on the relation between contact recommendation in social networks by a theoretical and empirical analysis of the fundamental IR axioms (Fang et al., 2011). First, we have translated the axioms to the contact recommendation task, and analyzed whether the mapping we introduced in Chapter 5 is sound and complete. Then, we have conducted experiments with friends of friends models to identify which of the constraints are interesting and relevant for the contact recommendation task, stating that an IR axiom is relevant if satisfying it leads to an improvement on the accuracy of the algorithms. The main conclusions arisen from this study are summarized below:

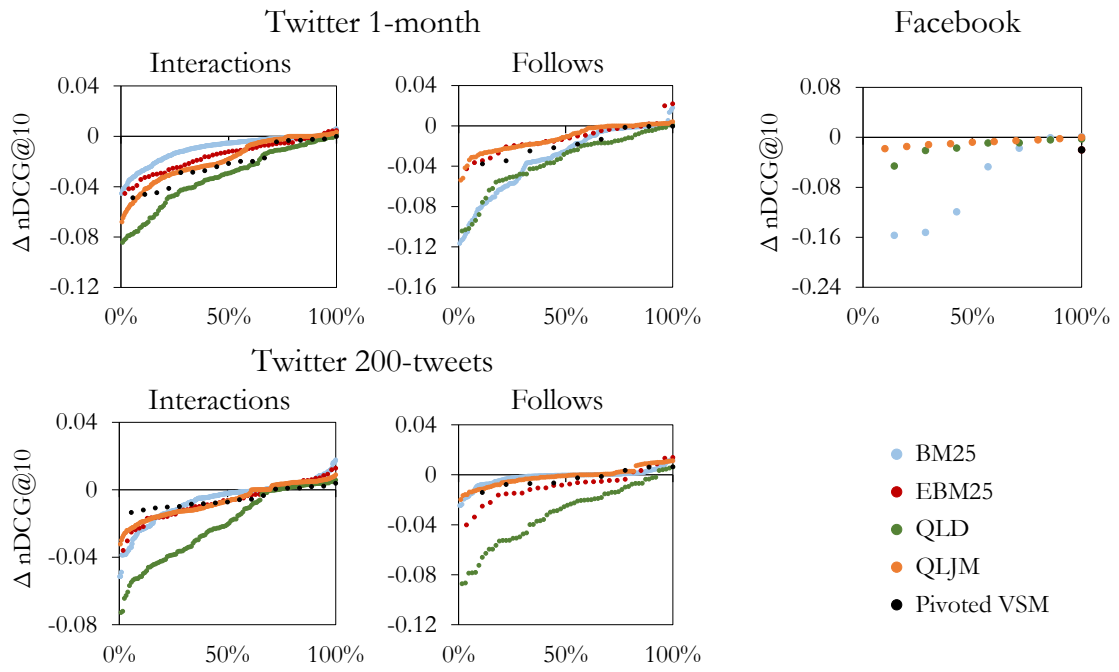


Figure 6.14: Difference in nDCG with and without length normalization for different configurations of IR-based algorithms, sorted by difference value. Each dot represents a different configuration of the corresponding algorithm. A positive value indicates that the variant with length normalization is more effective.

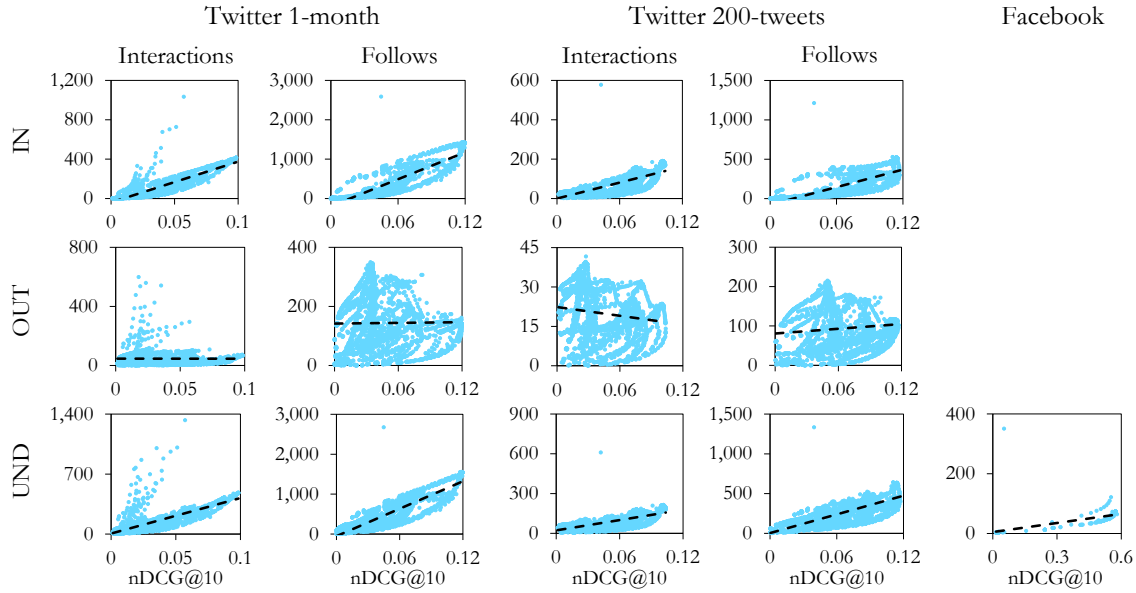


Figure 6.15: Relation between accuracy and degree for different configurations of contact recommendation algorithms.

- The fundamental IR axioms, as they are, can only be used to analyze friends of friends methods, since they rely on the query-term-document relation. Therefore, new heuristics should be developed for algorithms which are able to recommend people at distance greater than 2, such as random walks or matrix factorization.
- The mapping we define in Chapter 5 generally preserves the properties of the original IR models when applied. The only exception we have found occurs when we apply the mapping in directed networks, and the orientations selected for computing the candidate user length differs from the usual (i.e. we choose the

neighbors of the candidate user using one neighbor orientation, and we compute the length using another). In our case, this only happens with the BM25 and EBM25 approaches.

- In general, using edge weights does not improve the accuracy of contact recommendation algorithms. However, when the methods are able to gain advantage from using these weights, satisfying the term frequency constraints leads to a performance improvement.
- Favoring candidate users sharing a greater number of common neighbors with the target user has a positive effect on the accuracy of the friends of friends algorithms. Similarly, weighting the common users to promote the effect of nodes with smaller degree in the recommendation, as proposed by algorithms such as Adamic-Adar (Adamic and Adar, 2003) or resource allocation (Zhou et al., 2009), improves the effectiveness of the approaches.
- Length normalization axioms interfere with one of the main evolutionary principles of social networks, the preferential attachment phenomenon (Barabási and Albert, 1999), causing that satisfying the length normalization constraints has an opposite effect to the expected one, decreasing the accuracy of the approaches.

7
Advanced adaptations of IR models

Highlights

- *Due to the particularities of the task, any contact recommendation algorithm can be used to select neighbors in a kNN scheme.*
- *IR models are shown to be more effective as neighbor selection methods in kNN than as standalone recommenders.*
- *We achieve further effectiveness enhancements by applying learning to rank techniques upon IR models.*

In previous chapters, we study the adaptation of information retrieval models like BM25 (Robertson and Zaragoza, 2009) or PL2 (Amati, 2003, Amati and Van Rijsbergen, 2002) for the contact recommendation task. In the experiments we run in Chapter 5 we observe that they regularly achieve high accuracy and they are very efficient algorithms, and in Chapter 6 we explore the reasons behind that effectiveness. However, we also notice that other baselines such as matrix factorization (Hu et al., 2008) in directed networks or personalized PageRank (White and Smyth, 2003) in undirected networks are hard to beat, often reaching the top accuracy values in our comparisons.

In this chapter, we seek to close the accuracy gap between IR models and those top-performing approaches, by exploring alternative uses of the search-oriented algorithms. In particular, we study two possible applications with potential improvements: first, we take them as possible similarities for user-based and item-based nearest-neighbor (kNN) collaborative filtering, following previous work in general recommendation (Valcarce, 2015, Valcarce et al., 2017); second, we explore the adaptation of supervised learning to rank techniques (Liu, 2007) to this contact recommendation task, where we take the outcome of the direct IR-based recommendation and the new kNN variants as features.

The work we present in the following pages has been partialy published in:

- **Sanz-Cruzado et al. (2020a):** Javier Sanz-Cruzado, Pablo Castells, Craig Macdonald and Iadh Ounis. Effective Contact Recommendation in Social Networks by Adaptation of Information Retrieval Models. *Information Processing and Management*, 57(5),Article 102285, September 2020.

7.1 Research questions

The contents of this chapter address the following research questions:

- **RQ1:** Can we use IR models as similarities in nearest-neighbor kNN schemes for contact recommendation?
- **RQ2:** Does the use of IR models as similarities in nearest-neighbors improve the performance of such models as straightforward recommenders? And does it improve the accuracy of the top performing baselines?
- **RQ3:** How can we adapt learning to rank techniques from text search for the contact recommendation task?
- **RQ4:** Do learning to rank techniques achieve better accuracy results than unsupervised models?
- **RQ5:** Which are some important features for learning to rank techniques?

7.2 Neighbor selection in collaborative filtering

The first step we take towards building more effective contact recommendation approaches based on information retrieval models is their use as similarities in nearest-neighbor collaborative filtering methods, following the

idea by Valcarce et al. (2017) for general item recommendation. In that work, the authors compare several similarities for kNN, including some of them based on query likelihood models (Ponte and Croft, 1998, Zhai and Lafferty, 2004). Our work expands and generalizes this idea for the particular case of recommending contacts.

This is not the first time we use these kind of approaches for recommending contacts in social networks: we first define these algorithms in Chapter 3, and then, we use them in the experiments in Sections 4.3 and 5.4 as baselines. However, in all these sections, they have one thing in common: they only use the cosine similarity. Here, we expand them, so they can use other similarities, based on friends of friends methods. For that, we first refresh how the algorithm works and explain why these approaches can be applied as similarities. Then, we provide some insights about why these variants might be good for suggesting people and, finally, we show some experimental results.

7.2.1 Algorithm definition

Nearest-neighbor algorithms (also known as memory-based algorithms) are a family of recommendation algorithms which rely on the same principle: that similar users prefer similar items and similar items are preferred by similar users (Ning et al., 2015). In a general item recommendation setting, the idea behind this algorithm is to find the set of k users (or items) in the system which are the most similar ones to the target user (or the candidate item). This set of users (or items) is known as the neighborhood of the user (item), and it is used to generate the recommendation scores by applying a linear combination of the ratings provided by those users to the candidate items (or to those items by the target user). We differentiate two families of algorithms:

When the neighborhood of the target user is exploited, we refer to these approaches as user-based kNN methods. Depending on factors as the normalization of the approach or bias correction, there are multiple algorithms in this family. In this thesis we focus on a simple and straightforward variant which has been proven effective in several recommendation domains (Cañamares and Castells, 2017) and it is described with the following ranking function:

$$f_u(i) = \sum_{w \in N_k(u)} \text{sim}(u, w) r_w(i) \tag{7.1}$$

where $N_k(u)$ represents the top k most similar users to u according to the similarity we represent as $\text{sim}(u, v)$ and $r_w(i)$ is the rating provided by user w to the item i.

When we choose the neighbor as the set of most similar items to the candidate one, we talk about item-based kNN. As they follow similar principles to the user-based approach, there are, again, multiple possibilities for item-based methods, but, the only one we explore in our experiments is the approach defined by the following equation:

$$f_u(i) = \sum_{j \in N_k(i)} \text{sim}(i, j) r_u(j) \tag{7.2}$$

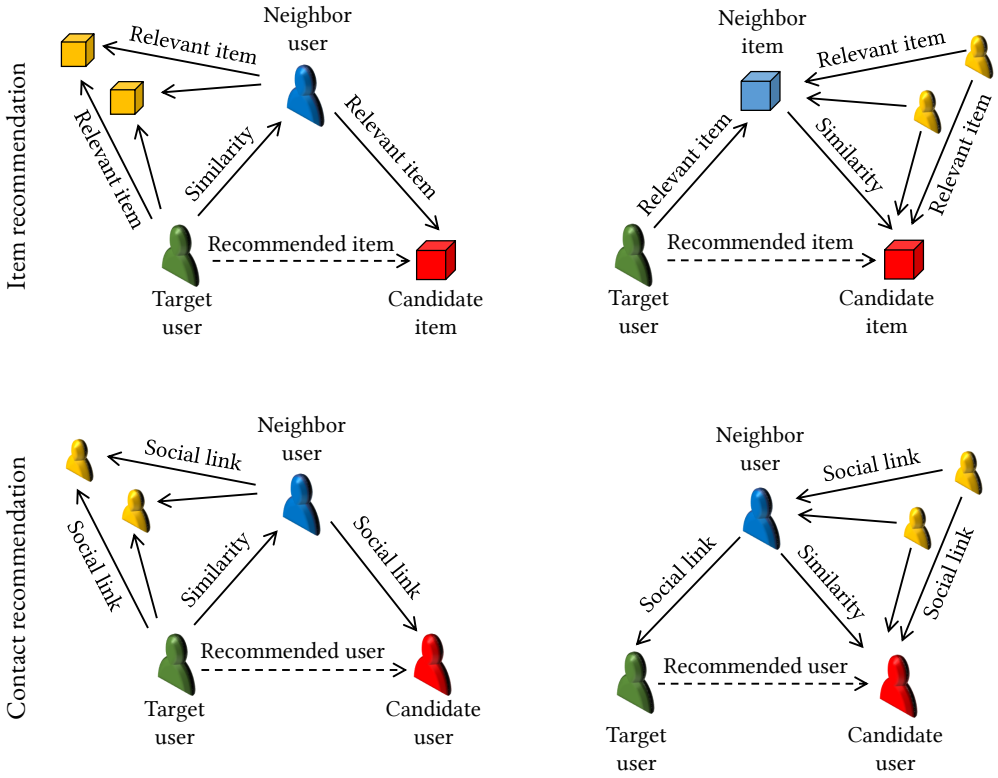
where $N_k(i)$ is the top k most similar items to the candidate item i, ranked using the values of the similarity we include in the equation as $\text{sim}(i, j)$ and $r_u(j)$ is the rating that the target user u has provided to the neighbor user j.

Although depending on how we define the similarities, we can use user-based and item-based methods as representatives of different categories of algorithms, such as content-based or demographic recommenders, the most common way to use these algorithms is in a collaborative filtering manner. This way, the similarity between users (or items) is computed as a function of the ratings of the users (items) in the system. For example, it is possible to define a collaborative filtering user-based approach by using the cosine similarity between two users as:

$$\text{sim}(u, v) = \cos(u, v) = \frac{\sum_{i \in \mathcal{I}_u \cap \mathcal{I}_v} r_u(i) \cdot r_v(i)}{\sqrt{\sum_{i \in \mathcal{I}_u} r_u^2(i)} \sqrt{\sum_{i \in \mathcal{I}_v} r_v^2(i)}} \tag{7.3}$$

where $\mathcal{I}_u$ represents the set of items rated by user u. These collaborative filtering approaches are illustrated in the first row of Figure 7.1.

Their adaptation for the contact recommendation task is straightforward: as users also play the role of items in this task, we just need to substitute the items in the general recommendation framework by the set of users



(a) User-based kNN (b) Item-based kNN

Figure 7.1: User-based and item-based kNN in classical item recommendation task vs. the same models in contact recommendation.

in the network. This is shown in the second row of the figure. We can modify the equations for these methods as follows to adapt them to the contact recommendation task:

$$f_u(v) = \sum_{t \in N_k(u)} \text{sim}(u,t) w(t,v) \tag{7.4}$$

for the user-based version, and

$$f_u(v) = \sum_{t \in N_k(v)} \text{sim}(v,t) w(u,t) \tag{7.5}$$

for item-based kNN.

Similarities in contact recommendation

Beyond choosing different neighborhood sizes, the use of different similarities represents a typical way to explore different variants of these approaches (Valcarce et al., 2017). Despite using cosine similarity – a common and effective option for kNN algorithms (Ning et al., 2015) – as our only configuration option in previous experiments, this is not (by far) the only similarity we can apply. If we study similarities from a more mathematical point of view, for the particular task of recommending contacts, we can define a similarity function as a real-valued function over pairs of users: $\text{sim} : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$.

Considering that definition, it is simple to notice that any ranking function for contact recommendation can be applied as a neighborhood selection function in a kNN scheme. For a single user in the network, we define the ranking function for a recommendation approach as $f_u : \mathcal{U} \to \mathbb{R}$. We can easily extend that function to consider the rest of users in the network, reformulating it as $f : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$, where $f(u,v) = f_u(v)$. It is thus possible to take $\text{sim}(u,v) = f(u,v) = f_u(v)$, so this observation proves that any contact recommendation algorithm can be applied as a similarity function for nearest neighbors, and we can do the opposite – use similarities as recommendation algorithms – by taking $f_u(v) = \text{sim}(u,v)$.

Table 7.1: Percentage of edges in the test set whose endpoint users are at a given (undirected) distance from each other in the input network.

	Twitter 1-month		Twitter 200-tweets		Facebook
Distance	Interactions	Follows	Interactions	Follows	
2	79.61%	96.45%	57.60%	91.90%	99.43%
3	19.62%	3.53%	37.73%	8.05%	0.55%
4	0.75%	0.02%	4.53%	0.06%	0.01%
5	0.02%	–	0.13%	–	–
∞	–	–	–	–	0.01%

Although we can use the same algorithms to perform both tasks, we should stress that contact recommendation and neighborhood selection are not the same problem. We highlight here some of the properties that make them distinct. First, if we overlook these tasks from an information retrieval point of view, their meaning is different: contact recommendation can be understood as the task of predicting the new terms that will appear in a document in the future (and were not there earlier), whereas neighborhood selection represents a query reformulation task (Huang and Efthimiadis, 2009): given a representation of a user by the connections he has in the network, we want to find another representation that better expresses what kind of people we are looking for. Another distinction between them lies on the set of candidate users for both tasks: while in contact recommendation only those users with whom the target user has no connection can be candidates (we would otherwise give the user a rather uninteresting and redundant recommendation), when we filter users to retrieve the optimal neighbors, the only person in the network who must not be selected is the target user for the task, as he would not provide any additional information to the nearest neighbors approach. These observations show that the optimal properties of contact recommendation might differ from the optimal ones for selecting neighbors.

In this thesis, we focus on the use of friends of friends methods (with a particular focus on information retrieval models) as similarities. Indeed, these algorithms are more similar to the classic neighborhood-based approaches, as it can be observed in Figure 7.1, but, for the particular case of contact recommendation, nothing prevents us from using other algorithms such as random walks or matrix factorization (Koren et al., 2009) as similarities for kNN methods.

We want to observe that, in the classical approaches for general recommendation, when we want to find the similarities between users, we use the ratings they provide to the items (represented by the outgoing edges in contact recommendation). In the item-based case, we use the ratings provided to the different items (in this case, the incoming edges). However, in this work we continue the line in previous chapters, where we find interesting to allow different orientations for the target/candidate user and their neighbors. Therefore, for this task, we can choose a neighborhood representation for the target/candidate users, and other one for their possible neighbors.

7.2.2 Motivation

Before testing the models, we find interesting to provide an intuition about why these models might improve the IR and friends of friends approaches, as well as the user-based and item-based methods studied in previous chapters. We have two main reasons to consider the potential of these algorithms.

The first reason responds to a limitation of all friends of friends algorithms: they are only capable of recommending links towards people at distance 2 from the target user. As we can observe in Table 7.1, this is the case of most of the links that appear in the test network (including more than 90% of them in several networks). However, there is a reasonably high amount of edges created at distance three from the target user, which cannot be reached by these approaches on their own. If we managed to obtain an approach that recommended links (at most) at such distance, we would be able to cover up more than 95% of the newly created links in all the networks.

This is the case of user-based and item-based kNN methods when we take friends of friends algorithms as similarities. We can observe Figure 7.1 to understand why: let's take user-based kNN. If we take the target user as origin and such a similarity, all the common users between him and a possible selected neighbors are

at undirected distance 1. Because of that, the selected neighbors of the target user might be at distances 1 or 2 from the target user. As the candidate users are at one additional step from the neighbors, we could recommend either a user at distance 2 or a user at distance 3, thus showing that what we claim is true.

The second reason why these approaches might work is found in the experiments reported in section 5.4. In those experiments, we compare the user-based and item-based methods using cosine similarity against cosine similarity as a standalone algorithm. As we can observe in Table 5.6, there is a marked improvement when we use such approach as a similarity in kNN schemes. It is hence natural to wonder whether IR models might also lead to a better performance when we integrate them in kNN schemes in a similar manner. This idea seems even more promising considering that IR models typically perform much better than the cosine similarity when applied as contact recommenders.

7.2.3 Experiments

Experimental setup

In order to evaluate these new approaches, we consider the same four Twitter networks and the Facebook dataset we use in our previous experiments, and we apply the same experimental setup we first describe in Section 4.3: we first tune our parameters using the validation network, and then, taking the training graph as an input, we generate recommendations and evaluate them on the test set, using nDCG and MAP as evaluation metrics.

As algorithms, as the main contribution in this chapter, we take several variants of user-based and item-based methods, using different similarities. In particular, we use two groups of similarities. Representing the IR models, we take BIR, BM25, extreme BM25 (Robertson and Zaragoza, 2009), query likelihood (Ponte and Croft, 1998) with Jelinek-Mercer, Dirichlet and Laplace smoothings, PL2 (Amati and Van Rijsbergen, 2002), DFRee, DFReeKLIM (Amati et al., 2011), DPH (Amati et al., 2007), DLH (Amati, 2006) and the simple vector space model (Baeza-Yates and Ribeiro-Neto, 2011). Then, representing other friends of friends approaches, we also take most common neighbors (Liben-Nowell and Kleinberg, 2007), Jaccard (Jaccard, 1901), cosine similarity (Lü and Zhou, 2011) and Adamic-Adar as similarities. In Appendix C, we report the optimal parameter selection for both families of kNN algorithms in Tables C.5 to C.7.

In addition to user-based and item-based, for each similarity, we also take the similarity function as a standalone recommender, and we compare these approaches against the most effective algorithms in the comparison in Section 5.4: iMF (Hu et al., 2008) for the Twitter networks and personalized PageRank for the Facebook one (White and Smyth, 2003). Again, we take popularity-based and random recommendation as sanity-check baselines.

Experimental results

Figure 7.2 gives a first observation of the potential of the proposed kNN approaches. It provides a comparison between the optimized direct friends of friends approaches and both optimized kNN algorithms in terms of nDCG@10. Except for the Facebook network, we observe that user-based kNN approaches generally improve the performance of the friends of friends baselines on their own (green points are above the $y = x$ dotted line in the different plots, by considerable distances in many cases). In the Facebook network, differently, only a small selection of algorithms achieve that improvement: BM25, extreme BM25, Jaccard and cosine similarity.

When we analyze the item-based variant, however, results are mixed. In general, the worst approaches are improved, but, as we observe the most effective IR models, such as BM25, the advantage of using these methods becomes unclear: these algorithms achieve a noticeable advantage for the interaction network of the 1-month dataset. However, the standalone algorithms work much better in both 200-tweets networks.

Surprisingly, there is one kNN variant which does not provide effective recommendations for neither the user-based or the item-based algorithms on any of the studied datasets: the query likelihood model with Laplace smoothing does not seem to work well as a neighbor selector, achieving low nDCG values in all datasets.

Another observation of Figure 7.2 also reveals that, over these networks, user-based kNN approaches achieve a better performance than item-based kNN, consistently with the results found in previous experiments. We also see that, in the interaction networks, many points in the plots lie above the blue horizontal line, showing that many kNN combinations perform better than the matrix factorization baseline, which was hard

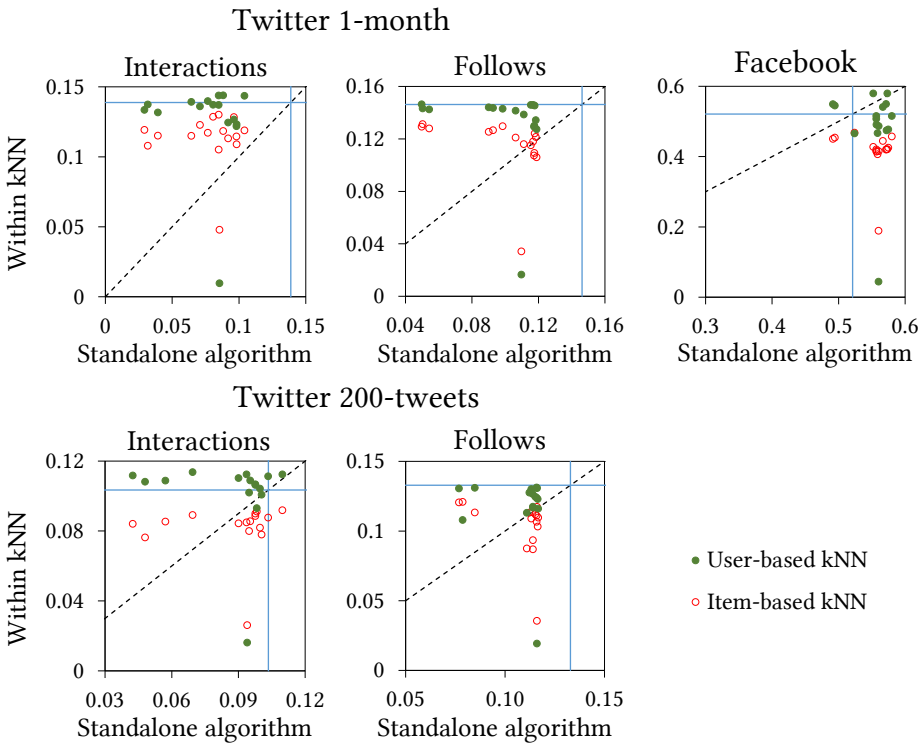


Figure 7.2: Comparison (in terms of nDCG@10) between friends of friends approaches as standalone models vs. when integrated in user-based (green) and item-based (red) kNN approaches. The x axis represents the nDCG@10 value for the standalone approach, whereas the y axis is the nDCG@10 value for the kNN versions. The dotted line shows the $y = x$ cut, and blue lines show the nDCG@10 value for iMF as the top-performing algorithm in the standalone experiments.

to beat by the standalone alternatives in our previous experiments (as shown by the blue vertical lines in the plots).

We detail the comparison with the top performing baselines in Table 7.2, where we compare the different user-based kNN methods integrating different contact recommendation algorithms as neighbor selection functions, against the most effective algorithms in our experiments in Section 5.4 in terms of nDCG@10 and MAP@10. As it can be observed in the table, user-based kNN with BM25 similarity stands out among the rest of approaches in all the directed networks: in both interaction graphs, this algorithm beats matrix factorization with a significant difference in terms of nDCG (and, for the 1-month interactions network, also for MAP@10). In the follows networks, the model is slightly behind iMF when we observe nDCG@10 and a slightly above it when we observe MAP@10. These differences are not significant for any of the metrics, thus indicating a technical tie between iMF and user-based kNN with BM25. On the Facebook network, user-based BM25 is again the best kNN approach, but it does not improve over personalized PageRank. It just reaches a technical tie when we use the Tukey HSD test to determine whether the differences are significant or not.

Beyond BM25, other models which have been shown to be effective as neighborhood selectors are extreme BM25 and query likelihood with Jelinek-Mercer smoothing. Surprisingly, some models that provided effective recommendations as standalone recommenders, as BIR and PL2, are here among the worst kNN approaches, reinforcing that way the intuition that models that provide good results for one task might not be so good for the other.

In conclusion, the use of IR models as similarities within nearest neighbors recommendation schemes seems to provide an accuracy improvement over their use as standalone recommenders, specially on directed networks like Twitter. On such directed networks, similarities like BM25 or query likelihood manage to beat (or at least, tie with) the results of the best, hard to beat baselines such as matrix factorization. This is interesting, since, as we can observe in Section 5.5, despite not being as fast as IR models, the training and update times for user-based nearest-neighbors approaches are much smaller than the ones needed for the iMF algorithm, thus being a good baseline for link recommendation.

Table 7.2: Effectiveness of the user-based kNN (abbreviated as "UB" in the table) + the IR model adaptations and other baselines. The cell color goes from red (lower) to blue (higher values) for each metric/dataset, with the top value both underlined and highlighted in bold. The best user-based approach (with BM25 similarity) differences with iMF are statistically significant for the Facebook and the interactions networks in terms of nDCG@10 and MAP@10 (except for the 200-tweets dataset) and are not significant in the case of the follows networks in any of the metrics. Full additional detail of statistical significance is given in Figures D.11 to D.15 in the supplementary material.

		Twitter 1-month				Twitter 200-tweets				Facebook	
		Interactions		Follows		Interactions		Follows			
	Algorithm	nDCG	MAP	nDCG	MAP	nDCG	MAP	nDCG	MAP	nDCG	MAP
UB IR	UB BM25	**0.1494**	0.0730	0.1455	**0.0603**	0.1125	0.0588	0.1313	**0.0467**	0.5802	0.3734
	UB extreme BM25	0.1493	**0.0730**	0.1460	0.0597	0.1114	0.0585	0.1302	0.0459	0.5799	0.3733
	UB QLJM	0.1482	0.0718	0.1460	0.0594	**0.1137**	0.0597	0.1309	0.0455	0.5414	0.3406
	UB QLD	0.1444	0.0701	0.1431	0.0577	0.1125	**0.0602**	0.1247	0.0448	0.5156	0.3194
	UB DLH	0.1439	0.0698	0.1442	0.0584	0.1103	0.0576	0.1277	0.0438	0.5159	0.3225
	UB DFReeKLIM	0.1407	0.0672	0.1437	0.0581	0.1090	0.0571	0.1267	0.0432	0.5075	0.3150
	UB VSM	0.1387	0.0675	0.1432	0.0581	0.1118	0.0597	0.1080	0.0354	0.4664	0.2811
	UB DFRee	0.1402	0.0668	0.1386	0.0552	0.1068	0.0552	0.1231	0.0420	0.4872	0.3018
	UB DPH	0.1410	0.0671	0.1416	0.0569	0.1064	0.0549	0.1239	0.0427	0.4914	0.3035
	UB BIR	0.1287	0.0615	0.1294	0.0507	0.1008	0.0514	0.1176	0.0402	0.4748	0.2881
	UB PL2	0.1260	0.0636	0.1344	0.0553	0.0931	0.0487	0.1162	0.0405	0.5498	0.3449
	UB QLL	0.0097	0.0037	0.0166	0.0054	0.0162	0.0074	0.0194	0.0057	0.0443	0.0247
UB FOAF	UB Cosine	0.1367	0.0669	**0.1464**	0.0599	0.1082	0.0569	0.1306	0.0456	0.5457	0.3491
	UB Jaccard	0.1392	0.0683	0.1425	0.0589	0.1089	0.0572	0.1311	0.0459	0.5494	0.3521
	UB Adamic-Adar	0.1264	0.0597	0.1301	0.0529	0.1043	0.0539	0.1168	0.0400	0.4775	0.2915
	UB MCN	0.1291	0.0611	0.1274	0.0511	0.1020	0.0530	0.1132	0.0382	0.4670	0.2832
Standalone	iMF	0.1388	0.0663	0.1462	0.0590	0.1035	0.0558	**0.1329**	0.0465	0.5210	0.3207
	BM25	0.1042	0.0440	0.1177	0.0479	0.1097	0.0583	0.1159	0.0405	0.5731	0.3686
	Pers. PageRank	0.0800	0.0315	0.0965	0.0362	0.0630	0.0317	0.0843	0.0276	**0.5891**	**0.3826**
	Popularity	0.0572	0.0291	0.0449	0.0161	0.0422	0.0212	0.0397	0.0098	0.0523	0.0234
	Random	0.0014	0.0005	0.0011	0.0002	0.0003	0.0001	0.0018	0.0003	0.0030	0.0009

7.3 Learning to rank

Until now, all the search-based methods we have studied have been unsupervised: they do not receive as input any information about the relevance of the documents for specific queries. However, in the last few years, it has been observed that top performance in text retrieval is commonly achieved by applying supervised machine learning approaches, and, in particular, learning to rank techniques (Liu, 2007). Therefore, we analyze in this section their utility for suggesting people to follow or befriend in social networks.

7.3.1 Adaptation to contact recommendation

As it implies the use of supervised approaches, the application of learning to rank techniques for the text search task differs from how the traditional unsupervised methods solve it. Therefore, we have to find a way to adapt these techniques for suggesting users in networks. For that, we first describe the general process for deploying learning to rank techniques in text IR and, then, we explore their adaptation to contact recommendation.

Learning to rank in text IR

When we devise a search system using learning to rank approaches, we can distinguish two steps: first, how the model is built (i.e. how we train the model), and then, how the trained model is applied in production.

In order to train the model, the first step is to obtain two sets of queries: a set of training queries and a set of validation queries. The validation queries must be different from the first ones. As learning to rank methods are supervised, along with these queries, we need to produce relevance judgments for each one of them. Then, once both sets of queries (and judgements) have been acquired, the following steps are typically followed:

1. For each training query, obtain a set of k candidate documents using a simple IR model.
2. For each retrieved document, generate a fixed set of features involving the document and the query.
3. Using those features and example relevance judgments for each query-document pair, train the model. Validation data is used to prevent overfitting.

The first step allows the system to reduce the number of candidate documents for the search, by applying an initial pruning. As, typically, only a fraction of the documents in a collection contains useful information for a given query, this is a reasonable step. To do this pruning, we have to ensure that we collect an important fraction of relevant documents for the query, as well as strong negative examples, which are difficult to differentiate from the relevant ones. We can do this by applying a simple, unsupervised IR model, since they are usually effective for the task on their own. A good model to select is one with a high recall value at cutoff k (where k here represents the number of candidate documents to retrieve), since that way, more relevant results will be retrieved in this step. For instance, Liu (2007) described the BM25 weighting model (Robertson and Zaragoza, 2009) as a commonly used model to retrieve documents in a learning to rank framework, with $k = 1,000$. Macdonald et al. (2013a) carried a study for determining a good k value, and determined that, for large corpora such as ClueWeb09[1], it is necessary to retrieve more than $1,000$ documents, but, for navigational queries, this can be decreased when anchor text is used.

Then, for each retrieved query-document pairs, we have to determine a good set of features that effectively capture the search task. This second step is crucial for the success of the learning to rank approaches. Different feature selections can lead to major differences on the accuracy of the methods. In IR, multiple features have been proposed and evaluated for the task, from query independent document properties such as the PageRank (Brin and Page, 1998) or the length of the document to the use of various weighting models (Macdonald et al., 2013b). Once such features have been computed, it is possible to train the model.

Afterwards, the trained model can be used to provide search results. Then, given a query received by the system, the procedure is described as follows:

1. Use the same IR model as before to select k initial documents.
2. Generate the features for each query-document pair. Features should be the same as used in training.
3. Using the feature vectors for each document, obtain the score of the document.
4. Sort the documents according to their scores to produce the final ranking.

Learning to rank in contact recommendation

As our goal is to apply learning to rank for the contact recommendation task, similarly to how we adapted IR models and axioms in Chapters 5 and 6, we have to adapt the steps we describe in the previous section to this new task. For doing this, we use, again, the mapping in Sections 5.2 and 5.3. We first substitute target users for queries, candidate users for documents and we compute the features for pairs of users.

In order to train these systems we need two disjoint sets of edges: the training and validation networks for the contact recommendation task. However, this time, these networks are not used for hyperparameter tuning. Instead of this, they are taken for generating the feature vectors for the target-candidate user pairs, and to obtain relevance judgements. The relevance judgements are taken from the validation set: the edges that are present in that set are taken as positive examples, whereas those which do not appear are considered as non-relevant for the learning to rank method. Then, the training set is used to build the feature vectors: first, a standalone recommender is used to retrieve the top k candidate contacts, i.e. to do the candidate pruning. Then, for each retrieved target-candidate user pair, we generate its feature vector (for instance, by computing the scores of several recommendation algorithms for that pair).

This set of relevance-labelled features is then split into new training and validation subsets, in a way that all feature vectors of the same user are assigned to the same side of the split (that way, it is the same as for the text search task, where the sets of queries for the training and validation set are disjoint). Then, using both sets as input, the learning to rank approaches can be trained without further modification.

[1] ClueWeb09 dataset: https://www.lemurproject.org/clueweb09.php/ (Accessed 19th December 2020)

Table 7.3: Relation between the split described in Section 4 and the elements for learning to rank.

Data subsets in unsupervised recommendation	Learning to rank elements
Training subset	Training/validation features
Validation subset	Training/validation relevance judgments
Input subset	Input features for evaluation
Test subset	Test relevance judgments

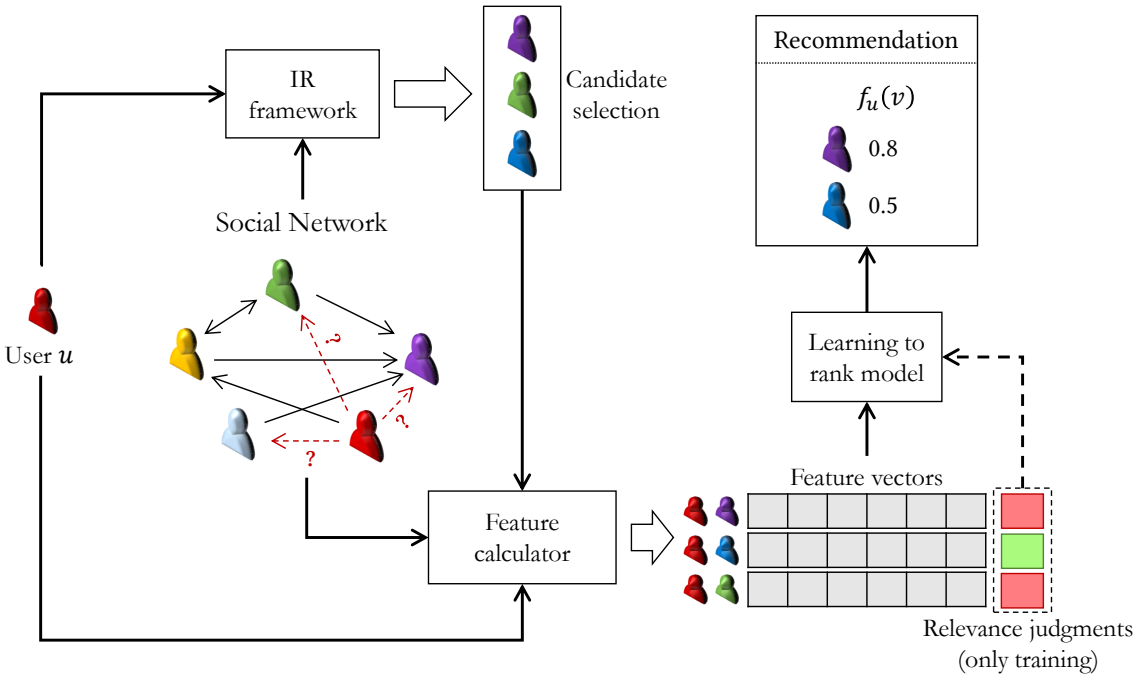


Figure 7.3: Framework for using learning to rank for contact recommendation.

Finally, taking the current state of the network (which, in our experiments it is represented as the union of the training and validation networks), we run the trained learning to rank algorithms in the same way it is done for text search: for each target user, we find a suitable set of candidate users, we generate features for each of them and, finally, we produce the final recommendations. Evaluation for this methods is the same as for the unsupervised ones.

We illustrate this procedure in Figure 7.3, whereas Table 7.3 summarizes the relations between the elements in the network splits for an offline evaluation task, as described in Chapter 4 and the feature sets in learning to rank, as we have just described.

A learning to rank algorithm: LambdaMART

As an effective and representative learning to rank approach, we explore in this thesis the use of the LambdaMART algorithm (Burges, 2010, Wu et al., 2008) as a contact recommendation algorithm. We provide here an explanation for this specific model. According to the taxonomy of learning to rank techniques, LambdaMART is a listwise algorithm, which tries to optimize ranking metrics such as nDCG (Järvelin and Kekäläinen, 2002) or ERR (Chapelle et al., 2009) by using gradient-boosted regression trees. The full details of the algorithm can be studied in (Burges, 2010), but, here, we summarize on how this method works.

Gradient boosting is a technique that allows the combination of several weak models into strong and effective ensembles (Ganjisaffar et al., 2011). The construction of such ensembles is iterative: on each iteration of the algorithm, a new ensemble is built by adding a new model to the ensemble in the previous iteration. We can define the score of such ensembles as:

$$f(x) = f_m(x) = f_{m-1}(x) + \gamma_m h_m(x) \tag{7.6}$$

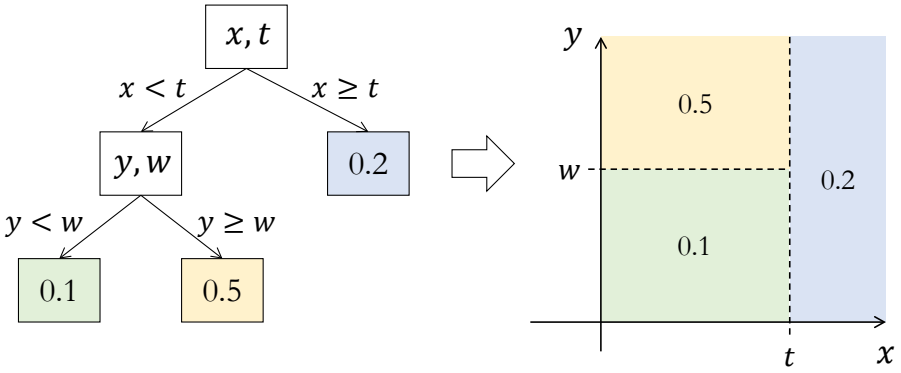


Figure 7.4: Regression tree example. Here, x and y represent two different features, and t and w the decision thresholds. The colors of the leaves correspond with their associated region in the space partition on the right.

where m is the total number of weak models in the ensemble, γ_i is the weight of the i-th model, $h_i : \mathbb{R}^d \to \mathbb{R}$ is the score provided by the i-th model to a feature vector, x, and $f_i(x)$ is the ensemble after i-th training steps. Given an ensemble with $i-1$ models, the i-th approach is selected by optimizing a cost function C in terms of f_{i-1}. For optimizing such function, gradient boosting applies gradient descent in the space of the possible functions that might be selected for the ensemble.

In the LambdaMART algorithm, we first map the gradient boosting vectors with the feature vectors for each query-document pair (or, in the case of contact recommendation, the feature vectors for each target-candidate user pair). Then, the cost function is selected so optimizing it is equivalent to optimizing a ranking metric. Finally, the weak models LambdaMART chooses are regression trees.

A regression tree (Bishop, 2006, Burges, 2010, Ganjisaffar et al., 2011) is a machine learning model that maps feature vectors $x \in \mathbb{R}^d$ to real values with the help of a binary tree. Such binary trees provide a partition of the space as follows: first, we take the root of the tree and we select a feature j and a threshold t. Then, from that root node, two children appear: the first children represents the region where $x_j < t$, whereas the other represents the region where $x_j \geq t$. The process is repeated several times, and, at the end, each leaf of the tree represents a separate region, each one with an assigned value (which is obtained during training). Then, in order to give a vector a real value, it is enough to determine the region of the feature space the vector falls in. We illustrate an example of regression tree in 7.4. In LambdaMART, decision trees are limited to a fixed number of leaves, l.

Then, considering all the information above, once the algorithm has been trained, we can express the LambdaMART score for contact recommendation as:

$$f_u(v) = \sum_{i=1}^{m} \gamma_i h_i(u, v) \tag{7.7}$$

where m is the number of trained regression trees, γ_i is the weight of the i-th tree and $h_i(u, v)$ is the score of the i-th tree for the feature vector for the (u, v) pair, where u and v are, respectively, the target and candidate users of the recommendation.

7.3.2 Experimental Setup

By applying the translation of learning to rank from text search to contact recommendation, we want to study whether it is possible to obtain a better performance by using supervised learning to rank techniques, as opposed to the unsupervised models (both as standalone approaches and as similarities within a kNN scheme) described in Sections 5.3 and 7.2. For this purpose, we describe here the experimental setting for the learning to rank algorithms: the sampling approach, the set of features and how to build the training, validation and test sets.

Table 7.4: Groups of features.

Group	Algorithms
Probability ranking principle (PRP)	BIR, BM25, extreme BM25
Query likelihood (QL)	QLD, QLJM, QLL
Divergence from Randomness (DFR)	DFRee, DFReeKLIM, DLH, DPH, PL2
Vector space model (VSM)	VSM
Link prediction (LP)	Adamic-Adar, Cosine, Jaccard, MCN
Item-based (IB kNN)	All item-based kNN algorithms
User-based (UB kNN)	All user-based kNN algorithms
All IR models (IR)	VSM, PRP, QL, DFR
Friends-of-friends (FOAF)	IR, LP
Nearest neighbors (All kNN)	UB kNN, IB kNN
All models	FOAF, UB kNN, IB kNN

Sampling approach

As a first configuration step, we describe the selection of the sampling approach. Taking as an example the way it is done in text search, we use the direct adaptations IR models as our sampling method (Liu, 2007, Macdonald et al., 2013a). We want to keep the same principle as in such task: first, in text IR, unsupervised models provide an effective way to retrieve documents related to the query (similarly to how they provide good results for contact recommendation); and second, if we understand the query/term/space as a tripartite graph (in the same way as in Figure 6.9), IR models return graph nodes (documents) at distance two from the query node. If we want to follow the same principles, we have to consider sampling models that return "document" users at undirected distance two from the target users. IR models also follow this constraint.

Then, taking the standalone IR recommenders described in section 5.3, we take the optimal configurations for each network, as described in Table C.4. Then, on the validation set, we select the method that maximizes recall at the same cutoff as the one we choose for the sampling approach: $k = 1,000$. Then, the selected samples are the following: BIR for Facebook and 200-tweets follows networks, BM25 for the 1-month interaction graph, and extreme BM25 for the remaining ones.

Feature selection

Next, we have to generate labelled features for each pair of sampled target-candidate users. As both friends of friends and kNN methods provide competitive baselines for contact recommendation, following previous works in IR (Macdonald et al., 2013b), we have chosen to combine the outcomes of several contact recommendation algorithms as learning to rank ensembles, to improve their individual effectiveness. In particular, we list the selection of ranking functions in Table 7.4, classified by families.

However, we do not take the recommendation scores as they are. Instead of tanking their raw values, we take a common approach that it is applied for combining multiple algorithms: we normalize the ranking scores by the min-max function (Lee, 1997). For each target user u, we normalize the feature values into the $[0, 1]$ interval by:

$$\bar{f}_u(v) = \frac{f_u(v) - \min_{w \in \mathcal{U}_u} f_u(w)}{\max_{w \in \mathcal{U}_u} f_u(w) - \min_{w \in \mathcal{U}_u} f_u(w)} \tag{7.8}$$

where $\mathcal{U}_u$ represents the set of candidate users that have been sampled for the target user u. If a target-candidate pair is not retrieved by the feature algorithm, we set $\bar{f}_u(v) = 0$.

Network splits and hyperparameter tuning

As we describe in the previous section, we use as a basis of the algorithm the training/validation/test set partitions we describe in Chapter 4, but we give them different uses. We summarize their new uses in Table 7.3: the training network is used to generate the training and validation feature vectors, the validation network is used to generate the training and validation labels and the union of both (the one we refer to as the input network) is used to produce the features for recommendation. The test network is then used to evaluate the approaches in the same manner as we do in our previous experiments.

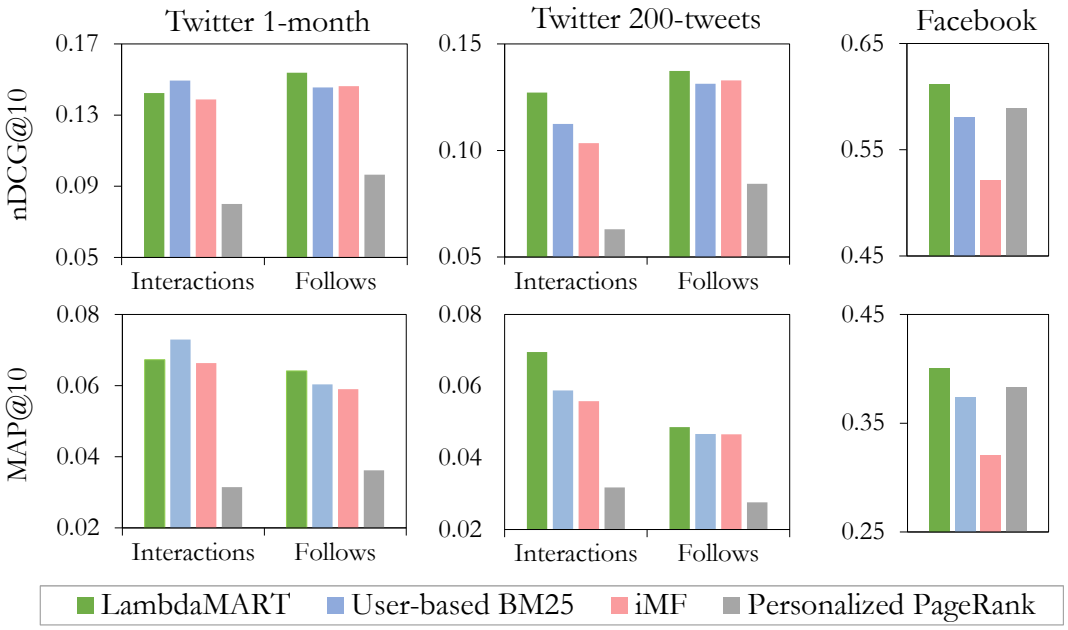


Figure 7.5: Comparison between LambdaMART and the best algorithms in previous comparisons in terms of nDCG@10 (upper row of figures) and MAP@10 (lower row). Full statistical significance values are detailed in Figure D.16 in Appendix D.

The only difference is that, once the labelled features for the training set are generated, we have to build the validation set. For that we randomly take the features for 80% of the target users in the network as the training set, and the features for the remaining 20% as validation.

We use as learning to rank algorithm the LambdaMART algorithm we define above. In particular, we use the Jforests LambdaMART implementation developed by Ganjisaffar et al. (2011). Differently from the experiments in previous chapters, we do not vary here the hyperparameters of the learning to rank algorithm (properties such as the number of trees or the number of leaves of each tree) – we use the default parameters provided by the authors in their implementation[2]. Applying a parameter tuning would only likely improve the effectiveness of the approach. However, as we show later in Section 7.3.3 using the default parameters already makes this method outperform unsupervised baselines in most cases.

7.3.3 Results

In the following, we first report the performance of learning to rank using all the defined models as ensembles. Afterwards, we assess the relative importance of the different features we use for learning to rank.

General effectiveness

First, we check the effectiveness of LambdaMART when we use all the algorithms in Table 7.4 as features for the model. Figure 7.5 shows the comparison of the ensemble with the top standalone recommendation algorithms in the experiments in Section 7.2.3: the implicit matrix factorization (Hu et al., 2008) algorithm as the baseline to beat in Twitter networks, personalized PageRank (White and Smyth, 2003) as the best approach in Facebook and, finally, user-based with BM25 similarity as the most effective IR-based algorithm across all networks. We note that, excepting the 1-month interactions dataset, where LambdaMART is behind the user-based kNN approach, in the rest of networks, LambdaMART manages to beat all the other algorithms, thus resulting in the most effective method in our experiments. This advantage is always statistically significant (for both two-tailed paired t-test and Tukey HSD with p-values $p < 0.05$) for both nDCG@10 and MAP@10, with the exception of the latter metric for the 200-tweets follows graph.

[2]JForests GitHub repository https://github.com/yasserg/jforests (Accessed 19th December 2020)

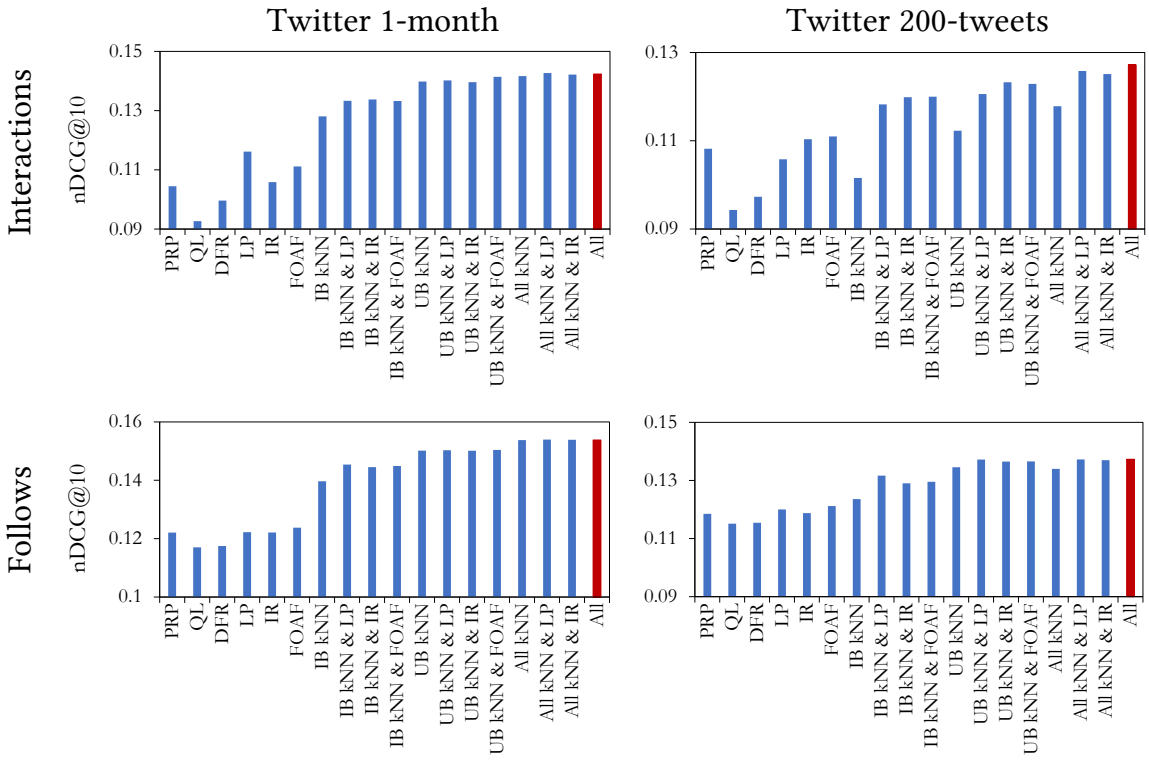


Figure 7.6: Comparison of LambdaMART models with different sets of features, defined in Table 7.4 for the Twitter networks.

In the remaining network, LambdaMART manages to obtain a small (though not significant) advantage over implicit matrix factorization, making it the second best approach. In that dataset, LambdaMART does not beat the user-based kNN on that dataset, despite including it as a feature. The reason why this occurs lies in the properties of nearest neighbor techniques: as we explain in Section 7.2, user-based approaches are able to recommend users up to distance three from the target user, while we intentionally restricted learning to rank algorithms to recommend users only at distance 2, as a result of the initial selection of candidate users by the IR models (in this case, BM25).

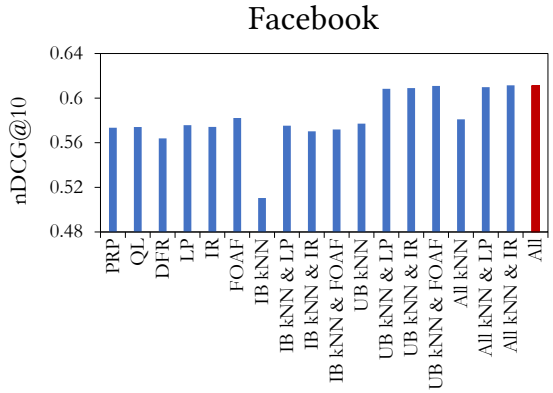


Figure 7.7: Comparison of LambdaMART models with different sets of features, defined in Table 7.4 for the Facebook network.

Feature Selection

Beyond the general effectiveness of the learning to rank model using all the possible features, we also want to identify which specific features make learning to rank so effective for the contact recommendation task. To

answer this question, we train different LambdaMART methods with different feature selections. We show the tested groups in Table 7.4. After that, we compare them in terms of nDCG@10 to analyze the differences between the learning to rank variants. We report the results in Figures 7.6 and 7.7.

As we can see, in all networks, the combination of all features provides the better results, thus showing that all of them are important for the recommendation. Beyond that, if we deepen on the analysis, we observe some commonalities and differences between the networks. When it comes to the IR algorithms, it is easy to see that the probability ranking principle (PRP) models usually achieve the best results, followed by divergence from randomness and query likelihood. This is a bit different in the Facebook network, where query likelihood models have a small advantage over the PRP models. The combination of link prediction friends of friends approaches tends to work better than the combination of all IR methods. The only exception to this is the interaction network of the 200-tweets dataset, where BM25 is so effective that counteracts this, making link prediction achieve slightly worse results than in other graphs.

Finally, for all three of the graphs, item-based kNN ensembles provide a better performance than the standalone friends of friends methods, and they represent the second best set of features after user-based kNN. The two exceptions for this are the Facebook network, where item-based kNN is among the worst features (as it occurred in our experiments in Section 7.2.3) and the 200-tweets interactions networks where, again, the high accuracy of BM25 makes it difficult for item-based approaches to perform better than IR models on their own.

7.4 Conclusions

In this chapter, we have expanded the study on the utility of information retrieval models for contact recommendation. For that, we have studied three additional uses of information retrieval models: first, as similarities in a nearest-neighbor collaborative filtering scheme (Ning et al., 2015, Valcarce et al., 2017) and second, as both features and samplers for the adaptation of learning to rank methods (Liu, 2007, Macdonald et al., 2013b) for contact recommendation. We have conducted experiments oriented to check whether these alternative uses of IR lead to improvements on the accuracy when compared to the standalone approaches and the top-performing baselines in previous sections, as implicit matrix factorization. The main conclusions from this study are summarized below:

- Any contact recommendation algorithm can be straightforwardly applied as a similarity in a kNN scheme, since, despite their differences, both contact recommendation and neighborhood selection tasks can be defined in a similar manner.
- Interestingly, IR models have been shown to be consistently more effective when they are used as neighbor selection methods within user-based kNN approaches. Even if they are not particularly effective at recommending contacts, they are effective at finding good neighbors for such algorithms.
- The aggregation of standalone, user-based and item-based kNN IR-based algorithms on a learning to rank scheme generally improves the accuracy of these methods for the contact recommendation task even further, consistently beating the state of the art approaches in all the tested datasets.
- Overall, IR models have been found effective in three different roles: as direct recommendation algorithms, as neighbor selectors and as samplers and features in learning to rank schemes.

Part III

Beyond accuracy in contact recommendation

"Even the technology that promises to unite us, divides us. Each of us is now electronically connected to the globe, and yet we feel utterly alone."
Dan Brown

Contact recommendation and link prediction have mainly been addressed as accuracy-targeting tasks, where their only goal is to correctly predict which edges are going to appear in the network. However, the great influence these approaches have on the users makes it interesting to analyze some other properties, like the novelty or the diversity of the predicted edges, so undesirable phenomena such as filter bubbles or glass ceiling effects can be prevented, or, at least, mitigated.

In Chapter 8 we discuss different evaluation dimensions of contact recommendation and link prediction beyond their accuracy. We set a particular focus on the positive effects these approaches might have on the evolution of the network structure – distance reduction, balance of the degree distribution, reduction of the redundancy of the edges and creation of links between communities. We also explore other utility dimensions by adapting the notions of novelty and diversity from the recommender systems and information retrieval fields, and provide some initial empirical observations over a set of algorithms using Twitter and Facebook data.

Afterwards, in Chapter 9, we study the possible value that structural diversity metrics – related to the notion of weak tie – might have for the mitigation of filter bubbles, by analyzing the effect contact recommendation approaches enhancing these metrics have on the novelty and diversity of the propagated information in social networks. We find that promoting the structural diversity of the networks (by adding links between communities and non redundant links) leads to a more rich flow of information reaching the users of the network.

8

Impact on network structure

Highlights

- *We propose several evaluation dimensions beyond accuracy: the novelty and diversity of the recommendations.*
- *We explore the potential positive effects contact recommendation approaches have on the evolution of the network, targeting the concept of structural diversity.*

The predominant perspective in the design and evaluation of contact recommendation (and, as a related task, link prediction) approaches addresses the task as an accuracy-targeting problem, where the aim is to increase the density of the network as much as possible (Backstrom and Leskovec, 2011, Goel et al., 2015, Hannon et al., 2010). In evaluation, ranking-based metrics like precision or recall (Baeza-Yates and Ribeiro-Neto, 2011) or classification ones like the area under the ROC curve (Fawcett, 2006) fall under this consideration. Suggesting links that users are willing to create is, obviously, essential for the success of a contact recommendation approach. However, it is not the only quality that might enhance the value and performance of online social networks for their users or the businesses running on the network.

This accuracy maximization perspective is oriented to optimize microscopic properties of the network: it just measures to what extent the number of links in the network grows. For increasing the density of the network, any correctly recommended link contributes to accuracy metrics as much as any other correct one, regardless of who the endpoints of the edge are, their location in the network, their social involvement or influence, etc. To stress this even more, when we use metrics like precision and recall, we assess the benefit that the recommendation brings to each target user, isolated from the rest, and then we aggregate the individual gains by averaging them over all the users to observe a "macroscopic" value.

Nevertheless, as their name indicates, social networks are not about isolated users, but about how multiple people connect and interact to each other. In this networks, it is well understood that a microscopic change, such as the formation of a single link has immediate direct and indirect effects not only for the endpoints of the edge, but also in their surroundings. The combination of a few of these small changes spread across the network might produce something bigger than the sum of the individual effects – even lead to substantial macroscopic changes on the properties and behaviour of the social network as a whole.

Therefore, it seems natural to consider what a recommendation can bring to the social network from a wider perspective. A non negligible fraction of the new edges that appear in social networks are created thanks to being recommended (Aiello and Barbieri, 2017, Goel et al., 2015, Guy, 2018). So link prediction and recommendation represent an opportunity to drive the evolution of the network towards desirable global properties, beyond (and in addition to) the short-term micro level value that the accuracy of the recommendation provides.

Understanding how a network evolves and assessing the probability that any possible link appears in it are very closely related tasks. Thus, link prediction and network growth modelling have been seen as equivalent problems to some extent. However, few works have addressed the impact that contact recommendation and link prediction have on the evolution of the network (Aiello and Barbieri, 2017, Huang et al., 2013, Su et al., 2016), and, to the best of our knowledge, this impact has barely been considered as part of the utility we seek of recommendation approaches.

In this chapter, we discuss and explore several perspectives on this direction. First, we take as a starting point the wide array of concepts, metrics and analytic methods that social network analysis proposes for measuring the structural properties of networks. We explore these notions to define novel evaluation metrics that assess the effect that link recommendation has on the network structure. Although selecting a desirable set of network properties might depend on the specific purpose of the network, we select here two notions that we consider

to be potentially positive: the reduction of the distances between users and the structural diversity of the network(Burt, 1995, De Meo et al., 2014, Granovetter, 1973, Ugander et al., 2012). Moreover, the recommender systems and information retrieval fields have developed an awareness that accuracy alone is just a partial view on the value of recommendation, and perspectives as novelty and diversity (Castells et al., 2015) can also be important. Accordingly, we consider the adaptations of metrics in those areas for evaluating link recommendation algorithms, and we find that, at more than one level, the global network analysis dimension of recommending edges in networks links to the underlying principles of novelty and diversity.

The contents of this chapter were partially published in the following research articles:

- **Sanz-Cruzado et al. (2018b):** Javier Sanz-Cruzado, Sofía M. Pepa and Pablo Castells. Structural Novelty and Diversity in Link Prediction. In *Proceedings of the 9th International Workshop on Modelling Social Media (MSM 2018), in Companion of the The Web Conference 2018 (WWW 2018)*, pages 1347–1351, Lyon, France, April 2018. IW3C2.
- **Sanz-Cruzado and Castells (2018b):** Javier Sanz-Cruzado and Pablo Castells. Enhancing structural diversity in social networks by recommending weak ties. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018)*, pages 233–241, Vancouver, Canada, October 2018. ACM.
- **Sanz-Cruzado and Castells (2019b):** Javier Sanz-Cruzado and Pablo Castells. Beyond Accuracy in Link Prediction. In *Proceedings of the 3rd Workshop on Social Media Personalization and Search (SoMePeaS 2019) co-located with the 41st European Conference on Information Retrieval (ECIR 2019)*, number 1245 in Communications in Computer and Information Science, pages 79–94, Cologne, Germany, April 2019. Springer.

8.1 Research questions

The contents of this chapter address the following

- **RQ1:** What can novelty and diversity mean in the context of contact recommendation?
- **RQ2:** How can we measure the structural effects contact recommendation algorithms have on the social network?
- **RQ3:** How do the different proposed metrics correlate with each other?
- **RQ4:** Which algorithms are best and worst for each metric?

8.2 Social network analysis

As a first perspective to consider, we notice that a non negligible fraction of the new edges in a social networks come from the recommendations provided by the different platforms (Goel et al., 2015, Guy, 2018). Consequently, the contact recommendation networks have a potential to direct the evolution of the network and its topological properties. Therefore, it seems natural to study these effects. To some extent, the effect of the recommendation on the structural properties in the graph is also considered when we evaluate the accuracy of the different approaches – the accuracy somehow measures how much the density of the network increases as a result of the recommendation –, but many other properties can be analyzed. In this section, we propose several metrics that can be considered for evaluating contact recommendation and link prediction methods, targeting topological properties of the networks.

A way to assess the effect that recommendation algorithms have on the evolution of the network is to consider an extension of the network $\mathcal{G}' = \langle \mathcal{U}, E' \rangle$, where $E' = E \cup \hat{E}$ by some subset of the recommended (or predicted) links, $\hat{E}$. For instance, we can consider the union of the top k predicted outgoing links for each user in the network, as if the group of users to whom we deliver the recommendations accepted all the links in $\hat{E}$. Over that expanded network, we can consider any social network analysis metric as a metric for the contact recommendation method.

Social network analysis is a very rich field, where myriads of metrics and concepts have been proposed to characterize and measure the network structural properties (Newman, 2018). Here, we select some classical

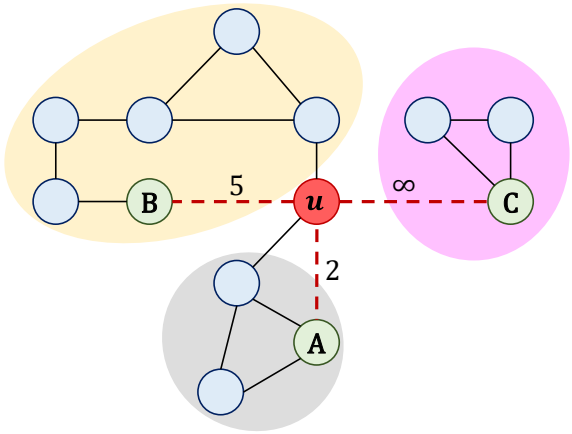


Figure 8.1: Possibilities for recommending people at different distances

properties we find interesting for the perspective we discuss. Some of our proposals are straightforward adaptations, whereas other require further elaboration.

8.2.1 Network distance

In addition to the effect of a recommendation on the density of the network, another perceivable effect is a general reduction of the distances in the graph. However, in this case, differently from the density, different recommended edges provide different distance reductions. This can be observed in Figure 8.1: considering node u as the target user of our recommendation, if we recommend him user A, distances between nodes in the yellow group and such node will be diminished by one step; however, if we recommend B to user u, some of these distances are reduced by up to 4 hops (for example, distance between A and B); and, what is more, if we recommend node C, we even connect separated components, allowing people from the yellow and grey groups to connect to users in the pink one. Shortening the distances between users is likely a desirable property in a network, since it makes people easier to reach from each other through a smaller number of hops through common acquaintances (and even, make some users reachable through those hops). Therefore, we propose several metrics for analyzing this effect. For the definitions of all the metrics, we define, in this section, $\delta(u, v)$ as the distance between nodes u and v in the original graph $\mathcal{G}$ and $\delta'(u, v)$ as the distance between them in the extended network $\mathcal{G}'$.

Average reciprocal shortest path length (ARSL)

We start by considering all the shortest path lengths between each pair of users in the network and averaging them. If we did this using a simple arithmetic mean, this would just be one of the most common social network metrics to measure distances (Newman, 2018): the average shortest path length (ASL). However, instead of an arithmetic mean, we average the inverse value of the lengths. There are two reasons for this: first, that the closest the nodes in the expanded network are the "better" the recommendation approach is; and, second, this way, we can consider distances between unconnected components as undesirable, since those distances are infinite. We denote this metric as average reciprocal shortest path length (ARSL), and its formulated as:

$$\text{ARSL}(\mathcal{G}') = \frac{1}{|\mathcal{U}|(|\mathcal{U}|-1)} \sum_{u,v} \frac{1}{\delta'(u,v)} \tag{8.1}$$

which takes values between 0 (if all the nodes are isolated) and 1 (if we have a complete network where every user is connected to each other). This metric is related to the concept of closeness of a node (Newman, 2018). The closeness of a node finds how far a single node is from each of the users in the network, and it is typically computed as:

$$\text{close}(u|\mathcal{G}) = \frac{|\mathcal{U}|-1}{\sum_{v \in \mathcal{U}} \delta(u,v)} \tag{8.2}$$

but, in unconnected networks, the value for this would be 0 for all the nodes. As an alternative definition to undertake this problem, the closeness can be also defined as the harmonic mean of the distances between a user and the rest of people in the network:

$$\text{close}(u|\mathcal{G}) = \frac{1}{|\mathcal{U}|-1}\sum_{\substack{v\in\mathcal{U}\\ v\neq u}}\frac{1}{\delta(u,v)} \tag{8.3}$$

Following this second definition of closeness, we also find our ARSL metric by averaging it over the whole set of people in the social network.

Eccentricity

Another possibility that we consider to measure distances between users is the notion of eccentricity. The eccentricity of a user (Dankelmann et al., 2004, Ilić, 2012) is the maximum distance from the user to any other (reachable) node in the network:

$$\text{ecc}(u|\mathcal{G}') = \max_{\substack{v\in\mathcal{U}\\ \delta(u,v)<\infty}}\delta'(u,v) \tag{8.4}$$

We consider two different metrics that use the concept of eccentricity. The first, takes the maximum eccentricity, also known as the diameter (the maximum distance between two nodes in the network). We propose the use of the reciprocal of this value as a metric. We denote it as **reciprocal diameter (RD)**, and it is computed as:

$$\text{RD}(\mathcal{G}') = \frac{1}{\max_{u\in\mathcal{U}}\text{ecc}(u|\mathcal{G}')} \tag{8.5}$$

The other metric we consider is the average eccentricity of the network. But, in order to make the value greater as the eccentricity values are reduced, we take the harmonic mean of the eccentricity. We denote this as the **reciprocal average eccentricity (RAE)**:

$$\text{RAE}(\mathcal{G}') = \frac{|\mathcal{U}|}{\sum_{u\in\mathcal{U}}\text{ecc}(u|\mathcal{G}')} \tag{8.6}$$

Both metrics take values in the range $(0, 1]$, and just consider improvements over the resulting connected components of the network.

Mean prediction distance (MPD)

The last distance metric, which we denote as mean prediction distance (MPD), measures in a straightforward way how the recommended edges bring people far from their usual social environment. This, under a recommendation point of view, can be seen as a measure of the novelty. We define this metric as the harmonic mean of the reciprocal distances between the target and the recommended users in the original network:

$$\text{MPD}(\hat{E}|\mathcal{G}) = \frac{|\hat{E}|}{\displaystyle\sum_{(u,v)\in\hat{E}}\frac{1}{\delta(u,v)}} - 2 \tag{8.7}$$

where we subtract 2 to set the minimum value of the metric at 0 (since $\delta(u,v) \leq 2\,\forall(u,v)\in\hat{E}$, as $\delta(u,v) < 2 \Rightarrow (u,v)\in E$, so those edges cannot be recommended). This metric takes values in the $[0,\infty]$ range. The latter value is only reached if all the recommended links are global bridges (Granovetter, 1973), i.e. all edges link two different disconnected components. In this metric, we use the harmonic mean since it can take in infinite distances.

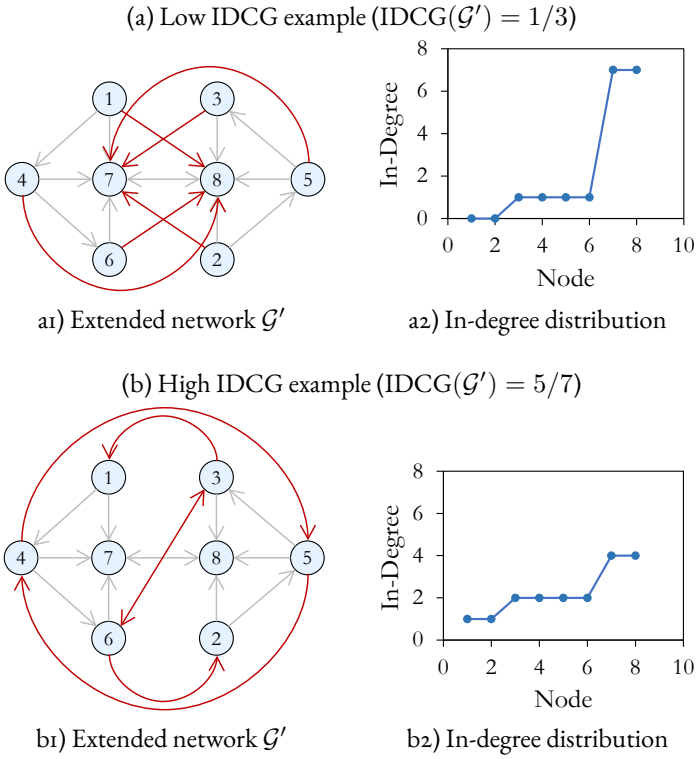


Figure 8.2: IDCG example. Top row shows an example with a skewed in-degree distribution in the extended graph (thus showing low IDCG value) wheras bottom row shows a more balanced distribution (with high IDCG value). Numbers in the graphs indicate the order of the nodes (by increasing in-degree). Grey lines represent previously existing edges, and red ones the recommended links for each user.

8.2.2 Structural diversity

As we observe in section 2.3.3, the concept of structural diversity in social networks has attracted the attention of many scholars under multiple perspectives (Barefoot et al., 2005, Burt, 1995, De Meo et al., 2014, Granovetter, 1973, Ugander et al., 2012). From the simplest perspective, we can consider the degree distribution as a primary (and simple) sign of connective diversity: a very skewed distribution reflects the concentration of most links in the network among some high degree users, whereas in a flatter distribution, each person has its own social circle, and is exposed to different interactions than other people. We can summarize the "flatness" of the distribution using the Gini index (Dorfman, 1979), a measure of the inequality among values of a frequency distribution. We can reverse this index into the **degree Gini complement (DGC)**, which we define as:

$$\mathrm{DGC}(\mathcal{G}') = 1 - \frac{1}{|\mathcal{U}| - 1} \sum_{i=1}^{|\mathcal{U}|} (2i - |\mathcal{U}| - 1) \frac{|\Gamma'(u_i)|}{|E'|} \tag{8.8}$$

where the people u_i are sorted by non-decreasing degree $|\Gamma'(u_i)|$, where $\Gamma'(u)$ represents the neighborhood of the user u in the extended graph $\mathcal{G}'$. Similarly to what we did for the distance metrics, we take the complement of the Gini index, taking values in the $[0, 1]$ interval, so greater values indicate a flatter distribution of the edges whereas values near zero show strong link concentration. When applied to link prediction, in directed network, it makes sense to decouple this metric in two: the in-degree and out-degree versions, which we respectively denote as IDGC and ODGC. However, when we apply it to the recommendation problem, and the same number of contacts are recommended to each user, only the in-degree version makes sense. We illustrate this variant with an example in Figure 8.2.

Richer notions of structural diversity are related to the concept of weak tie (Aral, 2016, De Meo et al., 2014, Granovetter, 1973). As we introduce in section 2.3.3, we have a sociological definition of the weak ties, which depend on the network domain. However, it is common to define the weakness of links in terms of their topological properties (De Meo et al., 2014, Granovetter, 1973), by assessing to what extent they are redundant

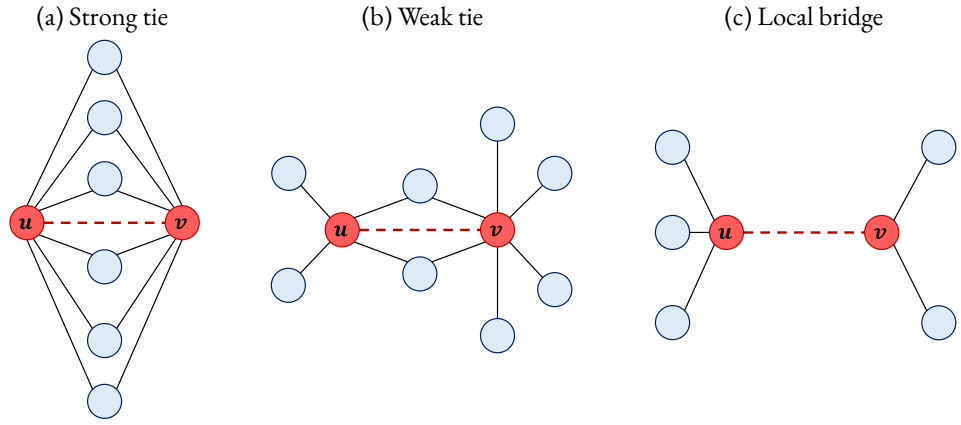


Figure 8.3: Examples of the embeddedness of an edge. Connections between blue nodes are not indicated.

in the network: a link is weak inasmuch as it is not redundant to the other links on its surroundings. Weak ties thus represent a somehow exclusive (and hence valuable) connection between specific individuals or regions of the network. We divide two groups of measures: global and local.

Local notions

Local notions of weak tie just consider the close environment of the link to determine its strength. A first possibility is considering the local notion of weak ties proposed by Granovetter (1973): he defines the notion of local bridge: a link between people who do not have any common neighbors. The binary nature of such definition is quite restrictive, deeming just counting the number of such links a coarse metric. Instead of this, we can use the edge embeddedness, a finer and more informative metric that measures the proportion of shared neighbors between two users, to assess the strength of a tie (Easley and Kleinberg, 2010, Zhao et al., 2010):

$$\text{embed}(u, v|\mathcal{G}') = \frac{|\Gamma'_{out}(u) \cap \Gamma'_{in}(v)|}{|\Gamma'_{out}(u) \cup \Gamma'_{in}(v)|} \tag{8.9}$$

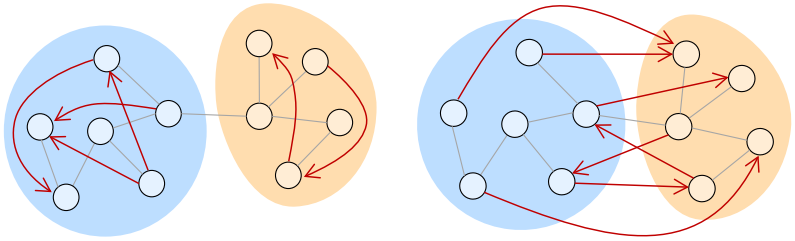
This metric provides a generalization of the concept of weak tie: if an edge has embeddedness equal to 0, it is a local bridge under the definition proposed by Granovetter (1973). We show some examples of strong and weak ties in Figure 8.3. If we want to measure to what extent a contact recommendation approach suggests weak ties, it is enough to measure the average of the complement of the embeddedness – which we shall call weakness – over the suggested links: the metric we name **average edge weakness (AEW)**:

$$\text{AEW}(\hat{E}|\mathcal{G}') = \frac{1}{|\hat{E}|} \sum_{(u,v)\in\hat{E}} \big(1 - \text{embed}(u, v|\mathcal{G}')\big) \tag{8.10}$$

This metric obtains values between 0 and 1 in a way that, the higher the value of the metric, the higher the weakness of the recommended edges.

In addition to this, we may consider triadic closure as the smallest unit of structural redundancy. If we do this, we can use the clustering coefficient. For a single node, the clustering coefficient is just defined as the ratio of neighbor pairs that are connected to each other. A global coefficient can be computed by averaging these value over all the people in the network (Watts and Strogatz, 1998), or by an alternative global definition: the ratio of closed triads in the network over paths of length two (Newman, 2018). We illustrate open and a closed triads in Figure 2.7. Similarly to how we do for other metrics, using the global definition, we take the **complementary clustering coefficient (CCC)** to get the metric values aligned with the notion of diversity we want:

$$\text{CCC}(\mathcal{G}') = 1 - \frac{|\{(u, v, w)|(u, v), (v, w), (u, w) \in E'\}|}{|\{(u, v, w)|(u, v), (v, w) \in E' \wedge u \neq w\}|} \tag{8.11}$$



(a) High redundancy (low MC)

(b) Low redundancy (high MC)

Figure 8.4: Modularity complement examples. Red links represent recommendations.

Global notions

Along with the local notion of bridge, Granovetter (1973) also proposed a global one: a global bridge is just a link connecting two different connected components. Nevertheless, the giant component phenomenon in social networks, for which social networks have a big connected component, outside of which only a few nodes remains, makes this, again, a rather restrictive definition. We consider instead a relaxed definition of global link, as proposed by De Meo et al. (2014), where we consider as weak ties connections between communities, taking links inside the clusters of nodes as the strong ones. Considering this notion of weak tie, different metrics assessing their presence can be considered.

A first and classical metric for quantifying the presence of links between communities is the so-called modularity (Clauset et al., 2004) – and, reciprocally, multiple community detection algorithms seek partitions minimizing it. Given a community partition of the network, $\mathcal{C}$, this metric compares the number of links between communities (strong links) to the expected number of links we would find in a random network generated by a configuration model (Newman, 2018), that is, a random network keeping the degree distribution of the network. This can be easily generalized to directed networks as (Arenas et al., 2007):

$$\text{mod}(\mathcal{G}'|\mathcal{C}) = \frac{\sum_{u,v\in\mathcal{U}} \left(A_{uv} - \frac{|\Gamma_{out}(u)||\Gamma_{in}(v)|}{|E'|} \right) \mathbb{1}_{c(u)=c(v)}}{|E'| - \sum_{u,v\in\mathcal{U}} \left(\frac{|\Gamma_{out}(u)||\Gamma_{in}(v)|}{|E'|} \right) \mathbb{1}_{c(u)=c(v)}} \tag{8.12}$$

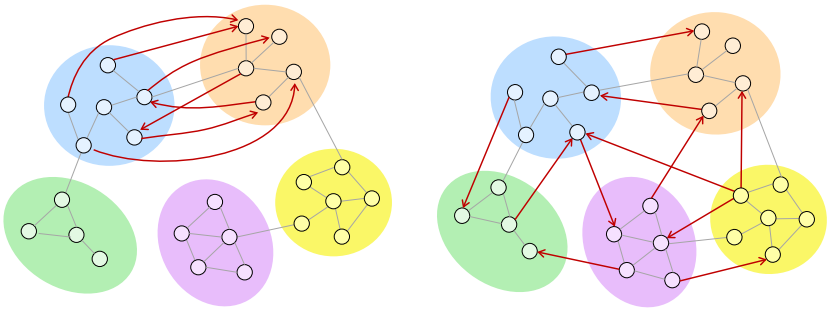
where $c(u) \in \mathcal{C}$ represents the community user u belongs to, A represents the unweighted adjacency matrix of the graph ($A_{uv} = 1 \Leftrightarrow (u, v) \in E'$) and $\mathbb{1}_{c(u)=c(v)} = 1$ if u and v belong to the same community. Since low modularity indicates high diversity (and many weak ties), we apply a linear transformation to reorient the metric and limit it to the $[0, 1]$ range. We denote this modified metric as **modularity complement (MC)**:

$$\text{MC}(\mathcal{G}'|\mathcal{C}) = (1 - \text{mod}(\mathcal{G}'|\mathcal{C}))/2 \tag{8.13}$$

We illustrate in Figure 8.4 two examples: the first one recommends all the links inside communities (obtaining a low modularity complement value) and the second one recommends all the links outside the original community (achieving a high diversity value).

A limitation of the modularity complement metric is that it just provides a raw measure of the abundance of links that cross communities. But, valuable as these links are, it might be interesting to distinguish whether they are concentrated on a few pairs of communities, or widely distributed among the different community pairs. This is illustrated in Figure 8.5. Considering this, we refine the notion by De Meo et al. (2014), taking into account the redundancy between these weak ties. We hypothesize here that, if the weak ties are many and they are well-distributed across many communities, they might be even more beneficial for the network as a whole.

To measure this, we apply the Gini index (Dorfman, 1979) over the frequency of weak ties between each pair of communities, as a measure of how well-spread this links are. Again, as we want the metric to be positively aligned with the structural diversity, we use the complement of the Gini index, defined as follows. First, given a



(a) Weak tie redundancy (b) Weak tie diversity

Figure 8.5: Different distributions of weak ties. Red links represent the recommended edges.

community partition, we count the set of edges crossing two communities, $c_i, c_j \in \mathcal{C}$ as:

$$n_{ij} = \left|\{(u,v) \in E' | c(u) = c_i \wedge c(v) = c_j\}\right| \tag{8.14}$$

Then, we put together these counts into $X = \{n_{ij} | i \neq j\}$ and we sort them in increasing order $X \rightarrow \langle x_1, x_2, ..., x_N \rangle$. We can then define the **inter-community edge Gini complement (ICEGC)** as the complement of the Gini index of X:

$$\text{ICEGC}(\mathcal{G}'|\mathcal{C}) = 1 - \frac{1}{N-1}\sum_{i=1}^{N}(2i - N - 1)\frac{x_i}{|\text{WT}(E')|} \tag{8.15}$$

where $N = |X| = |\mathcal{C}|^2 - |\mathcal{C}|$ and $\text{WT}(E')$ represents the set of weak ties (links between communities) in the network. This metric takes the 0 value if a single pair of communities concentrates all the weak links , and 1 if all pairs of communities have the same number of edges between them.

A limitation of such metric is that it does not consider the strong ties in the network, i.e. we could obtain high values of this metric if we had one link between each pair of communities, but thousands or millions of strong links in the network. Therefore, in its current definition, it is only possible to use this metric as a complement of MC. We can further refine this metric so it also penalizes the number of strong ties in the network. For this, it is enough to add to the X set a new value, which we denote as n_0, counting the number of intra-community edges (strong ties) as:

$$n_0 = \sum_{i=1}^{|\mathcal{C}|} n_{ii} \tag{8.16}$$

If we sort that extended set X' and we sort its values in increasing order, we can define the **community edge Gini complement (CEGC)** as the complement of the Gini index of x':

$$\text{CEGC}(\mathcal{G}'|\mathcal{C}) = 1 - \frac{1}{N-1}\sum_{i=1}^{N}(2i - N - 1)\frac{x'_i}{|E'|} \tag{8.17}$$

where $N = |X'| = |\mathcal{C}|^2 - |\mathcal{C}| + 1$. We show an example of how this metric is computed in Figure 8.6. We can observe that this metric rewards a high number of weak ties and low redundancy between them at the same time. This metric only reaches value 0 when all the links run between a pair of communities (or there are no weak links), and 1 when the same amount of edges crosses every pair of communities (and there are only as many strong intra-community ties as links there are between any pair of communities).

8.3 Novelty and diversity

In addition to the structural effects the algorithms have on the network, we also consider different dimensions beyond accuracy. In particular, diversity. Diversity is a rich concept which has been studied in many different fields and disciplines beyond social network. Pertinent to our research field, information retrieval and recommender systems have developed several notions (Agrawal et al., 2009, Castells et al., 2015, Clarke et al., 2008, Vargas and Castells, 2011) which can be adapted to contact recommendation context (Li et al., 2017).

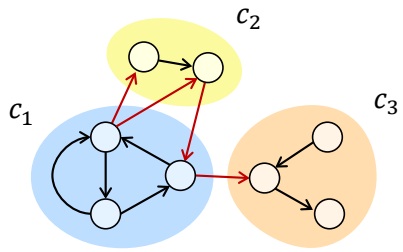
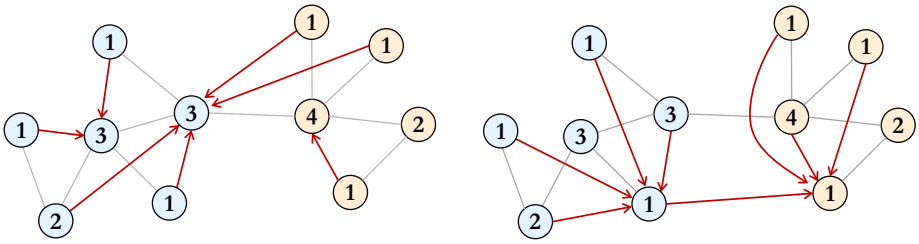


Figure 8.6: Community edge Gini complement example. In this case, we have $n_{12} = 2$,$n_{13} = n_{21} = 1$,$n_{23} = n_{31} = n_{32} = 0$ and $n_0 = 7$. Therefore, $X' \to \langle 0, 0, 1, 1, 2, 7\rangle$, which results in $\text{CEGC}(\mathcal{G}'|\mathcal{C}) = 0.3739$.



(a) Low long tail novelty (b) High long tail novelty

Figure 8.7: Long tail novelty example. Red arrows represent recommended links and numbers in the nodes indicate their degree.

8.3.1 Novelty

The novelty of a system indicates how different the past and present experiences of the users with the systems. It is a primary concern in scenarios where the main goal of a recommender is the discovery of new elements by the users (Castells et al., 2015). Several notions of novelty can be defined. The most common of them addresses recommending those users in the long tail of the popularity distribution, i.e. recommending those users with receive a minority of the links in the network. For instance, in Twitter, the most followed accounts are those of celebrities or very big companies, which, to some extent, are known by a vast majority of the network – whether they follow them or not. Consequently, recommending users with a lower degree is more likely to be surprising to the target user than recommending those famous users.

We can formalize the **long tail novelty (LTN)** as the prior probability that a random person in the network did not know about some of the recommended people to some other random user. This can be estimated by the proportion of people that link (in the original network $\mathcal{G}$) the recommended person v, i.e. $p(\neg\text{known}|v) = 1 - |\Gamma_{in}(v)|/|\mathcal{U}|$. Thus, the metric is defined as:

$$\text{LTN}(\hat{E}|\mathcal{G}) = \frac{1}{|\hat{E}|} \sum_{(u,v)\in\hat{E}} \left(1 - \frac{|\Gamma_{in}(v)|}{|\mathcal{U}|}\right) \tag{8.18}$$

This metric is inversely equivalent to the in-degree of the recommended contacts. In the general recommender systems field, this metric is named the *expected popularity complement* of the recommendation (Castells et al., 2015, Vargas and Castells, 2011). We illustrate two examples of long tail novelty in Figure 8.7: in Figure 8.7a) nodes with high degree are recommended (producing not so novel recommendations) whereas in Figure 8.7b) only nodes in the long tail of the distribution (with degree equal to 1) are recommended.

LTN measures the novelty from a global perspective, indicating how novel the recommended links are to everyone. However, it also makes sense to consider the individual viewpoints, and finding how novel the recommendations are for each specific user. **Unexpectedness** metrics have been proposed for evaluating recommender systems (Castells et al., 2015), assessing the dissimilarity between the recommended items and the items the user already knows (i.e. the items rated by the user in previous interactions with the system). In our case, these records of previous interactions are just the contacts the user already has in the network $\mathcal{G}$. Then, we can

define a metric as:

$$\text{Unexp}(\hat{E}|\mathcal{G}) = \frac{1}{|\hat{E}|} \sum_{(u,v)\in\hat{E}} d(v,w) \tag{8.19}$$

where $d(v,w)$ is the distance between users v and w. It can be defined in any meaningful way for the domain. In our particular case, if we measured this in terms of network distance, we would obtain redundant metrics to the ones in Section 8.2.1. We can also define dissimilarity in terms of the social network links (e.g. Jaccard distance) but, in general, they are not the most informative option: since many algorithm use these similarities as a means for recommending the users, we would obtain tautologically low values. More interestingly, we might define dissimilarity measures based on side information (for instance, the content of tweets).

The two novelty metrics we propose can measure how much the contact recommendation algorithm suggest people outside of the comfort zone of the target users and, consequently, bringing opportunities for broadening and diversifying their experience.

8.3.2 Diversity

Focusing on the concept of diversity, we consider for this work two lines: first, the notion of diversity which has been considered in recommender systems (Castells et al., 2015, Fleder and Hosanagar, 2007, Hurley and Zhang, 2011, Vargas and Castells, 2011, 2014), and then, the one in information retrieval (Agrawal et al., 2009, Chapelle et al., 2011, Clarke et al., 2008, Zhai et al., 2003). We adapt both groups of metrics so they can be applied to contact recommendation and link prediction.

Recommendation perspective

Considering a recommendation perspective, the diversity of a set of suggested links studies how different the recommended people are to each other. A common way to measure this is the **intra-list dissimilarity (ILD)**, which is defined as the average pairwise distance between the people recommended to each target user:

$$\text{ILD}(\hat{\mathcal{G}}) = \frac{1}{|\hat{E}|} \sum_{(u,v)\in\hat{E}} \sum_{w\in\hat{\Gamma}_{out}(u)} \frac{d(v,w)}{|\hat{\Gamma}_{out}(u)|} \tag{8.20}$$

where $\hat{\mathcal{G}} = \langle \mathcal{U}, \hat{E} \rangle$ is a network containing only the set of recommended edges, $\hat{\Gamma}_{out}(v)$ is the people recommended to user u and $d(v,w)$ is a dissimilarity measure between users, similar to the one defined for the unexpectedness. Again, we find that distance functions that take user features are more informative than those based on network structure.

This is not the only definition of diversity in recommendation. We can take a global perspective, commonly referred as *diversity of sales* (Castells et al., 2015, Fleder and Hosanagar, 2007). We consider that a recommendation for a set of target users is diverse if all the candidates are evenly recommended, opposing to just recommending a few users to everyone. Again, the Gini index is a suitable metric to assess this aspect, so we define our metric (named as **predicted Gini complement (PGC)**) as:

$$\text{PGC}(\hat{\mathcal{G}}) = 1 - \frac{1}{|\mathcal{U}|-1} \sum_{i=1}^{|\mathcal{U}|} (2i - |\mathcal{U}| - 1) \frac{|\hat{\Gamma}_{in}(v_i)}{|\hat{E}|} \tag{8.21}$$

where v_i represents the i-th user in the network by a non-decreasing number of times $|\hat{\Gamma}_{in}(v_i)|$ he/she is recommended. When every user is recommended equally often, this metric is equal to 1, and, when all the recommendations point to the same user, it is equal to 0.

Information retrieval perspective

Information retrieval considers a different take on the diversity when we apply it to search tasks. This perspective considers that returning diverse results consists in covering the possible aspects or intents behind an ambiguous query (Chapelle et al., 2011, Clarke et al., 2008, Zhai et al., 2003). Contact recommendation and link prediction do not involve any explicit queries. However, we can adapt these metrics by matching the users in the network

to queries and documents (as we do in Part II). Here, we can consider, for example, latent communities as query aspects. Using this, we consider several metrics.

The first and simplest metric is the one called *subtopic recall* (Zhai et al., 2003), which counts and averages the ratio of aspects covered by our results. As we consider the communities of the network as aspects, we denote this metric as **community recall** in our context, which is defined as:

$$\text{CRecall}(\hat{\mathcal{G}}|\mathcal{C}) = \frac{1}{|\mathcal{U}||\mathcal{C}|}\sum_{u\in\mathcal{U}}\left|\bigcup_{v\in\Gamma_{out}(u)} c(v)\right| \tag{8.22}$$

where $\mathcal{C}$ is the set of communities, and $c(v)$ is the community user v belongs to.

Another (more elaborate) way to evaluate aspect-based diversity is known as the intent-aware scheme. Intent-aware metrics (Agrawal et al., 2009) adapt relevance metrics, such as expected reciprocal rank (Chapelle et al., 2009) or nDCG (Järvelin and Kekäläinen, 2002) to create new diversity metrics. Given an aspect, $z \in \mathcal{Z}$ these metrics consider different probabilities that a user is interested on them, $p(z|u)$. Then, this metric is computed as the expected value of the relevance metric over aspects (here, over communities):

$$\text{M-IA}(\hat{E}|\mathcal{G},\mathcal{C},E_{test}) = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\sum_{c\in\mathcal{C}} p(c|u)M_z(u,c) \tag{8.23}$$

One of the most widely used and substantial metrics on this perspective is the **intent-aware expected reciprocal rank (ERRIA)** first proposed by Chapelle et al. (2011) which is based on the expected reciprocal rank (ERR) accuracy metric (Chapelle et al., 2009). In our perspective, given a correctly predicted link, it weights down its importance when the community of the recommended candidate has already appeared above in the ranking:

$$\text{ERR-IA}(\hat{E}|\mathcal{G},\mathcal{C},E_{test}) = \frac{1}{|\mathcal{U}|}\sum_{u\in\mathcal{U}}\sum_{c\in\mathcal{C}} p(c|u)\text{ERR-IA}(u,c) \tag{8.24}$$

$$\text{ERR-IA}(u,c) = \sum_{k=1}^{|\hat{\Gamma}_{out}(u)|}\left[\frac{1}{k}\cdot p(\text{rel}|v_k,c)\prod_{j=1}^{k-1}(1-p(\text{rel}|v_j,c))\right] \tag{8.25}$$

where v_k is the candidate user at the k-th position in the recommendation ranking of links for user u, we can define $p(\text{rel}|v,c)$ as

$$p(\text{rel}|v,c) = \frac{1}{2}\cdot \mathbb{1}_{\{(u,v)\in E_{test}\wedge c(v)=c)\}}(u,v) \tag{8.26}$$

and E_{test} represents the set of test links we use to predict the accuracy of the algorithms. The probability $p(c|u)$ indicates the probability that a community is pertinent to a user, and it is estimated by the ratio of followees of u that belong to c:

$$p(c|u) = \frac{|\{v\in c|(u,v)\in E\cup E_{test}\}|}{\sum_{c'\in\mathcal{C}}|\{v\in c'|(u,v)\in E\cup E_{test}\}} \tag{8.27}$$

8.4 Experiments

In order to get a first observation of the different suggested metrics, we run a short experiment where we apply such metrics to a set of contact recommendation approaches. We explore two things: first, the values of each metric for every algorithm in the comparison, for providing some additional insights on how they work and what kind of people they do recomend; second, how the different metrics correlate to each other, to be aware of possibly redundant measures.

8.4.1 Setup

We consider for our experiments the five networks reported in Section 4.1: the two Twitter interaction networks, the two Twitter follows networks and the Facebook graph. Over such networks, we execute the different state of the art methods that we selected for the experiments in Section 4.3 along with BM25 as a representative of the IR models introduced in Chapter 5. The selection of algorithms is then:

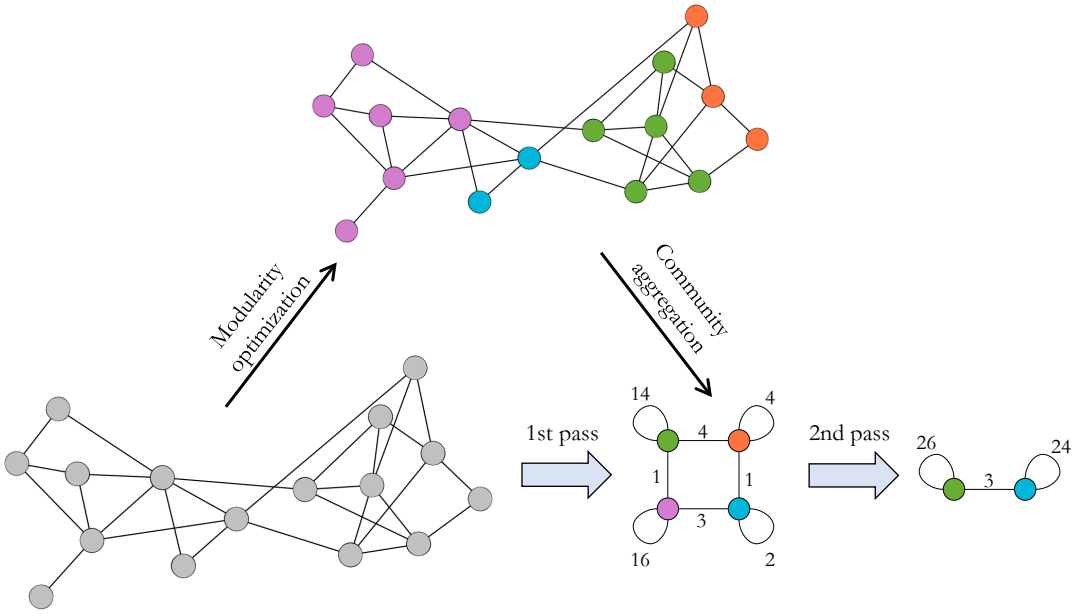


Figure 8.8: Louvain community detection algorithm example (adapted from Blondel et al. (2008))

- **Friends of friends:** most common neighbors (Liben-Nowell and Kleinberg, 2007), the Jaccard similarity (Jaccard, 1901, Liben-Nowell and Kleinberg, 2007), cosine similarity (Lü et al., 2009, Salton et al., 1975), Adamic-Adar (Adamic and Adar, 2003, Liben-Nowell and Kleinberg, 2007) and resource allocation (Zhou et al., 2009). We also include in this family the BM25 algorithm (Robertson and Zaragoza, 2009, Sanz-Cruzado and Castells, 2019a, Sanz-Cruzado et al., 2020a) as an IR representative.
- **Path-based:** We consider four different methods: the Katz algorithm (Katz, 1953, Liben-Nowell and Kleinberg, 2007), the local path index (Lü and Zhou, 2011, Lü et al., 2009), distance-based recommendation (Liben-Nowell and Kleinberg, 2007) and the global Leicht-Holme-Newman index (Leicht et al., 2006, Lü and Zhou, 2011).
- **Random walks:** As representative of the random walk family, we take the Money algorithm (Goel et al., 2015, Gupta et al., 2013), personalized (or rooted) PageRank (Liben-Nowell and Kleinberg, 2007, White and Smyth, 2003), the PropFlow algorithm (Lichtenwalter et al., 2010) and hitting and commute times (Liben-Nowell and Kleinberg, 2007).
- **Collaborative filtering:** We take three different approaches: the matrix factorization algorithm for implicit feedback proposed by Hu et al. (2008), user-based and item-based nearest-neighbor collaborative filtering recommenders with cosine similarity (Ning et al., 2015).
- **Content-based:** As context-based methods, we consider the Twittomender algorithm (Hannon et al., 2010) as well as the centroid-based method introduced in Section 3.4.7.

In addition to those algorithms, we include, as sanity-check baselines, random and popularity-based recommendation. As it has been done in previous chapters, we run the different algorithms over the input network, and, if we need relevance judgments, we take them from the test set. As hyperparameters for each algorithm we use the same as we take in the experiments in Chapters 4 and 5, which are reported in Appendix C.

For the structural diversity metrics, we select as the additional set of edges, $\hat{E}$, all the links from the target user to the recommended contacts at cutoff k, and, for each recommender, we generate its extended network as described at the beginning of Section 8.2. Although adding all the recommended links seems a quite strong assumption (we are considering them as if we have a perfect accuracy), we consider that evaluating such metrics this way allows us to have an idea of their potential effect, regardless of the which links are actually added to the network.

For the unexpectedness and ILD metrics, we take the tf-idf cosine similarity as the distance (or dissimilarity) function. For each user, we concatenate the text of the tweets they have published before the temporal

Table 8.1: Community-based statistics.

	Twitter 1-month		Twitter 200-tweets		Facebook
	Interactions	Follows	Interactions	Follows	
# Communities	8	10	6	10	17
Modularity	0.7073	0.5906	0.5837	0.5766	0.9781
MC	0.1464	0.2047	0.2081	0.2117	0.0109
ICEGC	0.1698	0.2558	0.3062	0.5965	0.0380
CEGC	0.0390	0.0873	0.1134	0.2096	0.0020

split point (similarly to how it is done for the Twittomender method (Hannon et al., 2010)). For the Facebook network, we do not have any side information that we can use for computing such metrics. Consequently, similarly to how we do not apply the content-based approaches for this network, we do not compute the results for ILD and unexpectedness. Finally, for the community-based metrics, we tested different community detection algorithms, but, as we find the results to be rather insensitive to the choice of algorithm (as already noted by De Meo et al. (2014)), we only report the results for one of the best known and most effective algorithms in the literature: the Louvain algorithm.

The Louvain algorithm (Blondel et al., 2008) is a fast agglomerative community detection algorithm which greedily maximizes the modularity of the network in two steps: the first step starts by assigning each node to a community, and moves nodes to other clusters by maximizing the modularity gain obtained by the change; when no further modularity improvement can be achieved by doing that, it builds a new network where the remaining communities are nodes, and links between nodes travel between the communities of the endpoints. The previous process is iteratively repeated until the modularity cannot be improved. An example of the algorithm is illustrated in Figure 8.8.

We show in Table 8.1 the main statistics of the community partition found by the Louvain algorithm after we apply it over the training networks of the different datasets. Figure 8.9 illustrates the size distribution of the communities, sorted by descending number of users.

8.4.2 Results

Metric correlations

We begin our analysis by observing the correlations between the different proposed metrics and the accuracy of the recommendation approaches. We do it by computing the pairwise Pearson correlation over the set of 20 recommendation algorithms (18 in the case of Facebook networks), which is reported in Figure 8.10. In that figure, each cell in the table represents the value of the coefficient, with red values showing negative correlation and blue values indicating positive correlation. The more intense the color, the greater is the absolute value of the Pearson coefficient. A first observation shows that greater nDCG values correspond with smaller values of most novelty and diversity metrics. The only exception to this is the ERRIA metric, which, in addition to the communities of the recommended users, considers the relevance of the links. This explains the relation between these metrics. Aside from ERRIA, there are two metrics which do not correlate negatively with nDCG: unexpectedness and ILD. In this case, the correlation between the metrics is close to zero, showing that the values for these metrics are independent of the accuracy of the recommender. These two metrics have no clear correlation to others, but they correlate with each other, showing that recommendations comprising a wide variety of users tend to differ (on average) from the contacts of the target users.

Distance-based and community-based metrics (particularly MC, CEGC and the community recall diversity metric) are highly correlated in the five tested datasets. This finding shows that these metrics capture related sides of similar notions: connecting communities shortens distances between many users – which is consistent with the theory of weak ties proposed by Granovetter (1973). Those community metrics have similar behaviour to the average edge weakness of the recommendation, thus showing that the definition of the metrics is consistent, as, given the definition of community structure, non-redundant ties are more likely to travel between different clusters of people. Interestingly, the remaining community metric, ICEGC, does not correlate with the modularity complement (and it even shows negative correlation in the 200-tweets interaction network),

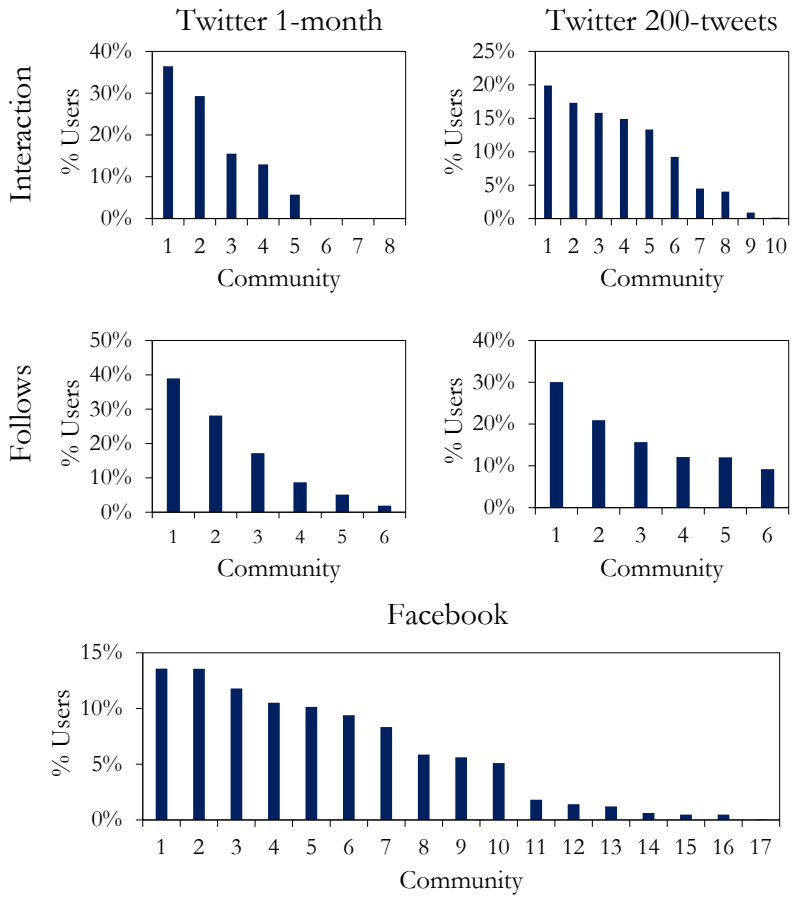


Figure 8.9: Community sizes of the Louvain partition for the different networks.

illustrating that having more connections between different communities does not necessarily mean that they are similarly distributed among the different community pairs. Notably, the CEGC metric does have positive correlations with both metrics, proving that this metric is able to capture both the presence of a larger number of inter-community links in the network (as it correlates with MC) and the balance of their distribution (as it correlates with ICEGC).

The long tail novelty is yet another metric showing relevant correlation to others. Besides its relation to accuracy metrics, LTN also displays a negative correlation with the distance-based metrics and MC: this likely occurs because connecting to low-degree users (with high LTN value) does not reduce distances as much as creating links towards hubs, and such connections towards low-degree users do not escape the communities of the target users. This opposite trend can also be observed for AEW, since, when a user in the long tail of the distribution is recommended, it just contributes a smaller neighborhood side in the denominator for the embeddedness. In directed networks, LTN is positively correlated to the ICEGC metric, showing that, although the creation of links towards low in-degree users does not lead to the creation of many edges traversing communities, it balances the weak tie distribution. This is no longer true in undirected neighbors such as Facebook. The reason is the following: in a directed network, when we create links towards popular users, links travel from any community in the network towards the ones where the popular users live, but not in the opposite direction, producing inequalities in the distribution. In undirected networks, this does not occur, since connections are bidirectional.

Finally, other metrics which are strongly correlated are IDGC and PGC, showing that – at least on the studied networks – there is no big difference between computing the Gini coefficient over the final, extended network and restricting it over the set of recommended links. They are also correlated to the LTN metric. This seems reasonable for the IDGC metric: recommending those least popular users tends to balance the degree distribution.

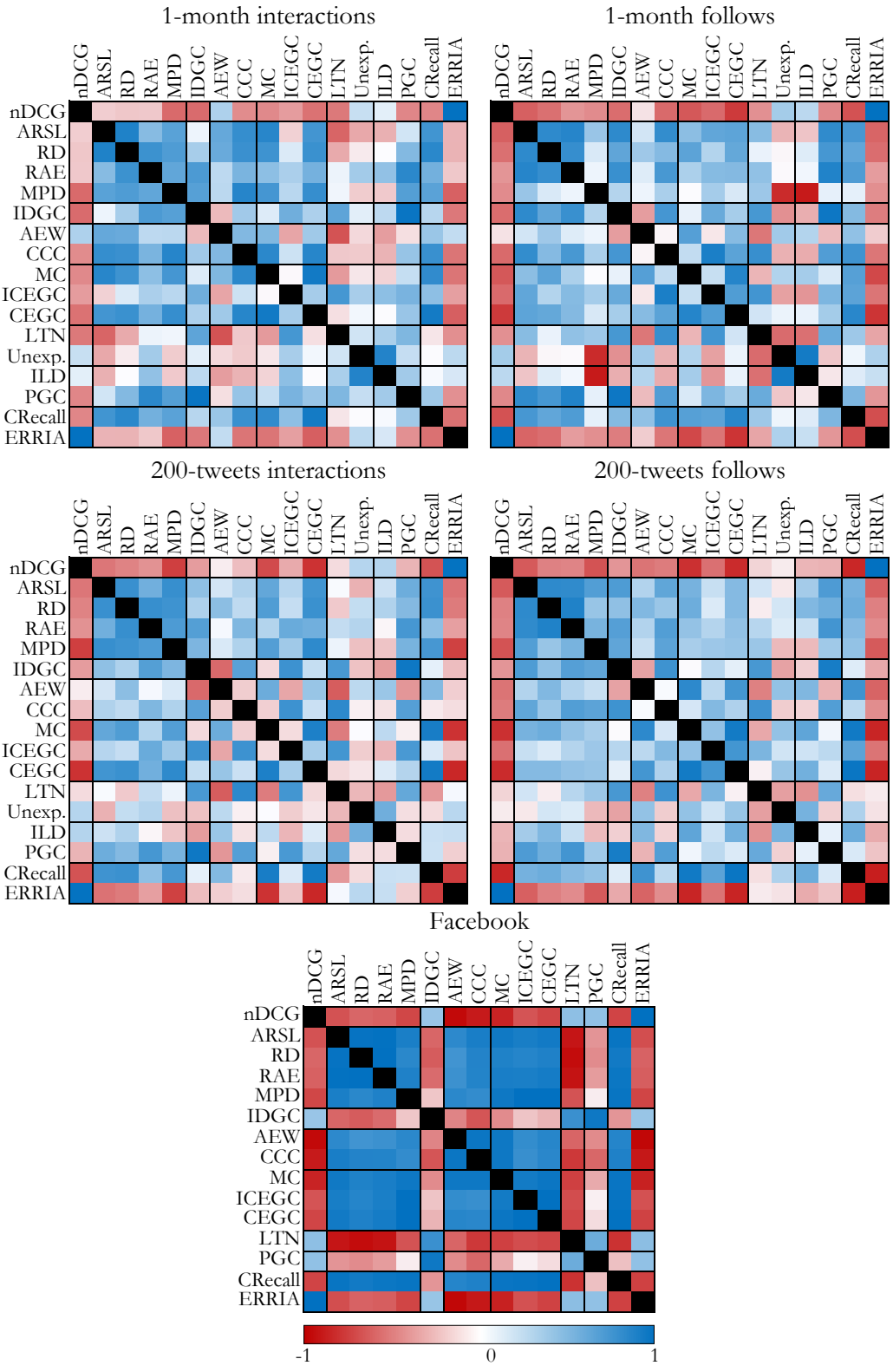


Figure 8.10: Pairwise Pearson correlation between metrics (@10 cutoff).

Algorithmic comparison

Now, we analyze the metric results for the different contact recommendation methods in our comparison. Tables 8.2 and 8.3 show the results for the 1-month dataset, Tables 8.4 and 8.5 do the same for the 200-tweets one, and Facebook results are shown in Table 8.6. In all tables, for the topological metrics, blue cells represent higher structural diversity than the training network, and red cells represent lower structural diversity (the training

Table 8.2: Beyond accuracy metrics for the 1-month interactions network at cutoff 10. For the structural metrics (excluding), the first row shows the value of the metric in the training network. For such metrics, red values indicate lower values than the training network one and blue values indicate higher values. For the rest, lower metric values appear in red whereas higher values appear in blue. The top value for each metric is highlighted in bold and underlined.

			Distance				Structural diversity						Novelty		Diversity			
	Algorithm	nDCG	ARSL	RD	RAE	MPD	IDGC	AEW	CCC	MC	ICEGC	CEGC	LTN	Unexp.	ILD	Gini	CRecall	ERRIA
	Input network	-	0.256	0.077	0.150	-	0.161	0.976	0.944	0.146	0.170	0.039	-	-	-	-	-	-
CF	iMF	**0.139**	0.296	0.111	0.149	0.377	0.117	0.980	0.902	0.155	0.179	0.045	0.957	0.841	0.748	0.029	0.238	0.103
CF	UB kNN	0.137	0.293	0.111	0.165	0.232	0.120	0.982	0.904	0.154	0.169	0.042	0.940	**0.901**	0.782	0.021	0.199	**0.104**
CF	IB kNN	0.117	0.280	0.100	0.163	0.221	0.135	0.976	0.891	0.147	0.159	0.038	0.945	0.883	0.744	0.033	0.150	0.094
FOAF	BM25	0.104	0.298	0.125	0.161	0.184	0.129	0.972	0.878	0.150	0.167	0.041	0.956	0.836	0.766	0.046	0.229	0.082
FOAF	Adamic-Adar	0.098	0.301	0.125	0.168	0.192	0.142	0.965	0.882	0.149	0.167	0.041	0.959	0.847	0.779	0.070	0.225	0.079
FOAF	MCN	0.092	0.298	0.111	0.157	0.188	0.145	0.962	0.879	0.145	0.167	0.040	0.963	0.841	0.772	0.067	0.211	0.075
FOAF	Resource Allocation	0.088	0.306	0.125	0.171	0.211	0.153	0.962	0.890	0.158	0.173	0.045	0.966	0.849	0.788	0.104	0.253	0.070
FOAF	Cosine	0.039	0.289	0.111	0.165	0.216	0.178	0.944	0.897	0.152	0.181	0.045	0.994	0.865	0.815	0.117	0.236	0.029
FOAF	Jaccard	0.032	0.286	0.100	0.163	0.222	0.219	0.930	0.891	0.146	0.179	0.042	0.996	0.854	0.790	0.206	0.225	0.028
Random walk	Money	0.132	0.319	0.125	0.161	0.388	0.112	0.985	0.904	0.166	0.169	0.045	0.928	0.834	0.729	0.011	0.238	0.097
Random walk	Pers. PageRank	0.100	0.318	0.143	0.166	0.209	0.123	0.979	0.915	0.182	0.181	0.054	0.949	0.844	0.774	0.029	0.311	0.068
Random walk	PropFlow	0.097	0.302	0.125	0.156	0.233	0.127	0.976	0.905	0.175	0.187	0.054	0.963	0.850	0.761	0.035	0.305	0.070
Random walk	Hitting time	0.051	**0.372**	0.167	0.167	0.902	0.104	**0.997**	0.934	0.290	0.170	0.083	0.907	0.818	0.703	0.001	0.490	0.026
Random walk	Commute time	0.051	0.372	0.167	0.167	0.902	0.104	0.997	0.934	0.290	0.170	0.083	0.907	0.818	0.703	0.001	0.490	0.026
Path-based	Local Path Index	0.116	0.313	0.125	0.155	0.423	0.110	0.988	0.898	0.168	0.144	0.040	0.920	0.829	0.736	0.006	0.160	0.090
Path-based	Katz	0.022	0.296	0.111	0.151	0.706	0.116	0.985	0.910	0.261	0.180	0.078	0.973	0.876	0.752	0.014	0.331	0.017
Path-based	Distance	0.015	0.294	0.125	0.155	0.163	0.131	0.982	0.903	0.189	0.179	0.055	0.991	0.843	0.742	0.019	0.361	0.010
Path-based	Global LHN Index	0.003	0.299	0.100	0.135	0.947	0.220	0.959	0.916	0.145	0.170	0.040	**1.000**	0.732	0.560	0.084	0.179	0.002
CB	Centroid CB	0.032	0.285	0.100	0.147	0.288	0.191	0.916	0.885	0.154	0.177	0.044	0.990	0.858	0.804	0.168	0.202	0.026
CB	Twittomender	0.002	0.310	0.125	0.172	1.083	0.230	0.925	0.919	0.238	0.198	0.075	0.998	0.870	**0.815**	0.197	0.350	0.001
	Popularity	0.057	0.357	0.143	0.167	0.755	0.104	0.997	0.924	**0.295**	0.124	0.061	0.883	0.808	0.684	0.001	0.250	0.039
	Random	0.001	0.353	**0.200**	**0.237**	**1.922**	**0.439**	0.995	**0.952**	0.280	**0.201**	**0.091**	0.998	0.873	0.812	**0.824**	**0.497**	0.001

Table 8.3: Beyond accuracy metrics for the 1-month follows network at cutoff 10. For the structural metrics (excluding MPD), the first row shows the value of the metric in the training network. For such metrics, red values indicate lower values than the training network one and blue values indicate higher values. For the rest, lower metric values appear in red whereas higher values appear in blue. The top value for each metric is highlighted in bold and underlined.

			Distance				Structural diversity						Novelty		Diversity			
	Algorithm	nDCG	ARSL	RD	RAE	MPD	IDGC	AEW	CCC	MC	ICGC	CEGC	LTN	Unexp.	ILD	PGC	Crecall	ERRIA
	Input network	-	0.288	0.100	0.164	-	0.163	0.947	0.775	0.205	0.256	0.087	-	-	-	-	-	-
CF	iMF	**0.146**	0.297	0.125	0.165	0.077	0.149	0.961	0.751	0.200	0.260	0.087	0.886	0.912	0.822	0.042	0.282	**0.102**
CF	UB kNN	0.141	0.297	0.111	0.146	0.035	0.148	0.963	0.740	0.200	0.258	0.086	0.863	0.938	0.835	0.031	0.260	0.100
CF	IB kNN	0.130	0.297	0.125	0.168	0.026	0.149	0.960	0.744	0.199	0.256	0.085	0.860	0.935	0.825	0.028	0.211	0.095
FOAF	BM25	0.118	0.294	0.111	0.166	0.000	0.148	0.967	0.735	0.206	0.253	0.087	0.862	0.939	0.834	0.026	0.288	0.084
FOAF	Adamic-Adar	0.118	0.295	0.111	0.166	0.000	0.148	0.965	0.739	0.206	0.253	0.087	0.865	0.937	0.830	0.030	0.287	0.085
FOAF	MCN	0.119	0.303	0.125	0.162	0.061	0.147	0.970	0.733	0.206	0.254	0.087	0.853	0.917	0.832	0.025	0.287	0.085
FOAF	Resource Allocation	0.104	0.298	0.111	0.179	0.000	0.155	0.958	0.748	0.207	0.258	0.090	0.894	0.936	0.828	0.067	0.308	0.074
FOAF	Cosine	0.050	0.303	0.125	0.170	0.095	0.217	0.916	0.748	0.201	0.271	0.090	0.987	0.902	0.804	0.297	0.303	0.040
FOAF	Jaccard	0.054	0.302	0.125	0.169	0.087	0.202	0.918	0.748	0.199	0.262	0.088	0.986	0.908	0.816	0.279	0.274	0.044
Random walk	Money	0.110	0.297	0.125	0.164	0.079	0.144	0.978	0.731	0.212	0.253	0.089	0.802	0.922	0.832	0.009	0.299	0.075
Random walk	Pers. PageRank	0.097	0.301	0.125	0.161	0.054	0.148	0.972	0.739	0.212	0.261	0.093	0.869	0.922	0.843	0.020	0.363	0.060
Random walk	PropFlow	0.093	0.295	0.111	0.165	0.054	0.151	0.968	0.742	0.211	0.262	0.093	0.890	0.919	0.829	0.032	0.368	0.058
Random walk	Hitting time	0.046	0.302	0.143	0.160	0.171	0.142	0.991	0.723	0.248	0.246	0.101	0.736	**0.941**	0.861	0.002	0.333	0.025
Random walk	Commute time	0.046	0.302	0.143	0.160	0.171	0.142	0.991	0.723	0.248	0.246	0.101	0.736	0.941	**0.861**	0.002	0.333	0.025
Path-based	Local Path Index	0.109	0.303	0.125	0.162	0.056	0.145	0.977	0.741	0.214	0.256	0.091	0.830	0.917	0.815	0.010	0.299	0.076
Path-based	Katz	0.009	0.309	0.125	0.166	0.414	0.150	0.988	0.783	0.248	**0.307**	**0.126**	0.949	0.935	0.822	0.015	0.489	0.005
Path-based	Distance	0.007	0.299	0.125	0.167	0.053	0.151	0.983	0.760	0.231	0.260	0.101	0.964	0.934	0.855	0.006	0.490	0.003
Path-based	Global LHN Index	0.002	0.323	0.125	0.158	**8.685**	0.184	0.969	0.761	0.207	0.277	0.095	**1.000**	0.842	0.682	0.030	0.301	0.001
CB	Centroid CB	0.047	0.301	0.125	0.164	0.176	0.198	0.917	0.748	0.204	0.267	0.090	0.966	0.903	0.809	0.322	0.240	0.038
CB	Twittomender	0.001	0.354	0.143	0.193	1.507	0.233	0.949	0.792	0.238	0.286	0.112	0.994	0.913	0.819	0.377	0.492	0.001
	Popularity	0.045	0.304	0.143	0.160	0.175	0.142	0.992	0.727	**0.251**	0.247	0.103	0.727	0.937	0.853	0.002	0.299	0.023
	Random	0.001	**0.376**	**0.200**	**0.246**	1.538	**0.270**	**0.997**	**0.800**	0.245	0.287	0.116	0.994	0.922	0.834	**0.822**	**0.591**	0.001

Table 8.4: Beyond accuracy metrics for the 200-tweets interaction network at cutoff 10. For the structural metrics (excluding MPD), the first row shows the value of the metric in the training network. For such metrics, red values indicate lower values than the training network one and blue values indicate higher values. For the rest, lower metric values appear in red whereas higher values appear in blue. The top value for each metric is highlighted in bold and underlined.

		Distance					Structural diversity						Novelty		Diversity			
	Algorithm	nDCG	ARSL	RD	RAE	MPD	IDGC	AEW	CCC	MC	ICGC	CEGC	LTN	Unexp	ILD	PGC	Crecall	ERRIA
	Input network	-	0.202	0.053	0.157	-	0.309	0.972	0.906	0.208	0.306	0.113	-	-	-	-	-	-
CF	iMF	0.103	0.264	0.125	0.156	0.662	0.225	0.957	0.828	0.208	0.323	0.124	0.988	0.843	0.848	0.092	0.240	0.056
CF	User-based CF	0.095	0.239	0.125	0.174	0.395	0.227	0.964	0.815	0.211	0.318	0.120	0.974	**0.941**	0.838	0.071	0.225	0.054
CF	Item-based kNN	0.072	0.240	0.125	0.184	0.498	0.360	0.933	0.826	0.199	0.334	0.115	0.990	0.927	0.812	0.223	0.194	0.046
FOAF	BM25	**0.110**	0.259	0.111	0.156	0.494	0.270	0.950	0.795	0.206	0.322	0.118	0.983	0.839	0.838	0.167	0.271	**0.063**
FOAF	Adamic-Adar	0.100	0.263	0.111	0.155	0.511	0.291	0.940	0.803	0.207	0.319	0.118	0.983	0.841	0.839	0.212	0.270	0.057
FOAF	MCN	0.095	0.261	0.111	0.157	0.559	0.315	0.924	0.798	0.202	0.316	0.114	0.986	0.839	0.834	0.227	0.255	0.055
FOAF	Resource Allocation	0.091	0.268	0.125	0.161	0.551	0.312	0.941	0.817	0.218	0.328	0.128	0.986	0.842	0.842	0.266	0.297	0.053
FOAF	Cosine	0.048	0.274	0.125	0.167	0.671	0.420	0.889	0.829	0.219	0.327	0.128	0.998	0.833	0.821	0.382	0.285	0.029
FOAF	Jaccard	0.057	0.265	0.111	0.163	0.744	0.456	0.870	0.808	0.205	0.328	0.120	0.998	0.832	0.818	0.439	0.254	0.035
Random walk	Money	0.093	0.266	0.125	0.155	0.672	0.223	0.970	0.801	0.234	0.320	0.134	0.972	0.840	0.845	0.066	0.296	0.051
Random walk	Pers. PageRank	0.063	0.259	0.111	0.150	0.348	0.243	0.966	0.837	0.256	0.338	0.154	0.980	0.845	0.852	0.100	0.349	0.032
Random walk	PropFlow	0.063	0.254	0.111	0.149	0.443	0.251	0.960	0.824	0.248	0.342	0.151	0.987	0.844	0.849	0.111	0.314	0.034
Random walk	Hitting time	0.041	0.273	0.143	0.167	1.238	0.180	0.998	0.700	**0.337**	0.302	0.182	0.943	0.859	0.856	0.001	0.492	0.014
Random walk	Commute time	0.041	0.273	0.143	0.167	1.238	0.180	0.998	0.700	**0.337**	0.302	0.182	0.943	0.859	0.856	0.001	0.492	0.014
Path-based	LPI	0.089	0.269	0.125	0.157	0.691	0.222	0.967	0.793	0.223	0.282	0.113	0.966	0.842	0.833	0.047	0.260	0.050
Path-based	Katz	0.017	0.271	0.125	0.156	1.527	0.194	0.988	0.846	0.324	0.291	0.170	0.986	0.857	0.843	0.010	0.321	0.008
Path-based	Distance	0.024	0.262	0.125	0.154	0.324	0.248	0.974	0.861	0.260	0.335	0.155	0.994	0.847	**0.859**	0.061	0.350	0.013
Path-based	Global LHN	0.012	0.253	0.100	0.157	1.515	0.376	0.925	0.815	0.225	0.349	0.141	**1.000**	0.752	0.676	0.132	0.296	0.007
CB	Centroid CB	0.064	0.254	0.111	0.150	0.781	0.355	0.869	0.797	0.218	0.325	0.126	0.991	0.836	0.823	0.249	0.261	0.039
CB	Twittomender	0.002	0.288	0.143	0.187	2.137	0.467	0.876	0.858	0.301	0.369	0.197	0.999	0.838	0.821	0.395	0.430	0.001
	Popularity	0.042	0.267	0.143	0.149	1.165	0.180	0.998	0.701	**0.337**	0.257	0.156	0.939	0.855	0.838	0.001	0.426	0.015
	Random	0.001	**0.310**	**0.200**	**0.217**	**2.973**	**0.579**	**0.999**	**0.957**	0.332	**0.387**	**0.227**	0.998	0.850	0.842	**0.823**	**0.554**	0.000

Table 8.5: Beyond accuracy metrics for the 200-tweets follows network at cutoff 10. For the structural metrics (excluding MPD), the first row shows the value of the metric in the training network. For such metrics, red values indicate lower values than the training network one and blue values indicate higher values. For the rest, lower metric values appear in red whereas higher values appear in blue. The top value for each metric is highlighted in bold and underlined.

			Distance				Structural diversity						Novelty		Diversity			
	Algorithm	nDCG	ARSL	RD	RAE	MPD	IDGC	AEW	CCC	MC	ICGC	CEGC	LTN	Unexp.	ILD	PGC	Crecall	ERRIA
	Input network	-	0.293	0.100	0.164	-	0.288	0.944	0.822	0.212	0.597	0.210	-	-	-	-	-	-
CF	iMF	**0.133**	0.304	0.125	0.164	0.139	0.265	0.937	0.793	0.205	0.614	0.211	0.966	0.903	0.822	0.120	0.351	**0.093**
CF	UB kNN	0.127	0.304	0.125	0.168	0.048	0.256	0.943	0.782	0.202	0.601	0.203	0.949	**0.941**	0.833	0.064	0.279	0.092
CF	IB kNN	0.121	0.304	0.111	0.170	0.057	0.272	0.927	0.783	0.201	0.600	0.201	0.956	0.938	0.827	0.102	0.258	0.089
FOAF	BM25	0.116	0.313	0.125	0.159	0.117	0.260	0.947	0.769	0.206	0.601	0.207	0.948	0.907	0.840	0.084	0.356	0.081
FOAF	Adamic-Adar	0.114	0.318	0.125	0.161	0.122	0.260	0.947	0.780	0.206	0.600	0.206	0.946	0.907	0.839	0.084	0.348	0.082
FOAF	MCN	0.111	0.317	0.125	0.160	0.123	0.258	0.950	0.779	0.207	0.600	0.207	0.945	0.907	0.839	0.076	0.344	0.080
FOAF	Resource Allocation	0.112	0.318	0.125	0.162	0.128	0.274	0.940	0.782	0.207	0.611	0.211	0.957	0.906	0.840	0.149	0.386	0.079
FOAF	Cosine	0.077	0.317	0.125	0.173	0.153	0.369	0.893	0.789	0.204	0.612	0.209	0.992	0.900	0.829	0.456	0.370	0.059
FOAF	Jaccard	0.085	0.315	0.125	0.166	0.157	0.343	0.895	0.785	0.199	0.613	0.204	0.991	0.903	0.833	0.440	0.332	0.064
Random walk	Money	0.113	0.306	0.111	0.165	0.138	0.252	0.961	0.778	0.215	0.596	0.214	0.926	0.906	0.821	0.032	0.405	0.078
Random walk	Pers. PageRank	0.084	0.316	0.125	0.161	0.096	0.267	0.953	0.798	0.221	0.633	0.233	0.962	0.911	0.848	0.088	0.474	0.056
Random walk	PropFlow	0.082	0.303	0.111	0.165	0.099	0.271	0.950	0.796	0.220	0.638	0.234	0.969	0.908	0.825	0.100	0.473	0.055
Random walk	Hitting time	0.036	0.322	0.143	0.167	0.317	0.239	0.994	0.753	**0.265**	0.628	0.277	0.889	0.929	**0.862**	0.001	**0.801**	0.016
Random walk	Commute time	0.036	0.322	0.143	0.167	0.317	0.239	0.994	0.753	**0.265**	0.628	0.277	0.889	0.929	**0.862**	0.001	**0.801**	0.016
Path-based	Local Path Index	0.093	0.317	0.125	0.161	0.102	0.248	0.966	0.787	0.219	0.582	0.213	0.927	0.907	0.836	0.022	0.329	0.067
Path-based	Katz	0.013	0.306	0.111	0.156	0.686	0.245	0.994	0.797	0.264	0.717	0.318	0.978	0.936	0.841	0.004	0.784	0.004
Path-based	Distance	0.014	0.309	0.125	0.167	0.096	0.261	0.982	0.814	0.237	0.634	0.251	0.986	0.917	0.842	0.016	0.629	0.010
Path-based	Global LHN Index	0.008	0.332	0.125	0.173	**2.690**	0.355	0.951	0.820	0.217	0.627	0.227	**1.000**	0.886	0.802	0.105	0.489	0.005
CB	Centroid CB	0.074	0.314	0.100	0.141	0.210	0.309	0.913	0.796	0.208	0.612	0.212	0.980	0.904	0.835	0.300	0.323	0.056
CB	Twittomender	0.002	0.356	0.143	0.190	1.440	0.385	0.949	0.845	0.258	0.653	0.280	0.995	0.919	0.854	0.516	0.721	0.001
	Popularity	0.040	0.326	0.143	0.167	0.315	0.239	0.993	0.770	0.265	0.549	0.242	0.869	0.923	0.843	0.001	0.526	0.019
	Random	0.002	**0.366**	**0.200**	**0.247**	1.455	**0.406**	**0.997**	**0.854**	0.262	0.658	0.286	0.995	0.922	0.855	**0.822**	0.794	0.001

Table 8.6: Beyond accuracy metrics for the Facebook network at cutoff 10. For the structural metrics (excluding MPD), the first row shows the value of the metric in the training network. For such metrics, red values indicate lower values than the training network one and blue values indicate higher values. For the rest, lower metric values appear in red whereas higher values appear in blue. The top value for each metric is highlighted in bold and underlined.

			Distance				Structural diversity						Novelty	Diversity		
	Algorithm	nDCG	ARSL	RD	RAE	MPD	IDGC	AEW	CCC	MC	ICGC	CEGC	LTN	PGC	Crecall	ERRIA
	Input network	-	0.283	0.091	0.126	-	0.460	0.448	0.584	0.011	0.038	0.002	-	-	-	-
CF	iMF	0.425	0.312	0.143	0.179	0.032	0.536	0.711	0.539	0.012	0.059	0.003	0.981	0.159	0.070	0.346
CF	UB kNN	0.514	0.313	0.143	0.174	0.019	0.553	0.670	0.523	0.011	0.055	0.003	0.982	0.249	0.070	0.396
CF	IB kNN	0.454	0.293	0.125	0.153	0.024	0.564	0.649	0.501	0.011	0.046	0.002	0.986	0.286	0.062	0.367
FOAF	BM25	0.573	0.309	0.125	0.158	0.000	0.567	0.636	0.505	0.011	0.054	0.003	0.983	0.398	0.071	0.428
FOAF	Adamic-Adar	0.575	0.309	0.125	0.158	0.000	0.566	0.648	0.507	0.011	0.054	0.003	0.983	0.384	0.072	0.428
FOAF	MCN	0.559	0.308	0.125	0.158	0.000	0.568	0.649	0.506	0.011	0.053	0.003	0.983	0.377	0.071	0.419
FOAF	Resource Allocation	**0.592**	0.309	0.125	0.158	0.000	0.569	0.644	0.505	0.012	0.056	0.003	0.984	0.425	0.074	**0.439**
FOAF	Cosine	0.494	0.291	0.125	0.154	0.000	0.587	0.634	0.497	0.011	0.048	0.003	0.990	0.505	0.064	0.389
FOAF	Jaccard	0.491	0.291	0.125	0.154	0.000	0.596	0.618	0.483	0.011	0.051	0.003	0.990	0.583	0.064	0.386
Random walk	Money	0.587	0.317	0.143	0.180	0.005	0.557	0.653	0.518	0.012	0.058	0.003	0.982	0.347	0.073	0.434
Random walk	Pers. PageRank	0.589	0.324	0.167	0.212	0.005	0.557	0.648	0.516	0.012	0.059	0.003	0.982	0.338	0.073	0.436
Random walk	PropFlow	0.570	0.314	0.167	0.182	0.007	0.554	0.652	0.517	0.012	0.058	0.003	0.981	0.295	0.072	0.425
Random walk	Hitting time	0.067	**0.507**	**0.500**	0.501	1.192	0.475	0.986	**0.945**	0.103	0.268	0.104	0.923	0.002	0.422	0.051
Random walk	Commute time	0.074	0.507	**0.500**	0.501	1.167	0.475	0.985	0.944	0.102	0.267	0.103	0.923	0.002	**0.422**	0.058
Path-based	Local Path Index	0.457	0.312	0.143	0.170	0.010	0.526	0.731	0.590	0.017	0.043	0.003	0.976	0.141	0.071	0.357
Path-based	Katz	0.053	0.344	0.125	0.165	0.278	0.521	0.892	0.739	0.057	0.063	0.016	0.982	0.076	0.104	0.052
Path-based	Distance	0.046	0.328	0.143	0.163	0.004	0.512	0.900	0.745	0.043	0.056	0.011	0.990	0.029	0.132	0.036
Path-based	Global LHN Index	0.089	0.293	0.125	0.154	0.005	0.558	0.854	0.709	0.029	0.033	0.005	0.998	0.119	0.064	0.063
	Popularity	0.052	0.507	**0.500**	**0.501**	1.080	0.475	0.985	0.944	**0.106**	0.243	0.096	0.913	0.002	0.411	0.032
	Random	0.003	0.411	0.250	0.324	**1.608**	**0.654**	**0.993**	0.732	0.099	**0.331**	**0.121**	0.991	**0.820**	0.392	0.004

network value is displayed in the first row of the table). For the rest of metrics (where we do not have a base value), the blue cells just represent high metric results, and the red ones low values.

We start the algorithmic comparison by noticing that, for the vast majority of the metrics, the most inaccurate algorithm, random recommendation, trivially reaches some of the highest novelty and diversity metrics. The only notable exceptions to this are ERRIA (which, as stated before, considers the relevance the recommended contacts have for the target user) and, to a lesser extent, ILD and unexpectedness – for which random recommendation seems to be far from optimal on the 1-month follows and 200-tweets interaction networks). Therefore, for the rest of metrics, we shall only comment this approach in the case some algorithm manages to consistently beat it.

For the unexpectedness, user and item-based kNN approaches reach the highest values in all datasets, whereas the global Leicht-Holme-Newman index provides the worst values in our comparison. The global LHN is also the worst method in terms of ILD, whereas hitting and commute time provide the most diverse recommendation lists. Although we would expect the content-based approaches to be the worst for the two metrics, this does not happen in our experiments due to the optimal configuration: we are not using the tweets of the target and candidate users, but those of their followers. Nonetheless, content-based algorithms are still far from optimal.

In terms of the long tail novelty, unsurprisingly, popularity-based recommendation establishes the bottom reference in all the dataset, closely followed by commute and hitting time. To a lesser extent, the Money algorithm also gets suboptimal results for this metric – something reasonable, since the random walk algorithm this approach is based on, SALSA, generates scores for each node which are proportional to their in-degree (Lempel and Moran, 2001) in the case of the authorities (which is the configuration for all Twitter networks). At the other extreme, we find three algorithms which apply a heavy penalization to popular users: the Jaccard and cosine similarities and the global LHN index. Both content-based methods also tend to recommend unpopular users. For the top algorithms in our accuracy comparison, we find that iMF shows reasonably good values for this metric.

Following that LTN is strongly correlated to the complementary Gini indexes over the in-degree distribution (IDGC) and the recommended nodes distribution (PGC), we can make similar observations about both metrics than the ones above. When compared to the original degree distribution, most of the methods tend to make the distribution skewer, following, to some extent, the preferential attachment principle (Barabási and Albert, 1999). The only exceptions to this are the approaches with high LTN. However, although there are few differences between IDGC and PGC, one of such algorithms shows an exception: the global LHN index manages to achieve good IDGC values, since it recommend users with few contacts. However, it is so focused on this that it concentrates the links in the same part of the network (leading to suboptimal PGC values). Another interesting observation about the PGC metric is that hitting and commute time act as not personalized algorithms (they have similar PGC values to those of popularity-based recommendation). This might be due to the fact that both take a not personalized PageRank as the underlying random walk.

Observing the distance-based metrics, we notice that all algorithms manage to increase the ARSL and reciprocal diameters values to some extent: as there are now more links in the network, and the number of users remains the same, it is natural to observe that distances between nodes are reduced. The observation we make for these two metrics is not necessarily true for the RAE metric, but this has a simple explanation: RAE does not consider distances to unreachable nodes. If the suggested links connect two previously disconnected users, it is possible that the maximum distance between a node and any other reachable one in the network increases. It this occurs, it is possible for the RAE metric to achieve lower values.

The Twittomender algorithm is the approach that, beyond random recommendation, manages to reduce distances between users the most in Twitter networks. Beyond this method, it is not easy to establish a clear order on which algorithms are better or worse. In the Facebook network, we observe something different: the not personalized approaches (popularity-based and, according to PGC, hitting and commute time) manage to reduce distances the most – even more than random recommendation. This is caused by the specific properties of undirected networks: in such networks, if we manage to create a link from all the users in the network towards a single node, the maximum distance between all nodes is equal to 2, explaining the results. In directed networks, this is not true, since we would still need a link between such node and the rest of people in the network.

We also notice that the different recommendation algorithms suggest people close to the target user. Popularity-based recommendation, Katz and content-based approaches manage to reach farther nodes than the average recommender. Among the algorithms that recommend close people, we notice all the friends of friends methods, distance-based recommendation, personalized PageRank and PropFlow – all of which make sense because of their design. Surprisingly, in some of the datasets – notably on the 200-tweets dataset – there are two approaches which manage to recommend people at even closer distance than the previous ones: user-based and item-based kNN.

As stated in Section 7.2, kNN variants are able to recommend people at undirected distance 3, whereas BM25 or Adamic-Adar cannot reach distance farther than 2. This can be, indeed, observed for the Facebook dataset, where all the friends of friends approaches reach a MPD value equal to 0, and both nearest-neighbors methods recommend some vertices at distance 3. However, since we measure the directed distance for the Twitter networks, the maximum distance between the target and the candidate users for friends of friends approaches depends on the orientation selection we discuss in 5.2. For example, if we took the incoming neighbors of the target and candidate users for BM25, we could recommend a person who needs two hops to reach the target user, but no path might exist between such target user and her (which would lead to recommending a user at infinite distance from the target user, despite being at two steps if the graph were undirected). As we use the undirected neighbor of the target user for most of our friends of friends algorithms, BM25 or Adamic-Adar are able to recommend farther people than UB kNN or IB kNN in some of the networks.

For the average edge weakness, random walk approaches and Katz manage to increase the average value of the network in all the networks. On the other hand, as expected, the Jaccard similarity achieves very low values for this metric, since it uses the embeddedness of the edges as ranking function. Cosine similarity and content-based approaches also tend to recommend strong ties under this definition. In terms of CCC, most methods increase the number of closed triads in the network: similarly to how contact recommendation algorithms follow the preferential attachment phenomenon, they do the same with triadic closure (Newman, 2018). Random recommendation is the only approach that consistently improves this value.

Finally, for the community metrics, we find that popularity-based recommendation manages to achieve the highest values for the modularity complement: it generates multiple links towards other communities. However, in the directed Twitter networks, this increases the imbalance of edges between communities, leading to bad ICEGC values. As we state earlier, due to the nature of the edges, this does not happen in the Facebook network, where popularity-based recommendation also manages to achieve good values for the ICEGC and CEGC metrics (although lower than the random recommendation approach, which uniformly spreads links between the different communities). Friends of friends methods achieve bad modularity complement values, since they tend to recommend people inside communities whereas random walks provide good results for all three metrics.

Focusing our attention on the top accuracy recommendation approaches, we observe that, in general, diversity results are far from the best values. Having that in mind, for Twitter, the implicit matrix factorization algorithm seems to provide the best trade-off between accuracy and diversity: among the top baselines, it is the method with greater long tail novelty, that reaches farther nodes in the network, and a more balanced edge distribution between communities. In addition, user-based kNN manages to optimize the unexpectedness of the recommendations (leading users to more surprising contents) and the diversity of the ranking list (ILD). In Facebook, the personalized PageRank algorithm provides better diversity results than the resource allocation algorithm, except for the Gini distribution of the degree.

8.5 Conclusions

The accuracy of the recommendation, although important, represents a partial perspective for the evaluation of contact recommendation, given the role that social networks are acquiring as a service, a communication platform and a business. Therefore, it seems natural to consider other perspectives as the target of contact recommendation methods. In this chapter, we define some of this new dimensions and provide a brief exploration of such perspectives by comparing several contact recommendation approaches. We summarize the main findings of this chapter here:

- We can measure several properties beyond accuracy in contact recommendation. In addition to the classical novelty and diversity of the recommendations, we add the potential effect the recommendations have on the topological properties of the network, such as the distance between users or the presence of links between communities.
- In general, the accuracy of the recommendation approaches goes against most of the diversity properties we consider. The only exception to this are the intent-aware metrics from IR, since they also take into account the concept of relevance.
- Distance-based and community-based metrics are highly correlated: recommending links between communities tends to notably reduce distances between pairs of users in the network.
- Our experimental results show that, in general, contact recommendation algorithms favor more skewed degree distributions – due to the preferential attachment phenomenon, (Barabási and Albert, 1999) – and redundant edges – since they tend to close distance two paths in the network.

Beyond our initial exploration, we still need to further understand and motivate the value that the proposed metrics might have for the different actors involved in the network: users in the platform, network administrator, owner, etc. For instance, enhancing the degree equality or recommending low tail users might prevent the disengagement of those users less involved in the platform, or recommending people far from the users in the network might enrich their experience. In the next chapter, we delve further on the meaning some of the structural diversity metrics might have for alleviating filter bubbles (Pariser, 2011) that prevent them from receiving diverse information.

9 Effects on information diffusion

Highlights

- *We propose an effective greedy algorithm for enhancing global (structural diversity) properties of a recommendation.*
- *We explore novelty and diversity concepts on the information that travels through networks.*
- *Enhancing structural diversity properties leads to a more novel and diverse information diffusion.*

Creating, consuming and sharing information are some of the most prominent uses of online social networks, as they are the way people communicate to each other. In social media, links represent the paths through which information travels. For example, in Twitter, each time a tweet is posted, it traverses the links between its creator and each of her followers, to be placed in their timelines. Due to this, the presence or absence of links is fundamental to understand how a single piece of information is spread. Looking from a wider perspective, the structure of the network is known to have a great influence on the properties of the diffusion (Burt, 1995, De Meo et al., 2014, Doerr et al., 2011, 2012, Granovetter, 1973).

In Chapter 8, we observe that contact recommendation approaches have clear effects on the structural diversity of the network. However, the consequences these topological changes have for the users in the platform are yet to be considered. As information has a very notable role in online social networks, it seems natural then to study the potential effects that contact recommendation methods have on its properties. In order to give a practical meaning to the structural diversity changes, we do this indirectly: through the direct effect that recommendation algorithms have on the structure of the network, we analyze the changes on different properties of the information flow – the speed of the diffusion, how novel the received information is for the users, and how diverse the messages spread through the network are. We illustrate this in Figure 9.1.

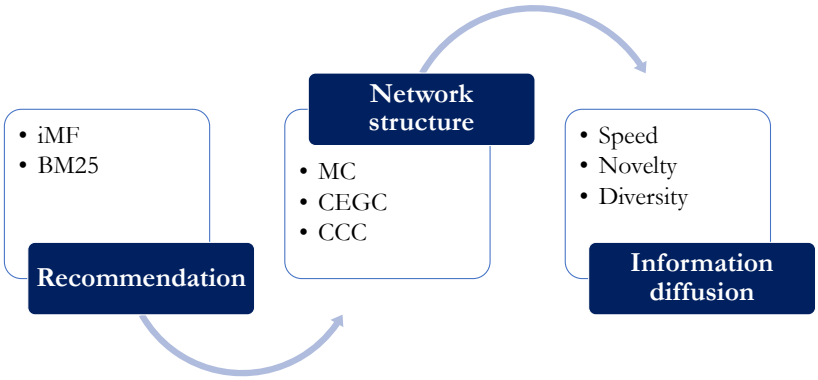


Figure 9.1: Effects of contact recommendation

To check this, we proceed as follows: we first examine the trade-off between accuracy and diversity in recommender systems by greedily optimizing the structural metrics at gradually aggressive diversification of recommendations. Afterwards, we analyze the effect such structural enhancement has on the properties of the information spread, confirming the advantages of those properties for increasing the novelty and the diversity of information that reaches the different users in the network: a positive effect leading to the mitigation of filter bubbles (Pariser, 2011).

The contents of this chapter were partially published in:

- **Sanz-Cruzado and Castells (2018b):** Javier Sanz-Cruzado and Pablo Castells. Enhancing structural diversity in social networks byrecommending weak ties. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018)*, pages 233–241, Vancouver, Canada, October 2018. ACM.

9.1 Research questions

The contents of this chapter address the following research questions:

- **RQ1:** Do contact recommendation algorithms impact the properties of the information flow through the network?
- **RQ2:** Does the enhancement of structural diversity properties have an effect on the properties of the information traversing the network?
- **RQ3:** Is it useful to recommend weak ties?

9.2 Structural diversity enhancement

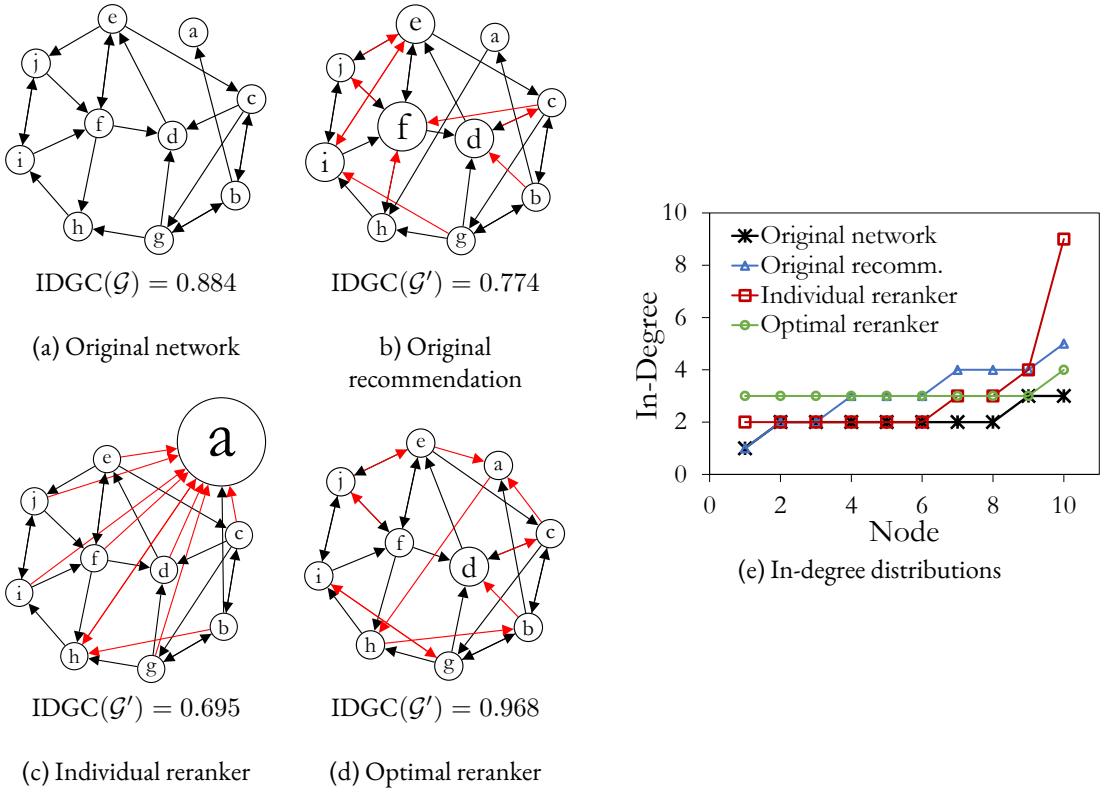
As a first step to determine how the different structural diversity metrics correlate with the properties of the information flow, we want an algorithm that allows us to take the outcome of a recommendation algorithm, and modify it to enhance some these structural metrics. After we have it, we can compare the original algorithm with the modified one, and study the variation on the properties of the diffusion. In this section, we introduce an algorithm that considers gradual aggressivity optimization levels for promoting global properties of the recommendation. The idea behind our approach is to apply a greedy reranking that partially targets the corresponding metric over a well-performing baseline (in terms of accuracy).

9.2.1 Individual reranking

The problem of jointly optimizing a ranking in terms of accuracy and some other property such as its diversity is a NP-hard problem (Agrawal et al., 2009). This is the reason why such optimization has been typically addressed in a greedy manner in the information retrieval and contact recommendation fields (Castells et al., 2015). In IR, two families of approaches have been studied: first, algorithms that maximize the coverage of different explicit query aspects, such as xQuAD (Santos et al., 2010b) or IA-Select (Agrawal et al., 2009); second, methods that optimize the difference between the documents in the ranking, such as maximum marginal relevance (MMR) (Carbonell and Goldstein, 1998, Santos et al., 2010a). In recommendation, similar techniques to these have been applied (Vallet and Castells, 2012). All these approaches are used to promote individual metrics, such as long-tail novelty, subtopic recall or intra-list dissimilarity, where we can achieve global improvements by independently modifying individual rankings.

However, we expose here the reasons why these classical reranking techniques (Carbonell and Goldstein, 1998, Castells et al., 2015, Vallet and Castells, 2012) are not suitable for the present work, where we want to promote global properties. For this, we make use of an example, illustrated in Figure 9.2, in which we apply independent greedy rerankers for each target user in the network to optimize a global metric. In the mentioned example, we want to attain a more balanced in-degree distribution for the network in Figure 9.2(a), i.e. we want to achieve a larger value for the in-degree Gini complement metric. To achieve this, we want to recommend a single contact to each user in the network. We first generate a recommendation, targeting accuracy. We show an example of such recommendation in Figure 9.2(b). We can easily notice that the degree distribution for such recommendation makes the distribution more skewed (contrary to what we seek). So, we apply a greedy reranker to each recommendation ranking.

To apply this strategy, we make two assumptions: first, when we do the reranking of each user's recommendation, the reranker only knows about the original network (Figure 9.2(a)) and the recommendation provided to the target user; second, we assume here that we can select any of the users in the network instead of the recommended one (in more realistic scenarios, we only have access to a fraction of the users of the network, but, since this network is small, this assumption is reasonable for our example). Then, for any user different from "a" and "b", there is a unique way of achieving the largest possible improvement on the IDGC metric: recommending



(a) Original network

b) Original recommendation

(c) Individual reranker

(d) Optimal reranker

(e) In-degree distributions

Figure 9.2: Limitations of independent local rerankers. Red links represent recommendations, and the size of the nodes is proportional to their in-degree. In Figure (e), the nodes are sorted in increasing in-degree order.

them the only node with in-degree equal to 1, i.e. node "a". This is not possible for "a" and "b", since "a" cannot follow himself, and "b" already follows "a". So, for them, the system might recommend any node in the network with in-degree equal to 2 (if someone with larger in-degree was recommended, the distribution would be even more skewed). Figure 9.2(c) shows the problem of this: after the greedy reranker has been applied, every single node in the network has a link towards node "a", making the in-degree distribution even more skewed than the one resulting from the original recommendation (as it can be observed in Figure 9.2(e)). What is more: it might happen (as we show in the figure) that the same node is suggested to both users "a" and "b", further aggravating the problem.

As a result of this limitation, the outcome of applying independent greedy rerankers for each user is far from what we seek: a recommendation that enhances some global property of the network. For this particular example, we would want a resulting network similar to the one illustrated in Figure 9.2(d), which shows the optimally balanced in-degree distribution which we can achieve by recommending one contact to each user in the network. Consequently, we need to define a greedy reranking approach that considers the global nature of the metrics we want to optimize. We introduce such an algorithm in the following section.

9.2.2 Global reranking

Considering the limitations of the individual rerankers we present above, we define a global optimization algorithm, which considers the effect that the recommendation of a single user might have on the result diversification for the others. We first formalize the problem to solve. Let's denote by $\mathcal{R} \subset \mathcal{U}_*^2$ the outcome of a contact recommendation algorithm $\mathcal{R} = \bigcup_{u \in \mathcal{U}} \{u\} \times \mathcal{R}_u$, where $\mathcal{R}_u \subset \mathcal{U}$ is the set of recommended users for the user $u \in \mathcal{U}$. This recommendation is defined by a ranking function $f : \mathcal{U}_*^2 \to \mathbb{R}$ (the function that provides the recommendation scores $f_u(v)$ for each target user u and candidate user v). We seek obtaining new rankings providing a balance between the accuracy of the initial recommendation and the diversity metric that we want to enhance. As we consider a cutoff k for the recommendation we finally provide to the user, defining these new rankings amounts to selecting a subset $\mathcal{S} \subset \mathcal{R}$, where $\mathcal{S} = \bigcup_{u \in \mathcal{U}} \{u\} \times \mathcal{S}_u$ so that $\mathcal{S}_u \subset \mathcal{R}_u$ and $|\mathcal{S}_u| = k$. A common way to approach this is to combine this dual goal by a linear combination of the two objectives (the

Table 9.1: Notation summary for the reranker.

Notation	Meaning
$\mathcal{R} \subset \mathcal{U}_*^2$	Set of (original) recommended links.
$\mathcal{R} \subset \mathcal{U}$	Original set of recommended contacts for user u
$\mathcal{S} \subset \mathcal{R}$	Set of (reranked) recommended links.
$\mathcal{S}_u \subset \mathcal{R}_u$	Reranked set of recommended contacts for user u.
$f_u(v) \in \mathbb{R}$	Recommendation score of the (u, v) link.
$\mu(\mathcal{G}, \mathcal{S})$	Global property to enhance
λ	Trade-off between accuracy and diversity.
$\mathcal{G}'_{\mathcal{S}}$	Extended network by adding to $\mathcal{G}$ the links in $\mathcal{S}$

Algorithm 1: Global Greedy Reranking

Data:
- $\mathcal{R} \subset \mathcal{U}_*^2$ — original recommendations
- $f : \mathcal{U}_*^2 \to \mathbb{R}$ — original recommendation ranking function
- μ — metric to optimize
- k — diversification cutoff
- $\lambda \in [0, 1]$ — degree of diversification
- $\mathcal{G} = \langle \mathcal{U}, E \rangle$ — training network

Result:
- $\mathcal{S} \subset \mathcal{R}$ — modified recommendations

begin
- $\mathcal{S} \leftarrow \text{sort}(\mathcal{R}, f)$ `// Edges are grouped by source node and sorted by` f
- **for** $u \in \mathcal{U}$ **do**
  - **for** $i \leftarrow 1$ **to** k **do**
    - $j_0 \leftarrow \underset{j:k<j\leq|S_u|}{\arg\max}\ \phi\left(j|\mathcal{S}, u, i, f, \mu, \lambda\right)$ `//` $\mathcal{S}_u \equiv$ `ranking for user` u `in` $\mathcal{S}$
    - **if** $\phi\left(j_0|\mathcal{S}, u, i, f, \mu, \lambda\right) > \phi\left(i|\mathcal{S}, u, i, f, \mu, \lambda\right)$ **then**
      - $\text{swap}(S_u, i, j_0)$
- **return** R

Function $\phi(j|S, u, i, f, \mu, \lambda)$**:** `// The dual objective function`

begin
- **return** $(1 - \lambda) \cdot \mathit{norm}\left(f_u(S_u[j])\right) + \lambda \cdot \mathit{norm}\left(\mu\left(\mathcal{G}, \mathcal{S}_{\langle u:i/j\rangle}@k\right)\right)$

original ranking – for accuracy – and the diversity). We can express the function we seek to optimize as:

$$\mathcal{S} = \underset{\substack{\mathcal{S}'\subset\mathcal{R}\\|\mathcal{S}'_u|=k}}{\arg\max}\,(1-\lambda)\sum_{u\in\mathcal{U}}\sum_{v\in\mathcal{S}'_u} f_u(v) + \lambda\mu\left(\mathcal{G}, \mathcal{S}'\right) \tag{9.1}$$

where $\mu(\mathcal{G}, \mathcal{S})$ is the targeted diversity metric. For instance, if we wanted to optimize some structural diversity metric, we could easily build the extended network $\mathcal{G}'_{\mathcal{S}}$ by adding the links in the recommendation subset $\mathcal{S}$ (i.e. $\mathcal{G}'_{\mathcal{S}} = \langle \mathcal{U}, E \cup \mathcal{S}\rangle$). Parameter λ adjusts how aggressive the reranking is, from no change at $\lambda = 0$ to ignoring the initial ranking at $\lambda = 1$. We summarize the previous notation in Table 9.1.

Once we have defined the target function for our problem, we define our greedy procedure (see Algorithm 1). For each individual ranking, S_u, using a top-down perspective, we consider swapping the i-th element in the top k of S_u with the user at position j below k that maximizes the following greedy objective function:

$$\phi(j|S, u, i, f, \mu, \lambda) = (1 - \lambda) \cdot \text{norm}\left(f_u(S_u[j])\right) + \lambda \cdot \text{norm}\left(\mu\left(\mathcal{G}, \mathcal{S}_{\langle u:i/j\rangle}@k\right)\right) \tag{9.2}$$

where $\mathcal{S}_{\langle u:i/j\rangle}$ denotes swapping the i-th and j-th elements in the ranking for S_u, and $\mathcal{S}@k$ is just the subset of the elements ranked in the top k of each ranking $\mathcal{S}_v$. Before applying this best swap, we check if $\phi(j|S, u, i, f, \mu, \lambda) > \phi(i|S, u, i, f, \mu, \lambda)$, that is, if the swap improves the value of the target function. Otherwise, we do not apply the change. This way, we can ensure that the reranking does not provide worse results than the original recommendation when optimizing the target function in equation 9.1. Finally, as the values of

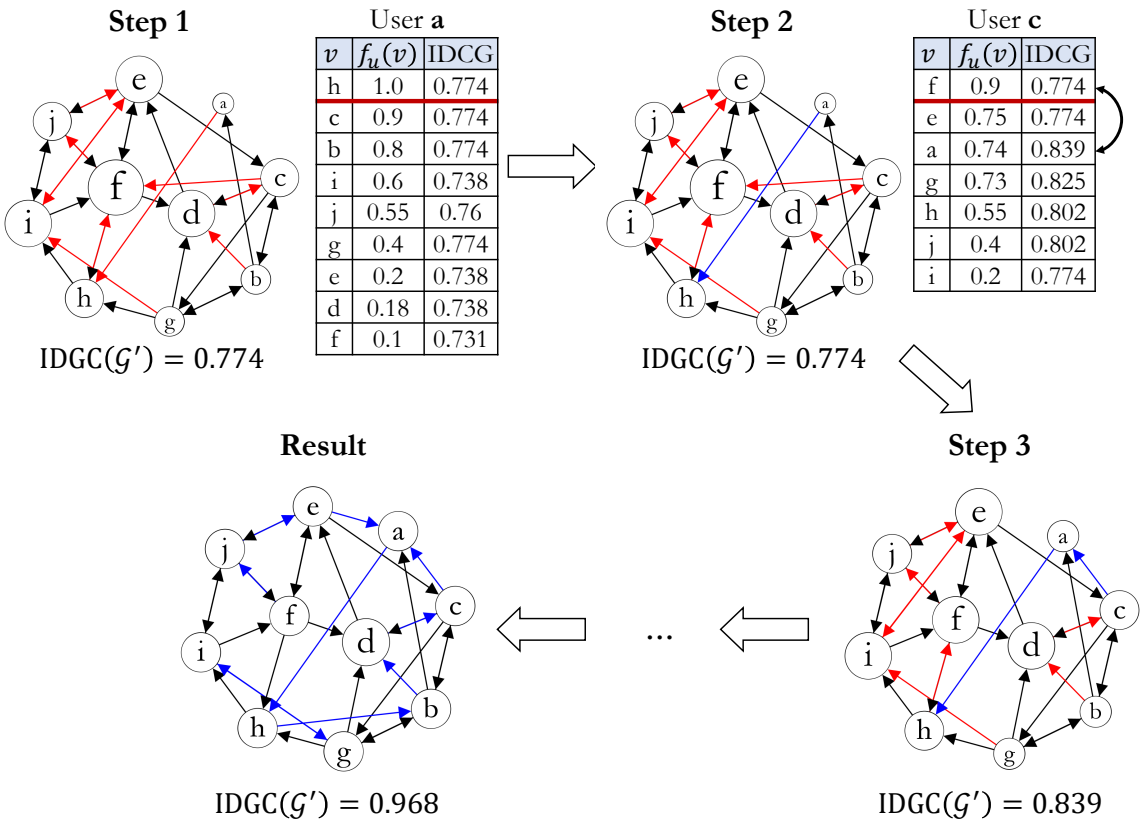


Figure 9.3: Example of our global greedy reranking algorithm with $\lambda = 1$. We seek to optimize IDGC@1. For finding the resulting network in the figure, the selected order for the users is {a,c,b,d,e,f,g,h,i,j}

f and μ have to be comparable for the aggregation (Vallet and Castells, 2012), we have to apply a normalization to both values, which we denote as norm$(\cdot)$ in our greedy function. In our case, we use the rank-sim normalization scheme (Lee, 1997). Given a ranking of candidate users $\mathcal{R}$ depending on some function g, the normalized score is defined as

$$\text{norm}\left(g(v)\right) = \frac{|\mathcal{R}| - \text{rank}(v) + 1}{|\mathcal{R}|} \tag{9.3}$$

where $|\mathcal{R}|$ is the size of the ranking, and rank(v) is the position of candidate v in the corresponding ranking.

We illustrate an example of this algorithm in Figure 9.3, following the example we introduce in Figure 9.2. Again, we want to greedily optimize the IDGC metric at $k = 1$. Supposing that we take $\lambda = 1$, we represent the edges of the original network as black arrows, the original recommendation links as red arrows and the (already) reranked edges in blue. Our first observation is that, after each step, the global metric does not decrease (as we expected) and, what is more: in some cases, it might even reach the optimal value we seek (depending on the original score values and the selected order of users, we can observe the network we indicate in the figure).

9.2.3 Results

We illustrate in Figure 9.4 the effect of applying the reranking approach for the modularity complement, community edge Gini complement (forcing links to go outside communities) and clustering coefficient complement for the Twitter interaction networks. We include in Appendix E specific details about fast implementations of such rerankers. As the initial recommendation, we take the links recommended by the BM25 (Robertson and Zaragoza, 2009) and implicit matrix factorization algorithms (Hu et al., 2008), as the most effective algorithms for the interaction networks in our experiments in Chapters 5 and 8. We target the three metrics at cutoff $k = 10$. The figures show the common trade-off between the accuracy and the diversity metrics we enhance at cutoff $k = 10$ (Castells et al., 2015). We also observe that each reranker is the best at enhancing the optimized metric, and both CEGC and CCC also provide the best trade-off with accuracy (in the case of MC, the modu-

larity complement values for the enhancing CEGC and MC approaches are similar, and CEGC shows a better trade-off).

An interesting observation can be made for the community metrics: if we just enhance the number of weak ties in the network without giving any attention to their distribution in the network (as it occurs with the modularity complement reranker), such distribution might become even more imbalanced than before (as we show in the metric comparing ICEGC with accuracy). This does not happen, however, when we optimize, at the same time, the number of links traveling outside their communities and their distribution (the CEGC reranker), which generally improves both the CEGC and the ICEGC metrics, evidencing the claim that not all ties are equally weak – something that the modularity complement does not properly take into account.

9.3 Information diffusion

Once that we have studied how to greedily enhance structural diversity properties of recommender systems, we want to analyze the effect that these metrics have on the flow of information that spreads through the network in the form of messages that the users post and share in the network (for example, tweets in Twitter).

9.3.1 Speed, novelty and diversity

We first define how we are going to measure the changes that the recommendation produces on the information flow. We explore three different dimensions of the information diffusion: the speed, the novelty and the diversity of the information that reaches the different users in the network.

Speed

The speed of the diffusion is one of the most commonly analyzed features in diffusion processes (De Meo et al., 2014, Doerr et al., 2011, 2012) and it is related to the efficiency of the network to propagate messages. The speed of the information flow is measured as the number of messages received by each user in the network at a given moment of time t:

$$\text{speed}(t) = \sum_{u \in \mathcal{U}} |\mathcal{M}_u(t)| \tag{9.4}$$

where $\mathcal{M}_u(t)$ denotes the set of messages that user u has received since the diffusion started to be measured, until time t.

Novelty

In order to measure the novelty and diversity of the information flow, we need some kind of topical information about the propagated messages or tags. For example, we use in this work the hashtags contained in the tweets published by users in Twitter. If we denote by $\mathcal{H}$ the set of hashtags, then the messages (the tweets) can be defined as subsets $i \subset \mathcal{H}$. Considering this representation, the novelty of a tweet is determined by the number of tweet hashtags that the user has not used before. Specifically, we define the **external hashtag rate (EHR)** of the information flow as:

$$\text{EHR}(t) = \frac{\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{M}_u(t)} \left| i \setminus \mathcal{H}_u^0(t) \right|}{\sum_{u \in \mathcal{U}} \sum_{i \in \mathcal{M}_u(t)} |i|} \tag{9.5}$$

where $\mathcal{H}_u^0(t) = \bigcup_{i \in \mathcal{M}_u^0(t)} i$ denotes the set of tags used by u in the messages she created up to time t (denoted here as $\mathcal{M}_u^0(t)$).

Diversity

Finally, for measuring the diversity of the information flow, we consider that the information is diverse as long as the hashtags are evenly distributed in the information diffusion. We consider two possible metrics, both relying (again) on the complement of the Gini coefficient as a measure of diversity.

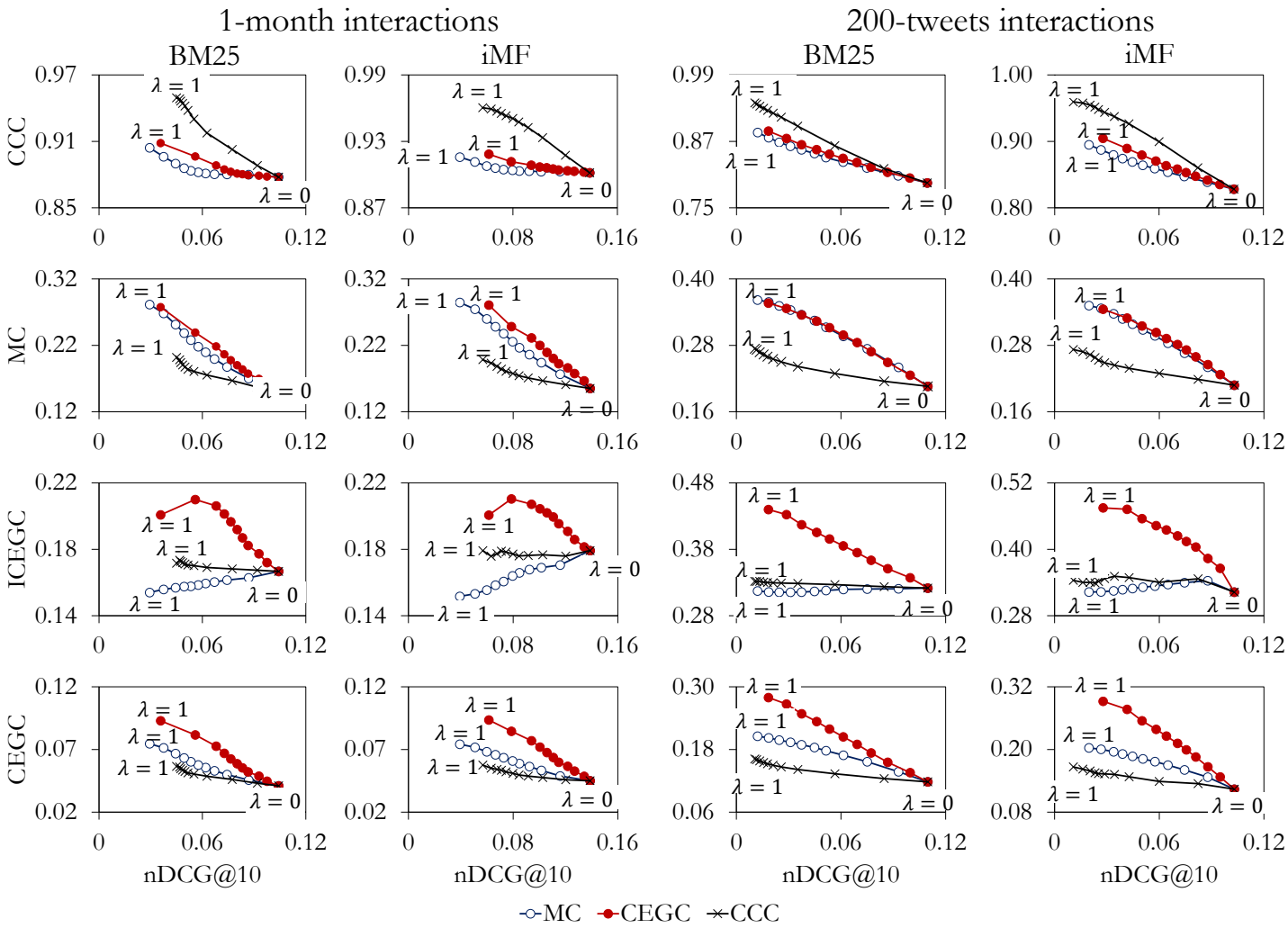


Figure 9.4: Plot of structural diversity (y axis) against nDCG (x axis) for the MC, ICEGC, CEGC and CCC metrics, and the MC, CEGC and CCC rerankers. Points on the curves represent reranked recommendations for λ ranging from 0 (initial recommendation) to 1 (maximum diversity) by increments of 0.1.

The first metric considers how evenly the hashtags have been received by the users. We denote this as **hashtag Gini complement (HGC)** , and we define it by:

$$\mathrm{HGC}(t) = 1 - \frac{1}{|\mathcal{H}| - 1} \sum_{j=1}^{|\mathcal{H}|} (2j - |\mathcal{H}| - 1) p(h_j|t) \tag{9.6}$$

where h_j represents the j-th least spread hashtag and:

$$p(h|t) = \frac{\sum_{u \in \mathcal{U}} |\{i \in \mathcal{M}_u(t) | h \in i\}|}{\sum_{u \in \mathcal{U}} |\{i \in \mathcal{M}_u | i \neq \emptyset\}|} \tag{9.7}$$

The second diversity metric that we consider in this thesis measures how balanced the number of users which have received each hashtag is. We denote this metric as **hashtag user Gini complement (HUGC)**, and we define it using the same equation as

$$\mathrm{HUGC}(t) = 1 - \frac{1}{|\mathcal{H}| - 1} \sum_{j=1}^{|\mathcal{H}|} (2j - |\mathcal{H}| - 1) p(h_j|t) \tag{9.8}$$

with

$$p(h|t) = \frac{|\{u \in \mathcal{U} | h \in \mathcal{H}_u(t)\}|}{\sum_{h^* \in \mathcal{H}} |\{u \in \mathcal{U} | h^* \in \mathcal{H}_u(t)\}|} \tag{9.9}$$

where $\mathcal{H}_u(t) = \bigcup_{i \in \mathcal{M}_u(t)} i$ is the set of tags in the tweets u has received up to time t.

9.3.2 Experiments

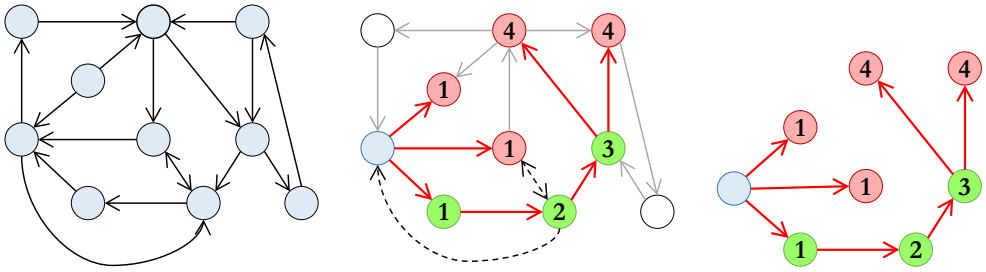
Finally, we describe the experiments we have conducted to check on the effects that contact recommendation has on the exchange of information that occurs through the network.

Diffusion procedure

In order to analyze the impact of contact recommendation in the information diffusion process, we simulate the publication and transmission of user generated contents over a network $\mathcal{G}'$ extended by adding all the recommended links to the original graph (as we do in Chapter 8 and Section 9.2). The procedure we follow for such simulations is a simplification of the way information spreads in Twitter.

In Twitter, the information passing procedure is the following: first, a user creates and posts a tweet in the platform. That tweet appears in the timeline of her followers, who might decide (or not) to forward that content to their own followers by retweeting such tweet. If one of such users decides not to forward that tweet, it becomes the end of the path for the message. Otherwise, the tweet reaches the followers of those users and the process is repeated until no one retweets it.

Our simulation procedure roughly mimics this procedure, by considering the data we collected from the network. Each user in the network starts with two lists of tweets: the first one contains the set of tweets she has posted on Twitter, and the second, the list of contents she has retweeted. To simplify, our simulation model considers that the time is discrete: we run an iterative procedure where each iteration is considered as a "time point". At each step, every user in the network randomly selects and posts one of her authored tweets (without replacement – if no tweet is available to post, the user does not post anything). Such tweets are sent to the timelines of their followers (which act here as a personal inbox of the received tweets). Then, for such inbox, we observe the received messages, and each user decides whether they are worth forwarding or not. The criteria we use is whether they retweeted the content for real according to our Twitter data. Hence at every step, each user adds an original tweet to the simulation and forwards some contents from other users, depending on their availability. If a user receives one of her authored contents or receives a content more than once, she ignores it



(a) Original network (b) Tweet diffusion (c) Diffusion tree

Figure 9.5: Example of the diffusion of a tweet in a network.

Table 9.2: Data description for the information diffusion experiment

Dataset	#Tweets	#Hashtags (unique)	#Retweets
1 month	87,837	110,578 (1,115)	24,661
200 tweets	21,513	24,623 (378)	2,202

(and the simulation acts as if she had never received such repeated content). Once all original tweets have been published and all the timelines are empty, the simulation stops.

The configuration of our simulation procedure makes it deterministic: since it applies an exhaustive propagation, this procedure is equivalent to determine the diffusion paths for every tweet, following the links in the network backwards, and, stopping the spreading each time the tweet reaches a user who has not retweeted it in our dataset. We obtain a search tree for each tweet, covering the set of users which have received the tweet (everything we need to compute our interest metrics). However, the simulation perspective enables a generalization of our procedure to any other communication protocol one may wish to explore (Demers et al., 1987, Doerr et al., 2011, Goldenberg et al., 2001, Kempe et al., 2003). We exemplify the exploration for a single tweet in Figure 9.5. Figure 9.5a) shows the original directed network and Figure 9.5b) illustrates the possible diffusion paths. Starting from the user highlighted in blue, we show with red arrows the paths followed by a tweet. Green nodes retweet the content whereas red nodes do not, and the numbers inside indicate the distance to the source node in the propagation tree we plot in Figure 9.5c). As it can be observed, dotted black lines represent paths that the tweet might have followed if the destination nodes had not received (or created) the tweet in an early step. Finally, white nodes and grey links represent nodes and paths which have not been reached by the spread of the user authored content.

Setup

In our experiments, we use the two Twitter interaction networks described in Section 4.1: as we do in chapter 8, we apply the contact recommendation algorithms over the input network, and, then, we add all the recommended links to the graph to form the expanded network. As the recommendation algorithms, we take the optimal iMF and BM25 approaches for each network (according to the parameter selection in Appendix C), and we apply over them the corresponding MC, CEGC and CCC rerankers described in Section 9.2. For each algorithm, we consider gradual reranking levels, ranging from $\lambda = 0$ to $\lambda = 1$ in increments of 0.1 (as it is shown in Figure 9.4). In total, for each recommender, we consider 33 different recommendations (3 rerankers $\times$ 11 values of λ). The information diffusion protocol we describe in the previous section is then run over each expanded network, and we measure the speed, novelty and diversity of the propagated messages.

As the information passed through the networks, we use the tweets collected during the dataset download, as described in Section 4.1. In particular, we resort to the test tweets (i.e. those posted by the users after the temporal split point). We restrict that set of tweets in two ways: first, we only consider those tweets containing hashtags – otherwise, they do not have any impact on our novelty and diversity measures; second, to avoid noise and heavy-tail distortion, we do not consider any hashtag that appears in less than 25 tweets. Table 9.2 shows the details on the number of tweets and hashtags in our experiments, as well as the maximum number of retweets that might be done during our experiments.

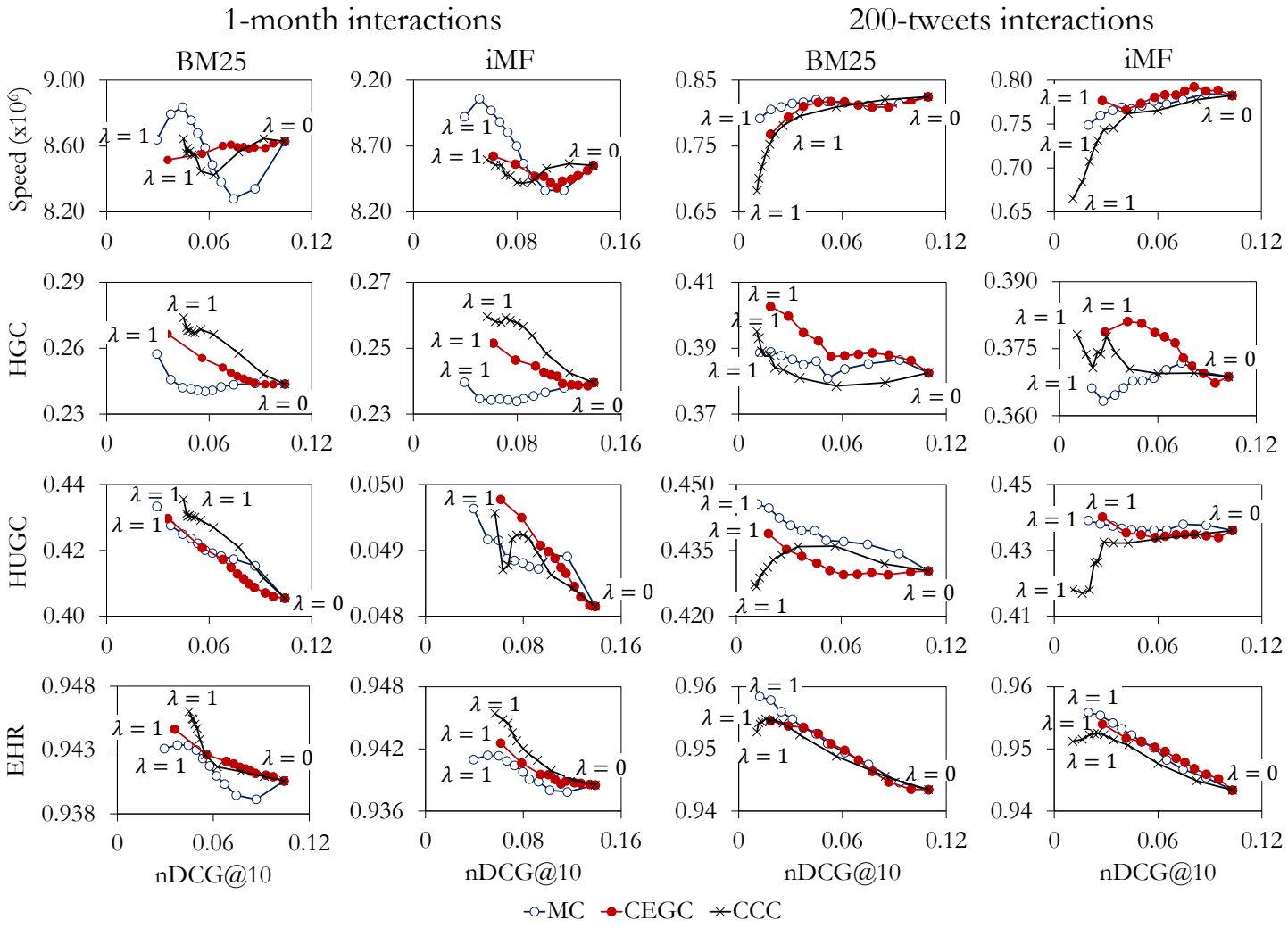


Figure 9.6: Information flow efficiency, diversity and novelty (y axis) against recommendation accuracy (nDCG@10, x axis) for the diversity-targetting rerankers, from $\lambda = 0$ to $\lambda = 1$.

Results

The outcome of the diffusion experiments is shown in Figure 9.6, as the trade-off between the properties of the information flow at the end of the process (in the y axes) and the accuracy of the recommendations in terms of nDCG at cutoff 10 (in the x axis). A first observation is that (to different extent), the novelty and diversity of the information flow are improved by the three structural diversity metrics we consider in our experiments. This provides a practical meaning for these measures, since higher values contribute to a healthier social media and the mitigation of filter bubbles (Pariser, 2011) in the network.

We cannot, however, state that such metrics are useful for increasing the propagation speed: for the 1-month dataset, the optimization of the recommendation metrics leads to mixed results in the speed, and they lead to a slowdown in the 200-tweets one. Notably, we can observe that there is no clear correlation between speed, novelty and diversity, thus showing that the latter properties are not just a trivial reflection of the information volume.

Focusing on the results for the particular metrics and rerankers, we observe some interesting behaviours. Overall, seems to provide the best results in terms of the diversity of the information flow, as, for both diversity metrics (HGC and HUGC), our results consistently show a (almost) monotone growth as this network property increases. It is also the best performing reranker in terms of HGC in the 200-tweets dataset and for the iMF baseline in the HUGC values for the 1-month dataset. The modularity complement rerankers are rather suboptimal for the HGC metric, although it provides good values for HUGC in both datasets. This observation supports our hypothesis that, although creating links between communities does improve the network, taking a further step and balancing the number of links which travel between different community pairs brings an even greater advantage to the network. As for the CCC metric, it represents the metric with the most erratic behaviour (for instance, we can observe the HUGC metric for the iMF reranker in the 1-month dataset).

Finally, for the novelty of the information flow, measured by the EHR metric, there seems to be no clear winner, as the three reranking strategies show a similar level of improvement. This observation is consistent with the theory of weak ties proposed by Granovetter (1973), who states that weak links act as sources of novel information for the different people in a social network.

9.4 Conclusions

In this chapter, we analyze the effects of contact recommendation approaches in the diffusion of information through social networks. We focus on the effects they have on the information novelty and diversity, as a desirable property that might lead to filter bubble mitigation (Pariser, 2011). To perform such analysis, we first propose a global greedy optimization algorithm for network properties that allows a gradual re-targeting of the purpose of the recommendation towards the desired global metrics. We apply such approach to several recommendation methods and we explore the relation between the enhanced structural diversity properties of the recommendation and the properties of the information diffusion. We summarized next the main conclusions of the study we present in this chapter:

- Classical reranking strategies that only consider one recommendation ranking at a time are not able to properly enhance global properties such as the structural diversity of the network, due to clashes between the recommendations.
- Such limitations can be overcome by defining greedy algorithms that consider all the recommendations for the different users in the network at the same time.
- There is a positive correlation between enhancing structural diversity metrics (MC, CEGC and CCC) and the novelty and diversity of the information flow. Therefore, using contact recommendation to promote the structural diversity of the network appears as a potential tool to mitigate filter bubbles in social media, leading to healthier platforms.
- Taking weak ties as links between communities, if we consider balancing their distribution over the different communities in addition to promoting their presence leads to higher gains on information diversity.

Part IV

Interactive recommendation

"Chance encounters are what keep us going."
Haruki Murakami

Common offline evaluation settings treat recommender systems as (mostly) static objects. However, in real world applications, this is far from true: recommender systems are constantly adapting, so they can integrate the information provided by the interactions of the users with the system. This cyclic information exchange between users in the system is increasingly attracting the interest of researchers, who model this problem – known as interactive recommendation – as a genuine reinforcement learning problem.

In Chapter 10 we investigate the interactive recommendation task from a collaborative filtering perspective. We first formalize the problem and establish connections to reinforcement learning, and to a family of strategies known as multi-armed bandits. We then elaborate a novel recommendation method that integrates multi-armed bandit in neighbor-based collaborative filtering. Our proposal can be understood as a variant of the user-based nearest neighbors scheme, endowed with a stochastic mechanism for exploring the potential neighbors of the target user: a Thompson sampling multi-armed bandit approach. Then, we run experiments over datasets from different recommendation domains – including contact recommendation as a particularly interesting case – and show that our bandit algorithm achieves accuracy enhancements when compared to other (both myopic and stochastic) state of the art approaches in an extreme cold start situation.

10
Multi-armed bandits in recommendation

Highlights

- *We formalize interactive recommendation and explore its connection with reinforcement learning.*
- *We propose a novel interactive recommendation algorithm based on neighbor-based collaborative filtering and multi-armed bandits.*
- *Our method shows great potential for overcoming the extreme cold start problem in recommendation.*

Studies regarding offline evaluation methodologies in recommendation are commonly addressed in a static way, where only a single recommendation is performed for each target user in the network. In contact recommendation, a fixed amount of data (links, features, etc.) is provided as input to the system for its training phase, and, afterwards, a set of people is suggested to each user in the network. Those recommendations are then evaluated according to some specific metrics. As the reader might observe, this is the methodology we apply in our experiments in Parts I,II and III, regardless of whether we assess the accuracy of the recommenders, their novelty, their diversity or the effect they have on the evolution of the underlying social network structure.

Nevertheless, following the fact that we define social networks as dynamic objects in a continuous evolutionary process, the same can be stated for recommender systems. It is common for the users to change their preferences over time (Hariri et al., 2014): personal circumstances such as age, location, education, job, close relations, etc. might affect who we follow in social networks (or, in other contexts, what movies we want to watch). In case recommenders were static objects, which did not change over time, they would not be able to capture this shifts in taste. This, consequently, would greatly reduce their utility for the users.

Therefore, if we observe recommender systems in online environments, their functioning is the following: considering the information gathered by the recommender, it provides suggestions to the users, impelling them to consume certain items (or, in contact recommendation, create new connections). Users provide feedback to those suggestions, indicating whether they have found them relevant or not. Such information is then collected by the system, which uses it to improve the quality of the recommendations. This information trade can be modelled as a feedback loop between the users and the recommenders, as illustrated in Figure 10.1. This vision of recommendation as a feedback loop is known as interactive recommendation.

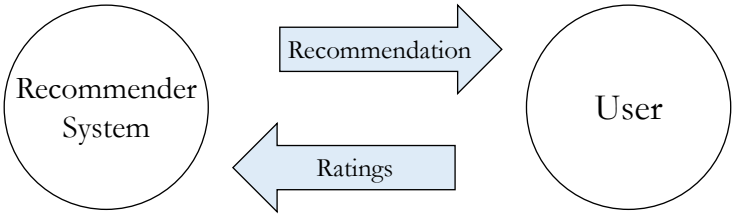


Figure 10.1: Feedback loop in recommender systems.

This cyclic perspective represents a more complex vision of recommender systems, but also a much more realistic one. Consequently, it has increasingly attracted the interest of both industry and academy in the last few years (Kawale et al., 2015, Li et al., 2010, 2016, Wang et al., 2019a, Zhao et al., 2013). Most of the research and development associated to the feedback loop in recommender systems has come hand in hand with the adoption of principles and techniques from the machine learning paradigm known as reinforcement learning (Sutton and Barto, 2018). Establishing meaningful connections between both tasks has lead to the development of new

recommendation approaches (Gentile et al., 2014, Kawale et al., 2015), as well as new evaluation frameworks considering the feedback loop (Li et al., 2011).

This chapter explores the application of a particular family of reinforcement learning techniques for addressing the interactive recommendation problem: multi-armed bandits (Lattimore and Szepesvári, 2020, Sutton and Barto, 2018). Our work approaches those techniques from a pure collaborative filtering perspective (Bresler et al., 2014, Zhao et al., 2013), where algorithms do not need any side-information about users and/or items. We elaborate a simple multi-armed bandit algorithm based on a nearest-neighbors scheme, and we test its potential when compared to other recommendation techniques in extreme cold start conditions, i.e. when we have no ratings in our system.

Although the research we undertake in the rest of the thesis can only be applied on the particular contact recommendation domain, the concepts and techniques we develop and introduce in this chapter can be applied to other recommendation domains. Consequently, we broaden our perspective to consider the general item recommendation case. Nonetheless, the methods we consider here can also be applied for recommending people to people in social networks, and, we therefore keep an important focus on this domain, as a particular and compelling one.

A significant amount of the work introduced in this chapter was published in

- **Sanz-Cruzado et al. (2019):** Javier Sanz-Cruzado, Pablo Castells and Esther López. A Simple Multi-armed Nearest-neighbor Bandit for Interactive Recommendation. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019), pages 358–362, Copenhagen, Denmark, September 2019. ACM.

10.1 Interactive recommendation

Interactive recommendation represents a perspective on recommender systems that considers not only their properties in a static, single time step, but also the continuous information exchange between the users and the systems. It represents a realistic scenario, mimicking that of real online platforms, such as Twitter, Netflix or Spotify, where not only the platform evolves: the algorithms do it too. As a first step in our research, we formalize this task and establish meaningful relations between it and reinforcement learning, so we can later apply reinforcement learning techniques for solving the problem.

10.1.1 Formulation and notation

We introduce here a formal definition and the corresponding notation for the interactive recommendation task. In this task, we have a system with a set of users and items, denoted, respectively, as $\mathcal{U}$ and $\mathcal{I}$, and a discrete time scale[1] $t = 1, 2, \ldots, T$. The system has access, at every time step, to all the past interactions of users in the system, the history $\mathcal{H}_t$, which contains a list of triplets $(u, i, r_u(i))$ where $u \in \mathcal{U}, i \in \mathcal{I}$ and $r_u(i)$ shows the (either implicit or explicit) rating u has provided to i at time t. If the triplet $(u, i, r_u(i))$ does not appear in the history, we assume that the user is unaware of the item.

Then, at time t, a single user, u_t, arrives at the system. If we denote the set of items the user u_t already knows by $\mathcal{I}_u(t) = \{i \in \mathcal{I} | (u, i, r_u(i))\}$, the system provides a recommendation $R(u_t) \subset \mathcal{I} \setminus \mathcal{I}_u(t)$ according to a certain recommendation algorithm π, which we describe later. Then, the user observes the provided items, consumes some of them and provides feedback to the system in the form of a rating $r_{u_t}(i)$, where $i \in R(u_t)$. To model all the possible outcomes, we might say that $r_{u_t}(i) \in \mathbb{R} \cup \{\emptyset\}$ where $r_{u_t}(i) = \emptyset$ means that the user did not rate the item. The system takes this ratings as input for updating π.

Finally, a recommendation algorithm π is defined as an object with three different methods:

- **Train:** This function takes as input the history of the system before any recommendation has been done (which we denote as $\mathcal{H}_0$, and uses it to learn the parameters of the algorithm, which we denote as θ. Examples of such parameters include the latent factors in matrix factorization (Koren et al., 2009) or the similarities in nearest neighbors (Ning et al., 2015). When the history is empty, this function just initializes such parameters.

[1] This time scale does not necessarily represent equally separated time intervals. For example, we can take each time point as an event in the system (e.g. a user login)

Algorithm 2: Interactive recommendation.

Data:
$\mathcal{H}_0 = \{(u, i, r_u(i)) | u \in \mathcal{U}, i \in \mathcal{I}\}$ Training data
π Recommendation algorithm (policy)

begin
  $\mathcal{H} \leftarrow \mathcal{H}_0$
  π.train($\mathcal{H}_0$)
  for $t \leftarrow 1$ **to** T **do**
    Observe user $u_t \in \mathcal{U}$
    $R \leftarrow \pi$.recommend(u_t) // $R \subset \mathcal{I} \setminus \mathcal{I}_u(t)$
    $\mathcal{H}_{new} \leftarrow \emptyset$ **for** $i \in R$ **do**
      $r_{u_t}(i) \leftarrow u_t$.rate($i$)
      $\mathcal{H}_{new} \leftarrow \mathcal{H}_{new} \cup (u_t, i, r_{u_t}(i))$
      $\mathcal{H} \leftarrow \mathcal{H} \cup (u_t, i, r_{u_t}(i))$
    π.update($\mathcal{H}_{new}$)

- **Update:** This method modifies the θ parameters according to the most recent knowledge acquired by the system. It keeps the system up to date, so it is able to provide relevant suggestions to the users. Depending on the recommendation algorithm, this is equivalent to the training function (but using the full history at update time, $\mathcal{H}_t$ instead of $\mathcal{H}_0$), or it is able to modify the θ parameters using the last retrieved interactions $\{(u_t, i, r_{u_t}(i)\}_{i \in R(u_t)}$.
- **Recommend:** It uses the previous knowledge to select a subset of items $R(u) \in \rho(\mathcal{I} \setminus \mathcal{I}_u(t))$ where $\rho(X)$ is the power set of X, i.e. a set containing all the possible sets containing elements in X. Given a user u, this methods is represented as a (stochastic) function $f_u : \mathcal{I} \setminus \mathcal{I}_u(t) \rightarrow \rho(\mathcal{I} \setminus \mathcal{I}_u(t))$. As we explore in previous chapters, it is common to fix the size of $R(u)$ to a cutoff k.

Then, the goal of interactive recommendation consists on providing, in the interval of time between $t = 1$ and time $t = T$, a set of recommendations which are useful for the different users $u \in \mathcal{U}$ in the platform, or, in other words, maximize the positive ratings over time: $\sum_{t=1}^{T} \sum_{i \in R(u_t)} r_{u_t}(i)$.

Considering this, we can model the cyclic task as illustrated in Algorithm 2: first, we train our recommendation algorithm using some initial data $\mathcal{H}_0 = \{(u, i, r_u(i) | u \in \mathcal{U}, i \in \mathcal{I}\}$. Then, the iterative process begins: at time t, a user u_t receives a recommendation from the system. Typically, the item receives a reduced set of items. The user then provides feedback over that recommendation (we note here that even ignoring a recommendation does provide information about how good the recommendation is). The recommendation uses that feedback to update itself and their value functions. Although in real applications it is common to recommend several items at once, in this chapter we only consider the case where we recommend a single item.

10.1.2 Relation to reinforcement learning

The interactive recommendation problem has multiple similarities to the reinforcement learning (RL) problem – and, because of this, several works have addressed the study of the recommendation feedback loop as such. In order to use RL techniques in recommender systems, it is first necessary to relate the different elements of RL (as described in section 2.4) to the aspects of interactive recommendation. These connections between the problems are summarized in Table 10.1 and discussed next.

The first relations we have to establish are the ones with the fundamental elements in reinforcement learning: the agent – the element that takes actions to satisfy its objective – and the environment – the interacted element which provides feedback to the agent. Observing the feedback loop in Figure 10.1, it is reasonable to think that the system has to play the role of the agent, and the users have to act as the environment: the system performs actions (here, recommends items) to maximize a numerical quantity (which reflects properties such as the satisfaction of the users, the revenue of the platform, etc.). These recommendations are aimed at the people in the platform, who react to them, providing feedback to the system.

Table 10.1: Relation between reinforcement learning and recommendation

Reinforcement learning	Recommender systems
Agent	System
Environment	Users $\mathcal{U}$
Policy	Recommendation algorithm
Reward	Ratings $r_u(i)$
Estimated value	Ranking function $f_u(i)$
Actions	Possible recommendations $i \in \mathcal{I}$

That feedback, which comes in the form of (either implicit or explicit) ratings, constitutes the immediate reward on the reinforcement learning task. The system provides recommendations according to a given algorithm. This algorithm plays the role of the policy, and, given a context or state (the target user of the recommendation) somehow estimates the probability of selecting a candidate item for the recommendation (i.e. the ranking function $f_u(i)$ of the recommendation provides the value function for the item). Finally, for some algorithms, we have a model of the environment: for example, in matrix factorization, the latent factors of the users model their behaviour by establishing how likely they are to enjoy different aspects of the items (Koren et al., 2009).

Now that we have formally established the interactive recommendation task, and relations with the reinforcement learning problem, we can understand and develop new recommendation approaches based on reinforcement learning techniques. In the following sections, we apply this to multi-armed bandit techniques for recommendation.

10.2 Collaborative filtering bandits

From the wide range of reinforcement learning methods and techniques (Sutton and Barto, 2018), the ones which have received the greatest attention in the recommender systems field are the multi-armed bandits (MAB) techniques (Bresler et al., 2014, Hariri et al., 2014, Li et al., 2010, 2016, Zhao et al., 2013). As we introduce in Section 2.4, multi-armed bandit problems are state-less reinforcement learning methods which, at each time point t, select an action a from a pool of pre-selected ones $\mathcal{A}$ (known as arms) according to an estimation of how good this action is – the value of the action. That value might be different according to a context x_t, which represents the current state of the environment (for example, in personalized interactive recommendation, this context might represent the target user). Therefore, we denote the value of an arm as $v(a, x_t)$.

In this section, we explore a selection of collaborative filtering bandits for item recommendation. Considering the mapping between interactive recommendation and RL, the multi-armed bandit strategies we consider here take the role of the policy. The candidate items $\mathcal{I} \setminus \mathcal{I}_u(t)$ are then the set of (available) candidate items in the recommendation. As we make here the common assumption (at least in offline experiments) that an item cannot be recommended to a user if the user has already rate it, we note that the set of available arms (items) is determined by the target user u_t – which takes then the role of the context for the bandits.

In the description of the different approaches, we include an analysis of their computational complexity, in terms of both the time they need to learn their optimal parameters, the time they need to generate a recommendation and the time they need to update their knowledge once they receive a single rating, as well as the maximum amount of memory they need to perform well. For simplifying, in such analysis, we assume that the number of candidate items is $|\mathcal{I}|$ instead of $|\mathcal{I} \setminus \mathcal{I}_u(t)|$ (which, given that the rating matrix is usually sparse, is a good approximation), and we remove the temporal indexes. The complexity analysis is summarized in Table 10.2, along with the values for some myopic algorithms: iMF (Hu et al., 2008), user-based kNN (Ning et al., 2015), random and popularity-based recommendation.

10.2.1 Non personalized methods

The first (and simplest) approximation to the use of multi-armed bandit policies for recommendation involves the straightforward use of bandit algorithms to produce recommendations (Li et al., 2010). Algorithms in this

Table 10.2: Bandit computational cost for a single iteration.

	Algorithm	Memory	Training time	Recommendation + Update time
Stochastic	ε-greedy	$O(\lvert\mathcal{I}\rvert)$	$O(\lvert\mathcal{H}_0\rvert)$	$O(\varepsilon + (1-\varepsilon)\lvert\mathcal{I}\rvert)$
	Thompson sampling	$O(\lvert\mathcal{I}\rvert)$	$O(\lvert\mathcal{H}_0\rvert)$	$O(\lvert\mathcal{I}\rvert \cdot b)$
	InterPMF	$O\left(k^2 \cdot (\lvert\mathcal{U}\rvert + \lvert\mathcal{I}\rvert)\right)$	$O\left(c \cdot k^2 \cdot (\lvert\mathcal{H}_0\rvert + + k \cdot (\lvert\mathcal{I}\rvert + \lvert\mathcal{I}\rvert)\right)$	$O(k \cdot \lvert\mathcal{U}\rvert + k^3)$
	CLUB	$O\left(\lvert\mathcal{H}\rvert + \lvert\mathcal{C}\rvert\lvert\mathcal{I}\rvert + + p\lvert\mathcal{U}\rvert^2 + \lvert\mathcal{U}\rvert\right)$	$O(\lvert\mathcal{U}\rvert^2 + \lvert\mathcal{H}_0\rvert)$	$O\Big(\lvert\mathcal{I}\rvert + \text{avg}_u \sum_{v\in\Gamma(u)} \lvert I_v\rvert + + \text{avg}_u \lvert c(u)\rvert + \text{avg}_u \lvert E_{c(u)}\rvert + + \text{avg}_u \sum_{v\in c(u)} \lvert I_v\rvert\Big)$
	kNN bandit	$O(\lvert\mathcal{U}\rvert^2)$	$O(\lvert\mathcal{U}\rvert^2)$	$O(\lvert\mathcal{U}\rvert b \log k + k \cdot m_u + m_i)$
Myopic	UB kNN	$O(\lvert\mathcal{U}\rvert^2)$	$O(\lvert\mathcal{U}\rvert^2)$	$O\left(\lvert\mathcal{U}\rvert \log(k) + k m_u + m_i\right)$
	iMF	$O\left(k^2 \cdot (\lvert\mathcal{U}\rvert + \lvert\mathcal{I}\rvert)\right)$	$O\left(c \cdot k^2 \cdot (\lvert\mathcal{H}_0\rvert + + k \cdot (\lvert\mathcal{U}\rvert + \lvert\mathcal{I}\rvert)\right)$	$O\left(\lvert\mathcal{I}\rvert \cdot k) + c(k^2 \cdot \lvert\mathcal{H}_0\rvert + + k^3 \cdot \lvert\mathcal{U}\rvert + k^3 \cdot \lvert\mathcal{I}\rvert)\right)$
	Popularity	$O(\lvert\mathcal{I}\rvert)$	$O(\lvert\mathcal{H}_0\rvert)$	$O(\lvert\mathcal{I}\rvert)$
	Random	$O(1)$	$O(1)$	$O(1)$

group estimate the value of each item as its average rating in the system, i.e.:

$$v(i, x_t) = v(i) = \frac{1}{|\mathcal{U}_i(t)|} \sum_{u \in \mathcal{U}_i(t)} r_u(i) \tag{10.1}$$

where $\mathcal{U}_i(t)$ represents the set of users in the system who have provided a rating to item i at time t. As they do not use any context (beyond reducing the available items to each user depending on the previously consumed ones), these approaches are an example of not personalized bandit algorithms for recommendation. Indeed, they represent stochastic approximations to the classical average rating algorithm (Cañamares and Castells, 2018, Cremonesi et al., 2010).

There are multiple bandit strategies which we can apply here (Auer et al., 2002, Lattimore and Szepesvári, 2020, Sutton and Barto, 2018). For all of them, the training and update methods are very similar: they just need to store the $v(i)$ values for each item $i \in \mathcal{I}$. The point where they differ is in the arm selection (i.e. in the recommendation step). We focus here on two of the most well-known and effective bandit policies: ε-greedy (Sutton and Barto, 2018) and Thompson sampling (Chapelle and Li, 2011).

Although simple, these approaches act as the basis of more complex and personalized recommendation algorithms. Therefore, as they are first introduced here, we provide a general algorithmic description of the bandit algorithms, before focusing on the particularities of each of them as simple and non personalized algorithms.

ε-greedy

The ε-greedy (Sutton and Barto, 2018) family of bandit algorithms manages the exploration-exploitation dilemma by fixing a fraction of time $\varepsilon \in [0, 1]$ the policy spends exploring "suboptimal" actions. With probability ε, the algorithm selects an action at random. If the algorithm does not explore, it takes the arm $a \in \mathcal{A}$ that maximizes the value $v(a, x_t)$. When $\varepsilon = 0$m, the algorithm always greedily selects the option that has the greatest value estimation $v(a, x_t)$, whereas, when $\varepsilon = 1$ this method is equivalent to selecting actions at random. In the case of non personalized recommendation, $\varepsilon = 0$ would be then equivalent to selecting the algorithm with the greater average rating.

The main advantages of this approach are its simplicity and fastness. However, it has a limitation on the long term: as we execute an algorithm, we obtain more and more information, making exploration less necessary for it to work. However, ε-greedy performs the same amount of exploration over time – something that might lead to a lose of accuracy if we keep it running for long.

Complexity

Algorithm 3: ε-greedy arm selection procedure.

Data:
- x_t Context
- $\varepsilon \in [0, 1]$ Probability of exploration
- t Iteration number

Result:
- $a_t \in \mathcal{A}$ action

begin
- rnd $=$ random$(0, 1)$ `// Determine whether we explore or exploit`
- `// Explore`
- **if** *rnd* $< \epsilon$ **then**
  - $i_t \leftarrow$ random$(\mathcal{A})$
- `// Exploit`
- **else**
  - $a_t \leftarrow \arg\max_{a \in \mathcal{A}} v(a, x_t)$
- **return** a_t

As a not personalized recommendation approach, we can estimate the time and memory complexity as follows: first, if we receive some training data, $\mathcal{H}_0$, it is enough to run over them once to find the average rating values for each item. Then, each time we receive a rating, we only have to update the value of the rated item, which can be done in $O(1)$. Finally, we have to estimate the time needed to select a single item: when we select it randomly, the cost is $O(1)$; when we select an item greedily, we have to run over the set of items $O(|\mathcal{I}|)$, thus making the complexity $O(\varepsilon + (1 - \varepsilon)|\mathcal{I}|)$. In terms of memory, it is enough to store the average ratings for each item (i.e. its cost is $O(|\mathcal{I}|)$).

Thompson sampling

Thompson sampling (Chapelle and Li, 2011) is a stochastic multi-armed bandit algorithm based on Bayesian statistics. Starting from a set of past interactions $\mathcal{H}_T = \{(a_t, x_t, r_t)\}_{t=1}^T$, where $a_t \in \mathcal{A}$ represents the action taken at time t, r_t its corresponding reward, and x_t the context, Thompson sampling assumes that we can model the likelihood of the reward using a parametric probability distribution $p(r|a, x, \theta)$ where θ represents the set of parameters of the distribution.

The reward is then a stochastic function, which depends on the action taken, the context and the parameters of the distribution, θ. If we knew the correct θ values for each arm distribution, the optimal strategy would just consist in choosing the arm $a \in \mathcal{A}$ maximizing the expected reward – i.e. $\arg\max_{a \in \mathcal{A}} \mathbb{E}[r|a, x, \theta]$. Although we can obtain an estimation from the data, the real value of θ is, however, unknown. So, the best we can do is select an arm a according to the probability that it is optimal. This would be equivalent to selecting the arm maximizing the following equation:

$$\int \mathbb{1}\left[\mathbb{E}(r|a, x, \theta) = \max_{a' \in \mathcal{A}} \mathbb{E}(r|a', x, \theta)\right] p(\theta|\mathcal{H}_T) d\theta \tag{10.2}$$

Fortunately, despite the complexity of the previous integral, its explicit computation is not necessary. Instead of it, we can use the Bayes' rule to estimate the posterior distribution of the θ parameter: given a suitable prior, $p(\theta)$, the posterior distribution of the parameter is defined as $p(\theta|\mathcal{H}_t) \propto \prod_{t=1}^T p(r_i|a_i, x_i, \theta)p(\theta)$. Once this distribution has been found, it is possible to sample a reasonable value for the parameters, $\hat{\theta}$, according to our observations $\mathcal{H}_T$. The arm a is then selected as $\arg\max_{a \in \mathcal{A}} \mathbb{E}[r|a, x, \hat{\theta}]$. This is illustrated in Algorithm 4.

When we want to apply Thompson sampling as a non personalized recommendation approach, it is common to consider that the different arms are independent (i.e. that the ratings provided to the items are independent from each other), and that the ratings are binary ($r_u(i) = 1$ if the user interacts / likes the item, $r_u(i) = 0$ otherwise). Considering that, for each item, we consider that its rewards are determined by a Bernoulli distribution, where an item is relevant for a user (i.e. $r_u(i) = 1$) with probability $p_i = v(i)$.

Algorithm 4: Thompson sampling selection strategy

Data:
x_t Context
t Iteration number
Result:
$a_t \in \mathcal{A}$ action
begin
Draw $\hat{\theta}_t \sim P(\theta|\mathcal{H}_t, x_t)$
$a_t \leftarrow \arg\max_{a \in \mathcal{A}} \mathbb{E}[r|a, x_t, \hat{\theta}_t]$

If we consider a Beta distribution with parameters α_0 and β_0 as the prior probability of the p_i parameter (i.e. $p(p_i) \sim \text{Beta}(\alpha_0, \beta_0)$), its posterior distribution is another Beta distribution with parameters α and β, where α represents the number of hits (i.e. the number of times a positive rating has been given to the item), and β the number of failures (i.e. the number of times no rating or negative rating has been obtained):

$$\alpha = \alpha_0 + \sum_{u \in \mathcal{U}_i(t)} r_u(i) \tag{10.3}$$

$$\beta = \beta_0 + \sum_{u \in \mathcal{U}_i(t)} (1 - r_u(i)) \tag{10.4}$$

i.e. $\hat{p}_i \sim B(\alpha, \beta)$. Then, as, the p parameter of the Bernoulli distribution is the same as its average value, we just have to select the item i_t maximizing:

$$i_t = \arg\max_{i \in \mathcal{I} \setminus \mathcal{I}_u(t)} \mathbb{E}[r_u(i)|i, x_t, \hat{p}_i] = \arg\max_{i \in \mathcal{I} \setminus \mathcal{I}_u(t)} \hat{p}_i \tag{10.5}$$

The main advantage of this approach is that it is able to manage the balance between exploration and exploitation using the sampling strategy: we illustrate an example of this in 10.2. In that example, we plot the evolution of the posterior distributions of a single arm. The rewards obtained by the arm are obtained from a Bernoulli distribution with $p = 0.7$. The figure starts from a Beta distribution $\text{Beta}(1, 1)$ and, as time passes, stretches more and more around the real value of p. As this occurs with all the arms in the bandit as the amount of available data grows, we observe that, differently from ε-greedy, exploration decreases over time.

On the other hand, Thompson sampling has many drawbacks: first, we need to model the value of the arm after a parametric distribution (which is not always possible); second, given a distribution, it might not be easy to define a posterior distribution for its parameters; and third, sampling from such distribution might be computationally expensive.

Complexity

When applied as a not personalized bandit approach for item recommendation, we obtain a similar computational and memory cost to that of ε-greedy: we find two differences: first, recommendation time changes to $O(b \cdot |\mathcal{I}|)$, with b being the average time we need to sample from the posterior Beta distribution; second, instead of one value for each arm, we need to store two. Nonetheless, this does not change the space complexity, which remains $O(|\mathcal{I}|)$.

10.2.2 Personalized methods

In addition to the not personalized methods, several other methods have been considered which make use of MAB for the development of personalized recommendation algorithms (Bresler et al., 2014, Gentile et al., 2014, Kawale et al., 2015, Li et al., 2010, 2016, Zhao et al., 2013). From all of them, only a few of them consider a pure collaborative filtering perspective, whereas most of them take contextual variables for either the users or the items as a basis.

In this work, we consider two examples of personalized multi-armed bandits. The first example, similarly to the method we introduce in Section 10.3, adapts a classical recommendation algorithm into a bandit setting.

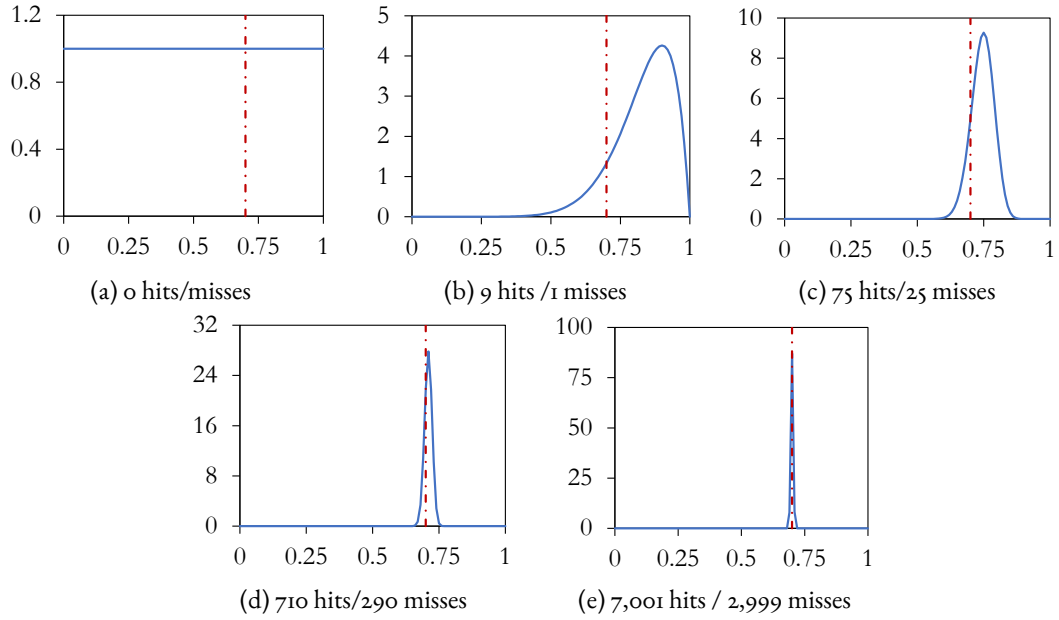


Figure 10.2: Thompson sampling example for a Bernoulli arm with $p = 0.7$. Blue line represents the probability density function of the posterior distribution, and red dotted line the actual p value.

In particular, it reformulates the probabilistic matrix factorization (Salakhutdinov and Mnih, 2008) algorithm. We shall name this algorithm (Zhao et al., 2013) as interactive probabilistic matrix factorization (InterPMF). The second example, named "clustering of bandits" (CLUB, Gentile et al. (2014)) belongs to a family of bandit-based algorithms which divide users (and sometimes items) in the system according to their preferences.

Interactive probabilistic matrix factorization

As we state earlier, a possible way to develop interactive recommendation approaches involves the adaptation of classical algorithms to a reinforcement learning setting (Bresler et al., 2014, Kawale et al., 2015, Zhao et al., 2013). The first (and the most relevant) algorithm in this research line is the proposal by Zhao et al. (2013), who develop a linear contextual MAB algorithm based on the probabilistic matrix factorization (PMF) collaborative filtering algorithm. We shall name this bandit as interactive probabilistic matrix factorization (InterPMF). We should note that, although relevant, as we shall later see in Section 10.4, this approach does not work well with our experimental setting because its design requires a fair amount of training data – in contrast with our extreme cold start setting without any data. Nonetheless, since it poses as an example of adapted algorithm, we provide here a full description of the algorithm, and we later include it in our experiments.

Probabilistic matrix factorization (PMF) (Salakhutdinov and Mnih, 2007, 2008) algorithms, as their name indicates, are a group of matrix factorization techniques considering that we can approximate the rating matrix R as the product of two matrices: a user matrix $X \in \mathbb{R}^{k \times |\mathcal{U}|}$ and an item matrix $Y \in \mathbb{R}^{k \times |\mathcal{I}|}$, with each row representing a user or item in a joint space of dimension k. Differently from other methods, such as implicit matrix factorization (Hu et al., 2008, Pilászy et al., 2010), PMF assumes that, given such matrices, the ratings follow a given conditional probability distribution, i.e. we can define $p(r_u(i)|x_u^T y_i)$ where x_u is the latent vector for user u and y_i is the latent vector for item i. Considering this, it is possible to estimate the latent factors using Bayesian statistics.

The most common assumption – and the one Zhao et al. (2013) make – is to take the distribution of the ratings as Gaussian distribution, i.e.

$$p\left(r_u(i)|x_u^T y_i\right) = \mathcal{N}\left(r_u(i)|x_u^T \cdot y_i, \sigma^2\right) \tag{10.6}$$

where $\mathcal{N}(x|\mu, \sigma^2)$ represents a normal distribution with μ as its mean and $\sigma^2 > 0$ as its variance. Under this assumption, and by taking (multivariate) Gaussian priors for the user and item latent vectors, we can estimate

the different latent vectors for the users as as:

$$\hat{x}_u = \mathbb{E}\left[x_u | R, Y, \sigma^2, \sigma_u^2, \sigma_i^2\right] = \left(D_u D_u^T + \frac{\sigma^2}{\sigma_u^2} I\right)^{-1} D_u^T r_u \tag{10.7}$$

where $I \in \mathbb{R}^{k \times k}$ is the unit matrix, σ_u^2 is the variance of each user factor, σ_i^2 is the variance of the item ones, $D_u \in \mathbb{R}^{|\mathcal{I}_u(t)| \times k}$ is a matrix which contains, as rows, the latent factors of the items rated by the user u, and $r_u \in \mathbb{R}^{|\mathcal{I}_u(t)|}$ contains the ratings given by u to those items. A similar solution is found for the item factors:

$$\hat{y}_i = \mathbb{E}\left[y_i | R, X, \sigma^2, \sigma_u^2, \sigma_i^2\right] = \left(E_i E_i^T + \frac{\sigma^2}{\sigma_i^2} I\right)^{-1} E_i^T r_i \tag{10.8}$$

where E_i and r_i are the equivalent of D_u and r_u for computing $\hat{y}_i$. The recommendation score for an user-item pair is just the dot product of $\hat{x}_u$ and $\hat{y}_i$.

Taking the history of the system $\mathcal{H}_0$, and, similarly to implicit matrix factorization (Hu et al., 2008), InterPMF iteratively applies the previous equations to find initial estimations for the user and item vectors. It assumes that, if enough data is provided as input for training the algorithm, the item factors remain stable (they do not change much as we receive new ratings), so it suffices to fix those values, and focus on selecting candidate users and updating the user vectors. For this, it considers several contextual bandit strategies, such as ε-greedy (Sutton and Barto, 2018), upper confidence bound (UCB) (Auer et al., 2002, Li et al., 2010) or Thompson sampling (Chapelle and Li, 2011). However, this assumption makes the algorithm need enough training data to learn reasonable item factors – therefore losing utility in cold start situations, or for a scenario where a new item arrives.

Among the several bandit strategies, in this thesis, we have just focused on one of them, which we describe next: the ε-greedy approach. Given a target user u, following the previously described algorithm, with probability ε, it selects an item at random and, otherwise, it selects the item $i \in \mathcal{I} \setminus \mathcal{I}_u(t)$ maximizing

$$f_u(i) = \hat{x}_u^T \cdot \hat{y}_i \tag{10.9}$$

Once the system receives a rating, $r_u(i_t)$, the $\hat{x}_u$ vector for the target user is updated (for the rest of users, it remains the same). For this, it is enough to consider that we can separate it in two matrices:

$$\hat{x}_{u,t} = A_{u,t}^{-1} b_{u,t}$$

where

$$A_{u,t} = D_u D_u^T + \frac{\sigma^2}{\sigma_u^2} I = \frac{\sigma^2}{\sigma_u^2} I + \sum_{i \in \mathcal{I}_u(t)} \hat{y}_{i_t} \hat{y}_{i_t}^T = A_{u,t-1} + \hat{y}_{i_t} \hat{y}_{i_t}^T \tag{10.10}$$

$$b_{u,t} = D_u^T r_u = \sum_{i \in \mathcal{U}_i(t)} \hat{y}_i \cdot r_u(i) = b_{u,t-1} + r_u(i_t) \cdot \hat{y}_{i_t} \tag{10.11}$$

Here, we add the time index t to remark the incremental nature of the algorithm.

Complexity

The training procedure that allow us to obtain the initial values for user and item factors is quite similar to the one for the implicit matrix factorization method proposed by Hu et al. (2008) which we use in previous chapters. Consequently, the time complexity for such approach is similar: $O\left(c\left(k^2|\mathcal{H}_0| + k^3|\mathcal{U}| + k^3|\mathcal{I}|\right)\right)$. The same occurs with the recommendation time: as both compute the recommendation scores as the product of user and item latent factors, they need $O(k|\mathcal{I}|)$ time for selecting a single item. The point where this method takes advantage is in the time needed to update it: in iMF, it is necessary to retrain the whole model to achieve lossless updates; here, however, since item features are fixed, we only have to update the user feature of the target user, which can be done in $O(k^3)$. Memory consumption is also similar: the greatest amount of memory is needed at training time, when we need to store $O\left(k^2|\mathcal{U}| + k^2|\mathcal{I}|\right)$ values (all the D_u and E_i matrices for users and items).

Clustering of bandits

The CLUstering of Bandits (CLUB) algorithm (Gentile et al., 2014) is an interactive recommendation algorithm working under the hypothesis that a partition of the users in the system $\mathcal{C}$ can be found, so users within the same clusters have similar tastes, differentiating them from people in others. It represents the items as feature vectors $y_i \in \mathbb{R}^d$ where d is the number of features, and associates each group of users $c \in \mathcal{C}$ to another vector $w_c \in \mathbb{R}^d$ expressing the interest of the different users on such features. Then, it expects the rating of a user u to an item i (i.e. given the user u as context, the value of the arm) to be:

$$v(i,u) = w_{c(u)} \cdot y_i \tag{10.12}$$

where $c(u) \in \mathcal{C}$ is the cluster user u belongs to. To maximize the gain, the algorithm has to learn the cluster structure and the vectors associated to the clusters.

In order to learn both, after producing t recommendations, the algorithm stores a user vector $x_{u,t} \in \mathbb{R}^d$ showing the interest of the user u on each feature according to the ratings she has produced to the system. This vector is computed as the product of a matrix $M_u^{-1}, t \in \mathbb{R}^{d\times d}$, representing the inverse correlation matrix for the user, computed as:

$$M_{u,t} = I + \sum_{i\in\mathbb{I}_u(t)} y_i \cdot y_i^T \tag{10.13}$$

where I is the unit matrix, and a vector $b_{u,t}$ defined as:

$$b_{u,t} = \sum_{i\in\mathbb{I}_u(t)} r_u(i) \cdot y_i \tag{10.14}$$

Taking the user u_t and the cluster partition at time t, $\mathcal{C}_t$, the algorithm first identifies the cluster $c_t(u_t)$ user u_t belongs to. For simplicity, we denote this cluster as c_t. Then, the algorithm finds an approximation of the vector w_{c_t}, we shall denote as $\hat{w}_{c_t,t}$. Similarly to how we do for each individual user, this vector is estimated as the product of an inverse correlation matrix $\hat{M}_{c_t,t}$ and a vector $\hat{b}_{c_t,t}$ aggregating the ones for the individual users in the cluster:

$$\hat{w}_{c_t,t} = \hat{M}_{c_t,t}^{-1} \cdot \hat{b}_{c_t,t} \tag{10.15}$$

$$\hat{M}_{c_t,t} = I + \sum_{v\in c_t} (M_{v,t} - I) \tag{10.16}$$

$$\hat{b}_{c_t,t} = \sum_{v\in c_t} b_{v,t} \tag{10.17}$$

Considering that estimation, the selected arm is the one maximizing

$$i_t = \arg\max_{i\in\mathcal{I}\setminus\mathcal{I}_u} \hat{w}_{c_t,t} \cdot y_i + \alpha\sqrt{y_i^T \cdot \hat{M}_{c_t,t}^{-1} \cdot y_i \log(t+1)} \tag{10.18}$$

where the right term represents an upper confidence bound (Auer et al., 2002) for the estimated $\hat{w}_{c_t,t}$ for favouring the exploration of the approach and $\alpha > 0$ is a parameter that manages the exploration degree.

Finally, the clusters are determined with the help of a user graph $\mathcal{G}_t = \langle \mathcal{U}, E_t \rangle$, in which an edge exists between two users u, v if their vectors $x_{u,t}, x_{v,t}$ are similar enough. To check this, at time t, we check, for each neighbor v of the target user u_t if the pair satisfies the following condition:

$$\|x_{u_t,t} - x_{v,t}\|_2 > \gamma \left(\sqrt{\frac{1 + \log\left(1 + |\mathcal{I}_{u_t}(t)|\right)}{1 + |\mathcal{I}_{u_t}(t)|}} + \sqrt{\frac{1 + \log\left(1 + |\mathcal{I}_v(t)|\right)}{1 + |\mathcal{I}_v(t)|}} \right) \tag{10.19}$$

where $\gamma > 0$ is a free parameter that establishes how difficult is to do it. If the previous inequality is true, edge (u_t, v) is removed from the graph (since we have evidence that the distance between both vectors is significantly large). The set of clusters is just the connected components of the network.

At an extreme cold start situation, with no known information, the starting graph would be a complete graph, where all the nodes are connected to each other. However, since storing and managing all the possible

links in a network is very costly ($O(|\mathcal{U}|^2)$), a sparser Erdös-Renyi network (Erdös and Rényi, 1959) among the users in the system is considered instead, where a probability p that a link between two nodes exists is considered. Gentile et al. (2014) show that, if we take p large enough, there is a great probability that the underlying clusters are still connected.

The previous algorithm can be applied in a collaborative filtering manner if, instead of representing the items according to some features, we do it as $|\mathcal{I}|$-dimensional unit vectors, where all coordinates are equal to 0 except for the i-th one. This is the way we use this approach in our experiments, to prevent the necessity of features. Under this perspective, predicted scores for each cluster represent the average rating of the items over the set of users in the cluster.

Complexity

We analyze the complexity of the algorithm for the collaborative filtering variant of the approach. We start by studying the time needed to generate a recommendation. If we have the clusters and cluster vectors computed, we can retrieve the score for each item in $O(1)$, leading to an overall time of $O(|\mathcal{I}|)$ to select one of them. Then, when we receive the rating for the item, we update the different values. The cost for this update is highly variable, and depends on many factors:

When we receive a rating, there are two steps which always have to be done: updating the weight vector for the target user and check whether any edge has to be deleted from the user graph. The first step can be done in $O(1)$ considering that, in the collaborative filtering case

$$x_{u,i} = \frac{r_u(i)}{1 + \mathbb{1}_{r_u(i)\neq\emptyset}(u,i)} \tag{10.20}$$

Also, if we store these w_u vectors, we need $O(\text{avg}_{u\in\mathcal{U}} \sum_{v\in\Gamma(u)} \cdot |I_u|)$ time to find the distances between the target user and her neighbors.

Differences arise depending on a) whether an edge is deleted or not and b) whether new clusters are found or not. If no edge is deleted, the clusters remain the same, so it is only necessary to update the values for the $\hat{w}_{c(u)}$ vector. As its i-th coordinate is defined as

$$\hat{w}_{c(u),i} = \frac{\sum_{u\in c(u)} r_u(i)}{1 + \sum_{u\in c(u)} \mathbb{1}_{r_u(i)\neq\emptyset}(u,i)} \tag{10.21}$$

if we store the numerator and denominator values, we can compute it in $O(1)$ time (otherwise, it would be $O(\text{avg}_{u\in\mathcal{U}} |c(u)|)$ time). However, if an edge is deleted, we have to check whether new clusters are found or not. This can be reduced to a breadth-first search over the (at least previously) connected component of the graph containing the target user u, i.e. the subgraph $\mathcal{G}_{c(u)} = \langle c(u), E_{c(u)}\rangle$, where $E_{c(u)}$ is the set of links between users in the cluster. We can obtain the newly found clusters with cost $O\left(\text{avg}_{u\in\mathcal{U}} |c(u)| + \text{avg}_{u\in\mathcal{U}} |E_{c(u)}|\right)$. If the cluster is not divided, the cost of updating the cluster vector is, again $O(1)$. If it is divided, we have to recompute the weight vector for each of the clusters, which has an average computational cost of $O\left(\text{avg}_{u\in\mathcal{U}} \sum_{v\in c(u)} |\mathcal{I}_v|\right)$.

As for training, we first have to initialize the Erdös graph, which takes $O(|\mathcal{U}|^2)$ time. Then, we have to find the user vectors w_u, which can be done in $O(|\mathcal{H}_0|)$ considering Equation (10.20). And, finally, we have to delete edges, find the clusters and find their vectors, which is done in $O(|\mathcal{U}| + p|\mathcal{U}|^2 + \mathcal{H}_0|)$ (where $p \in (0,1)$ is the probability of link formation in the Erdös-Renyi, and $p|\mathcal{U}|^2$ is (approximately) the initial number of edges in the network). This leads to an overall time complexity of $O(|\mathcal{U}|^2 + |\mathcal{H}_0|)$.

As of space complexity, we have to store the user graph (around $O(p|\mathcal{U}|^2)$ values at the beginning – less later), the relation between clusters and users ($O(|\mathcal{U}|$ since each user belongs to a single cluster), the cluster vectors ($O(|\mathcal{C}||\mathcal{I}|)$ and the user vectors. For this last element, it is enough to store the values for those items who each user have rated, so the memory requirement is $O(|\mathcal{H}|)$.

10.3 Nearest-neighbor bandit

As novel interactive recommendation approach, we develop an elaboration of neighbor-based collaborative filtering. The algorithm we explore can be understood as a variant of the user-based nearest-neighbors scheme

where the selection of the user's neighborhood is stochastic. Differently from previous interactive recommendation algorithms, like the ones introduced in Section 10.2, where the arms of the bandit are represented by the candidate items of the recommendation, here, the potential neighbors of the target user take that place.

Our proposal is the following: given the target user of the recommendation, u, acting as the context, we consider any other person in the system as her potential neighbor. Using a stochastic MAB policy, we select one of them, $v \in \mathcal{U}$, as the preferred neighbor for the target user. Afterwards, the selected neighbor chooses an item $i \in \mathcal{I} \setminus \mathcal{I}_u(t)$ and recommends it to the target user. We thus need to define two elements: first, the value function for each neighbor, and, second, the strategy for the neighbor for selecting an item.

We start by the first one: in our approach, we consider as the value of a user v as potential neighbor (i.e. as the selected arm of the bandit) of each user is the conditional probability that the target user u being pleased by an item that user v liked (which we denote as $P(u|v)$). Considering binary ratings, an estimation of this is:

$$p_t(u|v) = \frac{\sum_{i \in \mathcal{I}_u(t) \cap \mathcal{I}_v(t)} r_u(i) r_v(i)}{\sum_{i \in \mathcal{I}_v(t)} r_v(i)} \tag{10.22}$$

Any simple and context-less MAB policy (Lattimore and Szepesvári, 2020) might be applied here to select a suitable best neighbor for the target user. We have nonetheless focused on the development of a neighborhood selection strategy based on Thompson sampling (Chapelle and Li, 2011). Similar to how we do for the non personalized recommendation Thompson sampling strategy in Section 10.2.1, we make three assumptions: first, that ratings are binary ($r_u(i) = 1$ if the item i is relevant to u, $r_u(i) = 0$ otherwise); second, that the probability of drawing different neighbors are independent from each other; third, and last, the probability $P(u|v)$ follows a Bernoulli distribution. Then, at each time point t, we draw an estimate $\hat{p}_t(u|v)$ of the unknown average value $P(u|v)$. This value is sampled from a Beta posterior with parameters $\alpha = \alpha_t + \alpha_0$ and $\beta = \beta_t + \beta_0$, where their value for v given the value u are:

$$\alpha_t(v|u) = \sum_{i \in \mathcal{I}_u(t) \cap \mathcal{I}_v(t)} r_v(i) r_u(i) \tag{10.23}$$

$$\beta_t(v|u) = \sum_{i \in \mathcal{I}_v(t)} r_v(i) \left(1 - r_u(i)\right) = n_t(v) - \alpha_t(v|u) \tag{10.24}$$

where α_0, β_0 represent the parameters of the prior Beta distribution for each neighbor (arm),

$$n_t(v) = \sum_{i \in \mathcal{I}_v(t)} r_v(i) \tag{10.25}$$

is the number of items that user v has liked, and we take $r_u(i) = 0$ if $r_u(i) = \emptyset$ in the formulation of $\beta_t(v|u)$. Analyzing this, we can observe that $\alpha_t(v|u)$ is just the number of items that they like in common, and $\beta_t(v|u)$ is the number of items v likes, but u does not (or does not know about them). We should note here that $\alpha_t(v|u) = \alpha_t(u|v)$, so it is enough to store one of them.

Once that we have selected the neighbor v maximizing $p_t(u|v)$, we ask v to suggest an item to the target user. User v picks every time the item they like the most, as governed by a certain distribution $P(i|v)$. Again, as this distribution is unknown, we can estimate it by:

$$p_t(i|v) = \frac{r_v(i)}{\sum_{j \in \mathcal{I}_v(t)} r_v(j)} \propto r_v(i) \tag{10.26}$$

i.e. we take the item with the greatest rating value. Since we use binary ratings, this just means recommending one item that v has liked. Ties are broken at random, i.e. if v has enjoyed multiple items, one of them is picked for recommendation with no further criteria.

Finally, we update the arms of our neighbor selection bandit each time a reward is produced, i.e. each time the user u_t provides a rating $r_{u_t}(i_t)$ for the recommended item i_t. In this case, we only have to update the $\alpha_t(v|u_t)$ values for those users $v \in \mathcal{U}$ who have previously rated i_t and the $n_t(u_t)$ value for the target user. We update this values as:

$$\alpha_{t+1}(v|u_t) = \alpha_{t+1}(u_t|v) = \alpha_t(v|u_t) + r_{u_t}(i_t) r_v(i_t) \tag{10.27}$$

$$n_{t+1}(u_t) = n_t(u_t) + r_{u_t}(i_t) \tag{10.28}$$

Table 10.3: Relation between interactive recommendation algorithms and multi-armed bandits.

Algorithm	Arms	Reward	Estimated arm value	Context
Not personalized	Items	Rating	Metric (e.g. CTR)	Target user
ICF	Items	Rating	Expected rating $x_u^T \cdot y_i$	Target user Item factors y_i
CLUB	Items	Rating	Expected rating for the user cluster $w_{c(u)}^T \cdot y_i$	Target user Item features q_i
CLUB (CF)	Items	Rating	Average rating (within cluster $c(u)$)	Target user
kNN bandit	Neighbors	Rating	Cond. preference $P(u\|v)$	Target user

The rest of values do not need updating: the β_t values can be computed from the α_t and n_t values, and the rest of users are not involved in the recommendation of the item.

Generalization to k neighbors

The approach we have just described only considers the selection of a single neighbor of the target user. Nevertheless, in recommendation scenarios, it is common to consider much larger neighborhoods. The selection of a Thompson sampling strategy makes easy to extend and generalize our recommendation policy to deal with k neighbors instead of one. Our estimations of $p_t(v|u)$ define a ranking of users according to their probability of being good neighbors of the target user u, so, instead of just picking one, we can take the top k values in that ranking. This is equivalent to using a multiple play bandit that selects many arms at once (Louëdec et al., 2015).

With these new neighborhood, which we denote as $\mathcal{N}_k^t(u)$, we can generate the scores for the different items, but, instead of picking items randomly, we use a weighted sum, similar to the one in the basic and myopic user-based nearest-neighbor algorithm:

$$i_t = \arg\max_{i \in \mathcal{I} \setminus \mathcal{I}_u(t)} \sum_{v \in \mathcal{N}_k^t(u)} p_t(u|v) r_v(i) \tag{10.29}$$

It is easy to see that the original algorithm is retrieved by taking $k = 1$.

Complexity

Similarly to how we have done with the algorithms in Section 10.2, we analyze the complexity of our bandit algorithm proposal. Its cost is included in the comparison included in Table 10.2, and, following the previous analyses, we decouple it in four parts: training, recommendation and update time, and memory consumption. As we see next, we have devised an algorithm with similar complexity to myopic user-based nearest-neighbors approaches (Ning et al., 2015).

First, we analyze the amount of memory needed to execute the algorithm. The major cost comes from the storage of the α and n values for each pair of users. In order to save some space, we can take into account that the matrix is sparse to only store those α with value greater than 0, and we can notice that $\alpha_t(u|v) = \alpha_t(v|u)$ for all pairs. Nonetheless, we estimate the overall cost as $O(|\mathcal{U}|^2)$, similar to other user-based kNN approaches which store similarity values.

Then, we study the time complexities. First, we analyze the training one. In this method, we have to initialize both the initial α_0 and n_0 values for each pair of users. Since n_0 just depends on individual users, it would be enough to run once over each rating (i.e. in $O(|\mathcal{H}_0|)$ time). However, since α_0 is different for each user pair, the cost is superior. A good solution for this is to run over all the items with ratings, and, for each of them, update the values for the users which have rated them. Assuming all items have (at least) one rating, that would need $O(\sum_{i \in \mathcal{I}} |\mathcal{U}_i(0)|^2)$ time. Although this seems costly, we should note that because of the sparsity of the rating matrix, $|\mathcal{U}_i|$ should be quite small for most items. As for the recommendation time, we consider that we have all the α_t and n_t values stored in memory. Then, we divide the cost in two steps: finding the neighborhood and generating the recommendation scores. If we use a heap to sort the neighbors, the first part can be done in $O(|\mathcal{U}| \cdot b \log(k))$ (with b representing the cost of sampling from a posterior distribution). Then, we would

Table 10.4: Dataset statistics

Dataset	# Users	# Items	# Ratings	# Relev. Ratings	Density (%)	Relev. density (%)
Foursquare New York	1,083	38,333	91,204	91,204	0.2200	0.2197
Foursquare Tokyo	2,293	61,858	211,955	211,955	0.1494	0.1494
Movielens 1M	6,040	3,706	1,000,209	575,281	4.4684	2.5700
Twitter 1-month	9,511	9,511	650,937	650,937	0.7197	0.7197
Twitter 200-tweets	9,253	9,253	475,608	475,608	0.5556	0.5556

need $O(km_u)$ time to generate the ratings for all the candidate items. In the case $k = 1$, the total cost of both parts would be reduced to $O(|\mathcal{U}|)$. Finally, once we receive a positive rating, we have to update the α_t and n_t values. The only n_t value which has to be updated is the one corresponding to the target user (since it is the only one gaining a new positive rating). However, as we have to update the α_t value for all users who have previously rated the candidate item positively, the cost grows up to $O(m_i)$, with m_i representing the average number of ratings per item.

10.4 Experiments

10.4.1 Experimental procedure

We test our kNN bandit over offline data by comparing its empirical effectiveness with respect to other bandit and non stochastic approaches. Our evaluation procedure consists in the simulation of a recommendation feedback loop as the one we illustrate in Algorithm 2, but recommending a single item at each time step and selecting the user u_t randomly from the pool of target users. Each iteration, the recommendation algorithm is updated using the feedback provided to the suggested items.

As an evaluation metric, we use the global cumulative recall of the system at time t, i.e. the fraction of positive ratings in the dataset that the recommendation approach has discovered at time t (transversal to all users). If we denote by $\mathcal{H}_t^+$ the set of relevant ratings in the history, $\mathcal{H}_t^+ = \{(u, i, r_u(i)) \in \mathcal{H}_t \setminus \mathcal{H}_0 | r_u(i) \text{ is relevant}\}$, we can express the cumulative recall of the system as:

$$\text{Recall(t)} = \frac{|\mathcal{H}_t^+|}{|\mathcal{H}_\infty^+|} = \frac{|\{(u, i, r_u(i)) \in \mathcal{H}_t \setminus \mathcal{H}_0 | r_u(i) \text{ is relevant}\}|}{|\{(u, i, r_u(i)) \in \mathcal{H}_\infty \setminus \mathcal{H}_0 | r_u(i) \text{ is relevant}\}|} \tag{10.30}$$

where $\mathcal{H}_\infty$ is, in the offline dataset, the whole set of ratings.

For each offline dataset, we consider an experiment where we start in an extreme cold start situation, where we do not have any training ratings (i.e. $\mathcal{H}_0 = \emptyset$). We then grow the set of observations by using the dataset to simulate user feedback – each time an item is suggested, we observe whether there is a rating for it in the data, and, if there is, we provide it to the recommender as input for updating it. For each algorithm and dataset, we run the simulation until around 500 recommendations per user are generated, and we observe the data. In this case, since we do not have any training data, we select the hyperparameters by taking the variants with the greater cumulative recall value after 50 recommendations (similarly to how Gentile et al. (2014) do for tuning the parameters). Optimal parameters for the experiments are reported in Table C.8 in Appendix C.

10.4.2 Setup

Data

We run our experiments on several offline rating datasets. As we state earlier, in this chapter we do not only study the contact recommendation task, but also consider other domains of application for our bandit algorithms. In particular, we consider two: movies and venue recommendation. For all the recommendation algorithms, we consider binary ratings $r_u(i) \in \{0, 1\}$ in all datasets, regardless of their nature, where $r_u(i) = 1$ means that the item i is relevant for user u, and 0 means the opposite. We use the following datasets in our experiments:

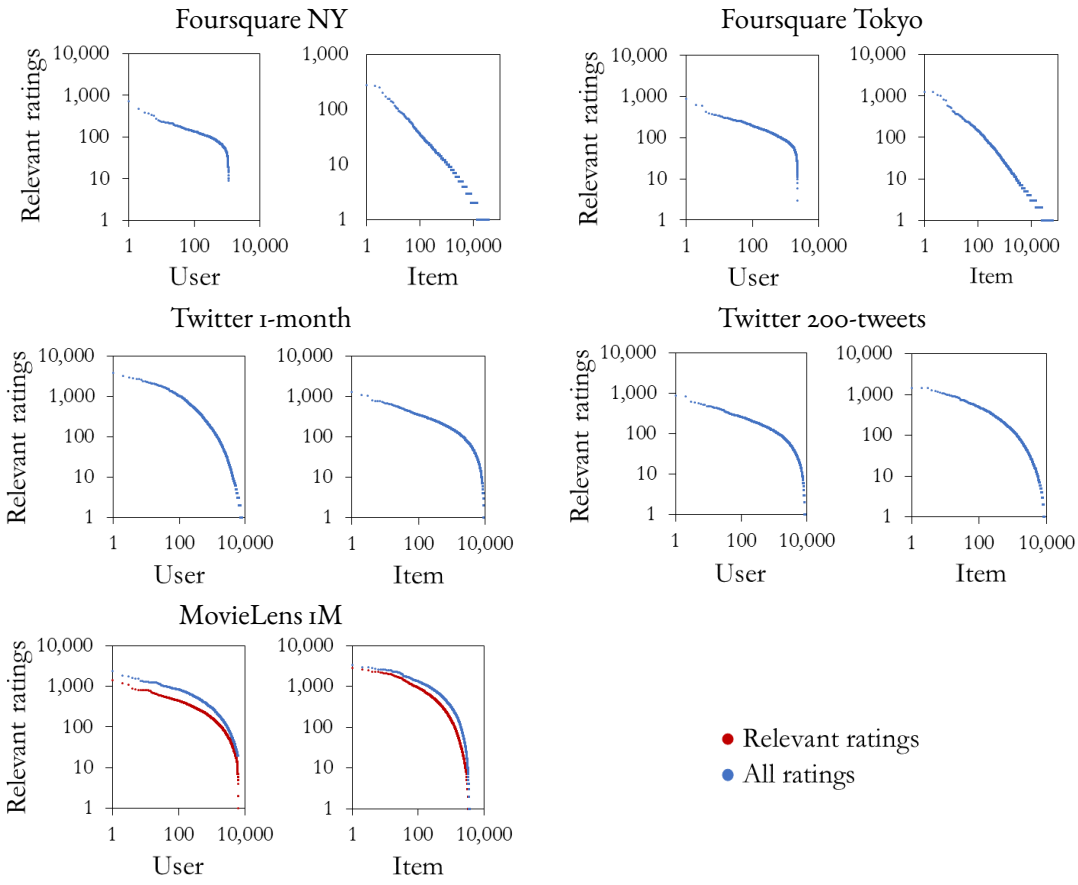


Figure 10.3: Rating distributions in the different datasets. Left figures show the number of ratings for each user, right ones the number of ratings for each item. Plots are in log-log scale. In the MovieLens 1M figure, red dots represent the subset of relevant ratings (ratings with value greater or equal than 4).

- **Foursquare:** The Foursquare dataset[2] is an implicit feedback venue recommendation dataset, where the collected data represents real check-ins in different facilities around the world, collected from the Foursquare service[3]. From the many datasets collected from this system, we use the New York and Tokyo datasets (Yang et al., 2015), which collect check-ins in both cities from April 2012 to February 2013. We process this dataset by combining user-item pairs, and we consider that the rating $r_u(i)$ if the user u has done a check-in in the venue i at least once.
- **MovieLens 1M:** The Movielens 1M dataset[4] is one of the most widely known and used collection of ratings in the literature (Harper and Konstan, 2016). It contains around one million explicit movie ratings (with values ranging from 1 to 5) extracted from the MovieLens platform[5]. For this dataset, we consider that a rating is relevant (i.e. $r_u(i) = 1$) if the user has given given a rating greater or equal than 4 and irrelevant ($r_u(i) = 0$) otherwise.
- **Twitter:** We finally check the accuracy of the different approaches in a contact recommendation setting. For this, we use the Twitter 1-month and Twitter 200-tweets follow datasets we introduced in Section 4.1. In the experiments we present in this section, we take the whole third snapshot of each dataset. Both Twitter datasets act as implicit feedback datasets, where $r_u(i) = 1$ if the link between users u, i exists, 0 otherwise (similarly to how we use this dataset in previous chapters).

The properties of each dataset are summarized in Table 10.4. Figure 10.3 shows, in log-log scale, the number of ratings for each user/item. We observe that both Foursquare datasets have the smallest density and the most

[2]Foursquare Dataset: https://sites.google.com/site/yangdingqi/home/foursquare-dataset (Accessed 19th December 2020)
[3]Foursquare: https://www.foursquare.com/ (Accessed 19th December 2020)
[4]MovieLens 1M dataset: https://grouplens.org/datasets/movielens/1m/ (Accessed 19th December 2020)
[5]MovieLens: https://movielens.org/ (Accessed 19th December 2020)

skewed distribution (in the case of the ratings-per-item distribution, the plot reveals a power law distribution). MovieLens 1M, on the other hand, has the larger density (even when we only consider the relevant ratings). Beyond the density, distributions for Twitter and MovieLens 1M are similarly shaped.

Algorithms

In our experiments, we consider the following recommendation algorithms:

- **Myopic collaborative filtering:** we compare the bandit approaches to a simple user-based kNN with cosine similarity (Ning et al., 2015) and a matrix factorization algorithm for implicit data (Hu et al., 2008, Pilászy et al., 2010). As updating the matrix factorization approach is quite slow (we do have to execute the whole training), instead of being updated every round, it is only updated after 100 positive ratings are collected.
- **Not personalized multi-armed bandits:** we consider for our experiments the ε-greedy (Sutton and Barto, 2018) and Thompson sampling (Chapelle and Li, 2011) bandits described above as not personalized algorithms.
- **Collaborative filtering bandits:** As stochastic collaborative filtering approaches, we consider two methods: first, the interactive ε-greedy InterPMF approach proposed by (Zhao et al., 2013) and then, the CLUB algorithm devised by (Gentile et al., 2014).
- **Nearest-neighbors bandit:** we consider the two variants of our bandit approach: $k = 1$ and $k > 1$.

We consider binary ratings ($r_u(i) \in \{0, 1\}$) for all the studied algorithms. In case we retrieve a missing value, it is consider by our algorithms as a failure ($r_u(i) = 0$). We do it like this because the ε-greedy, Thompson sampling and CLUB approaches estimate average ratings over users, and they would not properly work for implicit feedback datasets (such as social networks) otherwise. We should note that all the remaining approaches would not vary their behaviour if we did not update them after receiving such ratings.

Once the methods have been introduced, our evaluation procedure consists in the simulation of a recommendation feedback loop where the arms in the bandits (or the algorithms) are iteratively updated using the feedback provided to the recommended items.

10.4.3 Results

Cumulative gain under extreme cold start

Figure 10.4 illustrates the comparison of the tested approaches for different datasets. In that figure, the horizontal axis represents the number of iterations of the interactive recommendation process (each iteration representing a single recommendation) and the vertical axis shows the cumulative recall of each approach. As an overall observation, we find that our proposed kNN bandit (with k equal to 1) shows very promising results, beating the rest of alternatives in most of the datasets.

If we observe the myopic collaborative approaches (user-based kNN and implicit matrix factorization), we can note that they both have a slow start, after which they gradually improve. This is due to their inability to deal with both user and item cold start (so they work as random recommendation for a longer time than the other approaches, which manage to deal with this limitation thanks to their exploration capabilities). After enough recommendations have been made, these algorithms may accelerate enough to catch up with the bandits (sometimes, after the intervals that we show in the graphs). However, such fast recovery might come up too late to prevent user abandonment in real applications. In both Foursquare datasets, we even observe that the late recovery does not even occur: iMF never manages to beat the not personalized popularity-based recommendation baseline, and user-based kNN does not work at all, achieving similar results to random recommendation (even for high values of k). Something similar (but more extreme) occurs with the interactive PMF algorithm, which does not improve the random baseline in any of the datasets – the reason for this being the lack of update of the item factors: as we start without training data, the item factors are initialized randomly and left as they are, making it impossible for the method to learn the optimal factors. In this comparison, then, we find noteworthy that our proposed (bandit) kNN scheme is able not only to work but to outperform most of the tested approaches in these harsh cold start conditions.

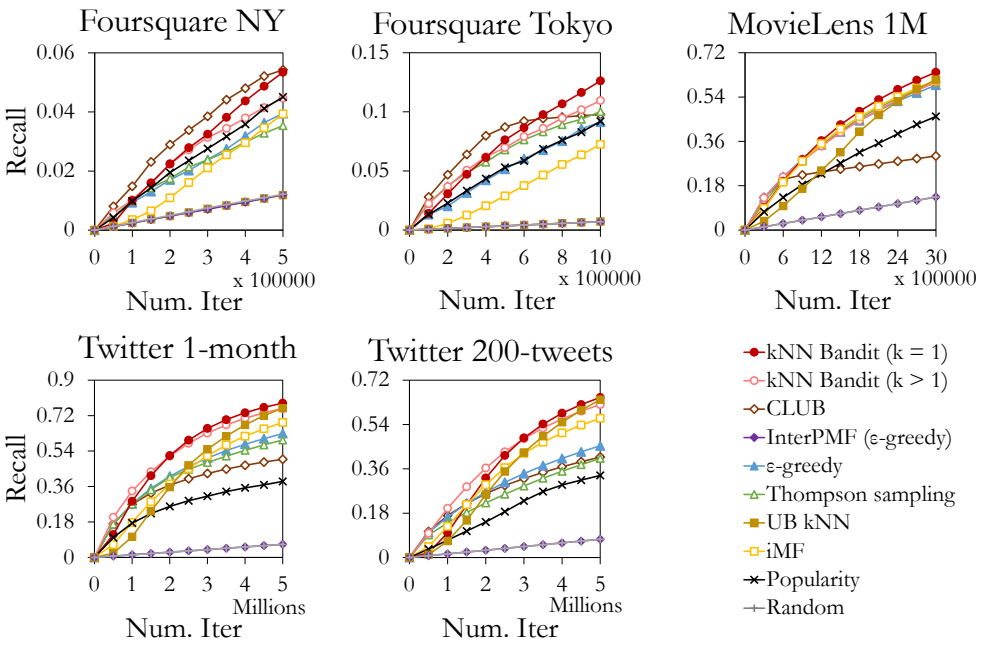


Figure 10.4: Cumulative recall over time for the different algorithms.

The second observation involves the popularity-based recommendation approach: although at the end of the simulations, it is far from being the optimal algorithm, at the beginning of the simulations it is generally close to the best approaches. However, as more information is available, they are beaten by personalized approaches (this is particularly notorious in both Twitter datasets, where it stagnates in the mid term). This is to be expected, since, when almost no information is available, aggregating such information is the best strategy we can use. Something similar occurs with the Thompson sampling and ε-greedy approaches: in the first few steps of the process, they work very well, but, as more information is available, they are beaten by the personalized methods. With the exception of Foursquare New York, the bandit algorithms are able to beat the myopic popularity-based recommendation, showing the advantages of applying some exploration over pure exploitation. With respect to these baselines, we find worth noting that the effectiveness improvement achieved by our bandit is achieved without introducing any additional parameters with respect to these approaches.

The CLUB algorithm shows a very different behaviour depending on the amount of available data. When only a few recommendations have been done, it achieves comparable results to Thompson sampling or ε-greedy. This is mostly because, when only a few ratings have been retrieved, there is a single cluster of users, and CLUB acts similarly to the not personalized bandit UCB1 (Auer et al., 2002). However, when more and more information is retrieved, differences between users are more likely to arise, and, therefore, lead to their separation in clusters. When this occurs, the behaviour is heavily dependent on the dataset: in the case of Foursquare NY, its performance does not decay, leading this algorithm to be the best performing algorithm – showing that it is able to separate the users in balanced clusters with different tastes. In the case of both Twitter datasets, its performance is very similar to ε-greedy and Thompson sampling. We hypothesize here that, in these two datasets, no division is applied, or there is a giant component of the graph concentrating most of the recommendations. Finally, for the remaining two datasets, the performance of the algorithm heavily decays (recovering positive ratings even slower than random recommendation) when enough information is found – thus showing that no good cluster structure is found. This is particularly notorious in MovieLens 1M, where the high density of the dataset might accelerate this behaviour.

Finally, as we have stated earlier, the kNN bandit is able to beat the rest of algorithms in most datasets from early steps of the simulation. This also applies to the multiple play variant (with $k > 1$): as we can see in Figure 10.4, results for this variant are not far from optimal and, at the early stages of the experiments, they even manage to beat it. However, in the end, the original approach is able to outperform the version with several neighbors in all the datasets.

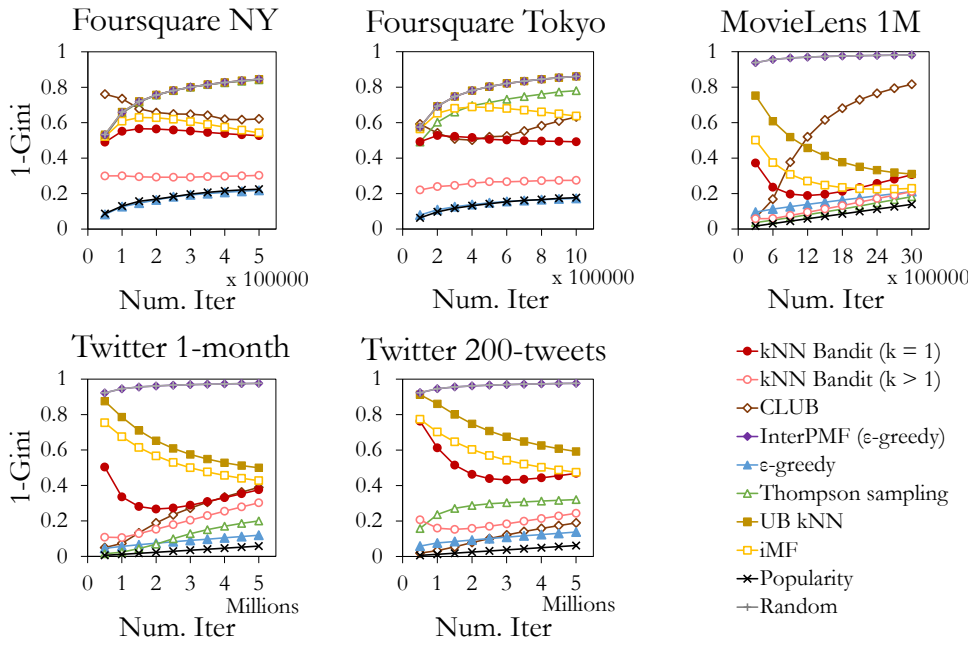


Figure 10.5: Evolution of the Gini coefficient of the recommendation over time for the different recommendation approaches.

Item diversity

In addition to the accuracy results, we provide some insights about how diverse the recommended items for the different approaches are. We consider for this the Gini index (Dorfman, 1979) of the distribution of recommended items over time:

$$\text{Gini}(t) = \frac{1}{|\mathcal{I}| - 1} \sum_{j=1}^{|\mathcal{I}|} (2j - |\mathcal{I}| - 1) \frac{|\mathcal{U}_{i_j}(t)|}{|\mathcal{H}_t|} \tag{10.31}$$

where the item i_j is the j-th item which has been less recommended. As this metric achieves value 1 when the distribution is completely imbalanced, we take its complement instead (subtracting the value of the Gini index to 1). We observe that, to some extent, this metric is useful to reveal the degree of exploration of the algorithms to some extent (exploration causes an algorithm to recommend very diverse items to retrieve useful knowledge of the users, and therefore, leads to high values). However, we should note that high diversity values do not necessarily correspond to high exploration: while exploiting the available knowledge of about the users in the system, it is possible for an algorithm to deliver very diverse items – also leading to such high Gini complement values.

A general observation we can made for the results in Figure 10.5 is that, as expected, popularity-based recommendation achieves the lower item diversity results, and, random recommendation (in most of the plots) does the same. More interesting observations can be made about the rest of approaches. We shall not make any comments on the InterPMF algorithm (since it works just as random recommendation) or user-based kNN on the Foursquare datasets (for the same reason).

Our proposed kNN bandit (with $k = 1$) stands as a mid-packer when it comes to provide diverse results. Indeed, as we consider greater values for k it provides even less diverse results. The reason for this is that greater values of k lead to a reduction on the exploration in early stages. To understand why this occurs, we might consider what happens when there is a single positive rating $r_u(i)$ in the system. As there is only a positive rating, there are not common ratings between pairs of users. Therefore, when we select k neighbors, we do it (mostly) at random, as the parameters for the Beta distributions are almost the same for all pairs: $\alpha_t(u|v) = 0$ for all $u, v \in \mathcal{U}$ and $n_t(v) = 0$ for all $v \neq u$ ($n_t(u) = 1$). If the user with the positive rating u is selected as part of the neighborhood, i is going to be the selected candidate item (unless the target user already knew the item): as $\text{sim}(u, u_t)$ is likely to be greater than 0 as it is sampled from a Beta distribution, then only i would receive a recommendation score greater than 0. Then, we can see that, as we increase k, the probability that

user u is selected increases (as we take a larger fraction of the users in the system) – and, consequently, the probability of selecting i as candidate item. In the extreme case where we take $k = |\mathcal{U}|$, item i would always be the recommended candidate item (unless it had been previously recommended to the target user) – thus making our bandit similar to popularity-based recommendation which, by definition, is the algorithm delivering the less diverse set of recommendations in terms of the Gini complement – something that we can observe in Figure 10.5.

The same observation we have made for our bandit is not true for the myopic user-based kNN, which, even for large k values, manages to achieve a better item diversity value. This occurs because, when no information is available, differently from our approach, where the Beta sampling makes $\text{sim}(u, v) > 0$ in most cases, the similarity is always 0 when there are no common ratings between the target user and her neighbors – preventing the previous phenomenon to occur. We observe that, in all the datasets, the diversity of this algorithm improves that of the kNN bandit (similarly to iMF). However, we observe that both myopic approaches decrease their diversity over time in the Twitter and MovieLens 1M dataset – suggesting that the acquired advantage is due to a longer exploration phase, as pure collaborative filtering cannot deal with cold start situations. Something similar can be observed in the Foursquare datasets for iMF (user-based kNN works as random in those cases). However, we can see that, at early stages, its diversity increases. As the density of such datasets is very low and the number of items is far superior on Foursquare than on the other sets, this effect might be caused because we are observing an early stage of the process, i.e. that we need more recommendations per user in Foursquare than in Twitter or MovieLens 1M to end the exploration phase.

About the not personalized bandits, we observe that, in general, ε-greedy achieves similar results to those of popularity-based recommendation. As the selected ε values are low (as seen in Table C.8), this is to be expected. Thompson sampling, however, manages to obtain much more diverse results. By definition of the algorithm, it reduces the amount of exploration it does over time. So, it is expected that, when starting from a cold start situation, the algorithm manages to explore much more than ε-greedy – something that is illustrated in our results. This is particularly notable in the Foursquare datasets, where, due to the large number of items, it takes even more time to explore the optimal parameters of the Bernoulli distributions.

Finally, we observe mixed values for the CLUB algorithm: in both Foursquare datasets, it achieves great item diversity – in early stages, even greater than random. Although this might seem unlikely, when few data is available, the upper confidence bound is what determines the recommended item. As the confidence bound for an item diminishes with the number of times it is recommended, when no more information is available, it is likely that a different item is recommended each time (leading to more diverse recommendations than random). As information is available, it focuses on the most popular items – thus diminishing the diversity over time.

This is what occurs at the first points shown for the MovieLens and Twitter datasets: it behaves close to ε-greedy and Thompson sampling. However, in those datasets, its diversity starts to grow very quickly (even leading CLUB to be the best in MovieLens1M). If we compare with Figure 10.4, we notice that the points of maximum growth of the item diversity correspond to the points where the performance of the algorithm starts to decay – as the clusters are divided, different items are recommended to different groups of users, leading to more diverse sets of recommendations.

10.5 Conclusions

In this chapter, we have explored the evolutionary dynamics of recommender systems, and we have considered their relation to reinforcement learning. As an initial step on the study of this interactive recommendation perspective, we have developed a novel algorithm, based on multi-armed bandits. This algorithm represents a stochastic take on the user-based nearest-neighbors algorithm, where the neighbors are selected according to a bandit strategy. We have then explored the potential of such approach in the interactive recommendation cycle when compared to other collaborative filtering strategies in five different datasets. The main conclusions of such study are summarized next:

- Our bandit seems to be able to provide relevant recommendations during the recommendation cycle, when compared to other (both stochastic and myopic) collaborative filtering approaches.

- When compared to the classical user-based scheme, our approach has been observed to be more sensitive to the uncertainty in the observations, and it is able to provide relevant recommendations even when a very small number of ratings have been observed.
- If the algorithms receive no information prior to the start of the recommendation cycle, the best version of our bandit algorithm just selects a single neighbor. If we select more neighbors, it is a) more time consuming and b) achieves worse accuracy and diversity results.

11 Conclusions and future work

In this thesis we have investigated the problem of recommending people in social networks with whom people want to connect: contact recommendation. First, we have researched equivalences between contact recommendation and information retrieval. On this basis, we have studied the factors that make a recommendation approach good for the task. We have then explored complementary evaluation dimensions to the accuracy of the algorithms, with a particular focus on the effect these algorithms have on the structural diversity of the network. We have observed the benefits that recommending weak ties has on the novelty and diversity of the information that arrives to the users and the mitigation of filter bubbles. Finally, we have analyzed what happens when we understand recommendation as an interactive cycle. In this perspective, we have developed a k nearest neighbour collaborative filtering approach that considers the uncertainty in the system knowledge when selecting neighbors for the target user.

We now summarize the work we have conducted throughout this thesis, and its main contributions to the field. Afterwards, we discuss possible directions which might be explored for continuing our research.

11.1 Summary and contributions

In this section, we summarize and discuss the main findings and contributions of our research, addressing the different goals stated in Chapter 1.

11.1.1 Review of the state of the art

As a first step in our research, we have introduced and formalized the contact recommendation task and we survey the state of the art algorithms. As a basis for empirical analysis of the recommendation methods developed in the thesis and in the literature, aiming to compare the different recommendation approaches under a unified offline evaluation setup, we have downloaded two directed network samples from Twitter, using variants of the snowball sampling approach (Goodman, 1961). We have considered both a) the explicit, stable and unweighted follows relations between users, and b) their implicit dynamic links, reflected in the interactions between them in the platform. Along with these samples, we consider an additional dataset, an undirected sample from Facebook, obtained by McAuley and Leskovec (2012).

We have then compared the effectiveness of state of the art approaches for the top-N recommendation task (Cremonesi et al., 2010), using ranking-oriented accuracy metrics: precision, recall and nDCG (Baeza-Yates and Ribeiro-Neto, 2011). Results show that collaborative filtering approaches like matrix factorization (Hu et al., 2008) and nearest neighbors approaches (Ning et al., 2015), originally devised for delivering item recommendations, appear to be the most effective contact recommendation methods. They are closely followed by friends of friends approaches, like Adamic-Adar or most common neighbors. On the other hand, content-based approaches and path-based ones do not work well under this setting.

11.1.2 Relation between IR and contact recommendation

We have found rich and meaningful connections between textual information retrieval, oriented to search documents in massive and unstructured spaces, and the contact recommendation task. By mapping the three fundamental spaces in the IR task (queries, documents and terms) to the people in the social network (playing the role of both users and items in the people recommendation task), we have built a framework for adapting IR principles and techniques to recommend people in social networks in a collaborative filtering fashion.

We have effectively adapted several IR models through this framework. We have explored their utility as contact recommendation approaches and show that adapting IR models leads to empirically effective solutions.

One of these models, BM25, stands out above the rest, achieving competitive results when compared to the best state of the art approaches. Also, taking advantage of search engine infrastructure, with elements like inverted indexes, and algorithms for fast computation of retrieval scores, IR models prove to be some of the most efficient and scalable contact recommendation algorithms in terms of execution time and memory consumption – much lighter than high performance algorithms like matrix factorization – as well as easy and fast to update without any loss of precision. Consequently, IR models appear as interesting effective and efficient alternatives to other contact recommendation methods.

We have explored alternative applications of these models in the contact recommendation task to further close the accuracy gap between the direct IR-based recommendation approaches and the best performing algorithms, like matrix factorization. We have first noticed that any people recommendation algorithm can be used as a neighborhood selection method in user-based and item-based kNN schemes. Therefore, we have explored the utility of IR-based approaches for the task. Interestingly, in our experiments, IR models appear to be – consistently and substantially – more effective when used to determine the optimal neighborhood of the target users than when used as direct recommenders. Even when IR models are not particularly effective as recommenders, they suceed in identifying good neighbors whose friends are interesting people for the target user to bond with.

as a natural step for exploring how far we can improve the effectiveness of contact recommendation by taking advantage of IR techniques, we have investigated the adaptation of learning to rank methods. We propose a scheme to adapt them for suggesting people, where we use IR-based algorithms (both direct methods and kNN variants) in two different roles: first, as samplers to retrieve an adequate subset of candidate users to rank, and then, as features for the target-candidate user user pairs. In our experiments, we observe that these learning to rank approaches manage to consistently outperform the state of the art and IR-based approaches.

Overall, we find IR models to be effective in three different roles for contact recommendation: first, as direct recommendation algorithms; second, as neighbor selectors in a k nearest neighbors scheme; and, last, as samplers and features in the application of learning to rank approaches. Our research shows that analyzing the close relation between textual search and people recommendation is not only interesting from a theoretical point of view, but also has a great potential for building principled and effective approaches.

11.1.3 Properties of effective contact recommendation approaches

In a formal axiomatic-based approach, we have studied the properties that make different methods effective for the task. We mostly focus on one of the broader families of algorithms: friends of friends methods – which includes all the standalone IR models..

We have used the mapping between the IR spaces and the users in contact recommendation to translate the fundamental axioms applied in the design of effective models for textual search. Then, we have checked whether they are useful for developing more effective approaches. We have found that, when considering people at distance 2 from the target user, candidate users with a large number of contacts in common with the target user make for accurate recommendations. Furthermore, recommendation accuracy is enhanced by weighting the common neighbors according to their degree: common contacts with low degree are better able to discriminate the actual preferences of the target users, and consequently, they should be given more importance . Finally, we have observed that penalizing popular candidate users decreases the utility of recommendation approaches, as it interferes with the preferential attachment trend commonly present in social networks (Barabási and Albert, 1999). The latter observation is consistent with other experiments, where penalizing popularity (as do Jaccard and the vector space model and the global LHN index)results in quite ineffective recommendations.

In addition to this axiomatic analysis, we have explored which neighbors we should choose for representing the target and candidate users in order to maximize accuracy in directed networks. We observe that the followers of the candidate users seem to describe their value better than their followees, and the union of both followers and followees appears to best represent the social needs of the target users.

11.1.4 Effects on the network structure

We have researched several utility notions to complement the accuracy of the recommendations – which just targets the density of the network. We have explored the definition of metrics for quantifying the potential

effect that recommending new links has on the structure of the social network. We have considered the effect on a) the distances between users in the network and b) its structural diversity, understood in terms of non redundant and weak ties. Apart from this perspective, we have also explored the novelty and the diversity of the recommendations, by adapting to our problem metrics from the recommender systems field, and aspect-based diversity metrics from the information retrieval field.

Measuring and comparing such dimensions for a selection of relevant contact recommendation methods, we have noticed that, generally, the trivial, suboptimal and not personalized algorithms are the ones which score best for the developed metrics: in particular, recommending people randomly is the best way to close the distances between users, balance the degree distribution and the connections across communities, and recommending popular users appears to be the best way to maximize the number of links between communities. In contrast, the novelty, diversity and positive effects of more accurate recommendation algorithms on the network structure are far from the effectiveness of non personalized approaches for such dimensions. Among the most accurate recommendation methods, matrix factorization provides a fair balance between accuracy and diversity, and memory-based collaborative filtering methods manage to provide unexpected (but relevant) recommendations to the users.

Afterwards, we have proposed an optimization strategy that enhances global properties of the system by gradually retargeting recommendations towards the desired metric, while keeping a trade off with the accuracy of the original recommendations. Differently from classical reranking strategies (Vargas et al., 2012), which independently reorder the recommendations for each user, our algorithm considers the recommendations made to the rest of the users to optimize every single ranking. We use it to find a practical signification for the enhancement of the structural diversity of the network via recommendations. Specifically, we have found that recommending weak ties produces an increase of the novelty and diversity of the information that arrives to the users in the networks. This can have potential positive effects in achieving healthy social media and mitigating filter bubbles (Pariser, 2011) and echo chambers.

11.1.5 Interactive recommendation

Finally, we have explored the recommendation task when it is understood to be an interactive process with a feedback loop. We have first provided a formalization of the interactive recommendation task and established links with the reinforcement learning paradigm.

We have developed a novel collaborative filtering approach, versioning the classical user-based nearest neighbors method. Differently from classic nearest-neighbors, our proposal applies a stochastic multi-armed bandit strategy (Thompson sampling) to select the neighbors for the target user. This results in an algorithm that is sensitive to the uncertainty present in the available information, and explores user neighbors further than the basic kNN. Moreover, selecting a single neighbor for the target user results in a compact and simple algorithm that works in the recommendation cycle – even dealing with extreme cold start situations, which are commonly problematic for collaborative filtering .

11.2 Future work

Our findings and developments open the way in many directions. We outline here some of them, which we see as worthwhile for extending our work.

11.2.1 Further IR methods for contact recommendation

In Part II, we explore the adaptation of different information retrieval concepts and techniques to the contact recommendation task, with positive results. Such adaptations have allowed us to obtain useful insights about the development of effective people recommendation approaches, but they are not the only developments originally devised for the search task which might be considered. We highlight here some possibilities for expanding this research line.

The first one represents the most straightforward extension of our research, and involves the adaptation of the most recent algorithmic developments in textual search. Particularly, the use of deep neural networks (Mitra and Craswell, 2018). Although the usefulness of deep learning developments for offline recommendation has

been recently discussed (Dacrema et al., 2019), the close relationship between contact recommendation and textual search, along with the success of these approaches for the text retrieval task in recent competitions like TREC (Craswell et al., 2019) advises to consider this perspective.

Beyond the adaptation of deep learning models, another possibility consists in drawing from other IR areas. Two of them seem particularly promising for contact recommendation: (pseudo) relevance feedback (Rocchio, 1971, Ruthven and Lalmas, 2003) and query reformulation (Huang et al., 2013). As nearest neighbors variants consistently outperform direct IR models in our experiments, this suggests that the actual neighbors of the target and candidate users may not be the absolute optimal representation. Therefore, applying those techniques to expand or reformulate this representation might lead to further enhancements.

11.2.2 Axiomatic analysis of recommender systems

We have seen in Chapter 6 that information retrieval axioms cannot be applied to examine many of the most accurate approaches, such as matrix factorization, nearest neighbors approaches or random walks because such methods are not restricted to recommend people at distance 2. We envision two possibilities to overcome this limitation.

First, building synthetic data, we can analyze to which extent these approaches are able to satisfy the basic IR axioms adapted in Chapter 6. This perspective is similar to the works by Rennings et al. (2019) and Câmara and Hauff (2020), who create text collections to explore how much deep learning models adhere to the constraints in the search task. This line of study may provide further insights about the utility and relevance of the original constraints in the design of new strategies.

The second way involves the generalization of the constraints we introduce in Chapter 6, so they are applicable to a larger family of algorithms. For example, we might start by studying the axioms for pseudo-relevance feedback (Ruthven and Lalmas, 2003), following the work by Clinchant and Gaussier (2013). Finding a suitable set of properties that contribute to the success of different contact recommendation algorithms is a desirable goal that might also be extended to other recommendation domains, such as movie or music recommendation.

Beyond the exploration of useful properties for the contact recommendation task in general, we might also apply axiomatic analysis to other (smaller) aspects of recommendation. For instance, in Chapter 7, we study the potential of IR models as similarities for neighbor-based collaborative filtering . Similarly to how we do for standalone approaches, it would be interesting to identify the optimal properties for selecting a suitable set of neighbors for the target and candidate users in such schemes, following previous work by Valcarce et al. (2017, 2018).

11.2.3 Beyond accuracy

Further utility dimensions and metrics beyond accuracy than we consider in Chapter 8 can be explored. For instance, we could take other social network analysis properties, or we could expand our work towards the fairness of the recommendation approaches (Ekstrand et al., 2019).

Apart from the proposal of novel metrics, we can further explore the proposed metrics. In Chapter 9, we focus on the potential benefit that weak tie related metrics have on the mitigation of filter bubbles (Pariser, 2011). However, we still need to better understand the real utility of other metrics – either for the people in the network or for other interested parties, such as platform owners or administrators, network data consumers, etc. The value of such properties might be domain dependent, but a certain level of abstraction might be possible. The development of novel recommendation techniques targeting both accuracy and some of these metrics is also an interesting perspective we would like to explore.

Furthermore, an extension of our work on information diffusion could be envisaged, by considering further communication dynamics and protocols besides the Twitter model considered in this thesis. We might use, for instance, the independent cascade model (Goldenberg et al., 2001), the linear threshold model (Kempe et al., 2003) or the push-pull protocol (Demers et al., 1987, Doerr et al., 2011) to model the spread of information. In addition, in our experiments we only consider hashtags as topical information. We might explore other information features or variants, such as the opinion diversity on a given topic. Furthermore, we might target other problems beyond reducing filter bubbles in the network, such as the mitigation of the glass ceiling effect (Stoica et al., 2018).

11.2.4 Interactive recommendation

The work on interactive recommendation presented in Chapter 10 serves as a starting point on the study of this perspective. As such, the possibilities for extending our work are manifold. First, as theoretical and practical developments on interactive recommendation are very recent, there is room for the design of better recommendation approaches based on reinforcement learning techniques – not only multi-armed bandits (Lattimore and Szepesvári, 2020), but also (more complex) Markov decision processes (Sutton and Barto, 2018).

We can also vary our experimental setup. In our work, we have focused on the case where a single item is recommended at a time, and we update the recommenders just after the feedback is retrieved. However, recommendation approaches are commonly updated every certain time (for example, Satuluri et al. (2020) reports that contact recommendations stay relevant for weeks on the Twitter "Who to follow" system), and more than one item is recommended at once. Analyzing how the effectiveness of approaches changes depending on such settings may provide further insights on interactive recommendation.

Another interesting perspective considers the use of training data in the interactive recommendation setting which can be leveraged e.g. for tuning the hyperparameters of interactive recommendation algorithms – in our work, we just explore the extreme cold start problem, where no information is provided to the algorithms beforehand. Nonetheless, when a recommendation algorithm starts to work on a platform, it is commonly trained on the whole set of previous ratings in the system. If a previous algorithm was running on the platform, the data might be greatly influenced by such approach (Gruson et al., 2019). Understanding the differences in the cyclic behaviour of recommendation algorithms depending on the properties of the training data appears then as a compelling problem.

Last, as far as we are concerned, the development and evaluation of interactive recommendation techniques has been mainly tied to their accuracy, and only a few works have been addressed additional evaluation dimensions. In this thesis, we measure the diversity of the items recommended by the different systems overtime, but we could envision the development of further novelty and diversity metrics for this problem. Also, we could design multi-armed bandit strategies that jointly optimize these dimensions and the accuracy of the system, following Li et al. (2020), Mehrotra et al. (2020).

11.2.5 Effects of interactive recommendation on network structure

In our work, we have separately studied the potential effect of recommendation algorithms on network properties, and the influence of user preferences on the cyclic learning process of recommender systems. However, in real world applications, both processes are intertwined. Therefore, we envision the analysis of long term changes in the structural properties of the network, by incorporating different approaches (both stochastic and deterministic) into the feedback loop for the contact recommendation task. A precursor of this proposal can be found in the work by Farajtabar et al. (2017), who study the relation between the evolutionary process of the social networks and changes on information diffusion.

11.2.6 User studies and online evaluation

The experiments we conduct in this thesis have been limited to an offline evaluation setting, where we analyze the outcome of our recommendation approaches over a set of network samples. Although offline evaluation represents the most common approach in academic recommender systems research, it is typically biased and differences often surface when we move from offline to the production system (Rossetti et al., 2016). User studies and/or A/B testing would complement our experiments with further objective perspectives.

In addition, in Chapter 8 and 9, we have considered the diversity of the recommendations (and the information flow) to be positive properties, following previous research on network science, recommender systems and information retrieval (Burt, 1995, Granovetter, 1973, Ugander et al., 2012, Vargas and Castells, 2014, Vargas et al., 2012). User studies would allow us to know the vision that actual users of social network platforms have on the beneficial properties of enhancing these properties, similarly to Ekstrand et al. (2014), Kotkov et al. (2018), who explore the relations between diversity, novelty and serendipity with user satisfaction. These user studies might even lead us to find more (and unexpected) practical signification and utilities of diversifying the network properties.

Appendices

"Home is now behind you, the world is ahead!."
J.R.R. Tolkien

A Introducción

A.1 Motivación

Las redes sociales online conforman el ejemplo perfecto de la progresiva transformación de la World Wide Web hacia un servicio más participativo, dirigido por los contenidos creados por sus usuarios y la comunicación entre ellos. Plataformas como Facebook, Twitter, LinkedIn o Tik Tok ayudan a millones de personas a contactar y establecer lazos significativos con usuarios alrededor del mundo, o a compartir con ellos fotos, intereses o sentimientos. Desde su concepción a finales de los años 90 del siglo XX, con sitios web como SixDegrees o Friendster (boyd and Ellison, 2007), la adopción de los medios sociales por parte del público general ha sido masiva, llevándoles incluso a aparecer entre las páginas Web más visitadas.[1] La importancia de estas plataformas ha ido en continuo crecimiento, causando un gran impacto en la sociedad moderna y en su estilo de vida, con efectos – tanto positivos como negativos – en diversos ámbitos, como la privacidad, la salud (O'Keeffe et al., 2011), la política (Hong and Kim, 2016) y la forma en la gente se relaciona.

Si ya en el año 2013 alrededor del 73% de la población adulta de E.E.U.U. admitía haber utilizado al menos una red social, y el 42% tener cuentas de usuario en varias de ellas,[2] hoy en día el número de usuarios activos en Twitter, LinkedIn y Facebook se mide en cientos de millones de personas en todo el mundo. Además, estos usuarios crean y publican miles de nuevos contenidos en estas plataformas cada segundo (de acuerdo con estadísticas recientes,[3] más de 9,000 tweets y más de 1,000 fotos son publicados, respectivamente, en Twitter e Instagram en este período de tiempo). Una consecuencia de disponer de una cantidad de datos de esta magnitud es que los usuarios pueden sufrir una *sobrecarga social* (Guy, 2015), un problema que aparece como la combinación de otros dos: por un lado, la cantidad de información a la que cualquier ciudadano puede acceder en cualquier red social online se ha vuelto inmanejable – un problema conocido como sobrecarga de información; por otro, los usuarios participan hoy en día en una cantidad tan enorme de interacciones al mismo tiempo que no son capaces de lidiar con ella – o, en otras palabras, sufren de sobrecarga de interacciones. La presencia de estos problemas pone en evidencia la necesidad de herramientas automáticas que permitan resumir y filtrar los diferentes contenidos, para que vuelvan a ser accesibles para los usuarios.

Los sistemas de recomendación (Adomavicius and Tuzhilin, 2005, Ricci et al., 2015) son una familia de herramientas para aliviar estos problemas, diseñada para asistir a los usuarios a la hora de filtrar la información disponible y descubrir aquellos productos, contenidos o personas que tengan valor para ellos. Su inclusión en un amplio espectro de servicios y aplicaciones ha disparado su popularidad en las últimas dos decadas. Hoy en día, servicios de contenido audiovisual (Netflix, HBO, Spotify) nos recomiendan películas que ver o canciones que escuchar, plataformas de comercio electrónico como Amazon o eBay proponen productos que comprar, repositorios de artículos académicos (Mendeley, Google Scholar) nos sugieren publicaciones que leer. Las redes sociales online son otro dominio de aplicación más de estos sistemas.

La confluencia entre las redes sociales online y los sistemas de recomendación ha dado lugar a nuevos e interesantes perspectivas y desafíos (Guy, 2015). Disponer de las conexiones entre los usuarios introduce conexiones potencialmente útiles entre la recomendación, el análisis de redes sociales y la ciencia de redes que pueden ser aprovechados para mejorar las recomendaciones de items (Chaney et al., 2015, Fan et al., 2019, Tang et al., 2013). Otro caso de gran interés surge cuando, en lugar de sugerir ítems, recomendamos gente en la red social con la que los usuarios quiera establecer conexiones (Guy, 2018, Hannon et al., 2010). A este problema se le conoce como recomendación de contactos.

[1] Alexa's top 50 sites: https://www.alexa.com/topsites (Visitado el 19 de diciembre de 2020)

[2] Social Media Update 2013: https://www.pewresearch.org/internet/2013/12/30/social-media-update-2013/ (Visitado el 19 de diciembre de 2020)

[3] Internet Live Stats (1 second): https://www.internetlivestats.com/one-second/ (VVisitado el 19 de diciembre de 2020)

Esta perspectiva particular de la recomendación añade a la presencia de enlaces entre usuarios la propiedad distintiva de que los usuarios y los ítems no están en espacios separados, sino que proceden del mismo conjunto de elementos. La recomendación de personas se ha convertido en un componente fundamental de las plataformas de red social online, y ha recibido como atención tanto de la academia (Backstrom and Leskovec, 2011, Hannon et al., 2010) como de la industria (Goel et al., 2015, Gupta et al., 2013, Guy, 2015, Satuluri et al., 2020). Las redes sociales más importantes comenzaron a ofrecer servicios de recomendación de contactos en la segunda mitad de la década de los 2000: entre otros, Twitter tiene su servicio "A quién seguir", Facebook tiene "Personas que quizá conozcas" y LinkedIn "Gente que podrías conocer". La investigación que se presenta en esta tesis se centra en el estudio de este problema desde diferentes perspectivas.

La primera perspectiva considera el desarrollo de soluciones efectivas que buscan maximizar el crecimiento de la densidad de la red – o, en otras palabras, la cantidad de nuevos enlaces que aparecen gracias a la recomendación. Se trata del principal foco en la investigación de la recomendación de contactos en la literatura, y ha inspirado la creación de una amplia variedad de estrategias desde diferentes campos – incluyendo la ciencia de redes (Liben-Nowell and Kleinberg, 2007, Lü and Zhou, 2011, Martínez et al., 2017), el aprendizaje automático (Hasan et al., 2006, Lichtenwalter et al., 2010), los sistemas de recomendación (Hannon et al., 2010), o, a menor escala, la recuperación de información textual (Hannon et al., 2010).

Si bien la relación entre la recuperación de información textual (Baeza-Yates and Ribeiro-Neto, 2011) – orientada a la creación de motores de búsqueda – y los sistemas de recomendación se ha estudiado ampliamente (Adomavicius and Tuzhilin, 2005, Bellogín et al., 2013, Parapar et al., 2013, Wang et al., 2008a,b), es posible establecer relaciones nuevas e interesantes entre ambas tareas cuando no hay distinción entre usuarios e ítems. Esto permite el desarrollo de nuevos métodos basados en la adaptación de modelos orientados a búsqueda. Y no sólo eso: dado que el diseño de estos modelos se basa en una serie de principios establecidos (Fang and Zhai, 2006, Fang et al., 2004), el estudio de esta relación puede dar lugar a valiosa información sobre los mecanismos que provocan la evolución de las redes sociales – lo cual, a su vez, puede desembocar en el desarrollo de estrategias de recomendación aún mejores.

La recomendación de contactos ha evolucionado hasta jugar un papel sustancial en el crecimiento de las redes sociales: una fracción considerable de los nuevos enlaces que aparecen en redes sociales online son creados como respuesta a una recomendación automática (Aiello and Barbieri, 2017, Goel et al., 2015, Guy, 2018). Diferentes algoritmos podrían dar lugar a la aparición de enlaces muy diferentes y, por tanto, a diversas estructuras de red. Dado que las redes sociales no tratan sobre usuarios aislados, sino sobre las conexiones e interacciones entre ellos, estos nuevos enlaces podrían no solo afectar al entorno más cercano de los usuarios, sino a la red en su totalidad. Por ejemplo, los cambios en la estructura de la red pueden dar lugar a efectos positivos, como una reducción de la polarización de los usuarios (Musco et al., 2018), pero también negativos, como la creación de filtros burbuja (Pariser, 2011) o la perdurabilidad de fenómenos discriminatorios como el efecto de techo de cristal (Stoica et al., 2018). Debido a ello, es fundamental comprender cómo los distintos algoritmos modifican las propiedades de la red, y las consecuencias que estos cambios tienen para los diferentes actores involucrados en la misma (no solo los usuarios, sino también los propietarios, accionistas o trabajadores).

Esta es la segunda perspectiva que consideramos en esta tesis. En concreto, nos centramos en el impacto que la recomendación de contactos tiene sobre diferentes nociones (tanto pasadas como nuevas) de diversidad estructural (Burt, 1995, De Meo et al., 2014, Easley and Kleinberg, 2010, Granovetter, 1973, Ugander et al., 2012) – diversidad que observamos como una posible característica positiva de las redes sociales. Además, analizamos las consecuencias que tiene la recomendación de contactos sobre la novedad y la diversidad de la información que llega a los usuarios de la red cuando utilizamos dicha recomendación para promover estas propiedades de diversidad estructural.

Finalmente, no solamente los sistemas de recomendación influirían en el comportamiento de los usuarios y provocan cambios en la red – lo contrario también es cierto: para mantener sus recomendaciones relevantes, las estrategias de recomendación de contactos modifican su comportamiento en función de las nuevas conexiones que aparecen en la red social. La recomendación tiene una naturaleza cíclica, en la que la mayor parte de la información que el sistema recibe proviene de la reacción de los usuarios a las recomendaciones. A esta perspectiva cíclica se le ha dado el nombre de recomendación interactiva, y ha captado recientemente la atención de la comunidad (Gentile et al., 2014, Li et al., 2010, 2016, Zhao et al., 2013), por sus enlaces con el paradigma del aprendizaje automático conocido como *aprendizaje por refuerzo* (Sutton and Barto, 2018). Dado que esta

es una visión más realista de la recomendación, entender sus mecanismos y desarrollar nuevos métodos para la tarea es la tercera y última perspectiva estudiada en esta tesis.

A.2 Objetivos de investigación

El objetivo general de la presente tesis es comprender la tarea de recomendación de contactos, así como los efectos que esta tiene en la evolución de las redes sociales, en términos de su densidad (acierto) y diversidad. Enfocamos nuestro estudio en las siguientes metas de investigación:

- **O1: Revisar y comparar estrategias previas para la tarea de recomendar contactos.** La literatura sobre recomendación de personas y la predicción de enlaces en redes sociales es muy extensa. Sin embargo, no nos consta que exista una comparativa entre una amplia variedad de algoritmos bajo la perspectiva de recomendar los N mejores contactos. Por ello, buscamos realizarla y obtener información de utilidad sobre qué propiedades hacen que un algoritmo sea bueno (o malo) para la tarea.
- **O2: Explorar la adaptación de modelos recuperación de información textual a la tarea de recomendación de contactos.** Nuestra investigación pretende realizar contribuciones en dos niveles: en primer lugar, a nivel teórico, queremos obtener una mayor comprensión de la recomendación de contactos a través del desarrollo de conexiones entre las tareas de recomendación y recuperación de información; en segundo lugar, a nivel práctico, buscamos abrir nuevos caminos mediante los que avanzar en la creación de nuevos métodos de recomendación de contactos mediante la importación de técnicas de recuperación de información.
- **O3: Estudiar el efecto de la recomendación de contactos en la evolución de las redes sociales.** Como primer paso, buscamos comprender cómo los diferentes algoritmos de recomendación de personas modifican las propiedades estructurales de las redes (tales como distancias entre usuarios, redundancia de los enlaces, grados). Tras ello, queremos determinar los posibles beneficios que estos cambios tienen para los usuarios de la red social. En concreto, nuestro trabajo se centra en el efecto que mejorar la diversidad estructural produce en la diversidad del flujo de información en la red y en el atenuamiento de filtros burbuja.
- **O4: Desarrollar nuevos algoritmos de recomendación interactiva basados en aprendizaje por refuerzo.** Queremos adaptar técnicas de aprendizaje por refuerzo para desarrollar nuevos métodos eficaces para la recomendación cíclica. Para ello, nos centramos en una familia de técnicas de aprendizaje por refuerzo conocidas como bandidos multi-brazo.

A.3 Contribuciones

La investigación llevada a cabo en esta tesis ha dado lugar a varias contribuciones, que resumimos a continuación.

- Una infraestructura para la adaptación de principios y técnicas de la recuperación de información a la tarea de recomendar contactos. Esta infraestructura se basa en la relación entre los espacios fundamentales de la tarea de búsqueda (consultas, documentos y términos) y el único espacio en la recomendación de contactos basada en filtrado colaborativo: los usuarios de la red social. Esto permite la creación de nuevos algoritmos de recomendación de contactos así como de procedimientos experimentales para su evaluación.
- Una colección de algoritmos eficaces para recomendar contactos basados en técnicas de recuperación de información (IR). Diferenciamos tres grupos de métodos: a) adaptaciones directas de modelos de IR, b) algoritmos (de filtrado colaborativo) de vecinos próximos que utilizan modelos de IR para seleccionar los vecinos del usuario objetivo y los usuarios candidatos, y, c) métodos supervisados de aprendizaje de rankings (en inglés, *learning to rank*).
- Varios principios para el análisis teórico de algoritmos que recomiendan amigos de amigos, basados en las propiedades fundamentales (axiomas) del diseño de los modelos de recuperación de información.
- Un conjunto de métricas, adaptadas del análisis de redes sociales, que nos permiten medir cuantitativamente cómo la estructura de las redes sociales varía (de forma relevante) al aplicar algoritmos de recomen-

dación de contactos. Primero, estudiamos los efectos sobre la reducción de las distancias entre pares de usuarios en la red, y, en segundo lugar, la presencia de enlaces débiles en la estructura, definidos como a) enlaces no redundantes y b) enlaces entre diferentes comunidades de usuarios.

- Un algoritmo avaro global para promover propiedades del sistema (como por ejemplo, las propiedades estructurales en redes sociales) mediante la reordenación de la salida inicial de un algoritmo de recomendación. El método está diseñado para optimizar conjuntamente el valor de dicha propiedad global y el acierto del los nuevos ranking resultantes. Cuando se reordena una recomendación individual, nuestra estrategia no toma solo en cuenta las sugerencias realizadas al receptor, sino también las recomendaciones realizadas al resto de usuarios del sistema.
- Evidencias empíricas del efecto que recomendar enlaces débiles tiene en la difusión de información en redes sociales. En concreto, hallamos que incrementar la diversidad estructural de la red tiene beneficios en la novedad y la diversidad de la información que le llega a las personas de la red.
- Un algorithmo interactivo que aplica una estrategia de bandidos multi-brazo conocida como muestreo de Thompson para seleccionar los vecinos que mejor representan los gustos del usuario objetivo. El método resultante es una reformulación estocástica de los algoritmos de filtrado colaborativo basados en memoria y centrados en el usuario que trata con la incertidumbre en los datos para seleccionar dichos vecinos.
- Dos nuevos conjuntos de datos de red social, descargados de Twitter, que nos permiten evaluar los diferentes algoritmos de recomendación de contactos. Cada una de las muestras contiene dos redes dirigidas: primero, una red explícita (y estática) que muestra las relaciones estables de seguimiento entre los usuarios, y una red implícita (y dinámica), construida a partir de las interacciones entre los usuarios (menciones y retweets) en Twitter.

A.4 Estructura de la tesis

A continuación, resumimos la estructura del presente documento. Esta tesis se divide en cuatro partes y 11 capítulos:

- **Capítulo 1 (Introducción).** Presenta la motivación para la realización de esta tesis, junto a los principales objetivos de nuestra investigación, nuestras principales contribuciones, la estructura del documento y las publicaciones relacionadas con la tesis.
- **Parte I – Preliminares:** El primer bloque de contenido de la tesis repasa las nociones básicas y el trabajo previo en los diferentes campos que exploramos a lo largo de la tesis. Se divide en tres capítulos.
  - *Capítulo 2 (Trabajo relacionado).* Explora el contexto en el que se desarrolla esta tesis y resume los conceptos básicos de campos relacionados con nuestra investigación: sistemas de recomendación, recuperación de información, análisis de redes sociales y bandidos multi-brazo.
  - *Capítulo 3 (Recomendación de contactos).* Se centra en el problema de recomendación de contactos y revisa sus soluciones algorítmicas y desafíos.
  - *Capítulo 4 (Experimentos preliminares).* Describe la metodología experimental y los datos que utilizamos en esta tesis, así como un experimento inicial comparando métodos del estado del arte de la recomendación de contactos.
- **Parte II – Modelos de recuperación de información (IR) para la recomendación de contactos:** El segundo bloque explora la adaptación de algoritmos y técnicas de recuperación de información para recomendar personas en medios sociales. Se divide en tres capítulos.
  - *Capítulo 5 (Adaptación de modelos de IR).* Introduce la infraestructura básica que permite aplicar técnicas de recuperación de información a la recomendación de contactos y explora el uso de modelos clásicos de búsqueda como algoritmos de recomendación.
  - *Capítulo 6 (Análisis axiomático).* Desde una perspectiva de recuperación de información, estudia las cualidades que hacen que los modelos de IR sean efectivos para recomendar personas.

  - *Capítulo 7 (Adaptación avanzada de modelos de IR).* Muestra dos usos adicionales para los modelos de IR en la recomendación de contactos: como similitudes en algoritmos de filtrado colaborativo basados en vecinos y como características en algoritmos de aprendizaje de rankings.

- **Parte III – Novedad y diversidad en la recomendación de contactos:** El tercer bloque explora dimensiones para la evaluación y el diseño de estrategias de recomendación de contactos más allá del acierto. Está dividido en dos capítulos.

  - *Capítulo 8 (Impacto en la estructura de la red).* Investiga la definición de a) métricas para la recomendación de contactos y predicción de enlaces basadas en los conceptos de novedad y diversidad en los sistemas de recomendación e IR y b) métricas orientadas a medir los efectos de la recomendación de contactos en las propiedades topológicas de la red.
  - *Capítulo 9 (Efectos en la difusión de información).* Analiza los efectos que los algoritmos de recomendación de contactos tienen en la difusión de información en redes sociales, centrándose en la correlación que existe entre mejorar la diversidad estructural y la novedad y diversidad de la información que llega a los usuarios.

- **Parte IV – Recomendación interactiva:** El cuarto bloque explora la tarea de recomendación desde una perspectiva cíclica, en la que el sistema cambia a lo largo del tiempo.

  - *Capítulo 10 (Bandidos multi-brazo en recomendación).* Investiga el uso de bandidos multi-brazo como algoritmos de basados en filtrado colaborativo, y propone un nuevo método de vecinos próximos para abordar el problema exploración y explotación en el ciclo de recomendación.

- **Capítulo 11 (Conclusiones y trabajo futuro).** Resume los hallazgos de la tesis, y propone varias líneas de investigación para su extensión.

Además del contenido principal de la tesis, incluímos varios apéndices con información adicional, complementando los principales descubrimientos.

- **Apéndice A (Introducción)** contiene la traducción al español del Capítulo 1.
- **Apéndice B (Conclusiones y trabajo futuro)** contiene la traducción al español del Capítulo 11.
- **Apéndice C (Selección de parámetros).** Muestra la selección de los hiperparámetros óptimos para los experimentos realizados en esta tesis.
- **Apéndice D (Signficatividad estadística).** muestra los resultados de los test de significatividad estadística realizados para cada una de las comparaciones de algoritmos en función de su acierto.
- **Apéndice E (Algoritmos específicos de reordenación global).** Proporciona detalles adicionales sobre la implementación de las estrategias de reordenación de recomendaciones descritas en el capítulo 9.

A.5 Publicaciones relacionadas con la tesis

La investigación llevada a cabo en esta tesis ha dado lugar a varias publicaciones en talleres, congresos, revistas y libros relacionados con el área de recuperación de información (Sanz-Cruzado and Castells, 2018a, 2019a,b, Sanz-Cruzado et al., 2018a,b, 2019, 2020a,b) que resumimos aquí.

Publicaciones relacionadas con la Parte I

En la siguiente publicación (Sanz-Cruzado and Castells, 2018a), realizamos una panorámica de la tarea de recomendación de contactos, incluyendo una amplia lista de algoritmos y desafíos prácticos de la tarea. Además, incluimos una comparativa de algoritmos representativos del estado del arte en términos de acierto, novedad y diversidad.

- **Sanz-Cruzado and Castells (2018a)**: Javier Sanz-Cruzado and Pablo Castells. Contact Recommendations in Social Networks. In Shlomo Berkovsky, Iván Cantador and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 519-569. World Scientific Publishing, Singapore, 2018.

Publicaciones relacionadas con la Parte II

Las siguientes contribuciones exploran la relación entre los modelos de recuperación de información y las técnicas de recomendación de contactos en redes sociales. En Sanz-Cruzado and Castells (2019a), Sanz-Cruzado et al. (2018a, 2020a) adaptamos los modelos básicos de IR para sugerir personas en redes, como explicamos en el Capítulo 5. Analizamos los motivos tras la efectividad de dichos modelos en términos de un análisis axiomático en Sanz-Cruzado et al. (2020b). Finalmente, los contenidos restantes de Sanz-Cruzado et al. (2020a) exploran el uso de modelos de IR en un esquema de recomendación basado en vecinos próximos y como características en métodos supervisados de aprendizaje de rankings.

- **Sanz-Cruzado et al. (2018a):** Javier Sanz-Cruzado, Sofía M. Pepa and Pablo Castells. Recommending Contacts in Social Networks Using Information Retrieval Models. In *Proceedings of the 5th Spanish Conference on Information Retrieval (CERI 2018)*, Zaragoza, Spain, June 2018. ACM.
- **Sanz-Cruzado and Castells (2019a):** Javier Sanz-Cruzado and Pablo Castells. Information Retrieval Models for Contact Recommendation in Social Networks. In *Proceedings of the 41st European Conference on Information Retrieval (ECIR 2019)*, number 11437 in LNCS, pages 148–163, Cologne, Germany, April 2019. Springer.
- **Sanz-Cruzado et al. (2020a):** Javier Sanz-Cruzado, Pablo Castells, Craig Macdonald and Iadh Ounis. Effective Contact Recommendation in Social Networks by Adaptation of Information Retrieval Models. *Information Processing and Management*, 57(5),Article 102285, September 2020.
- **Sanz-Cruzado et al. (2020b):** Javier Sanz-Cruzado, Craig Macdonald, Iadh Ounis and Pablo Castells. Axiomatic Analysis of Contact Recommendation Methods in Social Networks: an IR perspective. In *Proceedings ofthe 42nd European Conference on Information Retrieval (ECIR 2020)*, number 12035 in LNCS, pages 175–190, Online, April 2020. Springer.

Publicaciones relacionadas con la Parte III

Los artículos listados a continuación analizan propiedades de la recomendación de contactos que van más allá del acierto. Sanz-Cruzado and Castells (2019b), Sanz-Cruzado et al. (2018b) introducen varias métricas de novedad, diversidad y diversidad estructural, y dan lugar a una primera observación de las mismas mediante la comparación de sus valores para una selección de algoritmos de recomendación de contactos. Esto se realiza de manera similar a cómo lo hacemos en el Capítulo 8. En (Sanz-Cruzado and Castells, 2018b) exploramos el efecto que recomendar enlaces para promover nociones de diversidad estructural tiene en las propiedades de la difusión de información.

- **Sanz-Cruzado et al. (2018b):** Javier Sanz-Cruzado, Sofía M. Pepa and Pablo Castells. Structural Novelty and Diversity in Link Prediction. In *Proceedings of the 9th International Workshop on Modelling Social Media (MSM 2018), in Companion of the The Web Conference 2018 (WWW 2018)*, pages 1347–1351, Lyon, France, April 2018. IW3C2.
- **Sanz-Cruzado and Castells (2018b):** Javier Sanz-Cruzado and Pablo Castells. Enhancing structural diversity in social networks by recommending weak ties. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018)*, pages 233–241, Vancouver, Canada, October 2018. ACM.
- **Sanz-Cruzado and Castells (2019b):** Javier Sanz-Cruzado and Pablo Castells. Beyond Accuracy in Link Prediction. In *Proceedings of the 3rdWorkshop on Social Media Personalization and Search (SoMePeaS 2019) co-located with the 41st European Conference on Information Retrieval (ECIR 2019)*, number 1245 in Communications in Computer and Information Science, pages 79–94, Cologne, Germany, April 2019. Springer.

Publicaciones relacionadas con la Parte IV

El siguiente artículo (Sanz-Cruzado et al., 2019) se centra en la recomendación interactiva. Específicamente, propone un algoritmo novedoso que combina filtrado colaborativo basado en vecinos con bandidos multibrazo.

- **Sanz-Cruzado et al. (2019)**: Javier Sanz-Cruzado, Pablo Castells and Esther López. A Simple Multi-armed Nearest-neighbor Bandit for Interactive Recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, pages 358–362, Copenhagen, Denmark, September 2019. ACM.

B

Conclusiones y trabajo futuro

En esta tesis, hemos investigado el problema de recomendar personas en redes sociales con las que la gente quiere conectar: la recomendación de contactos. En primer lugar, hemos investigado las equivalencias entre la tareas de recomendación de contactos y la recuperación de información. Sobre esta base, hemos estudiado los factores que hacen que un algoritmo de recomendación sea bueno para la tarea. Tras ello, hemos explorado dimensiones complementarias al acierto para evaluar la tarea, con un interés particular en los efectos que la recomendación de personas puede causar en la evolución de las propiedades estructurales de las redes. Hemos observado los beneficios que la recomendación de enlaces débiles tiene sobre la novedad y diversidad de la información que llega a los diferentes usuarios, así como en la disminución de los filtros burbuja. Finalmente, hemos explorado qué ocurre cuando entendemos la recomendación como un ciclo interactivo. Bajo esta perspectiva, hemos desarrollado un algoritmo de filtrado colaborativo basado en vecinos próximos que considera la incertidumbre en el conocimiento del sistema cuando seleccionamos los vecinos del usuario objetivo.

A continuación, resumimos el trabajo que hemos realizado a lo largo de esta tesis, así como sus principales contribuciones al campo. Después, discutimos posibles direcciones a explorar para proseguir nuestra investigación.

B.1 Resumen y contribuciones

En esta sección, resumimos y discutimos los principales descubrimientos y contribuciones de nuestra investigación, teniendo en cuenta los objetivos planteados en el Capítulo 1.

B.1.1 Revisión del estado del arte

Como primer paso en nuestra investigación, hemos introducido y formalizado la tarea de recomendación de contactos, además de proporcionar una revisión de los algorimos del estado del arte. Como base para el análisis empírico de los métodos de recomendación desarrollados en esta tesis y en la literatura, y con el objetivo de comparar estas estrategias bajo una configuración experimental offline unificada, hemos realizado la descarga de dos muestras dirigidas de la red social de Twitter, utilizando para ello variantes del método de muestreo de bola de nieve (Goodman, 1961) para recopilar usuarios y enlaces. En dichas descargas, consideramos a) las relaciones explícitas y estáticas de seguimiento entre los usuarios (sin pesos en los enlaces) y b) sus conexiones implícitas y dinámicas, que se reflejan a partir de las interacciones entre ellos en la plataforma. Junto a estas muestras, consideramos para nuestros experimentos un conjunto de datos adicional: una muestra no dirigida de Facebook, obtenida originalmente por McAuley and Leskovec (2012).

Utilizando estos datos, hemos analizado la efectividad de los algoritmos del estado del arte cuando se utilizan para recomendar N personas a cada usuario (Cremonesi et al., 2010), utilizando métricas de acierto orientadas a ranking: precisión, recall y nDCG (Baeza-Yates and Ribeiro-Neto, 2011). Los resultados nos muestran que los algoritmos de filtrado colaborativo como factorización de matrices (Hu et al., 2008) o vecinos próximos (Ning et al., 2015), diseñados originalmente para la recomendación de ítems, parecen ser las estrategias de recomendación más efectivas para la tarea. Los algoritmos que recomiendan amigos de amigos, como Adamic-Adar (Adamic and Adar, 2003) o la selección de usuarios candidatos que maximicen el número de conexiones comunes con el usuario objetivo (Liben-Nowell and Kleinberg, 2007) les siguen de cerca. En el lado opuesto, los algoritmos basados en contenido y en los caminos entre usuarios no parecen funcionar bien bajo esta configuración experimental.

B.1.2 Relación entre la recuperación de información y la recomendación de contactos

Hemos encontrado conexiones interesantes y significativas entre la recuperación de información basada en texto, orientada a la búsqueda de documentos en espacios masivos y no estructurados, y la tarea de recomendación de contactos. Conectando los tres espacios fundamentales en IR (consultas, documentos y términos) a las personas en la red social (que juegan el papel tanto de los usuarios como de los ítems en la recomendación de contactos), construimos un marco que nos permite adaptar los principios y técnicas de la recuperación de información para recomendar gente en redes sociales empleando estrategias de filtrado colaborativo.

Hemos adaptado con éxito diversos modelos de IR empleando las anteriores conexiones, explorado su utilidad como recomendadores de contactos y hallado que la adaptación de modelos de IR da lugar a soluciones empíricamente efectivas. Uno de estos modelos, BM25 (Robertson and Zaragoza, 2009), destaca sobre el resto en nuestros experimentos, logrando un aciertcomparable con los mejores algoritmos del estado del arte. Además, aprovechando la infraestructura de los motores de búsqueda, con elementos como índices invertidos y algoritmos que nos permiten calcular de forma rápida las puntuaciones de los documentos a recuperar, hemos demostrado que los modelos de IR se encuentran entre los algoritmos más eficientes y escalables en cuanto a tiempo de ejecución y consumo de memoria – se trata de algoritmos notablemente más ligeros que aquellos con mejor rendimiento, como los de factorización de matrices. Además, son algoritmos fáciles y rápidos de actualizar, sin sacrificar precisión en el proceso. En consecuencia, los modelos de IR surgen como alternativas eficaces y eficientes para la recomendación de contactos.

Hemos explorado otras aplicaciones alternativas de estos modelos en la tarea de recomendación de contactos, para tratar de mejorar consistentemente el acierto de los algoritmos más efectivos (factorización de matrices). En primer lugar, hemos observado que cualquier algoritmo válido para recomendar personas puede servir como método de selección de vecinos para las variantes del algoritmo kNN orientadas a usuarios y a ítems. Por ello, hemos comprobado la efectividad de los métodos fundamentados en IR para esta tarea. Sorprendentemente, en nuestros experimentos, los modelos de IR son más efectivos cuando se utilizan para determinar el vecindario óptimo de los usuarios objetivos que cuando se utilizan directamente como recomendadores de forma consistente y sustancial. Incluso aquellos modelos que no funcionan particularmente bien sugiriendo contactos tienen éxito a la hora de identificar los vecinos del usuario objetivo.

Como un paso natural para determinar cuánto podemos mejorar el acierto de los sistemas de recomendación de contactos mediante la explotación de técnicas de IR, hemos investigado la adaptación de técnicas de aprendizaje automático de rankings. Hemos propuesto una estrategia para permitir su aplicación en la recomendación de personas, en la que utilizamos los algoritmos basados en IR en dos papeles: en primer lugar, como métodos de muestreo para seleccionar un subconjunto de los usuarios apropiado para la recomendación y, tras ello, como características de los pares usuario objetivo-usuario candidato. En nuestros experimentos, observamos que las propuestas de aprendizaje de rankings son capaces de mejorar de forma consistente el acierto de los algoritmos del estado del arte y aquellos basados en la recuperación de información.

En conclusión, encontramos que los modelos de IR son efectivos en diferentes roles para la tarea de recomendación de contactos: en primer lugar, como recomendadores directos; en segundo lugar, como selectores de vecinos en recomendación basada en vecinos próximos; y, finalmente, como características y métodos de muestreo en la aplicación de métodos de aprendizaje de rankings. Nuestra investigación muestra, por tanto, que el análisis de la relación cercana que existe entre la búsqueda textual y la recomendación de interesante no es únicamente interesante desde un punto de vista teórico, sino que además tiene un gran potencial en el desarrollo de propuestas fundamentadas y efectivas.

B.1.3 Propiedades de los algoritmos efectivos para la recomendación

De manera formal, basada en axiomas, hemos estudiado las propiedades que hacen que estas propuestas sean efectivas para la tarea. Nos hemos centrado, sobre todo, en comprender una de las familias de algoritmos más amplias: los métodos que recomiendan amigos de amigos – los cuales incluyen entre ellos todas las adaptaciones directas de métodos de IR.

Hemos utilizado las conexiones con IR para adaptar los axiomas fundamentales que se aplican en el diseño de soluciones efectivas para la búsqueda textual, y hemos comprobado si son útiles para desarrollar algoritmos de recomendación de contactos más efectivos. En este caso, nos hemos limitado a estudiar métodos basados

en amigos comunes debido a las limitaciones causadas la formulación de estas propiedades. Hemos hallado que, cuando recomendamos personas a distancia 2 del usuario objetivo, recomendar usuarios candidatos con un mayor número de contactos en común con el usuario objetivo dan lugar a recomendaciones más acertadas. Además, el acierto de la recomendación mejora cuando damos pesos a los usuarios comunes de acuerdo a su grado: aquellos contactos comunes con menor grado son más capaces de discriminar las preferencias de los usuarios, por lo que deberían tener más peso. Finalmente, hemos observado que penalizar usuarios candidatos populares da lugar a pérdidas en la utilidad de los algoritmos de recomendación, dada su interferencia con el principio de enlace preferente, muy común en redes sociales (Barabási and Albert, 1999). Esta observación es consistente con el resto de experimentos, donde los métodos que penalizan fuertemente la popularidad (Jaccard, el modelo vectorial y el índice LHN global) producen recomendaciones poco útiles.

Además del análisis axiomático, hemos explorado qué vecinos debemos escoger para representar al usuario objetivo y a los usuarios candidatos si queremos maximizar su acierto en redes dirigidas. Observamos que los seguidores de los usuarios candidatos parecen describir mejor su valor que los usuarios a los que siguen. Además, la unión de todos los vecinos (seguidores y seguidos) es lo que mejor parece representar las necesidades sociales de los usuarios objetivo.

B.1.4 Efectos en la estructura de la red

Hemos investigado diversas perspectivas de utilidad complementarias al acierto – que simplemente busca incrementar la densidad de la red. Hemos explorado la definicion de métricas que cuantifican el posible efecto que recomendar nuevos enlaces tiene sobre la estructura de la red social. Se ha considerado el efecto de la recomendación sobre a) las distancias entre los pares de usuarios de la red y b) la redundancia y fuerza de sus enlaces. Además de esta perspectiva, también hemos explorado la novedad y la diversidad de las recomendaciones mediante la adaptación a nuestro problemas de métricas del campo de los sistemas de recomendación y métricas de diversidad basada en aspectos, adaptadas del campo de recuperación de información.

Mediante una comparativa de los valores de estas métricas sobre una selección relevante de algoritmos de recomendación de contactos, hemos observado que, en general, los algoritmos triviales, subóptimos y no personalizados son aquellos que obtienen los mejores valores para las métricas propuestas: en particular,enlaces recomendar gente de forma aleatoria es la mejor forma de acercar a los usuarios entre ellos, equilibrar la distribuciones de grado y los enlaces entre comunidades, mientras que recomendar usuarios muy populares es una forma de maximizar el número de enlaces que viajan entre diferentes comunidades. Esto contrasta con la novedad, diversidad y efectos positivos sobre la red de los algoritmos con mayor acierto, que alcanzan valores mucho más bajos que las propuestas no personalizadas en estas dimensiones. Entre estos métodos podemos observar que la factorización de matrices es el algoritmo que mejor equilibrio tiene entre el acierto y la diversidad, mientras que los algoritmos de filtrado colaborativo basados en memoria son los que proveen recomendaciones más inesperadas (pero relevantes) a los usuarios.

Tras esta comparativa, hemos propuesto una estrategia de optimización para promover la recomendación de usuarios que mejoren propiedades globales del sistema. Este método modifica gradualmente las recomendaciones hacia la métrica que buscamos mejorar, teniendo en consideración el nivel de acierto de las recomendaciones originales. A diferencia de las estrategias de reordenación más clásicas (Vargas et al., 2012), que reordenan las recomendaciones de cada uno de los usuarios de manera independiente, nuestro algoritmo considera las sugerencias realizadas a los demás usuarios durante la reordenación. Hemos utilizado esta estrategia para darle una aplicación práctica a la promoción de la diversidad estructural de la red mediante recomendaciones. En concreto, hemos hallado que recomendar enlaces débiles aumenta la novedad y la diversidad de información que les llega a los usuarios de las redes. Esto tiene posibles efectos positivos en la creación de medios sociales saludables y en la mitigación posibles filtros burbuja (Pariser, 2011) y cámaras de eco.

B.1.5 Recomendación interactiva

Finalmente,hemos explorado la tarea de recomendación comprendida como un proceso interactivo dentro de un bucle de realimentación. En primer lugar, hemos formalizado la tarea de recomendación interactiva, y establecido lazos con el paradigma de aprendizaje por refuerzo.

Hemos desarrollado una nueva versión de un algoritmo clásico de filtrado colaborativo: el algoritmo de vecinos próximos orientado a usuario. A diferencia de la versión original, nuestra propuesta aplica una estrategia estocástica de bandidos multi-brazo (el muestreo de Thompson) para seleccionar los vecinos del usuario objetivo. Esto da como resultado un algoritmo sensible a la incertidumbre presente en la información disponible, y explora los vecinos del usuario en mayor medida que el algoritmo kNN básico. Además, seleccionar un único vecino da lugar a una propuesta simple y compacta que funciona a lo largo del ciclo de recomendación – incluso en situaciones extremas de arranque en frío, habitualmente problemáticas para los métodos basados en filtrado colaborativo.

B.2 Trabajo futuro

Nuestros descubrimientos y desarrollos abren camino a múltiples direcciones de interés. Esbozamos aquí varias de ellas, que podrían ser exploradas para extender nuestro trabajo.

B.2.1 Más métodos de IR para la recomendación de contactos

En la Parte II, exploramos la adaptación de diferentes conceptos y técnicas de la recuperación de información a la tarea de recomendación de contactos, con resultados positivos. Estas adaptaciones nos permiten obtener conocimientos interesantes sobre el desarrollo de métodos efectivos para recomendar personas, pero no se trata de las únicas estrategias diseñadas originalmente para la tarea de busqueda que podemos considerar. Destacamos aquí algunas posibilidades para expandir esta línea de investigación.

La primera posibilidad representa la vía más directa de extender nuestra investigación, mediante la adaptación de los desarrollos algorítmicos más recientes en búsqueda textual. En particular, el uso de redes neuronales profundas para la tarea (Mitra and Craswell, 2018). Si bien la utilidad del aprendizaje profundo en la tarea de recomendación offline ha sido discutida recientemente (Dacrema et al., 2019), la cercanía entre recomendación de contactos y búsqueda, junto al éxito de estas propuestas para la recuperación de información textual en competiciones recientes como TREC (Craswell et al., 2019), aconseja, al menos, considerar esta perspectiva.

Más allá de la adaptación de modelos de aprendizaje profundo, otra posibilidad involucra explorar otras áreas de IR. Dos de ellas parecen particularmente prometedoras para la recomendación de contactos: (pseudo) *relevance feedback* (Rocchio, 1971, Ruthven and Lalmas, 2003) y reformulación de consultas (Huang et al., 2013). Dado que los métodos basados en vecinos próximos mejoran consistentemente a los modelos de IR en nuestros experimentos, esto sugiere que los vecinos de los usuarios en la red social podrían no ser su mejor representación. Por ello, aplicar estas técnicas para expandir o reformular esta representación puede producir mejoras.

B.2.2 Análisis axiomático de los sistemas de recomendación

En el Capítulo 6 hemos visto que los axiomas de recuperación de información no pueden utilizarse para analizar varios de los algoritmos más eficaces, como factorización de matrices, vecinos próximos o caminos aleatorios, dado que no recomiendan usuarios a distancia mayor que 2. Por ello, planteamos dos posibilidades para superar esta limitación.

Primero, mediante la creación de datos sintéticos podemos analizar hasta qué punto estos algoritmos son capaces de satisfacer los axiomas fundamentales de IR que adaptamos en el Capítulo 6. Esta perspectiva es similar al trabajo de Rennings et al. (2019) y Câmara and Hauff (2020), quienes construyen colecciones de documentos de texto para explorar hasta qué punto los modelos de aprendizaje profundo mantienen las propiedades de la tarea de búsqueda. Esta linea podría proporcionar una mayor comprensión de la utilidad y relevancia de las restricciones originales para el diseño de nuevas estrategias.

La segunda vía consiste en generalizar las propiedades que introducimos en el Capítulo 6, para poder aplicarlas a una familia mayor de algoritmos. Por ejemplo, se podría comenzar mediante el estudio de los axiomas de IR para pseudo *relevance feedback* (Ruthven and Lalmas, 2003), siguiendo el trabajo de Clinchant and Gaussier (2013). Hallar un conjunto adecuado de propiedades que den contribuyan al éxito de distintos algoritmos de recomendación es un objetivo deseable, que podría incluso ser extendido a otros dominios de recomendación, como la música o el cine.

Además de explorar propiedades útiles de la recomendación de contactos en general, podemos aplicar el análisis axiomático a otros aspectos (más pequeños) de la recomendación. Por ejemplo, en el Capítulo 7, estudiamos el potencial de los modelos de IR como similitudes en algoritmos de filtrado colaborativo basado en vecinos. Al igual que hacemos para los modelos de IR en nuestro trabajo, sería interesante identificar las propiedades óptimas de algoritmos de selección de vecinos del usuario objetivo y de los usuarios candidatos en kNN, siguiendo el trabajo previo de Valcarce et al. (2017, 2018).

B.2.3 Más allá del acierto

Podemos explorar más métricas y dimensiones de utilidad de las que consideramos en el Capítulo 8. Por ejemplo, podríamos analizar otras propiedades de la red social, o expandir nuestro trabajo para tratar la ecuanimidad de las estrategias de recomendación (Ekstrand et al., 2019).

Más alla de proponer nuevas métricas, podemos profundizar en las ya propuestas. En el Capítulo 9, nos centramos en analizar los beneficios potenciales que las métricas relacionadas con enlaces débiles tienen en la eliminación de filtros burbuja (Pariser, 2011). Sin embargo, todavía tenemos que comprender mejor la utilidad real de otras métricas – ya sea para las personas que utilizan la red o para otras partes interesadas, como los propietarios o administradores de la red, consumidores de datos, etc. El valor de estas propiedades podría depender del dominio de la red, pero es posible que puedan ser estudiadas con un cierto nivel de abstracción. El desarrollo de nuevas técnicas de recomendación que tengan como propósito mejorar tanto el acierto como algunas de estas métricas representa otra perspectiva que tenemos interés en explorar.

También podemos considerar una extensión de nuestro trabajo sobre difusión de información, considerando dinámicas y protocolos de comunicación diferentes al modelo de Twitter que consideramos en esta tesis. Se podría utilizar, por ejemplo, el modelo de cascada independiente (Goldenberg et al., 2001), el de umbral lineal (Kempe et al., 2003) o el protocolo *push-pull* (Demers et al., 1987, Doerr et al., 2011) para modelizar la difusión de la información. Además, en nuestros experimentos solamente hemos utilizado hashtags como representación de la temática de los contenidos. Es posible explorar otro tipo de características de la información, como pueden ser la diversidad de opinión dado un tema. Finalmente, podemos tomar como objetivo otros problemas más allá de mitigar filtros burbuja en la red, como limitar el efecto del techo de cristal (Stoica et al., 2018).

B.2.4 Recomendación interactiva

El trabajo relacionado con la recomendación interactiva que presentamos en el Capítulo 10 está planteado como un punto de introducción a esta perspectiva. Como tal, las posibilidades de expandir nuestro trabajo son muchas. En primer lugar, puesto que los desarrollos teóricos y prácticos en la recomendación interactiva son bastante recientes, hay aún espacio para el diseño y creación de mejores recomendadores basados en técnicas de aprendizaje por refuerzo – considerando no solamente bandidos multi-brazo (Lattimore and Szepesvári, 2020), sino también los (más complejos) procesos de decisión de Markov (Sutton and Barto, 2018).

También podemos variar nuestra configuración experimental. En nuestro trabajo, nos hemos limitado a recomendar un ítem cada vez, y actualizamos los recomendadores tras recibir nueva información. Sin embargo, los algoritmos de recomendación suelen actualizarse cada cierto tiempo (por ejemplo, Satuluri et al. (2020) indican que las recomendaciones de contactos en el sistema "A quien seguir" de Twitter seguir siendo relevantes durante semanas), y se suele sugerir más de un ítem. Analizar cómo varía la efectividad de los algoritmos en función de como cambia la configuración puede mejorar nuestra comprensión de la recomendación interactiva.

Otra perspectiva interesante incluye el uso de datos de entrenamiento para la recomendación interactiva, del que se puede hacer uso, por ejemplo, para seleccionar los hiperparámetros de los algoritmos. En nuestro trabajo, solamente exploramos un caso extremo de arranque en frío, en el que los algoritmos no disponen de ningún tipo de información inicial. Sin embargo, cuando un algoritmo de recomendación comienza a funcionar en una plataforma, normalmente se entrena utilizando los datos disponibles en el sistema. Si un algoritmo anterior estuviese funcionando en la plataforma hasta ese momento, es probable que estos datos estén fuertemente influenciados por el mismo (Gruson et al., 2019). Comprender las diferencias en el comportamiento cíclico de los algoritmos de recomendación en función de las propiedades de los datos de entrenamiento es por tanto un problema atractivo.

Finalmente, hasta donde sabemos, el desarrollo y la evaluación de los sistemas de recomendación ha estado fundamentalmente unido al acierto, y solo unos pocos trabajos han tenido en cuenta dimensiones de evaluación adicionales. En esta tesis, medimos la diversidad de los ítems recomendados por los diferentes sistemas a lo largo del tiempo, pero podríamos desarrollar más métricas de novedad y diversidad para este problema. Además, sería interesante desarrollar estrategias de bandidos multi-brazo que optimicen conjuntamente dichas dimensiones junto al acierto del sistema, siguiendo a Li et al. (2020), Mehrotra et al. (2020).

B.2.5 Efectos de la recomendación interactiva en la estructura de la red

En nuestro trabajo, hemos estudiado, por separado, los posibles efectos que la recomendación tiene sobre las propiedades de la red, y la influencia de las preferencias del usuario en el proceso cíclico de aprendizaje de los sistemas de recomendación. Sin embargo, ambos procesos están intercalados en aplicaciones reales. Por ello, proponemos como perspectiva futura el análisis de los cambios a largo plazo en las propiedades estructurales de la red, mediante la incorporación de diversos recomendadores (tanto estocásticos como deterministas) en el bucle de realimentación para la recomendación interactiva de contactos. Un antecesor de esta propuesta se puede encontrar en el trabajo de Farajtabar et al. (2017), quienes estudian la relación entre el proceso evolutivo de las redes sociales y los cambios en la difusión de información.

B.2.6 Estudios de usuario y evaluación online

Los experimentos de esta tesis se han limitado a procedimientos de evaluación offline, en la que la salida de los diferentes sistemas de recomendación se analiza en comparación con un conjunto de muestras de red social. Si bien la mayor parte de la investigación académica en sistemas de recomendación se basa en la evaluación offline, ésta tiende a presentar sesgos, y, cuando nos pasamos al sistema en producción, suelen aparecer diferencias (Rossetti et al., 2016). Por ello, realizar estudios con usuarios y/o tests A/B podría complementar nuestros experimentos con más perspectivas objetivas.

Además, en los Capítulos 8 y 9, hemos considerado la diversidad de las recomendaciones (y del flujo de información) como propiedades positivas, de acuerdo con la investigación previa en ciencia de redes, sistemas de recomendación y recuperación de informació (Burt, 1995, Granovetter, 1973, Ugander et al., 2012, Vargas and Castells, 2014, Vargas et al., 2012). Los estudios de usuario son una herramienta que nos permitiría conocer la visión real que los usuarios de las plataformas de red social tienen sobre los beneficios de promover estas propiedades, de forma similar a Ekstrand et al. (2014), Kotkov et al. (2018), quienes exploraron las relaciones de la diversidad, novedad y serendipia de las recomendaciones con la satisfacción de los usuarios. Estos estudios pueden llevarnos a hallar más aplicaciones prácticas (quizá, inesperadas) de la diversificación de propiedades de las redes.

C Parameter selection

As a complement to the different experiments we report throughout the thesis (in Chapters 4, 5, 6 and 7), we detail in this appendix details about the experimental settings of our experiments. In particular, to increase their reproducibility, we report here the optimal hyperparameter selection for the different experiments and datasets. We separate the hyperparameter selection depending on the chapter where the experiments were described and analyzed.

C.1 Hyperparameter selection for Chapter 4

The first set of experiments we report is the algorithmic comparison in Chapter 4, where we compare several state of the art contact recommendation approaches in terms of accuracy. To obtain the optimal hyperparameters for such experiments, using the training and validation networks introduced in Section 4.3, we tune the algorithms using a simple grid search optimizing the nDCG metric (Järvelin and Kekäläinen, 2002) at cutoff 10. We show the set of values considered for the grid search in Table C.1.

Table C.1: Explored grid for tuning each hyperparameter in our experiments.

Algorithm	Possible hyperparameters
Katz	$\beta \in \{0.1, 0.2, \ldots, 0.8, 0.9\}$
LPI	$\beta \in \{0.1, 0.2, \ldots, 0.8, 0.9\}$
	$n \in \{3, 4, 5\}$
Global LHN	$\phi \in \{0.1, 0.2, \ldots, 0.8, 0.9\}$
Pers. PageRank	$r \in \{0.1, 0.2, \ldots, 0.8, 0.9, 0.99\}$
Money	authorities, hubs
	$\alpha \in \{0.1, 0.2, \ldots, 0.8, 0.9, 0.99\}$
PropFlow	$l \in \{2, 3, 4, 5\}$
iMF	$k \in \{10, 20, \ldots, 290, 300\}$
	$\alpha \in \{10, 40, 100\}$
	$\lambda \in \{10, 100, 150\}$
User-based kNN	$k \in \{10, 20, \ldots, 290, 300\}$
Item-based kNN	$k \in \{10, 20, \ldots, 290, 300\}$

Beyond the values we report in that table, for the directed networks we consider different variants depending on the directionality of the edges: for the neighbors algorithms (MCN, Adamic-Adar, Jaccard, cosine similarity and resource allocation) we consider different neighborhood selections: one for the target user, Γ^q and another one for the candidate users Γ^d. In the case of Adamic-Adar and resource allocation, we have to set yet another orientation, Γ^l, which is the one considered for the common neighbors in $\Gamma^q(u) \cap \Gamma^d(v)$. In the path-based approaches (distance, Katz and local path index), we also apply an orientation selection: we can take the natural orientation of the edges (i.e. the paths between the target and candidate users), which we denote as $\Gamma := \Gamma_{out}$, the paths from the candidate to the target user (denoted $\Gamma := \Gamma_{in}$) or the paths between both networks in an undirected network ($\Gamma := \Gamma_{in}$). The only exception to this is the global LHN index: since we need to compute eigenvalues, we ensure that they are exist and take real values by considering the network as undirected. Finally, in the Twitter interactions networks, we have to decide whether use the frequency of interactions as weights or ignore them in several algorithms: cosine similarity, PropFlow and the collaborative filtering approaches (iMF, user-based and item-based kNN).

The optimal hyperparameters are reported in Table C.2. For the Twitter directed networks, we take $\Gamma^q := \Gamma_{und}$ and $\Gamma^d := \Gamma_{in}$ for all friends of friends algorithms, except for Adamic-Adar and resource allocation

Table C.2: Parameter setting for the experimental comparison in Chapter 4.

	Twitter 1-month		Twitter 200-tweets		Facebook
Algorithm	Interactions	Follows	Interactions	Follows	
Adamic-Adar	$\Gamma^l := \Gamma_{und}$	$\Gamma^l := \Gamma_{und}$	$\Gamma^l := \Gamma_{und}$	$\Gamma^l := \Gamma_{und}$	–
Resource Allocation	$\Gamma^l := \Gamma_{und}$	$\Gamma^l := \Gamma_{und}$	$\Gamma^l := \Gamma_{und}$	$\Gamma^l := \Gamma_{und}$	–
Distance	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	–
Katz	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\beta = 0.1$
	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.4$	
Global LHN	$\phi = 0.1$	$\phi = 0.2$	$\phi = 0.1$	$\phi = 0.4$	$\phi = 0.1$
Local path index	$\Gamma := \Gamma_{und}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{und}$	$\Gamma := \Gamma_{out}$	$\beta = 0.1$
	$\beta = 0.3$	$\beta = 0.1$	$\beta = 0.1$	$\beta = 0.1$	$n = 3$
	$n = 3$	$n = 3$	$n = 3$	$n = 3$	
Pers. PageRank	$r = 0.4$	$r = 0.6$	$r = 0.5$	$r = 0.9$	$r = 0.6$
Hitting time	$r = 0.9$	$r = 0.5$	$r = 0.9$	$r = 0.9$	$r = 0.1$
Commute time	$r = 0.9$	$r = 0.5$	$r = 0.9$	$r = 0.9$	$r = 0.1$
PropFlow	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$\Gamma := \Gamma_{out}$	$l = 3$
	$l = 5$	$l = 2$	$l = 4$	$l = 2$	
Money	Authorities	Authorities	Authorities	Authorities	Hubs
	$\alpha = 0.99$	$\alpha = 0.9$	$\alpha = 0.99$	$\alpha = 0.99$	$\alpha = 0.7$
Twittomender	$\Gamma := \Gamma_{in}$	$\Gamma := \Gamma_{in}$	$\Gamma := \Gamma_{in}$	$\Gamma := \Gamma_{in}$	–
iMF	$k = 300$	$k = 260$	$k = 260$	$k = 290$	$k = 290$
	$\lambda = 150$	$\lambda = 100$	$\lambda = 100$	$\lambda = 150$	$\lambda = 100$
	$\alpha = 40$	$\alpha = 10$	$\alpha = 40$	$\alpha = 40$	$\alpha = 10$
User-based kNN	$k = 100$	$k = 40$	$k = 80$	$k = 80$	$k = 100$
Item-based kNN	$k = 290$	$k = 300$	$k = 290$	$k = 290$	$k = 300$

Table C.3: Explored grid for tuning each hyperparameter in our experiments.

Algorithm	Possible hyperparameters
BM25	$b \in \{0.1, 0.2, \ldots, 0.8, 0.9, 0.999, 1.0\}$
	$k \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$
Extreme BM25	$b \in \{0.1, 0.2, \ldots, 0.8, 0.9, 0.999, 1.0\}$
QLD	$\mu \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$
QLJM	$\lambda \in \{0.1, 0.2, \ldots, 0.8, 0.9, 0.999, 1.0\}$
QLL	$\gamma \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$
PL2	$c \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$

for the 1-month follows graph, where we take $\Gamma^q := \Gamma_{out}$. In the interactions networks, PropFlow and cosine similarity make use of the weights, whereas the collaborative filtering approaches use their unweighted versions.

C.2 Hyperparameter selection for Chapters 5 and 6

Next, we report the optimal parameter setting for the experiments with standalone information retrieval models that we introduce in chapters 5 and 6. As we report in those chapters, the hyperparameters are by optimizing nDCG@10 (Järvelin and Kekäläinen, 2002) over a simple grid search procedure. As this is the same procedure we do for the experiments in Chapter 4, we only report here the parameters for the new algorithms. For the rest, just take the values reported in Tables C.1 and C.2. First, in Table C.3, we show the points considered for the grid search over the IR algorithms.

Again, we do not report link orientations in such table, but, in the directed Twitter networks, we have to choose the orientations Γ^d and Γ^q for all the IR approaches. In the case of BM25 and Extreme BM25, we also have to set the orientation Γ^l. Finally, for the Twitter interactions graphs, we have to decide whether or not to use edge weights (measured as the frequency of interaction between two users in the training graph) for all the IR models.

Table C.4: Parameter settings for the experiments with standalone approaches. In Adamic-Adar, Γ^l represents the direction in the selection of common neighbors between the target and candidate users (see Sanz-Cruzado and Castells (2018a)). In the 1-month interactions graph, all algorithms perform best without weights, except for BM25 and VSM, and, in the 200-tweets interactions graph, the only algorithm that uses its weighted version is BM25.

	Twitter 1-month		Twitter 200-tweets		Facebook
Algorithm	Interactions	Follows	Interactions	Follows	
BM25	$\Gamma^l := \Gamma_{out}$ $b = 0.1$ $k = 0.1$	$\Gamma^l := \Gamma_{out}$ $b = 0.999$ $k = 0.01$	$\Gamma^l := \Gamma_{out}$ $b = 0.4$ $k = 1$	$\Gamma^l := \Gamma_{out}$ $b = 0.1$ $k = 10$	$b = 0.8$ $k = 0.001$
Extreme BM25	$\Gamma^l := \Gamma_{out}$ $b = 0.1$	$\Gamma^l := \Gamma_{out}$ $b = 0.1$	$\Gamma^l := \Gamma_{out}$ $b = 0.1$	$\Gamma^l := \Gamma_{out}$ $b = 0.2$	$b = 0.1$
QLD	$\mu = 1,000$	$\mu = 1,000$	$\mu = 1,000$	$\mu = 1,000$	$\mu = 1,000$
QLJM	$\lambda = 0.999$	$\lambda = 0.999$	$\lambda = 0.999$	$\lambda = 0.999$	$\lambda = 0.999$
QLL	$\gamma = 100$	$\gamma = 100$	$\gamma = 1,000$	$\gamma = 100$	$\gamma = 1,000$
PL2	$c = 10,000$	$c = 10,000$	$c = 1,000$	$c = 100$	$c = 10,000$

We then report the optimal hyperparameters for the experiments in both chapters in Table C.4. For the directed graphs, we take $\Gamma^q := \Gamma_{und}$ and $\Gamma^d := \Gamma_{in}$ for all friends of friends and IR algorithms, except for Adamic-Adar, BM25 and QLJM for the 1-month follows graph, where we take $\Gamma^q := \Gamma_{out}$ and VSM in the 200-tweets interactions graph and Extreme BM25 in the 200-tweets follows graph, where we take $\Gamma^d := \Gamma_{und}$. In the interaction networks, we only consider weights in BM25 and VSM for the Twitter 1-month network and the BM25 for the Twitter 200-tweets one.

C.3 Hyperparameter selection for Chapter 7

In this section, we report how the grid search is applied in the experiments in chapter 7 for the experiments that compare the different kNN variants with friends of friends and information retrieval models. For the newly defined user-based and item-based approaches, we tune the similarities hyperparameters using the sets of parameters for the friends of friends approaches described in Tables C.1 and C.3. As we report in Table C.1, we allow the number of neighbors of the target/candidate users to go from 10 to 300 by steps of 10 neighbors. Considering that, we show the optimal parameters for the 1-month dataset in Table C.5, for the 200-tweets dataset in Table C.6, and, finally, for the Facebook dataset in C.7.

Table C.5: Parameter selection for the kNN approaches in 1-month dataset. k represents the size of the neighborhood. In the interactions graph, all user-based models use their unweighted version, and, for the item-based kNN, all models use the weighted version excepting the query likelihood models, Extreme BM25, DLH and cosine similarity.

	Twitter 1-month interactions								Twitter 1-month follows							
	User-based kNN				Item-based kNN				User-based kNN				Item-based kNN			
Similarity	Γ^q	Γ^d	Parameters	k	Γ^q	Γ^d	Parameters	k	Γ^q	Γ^d	Parameters	k	Γ^q	Γ^d	Parameters	k
BIR	Γ_{out}	Γ_{und}	–	110	Γ_{out}	Γ_{in}	–	30	Γ_{und}	Γ_{und}	–	210	Γ_{out}	Γ_{in}	–	30
BM25	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.7$ $k = 10$	90	Γ_{und}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{in}$ $b = 0.2$ $k = 1$	290	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.8$ $k = 1,000$	60	Γ_{in}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.3$ $k = 1$	280
Extreme BM25	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.6$	80	Γ_{in}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.2$	280	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.7$	60	Γ_{in}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.2$	280
QLD	Γ_{und}	Γ_{und}	$\mu = 100$	90	Γ_{in}	Γ_{und}	$\mu = 1,000$	210	Γ_{und}	Γ_{und}	$\mu = 1,000$	120	Γ_{in}	Γ_{und}	$\mu = 1,000$	210
QLJM	Γ_{und}	Γ_{und}	$\lambda = 0.1$	100	Γ_{und}	Γ_{und}	$\lambda = 0.9$	200	Γ_{und}	Γ_{und}	$\lambda = 0.1$	40	Γ_{in}	Γ_{und}	$\lambda = 0.9$	200
QLL	Γ_{und}	Γ_{in}	$\gamma = 1,000$	10	Γ_{in}	Γ_{in}	$\gamma = 1,000$	10	Γ_{out}	Γ_{in}	$\gamma = 1,000$	10	Γ_{in}	Γ_{in}	$\gamma = 0.001$	10
DFRee	Γ_{und}	Γ_{und}	–	60	Γ_{und}	Γ_{in}	–	290	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{in}	–	290
DFReeKLIM	Γ_{out}	Γ_{out}	–	80	Γ_{und}	Γ_{in}	–	290	Γ_{und}	Γ_{und}	–	30	Γ_{in}	Γ_{in}	–	240
DLH	Γ_{out}	Γ_{out}	–	90	Γ_{in}	Γ_{in}	–	280	Γ_{und}	Γ_{und}	–	30	Γ_{in}	Γ_{in}	–	280
DPH	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{in}	–	190	Γ_{und}	Γ_{und}	–	40	Γ_{in}	Γ_{in}	–	150
PL2	Γ_{und}	Γ_{und}	$c = 100$	110	Γ_{und}	Γ_{in}	$c = 10$	290	Γ_{und}	Γ_{und}	$c = 0.1$	40	Γ_{und}	Γ_{und}	$c = 10$	250
VSM	Γ_{out}	Γ_{und}	–	100	Γ_{in}	Γ_{und}	–	290	Γ_{und}	Γ_{out}	–	50	Γ_{in}	Γ_{und}	–	290
Adamic-Adar	Γ_{out}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$	110	Γ_{und}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{out}$	40	Γ_{out}	Γ_{out}	$\Gamma^l \coloneqq \Gamma_{in}$	90	Γ_{out}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{in}$	50
Cosine	Γ_{out}	Γ_{out}	–	100	Γ_{und}	Γ_{in}	–	300	Γ_{und}	Γ_{und}	–	40	Γ_{in}	Γ_{in}	–	290
Jaccard	Γ_{out}	Γ_{out}	–	100	Γ_{und}	Γ_{in}	–	300	Γ_{und}	Γ_{und}	–	40	Γ_{in}	Γ_{in}	–	290
MCN	Γ_{und}	Γ_{und}	–	100	Γ_{und}	Γ_{in}	–	60	Γ_{und}	Γ_{und}	–	100	Γ_{out}	Γ_{in}	–	30

Table C.6: Parameter selection for the kNN approaches in the 200-tweets dataset. For the interactions network, all user-based models use their unweighted version, and, for the item-based kNN, all models use the weighted version excepting the query likelihood models, BIR, Extreme BM25, Jaccard and cosine similarity.

	Twitter 1-month interactions								Twitter 1-month follows							
	User-based kNN				Item-based kNN				User-based kNN				Item-based kNN			
Similarity	Γ^q	Γ^d	Parameters	k	Γ^q	Γ^d	Parameters	k	Γ^q	Γ^d	Parameters	k	Γ^q	Γ^d	Parameters	k
BIR	Γ_{und}	Γ_{und}	–	80	Γ_{und}	Γ_{in}	–	130	Γ_{out}	Γ_{out}	–	40	Γ_{in}	Γ_{und}	–	30
BM25	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.8$ $k = 10$	70	Γ_{und}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.9$ $k = 1$	150	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.9$ $k = 10$	40	Γ_{in}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.6$ $k = 1$	290
Extreme BM25	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.7$	70	Γ_{und}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.1$	300	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.7$	30	Γ_{in}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$ $b = 0.2$	290
QLD	Γ_{und}	Γ_{und}	$\mu = 100$	150	Γ_{in}	Γ_{und}	$\mu = 1,000$	260	Γ_{und}	Γ_{und}	$\mu = 100$	60	Γ_{in}	Γ_{und}	$\mu = 1,000$	210
QLJM	Γ_{und}	Γ_{und}	$\lambda = 0.1$	60	Γ_{und}	Γ_{in}	$\lambda = 0.2$	170	Γ_{und}	Γ_{und}	$\lambda = 0.1$	60	Γ_{und}	Γ_{und}	$\lambda = 0.1$	210
QLL	Γ_{out}	Γ_{in}	$\gamma = 0.001$	10	Γ_{in}	Γ_{in}	$\gamma = 10$	10	Γ_{und}	Γ_{und}	$\gamma = 0.001$	10	Γ_{und}	Γ_{in}	$\gamma = 100$	10
DFRee	Γ_{und}	Γ_{und}	–	60	Γ_{in}	Γ_{in}	–	130	Γ_{out}	Γ_{out}	–	40	Γ_{in}	Γ_{in}	–	170
DFReeKLIM	Γ_{und}	Γ_{und}	–	60	Γ_{in}	Γ_{und}	–	200	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{in}	–	170
DLH	Γ_{und}	Γ_{und}	–	60	Γ_{in}	Γ_{und}	–	270	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{und}	–	90
DPH	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{in}	–	190	Γ_{out}	Γ_{out}	–	40	Γ_{in}	Γ_{in}	–	40
PL2	Γ_{und}	Γ_{und}	$c = 0.1$	110	Γ_{und}	Γ_{in}	$c = 1,000$	200	Γ_{und}	Γ_{und}	$c = 0.1$	40	Γ_{in}	Γ_{in}	$c = 10$	140
VSM	Γ_{und}	Γ_{out}	–	50	Γ_{in}	Γ_{in}	–	220	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{und}	–	220
Adamic-Adar	Γ_{und}	Γ_{und}	$\Gamma^l \coloneqq \Gamma_{und}$	70	Γ_{und}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{und}$	220	Γ_{out}	Γ_{out}	$\Gamma^l \coloneqq \Gamma_{in}$	40	Γ_{out}	Γ_{in}	$\Gamma^l \coloneqq \Gamma_{in}$	50
Cosine	Γ_{und}	Γ_{out}	–	80	Γ_{in}	Γ_{und}	–	290	Γ_{und}	Γ_{out}	–	50	Γ_{in}	Γ_{in}	–	300
Jaccard	Γ_{und}	Γ_{und}	–	70	Γ_{und}	Γ_{in}	–	290	Γ_{und}	Γ_{und}	–	50	Γ_{in}	Γ_{in}	–	100
MCN	Γ_{und}	Γ_{und}	–	40	Γ_{und}	Γ_{in}	–	40	Γ_{out}	Γ_{out}	–	30	Γ_{out}	Γ_{in}	–	50

Table C.7: Parameter selection for the kNN approaches in the Facebook network.

	User-based kNN		Item-based kNN	
Similarity	Parameters	k	Parameters	k
BIR	–	70	–	240
BM25	$b = 0.9$ $k = 100$	20	$b = 0.2$ $k = 0.01$	230
Extreme BM25	$b = 0.9$	20	$b = 0.6$	270
QLD	$\mu = 100$	60	$\mu = 10$	20
QLJM	$\lambda = 0.1$	30	$\lambda = 0.1$	280
QLL	$\gamma = 0.001$	10	$\gamma = 1,000$	10
DFRee	–	30	–	250
DFReeKLIM	–	30	–	260
DLH	–	30	–	270
DPH	–	30	–	270
PL2	$c = 1$	30	$c = 1,000$	230
VSM	–	70	–	280
Adamic-Adar	–	60	–	290
Cosine	–	30	–	300
Jaccard	–	30	–	210
MCN	–	60	–	280

C.4 Hyperparameter selection for Chapter 10

Finally, we report here the optimal parameter selection for the different interactive recommendation strategies we study in Chapter 10. Table C.8 contains the optimal parameters for all the approaches in the comparison for all five datasets: Foursquare NY, Foursquare Tokyo, MovieLens 1M and the follows networks for the Twitter 1-month and Twitter 200-tweets datasets. Beyond the parameters in such table, for the CLUB algorithm (Gentile et al., 2014), we initialize the graph of relations between users using an Erdös-Renyi random graph (Erdös and Rényi, 1959). The probability of creating a link between two nodes is the same used in the paper introducing CLUB: $p = 3 \cdot \log(|\mathcal{U}|)/|\mathcal{U}|$.

Table C.8: Parameter setting for the different datatetsin the multi-armed bandit experiments.

Algorithm	Foursquare NY	Foursquare Tokyo	MovieLens 1M	Twitter 1-month	Twitter 200-tweets
ε-greedy	$\varepsilon = 0.05$	$\varepsilon = 0.2$	$\varepsilon = 0.1$	$\varepsilon = 0.05$	$\varepsilon = 0.05$
Thompson sampling	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$
	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 100$
InterPMF	$k = 2$	$k = 5$	$k = 3$	$k = 5$	$k = 10$
	$\sigma_p^2 = 10$	$\sigma_p^2 = 0.1$	$\sigma_p^2 = 10$	$\sigma_p^2 = 1$	$\sigma_p^2 = 10$
	$\sigma_q^2 = 10$	$\sigma_q^2 = 0.1$	$\sigma_q^2 = 10$	$\sigma_q^2 = 0.1$	$\sigma_q^2 = 1$
	$\sigma^2 = 0.1$	$\sigma^2 = 0.1$	$\sigma^2 = 0.1$	$\sigma^2 = 0.1$	$\sigma^2 = 1$
	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.1$	$\varepsilon = 0.2$	$\varepsilon = 0.1$
CLUB	$\alpha = 0.01$	$\alpha = 0.01$	$\alpha = 0.01$	$\alpha = 0.1$	$\alpha = 0.01$
	$\gamma = 5$	$\gamma = 5$	$\gamma = 5$	$\gamma = 5$	$\gamma = 5$
kNN bandit ($k = 1$)	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$
	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 100$
kNN bandit ($k > 1$)	$k = 10$	$k = 100$	$k = 10$	$k = 10$	$k = 10$
	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$	$\alpha_0 = 1$
	$\beta_0 = 10$	$\beta_0 = 100$	$\beta_0 = 100$	$\beta_0 = 10$	$\beta_0 = 100$
UB kNN	$k = 100$	$k = 100$	$k = 100$	$k = 100$	$k = 100$
iMF	$k = 20$	$k = 10$	$k = 10$	$k = 10$	$k = 10$
	$\alpha = 1$	$\alpha = 10$	$\alpha = 10$	$\alpha = 1$	$\alpha = 10$
	$\lambda = 0.1$	$\lambda = 10$	$\lambda = 10$	$\lambda = 0.1$	$\lambda = 10$

D
Statistical significance

In recommender systems and information retrieval, along the raw results, it is common to check whether the differences between algorithms can be caused by the variance of the system. The tools we use to perform this inspections are known as statistical hypothesis tests. In this appendix, we give full detail of the statistical significance tests we run for the reported accuracy metrics in our experiments in Chapters 4, 5 and 7. We first explain how these tests work, and, afterwards, we provide the results for the different chapters.

D.1 Description of the statistical tests

Offline evaluation tests compare several algorithms, $s_1, ..., s_n$ in terms of a metric, we denote as m. For the most common accuracy metrics, such as precision, recall or nDCG, given a system s, we compute a metric value for each target user u in the system, $m_u(s)$ and then, we provide a unique value for the system by averaging these values. We denote this average value as $\overline{m}(s)$:

$$\overline{m}(s) = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} m_u(s) \tag{D.1}$$

For all metrics which can be formulated this way, we can use statistical hypothesis tests to determine whether the difference between two algorithms s_1, s_2 in terms of the metric is large enough to affirm that it is not observed by chance. For this, we consider the metric m for the system s, $m(s)$, to be a random variable with mean μ_s, and the value for each user, $m_u(s)$, to be a sample from that random variable. Then, given two systems, s_1, s_2, we elaborate two hypothesis: the first hypothesis is the null hypothesis, H_0, which represents the default conclusion of the test: unless enough evidence is shown, H_0 is considered the truth. In recommender systems, it is common to take as null hypothesis that the expected value of the metric for both systems is the same, i.e. $H_0 : \mu_{s_1} = \mu_{s_2}$. The second hypothesis, known as the alternative hypothesis, H_1, suggests another possible relation between the two algorithms in terms of the metric – a common case in the field is to consider as the alternative hypothesis $H_1 : \mu_{s_1} \neq \mu_{s_2}$.

Then, statistical tests estimate how probable the results are given the null hypothesis: $p(m(s_1), m(s_2)|H_0)$. This probability is known as the p-value and, the lower it is, the higher the evidence against the null hypothesis is. Given a threshold value $\alpha \in [0, 1]$ we say that we reject the null hypothesis in favor of the alternative hypothesis with a confidence level of $(1 - \alpha)$ if $p(m(s_1), m(s_2)|H_0) < \alpha$. When this occurs, we also say that the difference between s_1 and s_2 in terms of m is statistically significant at p-value $p < \alpha$. In case $p(m(s_1), m(s_2)|H_0) \geq \alpha$, we cannot reject the null hypothesis, and, therefore, we consider that the difference between the systems is not significant (and due to variance).

Once these basic definitions have been provided, we proceed to the description of specific methods. In our experiments, we run two different tests: first, we run a simple two-tailed paired t-test, and, afterwards, to deal with multiple comparisons, we use a Tukey Honest Statistical Difference (HSD) test (Tukey, 1949). We describe next how they work.

D.1.1 Paired t-test

The paired t-test (Sakai, 2018) is an statistical test that only considers two systems. In this test, we consider the values of the metric for each user as independent samples from a normal distribution that depends on the system, i.e. $m_u(s) \sim \mathcal{N}(\mu_{s_i}, \sigma^2_{s_i})$ for $i = \{1, 2\}$ where μ_s is the average value of the $m(s)$ random variable and σ^2_s its variance. As we assume that the metric values for both systems are independent and follow normal

distributions, if we take the differences between metrics for the same users, $d_u = m_u(s_1) - m_u(s_2)$, we can also assume that they follow a Gaussian distribution $d_u \sim \mathcal{N}(\mu_{s_1} - \mu_{s_2}, \sigma^2_{s_1} + \sigma^2_{s_2})$.

Letting the null hypothesis be $H_0 : \mu_{s_1} - \mu_{s_2} = 0$ we just need then to find the probability that the sample mean

$$\overline{d} = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} d_u \tag{D.2}$$

is equal to zero. For this, we use the following fact: if we have n samples, $y_1, ...y_n$, independent and identically distributed, extracted from a normal distribution $Y \sim \mathcal{N}(\mu, \sigma^2)$, the following random variable follows a Student's t distribution with $n-1$ freedom degrees $t(n-1)$:

$$t = \frac{\overline{y} - \mu}{\sqrt{\sigma^2_y / n}} \sim t(n-1) \tag{D.3}$$

where $\overline{y}$ is the sample mean and σ^2_y represents the sample variance. The Student's t distribution we find represents the probability distribution of the sampled mean given the real one.

Considering (as the null hypothesis indicates) that the real mean $\mu_{s_1} - \mu_{s_2}$ is 0, we have different possibilities, depending on which alternative hypothesis we take. In our experiments, we consider $H_1 : \mu_{s_1} \neq \mu_{s_2}$ (what is called a two-tailed t-test), which is run by computing the p-value p as:

$$p = p_t \left(|x| \geq \left| \frac{\overline{d}}{\sqrt{\sigma^2_d / n}} \right| ; |\mathcal{U}| - 1 \right) \tag{D.4}$$

where $p_t(z; n)$ is the probability of condition z in a Student's t distribution with n degrees of freedom, and the sample variance is computed as:

$$\sigma^2_d = \frac{1}{|\mathcal{U}| - 1} \sum_{u \in \mathcal{U}} \left(\overline{d} - d_u \right)^2 \tag{D.5}$$

but there are other tests, which consider, for example $H_1 : \mu_1 > \mu_2$ as the alternative hypothesis.

D.1.2 Tukey Honest Statistical Difference tests

When we execute a t-test with p-value equal to α between two systems, the probability that we accept as significant a difference that it is not is, at most, α. However, if we execute k independent t-tests (one for each pair of users), the probability of having one of those errors grows up to $1 - (1 - \alpha)^k$ (Carterette, 2012). In order to address this problem, and limit the error rate to α, several methods have been proposed (Carterette, 2012, Sakai, 2018).

In this thesis, we have considered one of these methods, complementing the results for our paired t-tests: the Tukey's range test or Tukey HSD test (Tukey, 1949). The idea behind this is that, if the largest mean difference between systems is not significant, then, none of them is. Therefore, instead of finding a distribution that depends on the different between the two tested systems (as we do for the t-test) we find a distribution for checking the null hypothesis that depends on that maximum difference – known as the range. If the error between the two systems is at most α, then, the upper limit of overall error of all our comparisons is also α. Now, given a set of systems $S = \{s_1, s_2, ..., s_n\}$, and the following ψ^2 value

$$s^2 = \frac{1}{|S|(|\mathcal{U}| - 1)} \sum_{s \in S} \sum_{u \in \mathcal{U}} \left(m_{su} - \overline{(m)}_s \right)^2 \tag{D.6}$$

we have that

$$t = \frac{\max_{s \in S} \overline{m}_s - min_{s \in S} \overline{m}_s}{\sqrt{2\psi^2 / |\mathcal{U}|}} \sim \text{Tukey}(|S|, |S|(|\mathcal{U}| - 1)) \tag{D.7}$$

where Tukey(n,m) represents a Tukey distribution with n indicating the number of compared systems and m the degrees of freedom. Under this distribution, the p-value p for two systems s_1 and s_2 is found as:

$$p = p_{\text{Tukey}}\left(|x| \geq \left|\frac{\overline{m}_{s_1} - \overline{m}_{s_2}}{\sqrt{\psi_d^2/n}}\right|; |S|, |S|(|\mathcal{U}| - 1)\right) \tag{D.8}$$

D.2 Results

Complementing the experimental results shown in the thesis, in this section, we give full detail of the statistical significance tests we run for the reported metrics. As we state earlier, for each pair of algorithms, we run two different statistical tests: first, we run a simple two-tailed paired t-test, and then, to deal with multiple comparisons, a Tukey Honest Statistical Difference (HSD) test (Tukey, 1949). In all our experiments, we apply $p < 0.05$ as the significance threshold.

D.2.1 Results for Chapter 4

The statistical significance of the results shown in Table 4.3 for the comparison of standalone IR models with other state of the art approaches are reported in Figures D.1 and D.2 for the 1-month dataset, Figures D.3 and D.4 for the 200-tweets dataset and Figure D.5 for the Facebook dataset.

The figures show the significance of each comparison in a system-to-system matrix, where colors indicate whether or not the difference between the corresponding pair of algorithms is statistically significant for the corresponding metric. The code of the colors is the following:

- A blue cell indicates that the difference is significant by both the paired t-test and the Tukey HSD test.
- A light orange cell indicates that the difference is significant by the paired t-test but not by the Tukey HSD.
- And a red cell indicates a non-significant difference by both tests.

D.2.2 Results for Chapter 5

Following the same scheme as we take in the previous section, we report here the statistical significance tests for Chapter 5. The results of the different tests run for the results in Table 5.6 are reported in Figures D.6 and D.7 for the 1-month dataset, Figures D.8 and D.9 for the 200-tweets dataset and Figure D.10 for the Facebook dataset.

D.2.3 Results for Chapter 7

Finally, we report the results for our experiments in Chapter 7. The significance of the results shown in Table 7.2 in Section 7.2, comparing user-based kNN approaches with the best standalone approaches, are reported in Figures D.11,D.12,D.13, D.14 and D.15. Finally, Figure D.16 shows the significance of the comparisons between the LambdaMart learning to rank approach (Burges, 2010, Ganjisaffar et al., 2011) with implicit matrix factorization (Hu et al., 2008), user-based kNN with BM25 similarity and personalized PageRank (White and Smyth, 2003) (the best approaches in Tables 5.6 and 7.2) reported in Figure 7.5.

D.2.4 Discussion

In our results, the Tukey HSD test is strictly more conservative than the t-test: when a difference is significant for the Tukey HSD test, it is also significant according to the paired t-test. Consequently, it is enough to use three colors to cover all the observed cases in our experiments. Despite the more conservative nature of the Tukey HSD test, both tests broadly agree on the main conclusions of our experiments. The only discrepancies that we find worth to mention is that the comparison between nDCG@10 between iMF and BM25 in Tables 5.6

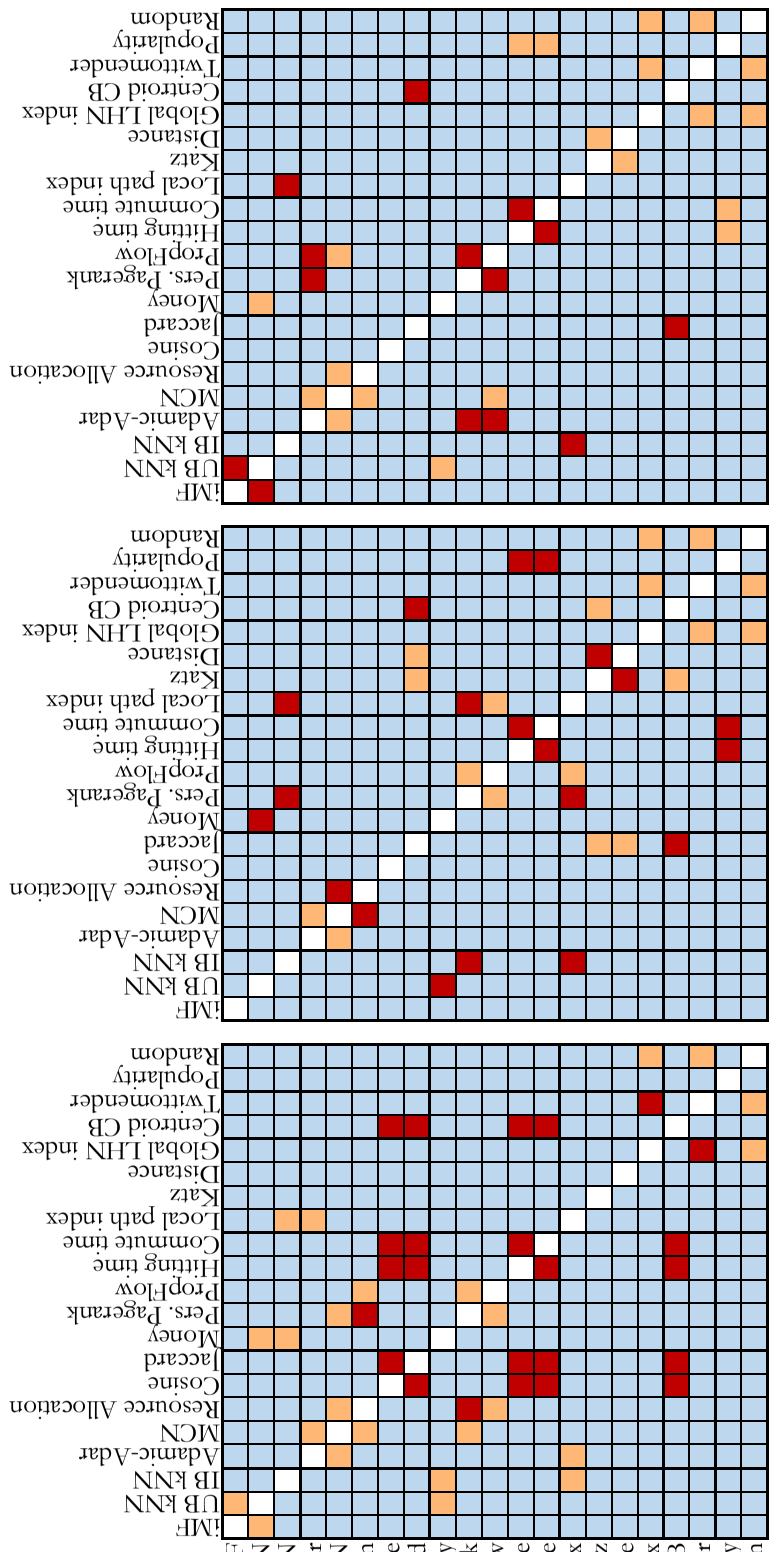


Figure D.1: Statistical significance of comparisons in Table 4.3 for the 1-month interactions dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

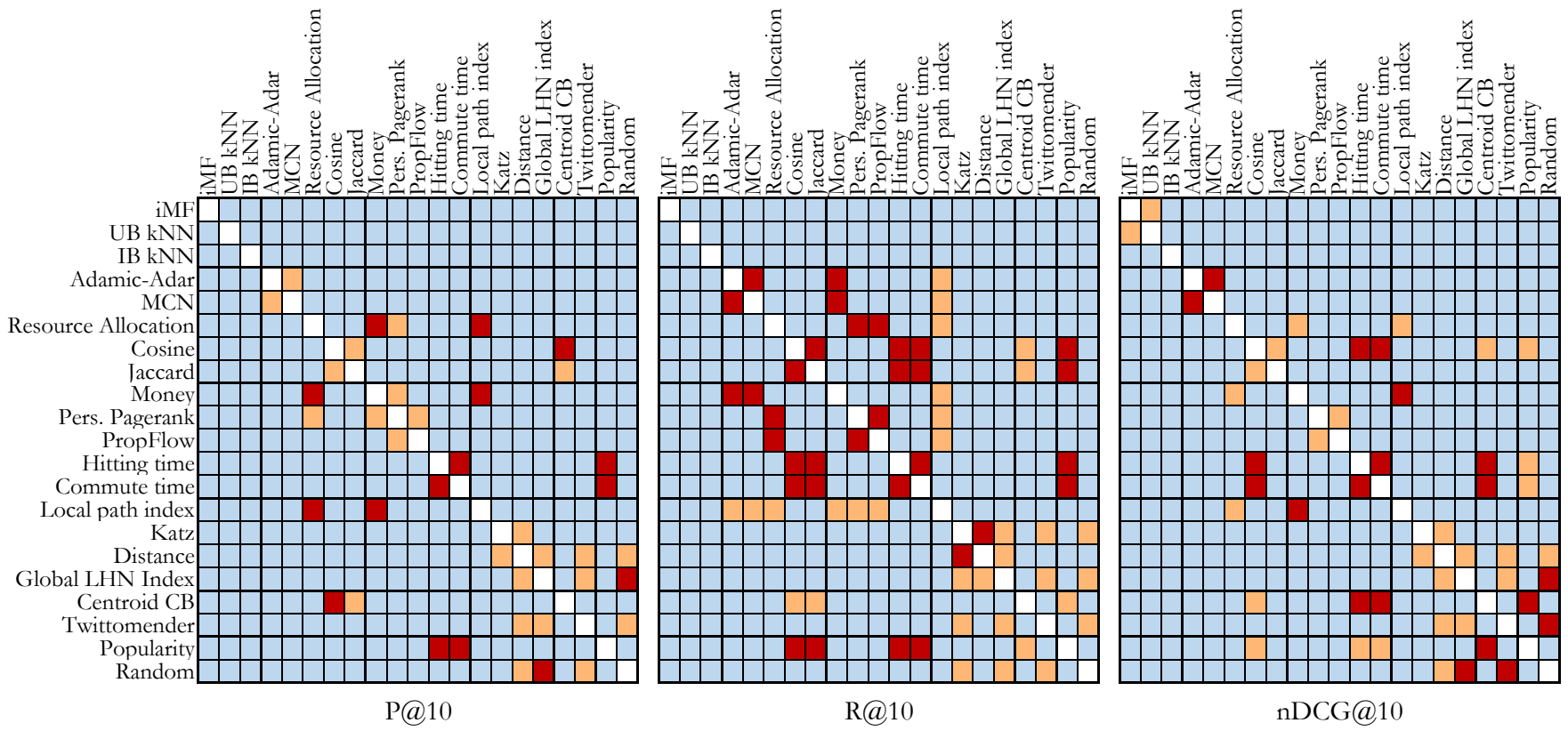


Figure D.2: Statistical significance of comparisons in Table 4.3 for the 1-month follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

Figure D.3: Statistical significance of comparisons in Table 4.3 for the 200-tweets interactions dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

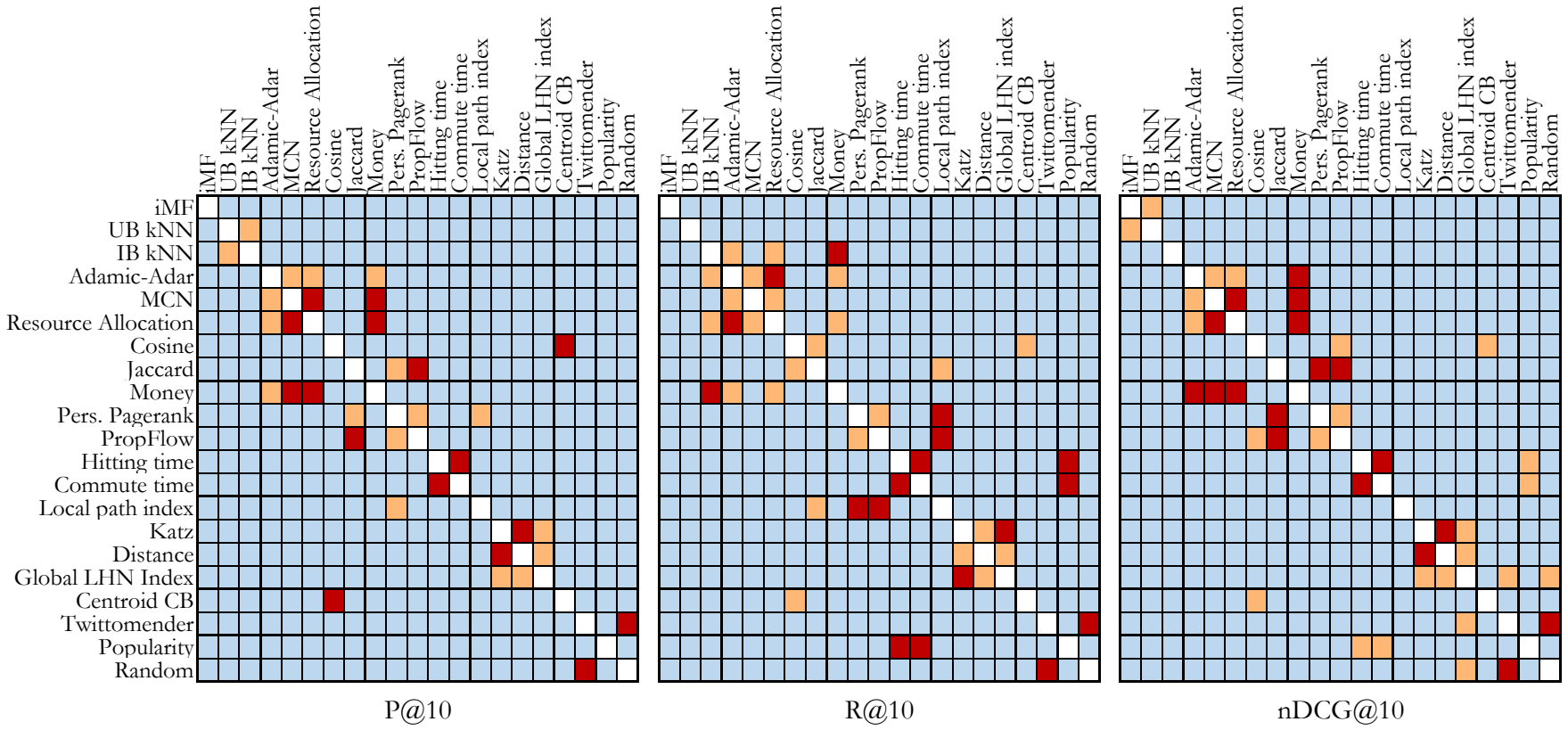


Figure D.4: Statistical significance of comparisons in Table 4.3 for the 200-tweets follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

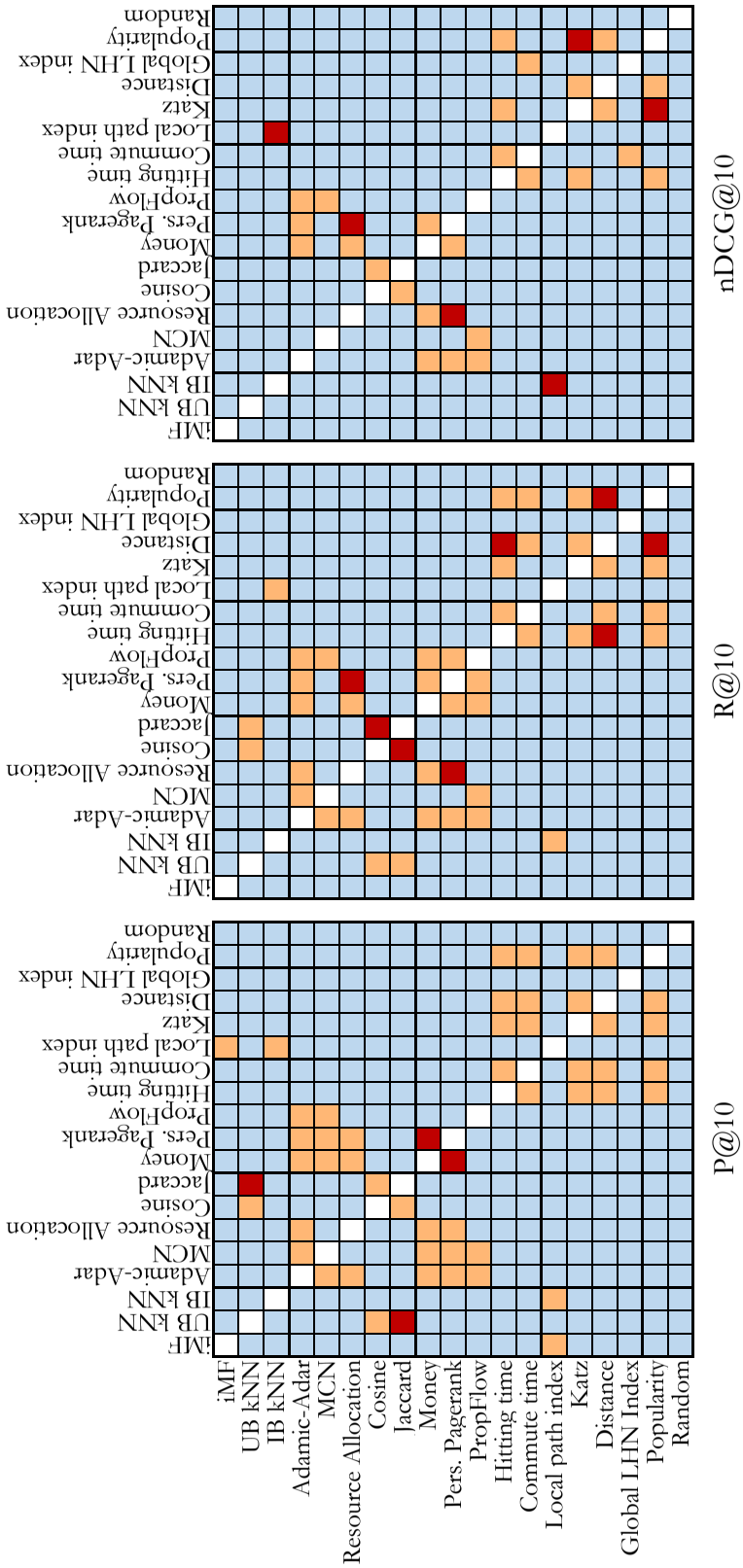


Figure D.5: Statistical significance of comparisons in Table 4.3 for the 200-tweets follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

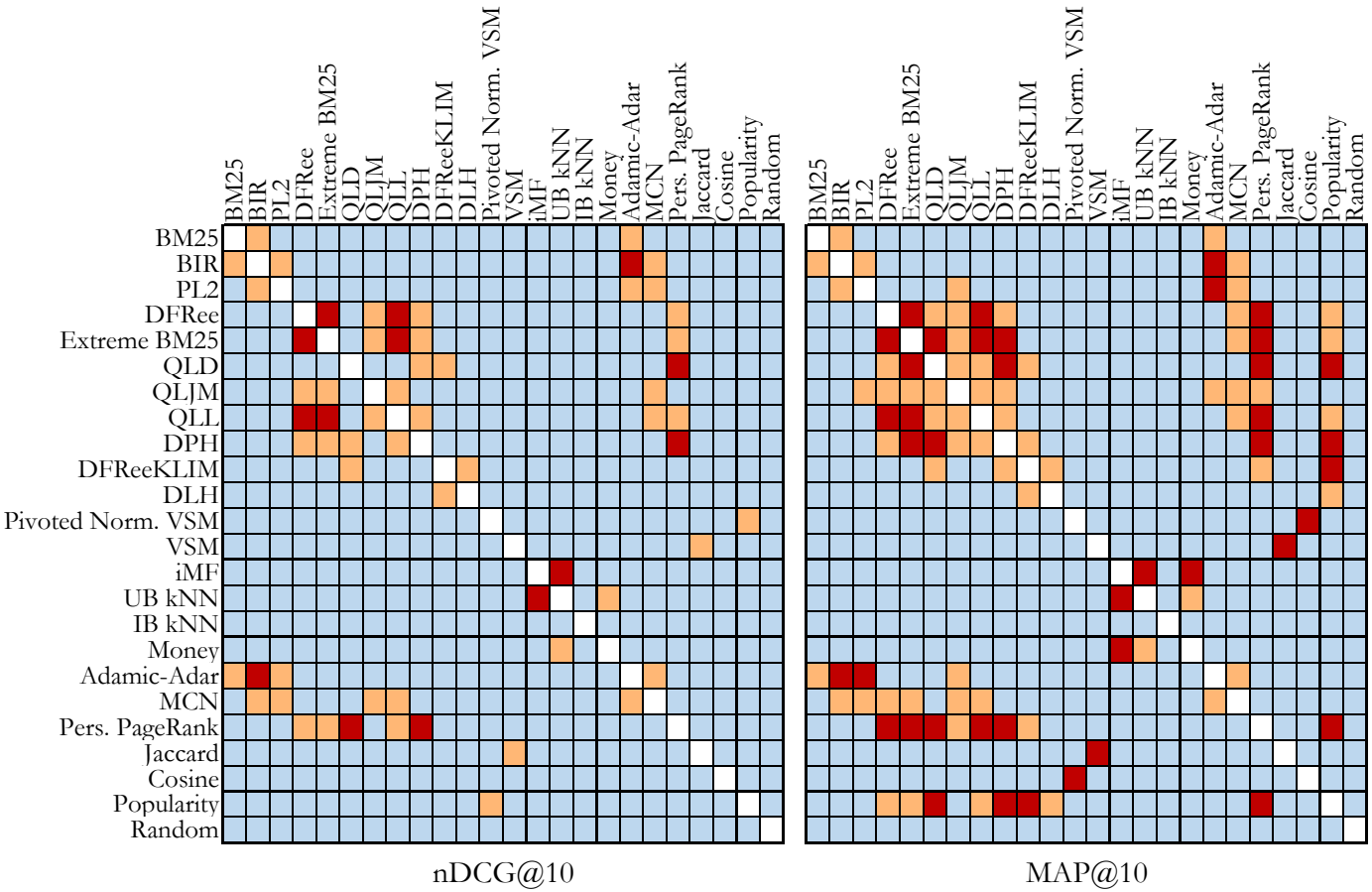


Figure D.6: Statistical significance of comparisons in Table 5.6 for the 1-month interactions dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

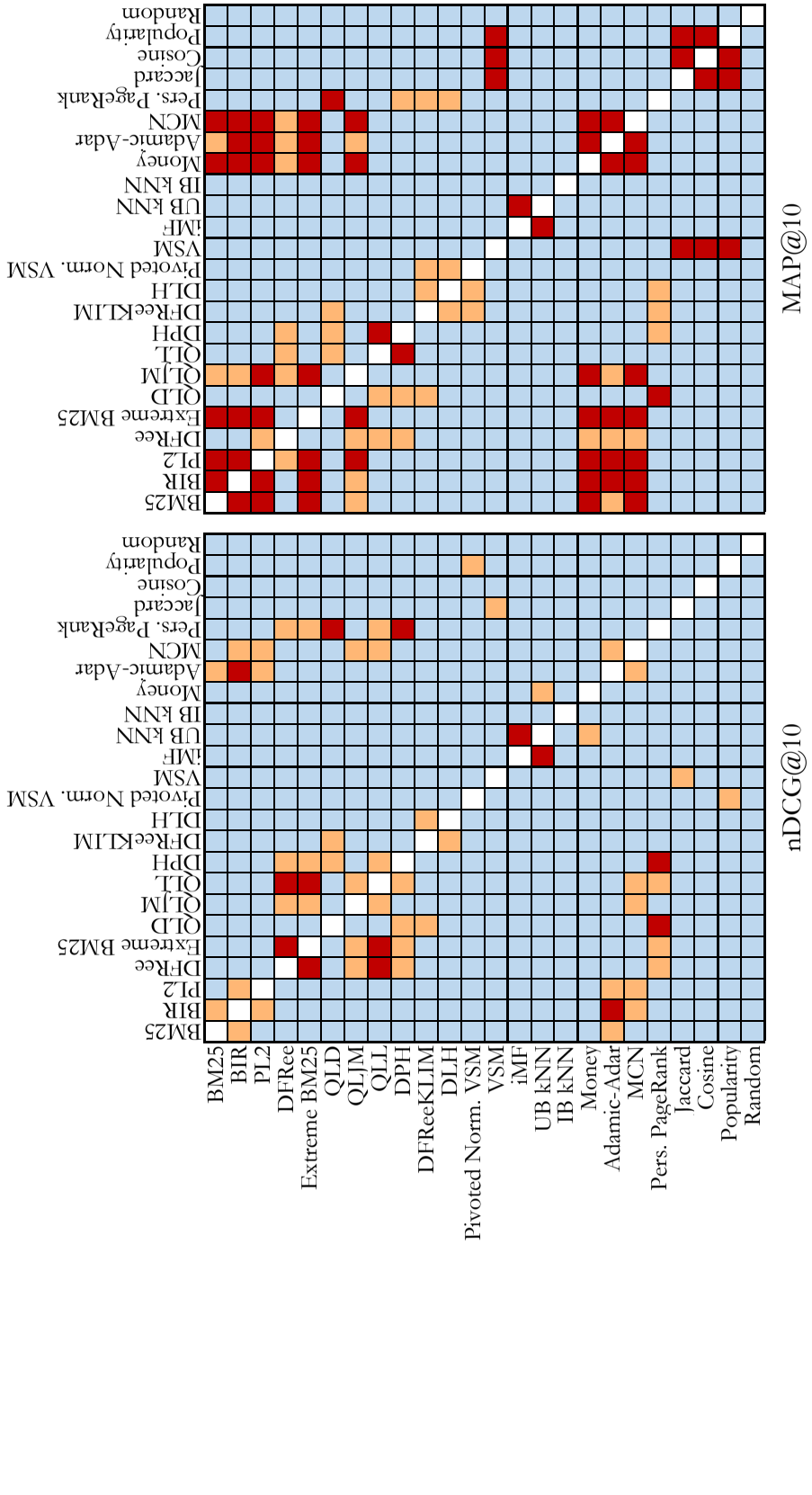


Figure D.7: Statistical significance of comparisons in Table 5.6 for the 1-month follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

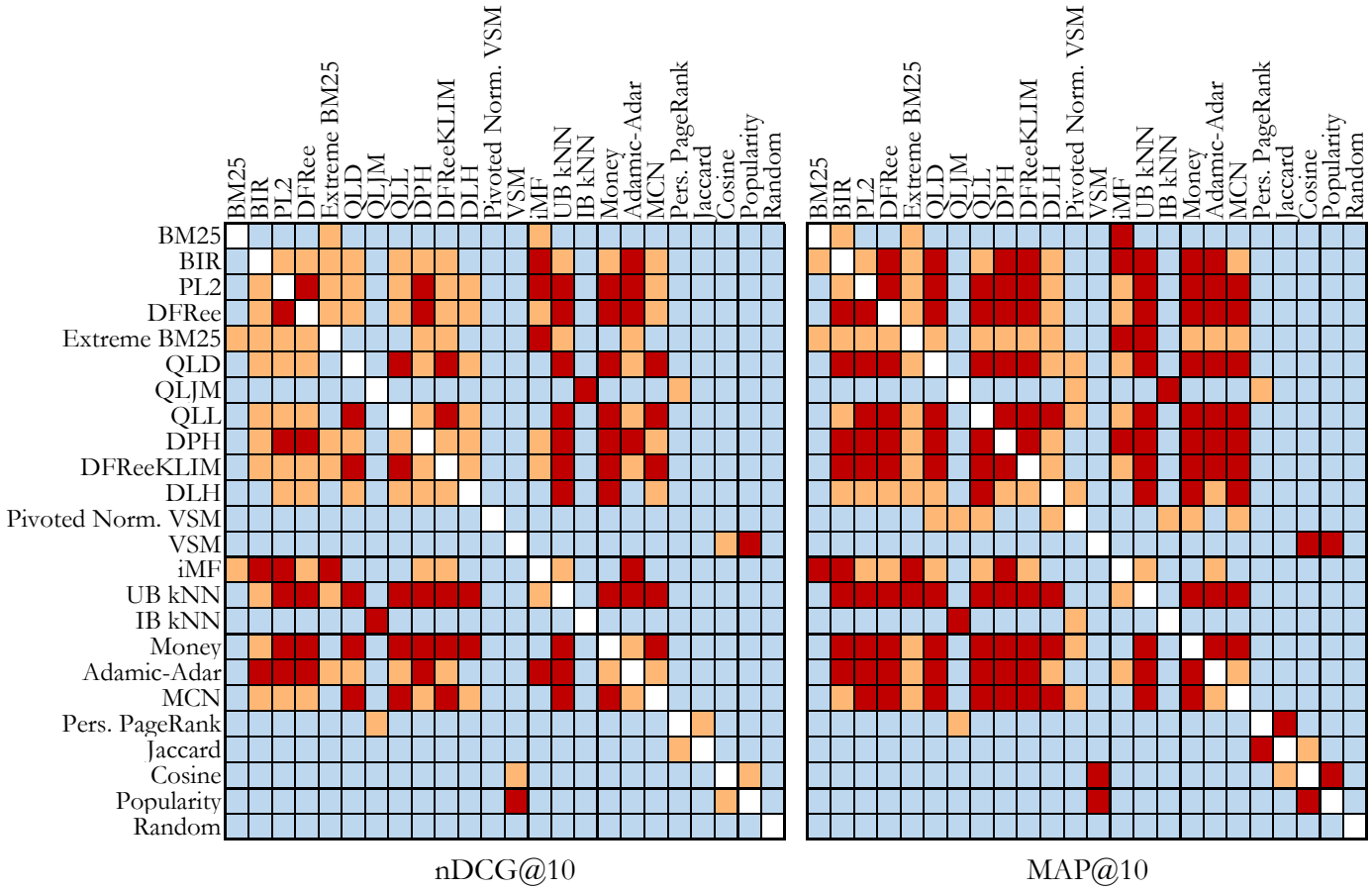


Figure D.8: Statistical significance of comparisons in Table 5.6 for the 200-tweets interactions dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

Figure D.9: Statistical significance of comparisons in Table 5.6 for the 200-tweets follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

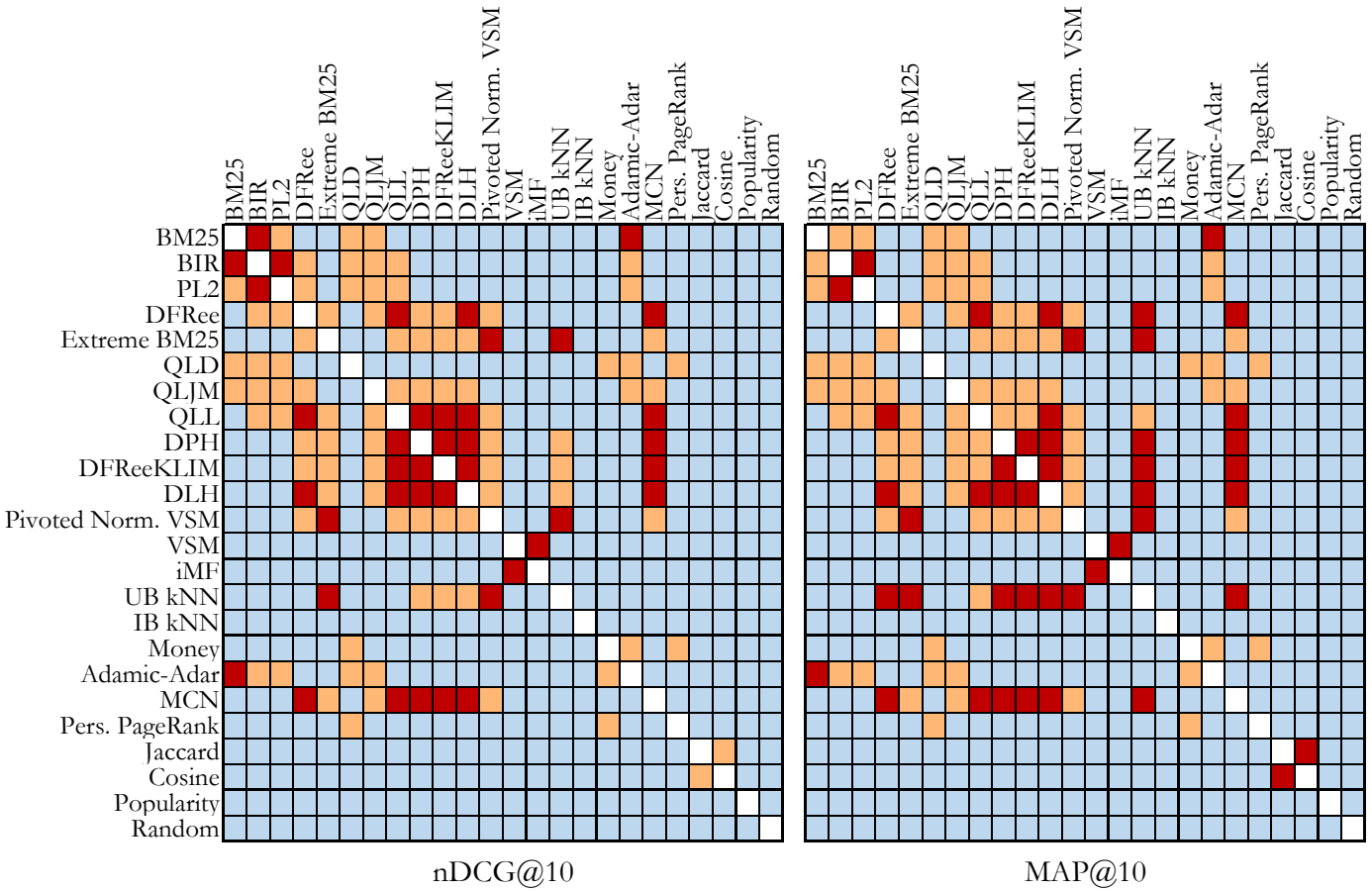


Figure D.10: Statistical significance of comparisons in Table 5.6 for the 200-tweets follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

Figure D.11: Statistical significance of comparisons in Table 7.2 for the 1-month interactions dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

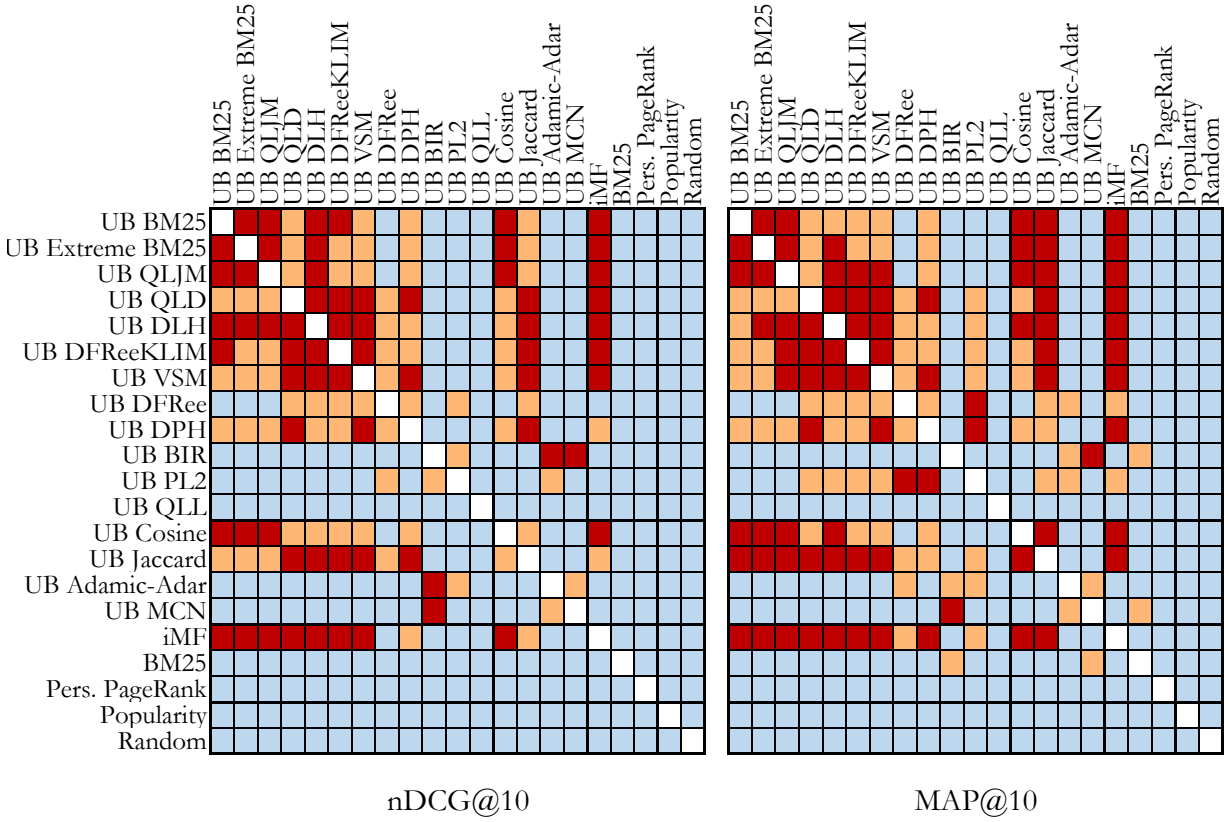


Figure D.12: Statistical significance of comparisons in Table 7.2 for the 1-month follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

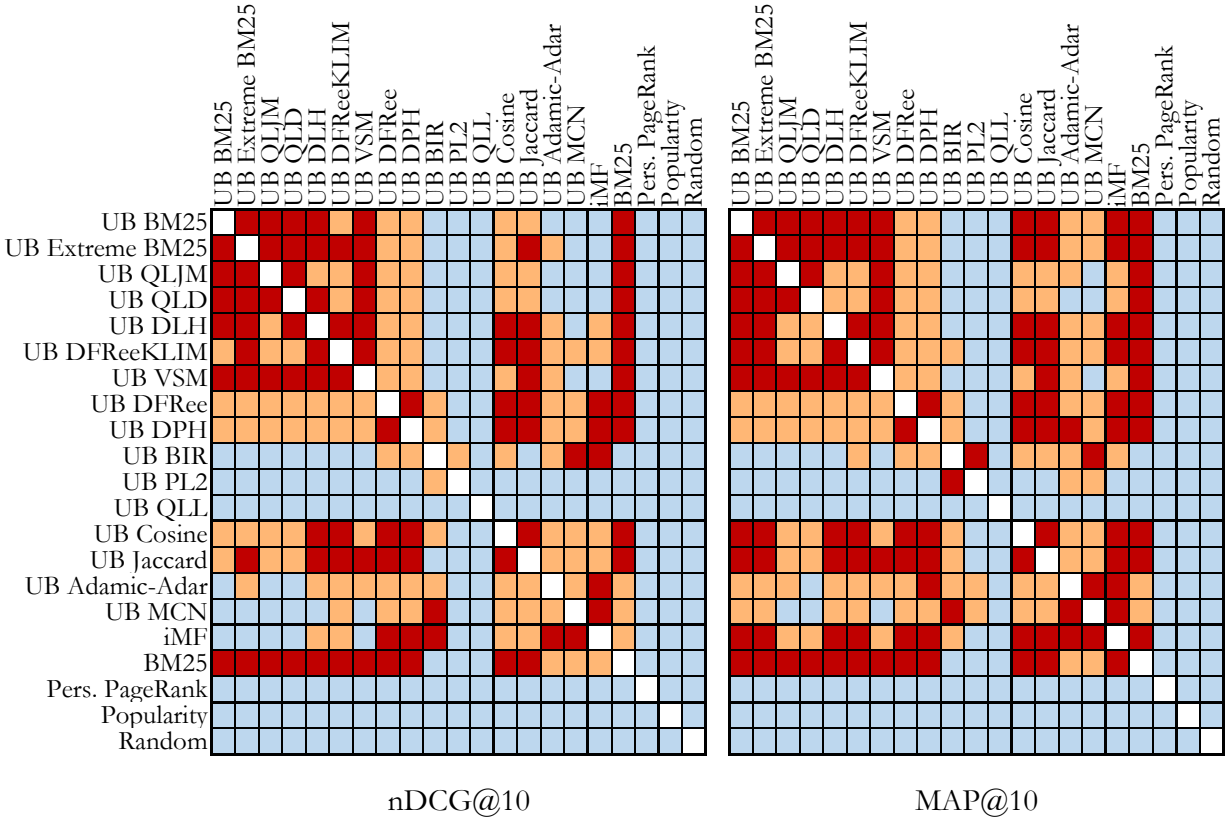


Figure D.13: Statistical significance of comparisons in Table 7.2 for the 200-tweets interactions dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

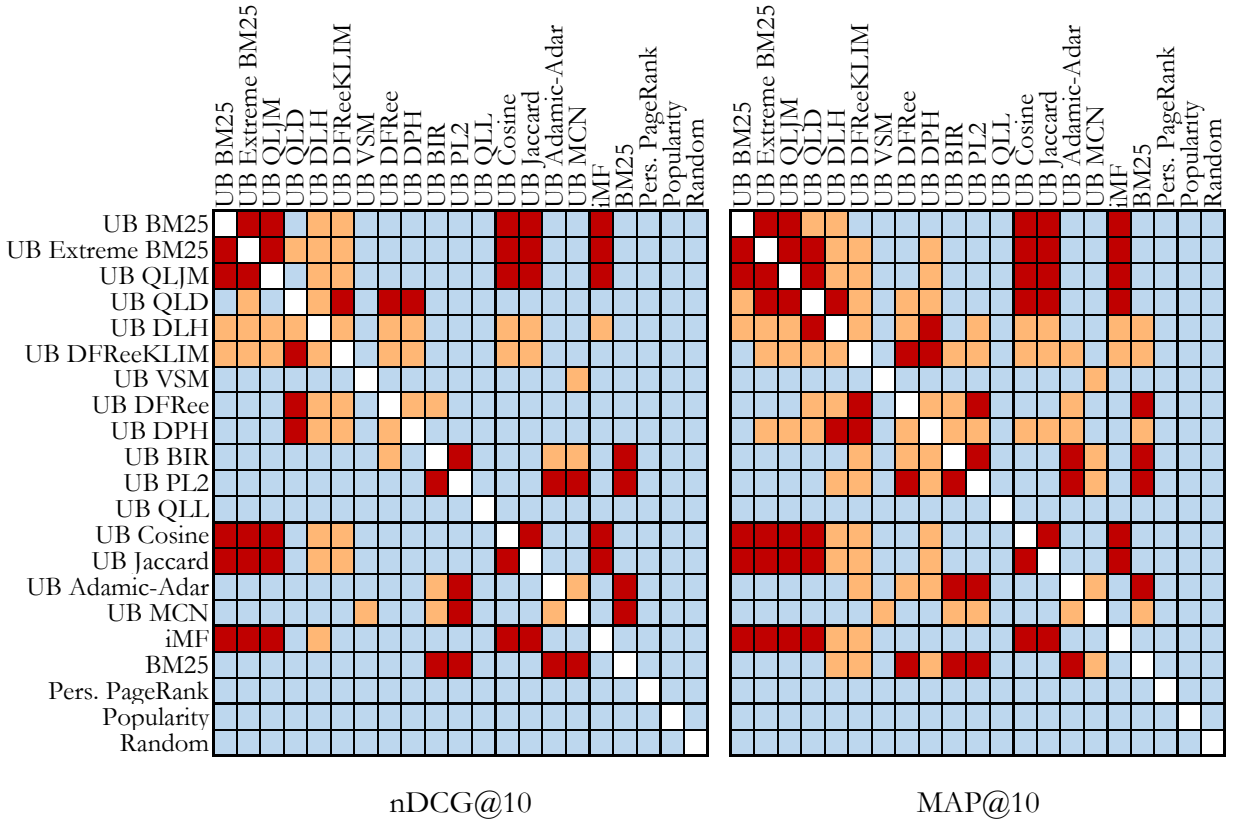


Figure D.14: Statistical significance of comparisons in Table 7.2 for the 200-tweets follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

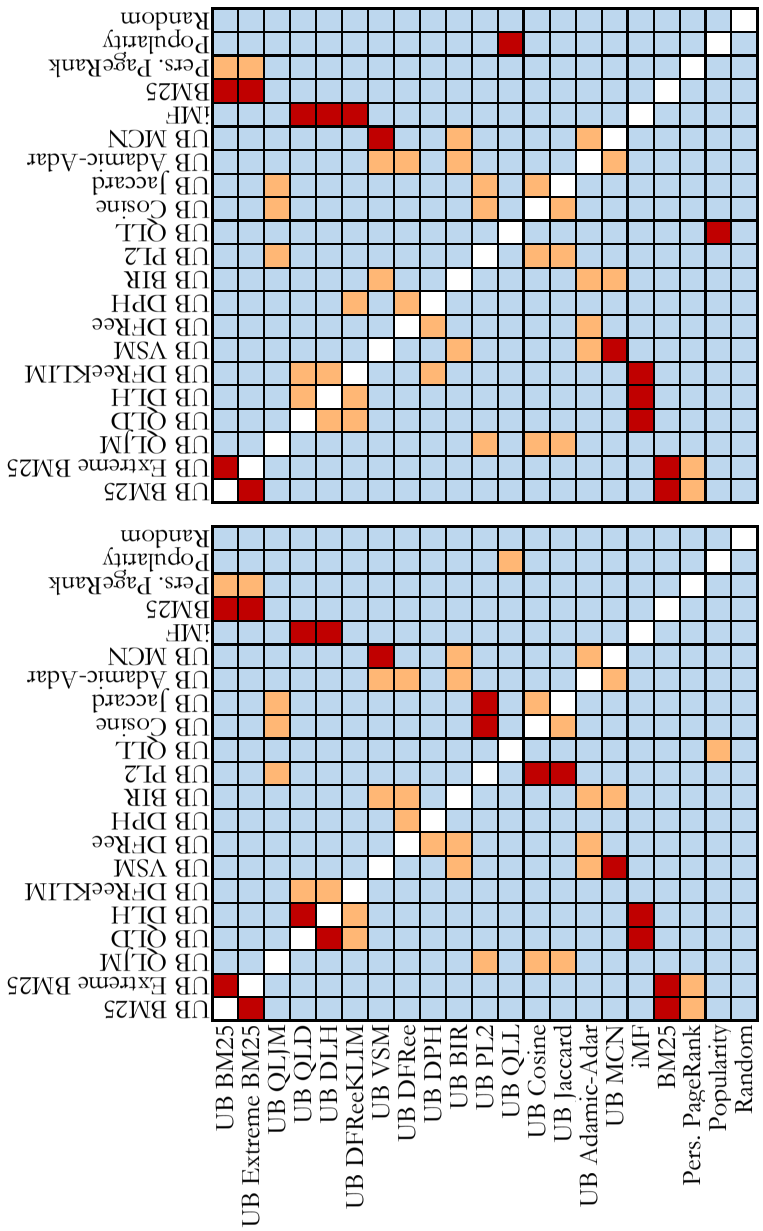


Figure D.15: Statistical significance of comparisons in Table 7.2 for the 200-tweets follows dataset. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

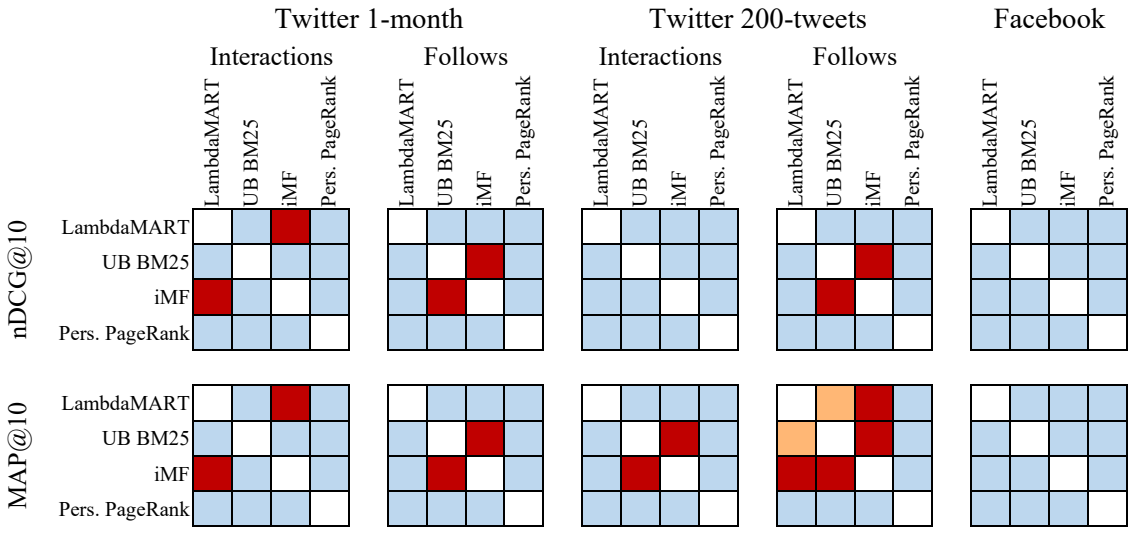


Figure D.16: Statistical significance of comparisons in Figure 7.5. Blue cells indicate significant differences by both the paired t-test and Tukey HSD at $p < 0.05$, light orange cells indicate that the difference is significant by the t-test but not by Tukey HSD, and red cells indicate non-significance by both tests.

and 7.2 is significant for t-test but not for Tukey HSD, and the same happens in the comparison between the user-based recommender with BM25 and personalized PageRank in terms of nDCG@10 and MAP@10 in Table 7.2 for the Facebook network.

The Tukey HSD is sensitive to the number of algorithms reported in a comparison: the larger the number of rows in the table reporting the experiments, the less likely the significance of the differences will be. This effect can be easily observed in Figures D.15 and D.16, where the difference between user-based BM25 and personalized PageRank is significant for the comparison in Figure 7.5 (with only four systems) whereas it is not for the comparison in Table 7.2 (with 21 systems).

E

Specialized global reranking

In Chapter 9, we introduce a global algorithm that we use for enhancing structural properties of the network by reranking the top results of a contact recommendation algorithm. However, the cost of separately enhancing such measures is sometimes high (sometimes, up to $O(|\mathcal{U}^2|)$). Therefore, in this appendix, we propose several optimizations that might reduce the computation cost for the new structural diversity measures when we remove or add a new link in the network. We explore such optimizations for the three measures we explore in this chapter: the clustering coefficient complement, the modularity complement and the community edge Gini complement.

E.1 Clustering coefficient complement

We first study how to optimize the clustering coefficient complement of a social network. As we introduce in Equation (8.11), we can formulate an equation for such metric as:

$$\mathrm{CCC}(\mathcal{G}') = 1 - \frac{\mathrm{triangles}(\mathcal{G}')}{\mathrm{triads}(\mathcal{G}')} \tag{E.1}$$

where $\mathrm{triangles}(\mathcal{G}') = |\{(u,v,w)|(u,v),(v,w),(u,w) \in E'\}|$ is the number of triangles in the extended network and $\mathrm{triads}(\mathcal{G}') = |\{(u,v,w)|(u,v),(v,w) \in E' \wedge u \neq w\}|$ is the number of triads of nodes in the network (i.e. two hop paths between different nodes). A first approach to find the value of this metric when an edge is added (or removed) consists in computing it from scratch. We show in Algorithm 5 an approach to perform such computation.

As we can observe, in order to find the number of triads and triangles in the network, we have to run over each user in the network and over pairs of his/her neighbors, leading to a $O\left(|\mathcal{U}|\ \mathrm{avg}^2_{u\in\mathcal{U}}\ |\Gamma(u)|\right)$ computational cost. Since this algorithm has to be run for each possible edge swap in our reranking approach, the cost of this computation is notably expensive. We nonetheless find that it is easier to improve such cost if we just assess the changes that the newly added (removed) edges cause in their nearby environment. We focus next on the addition of a new link (that did not previously exist) to the network. It should be noted that the reasoning for removing a single edge is equivalent to the one we introduce here (and, consequently, we omit it from the text).

To determine the effects of adding a simple link in a network over the clustering coefficient is that the addition of that edge (u,v) just adds new triplets and triangles to the network in their local environment: all such triplets and triads have to include the (u,v) link in their structure. Also, all the previous triads and triangles in the network remain unchanged. Therefore, it is enough to find how many new triangles and triads appear in the network as a consequence of the edge, and add them to the total. Then, the CCC metric is just computed as:

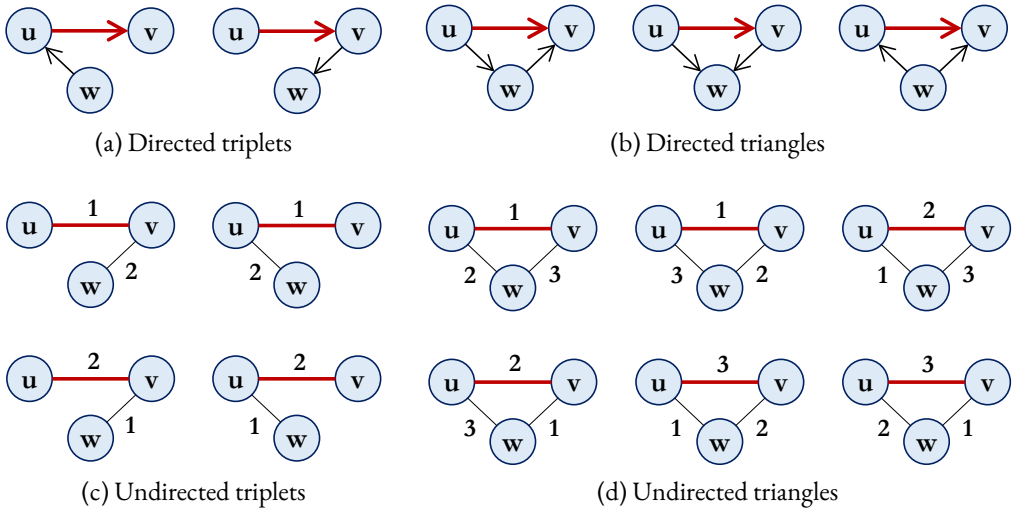
$$\mathrm{CCC}\left(\mathcal{G} \cup \{(u,v)\}\right) = 1 - \frac{\mathrm{triangles}(\mathcal{G}) + \Delta\mathrm{triangles}\left(\mathcal{G},(u,v)\right)}{\mathrm{triads}(\mathcal{G}) + \Delta\mathrm{triads}\left(\mathcal{G},(u,v)\right)} \tag{E.2}$$

where $\Delta\mathrm{triangles}(\mathcal{G},(u,v))$ is the new triangles that appear when the (u,v) edge is added to the network $\mathcal{G}$, and $\Delta\mathrm{triads}(\mathcal{G},(u,v))$ denotes the same for the triplets. The new possible triads and triplets in the networks are illustrated in Figure E.1.

We study first the number of new triplets that appear in the network. In directed networks, only two possible groups of new triads might be created that include the (u,v): triplets containing incoming neighbors of the first endpoint (w,u,v) or triplets containing outgoing neighbors of the second endpoint (u,v,w), so we just have to add as many new triads as the in-degree of u and the out-degree of v. In undirected networks, the inversed triplets also appear ((v,u,w) and (w,v,u)), so it is enough to sum up the double of the degrees of u and v.

Algorithm 5: Clustering coefficient complement

Data:
$\mathcal{G} = \langle \mathcal{U}, E \rangle$ network
Result:
$\mathrm{CCC}(\mathcal{G})$ the clustering coefficient complement
begin
  trd $\leftarrow 0$ `// Number of triads`
  trg $\leftarrow 0$ `// Number of triangles (closed triads)`
  for $u \in \mathcal{U}$ **do**
    for $v \in \Gamma_{in}(u)$ **do**
      for $w \in \Gamma_{out}(u)$ **do**
        if $v \neq w$ **then**
          `// We detect a triad`
          trd $\leftarrow$ trd $+ 1$
          if $(v, w) \in E$ **then**
            `// It is a triangle`
            trg $\leftarrow$ trg $+ 1$
  return $1 - trg/trd$



(a) Directed triplets (b) Directed triangles

(c) Undirected triplets (d) Undirected triangles

Figure E.1: Newly created triplets when adding an edge. The bold and red edge is the new one (from u to v). In the undirected case (lower row), numbers in edges indicate possible edge orderings.

Then, we analyze the triangles case. In undirected networks, for each common neighbor between u and v, we close six different triplets: the four newly created ones, and the (u, w, v) and (v, w, u) two hop paths. In directed networks, we distinguish three cases where we close a triplet:

- $w \in \Gamma_{in}(u) \cap \Gamma_{in}(v)$: The newly formed (w, u, v) triplet is closed thanks to the (w, v) edge.
- $w \in \Gamma_{out}(u) \cap \Gamma_{out}(v)$: The newly formed (u, v, w) triplet is closed thanks to the (u, w) edge.
- $w \in \Gamma_{out}(u) \cap \Gamma_{in}(v)$: In this case, the new edge, (u, v) adds a transitive edge to the 2-step path (u, w, v)

Then, we can summarize this by expressing a mathematical formulation for $\Delta\text{triangles}(\mathcal{G}, (u, v))$ and $\Delta\text{triads}(\mathcal{G}, (u, v))$. In the case of directed networks,

$$\Delta\text{triads}(\mathcal{G}, (u, v)) = |\Gamma_{in}(u) \setminus \{v\}| + |\Gamma_{out}(v) \setminus \{u\}| \tag{E.3}$$

$$\Delta\text{triangles}(\mathcal{G}, (u, v)) = |\Gamma_{in}(u) \cap \Gamma_{in}(v)| + |\Gamma_{out}(u) \cap \Gamma_{out}(v)| + |\Gamma_{out}(u) \cap \Gamma_{in}(v)| \tag{E.4}$$

whereas, for undirected ones:

$$\Delta\text{triads}(\mathcal{G},(u,v)) = 2\cdot(|\Gamma(u)|+|\Gamma(v)|) \tag{E.5}$$
$$\Delta\text{triangles}(\mathcal{G},(u,v)) = 6\cdot|\Gamma(u)\cap\Gamma(v)| \tag{E.6}$$

In all the equations, $\Gamma(u),\Gamma(v)$ refer to the neighborhood of the users before the (u,v) edge is added. If an edge is removed, we just substract this quantities to the number of triangles/triads before the removal. When applying this, if we store the neighborhoods for both users in hash tables, we only need to run over one of them to compute the intersections, reducing the cost to $O\left(\text{avg}_{u\in\mathcal{U}}|\Gamma(u)|\right)$ for each swapped pair of edges.

E.2 Modularity complement

The second metric that we study is the modularity complement. According to equation (8.12) (which we repeat here), given a community partition, the modularity of a network is defined as:

$$\text{mod}(\mathcal{G}'|\mathcal{C}) = \frac{\sum_{u,v\in\mathcal{U}}\left(A_{uv}-\frac{|\Gamma_{out}(u)||\Gamma_{in}(v)|}{|E'|}\right)\mathbb{1}_{c(u)=c(v)}}{|E'|-\sum_{u,v\in\mathcal{U}}\left(\frac{|\Gamma_{out}(u)||\Gamma_{in}(v)|}{|E'|}\right)\mathbb{1}_{c(u)=c(v)}} \tag{E.7}$$

The fastest way to compute such equation is by running over the set of pairs of users in the network, i.e. it has a $O(|\mathcal{U}|^2)$ computational cost. In the case of the modularity, adding a single edge modifies $2|\mathcal{U}|$ elements in the two sums appearing in the equation. Therefore, the minimum cost would be $O(|\mathcal{U}|)$ for each pair of edges that we swap.

The computational cost of this is still large, but it can be improved if, instead of finding the exact value of the modularity complement metric, we use a suitable heuristic. As we explain in Section 8.2.2, the modularity complement provides a measure of how many links traverse pairs of communities. Instead of optimizing such metric, we can thus maximize the number of weak links between communities. On the global greedy reranking strategy, we can then write the diversity metric as:

$$\mu\left(\mathcal{G},\mathcal{S}_{\langle u:i/j\rangle}@k\right) = \text{WT}\left(\mathcal{G},\mathcal{S}@k\right) + \mathbb{1}_{c(u)\neq c(\mathcal{S}_u[j])} - \mathbb{1}_{c(u)\neq c(\mathcal{S}_u[i])} \tag{E.8}$$

where $\text{WT}(\mathcal{G},\mathcal{S}@k)$ is the number of links that travel outside the community in the extended network where we add the links in $\mathcal{S}@k$ at the training graph. If we store the number of weak links in previous iterations, this measure can be computed in $O(1)$. We can even remove the $\text{WT}(\mathcal{G},\mathcal{S}@k)$ term, since it is going to be the same for all the edges.

If we remove this, we can observe that the diversity we are measuring is whether each of the candidate links is weak or not. As we do not recompute the communities after the recommendation is done, this property does not depend on others: just on the recommended link. We can thus apply any local reranking algorithm (Castells et al., 2015), like MMR (Carbonell and Goldstein, 1998) to optimize the number of weak links (and, at the same time, the modularity complement) of the network.

E.3 Community edge Gini complement

The last metric we consider in this chapter is the community edge Gini complement of the network. As explained in Section 8.2.2, the metric uses the Gini index (Dorfman, 1979) to provide a measure of how balanced the distribution of edges between pairs of communities is. The cost of enhancing such metric does thus depend on the cost of computing the Gini index of the edge distribution. Given a value distribution X, we recall the equation for the Gini index:

$$\text{Gini}(X) = \frac{1}{|X|-1}\sum_{i=1}^{|X|}(2i-|X|-1)\frac{x_i}{\sum_{j=1}^{|X|}x_j} \tag{E.9}$$

X	0	0	1	1	1	4	4	7	9	9	9	9
	1	2	3	4	5	6	7	8	9	10	11	12
	m_0	M_0	m_1		M_1	m_4	M_4	m_7 M_7	m_9			M_9
$\mathbb{V}(X)$	0		1			4		7	9			

Figure E.2: Example of the new values for the alternative formulation of the Gini index.

where x_i is the value of the i-th element of the distribution in ascending order.

Each time we compute everything from zero, we need to sort the values of the distribution (which, using some efficient sorting algorithms, as HeapSort, has a $O(|X| \log |X|)$ computational cost) and then, we have to run over each value of the distribution (adding an additional $O(|X|)$ cost). Depending on the size of the distribution, this cost could be rather expensive (for example, if we wanted to balance the in-degree distribution of the network, $|X|$ would just be the number of users in the network). To overcome that difficulty, it is possible to reduce this cost if we use incremental updates.

The incremental update method we propose here is based on an alternative formulation of the Gini coefficient for discrete distributions. Over the X distribution, we can define the following equivalence relation:

$$i \equiv j \Leftrightarrow x_i = x_j \tag{E.10}$$

for $i, j \in \{1, 2, ..., |X|\}$. We denote the set of equivalence classes for such relation as $\mathbb{V}(X)$. Each equivalence class is defined by a unique value $v \in \mathbb{R}$. Therefore, we define each equivalence class as $v = \{i \in \{1, 2, ..., |X|\} | x_i = v\}$[1]. For each of those equivalence classes $v \in \mathbb{V}(X)$ we define two values representing the minimum index of the value v in the sorted distribution, m_v, and the maximum index, M_v. Formally, these indexes are defined as:

$$m_v = \min\{i | i \in v\} = \min\{i | x_i = v\} \tag{E.11}$$

$$M_v = \max\{i | i \in v\} = \max\{i | x_i = v\} \tag{E.12}$$

Figure E.2 illustrates an example of the $\mathbb{V}(X)$, m_v and M_v values for an specific distribution. In such figure, the top row of the matrix represents the distribution and the lower the equivalence class set. Each color represents the set of indexes included on each equivalence class.

After finding all the values, we can rewrite the Gini equation as:

$$\text{Gini}(X) = \frac{1}{|X| - 1} \sum_{v \in \mathbb{V}(X)} \frac{(M_v - m_v + 1) \cdot (M_v + m_v - |X| - 1) \cdot v}{\sum_{w \in \mathbb{V}(X)} (M_w - m_w + 1) \cdot (M_w + m_w) \cdot w} \tag{E.13}$$

The previous formulation has an interesting advantage over the original one for incremental computation: as we only work with equivalence classes, we do not care about the order of the elements represented by each class: for example, if we added value 1 to each of the elements in the 4 class, we can suppose that it is in the last position M_4 to simplify the calculations. If we reduced it by one unit, we might suppose that it is in position m_4 instead: the variations of the Gini equation would be the same. Now that we have the alternative Gini index formula, we can decompose in three different elements: the normalization by the number of elements in the distribution (minus one) and the following two sums:

$$\text{S}(X) = \sum_{j=1}^{|X|} x_i = \sum_{w \in \mathbb{V}(X)} (M_w - m_w + 1)(M_w + m_w) \cdot w \tag{E.14}$$

$$\text{G}(X) = \sum_{i=1}^{N} (2i - |X| - 1) \cdot x_i = \sum_{v \in \mathbb{V}(X)} (M_v - m_v + 1)(M_v + m_v - |X| - 1) \cdot v \tag{E.15}$$

[1]Here, v defines both the equivalence class and the value of all its elements in the distribution. From now on, we shall use each meaning as it corresponds

Considering this, we can rewrite the Gini equation as

$$\text{Gini}(X) = \frac{\text{G}(X)}{(|X|-1)\cdot \text{S}(X)} \tag{E.16}$$

Then, if we increase the value of x_i in a unit (for example, if we add an edge between two different communities in CEGC), the resulting value for the Gini coefficient is defined as:

$$\text{Gini}(X|\hat{x_i} = v+1) = \frac{\text{G}(X) + 2M_v - |X| - 1}{(|X|-1)\cdot(\text{S}(X)+1)} \tag{E.17}$$

where v is the value (and equivalence class) of the i-th element of the distribution.

Proof. We take, without loss of generality, the i-th element in the distribution, x_i, with equivalence class v. As we add 1 to this value, we have to move it from v to $v+1$. It is easy to see that:

$$S(X|\hat{x}_i = v+1) = S(X) + 1$$

and the number of elements in the distribution, $|X|$, remains unchanged. Therefore, we have just to check how $G(X|\hat{x}_i = v+1)$ changes with respect to $G(X)$. In such sum, it is only necessary to modify the equivalence classes for v and $v+1$. As we remove one element from v and add it to the adjacent one $(v+1)$, both classes still occupy the same segment of the sorted distribution. Therefore, for $w \neq v, v+1$, the indexes m_w, M_w (and the addends for those w in $G(X|\hat{x}_i = v+1)$ remain unchanged. Then, the new sum can be written as:

$$G(X|\hat{x}_i = v+1) = G(X) + \Delta_v G(X|\hat{x}_i = v+1) + \Delta_{v+1} G(X|\hat{x}_i = v+1)$$

where

$$\Delta_w G(X|\hat{x}_i = v+1) = G_w(X|\hat{x}_i = v+1) - G_w(X) \tag{E.18}$$

is the variation of the element in the $G(X|\hat{x}_i = v+1)$ associated to the w equivalence class, and $G_w(X), G_w(X|\hat{x}_i = v+1)$ is the addend associated to w in each of the sums.

We first find the Δ value for the v equivalence class (the one that loses an element). For this case, we define $G_v(X) = (M_v - m_v + 1)(M_v + m_v - |X| - 1)v$ and we differentiate two cases, depending on the relation between m_v and M_v:

- **Case 1 ($m_v = M_v$):** In this case, we are removing the only element from the equivalence class. Therefore, the equivalence class v disappears, and $G_w(X|\hat{x}_i = v+1) = 0$. The variation is then computed as:

  $$\Delta_v G(X|\hat{x}_i = v+1) = -(M_v - m_v + 1)\cdot(M_v + m_v - |X| - 1)\cdot v = (|X| + 1 - 2M_v)\cdot v$$

  considering that $M_v = m_v$.

- **Case 2 ($m_v < M_v$):** In this case the equivalence class does not disappear: we just reduce the M_v in one unit. Therefore, $G_w(X|\hat{x}_i = v+1) = (M_v - m_v)\cdot(M_v + m_v - |X| - 2)\cdot v$ and we just have to compute the difference between the new and the old values for the equivalence class v:

  $$\begin{aligned}\Delta_v G(X|\hat{x}_i = v+1) &= (M_v - m_v)\cdot(M_v + m_v - |X| - 2)\cdot v \\ &\quad - (M_v - m_v + 1)\cdot(M_v + m_v - |X| - 1)\cdot v \\ &= (|X| + 1 - 2M_v)\cdot v\end{aligned}$$

Then, $\Delta_v G(X, \hat{x_i} = v+1) = (|X| + 1 - 2M_v)\cdot v$. By symmetry, we observe that $\Delta_{v+1} G(X, \hat{x_i} = v+1) = (2M_v - |X| - 1)\cdot(v+1)$. As both equivalence classes are adjacent, the new element in the $v+1$ class occupies the M_v position which previously occupied x_i in the v class. Therefore, we define $\hat{m}_{v+1} = M_v$. If the $v+1$ category did not exist in $\mathbb{V}(X)$ before, $\hat{M}_{v+1}$ also takes the M_v value. Then, to complete the proof, we have just to sum:

$$\begin{aligned}G(X|\hat{x}_i = v+1) &= G(X) + \Delta_v G(X|\hat{x}_i = v+1) + \Delta_{v+1} G(X|\hat{x}_i = v+1) \\ &= G(X) + (|X| + 1 - 2M_v)\cdot v + (2M_v - |X| - 1)\cdot(v+1) \\ &= G(X) + (2M_v - |X| - 1)\end{aligned}$$

Table E.1: Computational cost of the enhancement of a metric when a single edge is added. In the case of MC, the incremental update is an heuristic approach maximizing the number of weak ties in the network.

Metric	From scratch	Incremental update
CCC	$O\left(\lvert\mathcal{U}\rvert \operatorname{avg}^2_{u\in\mathcal{U}} \lvert\Gamma(u)\rvert\right)$	$O\left(\operatorname{avg}_{u\in\mathcal{U}} \lvert\Gamma(u)\rvert\right)$
MC	$O\left(\lvert\mathcal{U}\rvert^2\right)$	$O(1)$
CEGC	$O\left((\lvert\mathcal{C}\rvert^2 - \lvert\mathcal{C}\rvert + 1)\log(\lvert\mathcal{C}\rvert^2 - \lvert\mathcal{C}\rvert + 1)\right)$	$O(1)$

□

Following a similar reasoning, we can also prove that, if we do the opposite, i.e. we decrease the value of a distribution element from v to $v-1$, the new Gini index can be computed as:

$$\text{Gini}(X|\hat{x}_i = v-1) = \frac{\text{G}(X) + |X| + 1 - 2m_v}{(|X|-1)\cdot(\text{S}(X)-1)} \tag{E.19}$$

and, if $M_v > m_v$, then, m_v increases one position (otherwise class v disappears), $\hat{M}_{v-1} = m_v$ and, if $v-1 \notin \mathbb{V}(X)$, $\hat{m}_{v-1} = m_v$.

To sum up, we include the theoretical computational costs of the three metrics from scratch and using our optimized, incremental update approaches in Table E.1.

Bibliography

Lada A. Adamic and Eytan Adar. Friends and neighbors on the Web. *Social Networks*, 25(3):211–230, July 2003. doi:10.1016/S0378-8733(03)00009-1.

Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, April 2005. doi:10.1109/TKDE.2005.99.

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. Diversifying Search Results. In *Proceedings of the 2nd ACM International Conference on Web Search and Data Mining (WSDM 2009)*, page 5–14, Barcelona, Spain, February 2009. ACM. doi:10.1145/1498759.1498766.

Luca Maria Aiello and Nicola Barbieri. Evolution of Ego-networks in Social Media with Link Recommendations. In *Proceedings of the 10th ACM international conference on Web Search and Data Mining (WSDM 2017)*, pages 111–120, Cambridge, United Kingdom, February 2017. ACM. doi:10.1145/3018661.3018733.

Mohammad Yahya H. Al-Shamri. User profiling approaches for demographic recommender systems. *Knowledge-Based Systems*, 100:175–187, May 2016. doi:10.1016/j.knosys.2016.03.006.

Giambattista Amati. *Probability Information Models for Retrieval based on Divergence from Randomness*. PhD thesis, University of Glasgow, 2003.

Giambattista Amati. Frequentist and Bayesian Approach to Information Retrieval. In *Proceedings of the 28th European Conference on Information Retrieval (ECIR 2006)*, number 3936 in Lecture Notes in Computer Science, pages 13–24, London, United Kingdom, April 2006. Springer. doi:10.1007/11735106_3.

Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness. *ACM Transactions on Information Systems*, 20(4):357–389, October 2002. doi:10.1145/582415.582416.

Gianni Amati, Edgardo Ambrosi, Marco Bianchi, Carlo Gaibisso, and Giorgio Gambosi. FUB, IASI-CNR and University of Tor Vergata at TREC 2007 Blog track. In *Proceedings of The 16th Text REtrieval Conference (TREC 2007)*, Gaithersburg, Maryland, USA, November 2007. NIST.

Gianni Amati, Giuseppe Amodeo, Marco Bianchi, Giuseppe Marcone, Fondazione Ugo Bordoni, Carlo Gaibisso, Giorgio Gambosi, Alessandro Celi, Cesidio Di Nicola, and Michele Flammini. FUB, IASI-CNR, UNIVAQ at TREC 2011 Microblog Track. In *Proceedings of The 20th Text REtrieval Conference (TREC 2011)*, Gaithersburg, Maryland, USA, November 2011. NIST.

Xavier Amatriain. Mining Large Streams of User Data for Personalized Recommendations. *ACM SIGKDD Explorations Newsletter*, 14(2):37–48, April 2013. doi:10.1145/2481244.2481250.

Aris Anagnostopoulos, Ravi Kumar, and Mohammad Mahdian. Influence and Correlation in Social Networks. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008)*, page 7–15, Las Vegas, Nevada, USA, August 2008. ACM. doi:10.1145/1401890.1401897.

Sinan Aral. The Future of Weak Ties. *American Journal of Sociology*, 121(6):1931–1939, May 2016. doi:10.1086/686293.

Alexandre Arenas, Jordi Duch, Alberto Fernández, and Sergio Gómez. Size reduction of complex networks preserving modularity. *New Journal of Physics*, 9(6):176:1–176:15, June 2007. doi:10.1088/1367-2630/9/6/176.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47:235–256, May 2002. doi:10.1023/A:1013689704352.

Lars Backstrom and Jure Leskovec. Supervised random walks. In *Proceedings of the 4th ACM international conference on Web Search and Data Mining (WSDM 2011)*, pages 635–644, Hong Kong, China, February 2011. ACM. doi:10.1145/1935826.1935914.

Ricardo Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval: The Concepts and Technology behind Search*. Pearson Education Ltd., 2nd edition, 2011.

Bahman Bahmani, Abdur Chowdhury, and Ashish Goel. Fast Incremental and Personalized PageRank. *Proceedings of the VLDB Endowment*, 4(3):173–184, December 2010. doi:10.14778/1929861.1929864.

Eytan Bakshy, Itamar Rosenn, Cameron Marlow, and Lada A. Adamic. The Role of Social Networks in Information Diffusion. In *Proceedings of the 21st International Conference on World Wide Web (WWW 2012)*, page 519–528, Lyon, France, April 2012. ACM. doi:10.1145/2187836.2187907.

Eytan Bakshy, Solomon Messing, and Lada A. Adamic. Exposure to ideologically diverse news and opinion on Facebook. *Science*, 348(6239):1130–1132, June 2015. doi:10.1126/science.aaa1160.

Albert-László Barabási and Réka Albert. Emergence of Scaling in Random Networks. *Science*, 286(5439): 509–512, October 1999. doi:10.1126/science.286.5439.509.

Nicola Barbieri, Francesco Bonchi, and Giuseppe Manco. Who to Follow and Why: Link Prediction with Explanations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2014)*, page 1266–1275, New York, NY, USA, August 2014. ACM. doi:10.1145/2623330.2623733.

John C. Barefoot, Morten Grønbæk, Gorm Jensen, Peter Schnohr, and Eva Prescott. Social Network Diversity and Risks of Ischemic Heart Disease and Total Mortality: Findings from the Copenhagen City Heart Study. *American Journal of Epidemiology*, 161(10):960–967, May 2005. doi:10.1093/aje/kwi128.

Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12):29–38, December 1992. doi:10.1145/138859.138861.

Alejandro Bellogín and Alan Said. Offline and Online Evaluation of Recommendations. In Shlomo Berkovsky, Iván Cantador, and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 295–328. World Scientific Publishing, November 2018. doi:10.1142/9789813275355_0009.

Alejandro Bellogín, Jun Wang, and Pablo Castells. Bridging memory-based collaborative filtering and text retrieval. *Information Retrieval*, 16(6):697–724, December 2013. doi:10.1007/s10791-012-9214-z.

Alejandro Bellogín, Pablo Castells, and Iván Cantador. Statistical biases in Information Retrieval metrics for recommender systems. *Information Retrieval*, 20(6):606–634, December 2017. doi:10.1007/s10791-017-9312-z.

Shlomo Berkovsky, Iván Cantador, and Domonkos Tikk. *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*. World Scientific Publishing, November 2018. doi:10.1142/11131.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, August 2006.

Vincent D. Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008. doi:10.1088/1742-5468/2008/10/p10008.

danah m. boyd and Nicole B. Ellison. Social Network Sites: Definition, History, and Scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, October 2007. doi:10.1111/j.1083-6101.2007.00393.x.

Ulrik Brandes, Daniel Delling, Marco Gaertler, Robert Gorke, Martin Hoefer, Zoran Nikoloski, and Dorothea Wagner. On Modularity Clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2):172–188, February 2008. doi:10.1109/TKDE.2007.190689.

Guy Bresler, George H. Chen, and Devavrat Shah. A Latent Source Model for Online Collaborative Filtering. In *Proceedings of the 28th Conference on Neural Information Processing Systems (NeurIPS 2014)*, pages 3347–3355, Montréal, Quebec, Canada, December 2014.

Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. In *Proceedings of the 7th International Conference on World Wide Web (WWW 1998)*, page 107–117, Brisbane, Australia, April 1998. Elsevier.

Chris Burges. From RankNet to LambdaRank to LambdaMART: An Overview. Microsoft Research Technical Report MSR-TR-2010-82, Microsoft, June 2010.

Robin D. Burke. Knowledge-based Recommender Systems. *Encyclopedia of Library and Information Systems*, 69(Supplement 32), 2000.

Robin D. Burke. Hybrid Web Recommender Systems. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The Adaptive Web: Methods and Strategies of Web Personalization*, number 4321 in Lecture Notes in Computer Science, pages 377–408. Springer, Berlin, Heidelberg, Germany, 2007. doi:10.1007/978-3-540-72079-9_12.

Ronald S. Burt. *Structural Holes: The Social Structure of Competition*. Harvard University Press, November 1995.

Stefan Büttcher, Charles L. A. Clarke, and Gordon V. Cormack. *Information Retrieval: Implementing and Evaluating Search Engines*. The MIT Press, July 2010.

Rocío Cañamares. *An analysis of popularity biases in recommender system evaluation and algorithms*. PhD thesis, Universidad Autónoma de Madrid, 2019.

Rocío Cañamares and Pablo Castells. A Probabilistic Reformulation of Memory-Based Collaborative Filtering - Implications on Popularity Biases. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2017)*, pages 215–224, Tokyo, Japan, August 2017. ACM. doi:10.1145/3077136.3080836.

Rocío Cañamares and Pablo Castells. From the PRP to the Low Prior Discovery Recall Principle for Recommender Systems. In *Proceedings of the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2018)*, page 1081–1084, Ann Arbor, Michigan, USA, July 2018. ACM. doi:10.1145/3209978.3210076.

Rocío Cañamares, Marcos Redondo, and Pablo Castells. Multi-Armed Recommender System Bandit Ensembles. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, Copenhagen, Denmark, September 2019. ACM. doi:10.1145/3298689.3346984.

Rocío Cañamares, Pablo Castells, and Alistair Moffat. Offline evaluation options for recommender systems. *Information Retrieval*, 23:387–410, August 2020. doi:10.1007/s10791-020-09371-3.

Xiongcai Cai, Michael Bain, Alfred Krzywicki, Wayne Wobcke, Yang Sok Kim, Paul Compton, and Ashesh Mahidadia. Collaborative Filtering for People to People Recommendation in Social Networks. In *Proceedings of the 23rd Australasian Joint Conference on Artificial Intelligence (AI 2010)*, volume 6464 of *Lecture Notes in Computer Science*, pages 476–485, Adelaide, Australia, December 2010. Springer. doi:10.1007/978-3-642-17432-2_48.

Arthur Câmara and Claudia Hauff. Diagnosing BERT with Retrieval Heuristics. In *Proceedings of the 42nd European Conference on Information Retrieval (ECIR 2020)*, number 12035 in Lecture Notes in Computer Science, pages 605–618, Online, April 2020. Springer. doi:10.1007/978-3-030-45439-5_40.

Pedro G. Campos, Fernando Díez, and Iván Cantador. Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. *User Modelling User-Adapted Interaction*, 24(1-2):67–119, February 2014. doi:10.1007/s11257-012-9136-x.

Carlo Vittorio Cannistraci, Gregorio Alanis-Lobato, and Timothy Ravasi. From link-prediction in brain connectomes and protein interactomes to the local-community-paradigm in complex networks. *Scientific Reports*, 3(1613), December 2013. doi:10.1038/srep01613.

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai, and Hang Li. Learning to Rank: From Pairwise Approach to Listwise Approach. Microsoft Research Technical Report MSR-TR-2007-40, Microsoft Research, April 2007.

Jaime Carbonell and Jade Goldstein. The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1998)*, page 335–336, Melbourne, Australia, August 1998. ACM. doi:10.1145/290941.291025.

Benjamin A. Carterette. Multiple Testing in Statistical Analysis of Systems-Based Information Retrieval Experiments. *ACM Transactions on Information Systems*, 30(1):4:1–4:34, February 2012. doi:10.1145/2094072.2094076.

Pablo Castells, Neil J. Hurley, and Saul Vargas. Novelty and Diversity in Recommender Systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 881–918. Springer, Boston, MA, USA, 2nd edition, 2015. doi:10.1007/978-1-4899-7637-6_26.

Damon Centola. The Social Origins of Networks and Diffusion. *American Journal of Sociology*, 120(5):1295–1338, March 2015. doi:10.1086/681275.

Allison J.B. Chaney, David M. Blei, and Tina Eliassi-Rad. A Probabilistic Model for Using Social Networks in Personalized Item Recommendation. In *Proceedings of the 9th ACM Conference on Recommender Systems (RecSys 2015)*, pages 43–50, Vienna, Austria, September 2015. ACM. doi:10.1145/2792838.2800193.

Olivier Chapelle and Lihong Li. An Empirical Evaluation of Thompson Sampling. In *Proceedings of the 25th Anual Conference on Neural Information Processing Systems (NeurIPS 2011)*, pages 2249–2257, Granada, Spain, December 2011.

Olivier Chapelle, Donald Metlzer, Ya Zhang, and Pierre Grinspan. Expected reciprocal rank for graded relevance. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 621–630, Hong Kong, China, November 2009. ACM. doi:10.1145/1645953.1646033.

Olivier Chapelle, Shihao Ji, Ciya Liao, Emre Velipasaoglu, Larry Lai, and Su-Lin Wu. Intent-based diversification of web search results: metrics and algorithms. *Information Retrieval*, 14(6):572–592, May 2011. doi:10.1007/s10791-011-9167-7.

Olivier Chapelle, Thorsten Joachims, Filip Radlinski, and Yisong Yue. Large-Scale Validation and Analysis of Interleaved Search Evaluation. *ACM Transactions on Information Systems*, 30(1):6:1–6:41, February 2012. doi:10.1145/2094072.2094078.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Techniques. *Journal of Artificial Intelligence Research*, 16:321–357, June 2002. doi:10.1613/jair.953.

Jilin Chen, Werner Geyer, Casey Dugan, Michael Muller, and Ido Guy. Make New Friends, but Keep the Old: Recommending People on Social Networking Sites. In *Proceedings of the 27th SIGCHI Conference on Human Factors in Computing Systems (CHI 2009)*, pages 201–210, Boston, Massachussets, USA, April 2009. ACM. doi:10.1145/1518701.1518735.

Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. Novelty and Diversity in Information Retrieval Evaluation. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2008)*, pages 659–666, Singapore, Singapore, July 2008. ACM. doi:10.1145/1390334.1390446.

Aaron Clauset, Mark E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Physical Review E*, 70(6):066111:1–066111:6, December 2004. doi:10.1103/PhysRevE.70.066111.

Aaron Clauset, Cristopher Moore, and Mark E. J. Newman. Hierarchical structure and the prediction of missing links in networks. *Nature*, 453:98–101, May 2008. doi:10.1038/nature06830.

Stéphane Clinchant and Eric Gaussier. A Theoretical Analysis of Pseudo-Relevance Feedback Models. In *Proceedings of the 2013 Conference on the Theory of Information Retrieval (ICTIR 2013)*, pages 6–13, Copenhagen, Denmark, September 2013. ACM. doi:10.1145/2499178.2499179.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. The MIT Press, 3rd edition, 2009.

Federico Corò, Gianlorenzo D'Angelo, and Yllka Velaj. Recommending Links to Maximize the Influence in Social Networks. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI 2019)*, pages 2195–2201. IJCAI, July 2019. doi:10.24963/ijcai.2019/304.

Nick Craswell, Bashar Mitra, Emine Yilmaz, Daniel Campos, and Ellen M. Voorhees. Overview of the TREC 2019 Deep Learning Track. In *Proceedings of The 28th Text REtrieval Conference (TREC 2019)*, Gaithersburg, Maryland, USA, November 2019. NIST.

Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of Recommender Algorithms on Top-n Recommendation Tasks. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010)*, pages 39–46, Barcelona, Spain, September 2010. ACM. doi:10.1145/1864708.1864721.

W. Bruce Croft, Donald Metzler, and Trevor Strohman. *Search Engines. Information Retrieval in Practice*. Pearson, 2010.

Willian Cukierski, Benjamin Hamner, and Bo Yang. Graph-based features for supervised link prediction. In *Proceedings of the 2011 International Joint Conference on Neural Networks (IJCNN 2011)*, pages 1237–1244, San Jose, California, USA, August 2011. IEEE. doi:10.1109/IJCNN.2011.6033365.

Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. Are We Really Making Much Progress? A Worrying Analysis of Recent Neural Recommendation Approaches. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, page 101–109, Copenhagen, Denmark, September 2019. ACM. doi:10.1145/3298689.3347058.

Elizabeth M. Daly, Werner Geyer, and David R. Millen. The Network Effects of Recommending Social Connections. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010)*, page 301–304, Barcelona, Spain, September 2010. ACM. doi:10.1145/1864708.1864772.

Peter Dankelmann, Wayne Goddard, and Christine Swart. The Average Eccentricity of a Graph and its Subgraphs. *Utilitas Mathematica*, 65:41–51, May 2004.

Gautam Das, Nick Koudas, Manos Papagelis, and Sushruth Puttaswamy. Efficient Sampling of Information in Social Networks. In *Proceedings of the 2008 ACM Workshop on Search in Social Media (SSM 2008)*, page 67–74, Napa Valley, California, USA, October 2008. ACM. doi:10.1145/1458583.1458594.

Marco de Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. Semantics-Aware Content-Based Recommender Systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 119–159. Springer, 2nd edition, 2015. doi:10.1007/978-1-4899-7637-6_4.

Pasquale De Meo, Emilio Ferrara, Giacomo Fiumara, and Alessandro Provetti. On Facebook, Most Ties Are Weak. *Communications of the ACM*, 57(11):78–84, November 2014. doi:10.1145/2629438.

Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic Algorithms for Replicated Database Maintenance. In *Proceedings of the 6th Annual ACM Symposium on Principles of Distributed Computing (PODC 1987)*, pages 1–12, Vancouver, British Columbia, Canada, 1987. ACM. doi:10.1145/41840.41841.

Fernando Diaz, Donald Metzler, and Sihem Amer-Yahia. Relevance and Ranking in Online Dating Systems. In *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2010)*, page 66–73, Geneva, Switzerland, July 2010. ACM. doi:10.1145/1835449.1835463.

Benjamin Doerr, Mahmoud Fouz, and Tobias Friedrich. Social Networks Spread Rumors in Sublogarithmic Time. In *Proceedings of the 43rd Annual ACM Symposium on Theory of Computing (STOC 2011)*, pages 21–30, San Jose, California, USA, June 2011. ACM. doi:10.1145/1993636.1993640.

Benjamin Doerr, Mahmoud Fouz, and Tobias Friedrich. Why Rumors Spread so Quickly in Social Networks. *Communications of the ACM*, 55(6):70–75, June 2012. doi:10.1145/2184319.2184338.

Robert Dorfman. A Formula for the Gini Coefficient. *The Review of Economics and Statistics*, 61(1):146–149, February 1979. doi:10.2307/1924845.

Robin Ian MacDonald Dunbar. Coevolution of neocortical size, group size and language in humans. *Behavioral and Brain Sciences*, 16(4):681–694, December 1993. doi:10.1017/S0140525X00032325.

Émile Durkheim. *De la division du travail social: étude sur l'organisation des sociétés supérieyres*. Librairie Félix Alcan, 1893.

Nathan Eagle, Michael Macy, and Rob Claxton. Network Diversity and Economic Development. *Science*, 328 (5981):1029–1031, May 2010. doi:10.1126/science.1186605.

David Easley and Jon Kleinberg. *Networks, Crowds, and Markets*. Cambridge University Press, 2010.

Michael D. Ekstrand, F. Maxwell Harper, Martijn C. Willemsen, and Joseph A. Konstan. User Perception of Differences in Recommender Algorithms. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys 2014)*, page 161–168, Foster City, Silicon Valley, California, USA, October 2014. ACM. doi:10.1145/2645710.2645737.

Michael D. Ekstrand, Robin Burke, and Fernando Diaz. Fairness and Discrimination in Recommendation and Retrieval. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, pages 576–577, Copenhagen, Denmark, September 2019. ACM. doi:10.1145/3298689.3346964.

Mehdi Elahi, Francesco Ricci, and Neil Rubens. A survey of active learning in collaborative filtering recommender systems. *Computer Science Review*, 20:29 – 50, May 2016. doi:10.1016/j.cosrev.2016.05.002.

Ivan Erdelyi. On the matrix equation $Ax = \lambda Bx$. *Journal of Mathematical Analysis and Applications*, 17(1): 119–132, January 1967. doi:10.1016/0022-247X(67)90169-2.

Paul Erdös and Alfréd Rényi. On Random Graphs I. *Publicationes Mathematicae Debrecen*, 6:290–297, 1959.

Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph Neural Networks for Social Recommendation. In *Proceedings of the The Web Conference 2019 (WWW 2019)*, pages 417–426, San Francisco, CA, USA, May 2019. ACM. doi:10.1145/3308558.3313488.

Hui Fang and ChengXiang Zhai. Semantic Term Matching in Axiomatic Approaches to Information Retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2006)*, pages 115–122, Seattle, WA, USA, August 2006. ACM. doi:10.1145/1148170.1148193.

Hui Fang, Tao Tao, and Chengxiang Zhai. A Formal Study of Information Retrieval Heuristics. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004)*, pages 49–56, Sheffield, United Kingdom, July 2004. ACM. doi:10.1145/1008992.1009004.

Hui Fang, Tao Tao, and Chengxiang Zhai. Diagnostic Evaluation of Information Retrieval Models. *ACM Transactions on Information Systems*, 29(2):7:1–7:42, April 2011. doi:10.1145/1961209.1961210.

Mehrdad Farajtabar, Yichen Wang, Manuel Gomez-Rodriguez, Shuang Li, Hongyuan Zha, and Le Song. COEVOLVE: A Joint Point Process Model for Information Diffusion and Network Evolution. *Journal of Machine Learning Research*, 18(41):1–49, 2017.

Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, June 2006. doi:10.1016/j.patrec.2005.10.010.

Alexander Felfernig and Robin Burke. Constraint-based recommender systems: technologies and research issues. In *Proceedings of the 10th International Conference on Electronic Commerce (ICEC 2008)*, pages 3:1–3:10, Inssbruck, Austria, August 2008. ACM. doi:10.1145/1409540.1409544.

Daniel M. Fleder and Kartik Hosanagar. Recommender Systems and Their Impact on Sales Diversity. In *Proceedings of the 8th ACM Conference on Electronic Commerce (EC 2007)*, page 192–199, San Diego, California, USA, June 2007. ACM. doi:10.1145/1250910.1250939.

Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3):75–174, February 2010. doi:10.1016/j.physrep.2009.11.002.

François Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saraens. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369, March 2007. doi:10.1109/TKDE.2007.46.

Norbert Fuhr. Optimum Polynomial Retrieval Functions Based on the Probability Ranking Principle. *ACM Transactions on Information Systems*, 7(3):183–204, July 1989. doi:10.1145/65943.65944.

Yasser Ganjisaffar, Rich Caruana, and Cristina Videira Lopes. Bagging Gradient-boosted Trees for High Precision, Low Variance Ranking Models. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 85–94, Beijing, China, July 2011. ACM. doi:10.1145/2009916.2009932.

Claudio Gentile, Shuai Li, and Giovanni Zappella. Online Clustering of Bandits. In *Proceedings of the 31st International Conference on International Conference on Machine Learning (ICML 2014)*, pages 757–765, Beijing, China, June 2014.

Dorota Glowacka. Bandit Algorithms in Information Retrieval. *Foundations and Trends in Information Retrieval*, 13(4):299–424, May 2019. doi:10.1561/1500000067.

Ashish Goel, Pankaj Gupta, John Sirois, Dong Wang, Aneesh Sharma, and Siva Gurumurthy. The who-to-follow system at Twitter: Strategy, algorithms, and revenue impact. *Interfaces*, 45(1):98–107, February 2015. doi:10.1287/inte.2014.0784.

Jennifer Golbeck. Generating Predictive Movie Recommendations from Trust in Social Networks. In *Proceedings of the 4th International Conference on Trust Management (iTrust 2006)*, pages 93–104, Pisa, Italy, May 2006. Springer. doi:10.1007/11755593_8.

David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, December 1992. doi:10.1145/138859.138867.

Jacob Goldenberg, Barak Libai, and Eitan Muller. Talk of the Network: A Complex System Look at the Underlying Process of Word-of-Mouth. *Marketing Letters*, 12:211–223, August 2001. doi:10.1023/A:1011122126881.

Scott A. Golder, Alice Marwick, Sarita Yardi, and danah boyd. A structural approach to contact recommendations in online social networks. In *Proceedings of the Workshop on Search in Social Media (SSM 2009) at the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2009)*, Boston, MA, USA, July 2009.

Leo A. Goodman. Snowball Sampling. *The Annals of Mathematical Statistics*, 32(1):148–170, March 1961.

Mark S. Granovetter. The Strength of Weak Ties. *American Journal of Sociology*, 78(6):1360–1380, May 1973. doi:10.1086/225469.

Aditya Grover and Jure Leskovec. Node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, pages 855–864, San Francisco, California, USA, August 2016. ACM. doi:10.1145/2939672.2939754.

Alois Gruson, Praveen Chandar, Christophe Charbuillet, James McInerney, Samantha Hansen, Damien Tardieu, and Ben Carterette. Offline Evaluation to Make Decisions About Playlist Recommendation Algorithms. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining (WSDM 2019)*, page 420–428, Melbourne VIC, Australia, January 2019. ACM. doi:10.1145/3289600.3291027.

Adrien Guille, Hakim Hacid, Cecile Favre, and Djamel A. Zighed. Information Diffusion in Online Social Networks: A Survey. *ACM SIGMOD Record*, 42(2):17–28, July 2013. doi:10.1145/2503792.2503797.

Roger Guimerà and Marta Sales-Pardo. Missing and spurious interactions and the reconstruction of complex networks. *Proceedings of the National Academy of Sciences*, 106(52):22073–22078, December 2009. doi:10.1073/pnas.0908366106.

Asela Gunawardana and Guy Shani. Evaluating Recommender Systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 265–308. Springer, Boston, MA, USA, 2nd edition, 2015. doi:10.1007/978-1-4899-7637-6_8.

Pankaj Gupta, Ashish Goel, Jimmy Lin, Aneesh Sharma, Dong Wang, and Reza Zadeh. WTF: The Who to Follow Service at Twitter. In *Proceedings of the 22nd international conference on World Wide Web (WWW 2013)*, pages 505–514, Rio de Janeiro, Brazil, May 2013. ACM. doi:10.1145/2488388.2488433.

Ido Guy. Social Recommender Systems. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 511–543. Springer, Cham, Switzerland, 2015. doi:10.1007/978-1-4899-7637-6_15.

Ido Guy. People Recommendation on Social Media. In Peter Brusilovsky and Daqing He, editors, *Social Information Access: Systems and Technologies*, number 10100 in Lecture Notes in Computer Science, pages 570–623. Springer, 2018. doi:10.1007/978-3-319-90092-6_15.

John Hannon, Mike Bennett, and Barry Smyth. Recommending Twitter users to follow using content and collaborative filtering approaches. In *Proceedings of the 4th ACM conference on Recommender Systems (RecSys 2010)*, pages 199–206, Barcelona, Spain, September 2010. ACM. doi:10.1145/1864708.1864746.

Negar Hariri, Bamshad Mobasher, and Robin Burke. Context Adaptation in Interactive Recommender Systems. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys 2014)*, page 41–48, Foster City, Silicon Valley, California, USA, October 2014. ACM. doi:10.1145/2645710.2645753.

F. Maxwell Harper and Joseph A. Konstan. The MovieLens Datasets: History and Context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, January 2016. doi:10.1145/2827872.

Mohammad Al Hasan, Vineet Chaoji, Saeed Salem, and Mohammed J. Zaki. Link Prediction Using Supervised Learning. In *Proceedings of SDM 06 Workshop on Link Analysis, Counterterrorism and Security*, Bethesda, Maryland, April 2006. SIAM.

Claudia Hauff. Foundations of Machine Learning for IR. In *12th European Summer School in Information Retrieval (ESSIR 2019)*, Milan, Italy, July 2019.

Hussein Hazimeh and ChengXiang Zhai. Axiomatic Analysis of Smoothing Methods in Language Models for Pseudo-Relevance Feedback. In *Proceedings of the 2015 International Conference on The Theory of Information Retrieval (ICTIR 2015)*, page 141–150, Northampton, Massachusetts, USA, September 2015. ACM. doi:10.1145/2808194.2809471.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng 7 Chua. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web (WWW 2017)*, pages 173–182, Perth, Australia, April 2017. IW3C2. doi:10.1145/3038912.3052569.

Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating Collaborative Filtering Recommender Systems. *ACM Transactions on Information Systems*, 22(1):5–53, January 2004. doi:10.1145/963770.963772.

Herbert W. Hethcote. The Mathematics of Infectious Diseases. *SIAM Review*, 42(4):599–563, December 2000. doi:10.1137/S0036144500371907.

Russell A. Hill and Robin Ian MacDonald Dunbar. Social network size in humans. *Human Nature*, 14:53–72, March 2003. doi:10.1007/s12110-003-1016-y.

Katja Hofmann, Shimon Whiteson, and Maarten De Rijke. Fidelity, Soundness, and Efficiency of Interleaved Comparison Methods. *ACM Transactions on Information Systems*, 31(4), November 2013. doi:10.1145/2536736.2536737.

Sounman Hong and Sun Hyoung Kim. Political polarization on twitter: Implications for the use of social media in digital governments. *Government Information Quarterly*, 33(4):777–782, October 2016. doi:10.1016/j.giq.2016.04.007.

Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative Filtering for Implicit Feedback Datasets. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008)*, pages 263–272, Pisa, Italy, December 2008. IEEE. doi:10.1109/ICDM.2008.22.

Jeff Huang and Efthimis N. Efthimiadis. Analyzing and Evaluating Query Reformulation Strategies in Web Search Logs. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, page 77–86, Hong Kong, China, November 2009. ACM. doi:10.1145/1645953.1645966.

Xinyi Lisa Huang, Mitul Tiwari, and Sam Shah. Structural Diversity in Social Recommender Systems. In *Proceedings of the 5th ACM RecSys Workshop on Recommender Systems and the Social Web (RecSys RSWeb 2013) co-located with the 7th ACM conference on Recommender Systems (RecSys 2013)*, pages 1–7, Hong Kong, China, 2013.

Neil Hurley and Mi Zhang. Novelty and Diversity in Top-N Recommendation – Analysis and Evaluation. *ACM Transactions on Internet Technology*, 10(4):14:1–14:30, March 2011. doi:10.1145/1944339.1944341.

Aleksandar Ilić. On the extremal properties of the average eccentricity. *Computers and Mathematics with Applications*, 64(9):2877–2885, November 2012. doi:10.1016/j.camwa.2012.04.023.

Paul Jaccard. Étude de la distribution florale dans une portion des Alpes et du Jura. *Bulletin de la Société Vaudoise des Sciences Naturelles*, 37(142):547–579, January 1901. doi:10.5169/seals-266450.

Mohsen Jamali and Martin Ester. A Matrix Factorization Technique with Trust Propagation for Recommendation in Social Networks. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010)*, page 135–142, Barcelona, Spain, September 2010. ACM. doi:10.1145/1864708.1864736.

Dietmar Jannach, Iman Kamehkhosh, and Geoffray Bonnin. Music Recommendations. In Shlomo Berkovsky, Iván Cantador, and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 481–518. World Scientific Publishing, Singapore, November 2018. doi:10.1142/9789813275355_0015.

Kalervo Järvelin and Jaana Kekäläinen. Cumulated Gain-Based Evaluation of IR Techniques. *ACM Transactions on Information Systems*, 20:422–446, October 2002. doi:10.1145/582415.582418.

Glen Jeh and Jennifer Widom. SimRank: A Measure of Structural-Context Similarity. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2002)*, pages 538–543, Edmonton, Alberta, Canada, July 2002. ACM. doi:10.1145/775047.775126.

Frederick Jelinek and Robert L. Mercer. Interpolated estimation of Markov source parameters from sparse data. In Edzard S. Gelsema and Laveen N. Kanal, editors, *Pattern Recognition in Practice*, pages 381–402. North-Holland, Amsterdam, Netherlands, May 1980.

Leo Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18:39–43, March 1953. doi:10.1007/BF02289026.

Jaya Kawale, Hung H. Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. Efficient Thompson Sampling for Online Matrix-Factorization Recommendation. In *Proceedings of the 29th Anual Conference on Neural Information Processing Systems (NeurIPS 2015)*, pages 1297–1305. Curran Associates, Inc., Montréal, Canada, December 2015.

David Kempe, Jon M. Kleinberg, and Éva Tardos. Maximizing the Spread of Influence through a Social Network. In *Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2003)*, page 137–146, Washington, D.C., USA, August 2003. ACM. doi:10.1145/956750.956769.

Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *Journal of the ACM*, 46(5):604–632, September 1999. doi:10.1145/324133.324140.

Ioannis Konstas, Vassilios Stathopoulos, and Joemon M. Jose. On Social Networks and Collaborative Recommendation. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2009)*, page 195–202, Boston, Massachussets, USA, July 2009. doi:10.1145/1571941.1571977.

Irena Koprinska and Kalina Yacef. People-to-People Reciprocal Recommenders. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 545–567. Springer, Boston, MA, 2015. doi:10.1007/978-1-4899-7637-6_16.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix Factorization Techniques for Recommender Systems. *Computer*, 42(8):30–37, August 2009. doi:10.1109/MC.2009.263.

Denis Kotkov, Joseph A. Konstan, Qian Zhao, and Jari Veijalainen. Investigating Serendipity in Recommender Systems Based on Real User Feedback. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC 2018)*, pages 1341–1350, Pau, France, April 2018. ACM. doi:10.1145/3167132.3167276.

John Lafferty and Chengxiang Zhai. Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 111–119, New Orleans, Louisiana, USA, September 2001. ACM. doi:10.1145/383952.383970.

Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4):046110, October 2008. doi:10.1103/PhysRevE.78.046110.

Amy N. Langville and Carl D. Meyer. *Google's PageRank and Beyond: The Science of Search Engine Rankings*. Princeton University Press, 2006.

Tor Lattimore and Csaba Szepesvári. *Bandit Algorithms*. Cambridge University Press, July 2020. doi:10.1017/9781108571401.

Victor Lavrenko and W. Bruce Croft. Relevance based language models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2001)*, pages 120–127, New Orleans, Louisiana, USA, September 2001. ACM. doi:10.1145/383952.383972.

Joon Ho Lee. Analyses of Multiple Evidence Combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 1997)*, pages 267–276, Philadelphia, PA, USA, July 1997. ACM. doi:10.1145/258525.258587.

Elizabeth A. Leicht, Petter Holme, and Mark E.J. Newman. Vertex similarity in networks. *Physical Review E*, 73(2):026120, February 2006. doi:10.1103/PhysRevE.73.026120.

Ronny Lempel and Schlomo Moran. SALSA: The Stochastic Approach for Link-Structure Analysis. *ACM Transactions on Information Systems*, 19(2):131–160, April 2001. doi:10.1145/382979.383041.

Jure Leskovec and Christos Faloutsos. Sampling from Large Graphs. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2006)*, page 631–636. ACM, August 2006. doi:10.1145/1150402.1150479.

Jure Leskovec, Jon Kleinberg, and Christos Faloutsos. Graph Evolution: Densification and Shrinking Diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1):2:1–2:41, March 2007. doi:10.1145/1217299.1217301.

Chang Li, Haoyun Feng, and Maarten de Rijke. Cascading hybrid bandits: Online learning to rank for relevance and diversity. In *Proceedings of the 14th ACM Conference on Recommender Systems (RecSys 2020)*, page 33–42, Online, September 2020. ACM. doi:10.1145/3383313.3412245.

Lihong Li, Wei Chu, John Langford, and Robert E. Schapire. A Contextual-Bandit Approach to Personalized News Article Recommendation. In *Proceedings of the 19th International Conference on World Wide Web (WWW 2010)*, page 661–670, Raleigh, North Carolina, USA, April 2010. ACM. doi:10.1145/1772690.1772758.

Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the 4th ACM international conference on Web search and data mining (WSDM 2011)*, pages 297–306, Hong Kong, China, February 2011. ACM. doi:10.1145/1935826.1935878.

Shuai Li, Alexandros Karatzoglou, and Claudio Gentile. Collaborative Filtering Bandits. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2016)*, pages 539–548, Pisa, Italy, July 2016. ACM. doi:10.1145/2911451.2911548.

Zhepeng (Lionel) Li, Xiao Fang, and Olivia R. Liu Sheng. A Survey of Link Recommendation for Social Networks: Methods, Theoretical Foundations, and Future Research Directions. *ACM Transactions on Management Information Systems*, 9(1):1:1–1:26, October 2017. doi:10.1145/3131782.

David Liben-Nowell and Jon Kleinberg. The link-prediction problem for social networks. *Journal of the American Society for Information Science and Technology*, 58(7):1019–1031, March 2007. doi:10.1002/asi.20591.

Ryan N. Lichtenwalter, Jake T. Lussier, and Nitesh V. Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD 2010)*, page 243–252, Washington, DC, USA, July 2010. ACM. doi:10.1145/1835804.1835837.

Tie-Yan Liu. Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval*, 3(3):225–331, March 2007. doi:10.1561/1500000016.

Fabiana Lorenzi and Francesco Ricci. Case-Based Recommender Systems: A Unifying View. In Bamshad Mobasher and Sarabjot S. Anand, editors, *Proceedings of the Intelligent Techniques for Web Personalization Workshop (ITWP 2003) at IJCAI 2003*, volume 3169 of *Lecture Notes in Computer Science*, pages 89–113, Acapulco, Mexico, August 2003. Springer. doi:10.1007/11577935_5.

Jonathan Louëdec, Max Chevalier, Josiane Mothe, Aurélien Garivier, and Sébastien Gerchinovitz. A Multiple-Play Bandit Algorithm Applied to Recommender Systems. In *Proceedings of the 28th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2015)*, pages 67–72, Hollywood, Florida, USA, May 2015. AAAI Press.

Linyuan Lü and Tao Zhou. Link prediction in complex networks: A survey. *Physica A: Statistical Mechanics and its Applications*, 390(6):1150–1170, March 2011. doi:10.1016/j.physa.2010.11.027.

Linyuan Lü, Ci-Hang Jin, and Tao Zhou. Similarity index based on local paths for link prediction of complex networks. *Physical Review E*, 80(4):046122, October 2009. doi:10.1103/PhysRevE.80.046122.

Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. SoRec: Social Recommendation Using Probabilistic Matrix Factorization. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management (CIKM 2008)*, page 931–940, Napa Valley, California, USA, October 2008. ACM. doi:10.1145/1458082.1458205.

Craig Macdonald, Richard McCreadie, Rodrygo L.T. Santos, and Iadh Ounis. From puppy to maturity: Experiences in developing Terrier. In *Proceedings of the SIGIR 2012 Workshop in Open Source Information Retrieval (OSIR 2012)*, pages 60–63, Portland, Oregon, USA, August 2012.

Craig Macdonald, Rodrygo L.T. Santos, and Iadh Ounis. The whens and hows of learning to rank for web search. *Information Retrieval*, 16(5):584–628, October 2013a. doi:10.1007/s10791-012-9209-9.

Craig Macdonald, Rodrygo L.T. Santos, Iadh Ounis, and Ben He. About learning models with multiple query-dependent features. *ACM Transactions on Information Systems*, 31(3):11:1–11:39, August 2013b. doi:10.1145/2493175.2493176.

David J. C. MacKay and Linda C. Bauman Peto. A hierarchical Dirichlet language model. *Natural Language Engineering*, 1(3):289–307, September 1995. doi:10.1017/S1351324900000218.

Jochen Malinowski, Tobias Keim, Oliver Wendt, and Tim Weitzel. Matching People and Jobs: A Bilateral Recommendation Approach. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06)*, Kauai, Hawaii, USA, January 2006. IEEE. doi:10.1109/HICSS.2006.266.

Víctor Martínez, Fernando Berzal, and Juan Carlos Cubero. A Survey of Link Prediction in Complex Networks. *ACM Computing Surveys*, 49(4):69:1–69:33, February 2017. doi:10.1145/3012704.

Julian McAuley and Jure Leskovec. Learning to Discover Social Circles in Ego Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems (NIPS 2012)*, pages 539–547, Lake Tahoe, Nevada, December 2012. Curran Associates Inc.

David W. McDonald and Mark S. Ackerman. Expertise Recommender: A Flexible Recommendation System and Architecture. In *Proceedings of the 2000 ACM Conference on Computer Supported Cooperative Work (CSCW 2000)*, page 231–240, Philadelphia, Pennsylvania, USA, December 2000. ACM. doi:10.1145/358916.358994.

James McInerney, Benjamin Lacker, Samantha Hansen, Karl Higley, Hugues Bouchard, Alois Gruson, and Rishabh Mehrotra. Explore, Exploit, and Explain: Personalizing Explainable Recommendations with Bandits. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018)*, page 31–39, Vancouver, British Columbia, Canada, October 2018. ACM. doi:10.1145/3240323.3240354.

Miller McPherson, Lynn Smith-Lovin, and James M. Cook. Birds of a Feather: Homophily in Social Networks. *Annual Review of Sociology*, 27(1):415–444, August 2001. doi:10.1146/annurev.soc.27.1.415.

Rishabh Mehrotra, Niannan Xue, and Mounia Lalmas. Bandit based optimization of multiple objectives on a music streaming platform. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2020)*, page 3224–3233, Online, August 2020. ACM. doi:10.1145/3394486.3403374.

Zaiqiao Meng, Shangsong Liang, Hongyan Bao, and Xiangliang Zhang. Co-Embedding Attributed Networks. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining (WSDM 2019)*, pages 393–401, Melbourne, Australia, January 2019. ACM. doi:10.1145/3289600.3291015.

Aditya Krishna Menon and Charles Elkan. Link Prediction via Matrix Factorization. In *Proceedings of the 2011 Joint European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2011)*, volume 6912 of *Lecture Notes in Computer Science*, pages 437–452, Athens, Greece, September 2011. Springer. doi:10.1007/978-3-642-23783-6_28.

Carl D. Meyer Jr. The Role of the Group Generalized Inverse in the Theory of Finite Markov Chains. *SIAM Review*, 17(3):443–464, July 1975. doi:10.1137/1017044.

Stanley Milgram. The small world problem. *Psychology Today*, 1(1):61–67, May 1967.

Bhaskar Mitra and Nick Craswell. An Introduction to Neural Information Retrieval. *Foundations and Trends in Information Retrieval*, 13(1):1–126, 2018. doi:10.1561/1500000061.

Jacob Levy Moreno. *Who Shall Survive: A New Approach to the Problem of Human Interrelations*. Number 58 in Nervous and Mental Disease monograph series. Nervous and Mental Disease Publisher Co., 1934. doi:10.1037/10648-000.

Cameron Musco, Christopher Musco, and Charalampos E. Tsourakakis. Minimizing Polarization and Disagreement in Social Networks. In *Proceedings of The Web Conference 2018 (WWW 2018)*, pages 369–378, Lyon, France, April 2018. IW3C2. doi:10.1145/3178876.3186103.

Seth A. Myers, Aneesh Sharma, Pankaj Gupta, and Jimmy Lin. Information Network or Social Network? The Structure of the Twitter Follow Graph. In *Companion Proceedings of the 23rd International Conference on World Wide Web (WWW 2014)*, pages 493–498, Seoul, Korea, April 2014. ACM. doi:10.1145/2567948.2576939.

Ramesh Nallapati. Discriminative Models for Information Retrieval. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004)*, page 64–71, Sheffield, United Kingdom, July 2004. ACM. doi:10.1145/1008992.1009006.

James Neve and Iván Palomares. Latent Factor Models and Aggregation Operators for Collaborative Filtering in Reciprocal Recommender Systems. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, pages 219–227, Copenhagen, Denmark, September 2019. ACM. doi:10.1145/3298689.3347026.

Mark E. J. Newman. Clustering and preferential attachment in growing networks. *Physical Review E*, 64: 025102, April 2001. doi:10.1103/PhysRevE.64.025102.

Mark E. J. Newman. The Structure and Function of Complex Networks. *SIAM Review*, 45(2):167–256, 2003. doi:10.1137/S003614450342480.

Mark E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, June 2006. doi:10.1073/pnas.0601602103.

Mark E. J. Newman. *Networks: An Introduction*. Oxford University Press, 2nd edition, 2018. doi:10.1093/oso/9780198805090.001.0001.

Mark E. J. Newman and Michelle Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69(2):026113, February 2004. doi:10.1103/PhysRevE.69.026113.

Xia Ning, Christian Desrosiers, and George Karypis. A Comprehensive Survey of Neighborhood-Based Recommendation Methods. In Francesco Ricci, Lior Rokach, and Bracha Shapira, editors, *Recommender Systems Handbook*, pages 37–77. Springer, Boston, MA, USA, 2015. doi:10.1007/978-1-4899-7637-6_2.

Gwenn Schurgin O'Keeffe, Kathleen Clarke-Pearson, Council of Communication, and Media. The Impact of Social Media on Children, Adolescents, and Families. *Pediatrics*, 127(4):800–804, April 2011. doi:10.1542/peds.2011-0054.

Iadh Ounis, Gianni Amati, Vassilis Plachouras, Ben He, Craig Macdonald, and Christina Lioma. Terrier: A High Performance and Scalable Information Retrieval Platform. In *Proceedings of the 2nd workshop on Open Source Information Retrieval (OSIR 2006) at SIGIR 2006*, pages 18–25, Seattle, WA, August 2006.

Manos Papagelis, Gautam Das, and Nick Koudas. Sampling Online Social Networks. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):662–676, March 2013. doi:10.1109/TKDE.2011.254.

Javier Parapar, Alejandro Bellogín, Pablo Castells, and Alvaro Barreiro. Relevance-based language modelling for recommender systems. *Information Processing and Management*, 49(4):966–980, July 2013. doi:10.1016/j.ipm.2013.03.001.

Eli Pariser. *The Filter Bubble: What the Internet Is Hiding From You*. Penguin Press, May 2011.

Nikos Parotsidis, Evaggelia Pitoura, and Panayiotis Tsaparas. Centrality-Aware Link Recommendations. In *Proceedings of the 9th ACM International Conference on Web Search and Data Mining (WSDM 2016)*, pages 503–512, San Francisco, California, USA, February 2016. ACM. doi:10.1145/2835776.2835818.

István Pilászy, Dávid Zibriczky, and Domonkos Tikk. Fast ALS-Based Matrix Factorization for Explicit and Implicit Feedback Datasets. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys 2010)*, pages 71–78, Barcelona, Spain, September 2010. ACM. doi:10.1145/1864708.1864726.

Luiz Augusto Sangoi Pizzato, Tomek Rej, Thomas Chung, Irena Koprinska, and Judy Kay. RECON:a reciprocal recommender for online dating. In *Proceedings of the 2010 ACM Conference on Recommender Systems (RecSys 2010)*, pages 207–214, Barcelona, Spain, September 2010. ACM. doi:10.1145/1864708.1864747.

Luiz Augusto Sangoi Pizzato, Tomasz Rej, Joshua Akehurst, Irena Koprinska, Kalina Yacef, and Judy Kay. Recommending people to people: the nature of reciprocal recommenders with a case study in online dating. *User Modeling and User-Adapted Interaction*, 23:447–488, November 2013. doi:10.1007/s11257-012-9125-0.

Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in Information Retrieval (SIGIR 1998)*, pages 275–281, Melbourne, Australia, August 1998. ACM. doi:10.1145/290941.291008.

Daniele Quercia and Licia Capra. FriendSensing: Recommending Friends Using Mobile Phones. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys 2009)*, pages 273–276, New York, New York, USA, October 2009. ACM. doi:10.1145/1639714.1639766.

Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036106, September 2007. doi:10.1103/PhysRevE.76.036106.

Erzsébet Ravasz, A.L. Somera, D.A. Mongru, Zoltan N. Oltvai, and Albert-László Barabàsi. Hierarchical Organization of Modularity in Metabolic Networks. *Science*, 297(5586):1551–1555, August 2002. doi:10.1126/science.1073374.

Daniël Rennings, Felipe Moraes, and Claudia Hauff. An Axiomatic Approach to Diagnosing Neural IR Models. In *Proceedings of the 41st European Conference on Information Retrieval (ECIR 2019)*, number 11437 in Lecture Notes in Computer Science, pages 489–503. Springer, April 2019. doi:10.1007/978-3-030-15712-8_32.

Paul Resnick and Hal R. Varian. Recommender Systems. *Communications of the ACM*, 40(3):56–58, March 1997. doi:10.1145/245108.245121.

Francesco Ricci, Lior Rokach, and Bracha Shapira, editors. *Recommender Systems Handbook*. Springer, 2nd edition, 2015. doi:10.1007/978-1-4899-7637-6.

Stephen E. Robertson. The Probability Ranking Principle in IR. *Journal of Documentation*, 33(4):294–304, December 1977.

Stephen E. Robertson and Hugo Zaragoza. The Probabilistic Relevance Framework : BM25 and Beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389, April 2009. doi:10.1561/1500000019.

Joseph J. Rocchio. Relevance Feedback in Information Retrieval. In Gerald Salton, editor, *The SMART Retrieval System – Experiments in Automatic Document Processing*, pages 313–323. Prentice Hall, January 1971.

Everett M. Rogers. *Diffusion of Innovations*. Free Press of Glencoe, 1962.

Corby Rosset, Bhaskar Mitra, Chenyan Xiong, Nick Craswell, Xia Song, and Saurabh Tiwary. An Axiomatic Approach to Regularizing Neural Ranking Models. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2019)*, pages 981–984, Paris, France, July 2019. ACM. doi:10.1145/3331184.3331296.

Marco Rossetti, Fabio Stella, and Markus Zanker. Contrasting Offline and Online Results When Evaluating Recommendation Algorithms. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys 2016)*, pages 31–34. ACM, September 2016. doi:10.1145/2959100.2959176.

Martin Rosvall and Carl T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, January 2008. doi:10.1073/pnas.0706851105.

Ian Ruthven and Mounia Lalmas. A survey on the use of relevance feedback for information access systems. *The Knowledge Engineering Review*, 18(2):95–145, June 2003. doi:10.1017/S0269888903000638.

Tetsuya Sakai. *Laboratory Experiments in Information Retrieval - Sample Sizes, Effect Sizes, and Statistical Power*, volume 40 of *The Information Retrieval Series*. Springer, 2018. doi:10.1007/978-981-13-1199-4.

Ruslan Salakhutdinov and Andriy Mnih. Probabilistic Matrix Factorization. In *Proceedings of the 20th International Conference on Neural Information Processing Systems (NIPS 2007)*, page 1257–1264. Curran Associates Inc., December 2007.

Ruslan Salakhutdinov and Andriy Mnih. Bayesian Probabilistic Matrix Factorization Using Markov Chain Monte Carlo. In *Proceedings of the 25th International Conference on Machine Learning (ICML 2008)*, pages 880–887, Helsinki, Finland, July 2008. ACM. doi:10.1145/1390156.1390267.

Gerard Salton and Michael J. McGill. *Introduction to modern information retrieval*. McGraw-Hill, Inc., 1983.

Gerard Salton, Andrew Wong, and Chungshu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, November 1975. doi:10.1145/361219.361220.

Mark Sanderson and W. Bruce Croft. The History of Information Retrieval Research. *Proceedings of the IEEE*, 100(Special Centennial Issue):1444–1451, May 2012. doi:10.1109/JPROC.2012.2189916.

Rodrygo L. T. Santos, Craig Macdonald, and Iadh Ounis. Selectively Diversifying Web Search Results. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM 2010)*, page 1179–1188, Toronto, Ontario, Canada, October 2010a. ACM. doi:10.1145/1871437.1871586.

Rodrygo L. T. Santos, Jie Peng, Craig Macdonald, and Iadh Ounis. Explicit Search Result Diversification through Sub-queries. In *Proceedings of the 32nd European Conference on Information Retrieval (ECIR 2010)*, number 5993 in Lecture Notes in Computer Science, pages 87–99, Milton Keynes, United Kingdom, March 2010b. Springer Berlin Heidelberg. doi:10.1007/978-3-642-12275-0_11.

Javier Sanz-Cruzado and Pablo Castells. Contact Recommendations in Social Networks. In Shlomo Berkovsky, Iván Cantador, and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 519–569. World Scientific Publishing, Singapore, November 2018a. doi:10.1142/9789813275355_0016.

Javier Sanz-Cruzado and Pablo Castells. Enhancing structural diversity in social networks by recommending weak ties. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys 2018)*, pages 233–241, Vancouver, BC, Canada, September 2018b. ACM. doi:10.1145/3240323.3240371.

Javier Sanz-Cruzado and Pablo Castells. Information Retrieval Models for Contact Recommendation in Social Networks. In *Proceedings of the 41st European Conference on Information Retrieval (ECIR 2019)*, number 11437 in Lecture Notes in Computer Science, pages 148–163, Cologne, Germany, April 2019a. Springer. doi:10.1007/978-3-030-15712-8_10.

Javier Sanz-Cruzado and Pablo Castells. Beyond Accuracy in Link Prediction. In *Proceedings of the 3rd Workshop on Social Media Personalization and Search (SoMePeaS 2019) co-located with the 41st European Conference on Information Retrieval (ECIR 2019)*, number 1245 in Communications in Computer and Information Science, pages 79–94, Cologne, Germany, April 2019b. Springer. doi:10.1007/978-3-030-52485-2_9.

Javier Sanz-Cruzado, Sofía M. Pepa, and Pablo Castells. Recommending Contacts in Social Networks Using Information Retrieval Models. In *Proceedings of the 5th Spanish Conference on Information Retrieval (CERI 2018)*, pages 19:1–19:8, Zaragoza, Spain, June 2018a. ACM. doi:10.1145/3230599.3230619.

Javier Sanz-Cruzado, Sofía M. Pepa, and Pablo Castells. Structural Novelty and Diversity in Link Prediction. In *Proceedings of the 9th International Workshop on Modelling Social Media (MSM 2018), in Companion of the The Web Conference 2018 (WWW 2018)*, pages 1347–1351, Lyon, France, April 2018b. IW3C2. doi:10.1145/3184558.3191576.

Javier Sanz-Cruzado, Pablo Castells, and Esther López. A Simple Multi-armed Nearest-neighbor Bandit for Interactive Recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems (RecSys 2019)*, pages 358–362, Copenhagen, Denmark, September 2019. ACM. doi:10.1145/3298689.3347040.

Javier Sanz-Cruzado, Pablo Castells, Craig Macdonald, and Iadh and Ounis. Effective Contact Recommendation in Social Networks by Adaptation of Information Retrieval Models. *Information Processing and Management*, 57(5):102285, September 2020a. doi:10.1016/j.ipm.2020.102285.

Javier Sanz-Cruzado, Craig Macdonald, Iadh Ounis, and Pablo Castells. Axiomatic Analysis of Contact Recommendation Methods in Social Networks: an IR perspective. In *Proceedings of the 42nd European Conference on Information Retrieval (ECIR 2020)*, number 12035 in Lecture Notes in Computer Science, pages 175–190, Online, April 2020b. Springer. doi:10.1007/978-3-030-45439-5_12.

Badrul Munir Sarwar, George Karypis, Joseph A. Konstan, and John T. Riedl. Application of Dimensionality Reduction in Recommender System – A case study. In *Proceedings of the 2nd Workshop on Web Mining for e-Commerce (WebKDD 2000) co-located with the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2000)*, Boston, Massachussets, USA, August 2000.

Venu Satuluri, Yao Wu, Xun Zheng, Yilei Qian, Brian Wichers, Qieyun Dai, Gui Ming Tang, Jerry Jiang, and Jimmy Lin. SimClusters: Community-Based Representations for Heterogeneous Recommendations at Twitter. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD 2020)*, page 3183–3193, Online, August 2020. ACM. doi:10.1145/3394486.3403370.

Anne Schuth, Floor Sietsma, Shimon Whiteson, Damien Lefortier, and Maarten de Rijke. Multileaved Comparisons for Fast Online Evaluation. In *Proceedings of the 23rd ACM International Conference on Information and Knowledge Management (CIKM 2014)*, page 71–80, Shangai, China, November 2014. doi:10.1145/2661829.2661952.

Shahin Sefati, Jan Neumann, and Hassan Sayyadi. TV and Movie Recommendations: The Comcast Case. In Shlomo Berkovsky, Iván Cantador, and Domonkos Tikk, editors, *Collaborative Recommendations: Algorithms, Practical Challenges and Applications*, pages 465–479. World Scientific Publishing, Singapore, November 2018. doi:10.1142/9789813275355_0014.

Guy Shani, David Heckerman, and Ronen I. Brafman. An MDP-Based Recommender System. *Journal of Machine Learning Research*, 6:1265–1295, September 2005.

Claude E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, July 1948. doi:10.1002/j.1538-7305.1948.tb01338.x.

Shuming Shi, Ji-Rong Wen, Qing Yu, Ruihua Song, and Wei-Ying Ma. Gravitation-Based Model for Information Retrieval. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005)*, pages 488–495. ACM, August 2005. doi:10.1145/1076034.1076117.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, Yutian Chen, Timothy Lillicrap, Fan Hui, Laurent Sifre, George van den Driessche, Thore Graepel, and Demis Hassabis. Mastering the game of Go without human knowledge. *Nature*, 550:354–359, October 2017. doi:10.1038/nature24270.

Amit Singhal, John Choi, Donald Hindle, David D. Lewis, and Fernando C. N. Pereira. AT&T at TREC-7. In *Proceedings of the 7th Text REtrieval Conference (TREC 1998)*, pages 186–198. NIST, November 1998.

Rashmi R. Sinha and Kirsten Swearingen. Comparing Recommendations Made by Online Systems and Friends. In *Proceedings of the 2nd DELOS Network of Excellence Workshop on Personalisation and Recommender Systems in Digital Libraries (DELOS 2001)*, Dublin, Ireland, June 2001. ERCIM.

Barry Smyth and Paul McClave. Similarity vs. Diversity. In *Proceedings of the 4th International Conference on Case-Based Reasoning (ICCBR 2001)*, volume 2080 of *Lecture Notes in Computer Science*, pages 347–361, Vancouver, BC, Canada, August 2001. Springer. doi:10.1007/3-540-44593-5_25.

Thorvald Julius Sørensen. A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons. *Biologiske Skrifter*, 5(4):1–34, 1948.

Ana-Andreea Stoica, Christopher Riederer, and Augustin Chaintreau. Algorithmic Glass Ceiling in Social Networks: The Effects of Social Recommendations on Network Diversity. In *Proceedings of The Web Conference 2018 (WWW 2018)*, pages 923–932, Lyon, France, April 2018. IW3C2. doi:10.1145/3178876.3186140.

Jessica Su, Aneesh Sharma, and Sharad Goel. The Effect of Recommendations on Network Structure. In *Proceedings of the 25th International Conference on World Wide Web (WWW 2016)*, page 1157–1167, Montréal, Québec, Canada, April 2016. IW3C2. doi:10.1145/2872427.2883040.

Sadegh Sulaimany, Mohammad Khansari, Peyman Zarrineh, Madelaine Daianu, Neda Jahanshad, Paul M. Thompson, and Ali Masoudi-Nejad. Predicting brain network changes in Alzheimer's disease with link prediction algorithms. *Molecular BioSystems*, 13(4):725–735, February 2017. doi:10.1039/C6MB00815A.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 2nd edition, November 2018.

Jiliang Tang, Xia Hu, and Huan Liu. Social recommendation: a review. *Social Network Analysis and Mining*, 3(4):1113–1133, October 2013. doi:10.1007/s13278-013-0141-9.

Tao Tao and ChengXiang Zhai. An Exploration of Proximity Measures in Information Retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 295–302. ACM, July 2007. doi:10.1145/1277741.1277794.

Loren Terveen and David W. McDonald. Social Matching: A Framework and Research Agenda. *ACM Transactions on Computer-Human Interaction*, 12(3):401–434, September 2005. doi:10.1145/1096737.1096740.

Ming-Feng Tsai, Tie-Yan Liu, Tao Qin, Hsin-Hsi Chen, and Wei-Ying Ma. FRank: A Ranking Method with Fidelity Loss. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, pages 383–390, Amsterdam, The Netherlands, July 2007. ACM. doi:10.1145/1277741.1277808.

John W. Tukey. Comparing Individual Means in the Analysis of Variance. *Biometrics*, 5(2):99–114, June 1949. doi:10.2307/3001913.

Ferdinand Tönnies. *Gemeinschaft und Gesellschaft*. Fues' Verlag, 1887.

Johan Ugander, Brian Karrer, Lars Backstrom, and Cameron Marlow. The Anatomy of the Facebook Social Graph. *arXiv:1111.4503*, November 2011.

Johan Ugander, Lars Backstrom, Cameron Marlow, and Jon Kleinberg. Structural diversity in social contagion. *Proceedings of the National Academy of Sciences*, 109(16):5962–5966, April 2012. doi:10.1073/pnas.1116502109.

Daniel Valcarce. Exploring statistical language models for recommender systems. In *Proceedings of the 9th ACM conference on Recommender Systems (RecSys 2015)*, pages 375–378, Vienna, Austria, September 2015. ACM. doi:10.1145/2792838.2796547.

Daniel Valcarce, Javier Parapar, and Álvaro Barreiro. Axiomatic Analysis of Language Modelling of Recommender Systems. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 25(Suppl. 2):113–127, June 2017. doi:10.1142/S0218488517400141.

Daniel Valcarce, Javier Parapar, and Álvaro Barreiro. Finding and analysing good neighbourhoods to improve collaborative filtering. *Knowledge-Based Systems*, 159:193–202, November 2018. doi:10.1016/j.knosys.2018.06.030.

David Vallet and Pablo Castells. Personalized Diversification of Search Results. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2012)*, page 841–850, Portland, Oregon, USA, August 2012. ACM. doi:10.1145/2348283.2348396.

Jorge Carlos Valverde-Rebaza and Alneu de Andrade Lopes. Exploiting behaviors of communities of Twitter users for link prediction. *Social Network Analysis and Mining*, 3(4):1063–1074, October 2013. doi:10.1007/s13278-013-0142-8.

Jorge Carlos Valverde-Rebaza, Mathieu Roche, Pascal Poncelet, and Alneu de Andrade Lopes. The role of location and social strength for friendship prediction in location-based social networks. *Information Processing and Management*, 54(4):475–489, July 2018. doi:10.1016/j.ipm.2018.02.004.

Saúl Vargas and Pablo Castells. Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems. In *Proceedings of the 5th ACM Conference on Recommender Systems (RecSys 2011)*, pages 109–116, Chicago, Illinois, USA, October 2011. ACM. doi:10.1145/2043932.2043955.

Saúl Vargas and Pablo Castells. Improving Sales Diversity by Recommending Users to Items. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys 2014)*, pages 145–152, Foster City, Silicon Valley, California, USA, October 2014. ACM. doi:10.1145/2645710.2645744.

Saúl Vargas, Pablo Castells, and David Vallet. Intent-Oriented Diversity in Recommender Systems. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2011)*, pages 1211–1212, Beijing, China, July 2011. ACM. doi:10.1145/2009916.2010124.

Saúl Vargas, Pablo Castells, and David Vallet. Explicit Relevance Models in Intent-Oriented Information Retrieval Diversification. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2012)*, page 75–84, Portland, Oregon, USA, August 2012. ACM. doi:10.1145/2348283.2348297.

Oriol Vinyals, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, Richard Powell, Timo Ewalds, Petko Georgiev, Junhyuk Oh, Dan Horgan, Manuel Kroiss, Ivo Danihelka, Aja Huang, Laurent Sifre, Trevor Cai, John P. Agapiou, Max Jaderberg, Alexander S. Vezhnevets, Rémi Leblond, Tobias Pohlen, Valentin Dalibard, David Budden, Yury Sulsky, James Molloy, Tom L. Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Yuhuai Wu, Roman Ring, Dani Yogatama, Dario Wünsch, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Koray Kavukcuoglu, Demis Hassabis, Chris Apps, and David Silver. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575:350–354, November 2019. doi:10.1038/s41586-019-1724-z.

Jun Wang, Arjen P. de Vries, and Marcel J. T. Reinders. Unified Relevance Models for Rating Prediction in Collaborative Filtering. *ACM Transactions on Information Systems*, 26(3):16:1–16:42, June 2008a. doi:10.1145/1361684.1361689.

Jun Wang, Stephen Robertson, Arjen P. de Vries, and Marcel J. T. Reinders. Probabilistic relevance ranking for collaborative filtering. *Information Retrieval*, 11(6):477–497, December 2008b. doi:10.1007/s10791-008-9060-1.

Qing Wang, Chunqiu Zeng, Wubai Zhou, Tao Li, Sundaraja Sitharama Iyengar, Larisa Shwartz, and Genady Ya. Grabarnik. Online Interactive Collaborative Filtering Using Multi-Armed Bandit with Dependent Arms. *IEEE Transactions on Knowledge and Data Engineering*, 31(8):1569–1580, August 2019a. doi:10.1109/TKDE.2018.2866041.

Xi Wang, Iadh Ounis, and Craig Macdonald. Comparison of Sentiment Analysis and User Ratings in Venue Recommendation. In *Proceedings of the 41st European Conference on Information Retrieval (ECIR 2019)*, volume 11437 of *Lecture Notes in Computer Science*, pages 215–228, Cologne, Germany, April 2019b. Springer. doi:10.1007/978-3-030-15712-8_14.

Yazhe Wang, Jamie Callan, and Baihua Zheng. Should We Use the Sample? Analyzing Datasets Sampled from Twitter's Stream API. *ACM Transactions on the Web*, 9(3):13:1–13:23, June 2015. doi:10.1145/2746366.

Duncan J. Watts and Steven Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998. doi:10.1038/30918.

Scott White and Padhraic Smyth. Algorithms for estimating relative importance in networks. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge Discovery and Data Mining (KDD 2003)*, pages 266–275, Washington, DC, USA, August 2003. ACM. doi:10.1145/956750.956782.

Jiehua Wu, Guoji Zhang, and Yazhou Ren. A balanced modularity maximization link prediction model in social networks. *Information Processing and Management*, 53(1):295 – 307, January 2017. doi:10.1016/j.ipm.2016.10.001.

Qiang Wu, Chris Burges, Krysta Svore, and Jianfeng Gao. Ranking, Boosting and Model Adaptation. Microsoft Research Technical Report MSR-TR-2008-109, Microsoft Research, October 2008.

Xin Xin, Alexandros Karatzoglou, Ioannis Arapakis, and Joemon M. Jose. Self-Supervised Reinforcement Learning for Recommender Systems. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2020)*, page 931–940, Online, July 2020. ACM. doi:10.1145/3397271.3401147.

Jun Xu and Hang Li. AdaRank: A Boosting Algorithm for Information Retrieval. In *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2007)*, page 391–398, Amsterdam, The Netherlands, July 2007. ACM. doi:10.1145/1277741.1277809.

Dingqi Yang, Daquing Zhang, Vincent W. Zheng, and Zhiyong Yu. Modeling User Activity Preference by Leveraging User Spatial Temporal Characteristics in LBSNs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(1):129–142, January 2015. doi:10.1109/TSMC.2014.2327053.

Zhao Yang, René Algesheimer, and Claudio J. Tessone. A Comparative Analysis of Community Detection Algorithms on Artificial Networks. *Scientific Reports*, 6:30750, August 2016. doi:10.1038/srep30750.

Tong Yu, Ole J. Mengshoel, Alvin Jude, Eugen Feller, Julien Forgeat, and Nimish Radia. Incremental learning for matrix factorization in recommender systems. In *Proceedings of the 2016 IEEE International Conference on Big Data (Big Data 2016)*, pages 1056–1063, Washington, D.C., USA, December 2016. IEEE. doi:10.1109/BigData.2016.7840707.

Wayne W. Zachary. An Information Flow Model for Conflict and Fission in Small Groups. *Journal of Anthropological Research*, 33(4):452–473, 1977. doi:10.1086/jar.33.4.3629752.

Reza Zafarani, Mohammad Ali Abbasi, and Huan Liu. *Social Media Mining: An Introduction*. Cambridge University Press, July 2014. doi:10.1017/CBO9781139088510.

ChengXiang Zhai and John Lafferty. A Study of Smoothing Methods for Language Models Applied to Information Retrieval. *ACM Transactions on Information Systems*, 22(2):179–214, April 2004. doi:10.1145/984321.984322.

ChengXiang Zhai, William W. Cohen, and John Lafferty. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2003)*, pages 10–17, Toronto, Canada, July 2003. ACM. doi:10.1145/860435.860440.

Jichang Zhao, Junjie Wu, and Ke Xu. Weak ties: Subtle role of information diffusion in online social networks. *Physical Review E*, 82(1):016105:1–016105:8, July 2010. doi:10.1103/PhysRevE.82.016105.

Xiaoxue Zhao, Weinan Zhang, and Jun Wang. Interactive Collaborative Filtering. In *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management (CIKM 2013)*, page 1411–1420, San Francisco, California, USA, October 2013. ACM. doi:10.1145/2505515.2505690.

Tao Zhou, Linyuan Lü, and Yi-Cheng Zhang. Predicting missing links via local information. *The European Physical Journal B*, 71(4):623–630, October 2009. doi:10.1140/epjb/e2009-00335-8.

Boyao Zhu and Yongxiang Xia. Link Prediction in Weighted Networks: A Weighted Mutual Information Model. *PLOS ONE*, 11(2):1–13, February 2016. doi:10.1371/journal.pone.0148265.

Index

Page numbers in **bold** denote definitions or principal references

V

W