USER LATENT BEHAVIOR MODELING IN AN INTERCONNECTED WORLD

Draft of April 23, 2020 at 12 : 55

BY

KANIKA NARANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2020

Urbana, Illinois

Doctoral Committee:

Associate Professor Hari Sundaram, Chair
Assistant Professor Alexander Schwing
Professor ChengXiang Zhai
Dr. Chris Brew

ABSTRACT

User behavior modeling has become an indispensable tool with the proliferation of socio-technical systems to provide a highly personalized experience to the users. These systems are used in sectors as diverse as education, health, law to e-commerce, and social media. However, most of the work in this field operates on discrete user actions and do not leverage the vast amount of user-generated content to develop a comprehensive understanding of user behavior. Also, more often than not, users get influenced by other users around them, either their friends, peers, or even users with similar demographics. Thus, it is crucial to take into account these influences while modeling the user's behavior.

In this dissertation, we present multiple works that overcome these limitations. Specifically, we first discuss our approach that leverages user-generated content (questions or answers) in Community Question Answering (CQA) platforms to model their latent behavior. In particular, we estimate the latent aspect-based reliability of users in the forum to infer the trustworthiness of their answers. We then present our approach to incorporate influence from user's friends on their purchasing behavior in recommender systems. Even with the prevalence of forums with established social structures, many platforms still either lack a social structure or have sparse social signals. Besides, the assumption of similar behavior in user-to-user connections can be limiting. Thus, next, we focus on the task of best answer selection in CQA forums with no explicit user social connections. We induce multiple relationships between users based on the similarity and contrast in their behavior in the platform. These induced connections enable information sharing between connected nodes and significantly improves the user behavior model. We further extend our previous work on either modeling of latent behavior or user-to-user influence to joint modeling latent user behavior with social influence. In particular, we jointly modeled the latent abusive behavior of users and its effect on the behavior of their online friends to improve the offensive language prediction task on Twitter.

Users' behavior tends to evolve with experience; hence, finding behavioral similarity or contrast based on aggregated behavior is not optimal. Thus, finally, we present our approach to identify users with similar patterns of behavioral evolution. We applied our approach to identify users with similar patterns of research interests evolution in academia and a similar change in usage patterns in StackExchange.

To my parents, for their love and support.

ACKNOWLEDGMENTS

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Artificial Intelligence (AI) is a branch of computer science research that deals with the development of machines with intelligence rivaling those of humans. In other words, machines that can perform tasks that generally require human intelligence, such as visual perception, decision-making, speech recognition, and more. Most of the current AI research is task-driven and has achieved or, in some cases, even surpassed human intelligence [1**?**]. This advancement led to the assimilation of AI into our life inadvertently, present in the form of *socio-technical* systems around us. These intelligent systems provide self-paced learning in the education sector [2], enable targeted marketing in e-commerce websites [3], facilitate personalized medicare unique to patient's body type, genetics, and lifestyle [4] to even predict repeat crime incidence for convicts requesting bail [5].

The success of these intelligent machines lies in the fact that they can understand and model complex human behavior effectively and use it to extend highly personalized solutions at scale. They characterize user behavior by the interactions or activities performed by the user on a specific platform. For instance, in an e-commerce platform, activities denote items purchased by the user. Similarly, in a Community Question Answering (CQA) forums like StackExchange or Reddit, these activities are defined as posting questions or answers or voting on other people's answers. Artificial Intelligence, or specifically user behavior modeling, sifts through vast amounts of past user data to find recurring patterns and predict user's future purchases, search intent, or information need.

The problem of accurately modeling user behavior is far from being solved as there are still many challenges abound. These challenges primarily arise because humans are imperfect sensors of information. They do not necessarily conform to repetitive patterns and can be unpredictable. The user's behavior also tends to evolve. Users are biased as they tend to get influenced by other users, their environment, or even hold personal biases. With that being said, the current scenario also affords many more opportunities that were not present before. First, due to the close intertwining of the technology with our lifestyle, we have abundant user interaction data available to us now, more than ever before. These vast reserves of longitudinal data can help the machines to learn the nuances in the user behavior with more traces of behavioral change over time. Second, they are further aided by the concurrent improvements in computing power and computational techniques, such as deep neural networks, that can learn higher-degree polynomial functions needed to model and understand complex user behavior.

The philosophy of achieving at par human intelligence through AI is, in essence, to first

understand how humans process and extract knowledge from their environment and then emulate that in machines. Thus, to build a comprehensive user behavior model, we need to both *understand* and use the learned insights to efficiently *model* the activities of users in the platform. Deep Neural Networks have beaten human benchmarks in many language understanding and perception tasks [], but they are notorious for being uninterpretable. It thus creates a dichotomy between creating simpler models that offer a more in-depth understanding of the behavior versus using the advanced computational techniques to capture behavioral uncertainties accurately.

1.1 USER BEHAVIOR MODELING

The overall aim of this thesis is to achieve both *understanding* and an effective *modeling* of user behavior. However, it is not easy to build *interpretable* models that aid in understanding users that are *sophisticated* enough to capture behavior nuances precisely. Thus, we propose to view the problem of user behavior modeling from multiple angles. Each perspective will lead us to a different class of solutions; all of them bringing us closer to the overall goal of an improved understanding of user behavior.

1.1.1 Understanding user behavior

Understanding the human mind or, consequently, their behavior has been a matter of interest of researchers for decades. This interest has given rise to a new branch of science, psychology, that is devoted to the study of human behavior and mind. Research in this discipline is carried either through randomized or controlled experiments in the laboratory or through questionnaires and field-based data collection. The artificial settings of the laboratory or potentially biased responses from the users often pose a question to the validity of these studies' observations.

An abundance of user activity data online presents tremendous opportunities to analyze user behavior unhindered in the real world. Besides, these data often spans thousands or millions of users; a scale never achievable in field experiments. This opportunity led to the emergence of an interdisciplinary field known as *computational social science* that brought social scientists and computer scientists together. Researchers in this field use computational techniques to assess the validity of previous social science theories in online platforms at scale. Their research often unearths alternative findings about users, bringing a renewed understanding of user behavior.

The first and foremost perspective, thus, draws from the field of computational social science. This line of research aims to use advances in computational techniques to process vast amounts of user data and extract meaningful and comprehensible patterns of user behavior. These models primarily aiming at providing an in-depth understanding of user behavior fall into the category of *interpretable* models. This interpretability often comes at the expense of model precision.

These models can also provide an excellent framework to perform additional hypothesis testing of correlation of user's behavior with other covariates, that can be possibly predictive of their behavior. For instance, we can empirically test hypotheses like do changes in the posting pattern of a StackExchange user affect the upvotes their answers get? Furthermore, does that subsequently affect their activity level in the platform? These findings can be beneficial for moderators of the online communities to devise incentivization strategies for active users in order to retain them in the platform. Another exciting hypothesis can be the correlation of change in the publication behavior of scholars with the amount of research grants awarded to them. These findings can be particularly attractive to grant-awarding institutions to ascertain any potential biases or merits in their current grant-awarding scheme.

Users extensively interact with online platforms and produce massive amounts of multimodal user interaction data, such as text, video, speech, etc. User-generated textual data is the most popular form of interaction among them. Textual data is prevalent on multiple platforms in the form of reviews in e-commerce websites, questions, or answers text on CQA forums, tweets, or posts on social media platforms like Twitter or Facebook.

Textual data is more complex and sophisticated to comprehend than discrete activity features. Nevertheless, analyzing textual data opens the door to *understanding* the extensive and latent characteristics of user behavior that were not even possible to comprehend with activity features. For instance, the content of user reviews can be used to learn user affinity to different aspects of the product, such as relative importance of different aspects-food, service, location of a restaurant for a particular user. Similarly, the content of the user's answers or questions in a CQA forum can be used to discern latent features like the user's preferable topics to answer or their expertise for different topics. Prior works have leveraged the content of user's tweets or posts to understand user's political leanings [6], state of their mental health [7], and much more. Thus, it is imperative to leverage user-generated content to create a comprehensive understanding of user behavior.

Finally, works following this line of research should build interpretable models using extensive user data to provide an in-depth understanding of user behavior at scale.

1.1.2 Improving user behavioral models

The technological advancement in precise modeling of user behavior has afforded us with the seamless integration of AI into our lives. These intelligent systems provide *personalized* solutions at scale in sectors as diverse as e-commerce to education to medicine. Another complementary usage of creating powerful behavioral models is the ability to identify deviant users or even credible users in the platform. This outcome is highly desirable in the current circumstances, with the rise in illegitimate use of technology.

The second perspective, thus, mainly deals with pushing the performance boundaries of the state-of-the-art user behavioral models. Specifically, the solutions following this line of research strive to capture a comprehensive picture of user behavior by factoring in the myriad explicit and implicit influences on users. These methods can draw from a large body of social science research about user behavior in real-life settings. Note that the first perspective also deals with evaluating these theories in the online data traces with the primary aim of interpretability. As noted earlier, interpretability often occurs at the cost of model precision. On the contrary, the primary aim of this line of research is to build sophisticated behavioral models that can predict future user behavior accurately. In fact, the second perspective follows from the first one as it can leverage an improved understanding of user behavior online to build precise behavioral models.

The works following this line of research need to tackle multiple challenges posed to accurate behavioral modeling. For instance, a user's behavior tends to evolve with experience. Similarly, a user's friends, peers, or in general, other users with a similar background (demography, preferences, etc.) often influence their behavior. It is now possible to computationally model these effects due to the availability of extensive user interaction data on the platform. For instance, long-term user data in the platform provides the opportunity to model patterns of change in user behavior with time. Similarly, user-to-user influences manifest in many of the current online platforms due to their prevalent social structures. Connected users, i.e., users with established trust or friend relationships on these platforms, are empirically shown to exhibit similar behavior online, a phenomenon popularly known as user homophily [8].

Further, users hold an unconscious bias towards users with a similar background; a phenomenon also reverberated online. For instance, a user may trust movie recommendations of another user with similar demographics (same age or gender). Similarly, an Indian user may trust the ratings of another Indian user more than a non-Indian user when evaluating an Indian restaurant. Thus, it is crucial to capture these implicit influences between users who are alike based on general notions of similarity, such as demography or activity in the platform. Capturing the implicit influence is more complicated than explicit influence, but

it can provide vital cues for predicting the behavior of users with *few* social connections. They can also be particularly helpful in online platforms with no established social structure such as review platforms or CQA forums.

Thus, the creation of models that capture the explicit and implicit influences on users efficiently and at scale is prudent to bring advancement in the field of user behavior modeling.

1.1.3 Leveraging user behavior to improve task performance

Most of the current research related to user behavior modeling intends to model or predict user behavior primarily. However, there are related tasks pertinent to the estimation of characteristics of the user-generated content in the online platforms. Some of the examples of such related tasks are estimating the quality, credibility, or profanity of the content online.

Current work solving these tasks merely exploits the data features and completely ignores the user information. Users are the creators of the content, and their behavior remains broadly consistent across the platform. Thus, adding contextual information about user behavior estimated from their actions in the platform can immensely improve the prediction task. For instance, current models proposed for prediction tasks like best answer selection on CQA forums or hate speech prediction utilize the semantic meaning of the text to make such predictions. However, user expertise estimated through their prior answers on the platform can be used to differentiate between users. This differentiation can be consequently used to rank user-provided answers based on their trustworthiness. Similarly, the abusive behavior of users estimated from their prior content can provide a useful precedent when predicting the offensive nature of their new content. Furthermore, applying user homophily, behavioral priors of users can be shared to explicitly and implicitly similar users.

Thus, in this perspective, we propose to build models that leverage information about commonalities and disparities in user behavior to improve prediction tasks about user-generated content.

1.2 THIS THESIS

We outlined three different perspectives to solve the problem of user behavior modeling. These different perspectives are in no means comprehensive, but they pave a viable way to ultimately develop models than can attain the twin goal of interpretability and accurate modeling of user behavior. Since the field of behavioral modeling is so massive, there are numerous unsolved challenges within each perspective. In this dissertation, we propose a few foundational works under each perspective that provide potential approaches to solve

these posed challenges. Specifically, we attempt to answer the dichotomy of understanding versus modeling user behavior by first building interpretable models that primarily aim to provide detailed information about user behavior. Further, we develop frameworks to model user behavior or leverage information about user behavior to improve prediction tasks.

1.2.1 Understanding User Behavior

Under this perspective, we propose two works, the first one that directly models user activity data to understand patterns of behavioral change and another that leverages user-generated content to understand the latent characteristics of user behavior.

Firstly, in Chapter 3, we leverage user activity data to understand the behavioral evolution of individuals with experience. We introduce an interpretable Gaussian Hidden Markov Model (G-HMM) cluster model to identify archetypes of evolutionary patterns among users. Specifically, we apply our model to discover archetypical patterns of research interests' evolution among Academics and change in activity distribution of users of Stack Exchange communities. Our model allows us to correlate user behavior with external variables such as gender, income, etc.

In Chapter 4, we leverage the content of the user's answers in Community Question Answering (CQA) forums to learn latent characteristics of user behavior–*user latent reliability*. We use this latent behavior to solve the task of ranking answers of a given question based on its trustworthiness. This ranking is especially important as CQA forums are crippled with rampant unreliable content on their platform due to almost no regulations on post requirements or user background. This misinformation severely limits the forum's usefulness to its users.

We propose an unsupervised framework to learn the latent characteristic of user behavior–reliability and latent characteristic of answers–trustworthiness in a mutually reinforcing manner. In particular, our model learns a user representation vector capturing her reliability over fine-grained topics discussed in the forum. Besides, we also learn the semantic meaning of comments and posts through text representations or word embeddings. The learned latent representations using text affords an in-depth understanding about user reliability, improbable to comprehend using discrete activity data.

1.2.2 Improving User Behavioral Models

There are multiple unsolved challenges for accurate modeling of user behavior online. In this thesis, we focus on capturing the user-to-user influence to improve user behavioral

models. These influences can be either explicit in terms of social connections present in the platform itself or implicit in the absence or sparsity of established social connections. We propose to capture the implicit social influence, measured either through similarity or contrast in users' behaviors, by inducing connections between them. These connections enable information sharing among connected users resulting in an improved model of their behavior. To the best of our knowledge, no work models these regularities between user behavior explicitly through *induced* connections.

In this dissertation, we use *Graph Convolution Networks* [9] to model both explicit social connections and induced connections between users. Graph Convolution Networks (GCNs) is a recent class of neural networks that learns node representations in graph-structured data. Specifically, the model aggregates representations of the node itself, along with its neighbors, to compute a node representation. The model is very efficient with parallel batch processing and sparse computations. Thus, it can scale to large scale user graphs present on online platforms.

Specifically, in Chapter 5, we propose to incorporate the effect of *user-to-user influence* on the user's behavior in a recommender system. Recommender systems are a perfect example of a platform where user's preferences exhibit strong homophily with their friends on the platform. In this work, we exploit homophily in both user and item space. In the user space, apart from a user's explicit social connections in the platform, we also induce connections between users with similar purchasing history. In the item space, we construct a 'social graph of items' based on similarity in item features and co-occurrence in the dataset. These implicit similarity connections between items help the model to handle data sparsity in items (long-tail items, i.e., items with limited training data).

We propose a novel graph convolution-based aggregation models to estimate social influence in both user social and item similarity graphs. Besides, we also learn explicit attention weights for each pair of connected nodes to capture varying influence strengths on the behavior. We finally propose an interpretable aggregation strategy to combine the different factors influencing user preferences.

1.2.3 Improving task performance by leveraging user behavior

Under this perspective, we leverage user behavior information to aid in two diverse prediction tasks related to user-generated content. In addition, we propose different techniques to include user behavioral information for each task. The first approach models the similarities and contrast in user behavior through inducing connections amongst user-generated content. These induced connections aid in information sharing. We then propose a boosting

based ensemble approach to merge diverse feature sets. The second technique, on the other hand, learns powerful user representations encapsulating users' behavior. We subsequently use these representations in addition to the textual features to improve the prediction task.

First, in Chapter 6, we induce connections based on both similarity and contrast between users' behavior (answers) to improve the answer selection task in CQA forums. We induced a contrastive graph between user-provided answers replying to the same question and a similarity graph between answers across different questions if the replying users are exhibiting similar behavior. We specifically propose a modification to the original GCN to encode the notion of contrast between a node and its neighborhood. We also use state-of-the-art text representation learning approaches to compute representation for the user's answers and questions. We subsequently induce connections between user-generated answers based on these latent representations. Finally, multiple graphs expressing semantically diverse relationships are merged through an efficient boosting architecture to predict the best answer.

Thereafter, in Chapter 7, we work on leveraging textual features along with user features to detect the offensive language in tweets. Abusive behavior is rampant online and is affecting the experience of a large number of users on the platform. Hate attacks are often expressed in a sophisticated manner in the text (long clauses or complex scoping); thus, traditional sequential neural models are unable to capture it effectively. In this work, we learn an improved text representation of the tweets by leveraging syntactic dependencies between words. We achieve this by inducing a graph on the words of a tweet where edges represent a dependency relationship. We use these representations subsequently to estimate a user's latent abusive behavior, i.e., their likelihood of using offensive language online. Further, to capture homophily in abusive user accounts, we propagate this latent behavior through the user's social graph on Twitter using GCN. This user behavior information, in addition to the improved text representation of the tweet, dramatically improves the performance of offensive language detection models.

CHAPTER 2: LITERATURE REVIEW

Before delving into details of our proposed approach, we first discuss prior literature related to User Behavior Modeling in Online Social Networks, Academic Dataset and Recommender Systems. We also review text representation approaches used in Community Question Answering (CQA) forums and for short text in Twitter. We also briefly review recently proposed Graph convolution networks to model graph-structured data (used in our work) for these platforms.

2.1 ONLINE SOCIAL NETWORKS

There has been a lot of interest in the past on identifying and characterizing user behavior in online social networks (OSNs). Maia et al. [10] identified five distinct user behaviors of YouTube users based on their individual and social attributes. While Mamykina et al. [11] identified user roles based on just answer frequency in StackExchange. Adamic et al. [12] and Furtado et al. [13] worked on similar user behavioral studies on Yahoo Answers and Stack Overflow datasets, respectively. All these studies, however, ignore *temporal changes* in the behavior and use engineered features for behavior modeling.

Some behavioral studies do model evolution of user activities in the platform too. Benevenuto et al. [14] learned a Markov model to examine transition behavior of users between different activities in Orkut in a static snapshot. Yang et al. [15] and Knab et al. [16] proposed generative models that assigned each user action to a progression stage and classify event sequences simultaneously. They used their model to predict cancer symptoms, or products user would review in the future. However, the model did little to provide meaningful and interpretable stages and clusters. Angeletou et al. [17] constructed handcrafted rules to identify user roles and studied the change of user roles' composition in the community over time. Recently, Santos et al. [18] identified four distinct types of user activity pattern based on their activity frequency.

The Hidden Markov Model (HMM) has been widely used to model and cluster time sequences [19, 20, 21] in the past. However, most of these models learn an HMM for each user sequence and then employ clustering algorithms to cluster the learned HMMs. These approaches are not scalable, and the clusters thus identified are not interpretable.

2.2 RECOMMENDER SYSTEMS

In the following section, we provide a brief review of approaches that model users' historical interactions to improve recommender systems. We first enlist approaches assuming static user behavior (Collaborative Filtering). Consequently, we review approaches that model the evolution of user behavior (Temporal Recommendation), social influence (Social Recommendation), and few recently proposed methods which are looking at combining the two (Socio-Temporal Recommendation).

Collaborative Filtering: Collaborative Filtering (CF) is one of the most popular techniques for user modeling in recommender systems. Specifically, the methods employ Matrix Factorization (MF) to decompose a user-item rating matrix into user and item specific latent factors. Classical and seminal work for MF-based recommender systems [22] uses a Bayesian pairwise loss (BPR). Collaborative filtering is also performed in item space [23], where similar items are computed offline based on their rating similarity or co-occurrence in the dataset. Consequently, it recommends items similar to the ones used in the past by the user. Neural net approaches have been proposed recently to improve MF models. They learn more complex non-linearities in the user-item interaction data [24, 25].

However, most MF approaches assume a static user-item interaction matrix. Often, this assumption is not accurate, particularly for online communities where user preferences evolve over time — sometimes quickly — necessitating temporal recommendation.

Temporal Recommendation: There has been significant work in the area of temporal recommender systems that model a user's past interactions to inform a user's current preference. These temporal models generally assume a linear relationship between the events and model it using a Markov chain [26, 27]. However, these are often 'shallow' (i.e., linear) methods that are inept at modeling the more complex dynamics of temporal changes. Recent works [28, 29, 30] use deep net approaches involving convolution layers, attention networks, and recurrent neural nets to model complex relations. For example, Tang and Wang [31] applies convolutional filters on the embedding matrix computed from a few recent items of a user. This model captures a higher-order Markov chain, but it still has a limited scope as it does not consider the entire history of a user. In contrast, to model long term dependencies, Jannach and Ludewig [32] propose to model a user's sequential behavior within a session using recurrent neural nets. Wu et al. [33] apply a recurrent architecture to both user and item sequences and hence model dynamic influences in popularity of movies on users' viewing preference. Kang and McAuley [30] instead employ a self attention module for next item recommendation that adaptively learns the importance of all past items in a user's history. However, these models are limited as they do not leverage the social connections of

a user.

Social Recommendation: Social recommenders integrate information from a user's social connections to mitigate data sparsity for cold-start users, i.e., users with no or minimal history. They exploit the principle of social influence theory [8], which states that socially connected users exert influence on each other's behavior, leading to a homophily effect: similar preferences towards items. Jamali and Ester [34], Ma et al. [35] use social regularization in matrix factorization models to constrain socially connected users to have similar preferences. The recently proposed SERec [36] embeds items seen by the user's social neighbors as a prior in an matrix factorization model. The SBPR model [37] extends the pair-wise BPR model to incorporate social signals so that users assign higher ratings to items preferred by their friends. However, these models assume equal influence among all social neighbors. TBPR [38] distinguishes between strong and weak ties only when computing social influence strength.

Socio-Temporal Recommendation: Few of the recent approaches have started to look at merging temporal dependence with social influence. Cai et al. [29] extend Markov chain based temporal recommenders [27] by incorporating information about the last interacted item of a user's friends. This work assumes markov dependence i.e. the future item just depends on the current item. This assumption is limiting im modeling evolving user preferences.

In the context of session-based recommendation, Sun et al. [28] propose a socially aware recurrent neural network that uses a dynamic attention network to capture social influence. On the other hand, Song et al. [39] use graph attention nets to model social influence on a user's behavior in the session. Both these models learn a unified user representation based on social influence with a user's temporal history.

2.3 SCHOLARLY DATA

Most of the work on user behavioral mining concerns career movement within academia. Deville et al. [40] observed that transitions between academic institutions are influenced by career stage and geographical proximity. While Clauset et al. [41] found that academic prestige correlates with higher productivity and better faculty placement. Recently, Safavi et al. [42] studied career transitions across academia, government, and industry for Computer Science researchers. Wang et al. [43] proposed a statistical model to predict the most impactful paper, in terms of citations, of scientists across disciplines. They argued nonexistence of a universal pattern and showed that highest-impact work in a scientist's career is randomly distributed within her body of work.

Recent studies also looked at gender differences in funding patterns, productivity, and

collaboration trends in academia [44, 45]. Way et al. [44] did not observe any significant difference across gender in hiring outcomes in academia. However, they showed that indirect gender differences exist in terms of productivity, postdoctoral training rates, and in career growth. Some earlier studies also reported gender differences in academia. Kahn [46] identified gendered barriers in obtaining tenure for academics in economics, while Ward [47] found gendered differences in pay related to publication record.

There also has been considerable interest in mining scholarly data produced by researchers (bibliographic data, researchers' usage of social media, etc.). Prior studies have looked at the evolution of research interests on a community level. Liu et al. [48] studied the evolution of research themes in articles published in CHI conference on Human Computer Interaction through co-word analysis. They highlighted specific topics as popular, core, or backbone research topics within the community. While Biryukov and Dong [49] compared different scientific communities in DBLP dataset in terms of its interdisciplinary nature, publication rates, and collaboration trends. They also studied the variation of author's productivity with career length and observed that most of the authors have a short career spanning less than five years. Chakraborty and Nandi [50] studied trajectories of successful papers in computer science and physics by analyzing paper citation counts. They classified these trajectories into multiple categories including early riser, a late riser, steady riser, and steady dropper.

2.4 COMMUNITY QUESTION ANSWERING FORUMS

Community Question Answering forums are increasingly used to seek advice online; however, they often contain conflicting and unreliable information. This misinformation could lead to serious consequences to the users. Thus, most of the work that model user behavior in CQA forums deals with predicting user reliability or quality of posted answers to a question.

Prior works can be classified into Feature-driven models; which use user and content-based engineered features for the task; another is Deep Text models that only model relevance of the content of question and answers for prediction and disregard user information. Recently, unsupervised approaches based on Truth Discovery principle are applied to model user expertise and answer quality simultaneously in these forums.

Feature-Driven Model: Feature-driven models [51] develop features from three different perspectives: user features, content features, and thread features. These features are fed into classifiers, such as tree-based models [51, 52, 53] to identify the best answer. Tian et al. [53] found that the best answer is usually the earlier and most different one, and tends to have more details and comments. Jenders et al. [52] trained several classifiers for online

MOOC forums. Different from existing works, Burel et al. [51] emphasize on the thread-like structure of question & answer and introduce four thread-based normalization methods. These models predict the answer label independently of the other answers for the question. CQARank leverages voting information as well as user history and estimates user interests and expertise on different topics [54]. Barrón-Cedeno et al. [55] also look at the relationship between the answers, measuring textual and structural similarities between them to classify useful and relevant answers. All these supervised approaches need a large amount of labeled training data [56, 57, 58]. However, it is expensive and unsustainable to curate each answer manually for training these models. Alternatively, forums employ crowd sourced voting mechanisms to estimate information reliability but it could lead to under-provision [59].

Deep Text Models: Text-based deep learning models learn an optimal representation of question-answer text pairs suitable to select the best answer [60, 61, 62]. In SemEval 2017 on Community Question Answering (CQA), [63] developed a task to recommend useful related answers to a new question in the forum. SemEval 2019 further extends this line of work by proposing fact checking in community question answering [64]. Feng et al. [65] augment CNN with discontinuous convolution for a better vector representation; Wang and Nyberg [62] uses a stacked biLSTM to match question and answer semantics. Sukhbaatar et al. [66] use attention mechanism in an end-to-end memory framework. Text-based models take longer to train and are computationally expensive.

Truth discovery: Different approaches based on truth discovery principle have been proposed to address predict answer quality in CQA forums [67, 68, 69, 70, 71, 72]. Many truth discovery approaches are tailored to categorical data and thus assume there is a single objective truth that can be derived from the claims of different sources [73]. Faitcrowd [74] assumes an objective truth in the answer set and uses a probabilistic generative model to perform fine-grained truth discovery. It jointly models the generation of questions and answers to estimate the source reliability and correct answer. On the other hand, Wan et al. [75] propose trustworthy *opinion* discovery where the true value of an entity is modeled as a random variable with a probability density function instead of a single value.

Some truth discovery approaches also leverage text data to identify correct responses better. Li et al. [76] proposed a model for capturing semantic meanings of crowd provided diagnosis in a Chinese medical forum. In particular, they use a medical-related dictionary to extract terms in the response text and learn their semantic representations to discover trustworthy answers from non-expert users in crowdsourced diagnosis. Zhang et al. [67] proposed a Bayesian approach to capture the multifactorial property of text answers and used semantic representations of keywords to mitigate the diversity of words in answers. To model the user reliability, the authors proposed a two-fold reliability metric that uses both

false positive and true positive rates. These approaches only use certain keywords for each answer and are thus, limited in their scope.

2.5 TWITTER

Most previous methods for detecting offensive speech on Twitter rely entirely on the textual content. Most of these prior work includes using statistical features like bag-of-words or tf-idf features for automated detection.Wulczyn et al. [77] used character n-gram features for detecting abusive comments in the discussion on Wikipedia pages. On Twitter dataset, Waseem and Hovy [78] used character and word n-gram features along with lexical and users features to detect hate speech. Davidson et al. [79] worked with character n-grams on a different Twitter dataset to achieve competitive performance. On the other hand, Nobata et al. [80] combined n-grams features with linguistic, syntactic, and semantic features. However, they observed that n-gram features are most beneficial for the detection task. Even though bag-of-words approaches perform well, they are unable to capture nuanced hate speech as they fail to contextualize the word meanings. For instance, depending on the context, the word *gay* can be used to denote either ebullience or sexual preference. Only the latter is a candidate attack.

Recently, deep learning models are also proposed that leverage pre-trained word embeddings such as word2vec [81] and Glove [82] to capture aspects of the semantics of the tweets. These models aggregate individual word embeddings in a context-aware manner to compute tweet embeddings and later use them for classification. Gambäck and Sikdar [83] and Park and Fung [84] used the Convolutional Neural network to compute the tweet embeddings while Badjatiya et al. [85] and Agrawal and Awekar [86] showed that Gated Recurrent Units or Long-Short Term Memory networks are useful to compute these embeddings. On the other hand, Zhang et al. [87] used a combination of CNNs and GRU to achieve competitive performance.

The syntactic structure of the text can also be used to help identify the target group and the intensity of hate speech. For instance, Warner and Hirschberg [88] extracts POS-based trigrams such as DT jewish NN to extract hate speech against a specific target, Jews. While, Silva et al. [89] extends it further to look for generic syntactic structures like "I <intensity> hate <target>'. The primary difficulty of this work is that the space of possibly relevant rules is too large for an analyst to be confident that the list is truly comprehensive. In addition, it verges on the impossible to specify a set of rules that will do a decent job on the endless variety of possible implicit attacks.

A minority of approaches take advantage of non-textual user data in addition to the text.

Pavlopoulos et al. [90] added randomly-initialized user embeddings to their RNN model to obtain higher accuracy. Qian et al. [91] showed that incorporating intra-user and reinforced inter-user representations significantly improve the performance of their bi-directional LSTM model. However, both of these approaches work on the individual user level and ignore the social influence on their behavior. Mishra et al. [92] captured the social influence in abusive accounts by computing a representation of a user's neighborhood through node2vec features. The classifier described in Mishra et al. [93] extends the previous paper by computing a user representation from an extended graph of users and tweets.

2.6 GRAPH CONVOLUTION NETWORKS:

More recently, Graph Convolution Networks (GCNs) have been proposed to learn embeddings for graph-structured data [94]. Graph Convolution can be applied in both spatial and spectral domains to compute node representations. The learned node representations are then used for various downstream tasks like node classification [9], link prediction [95], multi-relational tasks [96] etc. Spatial approaches employ random walks or k-hop neighborhoods to compute node representations [97, 98, 99, 100]. Pioneer works on graph convolution in the spectral domain use fast localized convolutions [101, 102]. Recently proposed Graph Convolution Networks [9] outperforms spatial convolutions and are scalable to large graphs. Various extensions to the GCN model have been proposed for signed networks [103], inductive settings [104] and multiple relations [105, 95] and evolution [106]. All of the GCN variants assume label sharing as they only assume similarity between connected nodes.

In Recommender Systems, GCNs have been used to model the user-item interaction graph. GCMC [107] extends GCN by training an auto-encoder framework on a bipartite user-item interaction graph that performs differentiable message passing, aggregating data from a user's and an item's 'neighbors'. PinSage [108] proposed a random walk based sampling of neighbors to scale GCNs to web scale graphs. Fan et al. [109] further extend these methods to incorporate information from a user's social connections. Similarly, Wu et al. [110] use graph neural networks to model diffusion of social influence in recommender systems.

However, these methods either do not take a user's social neighbors into account or operate on static features. All these models also assign uniform weight to all their neighbors, which does not represent online social communities well. Typically in these communities, some friends are only superficially known while others are known personally for years. Thus, they exert a different degree of influence on a user's behavior. Graph Attention Networks [111] can capture the varying influence stengths as they learn attention weights between each pair of nodes in a static graph.

CHAPTER 3: IDENTIFYING USERS WITH SIMILAR BEHAVIOR EVOLUTION

Finally, we propose to identify similar users based on similar patterns of user behavior evolution in social platforms. Users behavior evolve with time and thus, it is not optimal to identify similar uses based on aggregated behavior over time. In this chapter, we propose an interpretable model that clusters users with similar archetypical patterns of user behavior evolution. We evaluate our approach to an interesting domain of patterns of change in research interests of scientists with experience. In addition, we evaluate our model for activity distribution evolution of users in StackExchange communities to showcase the generality of our framework [112].

3.1 OVERVIEW

In this chapter, we develop models to understand how individuals evolve with experience in social networks. The problem is important: as individuals interact with each other, they gain in experience, and behavioral changes reflect the newfound experience. However, despite a significant focus on community discovery and their evolution in social networks, our understanding of individual evolution is limited (Yang and Leskovec [113], McAuley and Leskovec [114] are some notable exceptions). Understanding evolutionary patterns, in general, is useful in a variety of applications: language evolution [115]; expertise evolution [114]; journey optimization in digital advertising platforms.

Our specific interest lies in understanding how academics change their research behavior with gain in research experience. In the academic community, authors' research interests are influenced by other authors' directly (collaboration) or indirectly (related published research) and in general, by the current research trends in the community. Analyzing the evolution of academic behavior on the community level has attracted persistent interest; previous works studied the evolution of research themes for a particular scientific community [116] or multiple communities [49, 117]. On an individual level, evolutionary studies have looked at modeling career transitions [42], citation evolution [43] and productivity or collaboration trends [45]. On the other hand, in this work, we want to identify dominant patterns of *research interests* evolution common among academics across different subfields. Our work can help to answer questions like, Do academics focus on a single research area throughout their career or they venture in multiple areas as they gain experience? If they work on multiple areas, when does this shift usually happens? Are some evolutionary patterns preferred over the others? This knowledge can assist in providing better career guidance to junior faculty

on how to structure their career. Moreover, it can help funding agencies identify researchers of particular evolutionary pattern that may need more assistance.

At the outset, discovering patterns of individual evolution appears to be a combinatorial problem: academics vary in not only the sub-field that they choose to start but also in subsequent areas of interest. Furthermore, their research interests may evolve at different rates. Despite variations in the chosen sub-field of an academic, and how academics can evolve, we observe regularities at different stages of their career. For instance, for an academic, transition through different stages— Ph.D. Student (focusing on a single research area), being an assistant professor (working on few highly related areas) to eventually post-tenure (multiple areas, interests in multidisciplinary collaborations, etc.)—mark changes in research behavior. These elementary behavioral evolutionary patterns are visible in almost all academic fields, suggesting that surface variations (i.e., area of research for an academic) hide deeper regularities in patterns of behavioral change. We refer to these *latent* regularities in individual behavior as *behavioral stages*. We refer to the dominant progression patterns through behavioral stages as *archetypes*. Note that researchers may evolve at different rates through these stages. We show that we can explain all individuals' surface variations (the observed research area on which the academic focuses) with a small set of such archetypes. Figure 7.5b shows a stylized example.

Thus a model for learning archetypes needs to: express variation in observable research behavior while exhibiting latent stochastic regularities governing the change of behavior. Furthermore, the model should allow individuals to evolve at different rates. Finally, the results ought to be interpretable in a post-hoc manner.
Our work makes the following contributions:

A framework for modeling evolutionary trajectories: We propose a sophisticated framework to identify dominant, interpretable, evolutionary archetypes amongst academics for modeling the evolution of their research interests. In contrast, prior work on academics has either focused on the qualitative analysis (e.g., [47]) or predicting career transitions [42]. In our work, we assume that an archetype is a *probabilistic model* that encodes individual progression through stages of *distinct* behavior. Specifically, we learn a Gaussian Hidden Markov Model (G-HMM) to capture this progression where latent states capture *behavioral stages* in the evolution. To encode the idea of experience, while we allow individuals to evolve into the higher stages, we constrain our model to prevent individuals from returning to a stage from which they have evolved. We model *all* individuals with a *small* set of archetypes. We jointly learn the mapping of users into their archetype and the archetype's associated model's parameters

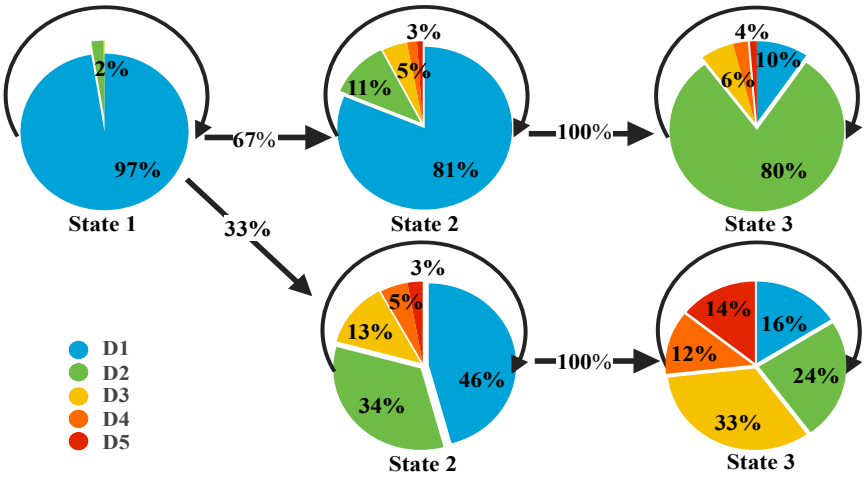


Figure 3.1: A stylized academic evolutionary trajectory. Each pie chart is a *behavior stage* in the trajectory. The numbers in each pie-chart show the fraction of chapters published in each research area D_m in that stage. We use a normalized representation focused on the change of areas: the label D_1 represents the first research area of every academic, D_2 the second research area, etc. Normalized representations allow us to discover commonalities in behavioral changes of academics across seemingly unconnected domains. In this example, the top group of researchers evolves to shift their research focus to a new domain while the bottom group becomes increasingly interdisciplinary.

through an Expectation-Maximization framework.

Finding: Dominant archetypes: While our framework is generic, we apply our model to understand the evolution of the research interests of Computer Scientists. We identify four archetypes with almost equal distribution of academics: (i) *Steady* researchers who primarily work in their first research area throughout their career (most popular); (ii) *Evolving* researchers, who continuously shift their dominant area of research; (iii) researchers with *Diverse* research interests; and (iv) researchers who have *Diffused* interests with infrequent contributions in multiple areas. Each archetype is significantly different ($p < .001$) from the others.

Finding: variation by gender within archetype: We examine empirically, a subset of our data—all full professors (as of Spring 2018) in the top 50 CS departments in the United States for gender differences in their academic trajectory. We observe similar gender distribution across archetypes with the least number of women professors in *Evolving* archetypes. However, within the same archetype, we observe significant differences in the models that explain the evolution of male and female researchers. For instance, the models that explain women and men differ ($p < .01$) in the *diverse*

archetype; we observe 30% men (8% women) tend to start from later stages while women skip more stages (50% women (36% men) skip stage 3; 14% (9 % men) women skip stage 2). Women also spend around a year more *exploring* mid-career than men (6.5 years for women vs 5.3 years for men) in the same stage.

Finding: variation in grant income: Next, we examine grant income (as of Spring 2018) from the National Science Foundation in the US for the same subset of CS academics, to understand the relationship between variations in awarded grant income over the course of academic trajectory and how difference in archetype or gender could serve as explanations. Although we did not observe any significant changes in the average grant income between archetypes, there exists income variability within stages of each archetype. In general, researchers are awarded more grant money as they gain experience, with the most notable uptick being after the first few years of their research career (between stages 2 and 3, $p < .001$) and in their last career stage (stage 5, $p < .05$). Specifically, researchers with *diverse* research interests receive subsequently increasing grant income while *evolving* researchers (who change their dominant research area in each stage) experience grant income variability with an area change. We find significant differences in grant income across genders *within a behavioral stage* of an archetype also mostly accompanied by an area shift. For the *steady* and *diverse* archetype, female professors are awarded lower grant income than their male counterparts ($p < .05$) in their early career stages. On the other hand, evolving women receive a significantly lower income than evolving men when they switch to new areas later in their career in state 4 and 5 ($p < .05$).

Also, we have strong quantitative results with competing baselines for behavior prediction and perplexity on the Academic dataset. We subsequently evaluate our model to multiple StackExchange communities to show generality of our work. The proposed G-HMM cluster model improves by 24% for Academic and on an average of 32% for Stack Exchange communities over the baselines for behavior prediction. Our model also exhibits lower perplexity than the baselines.

Significance: We propose a sophisticated probabilistic framework to identify dominant, interpretable, evolutionary archetypes. We show that the discovered archetypes are significantly different and are straightforward to use to test hypotheses (e.g., evolutionary variation with gender; effects of gender on income).

3.2 MODELING EVOLUTION TRAJECTORIES

In this section, we first introduce our datasets and then formally describe the problem statement followed by our proposed approach.

3.2.1 Data Collection

Academic dataset: We use the Microsoft Academic dataset [118] provided through their Knowledge Service API[1] to study evolutionary patterns of researchers with a focus on Computer Scientists. Microsoft Academic Service additionally annotates each publication with the year of publication, publication venue and the CS subfield (out of 35 identified fields) to which it belongs.

We can only query an individual author's publication history through the Microsoft Academic API, not the whole academic corpus [2]. Thus, we create an unbiased author list by identifying *prominent* scientists from each of the 35 CS subfields. This author data will help us discover the dominant archetypes of change in research interests of researchers, across different subfields. Also, *prominent* scientists usually have a long academic career to notice a change in research interests.

We identify *prominent* authors based on *prestige* of the conference venues in which they publish, in their respective subfield. We use the older dump of Microsoft Academic dataset[3] to identify *prestigious* conferences for each subfield. We construct a conference-conference citation graph where each conference in our dataset forms a node, and the weighted edges represent inter-conference citation frequency. Specifically, the weight of a directed edge from conference C_1 to conference C_2 is proportional to the fraction of papers published in C_2 cited by papers published in C_1. We then use the Pagerank algorithm [119] on this directed graph and define conference *prestige* as the Pagerank of the corresponding conference-node. After that, we define an author's *prominence* as the weighted sum of the prestige of the conferences (s)he has published in. Here, conference-prestige are further weighted by the fraction of the author's papers published in that venue.

We rank authors in decreasing order of their *prominence* in each of the 35 CS areas (as annotated by Microsoft API) in the dataset. To get equal representation from all subareas, we then extract the publication history of top 750 most-prominent authors from each of the

[1] `http://bit.ly/microsoft-data`

[2] There is an older dump of Microsoft Academic dataset `https://aminer.org/open-academic-graph` but it is noisy and contained multiple entries for the same authors; however, the online dataset is updated weekly, and API provides the most recent version.

[3] `https://aminer.org/open-academic-graph`

subareas in the dataset. Note that authors can be *prominent* in more than one subfield. We then filter unique authors from this set who have at least 15 years of publication history. This filtering is done to get a sufficient span of publication data to undergo evolution in research interests. Further, we restrict our analysis to papers published from 1970 to 2016 to avoid missing data. The resulting dataset consists of records of 4578 authors with an average publication history of 24.15 years [4].

Stack Exchange Dataset: Our second dataset consists of activity logs of users of Stack Exchange [5](as of Feb 2017), a popular online question-answering platform. In this paper, we work on 7 diverse communities of the platform: Stack Overflow, English, Money, Movies, CrossValidated, Travel and Law. These communities have varied sizes and cater to different audiences. For each user, the data contains details about their activities such as posting a question or answer on the community. Lastly, to focus on users who have spent enough time in the network to exhibit behavioral changes, we filter users with less than 10 sessions, and also remove outliers with more than 750 sessions. Table 3.1 shows the final data statistics.

Dataset	N	$\bar{t}$	$t_{\max}$	M
Academic	4578	24.15	47	6
StackOverflow	561937	47.13	750	5
English	3828	44.01	729	5
Money	873	44.41	706	5
Movies	678	48.40	598	5
CrossValidated	3728	38.94	738	5
Travel	1000	56.14	736	5
Law	195	47.79	584	5

Table 3.1: Dataset statistics for the Academic and Stack Exchange datasets. N: number of users; M: possible actions in each session; $t_{\max}$: maximum session length; $\bar{t}$: mean session length. For authors, $\bar{t}$ is their average career length (in years).

3.2.2 Problem Definition

We represent an author's academic life-cycle as a sequence, $\mathbf{X_i}$, comprising of session-vectors, $\vec{X}_{ij}$. We keep *session* as a year-long since most conferences occur annually. Thus, $\mathbf{X_i}$ is a sequence of session-vectors, $\vec{X}_{ij}$, where $j \in \{1, 2, \ldots t_i\}$ and t_i is the number of sessions for an author i. In general, lengths of sequences will vary across authors depending on the length of their academic career. A session, $\vec{X}_{ij}$, is a vector $\langle o_1, o_2, \ldots, o_M \rangle$, where M

[4]This data will be made available upon publication

[5]`https://data.stackexchange.com/`

denotes number of *area-of-interests* (`AoI`s). Each element o_m of the vector $\vec{X}_{ij}$, denotes the fraction of papers published in the D_m `AoI` by the i-th researcher during a single j-th year. This distribution of research areas of author's publications captures the research *behavior* of the individual in the year.

For defining an `AoI` of an author, we consider all papers published by the author in her academic life. We identify her primary `AoI`, D_1, as the *first* subfield (out of 35 subfields) in which she publishes *cumulatively* at least 3 papers in the first 3 years. Usually, an author's D_1 is about their Ph.D. dissertation work, and we expect students to *settle* down after a few years. Thus, after identification of D_1, hopefully with a steady paper count, we define her secondary `AoI`, D_2, as the subfield in which she publishes at least 3 papers in *one* year. Similarly, we also define tertiary (D_3), quaternary (D_4), and quinary (D_5) `AoI`. We do not define `AoI`s beyond D_5 because 80% of authors do not explore more than 5 subfields in our dataset. Also, in a given year, if an author publishes fewer than 3 papers in an unexplored subfield, these papers count towards a sixth dimension `AoI` called *Explore* (Ex). *Explore* dimension denotes that the author has started exploring new subfields but are not notable enough to be one of the D_m's ($m \in [1, 5]$), and indicate a possible shift in research interests.

To summarize, each session is a 6 dimensional vector ($M = 6$), and its elements are fraction of the author's publications in the 5 D_m's or the 6^{th} *Explore* dimension. This normalized session representation allows our model to discover behavioral patterns of the author's changing research interests in a domain-independent manner. For example, in a given year, the session-vector for an author who publishes 3 papers in theory (D_1; primary area) and 1 paper in graphics (D_2; secondary area), and the session-vector for another author who publishes 3 and 1 papers in NLP (D_1; primary area) and ML (D_2; secondary area) respectively will be exactly same: $X_{ij} = \langle 0.75, 0.25, 0, 0, 0, 0 \rangle$. Notice that normalization does not change the rate at which a specific author decides to switch domains and is also invariant to subarea publication norms ([44] observed productivity rates differ by subfield in DBLP).

Similar to Academic data, for StackExchange communities, we represent each user by a sequence, $\mathbf{X_i}$, of session vectors. We split the activity-sequence of a user into sessions using a time threshold similar to session definitions in web search [120]. Specifically, we create a new session if the difference between two consecutive activities is more than 6 hours. A gap longer than this marks a new visit to the community. Hence, a session is a subsequence of the user's activity-sequence and is formally represented as a distribution over the M possible activities; where its m^{th} element represents the fraction of total activity spent in the m^{th} activity in that session. Note that Stack Exchange allows $M = 5$ different activities : post a **Q**uestion; **A**nswer a question; **C**omment on a question or an answer; **E**dit operations like

assign tags, edit body or title of a post; and **M**oderator operations like voting.

The problem then addressed in this paper is to associate an *archetype* with each user's sequence. We assume that there exist C different archetypes, and given a sequence of session-vectors for an user $\vec{X}_i = \{\vec{X}_{i1}, \ldots \vec{X}_{it_i}\}$, the goal is to assign the sequence to one of the C *archetypes*—each associated with a set of K latent *behavioral stages*. During this assignment, we also identify how the individual evolves through its archetype's distinct stages by outputting the sequence $Y_i = \{Y_{i1}, Y_{i2} \ldots Y_{it_i}\}$, where Y_{ij} represents the behavioral stage $k \in [1, K]$ assigned to j-th session in individual i's sequence. We constrain the number of stages $K \ll t_i$ and allow skipping of stages while disallowing return to earlier stages.

3.2.3 A Framework for Identifying Archetypes

We use a Gaussian-Hidden Markov Model (G-HMM) based approach to model individual behavior. In our model, latent states of the G-HMM capture the *stochastic regularities* in behavior while Gaussian observations enable *variations* in the session-vector distributions (instead of fixed observations in vanilla HMM). Thus, a G-HMM captures an archetype with all individuals belonging to the archetype, going through the same set of *behavioral stages* or latent state. Note that G-HMM allows for skipping states and variable evolutionary rates among individuals. To capture broad variations amongst individuals, we learn a set of C G-HMMs where each G-HMM represents a distinct archetype. We jointly learn the partitioning of the individuals into different archetypes and the model parameters for each archetype.

Each Gaussian HMM, associated with an archetype c, has K discrete latent states or *behavioral stages*. The model makes a first-order Markovian assumption between state transitions using the transition probability matrix $\tau^{\mathbf{c}}$; where τ^c_{kl} represents the probability of transitioning from stage k to l in the c-th archetype. The prior probabilities of the latent states are represented by the K dimensional vector π^c. Lastly, the model assumes that given a latent behavioral stage, k, from an archetype c, the M dimensional session vector, X_{ij}, is Normally distributed with mean μ^c_k and covariance $\mathbf{\Sigma^c_k}$. The mean vector μ^c_k essentially encapsulates the typical behavior exhibited in the k-th *behavioral stage*.

In the above model, the G-HMM associated with different archetypes do not share latent states. In other words, each G-HMM has its own set of discrete latent states.[6] However, we fix the number of states (K) to be the same for each archetype.

Encoding Experience & Variable Evolutionary Rates: To encode the idea of experience, as well as to allow variable evolutionary rates, similar to [15] and [16], we allow only forward state transitions (including self-loop) within a G-HMM that represents

[6] Experiments with tied-states of archetypes led to worse results.

an archetype. This choice appears sensible to us since semantically, each latent state of the G-HMM represents a *behavioral stage* of evolution, and its corresponding mean vector encapsulates *behavior* in that stage. Then, forward transition captures *progression* through *behavioral stages*. We operationalize this idea by restricting the state transition matrix to be an upper triangular state transition matrix.

Algorithm 1 Gaussian HMM archetype

1: **Input:** $\vec{X_i}$ and $\lambda_{\mathbf{0}}^{\mathbf{c}}$ $\forall i \in \{1, 2, \ldots N\}$ $\forall c \in \{1, 2, \ldots C\}$
2: **Output:** $\vec{Y_i}$ and $\lambda^{\mathbf{c}}$ $\forall i \in \{1, 2, \ldots N\}$ $\forall c \in \{1, 2, \ldots C\}$
3: Initialize the c^{th} archetype with initial parameters, $\lambda_{\mathbf{0}}^{\mathbf{c}}$ $\forall c$
4: **while** not converged **do**
5: **M-Step:** Re-assign archetypes to sequences $\mathbf{X_i}$ as:
6: $c_i = argmax_c P(\mathbf{X_i}|\lambda^{\mathbf{c}})$ $\forall i \in \{1, 2, \ldots N\}$
7: **E-Step:** Re-estimate the G-HMM parameters, $\lambda^{\mathbf{c}} \forall c \in \{1, 2, \ldots C\}$, using modified Baum-Welch algorithm.
8: **end while**
9: **Convergence Criteria**

- Log Likelihood difference falls below threshold; or
- Number of iterations is greater than threshold; or
- Number of sequences re-assigned in an iteration is less than 1% of the data

Training: We train our G-HMM cluster model using a (hard) Expectation Maximization [121] based iterative procedure described in Algorithm 1. During training, the goal is to learn the G-HMM parameters, $\lambda^{\mathbf{c}}$, for each archetype c, where $\lambda^{\mathbf{c}} = \langle \mu^c, \mathbf{\Sigma^c}, \pi^c, \tau^c \rangle$ and archetype assignments for each user, c_i. We first initialize the Gaussian HMMs with initial parameters, $\lambda_{\mathbf{0}}^{\mathbf{1}}, \lambda_{\mathbf{0}}^{\mathbf{2}}, \ldots, \lambda_{\mathbf{0}}^{\mathbf{C}}$. After that, in the iterative training process, in the Expectation step, we use current estimates of $\lambda^{\mathbf{c}}$ to assign an archetype to each user sequence in the data. In the Maximization step, we use current archetype assignments to learn the corresponding G-HMM's parameters, $\lambda^{\mathbf{c}}$. We use a modified version of the Baum-Welch algorithm [122], allowing for forward-only transitions. Thus, this method jointly partitions the input sequences into different archetypes as well as learns the parameters of the associated G-HMMs.

Implementation Details: Our iterative training procedure requires initialization for G-HMM parameters, $\lambda_{\mathbf{0}}^{\mathbf{c}}$. We perform k-means clustering on all sessions of all user sequences in our corpus, treating the sessions as independent of each other (thus losing the sequential information). The cluster centers, thus obtained are used as the initial means, μ_0^c, for the latent states. We fix each Σ_k^c as an identical diagonal covariance matrix σI with $\sigma = 0.01$ based on preliminary experiments. We initialize transition matrices, τ_0^c, and states' prior probabilities, π_0^c, for each archetype randomly. Our implementation is based on Kevin

Murphy's HMM Matlab toolbox [7]. Also, we implement a parallelized version of our EM algorithm to reduce computation time. We test our model on Intel Xeon Processor with 128 Gb RAM and a clock speed of 2.5 GHz.

3.3 RESULT ANALYSIS

In this section, we perform analysis of archetypes identified by our model. We first describe the discovered archetypes of all researchers in Section 3.3.1. Then, we examine gender variation in academic trajectory in Section 6.7 and effect of archetype and gender on grant income in Section 3.3.3. Finally, we describe discovered archetypes for the largest Stack Exchange community, StackOverflow in Section 3.3.4.

3.3.1 Discovered Archetypes of Academics

Our analysis reveals four dominant archetypes: *Steady*, *Diverse*, *Evolving* and *Diffuse*. We chose the number of clusters $C = 4$ using the elbow method [123]: data log-likelihoods increased rapidly till four clusters with much slower increase beyond that. Further, we chose the number of states per cluster, $K = 5$: beyond five states, KL divergence[124] between mean vectors of new states with previous states started reducing rapidly, indicating redundant states.

We also conducted t-test to validate differences among the identified archetypes. Specifically, paired-sample t-test [125] is conducted between likelihood values of data points assigned to an archetype with their likelihood values obtained from rest of the archetypes. For instance, for each archetype pair (p, q), we conduct paired t-test between $\log P(X_i|\lambda^p)$ and $\log P(X_i|\lambda^q)$ $\forall i \ni c_i = p$. Note that test results for archetype pair (p, q) are not symmetric. We observed that all archetype pairs are significantly different ($p < .001$) from each other. Now, we proceed to discuss what is common to these discovered archetypes before examining each one in detail.

`Commonalities in Archetypes:` Table 3.2 summarizes the trajectories (state sequences) learned for the four different archetypes in this dataset. Each archetype is labeled according to our interpretation of the user behavior, looking at the learned mean vector of G-HMM states. We observe that all archetypes exhibit similarities, especially in the first two stages. Across all archetypes, the first *stage* typically spans around 3 years, and more than 72% of the published research is in the author's *primary* `AoI`: D_1. As noted before, this is most likely

[7] `bit.ly/hmmtoolbox`

Behavioral Stage	**Steady**	**Diverse**	**Evolving**	**Diffuse**
Stage **1**	{3Y, 5m} **D₁** (87%) Ex (11%)	{3Y, 3m} **D₁** (88%) Ex (11%)	{2Y, 9m} **D₁** (72%) Ex (24%)	{2Y, 7m} **D₁**(76%) Ex (22%)
Stage **2**	{4Y, 2m} **Ex** (74%) D_1 (23%)	{2Y, 6m } **Ex** (80%) D_1 (16%)	{2Y, 9m } **Ex** (83%) D_1 (12%)	{3Y, 7m } **Ex** (91%)
Stage **3**	{7Y, 5m} **D₁** (62%) **Ex** (32%)	{5Y, 6m} **D₁** (73%) Ex (17%)	{6Y, 2m} **D₁** (33%) **Ex** (28%) **D₂** (24%)	{8Y, 5m} **D₁** (50%) **Ex** (39%)
Stage **4**	{5Y, 9m} **D₂** (49%) **D₁** (27%) Ex (17%)	{5Y, 6m} **Ex** (46%) D_2 (20%) D_1 (17%)	{5Y} **D₂** (66%) *Ex* (18%)	{3Y, 9m} **D₂** (43%) **Ex** (26%)
Stage **5**	{2Y, 6m} **D₁** (49%) Ex (18%) D_2 (14%)	{6Y, 3m} **D₄** (29%) **Ex** (20%) D_3 (14%) D_1 (14%)	{6Y, 5m} **D₃** (43%) Ex (19%) D_2 (14%)	{4Y, 1m} **Ex** (74%)

Table 3.2: Learned mean vector for each latent state of four archetypes in the Academic Dataset. We list the *Area-of-Interests* (`AoI`) in sorted order and annotate them with their % contribution in the state. We only list significant `AoI` ($> 11\%$) for each state. Each state is also labeled with its average duration in {Years (Y), months (m)}. The labels given to these clusters reflect our interpretation of the user behavior and make disambiguation of the behavior easier in the text.

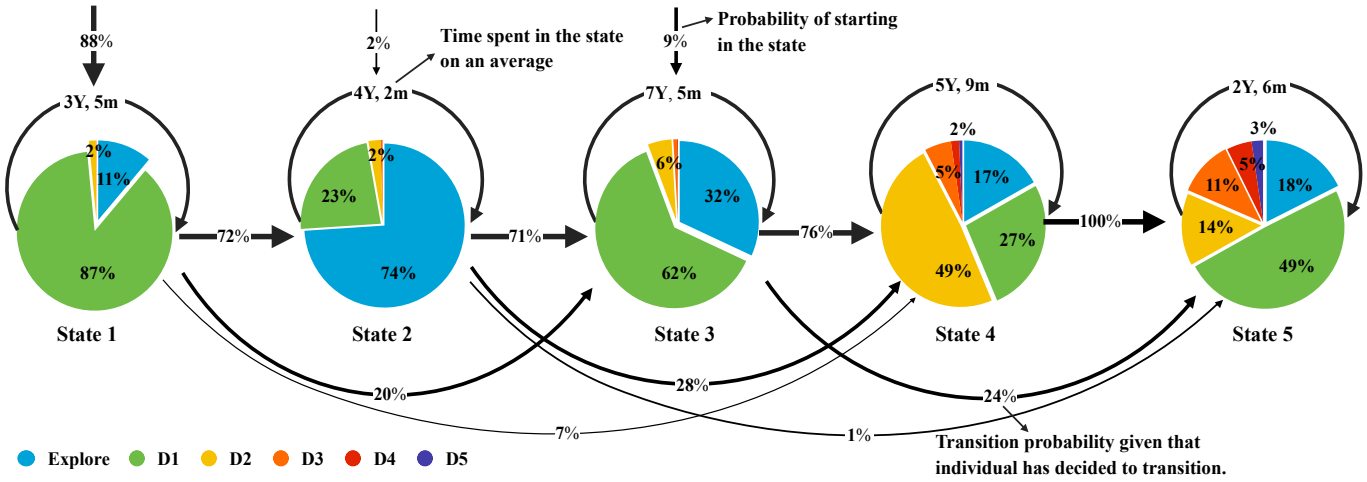


Figure 3.2: Trajectory (state sequence) for *Steady* archetype in the Academic Dataset. Each pie is a *latent state* or *behavioral stage* in the trajectory. It denotes the mean proportion of papers published in each *Area of Interest*'s in the latent state. Each state is also labeled with the average amount of time spent in the state. For example, in this cluster, 87% of publications in the first 3.5 years are in the author's primary `AoI` D_1 while rest 11% are in exploring other areas. The arrows on the top of each pie show the prior probability for starting in that state. As we learn a left-to-right G-HMM, an author can transition to its immediate next state or any later latent states. Each transition is labeled with the corresponding conditional transition probability i.e., transition probability given that the user has decided to transition. The arrows thickness is proportional to its weight. Authors in this cluster exhibit *steady* research interest in their primary `AoI` D_1. Some authors start contributing dominantly in their secondary `AoI`, D_2 in State 4. Though, they return to spending around half of their effort in D_1 in State 5.

their Ph.D. dissertation area, and hence, the research is more focused. After gaining some research experience, most authors move to the second *stage* where they start exploring other research areas denoted by a marked increase in their *Explore* `AoI`(more than 74%). However, in state 3 and beyond, authors from different archetypes follow different trajectories where they differ in how they change their dominant `AoI` over time while *exploring* other domains [8]. Below, we describe each archetype in more detail.

`Steady`: The first major archetype is of *steady* researchers, who mainly work in *one* `AoI` (i.e. their D_1) throughout their career. Fig 3.2 shows the state sequence of this archetype. We can see that most people start in their primary `AoI`, D_1 (state 1), which possibly reflects their Ph.D. education. After graduation, they spend some time *exploring* other areas while continuing to publish in D_1 (state 2), but move back to publishing in D_1 for a significant

[8] We observe that all archetypes have similar number of authors (table 3.3) and similar average number of active publication years.

portion of their careers, about 7.5 years (state 3). This shift is often again followed by a phase where they start working in another area, D_2, while continuing to publish in D_1 (state 4). They eventually revert to publishing in D_1 (state 5) towards the latter part of their careers. In the last state, they also publish widely in other areas (indicated by almost half of the pie divided between other D_m's), but their main interest remains D_1.

For example, Michael Jordan, professor at the University of California, Berkeley exhibits this research trajectory. He is a Machine Learning expert; his primary `AoI` D_1, and has secondary interests in Data Mining, Optimization, and Bioinformatics. Theory professor at University of Illinois, Urbana-Champaign (UIUC), Jeff Erickson is also assigned to this cluster; he also publishes in his primary `AoI` D_1 (Theory) with auxiliary interests in mathematical optimization.

`Diverse`: The second archetype consists of researchers with *diverse* research interests as they make significant contributions in multiple D_m's. Similar to *steady* researchers, these researchers research in their primary `AoI` D_1 while *exploring* other domains in the initial 3 states as shown in Table 3.2. They, then, publish in D_2 and D_1 while spending half time *exploring* other possible interests (state 4). They evolve to have a strong research presence in all 5 AoIs (state 5). This behavior suggests that authors of this archetype tend to work in interdisciplinary areas; or possibly projects with a broader scope which gains acceptance by different research communities. One notable example is Prof. Jiawei Han at UIUC, who started his academic career studying Databases and Data Mining, is also making notable contributions in Machine Learning and Bioinformatics lately. Another professor who started in Databases, Jaideep Srivastava of the University of Maryland, evolved on to research distributed implementation of databases, and also data mining and AI-related research simultaneously.

`Evolving`: These researchers have one dominant area of interest (`AoI`) in each state which *changes* with time. Their dominant area of interest (`AoI`) *evolves* from D_1 (72%) in state 1 to D_2 (66%) in state 4 to D_3(43%) in state 5. Even though their `AoI` shifts across stages, in any given stage, they remain focused on one area and do not publish much in other areas. James Foley, a professor in Georgia Tech, started in Computer Graphics and later switched to research on user-computer interfaces and recently, User Modeling. Natural Language Processing (NLP) expert Daniel Jurafsky at Stanford University, also steadily moved from pure NLP based research problems to Speech processing, and later to Machine Learning (ML). Also note, for Jurafsky, this evolution can be attributed to the broader field shift of using sophisticated ML models to solve NLP problems.

`Diffuse`: Authors of this archetype stay focused in one dominant area in each stage; while in the last stage, their research interests are *diffused*. Authors publish considerably in one

dominant area in first 3 stages; D_1 (state 1, 3) to D_2 (state 4). In the last state, which lasts around four years, the authors are infrequently publishing (less than three papers a year) in new subfields accounting for 74% of their publications. Hence, these authors have *diffused* research interests after they gain experience. Gerhard Weikum, professor at MPI Germany started in Databases area made a brief transition to Information Retrieval work and later started publishing in Machine Learning and Data Mining fields too. These area evolutions seem to be natural transitions as they are highly interrelated, which explains contributions in all fields. Anind Dey, professor at Carnegie Mellon University, initially worked on sensor technology and then switched to Web mining and Human Computing related research problems is also another example of this archetype.

3.3.2 Archetype variations across Gender

We now proceed to analyze the variations in the evolution of research interests (or archetypes) between male and female researchers. To this end, we manually annotate gender of all current and emeritus professors in top 50 Computer Science (CS) Universities as reported by U.S. News & World Report[9]. We consider only current and emeritus *Full* Professors as they typically have 15 or more years of publication history. This results in a total of 1084 authors in our dataset, 127 of whom are women. Table 3.3 shows the distribution across archetypes. We observe similar gender distribution in each archetype with the least number of women academics in *evolving* archetype.

Social Group	Steady	Diverse	Evolving	Diffuse
Male Professors (Top-50 US schools)	247	206	241	263
Female Professors (Top-50 US schools)	30	32	26	39
All authors in the dataset	1329	1080	1107	1062

Table 3.3: Statistics for discovered archetypes in relationship to different social groups.

While researchers from both genders in the same archetype c will traverse the same set of stages, they may differ in *how* they transition τ^c, and at *which* stage they start π^c. For this analysis, we first run our model on the entire dataset assigning archetypes to each individual. We then estimate separate model parameters for female λ_f^c and male λ_m^c researchers for each archetype c using the assigned values.

To quantify the difference between two models $(\lambda_f^c, \lambda_m^c)$ for archetype c, we compute their *likelihood ratio*. Likelihood ratio R_f^c of female researchers in archetype c is:

[9] `bit.ly/usnews-cs`

$$R_f^c = \exp\left(\frac{1}{|N_f^c|}\sum_{i\in N_f}\log\frac{P(X_i|\lambda_f^c)}{P(X_i|\lambda_m^c)}\right) \tag{3.1}$$

where N_f^c represents all female researchers in c-th archetype. The equation simplifies to say that $\log R_f^c$ is the average difference between log-likelihoods of a trajectory of a female researcher generated from their own model with those of male model of the same archetype. Thus, for instance, value of $R_f^c = 2$ denotes that female researchers are twice more likely to be generated by the model of their own gender than of the opposite gender. We compute a similar ratio, R_m^c, for men. We also conduct paired-sample t-test [125] between the two likelihood values similar to Section 3.3.1.

Gender	Steady	Diverse	Evolving	Diffuse
Male	2.10***	2.63**	1.15	1.10
Female	1.80***	1.64**	1.60***	1.38***

Table 3.4: Likelihood ratio for academics across genders within an archetype. It measures odds of a researcher being better explained by model for their gender than by model for the other gender.
$* = p < .05, ** = p < .01, *** = p < .001$

Table 3.4 shows the likelihood ratio with their p-values. Since most of the values are statistically significant, all researchers are better explained by the model for their gender, than by the model for the opposite gender. Male researchers are distinct for the steady and diverse archetypes, but not for the evolving and diffuse archetypes. For women, on average, the difference is larger, with the strongest difference seen for the steady, diverse, and evolving archetypes.

For the sake of brevity, we examine gender difference in only the *diverse* archetype in some detail. Figure 3.3 shows three interesting variations. First, we observe that women are much more likely to start in state 1 (92%), with a dominant area of interest (D_1) than in any other state. In contrast, men start in states 1, 2, 3, and 4, with only 70% starting in state 1. Both men and women skip stages, but women are more likely to skip a stage than men. For example, 50% of women skip stage 3, while only 36% of men do. Longer skips of two stages are rarer, and both women and men make these long skips at the same rate. Finally, there are clear differences between mid-career men and women (states 3, 4): women spend more time *exploring* mid-career (state 4) than men, and mid-career men spend more time in their starting area of interest (D_1, state 3) than women.

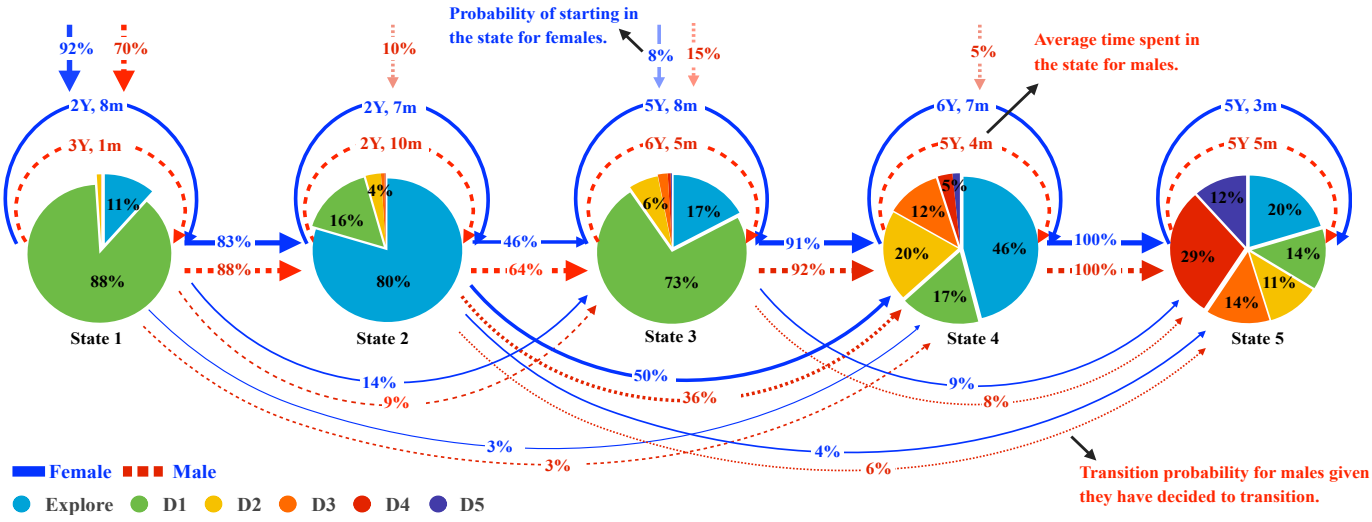


Figure 3.3: Gender wise representation of trajectory for researchers belonging to the *diverse* archetype in the Academic Dataset. The transitions in blue denote transition probabilities of female professors in the archetype while those in red represents probabilities for their male counterparts. Men start their career from later evolved stages while women make long term state transitions.

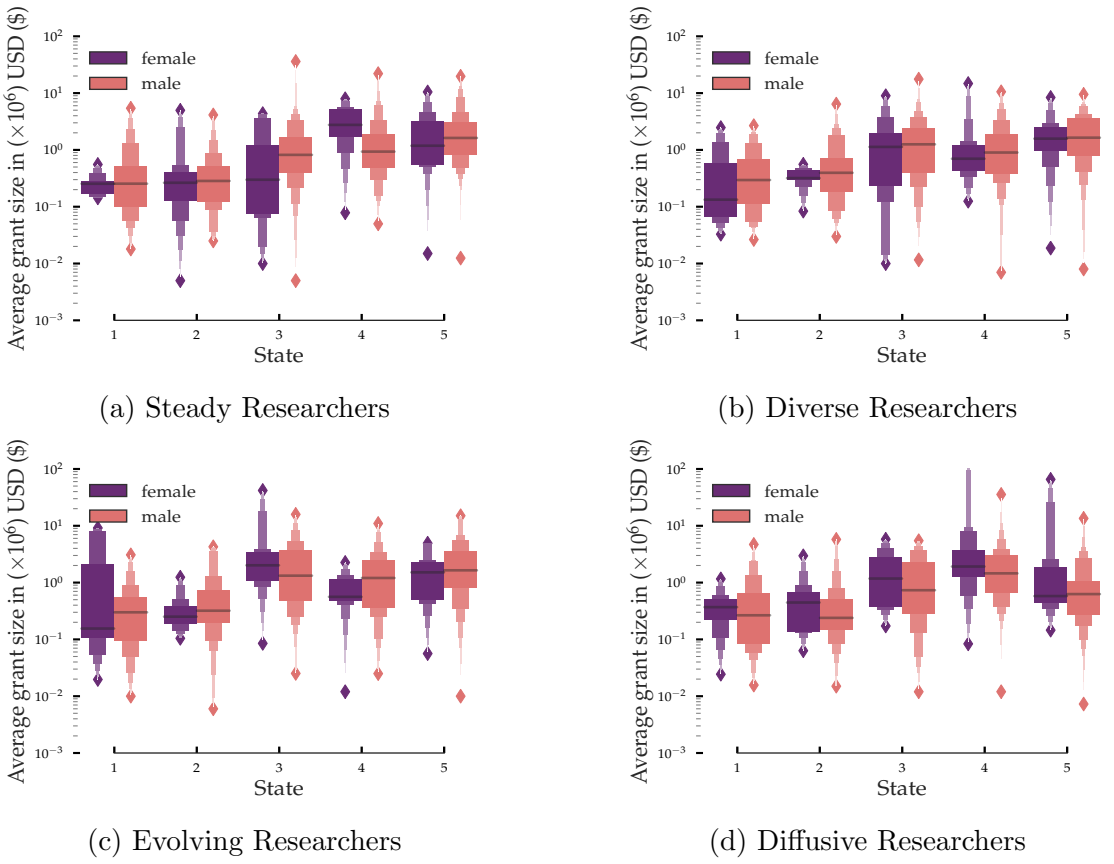
3.3.3 Grant income variability across Archetypes & gender within an Archetype

We next examine the relationship between variation in the academic trajectories and gender to research grants awarded at different stages of an academic career. We extract historical information of grants from the National Science Foundation, a large federal funding agency for Science & Engineering in the United States [10]. We consider grants with Principal Investigators (PI) from the same subset of CS professors in top-50 US universities as in Section 6.7. We collect information for 1062 professors and manually disambiguate names and identify gender by cross-validating with the researcher's webpage. Then, we compute the average grant money awarded to a researcher, at each stage in their trajectory. Figure 3.4, which shows letter-value plots of average grant size awarded as PI's, broken down by archetypes (steady, diverse, evolving or diffuse), stage within an archetype and gender, summarizes our findings.

Additionally, we conducted Kruskal-Wallis H-test [126] to establish the statistical significance of differences in grant money across latent states within an archetype. This test affirms that at least one latent state is different from another latent state within an archetype [11]. We then conducted Welch's t-test [127] between consecutive states to find the exact pair

[10] `bit.ly/nsfgrants`

[11] We also conducted H-test for the difference in average grant income across archetypes for the same state. However, we did not find any significant differences. Figure 3.4 can easily verify this lack of difference.



(a) Steady Researchers

(b) Diverse Researchers

(c) Evolving Researchers

(d) Diffusive Researchers

Figure 3.4: Letter value plots of total grant money awarded by NSF when author is a PI in each stage. In general, Professors get more grant money as they gain experience. Regardless of archetypes, grant income in state 3 is significantly higher from state 2 (p¡ .01). There are also significant differences across genders within a state of an archetype. For instance, for Evolving archetype, male professors get significantly more income than female professors in state 4 (p ¡ .01).

of states which are significantly different. We only tested with consecutive latent states as we are only interested in grant income changes as the author progresses through stages. Table 3.5 reports the state pairs for each archetype that are statistically different. In the rest of this section, we describe these results in detail.

Regardless of archetypes, we observe that in general authors tend to receive more grant money as they gain experience in Figure 3.4. On average, across archetypes and gender, PI's receive in state 5, four times the amount of grant money than state 1 ($p < .001$). Also for researchers across archetypes and across genders, we notice an uptick in grant income in state 3 from state 2 ($p < .01$ - Table 3.5). Let us qualitatively examine the *steady* researchers in detail, by comparing Figure 3.4 with Figure 3.2. State 2 in Figure 3.2 shows the researchers exploring different topics, whereas, in state 3, they are spending a significant part of their time on their main domain D_1. Also, notice that 36% of the researchers never

Archetype (H-test)	State Pair (t-test)
Steady***	State 2 vs 3**
Diverse***	State 2 vs 3*** State 4 vs 5*
Evolving***	State 2 vs 3*** State 3 vs 4* State 4 vs 5**
Diffuse***	State 2 vs 3*** State 4 vs 5*

Table 3.5: Statistical significance tests for the differences in grant money across latent states within an archetype. Shown are only those tests that are statistically significant. H-test [126] confirms that at least one state is different from another state of the archetype; t-test [127] was then conducted between each consecutive states within the archetype to determine the differing states.
$* = p < .05, ** = p < .01, *** = p < .001$

Archetype	Latent State (t-test)
Steady	State 1* State 4*
Diverse	State 2*
Evolving	State 4** State 5*
Diffuse	Not significant

Table 3.6: Statistical significance tests [127] for the differences in grant money across gender in each state within an archetype. Shown are only those tests that are statistically significant.
$* = p < .05, ** = p < .01, *** = p < .001$

visit state 2 - 27% skip state 2, and 9% of the researchers start in state 3. Since state 1 typically represents the time spent by the researchers in their Ph.D., and with 74% time spent in an explore stage in state 2, it is not surprising that we see limited grant income in their first two states. State 3, perhaps reflects a sustained focus on their domain D_1, and this pays off in terms of grant income. Similar qualitative arguments follow for the other archetypes.

However, the grant trajectories over states is different for each archetype ($p < .001$). Let us examine statistically different state pairs from Table 3.5 in Figure 3.4. Steady researchers see a big uptick in their grant income in state 3 and their subsequent grant income is similar in magnitude ($p < .01$). The grant income for diverse researchers (who have more than one dominant area) increases steadily over states($p < .05$). For evolving researchers (who change their dominant area), the grant income rises (state 3, $p < .001$), falls (state 4, $p < .05$) and

rises (state 5, $p < .01$), reflecting a degree of unpredictability accompanying *changing area of interest*. Diffuse researchers, have a pattern similar to steady researchers except in state 5 ($p < .05$), when the income dips, perhaps due to spending time in *too many areas*.

To determine differences in grant income across gender, we further conducted t-test [127] between grant distributions of female and male professors in each state within an archetype. Table 3.6 reports significantly different states within each archetype. We again examine these statistically different states from Table 3.6 in Figure 3.4. Evolving women receive significantly *lower* income than evolving men when they *switch to new areas* in state 4 and 5 ($p < .05$). On the other hand, in our dataset, steady women receive significantly *higher* grant income than steady men when they *switch areas* in state 4 ($p < .05$). In general, we observe that men show greater grant income variability than do women. The variability is statistically significant ($p < .05$) during early career in state 1 and 2 for Steady and Diverse researchers respectively. We do not observe significant differences in grant income of male and female Diffusive researchers.

3.3.4 StackOverflow Archetypes

We now describe archetypes learned for the Stack Overflow data. Table 3.7 depicts the latent states for all archetypes. We label the 4 archetypes discovered as *Expert*, *Seekers*, *Enthusiasts* and *Facilitators*. Posting Comments is the most frequent activity in all archetypes as it is a very low cost activity. Moderator actions and Edits are least favored activities by Stack Overflow users. Most of the users spend initial sessions for *posting questions* (state 1) and significant proportion of their later sessions in *posting answers or comments* (state 6).

`Experts` users join the community to answer queries or post clarifications or edit answers (state 1-6). They spend at least 68% of their sessions in *posting answers* (state 1 & 3). They rarely ask questions of their own. In communities like Stack Overflow, it is vital to have a dedicated group of experts answering queries for it to be sustainable.

`Information Seekers` join the community for getting answer to their queries accounting for 69% of activities per session (state 1). They briefly start contributing by posting answers to the community (state 3) but they end up again in *commenting* (state 6) or *asking questions* (state 5).

`Enthusiasts` start by asking questions and posting comments (state 1). They, then, start answering questions and commenting on other answers (state 2). They briefly stay (4 sessions) in edit state (state 3) but end up migrating to either commenting again (state 4) or asking questions and commenting (state 5). We denote them as *Enthusiasts* as they use the platform to post questions while simultaneously answering queries from their acquired

	Experts	Seekers	Enthusiasts	Facilitators
State 1	{14.4S}	{3.7S}	{12.6S}	{15.4S}
	A (68%)	**Q** (69%)	**C** (42%)	**Q**(50%)
	C (19%)	C (19%)	**Q** (39%)	C (32%)
State 2	{17.8S}	{8.4S}	{9.5S}	{26.4S}
	C (60%)	**C** (51%)	**A** (49%)	**C** (44%)
	A (25%)	*Q* (33%)	*C* (32%)	*A* (28%)
			E (12%)	*E* (22%)
State 3	{9S}	{2.2S}	{3.7S}	{8.4S}
	A (87%)	**A** (84%)	**E** (82%)	**A** (87%)
State 4	{12.3S}	{5.3S}	{9S}	{24.2S}
	C (82%)	**C** (72%)	**C** (75%)	**C** (68%)
		Q (15%)	*A* (10%)	**A** (14%)
				E (13%)
State 5	{5S}	{3.7S}	{10.5S}	{14.3S}
	E (45%)	**Q** (85%)	**Q** (40%)	**E** (63%)
	C (28%)		C (33%)	*C* (22%)
	A (21%)		*E* (16%)	
State 6	{11S}	{7.7S}	{5S}	{21S}
	C (48%)	**C** (55%)	**A** (61%)	**C** (57%)
	A (38%)	**Q** (23%)	C (23%)	*E* (24%)
		E (11%)	*E* (11%)	*A* (13%)

Table 3.7: Learned mean vector for each state for four archetypes in the Stack Overflow Dataset. We list the *activities* in sorted order and annotate them with their % contribution in the state. We list main activities ($> 11\%$) for each state. Each state is also labeled with it's average number of sessions. The labels reflect our own interpretation of the user behavior.

knowledge.

`Facilitators` join for information seeking (state 1) but start posting answers, clarifying and editing in state *2-3*. However, later on they take a more subdued approach and only post comments. The reason for this decreased interest is hard to gauge but identifying these users and retaining their interest could be important to sustain the community.

In summary, we identify four dominant archetypes for researchers: steady, diverse, evolving, and diffuse. We observe differences in the evolution of male and female researchers within the same archetype. When we examine the diverse archetype in detail, we observe that women and men differ in where they start, rate of transition, and time spent in mid-career. The differences in grant income are salient across states within an archetype. In general, grant income increases with experience. We also observe differences across genders within a stage of an archetype mostly accompanied by an area switch. Finally, we also identified dominant archetypes for StackOverflow community: experts, information seekers, enthusiasts and facilitators.

3.4 QUANTITATIVE EXPERIMENTS

In this section, we evaluate our model on two different tasks, Future Prediction and Perplexity. We describe the baselines in Section 3.4.1 and report results in Section 3.4.2.

3.4.1 Baselines

Distance G-HMM: Our first baseline uses the G-HMM clustering model as defined in [128]. In this baseline, we learn a G-HMM for each user and then cluster the models using distance metric δ, the symmetric KL divergence (d_{kl}) between two G-HMMs [129].

$$d_{kl}(\lambda^p, \lambda^q) = \frac{1}{N_p} \sum_{i \in N_p} log \frac{P(X_i|\lambda^p)}{P(X_i|\lambda^q)}, \tag{3.2}$$

We use k-medoids clustering; since this method does not give a representative model for each cluster, we additionally learn a G-HMM per cluster. For a fair comparison, we set k, the number of clusters to be the same as our model.

Vector Autoregressive Model (VAR): VAR models are used to model multivariate time series data [130]. It assumes that each variable in the vector is a linear function of its own past values as well as other variables. For each user sequence $\mathbf{X_i}$, jth session is modeled

as,

$$\vec{X}_{ij} = A_1\vec{X}_{ij-1} + \ldots + A_p\vec{X}_{ij-p} + u_j \tag{3.3}$$

where A_i is M X M matrix, $u_j \sim \mathcal{N}(0, \Sigma_u)$ and we set $p = 1$ as in first-order Markov models.

No Evolution: In this baseline, we assume that individuals *do not evolve* in their lifespan. This baseline is a simplified version of our model. It assumes that there are different archetypes but that each archetype has only one state. Hence, all sessions of a sequence are generated from a single multivariate Gaussian.

Prior work on activity sequence prediction baselines [15, 16] deals with discrete data. However, as we represent each session as a continuous vector, these approaches are not directly comparable and adapting them to our problem is nontrivial.

3.4.2 Tasks

Future Activity Prediction: In this task, we predict the future behavior of an individual given her history. We assign the first 90% sessions of each sequence for training and predict the behavior in future sessions (the remaining 10% of the sequence). We first use all the training sessions to learn the parameters of our model. Then, for each sequence, we run the Viterbi algorithm to decode the state assignment of its test sessions, t'_i. The test sessions of the i-th user will have same archetype assignment c_i determined in the training session for that user.

We compute Jensen-Shannon(d_{js}) divergence between the mean $\mu^{c_{ij}}$ of the assigned state Y_{ij} and the observed vector X_{ij}. d_{js} is a symmetric K-L divergence between two vectors. We report the average $\bar{\Delta}$ over all test sessions:

$$\bar{\Delta} = \frac{1}{|T|}\sum_{i\in N, j\in t'_i} d_{js}(\mu^{c_{ij}}, X_{ij}), \tag{3.4}$$

$$d_{js}(\mu^{c_{ij}}, X_{ij}) = \frac{1}{2}d_{kl}(\mu^{c_{ij}}, p) + \frac{1}{2}d_{kl}(X_{ij}, p), \tag{3.5}$$

where, $p = \frac{1}{2}(\mu^{c_{ij}} + X_{ij})$ and d_{kl} measures KL divergence distance. For VAR, we use the model learnt on training sessions of user i to make prediction for her future sessions.

Table 3.8 shows our results on this task. Our model outperforms the baselines for all Stack Exchange datasets with an average improvement of about 32% and 24% on the Academic dataset. Hence, learning archetypes can also help us to accurately predict an individual's future behavior in the social network.

Dataset	Our Model	VAR	Distance G-HMM	No Evolution
Academic	0.22	0.31	0.42	0.29
StackOverflow	0.23	0.36	NA	0.37
English	0.19	0.29	0.26	0.31
Money	0.19	0.52	0.32	0.32
Movies	0.23	0.35	0.35	0.37
CrossValidated	0.21	0.38	0.33	0.35
Travel	0.19	0.30	0.25	0.29
Law	0.19	0.26	0.33	0.27

Table 3.8: Average Jensen-Shannon divergence of future sessions using 90-10% split of each user sequence. Lower values are better. Distance HMM did not converge on StackOverflow dataset.

Perplexity Perplexity measures how surprised the model is on observing an unseen user sequence. A lower value of perplexity indicates low surprise and hence a better model.

$$P_x = -\frac{1}{|T|}\sum_{i \in T} \max_{c \in C} \left(\log P(\mathbf{X_i^T}|\lambda^c)\right) \tag{3.6}$$

where, $\mathbf{X_i^T}$ represents a test sequence in Test Set T, and λ_c represents the parameters of the G-HMM corresponding to the c-th archetype. We assign $\mathbf{X_i^T}$ to the archetype c with maximum likelihood. Perplexity is then computed as the average likelihood of all test sequences. In general, $P(\mathbf{X_i^T}|\lambda^c))$ is bound between [0,1] but as we model continuous data with multivariate Gaussian distribution, probability is computed as a density function and can be > 1.

Table 3.9 reports average perplexity after five-fold cross-validation. Note that for this experiment, the model predicts the entire trajectory of a new user. We could not use the regression baseline (VAR) as it is not a generative model and can not predict an entirely new sequence. Our model beats best performing baseline by 149% on Academic and by around 25% on average for StackExchange communities. Hence, our model also effectively predicts the behavior of future individuals joining the social network. Note that our model gives negative perplexity values i.e., negative log values. It indicates that the likelihood is more than one due to the Gaussian kernel, as mentioned earlier.

Discussion: For future prediction, our model performs better than the VAR model. It shows that modeling cluster of sequences gives a better estimate than modeling each user sequence separately. Also, if we assume no evolution and just cluster users according to their behavior i.e., *No Evolution* model, we obtain worse results indicating that individu-

Dataset	Our Model	Distance G-HMM	No Evolution
Academic	−18.37	37.73	100.79
StackOverflow	487.68	NA	678.62
English	306.38	559.65	471.14
Money	415.85	557.69	570.51
Movies	596.10	724.15	743.73
CrossValidated	398.44	514.74	554.31
Travel	494.06	645.64	666.97
Law	368.89	508.08	482.27

Table 3.9: Average Perplexity on unseen user sequences after 5-fold cross validation. Lower values are better. Note negative log values are because of continuous densities.

als behavior does not stay constant over time. Our model also outperforms the similarity distance-based clustering method: Distance G-HMM [128], which is also the strongest baseline. It first estimates the G-HMM model for each user sequence and then clusters these models. Estimating model for each sequence can be noisy, especially if the user sequence has a short length. Instead, when we jointly learn G-HMM model parameters and cluster sequences, we learn a better approximation.

Full vs. Left-Right Transition Matrix: We also test our model with unconstrained full transition matrix where users can jump from one state to any other state in the HMM. We obtain slightly better results with this model for the future prediction task. This improvement can be due to more degrees of freedom, but then, it is also computationally expensive to learn. However, our model gives comparable results with much fewer parameters. Also, with full transition matrix, learned states are not interpretable in the context of evolution. As Yang et al. [15] and Knab et al. [16] also noted, forward state transitions accurately models the natural progression of evolution, we thus chose to work with a forward transition matrix.

3.5 CONCLUSION

In this chapter, we aimed to discover the archetypical research behavior of Academics. The observation that despite surface variation in terms of sub-fields, the change in behavior exhibits regularities, motivated our research. We introduced a novel Gaussian Hidden Markov Model Cluster (G-HMM) to identify archetypes and evolutionary patterns within each archetype. We chose to work with G-HMM's since they allow for: variations in trajectories and different evolutionary rates; constrain how individuals can evolve; are interpretable.

We identified four distinct archetypes of computer scientists: steady, diverse, evolving, and diffuse and showed examples of computer scientists from different sub-fields that share the same archetype. We analyzed full professors from the top 50 CS departments to understand gender differences within archetypes. Women and men differ within an archetype (e.g., diverse) in where they start, rate of transition and research interests during mid-career. We further analyzed grant income of these professors to understand the effect of gender and archetype on income. The differences in income are salient across states within an archetype rather than across archetypes. There also exist significant differences across genders within a state of an archetype. We observed that most of the grant income variability is accompanied by a shift in the dominant research area of the academic. In light of our findings, we propose the funding agencies to be cautious of these differences when deciding on grant applications submitted by researchers venturing into new areas.

To the best of our knowledge, we are the first one to provide a principled framework to model and identify interpretable individual trajectories in academia. Our model can be easily used to identify trajectory in other domains like medicine, physics, and business. Further work on the comparison of research trajectories from the stem and non-stem fields could be an exciting research direction.

For StackOverflow, discovered archetypes could be labeled as: *Experts*, *Seekers*, *Enthusiasts* and *Facilitators*. We showed strong quantitative results with competing baselines for future activity prediction and perplexity.

CHAPTER 4: USER LATENT BEHAVIOR REPRESENTATION

User behavior is primarily represented using discrete actions like items purchased in the platform, actions performed in the social network or question/answers posted in a CQA forum. However, there is abundant information available in the form of textual content associated with most of these user actions, such as review text or answer text. Analysis of the textual content associated with these actions can provide an in-depth understanding of user preferences. To this end, in this chapter, we estimate latent user behavior–aspect-based reliability by exploiting the semantic similarity between user provided answers and questions. This latent reliability is in turn used to infer trustworthiness of answers in a CQA forum [131] [1].

4.1 OVERVIEW

Users are increasingly turning to community discussion forums to solicit domain expertise, such as querying about inscrutable political events on history forums or posting a health-related issue to seek medical suggestions or diagnosis. While these forums may be useful, due to almost no regulations on post requirements or user background, most responses contain conflicting and unreliable information [132]. This misinformation could lead to severe consequences, especially in health-related forums, that outweigh the positive benefits of these communities. Currently, most of the current forums either employ moderators to curate the content or use community voting. However, both of these methods are not scalable [59]. This creates a dire need for an automated mechanism to estimate the trustworthiness of the responses in the online forums.

In general, the answers written by reliable users tend to be more trustworthy, while the users who have written trustworthy answers are more likely to be reliable. This mutual reinforcement, also referred as the truth discovery principle, is leveraged by previous works that attempt to learn information trustworthiness in the presence of noisy information sources with promising results [133, 134, 135, 136]. This data-driven principle particularly works for community forums as they tend to be of large scale and exhibit redundancy in the posts and comments.

Community discussion forums encompass various topics, or aspects. A significant deficiency of previous work is the lack of aspect-level modeling of a user's reliability. This het-

[1] This is a joint work with Alex Morales. I was responsible for the conceptualization, experiments and writing. He took care of the idea conceptualization, data collection, model implementation and writing.

erogeneity is especially true for discussion forums, like Reddit, with communities catering to broad themes; while within each community, questions span a diverse range of sub-topics. Intuitively, a user's reliability will be limited to only a few topics, for instance, in a science forum, a biologist could be highly knowledgeable, and in turn reliable, when she answers biology or chemistry-related questions but may not be competent enough for linguistic queries.

Another challenge is the diversity of word expressions in the responses. Truth discovery based approaches treat each response as categorical data. However, in discussion forums, users' text responses can include contextually correlated comments [67]. For instance, in the *context* of a post describing symptoms like "headache" and "fever", either of the related responses of a viral fever or an allergic reaction can be a correct diagnosis. However, unrelated comments in the post should be unreliable; for instance, a comment giving a diagnosis of "bone fracture" for the above symptoms.

CrowdQM addresses both limitations by jointly modeling the aspect-level user reliability and latent trustworthy comment in an optimization framework. In particular, 1) CrowdQM learns user reliability over fine-grained topics discussed in the forum. 2) Our model captures the semantic meaning of comments and posts through word embeddings. We learn a trustworthy comment embedding for each post, such that it is semantically similar to comments of reliable users on the post and also similar to the post's context. Contrary to the earlier approaches [137, 55, 64], we propose an *unsupervised model* for comment trustworthiness that does not need labeled training data.

We verified our proposed model on the trustworthy comment ranking task for three Ask* *subreddit communities*. Our model outperforms state-of-the-art baselines in identifying the most trustworthy responses, deemed by community experts and community consensus. We also show the effectiveness of our aspect-based user reliability estimation and word embeddings qualitatively. Furthermore, our improved model of reliability enables us to identify reliable users per topic discussed in the community.

4.2 METHODOLOGY

A challenge in applying truth discovery to community discussion forums is capturing the diversity of user's knowledge and the diversity of word usage in the answers. To address it, we model user-aspect reliability and learn semantic representations of the comments.

4.2.1 Problem Formulation

Each *submission* is a post, i.e., question, which starts a discussion thread while a *comment* is a response to a submission post. Formally, each submission post, m, is associated with a set of terms, c_m. A user, n, may reply with a comment on submission m, with a set of terms $w_{m,n}$. $\mathcal{V}$ is the vocabulary set comprising of all terms present in our dataset i.e. all submissions and comments. Each term, $\omega \in \mathcal{V}$ has a corresponding word-vector representation, or word embedding, $\boldsymbol{v}_\omega \in \mathbb{R}^D$. Thus, we can represent a *post embedding* in terms of its constituent terms, $\{\boldsymbol{v}_c\}, \forall c \in c_m$. To capture the semantic meaning, we represent each comment as the mean word-vector representation of their constituent terms[2]. Formally, we represent the comment given on the post m by user n as the *comment embedding*, $\boldsymbol{a}_{m,n} = |w_{m,n}|^{-1} \sum_{\omega \in w_{m,n}} \boldsymbol{v}_\omega$. Our model treats the post embeddings as static and learns the comment word embeddings. The set of posts user n has commented on is denoted by $\mathcal{M}_n$ and the set of users who have posted on submission m is denoted as $\mathcal{N}_m$.

There are K aspects or topics discussed in the forum, and each post and comment can be composed of multiple *aspects*. We denote submission m's distribution over these aspects as the *post-aspect distribution*, $\boldsymbol{p}_m \in \mathbb{R}^K$. Similarly, we also compute, *user-aspect distribution*, $\boldsymbol{u}_n \in \mathbb{R}^K$, learned over all the comments posted by the user n in the forum. This distribution captures familiarity (or frequency) of user n with each aspect based on their activity in the forum. Each user n also has a *user reliability* vector defined over K aspects, $\boldsymbol{r}_n \in \mathbb{R}^K$. The reliability captures the likelihood of the user providing a trustworthy comment about a specific aspect. Note high familiarity in an aspect does not always imply high reliability in the same aspect. Table 4.1 presents all the symbols and their meanings.

For each submission post m associated with a set of responses $\{\boldsymbol{a}_{m,n}\}$, our goal is to estimate the real-valued vector representations, or *latent trustworthy comment* embeddings, $\boldsymbol{a}^*_m \in \mathbb{R}^D$. We also simultaneously infer the *user reliability* vector $\{\boldsymbol{r}_n\}$ and update the word embeddings $\{\boldsymbol{v}_\omega\}$. The latent trustworthy comment embeddings, $\boldsymbol{a}^*_m$, can be used to rank current comments on the post.

4.2.2 Proposed Method

Our model follows the truth discovery principle: trustworthy comment is supported by many reliable users and vice-versa. In other words, the weighted error between the trustworthy comment and the given comments on the post is minimum, where user reliabilities provide the weight. We extend the approach to use an aspect-level user reliability and

[2]Sentence, and furthermore document representation is a complex problem. In our work, we explore a simple aggregation method for comment semantic composition [138].

Notation	Definition
$\mathcal{V}$	vocabulary associated with submission/comments
$\mathcal{M}_n$	submissions where user n has commented
$\mathcal{N}_m$	users who have commented on submission m
$\mathcal{D}_\omega$	comment-submission pairs where ω term appears
c_m	set of terms associated with submission post m
v_ω	word embedding for term ω
$a_{m,n}^{-\omega}$	aggregate embedding of terms in w_m^n excluding term ω
$a_{m,n}$	embedding of the comment from user n on post m
a_m^*	embedding of the latent trustworthy comment for post m
$u_n^{(k)}$	kth aspect weight for user n
$p_m^{(k)}$	kth aspect weight for submission post m
$r_n^{(k)}$	learned user-aspect reliability for user n for aspect k
R_m^n	the user-post reliability for the nth user and the mth post
M	total number of posts
N	total number of users

Table 4.1: Symbols and their meaning

compute a post-specific reliability weight. We further compute the error in terms of the *embeddings* of posts and comments to capture their semantic meaning.

In particular, we minimize the *embedding error*, $E_{m,n} = ||\boldsymbol{a}_m^* - \boldsymbol{a}_{m,n}||^2$, i.e., mean squared error between learned *trustworthy comment* embeddings, $\boldsymbol{a}_m^*$ and comment embeddings, $\boldsymbol{a}_{m,n}$, on the post m. This error ensures that the trustworthy comment is semantically similar to the comments given for the post.

Next, to ensure context similarity of the comments with the post, we compute the *context error*, $Q_{m,n} = |c_m|^{-1} \sum_{c \in c_m} ||\boldsymbol{a}_{m,n} - \boldsymbol{v}_c||^2$, reducing the difference between the *comment embeddings* and *post embeddings*. The key idea is similar to that of the distributional hypothesis that if two comments co-occur a lot in similar posts, they should be closer in the embedding space.

Further, these errors are weighted by the aspect-level reliability of the user providing the comment. We estimate the reliability of user n for the specific post m through the *user-post reliability* score, $R_{m,n} = \boldsymbol{r}_n \odot s(\boldsymbol{u}_n, \boldsymbol{p}_m) = \sum_k \boldsymbol{r}_n^{(k)} \cdot (\boldsymbol{u}_n^{(k)} \cdot \boldsymbol{p}_m^{(k)})$. The $\odot$ symbol represents the Hadamard product. This scores computes the magnitude of *user reliability* vector, $\boldsymbol{r_n}$, weighted by the similarity function $s(.)$. The similarity function $s(\boldsymbol{u}_n, \boldsymbol{p}_m)$ captures user familiarity with post's context by computing the product of the aspect distribution of user n and post m. Thus, to get a high *user-post reliability* score, $R_{m,n}$, the user should both be reliable and familiar to the aspects discussed in the post.

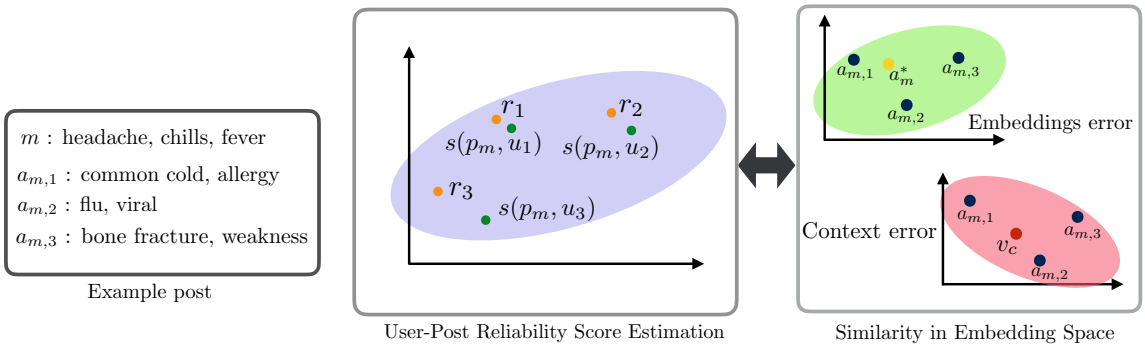


Figure 4.1: An illustrative toy example detailing our model components. The left-hand side details the user-post reliability score estimation, $R_{m,n}$, that is a function of similarity function $s(.)$ between the user and post aspect distributions and user aspect reliabilities, $\boldsymbol{r}_n$. In the right-hand, we learn trustworthy comment embedding, $\boldsymbol{a}^*_m$, such that it is similar to user comments, $\boldsymbol{a}_{m,n}$ which are, in turn, similar to the post context $\boldsymbol{v}_c$.

Finally, these errors are aggregated over all the users and their comments. Thus, we define our objective function as follows,

$$\min_{\{\boldsymbol{a}^*_m\},\{\boldsymbol{v}_\omega\},\{\boldsymbol{r}_n\}} \sum_{n=1}^{N} \sum_{m \in \mathcal{M}_n} \underbrace{R_{m,n}}_{\text{user-post reliability}} \left(\underbrace{E_{m,n}}_{\text{embedding error}} + \beta \odot \underbrace{Q_{m,n}}_{\text{context error}} \right) \tag{4.1}$$

$$\text{s.t.} \sum_{n=1}^{N} e^{-\boldsymbol{r}_n^{(k)}} = 1; \forall k$$

where N is the number of users. $R_{m,n} \cdot E_{m,n}$ ensures that the latent trustworthy comment embeddings are most similar to comment embeddings of *reliable* users for post m. While $R_{m,n} \cdot Q_{m,n}$ ensures trust aware learning of contextualized comment embeddings. The hyperparameter β controls the importance of context error in our method. The exponential regularization constraint, $\sum_{n=1}^{N} e^{-\boldsymbol{r}_n^{(k)}} = 1$ for each k, ensures that the reliability across users are nonzero. Figure 4.1 shows the overview of our model using a toy example of a post in a medical forum with flu-like symptoms. The commenters describing flu-related diagnoses are deemed more reliable for this post.

4.2.3 Solving the Optimization Problem

We use coordinate descent [139] to solve our optimization problem. In particular, we solve the equation for each variable while keeping the rest fixed.

Case 1: Fixing $\{\boldsymbol{r}_n\}$ and $\{\boldsymbol{v}_\omega\}$, we have the following update equation for $\{\boldsymbol{a}^*_m\}$:

$$\boldsymbol{a}^*_m = \frac{\sum_{n\in\mathcal{N}_m} R_{m,n}\boldsymbol{a}_{m,n}}{\sum_{n\in\mathcal{N}_m} R_{m,n}} \tag{4.2}$$

Thus, the latent *trustworthy comment* is a weighted combination of comments where weights are provided by the *user-post reliability* score $R_{m,n}$. Alternatively, it can also be interpreted as a reliable summarization of all the comments.

Case 2: Fixing $\{\boldsymbol{a}^*_m\}$, $\{\boldsymbol{v}_\omega\}$, we have the following update equation for $\{\boldsymbol{r}^{(k)}_n\}$:

$$\boldsymbol{r}^{(k)}_n \propto -\ln \sum_{m\in\mathcal{M}_n} s(\boldsymbol{u}^{(k)}_n, \boldsymbol{p}^{(k)}_m)\,(E_{m,n} + \beta Q_{m,n}) \tag{4.3}$$

Reliability of a user in aspect k is inversely proportional to the errors with respect to the latent trustworthy comment $\boldsymbol{a}^*_m$ ($E_{m,n}$) and submission's context $\boldsymbol{v}_c$ ($Q_{m,n}$) over all of her posted comments ($\mathcal{M}_n$). The embedding error ensures that if there is a large difference between the user's comment and the trustworthy comment, her reliability becomes lower. The context error ensures that non-relevant comments to the post's context are penalized heavily. In other words, a reliable user should give trustworthy and contextualized responses to posts.

This error is further weighed by the similarity score, $s(.)$, capturing familiarity of the user with the post's context. Thus, familiar users are penalized higher for their mistakes as compared to unfamiliar users.

Case 3: Fixing $\{\boldsymbol{a}^*_m\}, \{\boldsymbol{r}^{(k)}_n\}$, we have the following update equation for $\{\boldsymbol{v}_\omega\}$:

$$\boldsymbol{v}_\omega = \frac{\sum_{<m,n>\in D_\omega} R_{m,n}\left(\boldsymbol{a}^*_m + \beta|c_m|^{-1}\sum_{c\in c_m}\boldsymbol{v}_c\right) - R_{m,n}(\beta+1)|c_m|^{-1}\boldsymbol{a}^{-\omega}_{m,n}}{\sum_{<m,n>\in D_\omega} R_{m,n}(\beta+1)} \tag{4.4}$$

where $< m,n >\in D_\omega = \{(m,n)|\omega \in w_{m,n}\}$ and $\boldsymbol{a}^{-\omega}_{m,n} = |w_{m,n}|^{-1}\sum_{\omega'\in w_{m,n}\setminus\{\omega\}}\boldsymbol{v}_{\omega'}$. To update $\boldsymbol{v}_\omega$, we only consider those comment and submission pairs, D_ω, in which the particular word appears. The update of the embeddings depend on the submission context $\boldsymbol{v}_c$, latent trustworthy comment embedding, $\boldsymbol{a}^*_m$ as well as *user-post reliability* score, $R_{m,n}$. Thus, word embeddings are updated in a trust-aware manner such that reliable user's comments weigh more than those of unreliable users as they can contain noisy text. Note that there is also some negative dependency on the contribution of other terms in the comments.

Implementation Details: We used popular Latent Dirichlet Allocation (LDA) [140] to

estimate aspects of the posts in our dataset[3]. Specifically, we combined the title and body text to represent each post. We applied topic model inference to all comments of user n to compute its combined aspect distribution, $\boldsymbol{u}_n$. We randomly initialized the user reliability, $\boldsymbol{r}_n$. We initialized the word embeddings, $\boldsymbol{v}_\omega$, via word2vec [81] trained on our dataset. We used both unigrams and bigrams in our model. We fixed β to 0.15.[4] The model converges after only about six iterations indicating quick approximation. In general, the computational complexity is $O(|\mathcal{V}|NM)$; however, we leverage the data sparsity in the comment-word usage and user-posts for efficient implementation.

[3]We ran LDA with 50 topics for all experiments and examined its sensitivity in Section 4.3.3.
[4]We did not find a significant change in results for different values of β.

4.3 EXPERIMENTS

In this section, we first discuss our novel dataset, followed by experiments on the outputs learned by our model. In particular, we evaluate the trustworthy comment embeddings on the comment ranking task while we qualitatively evaluate user reliabilities and word embeddings. For brevity, we focus the qualitative analysis on our largest subreddit, askscience.

4.3.1 Dataset

We evaluate our model on a widely popular discussion forum Reddit. Reddit covers diverse topics of discussion and is challenging due to the prevalence of noisy responses.

Dataset	**Created**	N	N_e	M	$\lvert a_{m,e}\rvert$	$\lvert w_{m,n}\rvert$
*Docs	07/13	3,334	286	17,342	10,389	53.5
*Science	04/10	73,463	2,195	100,237	70,108	74.0
*Historians	08/11	27,264	296	45,650	30,268	103.4

Table 4.2: Dataset statistics for the subreddit communities. The symbol meaning are as follows: N and M denotes total users and posts respectively; N_e: number of experts; $|a_{m,e}|$: number of posts with at least one expert comment; $|w_{m,n}|$: average comment word length.

We specifically tested on *Ask** subreddits as they are primarily used to seek answers to a variety of topics from mundane issues to serious medical concerns. In particular, we crawled data from three subreddits, /r/askscience, /r/AskHistorians, and /r/AskDocs from their inception until October 2017 [5]. While these subreddits share the same platform, the communities differ vastly, see Table 4.2. We preprocessed the data by removing uninformative comments and posts with either less than ten characters or containing only URLs or with a missing title or author information. We removed users who have posted less than two comments and also submissions with three or fewer comments. To handle sparsity, we treated all users with a single comment as "UNK".

For each submission post, there is a flair text denoting the subtopic of the post, referred as the *submission flair* that is either Moderator added or self-annotated. We denote *submission flair* as the *category* of the post, e.g. Physics, Chemistry, Biology are some categories in AskScience. Similarly, users have *author flairs* attributed next to their user-name, e.g. Astrophysist, Bioengineering. They describe the user's educational background and help the OP in assessing user reliability. In order to get verified and obtain a flair, the users must send the moderators anonymized verification documents, including certification numbers,

[5] `praw.readthedocs.io/en/latest/`

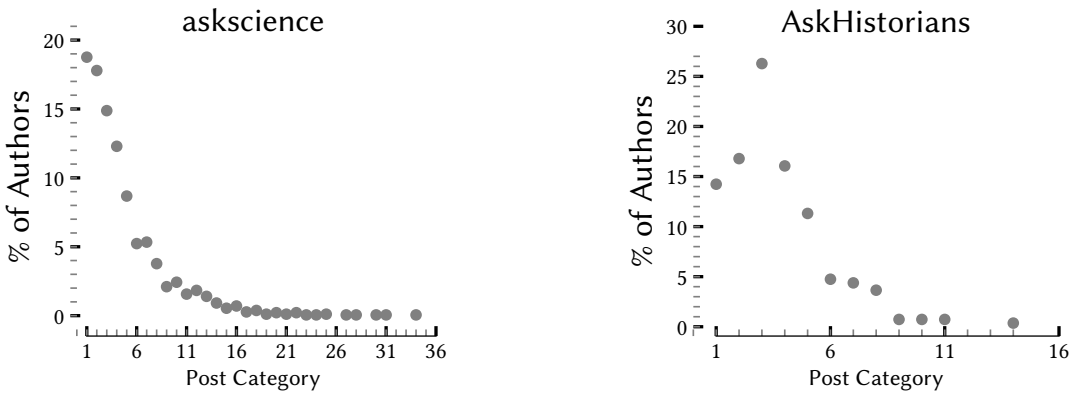


Figure 4.2: Frequency plot of % of authors commenting on the post with unique submission flairs.

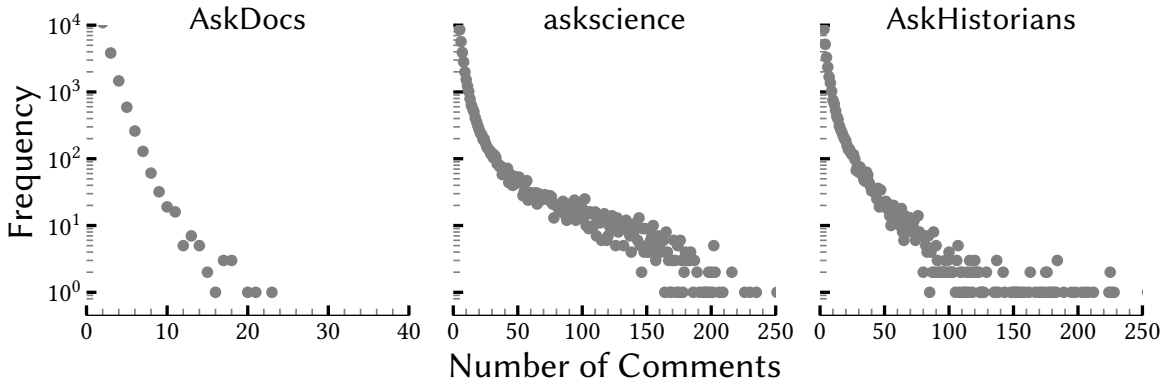


Figure 4.3: Frequency plot (log scale) of number of comments per post for three subreddits. A post on AskDocs tend to have fewer comments than the other two communities.

contact information. We denote verified users with *author flairs* as experts in the rest of the paper. Table 4.3 presents frequent submission and author flairs for all the subreddits. AskDocs does not have submission flairs as it is a smaller community. Figure 4.2 shows the distribution of the authors with the number of unique categories of the posts where the user commented. For both subreddits, we observed that around 80% of the users comment on posts from more than two categories. This shows that users engage with a variety of topics in Reddit.

Experts are highly active in the community, they answer around 60-70% of the posts (Table 4.2). askscience and AskHistorians have significantly higher and more detailed comments ($|w_{m,n}|$ in Table 4.2) per post than AskDocs, as shown in Figure 4.3. Due to the prevalence of a large number of comments, manual curation is very expensive, thus necessitating the need for an automatic tool to infer trustworthiness of the comments.

Sub-Reddit	Author Flairs	Submission Flairs
*Docs	Physician, Medical Student, Registered Nurse, Physician Assistant, Pharmacist, Nursing Student, Pharmacy Student, EMT, Doctor, Moderator, M.D., Nurse Practitioner	NA
*Science	Biochemistry, Molecular Biology, Immunology, Genetics, Microbiology, Neuroscience, Bioinformatics, Astrophysics, Biophysics, Cell Biology, Materials Science, Cosmology	Physics, Biology, Astronomy, Chemistry, Engineering, Medicine, Earth Sciences, Mathematics, Neuroscience, Computing, Human Body, Planetary Sci.
*Historians	Early Christianity, Ancient Near East, Andean Archaeology, Interesting Inquirer, Colonialism, Mesoamerican Archaeology, New Spain, American Civil War, Holocaust, Medieval Europe, Early Modern Europe, Environment	AMA, Urbanism, Fashion, Myth, Floating, Africa, Literature, Best Of, Pop Music, Home, Death, South America, Trade

Table 4.3: Top-K most frequent author and submission flairs that appear in our dataset, these flairs are used only in evaluation as our model is unsupervised.

4.3.2 Experimental Setup

We evaluate latent trustworthy comment learned by our model on a trustworthy comment ranking task. That is, given a submission post, our goal is to rank the posted comment based on their trustworthiness. For this experiment, we treat expert users' comment as the most trustworthy comment of the post.[6] Besides, we also report results using the highest upvoted comment as the gold standard. Highest upvoted comments represent community consensus on the most trustworthy response for the post [141]. While it is shown that there is widespread under-provision on Reddit, and thus, it is possible to miss high-quality content that is not highly voted; nevertheless, upvotes is a good proxy for community consensus [59].

In particular, we rank comments for each post m, in the order of descending cosine similarity between their embedding, $\boldsymbol{a}_{m,n}$, and the latent trustworthy comment embeddings, $\boldsymbol{a}^*_m$. We then report average Precison@k values over all the posts, where k denotes the position in the output ranked list of comments.

Baselines: We compare our model with state-of-the-art truth discovery methods proposed for continuous and text data and non-aspect version of our model. Note that there is no label information used, so we cannot compare to other supervised CQA models [137, 56, 63]

[6]While human judgment would be the most precise; it is also the most challenging to collect. For instance, in askscience we would need experts in over 35 science fields, reading up to 250 comments for a single post.

which need this supervision. Our *unsupervised model* is complementary to these approaches, and thus, a rigorous comparison is impossible.

Mean Bag of Answers (MBoA): In this baseline, we represent the trustworthy comment for a post as the mean comment embedding. This baseline assumes uniform user reliability.

CRH: is a state-of-the-art truth discovery-based model for heterogeneous data, i.e. categorical and numerical data [142]. CRH minimizes the weighted deviation of the trustworthy comment embedding from the individual comment embeddings with user reliabilities providing the weights.

CATD: is an extension of CRH that learns a confidence interval over user reliabilities to handle data skewness [143]. For both the above models, we represent each comment as the average word embeddings of its constituent terms.

TrustAnswer: Li et al. [76] modeled semantic similarity between comments by representing each comment with embeddings of its key phrase.

CrowdQM-no-aspect: In this baseline, we condense the commenter's aspect reliabilities to a single r_n and simplify the *user-post reliability* score computation. The updated optimization equation is,

$$\min_{\{a^*_m\},\{v_\omega\},\{r_n\}} \sum_{n=1}^{N} r_n \left(\sum_{m \in \mathcal{M}_n} s(\boldsymbol{u}_n, \boldsymbol{p}_m) \left(E_{m,n} + \beta \odot Q_{m,n} \right) \right)$$
$$\text{s.t.} \sum_{n=1}^{N} e^{-r_n} = 1$$

This model acts as a control to gauge the performance of our proposed model.

We do not compare with other truth discovery methods [133, 134, 144, 135, 136] as CRH and CATD are already shown to outperform them.

4.3.3 Results

Table 4.4 reports the Precision@1 results using expert's comments as gold standard. MBoA, with uniform source reliability, outperforms the CRH method that estimates reliability for each user separately. Thus, simple mean embeddings provide a robust representation for the trustworthy comment.

We also observe that CrowdQM-no-aspect performs consistently better than TrustAnswer. Note that both approaches do not model aspect-level user reliability but use semantic representations of comments. However, while TrustAnswer assigns a single reliability score for each comment, CrowdQM-no-aspect additionally takes into account the user's familiarity

Model	*Docs	*Science	*Historians
MBoA	0.592	0.633	0.602
CRH	0.585	0.597	0.556
CATD	**0.635**	0.700	0.669
TrustAnswer	0.501	0.657	0.637
CrowdQM-no-aspect	0.509	0.666	0.640
CrowdQM	0.617	**0.734**	**0.753**

Table 4.4: Precision@1 for all three Ask* subreddits, where the experts' comments are treated as the trustworthy comment.

with the post's context (*similarity* function, $s(.)$) to compute her reliability for the post. Finally, CrowdQM consistently outperforms both the models, indicating that aspect modeling is beneficial.

CATD uses a confidence-aware approach to handle data skewness and performs the best among the baselines. This skewness is especially helpful in Reddit as experts are the most active users (Table 4.2); and, CATD likely assigns them high reliability. Our model achieves competitive precision as CATD for AskDocs while outperforming for the others. This indicates that our data-driven model works better for communities which are less sparse (Section 4.3.1 and Figure 4.3).

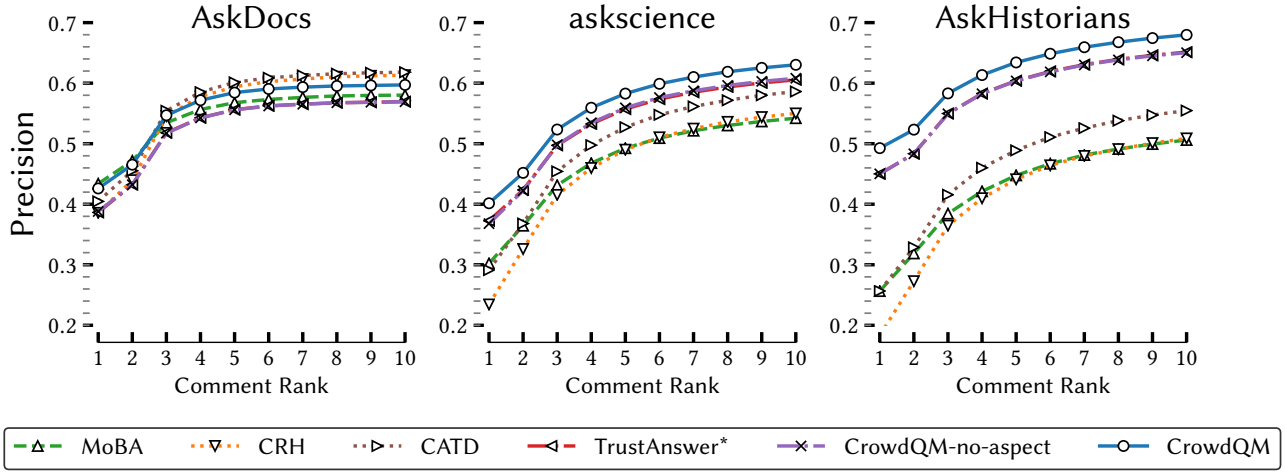


Figure 4.4: Precision of our model vs. comment rank computed by user's upvotes. Our model outperforms the baselines for askscience and AskHistorians while performs similarly for AskDocs.

Table 4.5 reports Precision@1 results using community upvoted comments as the gold standard, while Figure 4.4 plots the precision values against the size of the output ranked comment list. In general, there is a drop in performance for all models on this metric because

Model	*Docs	*Science	*Historians
MBoA	**0.434**	0.302	0.257
CRH	0.386	0.234	0.183
CATD	0.405	0.291	0.257
TrustAnswer	0.386	0.373	0.449
CrowdQM-no-aspect	0.388	0.368	0.450
CrowdQM	0.426	**0.402**	**0.493**

Table 4.5: Precision@1 for all three Ask* subreddits, where the highest upvoted comment is treated as the most trustworthy comment.

it is harder to predict upvotes as they are inherently noisy [59].

TrustAnswer and CrowdQM-no-aspect perform best among the baselines indicating that modeling semantic representation is essential for forums. CrowdQM again consistently outperforms the non-aspect based models verifying that aspect modeling is needed to identify trustworthy comments in forums.

CrowdQM remains competitive in the smaller AskDocs dataset, where the best performing model is MoBA. Thus, for AskDocs, the comment summarizing all other comments tends to get the highest votes.

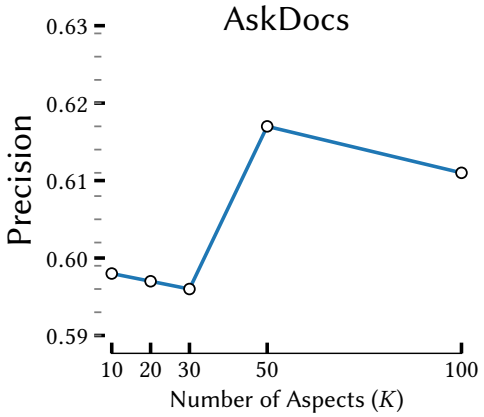
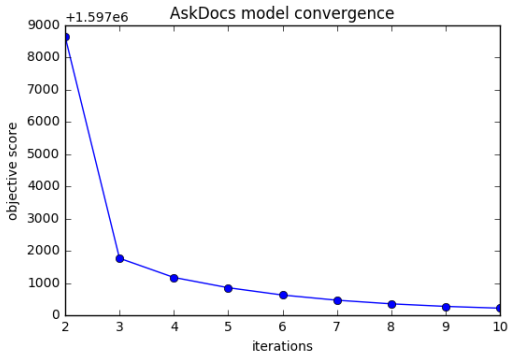


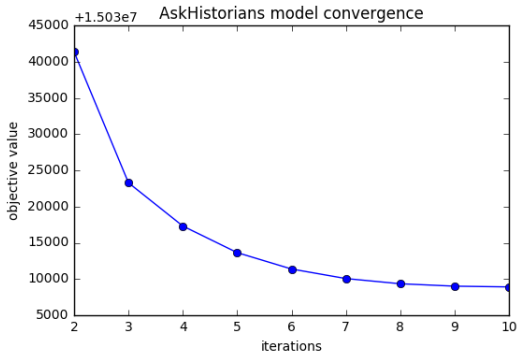
Figure 4.5: Precision of our model with changing number of aspects. Value of K does not have much impact on the precision value.

Parameter Sensitivity In Figure 4.5, we plot our model's precision with varying number of aspects. Although there is an optimal range around 50 aspects, the precision remains relatively stable indicating that our model is not sensitive to number of aspects. We also observed similar results for the other datasets and omitted those figures for lack of space. We also did similar analysis with β and did not find any significant changes to the Precision.

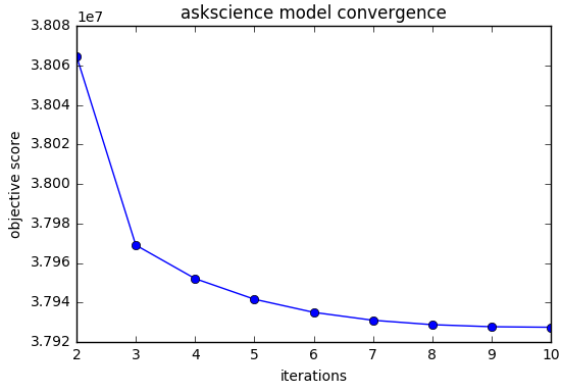
4.3.4 Model Convergence



(a) AskDocs



(b) AskHistorians



(c) askscience

Figure 4.6: CrowdQM model convergence for AskDocs, AskHistorians, and askscience respectively.

In Figure 4.6, we plot the objective function score at each iteration, for our model CrowdQM for the three subreddits. On all three subreddits, the model converges after only about six iterations indicating our model is quick to approximate a solution. In general, the computational complexity is $O(|\mathcal{V}|NM)$ for a single iteration. However, our implementation leverages the data sparsity in the comment-word usage and user-submissions posts to make the model efficient.

4.4 DISCUSSION

In this section, we report qualitative analysis of user-aspect reliabilities $\{r_n\}$ and word embeddings $\{v_w\}$ learned by our proposed CrowdQM model. For brevity, we focus our analysis on our largest subreddit, askscience.

4.4.1 Aspect Reliability Analysis

Post Category: Computing
Embedded Systems, Software Engineering , Robotics
Computer Science
Quantum Optics, Singular Optics
Robotics, Machine Learning, Computer Vision, Manipulators
Computer Science
High Performance Computing, Network Modeling and Simulation
Biomechanical Engineering, Biomaterials
Machine Learning, Deep Architectures, Scientific Computing
Machine Learning, Deep Architectures, Scientific Computing
Programming Languages, Computer Security
Post Category: Archaeology
Archaeology, Maya Stone Tools, Geoscience
Global Health, Tropical Medicine
Control, Robotics Engineering, Industrial Robotics
Archaeology, Collapse of Complex Societies
Archaeology, Archaeometallurgy
Criminal Justice
Computational and Evolutionary Archaeology
Evolutionary Biology, Plant-Herbivore Systems
Computational and Evolutionary Archaeology
Archaeology, Collapse of Complex Societies
Post Category: Biology
Animal Cognition
Cell and Developmental Biology
Biochemistry, Molecular Biology, Enzymology
Genetics, Cell biology, Bioengineering
Computational Physics, Biological Physics
Aquatic Ecology and Evolution, Active Acoustics
Genomic Instability, Cancer Development
Biochemistry, Genomics, Proteomics, Mass Spectrometry
Neuroscience, Psychopharmacology
Genetics and Genomic Sciences
Continued on next page

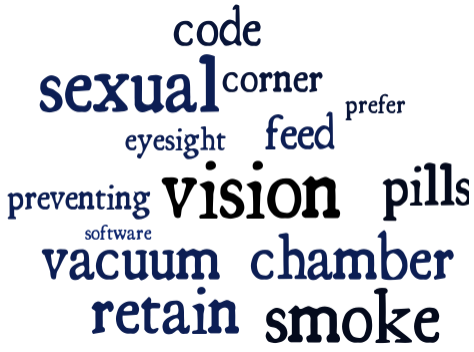
Table 4.6 – continued from previous page
Post Category:Linguistics
Linguistics, Hispanic Sociolinguistics
Comparative Political Behaviour
Historical Linguistics, Language Documentation
Linguistics, Hispanic Sociolinguistics
Historical Linguistics, Language Documentation
Cognitive Modeling
Nanostructured Materials, Heterogeneous Catalysis
Auditory Science
Cognitive Modeling
Linguistics, Phonetics and Phonology, Sound Change
Post Category: Medicine
Infectious Diseases, Pulmonary Immunology
Biomedical Engineeering, Biomechanics, Biomaterials
Pediatric Neurology
Anesthesiology, Post-Operative Pain, Traumatic Brain Injuries
Molecular Biology, Musculoskeletal Research
Immunology, Immune Regulation, Infectious Diseases
Molecular Biochemistry, DNA Damage Repair
Virology, Molecular Biology, Orthopoxviruses
Veterinary Medicine, Canine Lymphoma
Bioengineering, Cardiovascular Imaging
Post Category: Psychology
Clinical Psychology, Psychotherapy, Behavior Analysis
International Relations, Comparative Politics
Neuropsychology
Psychology, PTSD, Trauma, and Resilience
Cognitive Neuroscience, Neuroimaging, fMRI
Psychology, Legal psychology, Eyewitness testimonies
Experimental Psychology, Social Cognition and Statistics
Clinical Psychology
Clinical Psychology, Sleep, Insomnia
Visual Cognition, Cognitive Neuroscience

Table 4.6: Most reliable author flairs with their corresponding post categories according to user-post reliability score, $R_{m,n}$.

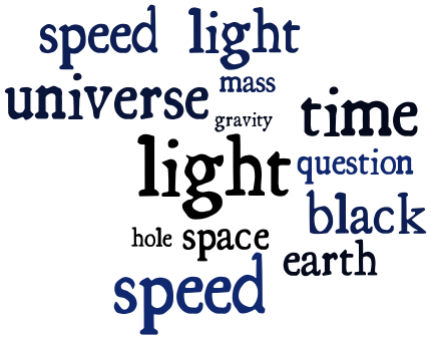
We evaluate learned user reliabilities through users commenting on a post with a *submission flair*. Note that a submission flair is manually curated and denotes post's category, and this information is not used in our model. Specifically, for each post m, we compute the *user-post reliability* score, $R_{m,n}$, for every user n who commented on the post. We then

ranked these scores for each category and report top-10 *author flairs* for few categories in Table 4.6.

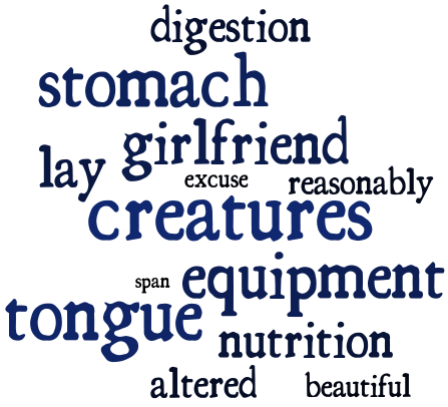
The top-performing *author flairs* for each category are experts for that domain. For instance, for the Computing category highly reliable users have author flairs like Software Engineering and Machine Learning, while for Linguistics authors with flairs Hispanic Sociolinguistics and Language Documentation rank high. These results align with our hypothesis that in-domain experts should have higher reliabilities. We also observe that few out of domain authors being ranked high, such as, authors with flairs like Comparative Political Behavior and Nanostructured Materials in the Linguistic category. This diversity could be due to the interdisciplinary nature of that domain. Our model, thus, can be used by the moderators of the discussion forum to identify and recommend potential reliable users to respond to new submission posts of a particular category.



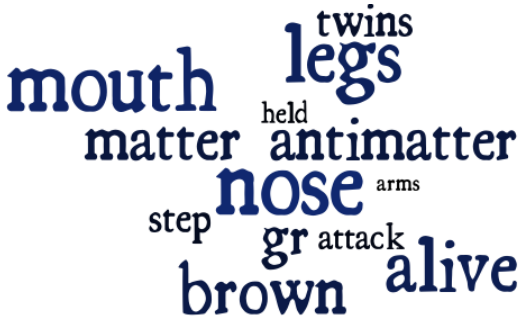
(a) Health



(b) Cosmos



(c) Diabetes



(d) Oceanography

Figure 4.7: Top words for highly correlated aspects between user reliability and user karma.

To further analyze the user reliability, we qualitatively examine the aspects with the largest reliability value of highly upvoted users in a post category. First, we identify users deemed reliable by the community for a category through a *karma* score. Category-specific user

karma is given by the average upvotes the user's comments have received in the category. We then correlate the category-specific user *karma* with her reliability score in each $k \in K$ aspect, $\boldsymbol{r}_n^{(k)}$ to identify aspects relevant for that category. Figure 4.7 shows the top words of the highest correlated aspects for some categories. The identified words are topically relevant thus our model associates aspect level user reliability coherently. Interestingly, the aspects themselves tend to encompass several themes, for example, in the Health category, the themes are software and health. Or in the Oceanography category, the themes are around animal's physiology and matter.

4.4.2 Word Embedding Analysis

Liquid		**Cancer**		**Quantum**	
word2vec	CrowdQM	word2vec	CrowdQM	word2vec	CrowdQM
unimaginably	gas	mg	disease	search results	model
bigger so	chemical	curie	white	sis	energy
two lenses	solid	wobbly	cell	shallower water	particle
orbiting around	air	subject	food	starts rolling	mechanics
fire itself	material	"yes" then	complete	antimatter galaxies	mathematical

Life		**Planet**	
word2vec	CrowdQM	word2vec	CrowdQM
molaison	species	esther	earth
around	natural	missing leg	star
machos	nature	chimps	plane
brain	production	while drinking	land
"dark" matter	size	living off	building

Table 4.7: Similar words identified using embeddings learned by CrowdQM, and initial word2vec, for the askscience subreddit. The left and right columns correspond to word2vec and CrowdQM models respectively.

The CrowdQM model updates word embeddings to better model semantic meaning of the comments. To evaluate the embeddings, we first identify the most representative aspect for each category. We denote the highest weighted aspect from the mean aspect distribution $\{\boldsymbol{p}_m\}$ of all posts belonging to a category as the most representative aspect of that category. Then, we extract frequent terms of that aspect and find its most similar keywords using cosine distance between the learned word embeddings.

The left column for each term in Table 4.7 are the most similar terms returned by the

initial embeddings from word2vec model while the right column reports the results from updated embeddings $\{\boldsymbol{v}_\omega\}$ from our CrowdQM model. We observe that there is a lot of noise in words returned by word2vec model as they are just co-occurrence based while words returned by our model are semantically similar and describe similar concepts. This improvement is because our model updates word embeddings in a trust aware manner such that they are similar to terms used by responses from reliable users.

4.5 CONCLUSION

We proposed an unsupervised model to learn a trustworthy comment embedding from all the given comments for each post in a discussion forum. The learned embedding can be further used to rank the comments for that post. We explored Reddit, a novel community discussion forum dataset for this task. Reddit is particularly challenging as posts typically receive a large number of responses from a diverse set of users and each user engages in a wide range of topics. Our model estimates aspect-level user reliability and semantic representation of each comment simultaneously in a unified optimization framework. Experiments show that modeling aspect-level user reliability improves the prediction performance compared to the non-aspect version of our model. We also show that the estimated user-post reliability can be further used to identify trustworthy users for particular post categories.

Future work includes exploring other validation methods for trustworthiness including manual annotations of the comments. Another direction is to experiment with more sophisticated methods to generate comments' semantic composition, such as recurrent neural networks. Adding learned users and comments representations to a supervised learning framework for improving them further is another interesting future research direction.

CHAPTER 5: MODELING SOCIAL INFLUENCE ON USER BEHAVIOR

In this chapter, we model the influence of the user's social connections on their behavior. Recommender systems are a perfect example of an online platform where user preferences exhibit strong user homophily with their friends. In this work, we exploit homophily in the user and item space. Besides, user's history itself influences their preferences. We propose separate modules to capture these different factors and later fuse them to predict future items accurately for platform users [145].

5.1 OVERVIEW

Recommender systems are ubiquitous and model user preferences on commercial and social websites as well as in apps. These systems predict with reasonable accuracy, products that we may be interested in, people which we may know, or songs and movies that we may appreciate. This success builds upon a long history of research. However, to this day, a large active community continues to improve recommender systems as many questions remain open, e.g., How to effectively model and merge multiple factors influencing user preferences like (1) temporal context, (2) social influence, and (3) similarity between items? We explore this question in detail in this chapter.

Classic collaborative filtering is one of the most successful approaches to model user preferences. It learns a low dimensional and often linear latent factor model for both users and items via matrix factorization of the user-item interaction matrix [22]. With deep learning taking a more prominent role, more complex models have been applied to learn increasingly non-linear relationships [24, 25]. However, those classical methods ignore all three of the factors above. Hence, many techniques have been developed, which augment classical recommender systems with one of those factors.

First, considering temporal context removes the assumption of a static interaction matrix, which generally doesn't hold as user preferences evolve with time. Thus, history from a distant past is not necessarily relevant to current preferences. To this end, Markov chains [27] and recently, Convolution Neural Network (CNN) [30] and Recurrent Neural Network (RNN) [32] based methods have been proposed to model this temporal dependence in recommender systems. Those methods remove the static interaction matrix of classical collaborative filtering and learn a user's and an item's hidden representation based on their recent interactions.

Second, considering social influence removes the restriction that users operate in isola-

tion. This idea is popularized by the social influence theory [8], which argues that a user's preference is influenced by their friends' behavior, leading to user homophily (similar user preferences). It is noteworthy that these influences are inherently dynamic as friends' preferences are evolving too (socio-temporal influence), a fact mostly ignored by current systems. For instance, recent works model static social effect [37, 146, 36, 109]. These methods look at the entire history of the user's friends instead of emphasizing the most recent actions. Moreover, these approaches assume uniform importance of all friends. While this is not suitable in general, it is an important first step to understand social influence for recommendation.

Third, exploiting similarities between items (based on co-occurrence or similar features) alleviates the data sparsity issue (many items with few ratings). Similar items hold similar attractiveness to users, leading to item homophily. Deep net based recommender models are prone to skew prediction results towards popular items with ample feedback in the training data (overfit to popular items) [147]. This is counterproductive to user experience as it leads to similar recommendations across users. Also, compared to highly frequented items, long-tail items (items with fewer ratings) result in higher profit margins for the platforms [148]. Item-Item collaborative filtering based methods [23] integrate these similarities but ignore the user's history; thus providing generic recommendations.

To make these three points concrete, let us consider the example shown in Figure 5.1. Alice is using an online social movie viewing platform. She is currently hooked onto superhero movies (temporal). While deciding which movie to watch next, she will be influenced by recent superhero movies watched by her friends on the platform (socio-temporal influence). She could also decide to watch other superhero movies in the platform not seen by her social circle yet (item-to-item similarity).

As illustrated earlier, a user's behavior is affected by at least the aforementioned three factors (others could include time-of-day, mood, etc.). However, the relative importance of these factors still remains unclear. It is indeed challenging to model these factors effectively and efficiently in a unified model as these cues mutually influence each other. This is emphasized by the fact that existing work often studies only a subset of those three cues.

To address this concern, we develop a 'Fusion Recommender' (FuseRec) model to jointly and efficiently capture all the factors in a unified model. It takes into account a user's temporal changes along with homophily in the user and item space. It treats each of the signals equally and combines them in an interpretable manner.

More specifically, we use three different modules to model each factor: (1) a user-temporal, (2) a user-social, and (3) an item-similarity module. To model the temporal behavior of a user's item viewing history, we use widely adopted recurrent neural nets. These networks are shown to capture complex and non-linear relationships of time-varying sequences. To

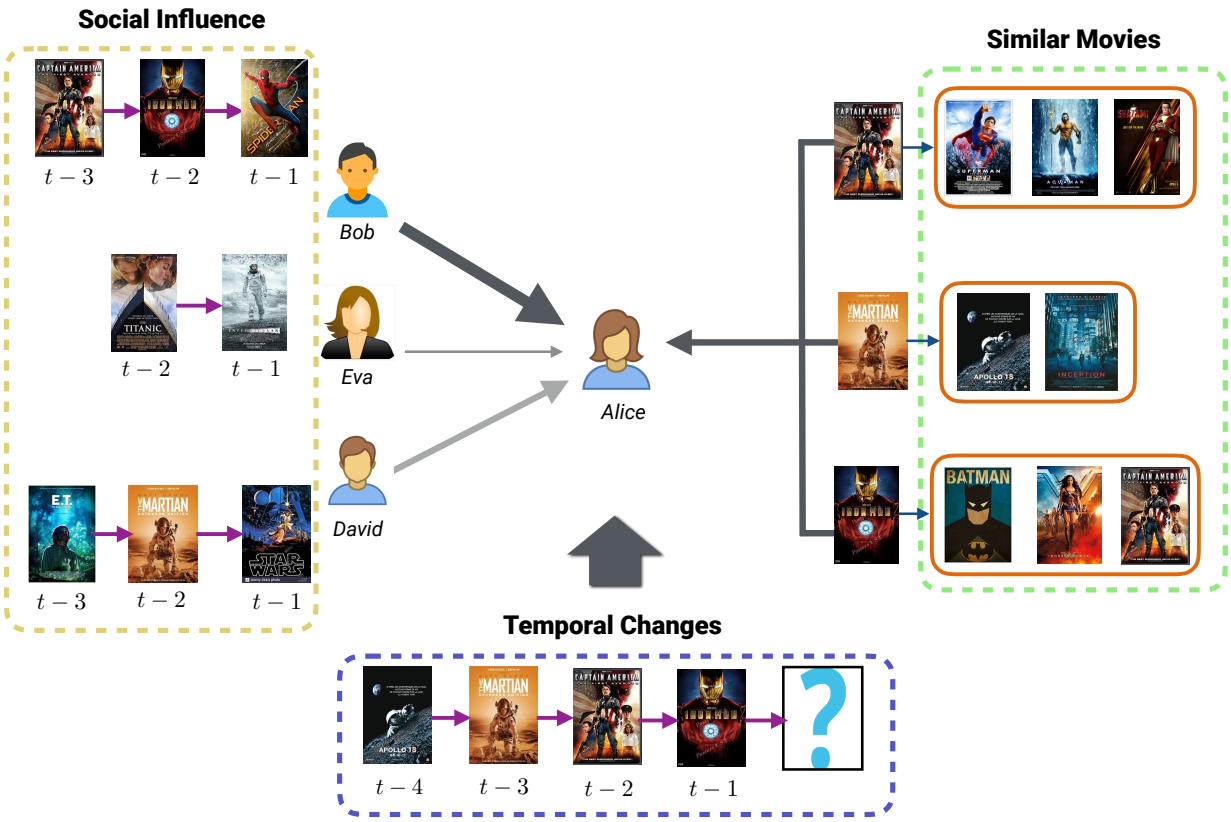


Figure 5.1: Illustrative diagram of factors affecting Alice's decision on which movie to watch next in an online social movie viewing platform. Her current interests are towards superhero movies (temporal); she could either decide to watch recent superhero movies seen by her friends (social) or other superhero movies not watched by her friends (similar items).

capture the effect of a user's friends' recent history, we develop a novel attention based social aggregation model. Different from existing works, it aggregates a user's friends' recent item history in a weighted manner. We learn attention weights separately for each pair of a user and her friend. For item-item similarity aggregation, we construct a 'social graph of items' based on similarity in item features and co-occurrence in the dataset. We develop a novel attention based aggregation model for the item similarity graph too. In contrast to existing work, we learn an attention weight for each similar item and later aggregate information of neighboring items in a weighted manner.

To provide an understanding of the importance of the three factors, we choose to linearly combine them via learned weights. The magnitude of the learned weights permits a glimpse at the importance of the individual modules.

We evaluate our model on three representative benchmark datasets: Ciao, CiaoDVD, and Epinions. We compare to an array of collaborative filtering, temporal, and social methods

and achieve a significant improvement of more than 14% for AUC on the Ciao and the Epinions dataset and around 2% for the CiaoDVD dataset. In addition, we provide a study on the importance of the three factors. Across all datasets, we find the temporal module to be the most significant factor for modeling user preferences.

In summary, our main contributions are as follows:

- We propose 'FuseRec,' a Fusion Recommender model which combines temporal, social, and item similarity aspects in an interpretable manner.
- We propose a novel attention based aggregation model to capture homophily in the user and item space.
- We evaluate our method on three benchmark datasets and compare to a variety of recent temporal, social, and socio-temporal models.
- We provide a detailed study regarding the importance of different factors used in our model.

5.2 PROPOSED METHOD

We first provide an overview of the proposed FuseRec approach to model user and item homophily via attention based nets while also modeling temporal relations. We subsequently discuss the details of each employed module.

5.2.1 Overall Architecture

Our goal is to create a ranked list of items, indicating the preference of a user u for interacting with item $i \in \mathcal{I}$ at time t. Here, $\mathcal{I}$ represents the set of all items available on the considered platform. We compute this ranked list by sorting probability scores $\hat{r}^t_{u,i}$ for a user u and item i at time t. Formally, given user u and time t, we obtain $\forall i$, the probability scores after scaling the output of a linear layer, i.e.,

$$\hat{r}^t_{u,i} = \sigma\left(\lambda_1 S_1 + \lambda_2 S_2 + \lambda_3 S_3 + \lambda_4 S_4 + b_c\right). \tag{5.1}$$

Hereby σ denotes the sigmoid function, b_c is a learnable bias, and $\lambda_k \in \mathbb{R}$, $k \in \{1, \ldots, 4\}$, are four learnable weights for the scores S_k. Importantly, because we learn a linear combination of scores, we are able to study their magnitude which provides evidence regarding

the importance of the different factors in the proposed FuseRec model. See Section 5.3.5 for our experimental results.

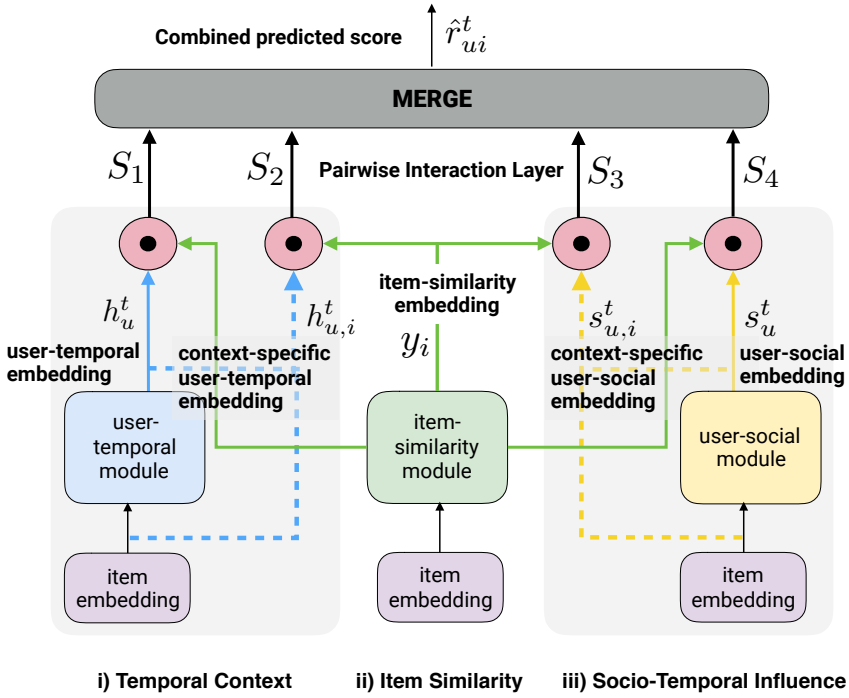


Figure 5.2: Overview of the proposed FuseRec model. Our model computes pairwise interaction scores which compares item embeddings from the item-similarity module with user embeddings from both the user-temporal module and the user-social module. These scores are then merged using learnable weights to compute the final predicted score.

As illustrated in Figure 5.2, we obtain the scores S_k by combining information from the following three modules: (1) the user-temporal module which leverages temporal information about a user; (2) the user-social module which captures information about the recent interactions of a user's friends; and (3) the item-similarity module, which captures information about item homophily.

Specifically, Eq. (5.1) combines four pairwise interactions: (1) $S_1 = h^t_u \cdot y_i$, (2) $S_2 = h^t_{u,i} \cdot y_i$, (3) $S_3 = s^t_u \cdot y_i$, and (4) $S_4 = s^t_{u,i} \cdot y_i$. Hereby, $\cdot$ indicates an inner product between two embeddings. Note, all pairwise interaction scores S_k assess the similarity between a D-dimensional representation obtained from the item-similarity module ($y_i \in \mathbb{R}^D$) and information obtained from either the user-temporal module ($h^t_u, h^t_{u,i} \in \mathbb{R}^D$) or the user-social module ($s^t_u, s^t_{u,i} \in \mathbb{R}^D$).

The user-temporal module encapsulates information from a history of user interactions. This module computes a time-dependent embedding h^t_u for user $u \in \mathcal{U}$ at time t. This embedding denotes a user's current preferences in general. We also compute a context-

specific user-temporal embedding, $h^t_{u,i}$ which encodes similarity of a user's current preferences (h^t_u) in the context of the candidate item $i \in \mathcal{I}$. Note that both embeddings capture different aspects (general and item specific) and are time-dependent.

The user-social module captures a user's social preferences based on the recent history of the user's social graph. Specifically, for user u at time t, we encode this information in an embedding referred to as user-social embedding, s^t_u. Similar to the user-temporal module, we also compute a context-specific user-social embedding for a user, $s^t_{u,i}$, encoding similarity of a user's social preferences (s^t_u) with respect to the candidate item i.

The item-similarity module employs item-item collaborative filtering, building an implicit similarity network between items based on their features and co-occurrence in the dataset. This module computes an item-similarity embedding y_i for item $i \in \mathcal{I}$, which is identical across time. We think this is a reasonable assumption as properties of items do not change over time[1].

We will next provide details about computation of the user-temporal embedding h^t_u, the context-specific user-temporal embedding $h^t_{u,i}$, a user-social embedding s^t_u, the context-specific user-social embedding $s^t_{u,i}$, and the item-similarity embedding y_i.

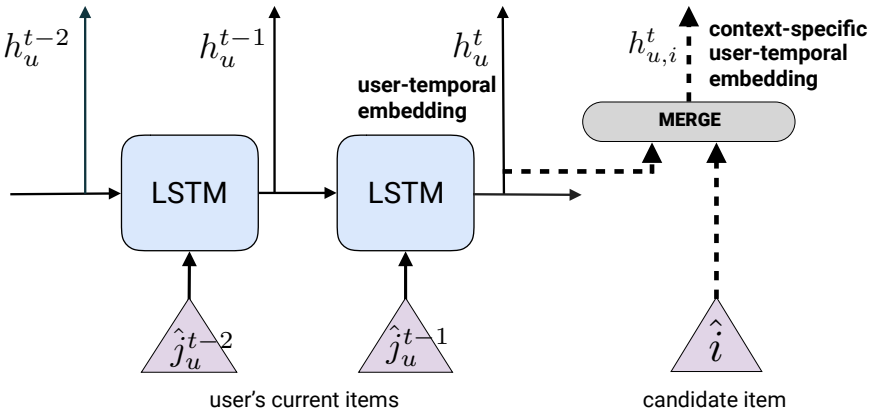


Figure 5.3: The user-temporal module uses an LSTM to compute an embedding h^t_u based on a user's history h^{t-1}_u and the current item $\hat{j}^{t-1}_u$. We also compute context-specific user-temporal embedding $h^t_{u,i}$ with respect to candidate item i.

5.2.2 User-Temporal Module

Users constantly interact with items offered on online platforms, e.g., users rate or watch movies. Importantly, a user's preference does not remain constant and changes over time.

[1]Experiments with time-sensitive item embeddings decreased accuracy of the reported results.

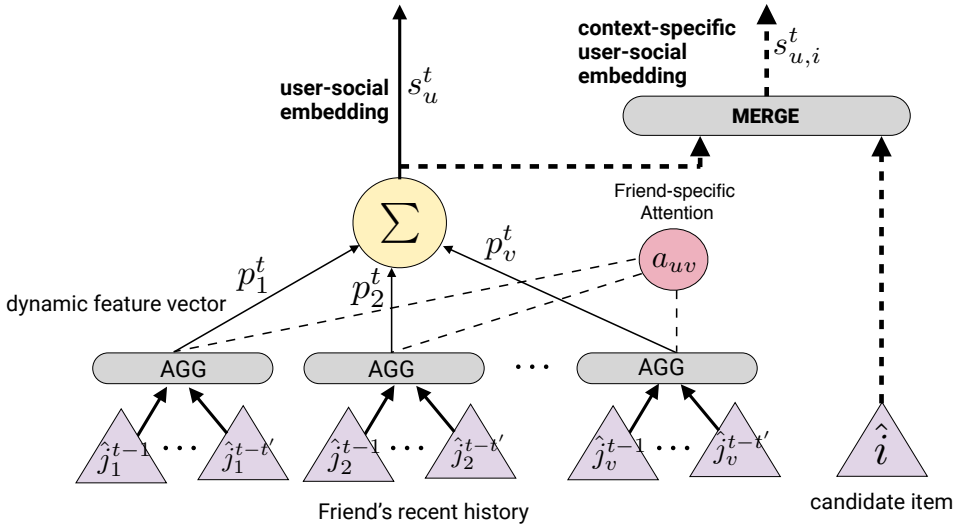


Figure 5.4: Illustration of the user-social module. The user-social module uses an attention based aggregation of the recent history of a user's social connections.

To model temporal dynamics, classical methods have explored Markov chains, attention networks and convolution networks [26, 30, 31]. However, these methods assume dependence on only recent history and thus do not capture long term dependencies within user-item preferences. To address this concern, we use recurrent neural nets (RNNs) based on long-short-term-memory (LSTM) components. Those are widely used in natural language processing to capture sequential dynamics [149].

In general, RNNs are based on the following recurrence relation, where h^t represents the hidden vector at time t, x^t is the input at time t and w refers to learnable weights:

$$h^t = f(h^{t-1}, x^{t-1}, w).$$

To specialize to our case, consider again a platform where users watch movies. Formally, for each user u at time $t-1$, let $j_u^{t-1} \in \mathcal{I}$ be the item which user u interacted with at time $t-1$. To compute its item embedding $\hat{j}_u^{t-1} \in \mathbb{R}^D$ (throughout we use '$\hat{\cdot}$' to indicate embeddings), we concatenate the item j_u^{t-1} with its one-hot category information $c(j_u^{t-1}) \in \{0,1\}^{|C|}$ and apply the linear transformation

$$\hat{j}_u^{t-1} = W_p[j_u^{t-1} \,||\, c(j_u^{t-1})] + b_p. \tag{5.2}$$

With slight abuse of notation j_u^{t-1} also denotes the one-hot representation. Here, W_p and b_p are trainable parameters and represent weight and bias of a linear layer, '||' represents the concatenation operation and C is the total number of item categories. This item embedding

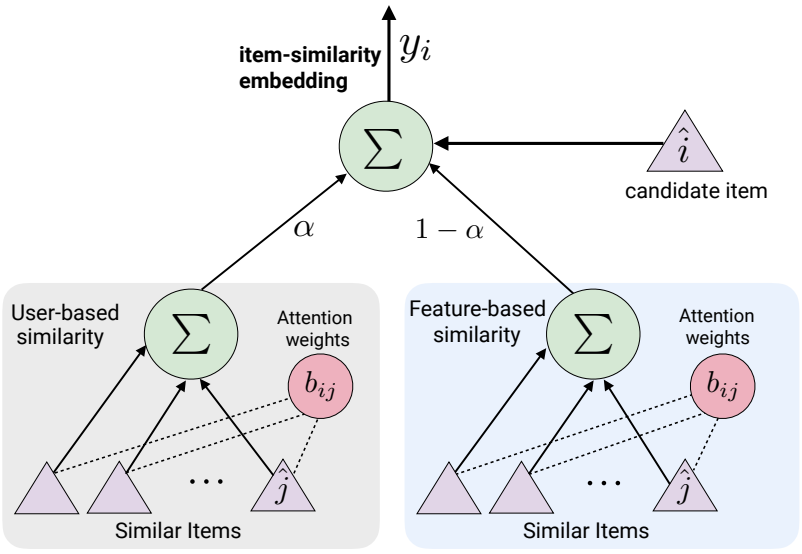


Figure 5.5: Illustration of the item-similarity module. The item-similarity module aggregates embeddings from similar items in an attention aware manner. The item graphs are constructed based on frequency of co-occurrence and feature similarity. The parameter α trades influence between both item social graphs.

$\hat{j}_u^{t-1}$ is used as an input for an LSTM module based RNN which computes the user-temporal embedding h_u^t via

$$h_u^t = f'(h_u^{t-1}, \hat{j}_u^{t-1}, w). \tag{5.3}$$

Here, f' represents the LSTM recurrence relation. This final user-temporal representation encodes a user's past behavior.

While h_u^t captures a user's current preferences in general, we also separately capture relevance of candidate item i with respect to the user's current preferences. For instance, if the user is currently watching action movies, we should capture if the candidate action movie i matches her preferences. Thus, context-specific user-temporal embedding $h_{u,i}^t$ encodes similarity between user-temporal embedding h_u^t and the candidate item embedding $\hat{i}$. Specifically, to capture similarity between the user-temporal embedding and the item embedding, we compute $h_{u,i}^t$ as

$$h_{u,i}^t = W_q[h_u^t \,||\, \hat{i} \,||\, h_u^t \otimes \hat{i}] + b_q, \tag{5.4}$$

where $\hat{i}$ is the embedding of candidate item i and $\otimes$ represents an element-wise product of two vectors. W_q and b_q are learnable parameters. Figure 5.3 depicts the architecture of this user-temporal module.

5.2.3 User-Social Module

Beyond temporal changes, users are influenced by recent behavior or ratings of trusted friends. Also, influences are not equal among all friends. To model this heterogeneous social influence, we use an attention based aggregation of a user's friends recent past behavior. Section 5.2.1 shows our user-social module. Formally, for user u at time t, the user-social embedding $s_u^t \in \mathbb{R}^D$ is computed as

$$s_u^t = \sum_{v \in F(u)} a_{uv} p_v^t, \tag{5.5}$$

where $F(u)$ represents social connections of user u, $a_{uv} \in \mathbb{R}$ is the attention weight for friend v and $p_v^t \in \mathbb{R}^D$ is a feature vector representing the recent history of friend v.

Most of the social recommender systems [37, 36, 150] employ a uniform weighting for all social friends when computing influence. However, we argue that this is sub-optimal, particularly for social media, where a user does not trust all friends equally. Indeed, we think modeling of trust is particularly important for online platforms due to large social circles with superficial acquaintance. Thus, we obtain the attention weights a_{uv} from an influence score $e(u, v)$ for each user u and friend v:

$$e(u, v) = \text{LeakyReLU}(W_q[\hat{u} \mid\mid \hat{v}] + b_q), \tag{5.6}$$

where $\hat{u}, \hat{v} \in \mathbb{R}^D$ are user embeddings for user u and v respectively, while W_q and b_q are learnable parameters. The attention weight a_{uv} is obtained by normalizing the influence score via a soft-max:

$$a_{uv} = \frac{\exp(e(u, v))}{\sum_{v \in F(u)} \exp(e(u, v))}. \tag{5.7}$$

Each friend v is represented by a dynamic feature vector p_v^t which captures recent past behavior and is computed via

$$\hat{j}_v^{t'} = W_r[j_v^{t'} \mid\mid c(j_v^{t'})] + b_r, \tag{5.8}$$

$$p_v^t = \text{AGG}(\{\hat{j}_v^{t'} \mid t' < t\}). \tag{5.9}$$

Here $\hat{j}_v^{t'} \in \mathbb{R}^D$ is the item embedding for item $j_v^{t'}$ clicked by user v at time t'. Each past item rated by the friend before the current timestamp t can be used to compute the historical profile of friend v. In practice, we found that using the recent past gives similar performance compared to using all previous items. Therefore, we consider only the last t' items of each friend. We aggregate these historical item embeddings using the mean aggregation operation

AGG. Note that it is possible to aggregate this information in multiple ways but mean aggregation performed well in our experiments. It is also worthwhile to note that attention weights remain static across time while a user's feature vectors are dynamic.

Similar to the user-temporal module, we also compute a context-specific user-social embedding $s^t_{u,i}$. This embedding captures similarity of the user's social preferences with respect to the candidate item i. Formally,

$$s^t_{u,i} = W_s[s^t_u \,||\, \hat{i} \,||\, s^t_u \otimes \hat{i}] + b_s, \tag{5.10}$$

where $\hat{i}$ is the item embedding of candidate item i and $\otimes$ is the element-wise product.

5.2.4 Item-Similarity Module

Online platforms operate with a large set of items, many of which are rated infrequently. It is consequently hard to construct meaningful representations of items, particularly if available information is scarce. Also, users are similarly attracted to related items but it is non-trivial to implicitly learn item-item similarity. To address this concern, we propose to construct a similarity aware item embedding based on information available for users which have interacted, e.g., clicked this item, and item features like category information. Section 5.2.1 illustrates our item-similarity module.

In particular, we represent each item $i \in \mathcal{I}$ via an n-hot vector $g_i \in \{0,1\}^{|\mathcal{U}|}$, where n is the number of users who have interacted with item i while $|\mathcal{U}|$ is the total number of users on the platform. We then compute k-nearest neighbors for each item using cosine similarity between these n-hot vectors. This results in an implicit social network for item i, denoted by $F(i)$. All these items are similar as the same users interacted with them. However, this approach of computing similarity between items is biased towards popular items with high user degree. Thus, we construct another item similarity network based on item features. In particular, we compute the network $F'(i)$ based on items which belong to the same category. We randomly connect k' items of the same category in $F'(i)$. In our experiments, we let $k = k'$, i.e., we use the same neighborhood size for both item networks.

We then aggregate both of these item graphs to learn item-similarity embeddings via

$$y_i = \alpha \sum_{j \in F(i)} b_{ij}\hat{j} + (1-\alpha) \sum_{j \in F'(i)} b_{ij}\hat{j} + \hat{i}, \tag{5.11}$$

where $\alpha \in [0,1]$ is a learnable parameter which controls the effect of co-occurring similarity versus category relationship. To compute the attention weights b_{ij}, we follow our earlier

approach: similar to the user model, $e(i,j)$ is the influence score for each item i and its similar item j:

$$e(i,j) = \text{LeakyReLU}\left(W_n[\hat{i} \,||\, \hat{j}] + b_n\right), \tag{5.12}$$

where $\hat{i}, \hat{j}$ are item embeddings for item i and j respectively, and W_n and b_n are learnable parameters. The final attention weight is obtained by normalizing the influence score via a soft-max function:

$$b_{ij} = \frac{\exp(e(i,j))}{\sum_{j \in F(i)} \exp(e(i,j))}. \tag{5.13}$$

Note that we use the same embeddings to compute attention weights for both graphs.

The final estimated 'item-similarity' module models similarity between items with similar features and frequently co-occurring items. This helps to address the data sparsity in the item space by exploiting item homophily.

Each module learns separates factors which influence user choices in recommender systems. We fuse these factors via a linear operation as detailed in Eq. (5.1).

5.2.5 Training

We train all the three modules jointly in a unified framework using the binary cross-entropy loss:

$$L_\theta = -\sum_{(u,i,t) \in \mathcal{B}} r^t_{ui} \log\left(\hat{r}^t_{ui}\right) + \left(1 - r^t_{ui}\right) \log\left(1 - \hat{r}^t_{ui}\right),$$

where θ represents all the learnable parameters in the model and $\mathcal{B}$ is the currently sampled mini-batch. Specifically, for each sample $(u,i,t) \in \mathcal{B}$, we also obtain user u's friends, $F(u)$ along with the corresponding item graph of the item i, i.e., $F(i)$ and $F'(i)$.

As we are dealing with implicit feedback, we don't have any negative samples. Thus, in each iteration of the training process, for every observed user-item interaction at time t, $(u,i,t) \in \mathcal{B}$, we sample m unobserved items for user u. Similar to Song et al. [39], we assign a weight of $1/m$ for each negative instance to provide a weak negative signal. This is done as each unobserved item does not necessarily mean that the user will not interact with this item in the future. For our experiments, we set $m = 5$.

Parameter Settings. For all three modules, we use an embedding size $D = 32$ for all user and item embeddings. We also initialize the embedding matrix using a Gaussian distribution with a mean of 0 and a standard deviation of 0.01. For the user-social and item-similarity modules, in each iteration, we subsample a set of friends F at each timestamp for a user (item) instead of considering all friends. This sampling has two benefits: (1) it avoids

overfitting by introducing random noise in the social module; and (2) it is computationally more tractable. We use a sample size of 10 for both user $F(u)$ and item $F(i), F'(i)$ social graphs. If a user has less than 10 friends, we pad the remainder with zeros. We set a friend's past history length, $t' = 3$ in the user-social module. We will study the effect of this and other hyper-parameters subsequently.

For the user-temporal and user-social module, we set the length of the historical sequence of user interactions to 30 for all users. For users with history length less than 30, we utilize all the available interactions. We also study the effect of this sequence length on our results in the next section. We use the Adam optimizer for training our model and perform a grid search over {0.1, 0.01, 0.001} for a suitable learning rate on the validation set. We report results on the test set for the best performance model on the validation set. We also use dropout along with gradient clipping with a value of 0.25 and L2 regularization on the user and item embeddings to avoid overfitting.

5.3 EXPERIMENTS

In this section, we first describe the datasets we use to evaluate the proposed approach, followed by the evaluation setup and competing state-of-the-art baselines. We then analyze the performance of our model, followed by ablation studies on different modules and hyper-parameters. Finally, we analyze the importance of each module using the learned weight parameters.

5.3.1 Datasets

We use benchmark datasets available from three online social review platforms: Ciao, Epinions, and CiaoDVD.

Ciao is a product review website where users provide reviews along with ratings for products ranging across a variety of categories. This dataset was crawled by Tang et al. [151] and contains rating information along with the creation timestamp given up to May 2011. Users also establish directed trust relations with other users on the platform and add them to their 'Circle of Trust.' Epinions is another popular online consumer review website like Ciao [151]. However, this dataset is longer spanning a decade from Aug. 1999 to Nov. 2013. This dataset also contains trust relations between users[2]. CiaoDVD is a movie review dataset of DVDs crawled from `dvd.ciao.co.uk` in December 2013. This dataset contains user reviews of

[2]Both Ciao and Epinions datasets are available at www.cse.msu.edu/ tangjili/trust.html

movies accompanied by their overall rating. It also contains directed trust relations between users[3].

Dataset	#users	#items	#ratings	#trusts
Ciao	1,653	16,862	26,190	32,955
Epinions	22,143	296,278	464,249	83,363
CiaoDVD	2,609	16,122	32,054	8,926

Table 5.1: Dataset statistics.

As we are dealing with implicit feedback (user-item interaction data), we convert all the observed interactions, i.e., ratings, into positive instances. For both datasets, we only keep users with at least five rated items. The final data statistics are summarized in Table 7.1. Epinions is by far the largest dataset.

5.3.2 Evaluation Protocol

We split each user sequence into training, validation, and test set. For each user, the most recent item is held out for testing. The second most recent item is held out for validation while we train the model on the remainder of the sequence. The validation set is used for tuning the model hyper-parameters, e.g., learning rate, embedding dimension, sample size of user's friends, etc.

We evaluate model performance on the test set via two widely used evaluation metrics: HitRate@10 (HR@10) and Area under Curve (AUC). HitRate@10 is computed as follows:

$$HR@10 = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \mathbb{1}(L_{u,i_u^t} < 10), \tag{5.14}$$

where i_u^t is the ground truth item that user u clicked at test time t, L_{u,i_u^t} is the rank of the ground truth item in the predicted ranked list, and $\mathbb{1}(.)$ is the indicator function. The HR@10 metric checks if the ground truth item is present in the top-10 ranking. In contrast, AUC measures the rank of the test item in the predicted ranked list. This is a harder metric as it takes into account the exact position of the ground truth item. Formally,

$$AUC = \frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{|\mathcal{I}|} \sum_{j \in \mathcal{I}} \mathbb{1}(L_{u,i_u^t} > L_{u,j}). \tag{5.15}$$

where $L_{u,j}$ is the rank of item j in the predicted ranked list for user u. Due to the spar-

[3]Dataset available from `www.librec.net/datasets.html`

sity of user-item interaction data (number of items is huge) and computational feasibility, following [152], HitRate@10 is reported on a sample subset of negative items instead of the whole set. For each user-item pair in the test set, we sample 99 negative items that the user has not rated before. Similarly, for AUC computation, we sample a set of 5,000 unobserved items for each user for both our model and the baselines.

5.3.3 Baselines

We compare to a large variety of state-of-the-art collaborative filtering (CF), temporal, social, and socio-temporal recommenders:

- BPR-MF [22] is a classic matrix factorization model which uses a pairwise ranking loss.
- NeuMF [24] is a recently proposed CF based model with neural architecture. It merges matrix factorization and multi-layer perceptron modules to predict item ranking.
- NAIS [153] is an item-to-item CF based model which employs user-specific attention between items to identify similar items[4].
- TBPR [38] extends the BPR model to capture strong and weak ties separately in a user's social network in order to distinguish the degree of influence of a user's connections on her behavior.
- SERec [36] is a state-of-the-art CF based social recommender model that augments collaborative filtering with social exposure through regularization and boosting.
- DiffNet [110] is a state-of-the-art graph convolution based social recommender that propagates user influence through their social network. Due to the absence of attributes in our dataset, we replace the user and item feature vectors with one-hot embeddings in their model[5].
- GraphRec [109] is another recently proposed social recommender that uses graph convolution networks to incorporate information from both a user-item and a user's social graph. The original model was proposed for rating prediction, and we adopt it for our item ranking task. For that purpose, we change the loss function to a log loss and augment the training data with randomly sampled negative items following other ranking prediction models [24].

[4] `github.com/AaronHeee/Neural-Attentive-Item-Similarity-Model`
[5] `github.com/PeiJieSun/diffnet`

- SASRec [30] is a state-of-the-art model for the temporal recommendation that uses a self attentive module to capture long-term interests of a user[6].
- SR-GNN [154] is a graph neural network based temporal recommender that aggregates information from all previous timesteps of the user[7].
- SPMC [29] is an extension of a Markov chain based [27] model which captures both temporal and social dynamics for recommendation[8].
- ARSE [28] is a state-of-the-art social session recommendation model. It employs a session LSTM module to model change in user preferences across sessions while incorporating social influence as an input to the LSTM module[9]. Similar to their paper, we divide the dataset into monthly intervals and predicted for the next session[10].
- DGRec [39]: Another social session recommender method that uses graph attention networks to merge preferences of social neighbors from the previous session with a user's preference in the current session. We use similar month-wise intervals to denote each session[11].

BPR-MF, NeuMF and NAIS are Collaborative Filtering models, while TBPR and SERec are social recommenders. Diffnet and GraphRec are graph convolution based social recommenders. SASRec and SR-GNN are temporal recommender models. We omit a comparison with Fossil and GRU [32] as they are RNN based temporal recommendation methods. The results for our user-temporal module are equivalent to their models. SPMC is most similar to our model as it also models temporal behavior and socio-temporal influence. However, it is a shallow model as it extends a Markov chain to model linear dependence. ARSE and DGRec are also socio-temporal recommenders proposed for session recommendations.

We use the Adam optimizer for our models with a batch size of 256 for Ciao, CiaoDVD, and 1024 for Epinions. We used the implementation provided by the RecQ python library[12] for BPR-MF, NeuMF, TBPR, and SERec models. We used the authors' implementation for all the other models. Specifically, for ARSE and GraphRec, we wish to thank the authors as they generously shared their implementation. We keep the embedding dimension $D = 32$,

[6] `github.com/kang205/SASRec`
[7] `github.com/CRIPAC-DIG/SR-GNN`
[8] `github.com/cwcai633/SPMC`
[9] `github.com/DeepGraphLearning/RecommenderSystems/`
[10] We also experimented with constraining each session to comprise of just a single item, but that resulted in slightly worse performance.
[11] We also evaluated other intervals, but they all performed similarly.
[12] `github.com/Coder-Yu/RecQ`

for all the baselines to be comparable with our model. We report the best result of the baselines using either hyper-parameters based on the authors' specifications or values, which performed better on our validation set.

5.3.4 Performance Analysis

Table 5.2 details a comparison of HR@10 and AUC for our model and the state-of-the-art baselines on all three datasets. We report mean results over five runs. Our model significantly outperforms all the baselines on the AUC metric by at least around 2% for CiaoDVD, 14% for Epinions, and 18% for Ciao. On the Ciao and Epinions dataset, our FuseRec model also improves the HR@10 metric by at least 4%. Note that each of the baselines models one or a subset of three factors (temporal, socio-temporal, and item similarity), while our FuseRec model considers all factors jointly. Each of our modules outperforms their respective baselines. Further, the combined FuseRec model considerably outperforms the individual components. This improvement indicates that our model is effective at combining the individual modules, each capturing a distinct factor affecting a user's preference.

Model Type	Models	Ciao		Epinions		CiaoDVD	
		HR@10	AUC	HR@10	AUC	HR@10	AUC
Classical	BPR-MF [22]	0.297	0.630	0.509	0.731	0.517	0.759
	NeuMF [24]	0.342	0.570	0.470	0.709	**0.551**	0.760
	NAIS [153]	0.241	0.626	0.510	0.723	0.487	0.742
Social	SERec [36]	0.295	0.550	0.421	0.613	0.385	0.647
	TBPR [38]	0.322	0.601	0.47	0.717	0.518	0.745
	DiffNet [110]	0.342	0.583	-	-	0.527	0.759
	GraphRec [109]	0.234	0.534	0.452	0.702	0.33	0.708
Temporal	SASRec [30]	0.324	0.575	0.508	0.724	0.546	0.761
	SR-GNN [154]	0.320	0.590	0.509	0.732	0.546	0.759
Social + Temporal	SPMC [29]	0.223	0.599	0.483	0.717	0.53	0.758
	ARSE [28]	0.328	0.583	0.522	0.726	0.527	0.754
	DGRec [39]	0.329	0.610	0.479	0.726	0.521	0.753
Our Indiv. Modules	User-Temporal	0.308	0.666	**0.559**	0.726	0.535	0.751
	User-Social	0.344	0.637	0.503	0.736	0.551	0.741
	Item-Similarity	0.208	0.604	0.430	0.758	0.474	0.767
Ours Combined	FuseRec	**0.355**	**0.745**	0.549	**0.834**	0.538	**0.774**

Table 5.2: Comparison of results of our model to state-of-the-art baselines and variants of our own model on three datasets. Higher values are better for both metrics. Our model significantly outperforms the baselines that model a subset of three factors. '−' indicates an out of memory error.

Amongst our proposed modules, in general, the user-social module performs best on the HR@10 metric, while the item-similarity module outperforms the other modules on the AUC metric. This difference could be because the user-social module improves the ranking of items currently popular among a user's social connections. In contrast, the item-similarity module emphasizes items that frequently co-occur in the entire dataset, irrespective of the user preferences. This increases the bias of the item ranking only slightly, as co-occurrence is a weak signal for a specific user (it is averaged across all users). This difference further underlines that it is essential to model a variety of factors for an effective recommendation.

Collaborative: CF-based approach BPR-MF performs the best among the baselines when considering the AUC metric while NeuMF performs competitively for HR@10. This superior performance highlights that classical matrix factorization approaches are still very competitive for the recommendation. The user-specific attention-based NAIS model did not outperform the non-attention based CF models.

Social: Our proposed user-social module outperforms all baseline social recommenders for both metrics. Among the baselines, the TBPR model that models the user's social graph with different weights for strong and weak ties performs better than the SERec that considers each friend's contribution equally. This reaffirms our assumption that each friend exerts a different influence on the user. In general, we found the recently proposed SERec to underperform on the three datasets used here. It proposes that social connections have a limited influence on a user's preference. Social influence is weakly modeled through an exposure prior on the items. However, we think the low performance indicates the contrary, i.e., social influence plays a significant part in shaping a user's preference.

Our GCN based user-social module is inherently different from the other GCN based baselines, DGRec, and GraphRec. Our user-social module is time-dependent with dynamic user features while both baselines operate on static features (entire item history). Further, our architecture differs: we employ attention between a user and her social connections, whereas DiffNet does not use attention, and GraphRec computes attention based on user history rather than the user itself. Our superior performance supports the difference.

However, the DiffNet model performs the best among social recommenders. However, it is unable to scale to our largest dataset, Epinions. Note that we used the authors' implementation, and our largest dataset is bigger than the one reported in their paper. In contrast, the other GCN based model GraphRec performs poorly on all the datasets. Note that for results obtained in their paper, they only consider users with non-zero social connections and items previously seen in the training data, while we do not make any such assumptions.

Temporal: Our user-temporal module uses information from only the previous timestep for prediction at the current timestep while the baseline temporal recommenders aggregate

information from all previous timesteps. However, we argue that information from a distant past is not useful. Comparable performance of our user-temporal module with the temporal baselines confirms our argument. In general, both the temporal recommenders, SASRec and SR-GNN, perform better than social recommenders and are comparable to the best performing BPR baseline on the Epinions and CiaoDVD dataset when using the AUC metric. While they perform competitively with other baselines on HR@10.

Social + Temporal: SPMC, which combines temporal and social influence, performs worse than baselines, which model either of the two factors. This worse performance is expected as it is a shallow model with linear dependence. The other socio-temporal models ARSE and DGRec perform similarly, with ARSE performing slightly better for the HR@10 metric. Note that despite modeling socio-temporal influence, these methods do not outperform even other social and temporal only baselines. This could be since these models were originally proposed for the session-based recommendation. Sessions are typically defined as a sequence of activities performed by a user in a single visit to the website or platform. Thus, for the session prediction settings, models work on recommending the next item to consume by the user in the same session. They typically assume either none or limited past session information of the users and assume static user preferences per session. In contrast, we model the evolution per item for each user and expect a significant change in a user's preference over time.

Further, DGRec only models socio-temporal influence (based on a friend's last session) and ignores the evolution of a user's preference across sessions. In contrast, ARSE models a user's evolution across sessions but aggregates information per session resulting in limited flexibility. Also, it is worthy to note that none of these models exploit similarity in the item space that is used in our proposed model. Thus, the worse performance of these session-based socio-temporal methods indicates that they are not well suited for temporal recommendation owing to information aggregation within sessions.

5.3.5 Module Analysis

We now evaluate the importance of each module (user-temporal, user-social, and item-similarity) in our combined model using learned weights, λ_k. Figure 5.6 shows the learned magnitude of the weights λ_k for $k \in \{1, \ldots, 4\}$ for each of the S_k (Equation (5.1)) scores for all the datasets. Weight λ_1 corresponds to a user-temporal embedding (h^t_u) and contributes most to the final score. This high magnitude is expected as a user's history of interactions play a crucial role in modeling user preferences accurately. λ_2 corresponds to the context-specific user-temporal embedding ($h^t_{u,i}$) and is the second most important factor for Ciao

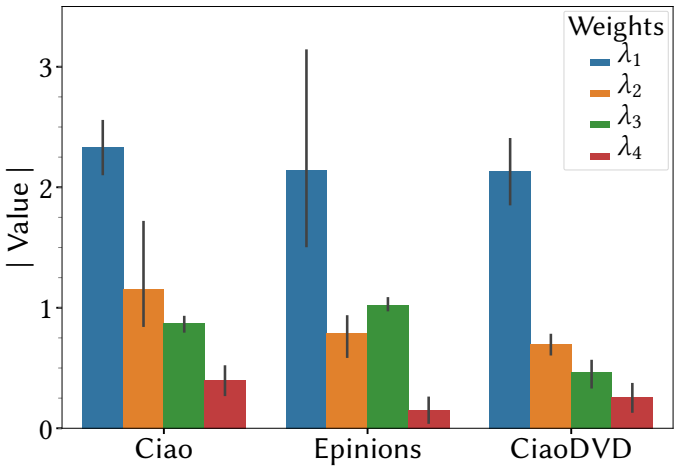


Figure 5.6: Learned value of the weights λ_k for $k \in \{1, \ldots, 4\}$ for different scores in our model. Weight λ_1 corresponding to user-temporal embedding contributes most to the final score.

and CiaoDVD datasets. In contrast, λ_3, which corresponds to a user-social embedding (s^t_u) is the second highest factor for Epinions. This difference indicates that social influence plays an important factor for users in the Epinions dataset while it is slightly less important for the Ciao and the CiaoDVD datasets. λ_4, which corresponds to the context-specific user-social embedding ($s^t_{u,i}$) contributes the least across all datasets. Note that all scores use item-similarity embeddings.

Next, we perform ablation experiments by removing each individual module at a time from the final model, as shown in Table 5.3. All variants perform worse than the final FuseRec model, emphasizing the need for each of the modules. For all the datasets, removing the user-temporal module results in the largest drop in performance. This is expected as a user's history encapsulates a great deal of information for the recommendation. Thus, personalized recommendations fare better than generic ones.

Model Variants	Ciao		Epinions		CiaoDVD	
	HR@10	AUC	HR@10	AUC	HR@10	AUC
{social + item}	0.329	0.574	0.478	0.713	0.526	0.738
{temporal + item}	0.335	0.631	0.557	0.798	0.531	0.743
{temporal + social}	0.327	0.679	0.536	0.736	0.540	0.745
FuseRec	0.355	0.745	0.549	0.834	0.538	0.774

Table 5.3: Performance of our FuseRec model for all three datasets, when removing one module at a time. All of these variants perform worse than the combined model.

Apart from the user-temporal module, for CiaoDVD, the model without the item-similarity

module performs best, while for Epinions and Ciao, the model without user-social module performs the best. Thus, for the movie reviewing platform CiaoDVD, preferences of a user's social connections play a significant role in predicting her movie preferences.

In contrast, for Epinions and Ciao, both of which contain broad categories of products, items frequently bought together by users are better predictors of purchasing behavior. This seems intuitive as movie preferences are subjective in general, while users tend to believe strangers on online reviewing platforms for products like electronics, furniture, etc.

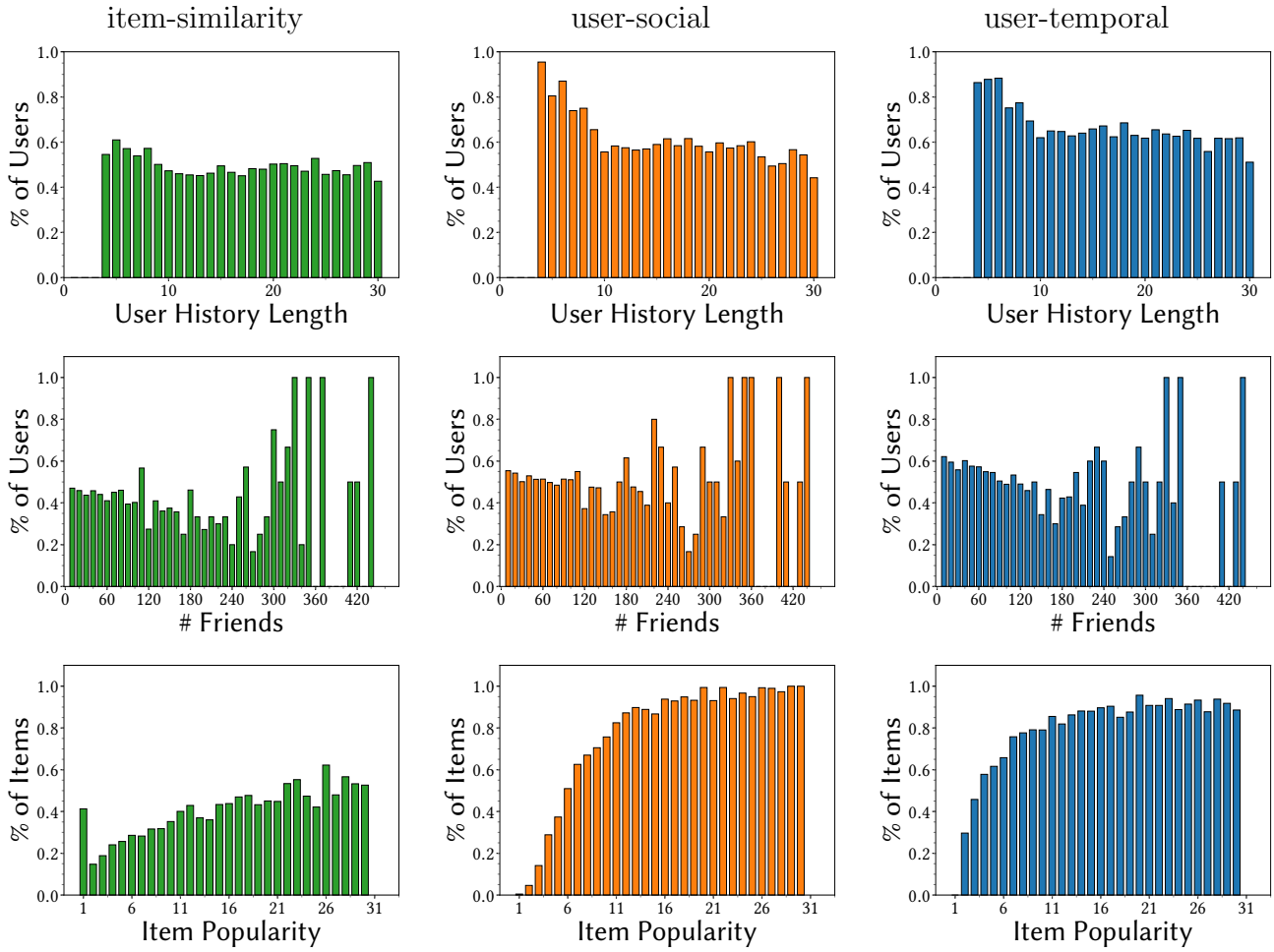


Figure 5.7: Fraction of correct item predictions on test data by individual modules with different user sequence length, varying size of a user's social connections and varying item popularity. The item-similarity module performs better for rare items while the user-social module better models cold-start users.

Further, we evaluated individual modules with varying item popularity in the dataset, different sequence length for users, and varying size of a user's social graph for the Epinions dataset as shown in Figure 5.7. The user-social module is able to predict cold-start users

better than the user-temporal module as it takes information from a user's social connections into account (top row). Note, the item-similarity module performs uniformly for different user sequence length as it does not contain any user information. Information about the past behavior of social connections (user-social module) improves performance compared to not using social connections (item-similarity module). Note that this effect increases with more friends and does not create noise as our attention model can distinguish between strong and weak connections (middle row). The item-similarity module exploits item homophily in the user space and the feature space. Thus, it can accurately predict rarely reviewed items (in the left section of the figure, close to zero) better than the other two modules (bottom row). Thus, our model provides an interpretable way of determining the importance of each of the factors used in our model through multiple experiments.

5.3.6 Cold-Start Analysis

Combining different factors in each of the three modules helps to alleviate data sparsity and to predict for cold-start users effectively. To verify this claim, we evaluate our model in cold-start settings using a decreasing number of past available interactions for a user. Figure 5.8 provides our results with varying user sequence length compared to the baselines. Our model significantly outperforms the baselines modeling a subset of the factors for cold-start settings on all the three datasets reaffirming our claim. Also, we observe that our model performs comparably for user history lengths 10 and above, while the performance drops for very short sequence length of 5. This further underlines that a user's temporal history is an essential cue for the recommendation.

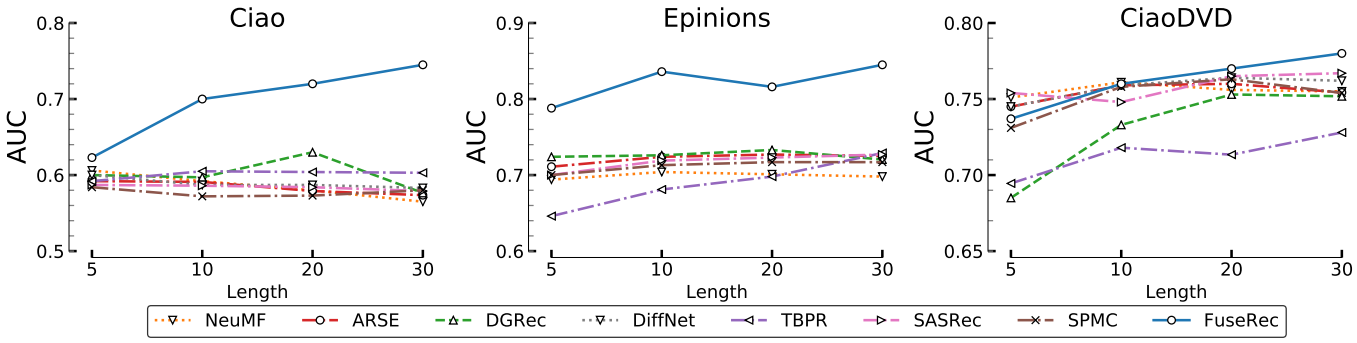


Figure 5.8: AUC over different number of past interactions to mimic cold-start settings. Our model consistently outperforms all the social and temporal based baselines.

5.3.7 Parameter Sensitivity

	Ciao		Epinions		CiaoDVD	
	HR@10	AUC	HR@10	AUC	HR@10	AUC
No Attn	0.313	0.707	0.536	0.697	0.515	0.742
Userside Attn	0.356	0.719	0.531	0.729	0.532	0.747
Itemside Attn	0.356	0.737	0.541	0.740	0.536	0.752
FuseRec	0.355	0.745	0.549	0.834	0.538	0.774

Table 5.4: Performance of our model without attention in the user-social and the item-similarity module. Attention in item-similarity module only performs better than only user-social module attention.

Table 5.4 details performance results after removing attention from our user-social and item-similarity modules, respectively. The variant without attention on both the modules performs the worst. Comparing the user-social and item-similarity module: only using attention in the item-similarity module performs better than using attention in the user-social module exclusively. This is expected as the ties between similar items are weaker than explicit social connections between users. Thus, attention results in lower weights for noisy connections in the item-similarity module.

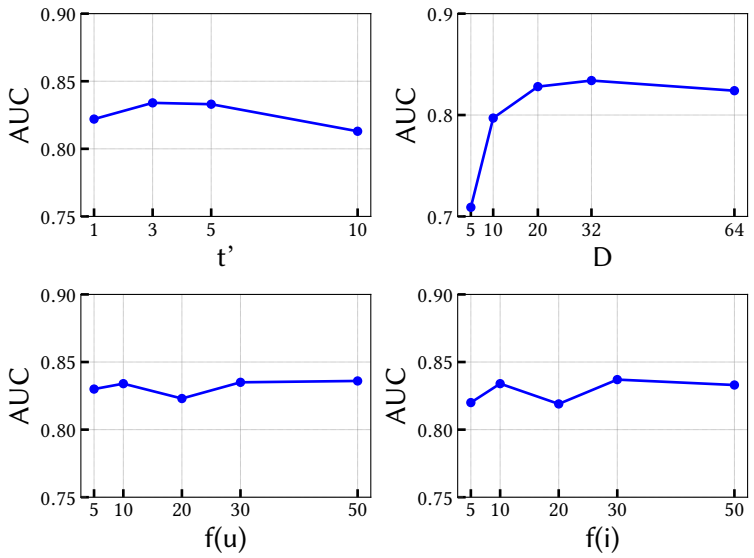


Figure 5.9: Ablation results of different model hyper-parameters on Epinions datatset.

Figure 5.9 illustrates the performance of our model with different hyper-parameters on the Epinions dataset. For a user's friend's history length t' (Equation (5.8)), we observe: using fewer past interactions performs better. However, no change in performance is observed if

t' increases to more than the last five interacted items. Increasing the size of the hidden dimension D improves performance initially due to larger model capacity. However, results decline beyond a certain point due to overfitting. For the user and item friend sample size, $F(u)$ (Equation (7.7)) and $F(i), F'(i)$ (Equation (5.11)) respectively, performance slightly increases with an increased size but plateaus for sample sizes larger than 10.

5.3.8 Induced relationships

We further experimented with different graphs both in the User-Social and Item-Similarity module. In the user-social module, apart from the explicit social connections, we also induced another user graph based on similar preferences. Specifically, we computed k-nearest neighbors for each user using cosine similarity between their past item interactions. Similar to item graphs, we keep $k = 30$ for the user induced graph. While for the item side, we experiment with two different graphs, one based on feature similarity between items, and another is based on frequent co-occurrence in the user's history.

Table 5.5 details performance results when using different user and item graphs for all datasets. In general, using an induced user graph based on similar history performs competitively or slightly outperforms the explicit social graph. This reaffirms our hypothesis that connected users exhibit similar behavior. Note that computing similar users can be computationally expensive if the dataset is large.

		Ciao		Epinions		CiaoDVD	
User Graph	Item Graph	HR@10	AUC	HR@10	AUC	HR@10	AUC
Social	Co-Occurrence	0.308	0.743	0.518	0.805	0.554	0.749
Social	Feature	0.292	0.709	0.520	0.827	0.537	0.747
Induced	Both	0.320	0.753	–	–	0.540	0.763
Social	Both	0.355	0.745	0.549	0.834	0.538	0.774

Table 5.5: Performance of our model with different graphs in the user-social and the item-similarity module. Induced relationships in the user graph fare little better than explicit social connections. In the item space also, item graph based on frequent co-occurrence in the user history is a better indicator of future item prediction. We could not extract induced user graph in Epinions because of out of memory error.

In the item space, for Ciao and CiaoDVD, the co-occurrence graph performs better than the feature similarity graph. This competent performance is intuitive as most of the item collaborative filtering methods also leverage co-occurrence for providing recommendations. Besides, a similar item category may not be a strong signal of repeat user behavior as users tend to buy things across different categories.

5.4 CONCLUSION

We presented a model which captures temporal changes and socio-temporal influence while taking into account item similarity and later combines them in an interpretable manner. We used an RNN model to capture the evolution of user preferences. We proposed an attention based user social module which aggregates historical features for all of the user's neighbors. To capture item-based homophily, we proposed an attention based item module to learn item embeddings using similar items frequently co-occurring in the platform. We compared our approach to a large number of temporal, social, and socio-temporal recommenders on three benchmark datasets. We report an improvement of more than 13% over state-of-the-art baselines on the Ciao and Epinions datasets for AUC metric. Our ablation study shows that each module is essential to capture different factors affecting user behavior in recommender systems. Further, user-temporal module is most important factor across all datasets.

CHAPTER 6: INDUCING SEMANTICALLY DIVERSE RELATIONSHIPS BASED ON USER BEHAVIOR

In this chapter, we remove the limiting assumption of the previous chapter that connected users only encode the similarity relationship. Further, we also extend the implicit social influence between users captured in the previous chapter. This implicit influence modeling is specially helpful for platforms with sparse social information or platforms with a lack of provision to establish social connections.

For this purpose, we focus on the task of selecting the best answer out of the given answers to a question in CQA forums. Current approaches predict answer quality in isolation of the other answers to the question and user activity across the forum (other posted questions or answers). This assumption is limiting as the best answer is, in general, selected based on how it differs from other answers to the same question. Similarly, answers given by expert users tend to be of higher quality. We argue that leveraging these cues can significantly help in the answer selection task. Specifically, we transform the answer selection task to a node binary label prediction (best answer or not) task where each node represents a user-provided answer in the forum. We then induce multiple graphs between these nodes based on similarity and contrast in the behavior of users providing the answers. Finally, multiple graphs expressing semantically diverse relationships are merged to predict the node label (best answer) [155].

6.1 OVERVIEW

Individuals often visit Community Question Answer (CQA) forums, like StackExchange, to seek answers to nuanced questions that are not readily available on web-search engines. Unlike other familiar Learning-to-Rank problems in the IR community [156, 157], CQA platforms can identify and leverage past questions asked by similar users and relevant answers to those questions. On some CQA sites like StackExchange, individuals who post questions may label an answer as 'accepted,' but other questions with answers (about 47% in our analysis) have none labeled as 'accepted.' On other CQA sites like Reddit, there is no mechanism for a person to label an answer as 'accepted.' As a first step to address the individual's information needs, in this work, we focus on the problem of identifying accepted answers on StackExchange.

One approach to identify relevant answers is to identify salient features for each question-answer tuple (q, a) and treat it as a supervised classification problem [51, 52, 53, 158]. Deep Text Models further develop this approach [60, 61, 62, 66]. These models learn the optimal text representation of (q, a) tuple to select the most relevant answer. While the deep text

models are sophisticated, text based models are computationally expensive to train. Furthermore, there are limitations to examining (q, a) tuples in isolation: an answer is "relevant" *in relationship* to other answers to the same question; second, it ignores the fact that same user may answer multiple questions in the forum. These relational aspects of user-generated content provide a unique dimension that is absent in textual search. However, there is only limited work in the context of identification of "best answers" among user-generated content that exploit these implicit and explicit connections. Thus, our key proposal is to use this alternative approach and build a flexible and expressive framework to incorporate the relational aspects of user-generated content for the answer selection task.

Relational aspects are best captured as graphs connecting content. Graph Convolutional Networks (GCNs) is a popular technique to incorporate graph structure, and are used in tasks including node classification [9] and link prediction [95]. Extensions to the basic GCN model include signed networks [103], inductive settings [104] and multiple relations [105, 95]. While GCNs are a plausible approach, we need to overcome a fundamental implicit assumption in prior work before we can apply it towards our problem. Prior work in GCNs adopt label sharing amongst nodes; label sharing implicitly assumes similarity between two nodes connected by an edge. In the Answer Selection problem, however, answers to the same question connected by an edge may not share acceptance label. In particular, we may label an answer as 'accepted' based on how it differs with other answers to the same question. Signed GCNs [103] can not capture this contrast despite their ability to incorporate signed edges. Graph attention networks [159] also could not learn negative attention weight over neighbors as weights are the output of a softmax operation. In other words, the relational views (or graphs) could capture similarity or contrast between connected content, depending on the relation in consideration.

We develop a novel framework to model the diverse relations between content through a separate *induced* graph across (q, a) tuples. The key idea is to use diverse strategies—label depends only on the answer (reflexive), label is determined in contrast with the other answers to the question (contrastive), and label sharing among answers across questions if it contrasts with other answers similarly(similarity by contrast)—to identify the accepted answer. Each strategy *induces* a graph between (q, a) tuples and then uses a particular label selection mechanism to identify the accepted answer. Our strategies generalize to a broader principle: pick an equivalence relation to induce a graph comprising cliques, and then pick a label selection mechanism (label sharing or label contrast) within each clique. We show how to develop GCN architecture to operationalize the specific label selection mechanism (label sharing or label contrast). Then, we aggregate results across strategies through a boosting framework to identify the label for each (q, a) tuple. Our Contributions are as follows:

Modular, Induced Relational Framework: We introduce a modular framework that separates the construction of the graph with the label selection mechanism. In contrast, prior work in answer selection (e.g., [51, 52, 53, 158].) looked at individual tuples, and work on GCNs (e.g., [9, 105]) use the given graph (i.e., no induced graphs) and with similarity as a mechanism for label propagation. We use equivalence relations to induce a graph comprising cliques and identify two label assignment mechanisms—label contrast, label sharing. Then, we show how to encode these assignment mechanisms in GCNs. In particular, we show that the use of equivalence relations allows us to perform *exact* convolution in GCNs. We call our framework Induced Relational GCN (IR-GCN). Our framework allows for parallelization and applies to other problems that need application semantics to induce graphs independent of any existing graphs[160].

Discriminative Semantics: We show how to encode the notion of label contrast between a vertex and a group of vertices in GCNs. Label contrast is critical to the problem of best answer selection. Related work in GCNs (e.g., [9, 105]) emphasizes node similarity, including the work on signed graphs [103]. In [103], contrast is a property of an edge, not a group and is not expressive enough for our problem. We show that our encoding of contrast creates *discriminative magnification*—the separation between nodes in the embedding space is most meaningful at smaller clique sizes; the effect decreases with clique size.

Boosted Architecture: We show through extensive empirical results that using common boosting techniques improves learning in our convolutional model. This improvement is a surprising result since much of the work on neural architectures develops stacking, fusion, or aggregator architectures.

We conducted extensive experiments using our IR-GCN framework with excellent experimental results on popular CQA forum—StackExchange. For our analysis, we collect data from 50 communities—the ten largest communities from each of the five StackExchange[1] categories. We achieved an improvement of over 4% accuracy and 2.5% in MRR on average over state-of-the-art baselines. We also provide Reddit [2] results using expert answers as a proxy for acceptance, to overcome the absence of explicit labels. Finally, we show that our model is more robust to label sparsity compared to alternate GCN based multi-relational approaches.

We organize the rest of this chapter as follows. In section 6.2, we formulate our problem statement and then discuss induced relations for Answer Selection problem in section 6.3. We

[1] https://stackexchange.com/sites
[2] https://www.reddit.com/

then detail the operationalization of these induced relations in Graph Convolution framework in section 6.4 and introduce our gradient boosting based aggregator approach in section 6.5. Section 6.6 describes experiments and Section 6.7 describes further ablation studies. We finally conclude in section 6.8.

6.2 PROBLEM FORMULATION

In Community Question Answer (CQA) forums, an individual asking a question seeks to identify the most relevant candidate answer to his question. On Stack-Exchange CQA forums, users annotate their preferred answer as "accepted."

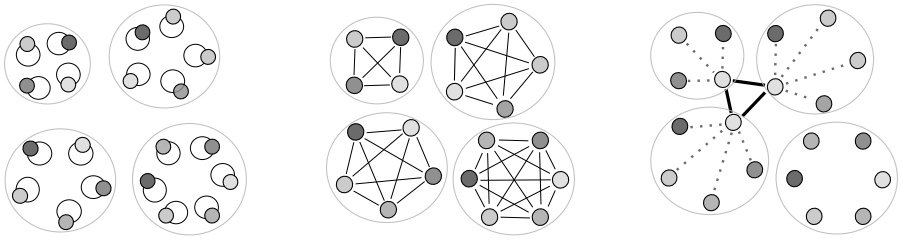
Let $\mathcal{Q}$ denote the set of questions in the community and for each $q \in \mathcal{Q}$, we denote $\mathcal{A}_q$ to be the associated set of answers. Each question $q \in \mathcal{Q}$, and each answer $a \in \mathcal{A}_q$ has an author $u_q, u_a \in \mathcal{U}$ respectively. Without loss of generality, assume that we can extract features for each question q, each answer $a \in \mathcal{A}_q$, user $u_q, u_a \in \mathcal{U}$.

Our unit of analysis is a question-answer tuple $(q, a), q \in \mathcal{Q}, a \in \mathcal{A}_q$, and we associate each (q, a) tuple with a label $y_{q,a} \in \{-1, +1\}$, where '+1' implies acceptance and '-1' implies rejection.

> The goal of this paper is to develop a framework to identify the accepted answer to a question posted on a CQA forum.

6.3 INDUCED RELATIONAL VIEWS

In this section, we discuss the idea of induced relational views, central to our induced relational GCN framework developed in Section 6.4. First, in Section 6.3.1, we introduce potential strategies for selecting the accepted answer given a question. We show how each strategy induces a graph G on the question-answer (q, a) tuples. Next, in Section 6.3.2, we show how each of these example strategies is an instance of an equivalence relation; our framework generalizes to incorporate any such relation.



(a) Reflexive (b) Contrastive (c) Similar Contrast

Figure 6.1: Reflexive(fig. 6.1a), Contrastive (fig. 6.1b) and Similar Contrast (fig. 6.1c) relations among (q, a) tuples. Reflexive assumes no dependence on other answers for prediction. Contrastive compares between all answers to a question; Similar Contrast connects answers across questions if they contrasts with other answers similarly. Solid lines show the similarity relation while dotted lines signify the contrast. The contrast is only significant in three questions.

6.3.1 Constructing Induced Views

In this section, we discuss in detail four example strategies that can be used by the individual posting the question to label an answer as 'accepted.' Each of the $S_i \in \mathbf{S}$ strategies *induces* a graph $G_i = (V, E_i)$ (also referred to as a relational view). In each graph G_i, a vertex $v \in V$ corresponds to a tuple (q, a) and an edge $e \in E_i, E_i \subseteq V \times V$ connects two tuples that are matched under that strategy. Note that each G_i has the same vertex set V, and the edge sets E_i are strategy dependent. Each strategy employs one of the three different relation types—reflexive, contrastive, and similar—to connect the tuples. We use one reflexive strategy, one contrastive, and two similar strategies. Figure 6.1 summarizes the three relations. Below, we organize the discussion by relation type.

Reflexive

A natural strategy is to examine each (q, a) tuple in isolation and then assign a label $y_{q,a} \in \{-1, +1\}$ corresponding to 'not accepted' or 'accepted.' In this case, $y_{q,a}$ depends on only the features of (q, a). This is a **Reflexive** relation, and the corresponding graph $G_r = (V, E_r)$ has a specific structure. In particular, in this graph G_r, we have only self-loops, and all edges $e \in E_r$ are of the type (v, v). That is, for each vertex $v \in V$, there are no edges (v, u) to any other vertices $u \neq v \in V$. Much of the prior work on feature driven answer selection [51, 52, 53, 158] adopts this view.

Contrastive

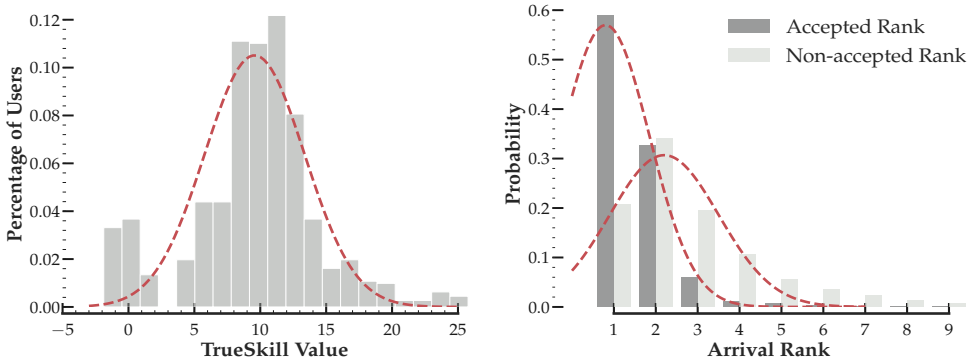
A second strategy is to examine answers *in relation* to other answers to the same question and label one such answer as 'accepted.' Thus the second strategy *contrasts* (q, a), with other tuples in $(q, a'), q \in \mathcal{Q}; a, a' \in \mathcal{A}_q; a' \neq a$. This is a **Contrastive** relation and the corresponding graph $G_c = (V, E_c)$ has a specific structure. Specifically, we define an edge $e \in E_c$ for all (q, a) tuples for the same question $q \in \mathcal{Q}$. That is, if $v = (q_1, a_1), u = (q_2, a_2)$, $e = (u, v) \in E_c \iff q_1 = q_2$. Intuitively, the contrastive relation induces cliques connecting all answers to the same question. Introducing contrasts between vertices sharpens differences between features, an effect (described in more detail in Section 6.4.2) we term *Discriminative Feature Magnification*. Notice that the contrastive relation is distinct from graphs with signed edges (e.g., [103]). In our framework, the contrast is a *neighborhood* property of a vertex, whereas in [103], the negative sign is a property of an *edge*.

Similar Contrasts

A third strategy is to identify *similar* (q, a) tuples *across* questions. Prior work [161] indicates that individuals on StackExchange use diverse strategies to contribute answers. Experts (with high reputation) tend to answer harder questions, while new members (with low reputation) looking to acquire reputation tend to be the first to answer a question.

How might similarity by contrast work? Consider two individuals Alice and Bob with *similar* reputations (either high or low) on StackExchange, who contribute answers a_A and a_B to questions q_1 and q_2 respectively. If Alice and Bob have high reputation difference with other individuals who answer questions q_1 and q_2 respectively, then it is likely that (q_1, a_A) and (q_2, a_B) will share the same label (if they are both experts, their answers might be accepted, if they are both novices, then this is less likely). However, if Alice has a high reputation difference with other peers who answer q_1, *but Bob does not have that difference* with peers who answer q_2, then it is less likely that the tuples (q_1, a_A) and (q_2, a_B) will share the label, even though the reputations of Alice and Bob are similar.

Thus the key idea of the **Similar Contrasts** relation is that link tuples that are *similar in how they differ* with other tuples. We construct the graph $G_s = (V, E_s)$ in the following manner. An edge $e = (v, u)$ between tuples v and u exists if the similarity $s(v, u)$ between tuples v, u exceeds a threshold δ. We define the similarity function $s(\cdot, \cdot)$ to encode similarity by contrast. That is, $e = (v, u) \in E_s \iff s(v, u) \geq \delta$.



(a) TrueSkill Distribution (b) ArrivalRank Distribution

Figure 6.2: Distribution of the TrueSkill values of users and ArrivalRank of accepted answers and non-accepted answers for the movie StackExchange. Early answers are more likely to be accepted and variance of TrueSkill similarity across users is high.

Motivated by [161] and our empirical analysis (fig. 6.2), we consider two different views that correspond to the similar contrast relation. The **TrueSkill Similarity** view connects all answers authored by a user where her skill (computed via Bayesian TrueSkill [162])

differs from competitors by margin δ. We capture both cases when the user is less or more skilled than her competitors. **TrueSkill Similarity** connects answers authored by a specific user, where the difference in his skill over peers is greater than margin δ. Specifically, if the user authors answers a, a' to questions q, q', we create a link between a and a' if

$$|S_{u,a} - S_{u,b}| > \delta; \forall b \in \mathcal{A}_(q)$$
$$|S_{u,a'} - S_{u,c}| > \delta; \forall c \in \mathcal{A}_(q')$$

where $S_{u,a}$ is the skill value for the user who authored answer a. Similarly, a link is created for the opposite case when difference is less than $-\delta$. We estimate the user skill values with the TrueSkill rating system (`https://pypi.org/project/trueskill/`) computed with their historic performance in the community. TrueSkill values are normally distributed among users (fig. 6.2).

In the **Arrival Similarity** view, we connect answers across questions based on the similarity in the relative time of their arrival (posting timestamp). The temporal arrival patterns of answers are correlated to their acceptance probabilities (fig. 6.2). For a specific user authoring answers a, a' to questions q, q', we establish a link between these answers if

$$|T_a - T_b| > \gamma \times \max(T_b); \forall b \in \mathcal{A}_(q)$$
$$|T_{a'} - T_c| > \gamma \times \max(T_c); \forall c \in \mathcal{A}_(q')$$

where T_a represents the relative time-gap between answer a and the question q. Conversely, we create links when difference is less than $-\gamma \times \max(T_b)$.

Our hypotheis is that a similar answering schedule indicates similar user confidence or skill across questions. Notice that two Similar Contrast views have different edge (E) sets since the corresponding similarity functions are different. Notice also, that the two similarity function definitions are transitive. [3]

6.3.2 Generalized Views

Now we present the general case of the induced view. First, notice that each of the three relation types that we consider—reflexive, contrastive, and similar—result in a graph

[3] One trivial way of establishing similarity is co-authorship i.e. connect all (q, a) tuples of a user (probably on the same topic) across different questions. Note that the accepted answer is labeled in relation to the other answers. As the competing answers are different in each question, we can not trivially assume acceptance label similarity for all co-authored answers. In our experiments, co-authorship introduced a lot of noisy links in the graph leading to worse performance.

$G_i = (V, E_i)$ comprising a set of cliques. This is not surprising, since all three relations presented here, are equivalence relations. Second, observe the semantics of how we select the tuple with the accepted answer. Within the three relations, we used two semantically different ways to assign the 'accepted' answer label to a tuple. One way is to share the labels amongst all the vertices in the *same clique* (used in the reflexive and the similar relations). Second is to *assign label based on contrasts with other vertices* in the same clique. We can now state the organizing principle of our approach as follows.

> A generalized *modular* framework: pick a meaningful equivalence relation on the (q, a) tuples to induce graph comprising cliques and then apply specific label semantics within each clique.

Equivalence relation results in a graph with a set of disconnected cliques. Then, within a clique, one could use application-specific semantics, different from two discussed in this paper, to label tuples as 'accepted.' Cliques have some advantages: they have well-defined graph spectra [163, p. 6]; cliques allows for *exact* graph convolution; parallelize the training as the convolution of a clique is independent of other cliques.

Thus, each strategy induces a graph $G_i = (V, E_i)$ using one of the three equivalence relations—reflexive, contrastive and similar—and then applies one of the two semantics ('share the same label'; 'determine label based on contrast').

6.4 INDUCED RELATIONAL GCN

Now, we will encode the two label assignment mechanisms within a clique via a graph convolution. First, we briefly review Graph Convolution Networks (GCN) and identify some key concepts. Then, given the views G_i for the four strategies, we show how to introduce label contrasts in Section 6.4.2 followed by label sharing in Section 6.4.3.

6.4.1 Graph Convolution

Graph Convolution models adapt the convolution operations on regular grids (like images) to irregular graph-structured data $G = (V, E)$, learning low-dimensional vertex representations. If for example, we associate a scalar with each vertex $v \in V$, where $|V| = N$, then we can describe the convolution operation on a graph by the product of signal $x \in \mathbb{R}^N$ (feature vectors) with a learned filter g_θ in the fourier domain. Thus,

$$g_\theta * x = U\, g_\theta\, U^T x, \tag{6.1}$$

where, Λ and U are the eigenvalues and eigenvector of the normalized graph Laplacian, $L = I_N - D^{-1/2}AD^{1/2}$, and where $L = U\Lambda U^T$. A denotes the adjacency matrix of a graph G (associated with a view) with N vertices. Equation (6.1) implies a filter g_θ with N free parameters, and requires expensive eigenvector decomposition of the adjacency matrix A. Defferrard et al. [101] proposed to approximate g_θ, which in general is a function of Λ, by a sum of Chebyshev polynomials $T_k(x)$ up to the k-th order. Then,

$$g_\theta * x \approx U \sum_{k=0}^{K} \theta_k T_k(\tilde{\Lambda})\, U^T x \approx \sum_{k=0}^{K} \theta_k T_k(\tilde{L})\, x, \tag{6.2}$$

where, $\tilde{\Lambda} = 2\Lambda/\lambda_{\max} - I_N$ are the scaled eigenvalues and $\tilde{L} = 2L/\lambda_{max} - I_N$ is the corresponding scaled Laplacian. Since $\tilde{L} = U\tilde{\Lambda}U^T$, the two equations are approximately equal.

The key result from Defferrard et al. [101] is that Equation (6.2) implies k-hop localization—the convolution result depends only on the k-hop neighborhood. In other words, Equation (6.2) is a k-hop approximation.

However, since we use equivalence relations in our framework that result in cliques, we can do an *exact* convolution operation since vertices in a clique only have one-hop (i.e., $k = 1$) neighbors (see lemma 5.2, [164]). The resulting convolution is linear in L and now has only two filter parameters, θ_0 and θ_1 shared over the whole graph.

$$g_\theta * x = \theta_0 x + \theta_1 \left(L - I_N\right) x \tag{6.3}$$

We emphasize the distinction with Kipf and Welling [9] who approximate the Defferrard et al. [101] observation by restricting $k = 1$. They do so since they work on arbitrary graphs; since our relations result in views with cliques, we do not make any approximation by using $k = 1$.

6.4.2 Contrastive Graph Convolution

Now, we show how to perform graph convolution to encode the mechanism of contrast, where label assignments for a tuple depend on the contrast with its neighborhood.

To establish contrast, we need to compute the *difference* between the vertex's own features to its neighborhood in the clique. Thus we transform Equation (6.3) by setting $\theta = \theta_0 = \theta_1$, which essentially restricts the filters learned by the GCN. This transformation leads to the

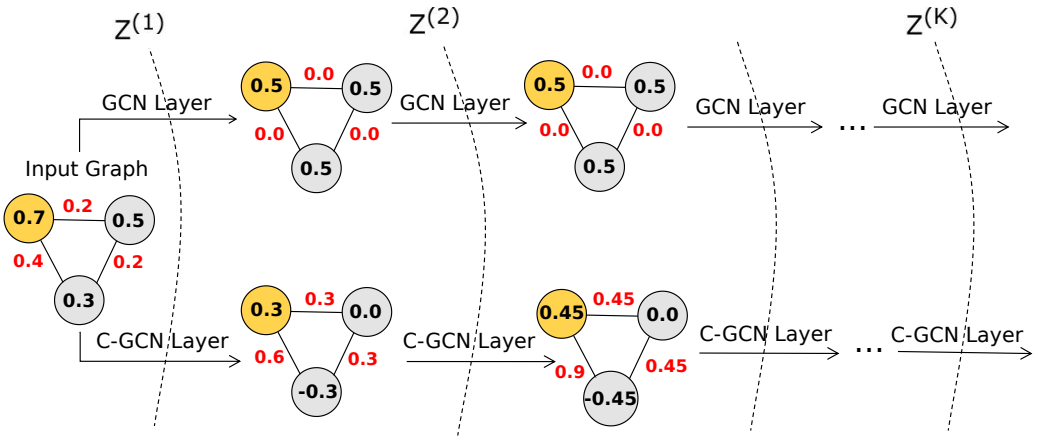


Figure 6.3: Stylized example showing the convolution results of GCN and proposed Contrastive GCN for a question with three answers. Edge labels denote the feature difference while node labels denote the resulting feature value. The feature difference between neighboring nodes increases with each convolution layer for Contrastive GCN while GCN averages the feature values among nodes.

following convolution operation:

$$g_\theta * x = \theta \left(I_N + L - I_N\right) x \tag{6.4}$$

$$g_\theta * x = \theta \left(I_N - D^{-1/2} A D^{-1/2}\right) x \tag{6.5}$$

Notice that Equation (6.5) says that for example, for any vertex u with a scalar feature value x_u, for a given clique with $n \geq 2$ vertices, the convolution operation computes a new value $\hat{x}_u$ for vertex u as follows:

$$\hat{x}_u = \theta \left(x_u - \frac{1}{n-1} \sum_{v \in \mathcal{N}_u} x_v \right). \tag{6.6}$$

where $\mathcal{N}_u$ is the neighborhood of vertex u. Notice that since our equivalence relations construct cliques, for all vertices u that belong to a clique of size n, $|\mathcal{N}_u| = n - 1$.

When we apply the convolution operation in Equation (6.5) at each layer of GCN, output for the k-th layer is:

$$\mathbf{Z}_c^k = \sigma \left(\left(I_N - D^{-1/2} A_c D^{1/2}\right) \mathbf{Z}_c^{k-1} \mathbf{W}_c^k \right) \tag{6.7}$$

with A_c denoting the adjacency matrix in the contrastive view. $\mathbf{Z}_c^k \in \mathbb{R}^{N \times d}$ are the learned vertex representations for each (q, a) tuple under the contrastive label assignment. N is the total number of tuples and d refers to the dimensionality of the embedding space. $\mathbf{Z}^{k-1}$ refers

to the output of the previous $(k-1)$-th layer, and $\mathbf{Z}^0 = X$ where X is the input feature matrix. $\mathbf{W}_c^k$ are the filter θ parameters learnt by the GCN; $\sigma(\cdot)$ denotes the activation function (e.g. ReLU, tanh).

To understand the effect of Equation (6.7) on a tuple, let us restrict our attention to a vertex u in a clique of size n. We can do this since the convolution result in one clique is unaffected by other cliques. When we do this, we obtain:

$$z_c^k(u) = \sigma\left(\left(z_c^{k-1}(u) - \frac{1}{n-1}\sum_{v\in\mathcal{N}_u} z_c^{k-1}(v)\right)\mathbf{W}_c^k\right). \tag{6.8}$$

Now consider a pair of contrasting vertices, u and v in the same clique of size n. Let us ignore the linear transform by setting $W_c^k = \mathbf{I}$ and set $\sigma(\cdot)$ to the identity function. Then we can easily verify that:

$$z_c^k(u) - z_c^k(v) = \underbrace{\left(1 + \frac{1}{n-1}\right)}_{\text{magnification}} \times \underbrace{\left(z_c^{k-1}(u) - z_c^{k-1}(v)\right)}_{\text{contrast in previous layer}}, \tag{6.9}$$

where, $z_c^k(u)$ denotes the output of the k-th convolution layer for the u-th vertex in the contrastive view. As a result, each convolutional layer magnifies the feature contrast between the vertices that belong to the same clique. Thus, the contrasting vertices move further apart. We term this as *Discriminative Feature Magnification* and Equation (6.9) implies that we should see higher magnification effect for smaller cliques. An illustration is provided in the bottom part of the fig. 6.3 with a uni-dimensional feature.

Contrasting nodes are shifted further apart by eq. (6.7) improving their separability in the learned manifold (further discussion in section 6.7.6).

6.4.3 Encoding Similarity Convolution

We next discuss how to encode the mechanism of sharing labels in a GCN. While label sharing applies to our similar contrast relation (two strategies: Arrival similarity; TrueSkill similarity, see Section 6.3.1), it is also trivially applicable to the reflexive relation, where the label of the tuple only depends on itself. First, we discuss the case of similar contrasts.

Encoding Similar Contrasts

To encode label sharing for the two similar by contrast cases, we transform Equation (6.3) with the assumption $\theta = \theta_0 = -\theta_1$. Thus

$$g_\theta * x = \theta \left(I_N + D^{-1/2} A D^{-1/2}\right) x, \tag{6.10}$$

Similar to the Equation (6.5) analysis, convolution operation in Equation (6.10) computes a new value $\hat{x}_u$ for vertex u as follows:

$$\hat{x}_u = \theta \left(x_u + \frac{1}{n-1} \sum_{v \in \mathcal{N}_u} x_v \right). \tag{6.11}$$

$$\hat{x}_u = \theta \left(\frac{n-2}{n-1} x_u + \frac{n}{n-1} \mu_x \right). \tag{6.12}$$

That is, in the mechanism where we share labels in a clique, the convolution pushes the values of each vertex in the clique to the average feature value, $\mu_x = \frac{1}{n} \sum_{v \in \mathcal{N}_u \cup u} x_v$, in the clique.

When we apply the convolution operation in Equation (6.10) at each layer of GCN, output for the k-th layer:

$$\mathbf{Z}_s^k = \sigma \left(\left(I_N + D^{-1/2} A_s D^{1/2}\right) \mathbf{Z}_s^{k-1} \mathbf{W}_s^k \right) \tag{6.13}$$

with A_s denoting the adjacency matrix in the similar views.

We analyze the similarity GCN in a maner akin to Equation (6.8) and we can easily verify that:

$$z_s^k(u) - z_s^k(v) = \underbrace{\left(1 - \frac{1}{n-1}\right)}_{\text{reduction}} \times \underbrace{\left(z_s^{k-1}(u) - z_s^{k-1}(v)\right)}_{\text{contrast in previous layer}}, \tag{6.14}$$

where, $z_s^k(i)$ denotes the output of the k-th convolution layer for the i-th vertex in the similar view. As a result, each convolutional layer reduces the feature contrast between the vertices that belong to the same clique. Thus, the similar vertices move closer (see top part in fig. 6.3).

The proposed label sharing encoding applies to both similar contrast strategies (TrueSkill; Arrival). We refer to the corresponding vertex representations as $\mathbf{Z}_{ts}^k$ (TrueSkill), $\mathbf{Z}_{as}^k$ (Arrival).

Reflexive Convolution

We encode the reflexive relation with self-loops in the graph resulting in an identity adjacency matrix. This relation is the trivial label sharing case, with an independent assignment

of vertex labels. Thus, the output of the k-th convolutional layer for the reflexive view, $\mathbf{Z}_r^k$ reduces to:

$$\mathbf{Z}_r^k = \sigma\left(I_N \mathbf{Z}_r^{k-1} \mathbf{W}_r^k\right) \tag{6.15}$$

Hence, the reflexive convolution operation is equivalent to a feedforward neural network with multiple layers and activation $\sigma(\cdot)$.

Each strategy $S_i \in \mathbf{S}$ belongs to one of the three relation types—reflexive, contrastive and similarity, where $\mathbf{R}$ denotes the set of strategies of that relation type. $\mathcal{R} = \bigcup \mathbf{R}$ denotes the set of all relation types. $\mathbf{Z}_i^K \in \mathbb{R}^{NXd}$ represents the d dimensional vertex embeddings for strategy S_i at the K-th layer. For each strategy S_i, we obtain a scalar score by multiplying $\mathbf{Z}_i^K$ with transform parameters $\widetilde{W}_i \in \mathbb{R}^{d\times 1}$. The sum of these scores gives the combined prediction score, $\mathbf{H}_\mathbf{R} \in \mathbb{R}^{NX1}$, for that relation type.

$$\mathbf{H}_\mathbf{R} = \sum_{S_i \in \mathbf{R}} \mathbf{Z}_i^K \widetilde{W}_i^T \tag{6.16}$$

In this section, we proposed Graph Convolutional architectures to compute vertex representations of each (q, a) tuple under the four strategies. In particular, we showed how to encode two different label assignment mechanisms—label sharing and determine label based on contrast—within a clique. The architecture that encodes label assignment based on contrast is a novel contribution; distinct from the formulations presented by Kipf and Welling [9] and its extensions [103, 95]. Prior convolutional architectures implicitly encode the label sharing mechanism (eq. (6.10)); however, label sharing is unsuitable for contrastive relationships across vertices. Hence our architecture fills this gap in prior work.

6.5 AGGREGATING INDUCED VIEWS

In the previous sections, we introduced four strategies to identify the accepted answer to a question. Each strategy induces a graph or relational view between (q, a) tuples. Each relational view is expected to capture semantically diverse neighborhoods of vertices. The convolution operator aggregates the neighborhood information under each view. The key question that follows is, *how do we combine these diverse views in a unified learning framework?* Past work has considered multiple solutions:

- **Neighborhood Aggregation**: In this approach, they represent vertices by aggregating feature representations of it's neighbors across all views [104, 95]. Specifically, the final adjacency matrix is the sum of all the individual adjacency matrices of each view, i.e., $A = \sum_{S_i \in \mathbf{S}} A_i$. They, then, apply Graph Convolution Network to this updated Adjacency matrix.

- **Stacking**: Multiple convolution layers stacked end-to-end (each potentially handling a different view) [165]. Specifically, they stacks all GCNs belonging to a view such that output of a lower GCN is fed as an input to the GCN directly above it. Thus, output from the last layer of GCN for view i, Z_i^K s.t. $S_i \in \mathbf{S}$ will act as input features, Z_j^0 for some other view j s.t. $S_j \in \{\mathbf{S} - S_i\}$ if view j is directly above the view i. In our experiments, we obtain the best performance by using the following order: Contrastive, Similarity by Contrast followed by Reflexive.

- **Fusion**: Follows a multi-modal fusion approach [166], where views are considered distinct data modalities. It treats each GCN as a separate model and appends the output from the final layer of each GCN i.e. $Z_i^K; \forall S_i \in \mathbf{S}$ to the input of all the other GCN's, i.e. $Z_j^0 \forall S_j \in \mathbf{S} - S_i$ along with the original features. Thus, the input of each GCN is linear in $|\mathbf{S}|$.

- **Shared Latent Structure**: Attempts to transfer knowledge across relational views (modalities) with constraints on the representations (e.g. [105] aligns embeddings across views).

Ensemble methods introduced in [95] work on multi-relational edges in knowledge graphs. None of these approaches are directly suitable for our induced relationships. Our relational views utilize different label assignment semantics (label sharing within a clique vs. determine label based on contrast within a clique). In our label contrast semantics, we must achieve feature discrimination and label inversion between contrasting vertices, as opposed to label homogeneity and feature sharing in the label sharing case. Thus, aggregating relationships

by pooling, concatenation, or addition of vertex representations fail to capture semantic heterogeneity of the induced views. Further, data induced relations are uncurated and inherently noisy. Directly aggregating the learned representations via Stacking or Fusion can lead to noise propagation. We also expect views of the same relation type to be correlated.

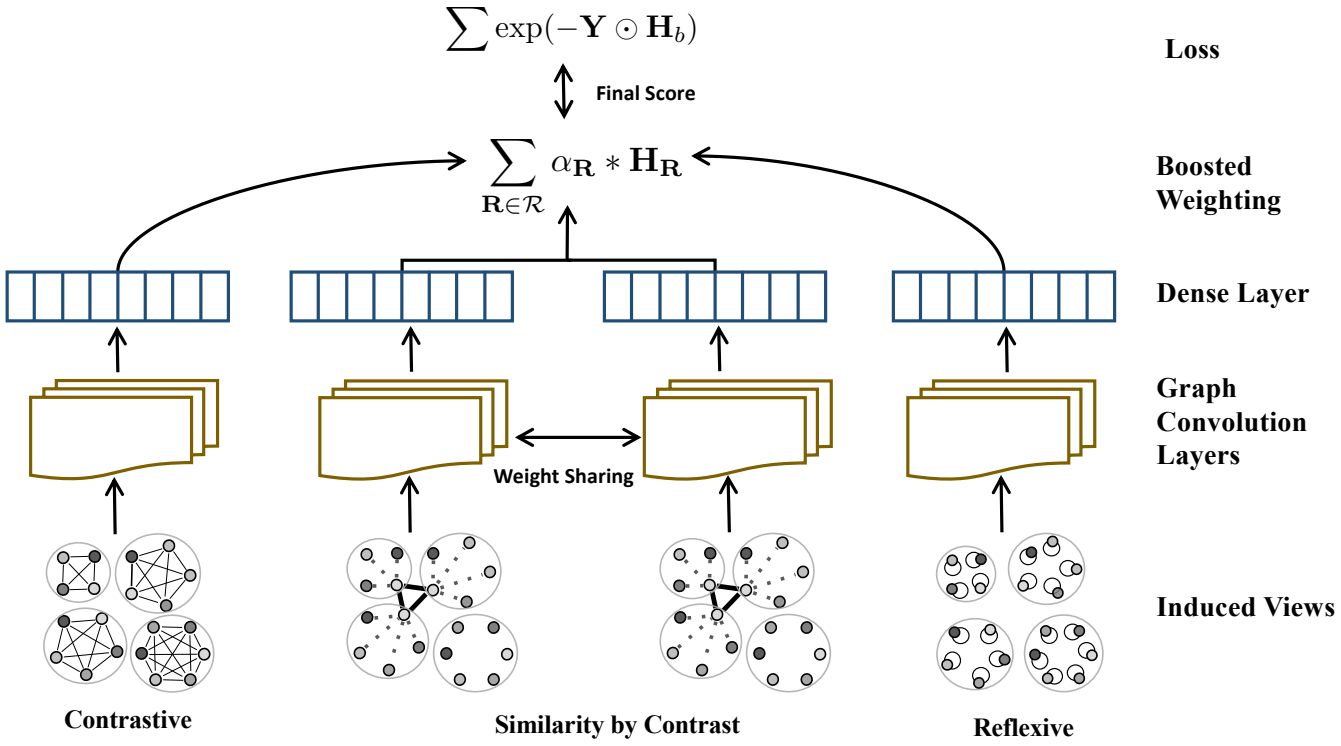


Figure 6.4: Schematic diagram of our proposed IR-GCN model.

We thus propose the following approach to aggregate information across relation types and between views of a relation type.

Cross-relation Aggregation: We expect distinct relation types to perform well on different subsets of the set of (q, a) tuples. We empirically verify this with the Jaccard overlap between the set of misclassified vertices under each relational view of a relation type on our dataset. Given $\mathbf{M}_A$ and $\mathbf{M}_B$, the sets of (q, a) tuples misclassified by GCNs A and B respectively, the jaccard overlap is,

$$\mathcal{J}_{A,B} = \frac{\mathbf{M}_A \cap \mathbf{M}_B}{\mathbf{M}_A \cup \mathbf{M}_B}$$

The $\mathcal{J}_{A,B}$ values are as follows for the relational pairings: (Contrastive, TrueSkill Similarity) = 0.42, (Contrastive, Reflexive) = 0.44 and (Reflexive, TrueSkill Similarity) = 0.48. Relatively low values of the overlap metric indicate uncorrelated errors across the relations.

Gradient boosting techniques are known to improve performance when individual classifiers, including neural networks [167], are diverse yet accurate. A natural solution then is

to apply boosting to the set of relation types and bridge the weaknesses of each learner. We employ Adaboost [168] to combine relation level scores, $\mathbf{H_R}$ (eq. (6.16)) in a weighted manner to compute the final boosted score, $\mathbf{H}_b \in \mathbb{R}^{N\times 1}$ representing all relation types (Line 12, algorithm 2). $\mathbf{Y} \in \mathbb{R}^{NX1}$ denotes the acceptance label of all tuples. Note that an entry in $(\mathbf{Y} \odot \mathbf{H_R}) > 0$ when the accepted label of the corresponding (q, a) tuple and sign of the prediction score, $sign(\mathbf{H_R})$, of relation type $\mathbf{R}$ match and < 0 otherwise. Thus, the weights $\alpha_{\mathbf{R}}$ adapt to the fraction of correctly classified tuples to the misclassified tuples by the relation $\mathbf{R}$ (Line 9, algorithm 2). The precise score computation is described in algorithm 2. We use the polarity of each entry in the boosted score, $sign(\mathbf{H}_b) \in \{-1, 1\}$, to predict the class label of the corresponding (q, a) tuple. The final score is also used to create a ranked list among all the candidate answers, $a \in \mathcal{A}(q)$ for each question, $q \in \mathcal{Q}$. $L_{(q,a)}$ represents the position of candidate answer a in the ranked list for question q.

Intra-relation Aggregation: Gradient boosting methods can effectively aggregate relation level representations, but are not optimal within a relationship type (since it cannot capture shared commonalities between different views of a relation type). For instance, we should facilitate information sharing between the TrueSkill similarity and Arrival similarity views. Thus, if an answer is authored by a user with a higher skill rating and answered significantly earlier than other answers, its probability to be accepted should be mutually enhanced by both signals. Empirically, we also found True Skill and Arrival Similarity GCNs to commit similar mistakes ($\mathcal{J}_{TS,AS} = 0.66$). Thus, intra-relation learning (within a single relation type like Similar Contrast) can benefit from sharing the structure of their latent spaces i.e., weight parameters of GCN.

Weight Sharing: For multiple views representing a relation type (e.g., TrueSkill and Arrival Similarity), we train a separate GCN for each view but share the layer-wise linear-transforms $\mathbf{W}_i^k$ to capture similarities in the learned latent spaces. Weight sharing is motivated by a similar idea explored to capture local and global views in [105]. Although sharing the same weight parameters, each GCN can still learn distinct vertex representations as each view convolves over a different neighborhood and employ random dropout during training. We thus propose to use an alignment loss term to minimize prediction difference between views of a single relation type[169]. The loss attempts to align the learned vertex representations at the *last layer* K (the loss term aligns pairs of final vertex representations, $||\mathbf{Z}_i^K - \mathbf{Z}_{i'}^K|| \ \forall \ S_i, S_i' \in \mathbf{R}$). In principle, multiple GCNs augment performance of the relation type by sharing prior knowledge through multiple Adjacency matrices ($\mathbf{A}_i \ \forall \ S_i \in \mathbf{R}$).

Training Algorithm: Algorithm 3 describes the training algorithm for our IR-GCN model. For each epoch, we first compute the aggregated prediction score $\mathbf{H}_b$ of our boosted model as described in algorithm 2. We use a supervised exponential loss $\mathcal{L}_b$ for training with elastic-

Algorithm 2 IR-GCN Boosted Score Computation

function FORWARD($\mathbf{X}, \mathbf{Y}, \{A_i\}_{S_i \in \mathbf{S}}$)
  $\mathbf{H}_b \leftarrow \mathbf{0}$
  for $\mathbf{R} \in \mathcal{R}$ **do**
    $\{\mathbf{Z}_i^K\}_{S_i \in \mathbf{R}} \leftarrow Conv(\mathbf{X}, \{A_i\}_{S_i \in \mathbf{R}})$
    ▷ Equation 6.7, 6.13, 6.15
    $\mathbf{H_R} = \sum_{S_i \in \mathbf{R}} \mathbf{Z}_i^K \times \widetilde{\mathbf{W}}_i$ ▷ Equation 6.16
    $\mathbf{e_R} \leftarrow \exp(-\mathbf{Y} \odot \mathbf{H}_b)$
    ▷ $\odot \rightarrow$ *Hadamard Product*
    $\alpha_{\mathbf{R}} \leftarrow \frac{1}{2} \ln \frac{\sum \mathbf{e_R} \odot \mathbb{1}\left((\mathbf{Y} \odot \mathbf{H_R}) > 0\right)}{\sum \mathbf{e_R} \odot \mathbb{1}\left((\mathbf{Y} \odot \mathbf{H_R}) < 0\right)}$
    ▷ $\sum \rightarrow$ *reduce-sum*
    ▷ $\mathbb{1}(.) \rightarrow$ *element-wise Indicator function*
    $\mathbf{H}_b \leftarrow \mathbf{H}_b + \alpha_{\mathbf{R}} * \mathbf{H_R}$ ▷ Update boosted GCN
  end for
  return $\mathbf{H}_b$, $\{\mathbf{H}_R\}_{\mathbf{R} \in \mathcal{R}}$, $\{\mathbf{Z}_i^K\}_{S_i \in \mathbf{S}}$
  ▷ Boosted scores, Relation level scores,
  ▷ Each GCN vertex representations
end function

net regularization (L1 loss - $\mathcal{L}_1(.)$ and L2 loss - $\mathcal{L}_2(.)$) on the graph convolutional weight matrices $\mathbf{W_i^k}\ \forall\ S_i \in \mathbf{S}$ for each view. Note that we employ weight sharing between all views of the same relation type so that only one set of weight matrices is learned per relation.

The exponential loss, $\mathcal{L}_{\mathbf{R}}$, for each relation type is added alternatingly to the boosted loss. We apply an *exponential annealing schedule*, $\lambda(t)$, i.e. a function of the training epochs (t), to the loss function of each relation. As training progresses and the boosted model learns to optimally distribute vertices among the relations, increase in $\lambda(t)$ ensures more emphasis is provided to the individual convolutional networks of each relation. Figure 6.4 illustrates the overall architecture of our IR-GCN model.

Algorithm 3 IR-GCN Training

Input: Input Feature Matrix X, Acceptance labels for each tuple, $\mathbf{Y}$, Adjacency matrix of each view $\{A_i\}_{S_i \in \mathbf{S}}$

Output: Trained Model i.e. Weight parameters $W_i^1 \dots W_i^k, S_i \in \mathbf{S}, \forall k \in [1, K]$ and transform parameters $\widetilde{W}_i$, $S_i \in \mathbf{S}$

for $t \leftarrow 1$ to *num-epochs* **do**
  $\mathbf{H}_b, \{\mathbf{H}_R\}_{\mathbf{R} \in \mathcal{R}}, \{\mathbf{Z}_i^K\}_{S_i \in \mathbf{S}} \leftarrow$ FORWARD$(X, Y, \{A_i\}_{S_i \in \mathbf{S}})$ ▷ Algorithm 2
  for $\mathbf{R} \in \mathcal{R}$ **do**
    $\mathcal{L}_b \leftarrow \sum \exp(-\mathbf{Y} \odot \mathbf{H}_b) + \gamma_1 \mathcal{L}_1(.) + \gamma_2 \mathcal{L}_2(.)$
    ▷ $\sum \rightarrow$ *reduce-sum*
    ▷ $\odot \rightarrow$ *Hadamard Product*
    $\mathcal{L}_{\mathbf{R}} \leftarrow 0$
    for $S_i \in \mathbf{R}$ **do**
      $\mathcal{L}_i \leftarrow \sum \exp(-\mathbf{Y} \odot \mathbf{H}_{\mathbf{R}})$
      $\mathcal{L}_{\mathbf{R}} \leftarrow \mathcal{L}_{\mathbf{R}} + \mathcal{L}_i + \frac{1}{2} \sum_{S_i' \neq S_i} ||\mathbf{Z}_i^K - \mathbf{Z}_{i'}^K||$
    end for
    $\mathcal{L}_b \leftarrow \mathcal{L}_b + \lambda(t) \mathcal{L}_{\mathbf{R}}$
    $W_i^k \leftarrow W_i^k + \eta_{\text{ADAM}} \frac{\partial \mathcal{L}_b}{\partial W_i^k}$ ▷ $\forall k \in [1, K], \forall S_i \in \mathbf{R}$
    $\widetilde{W}_i \leftarrow \widetilde{W}_i + \eta_{\text{ADAM}} \frac{\partial \mathcal{L}_b}{\partial \widetilde{W}_i}$ ▷ $\forall S_i \in \mathbf{S}$
  end for
end for

6.6 EXPERIMENTS

In this section, we first describe our dataset followed by our experimental setup; comparative baselines, evaluation metrics, and implementation details. We then present results across several experiments to evaluate the performance of our model on merging semantically diverse induced-relations.

6.6.1 Dataset

We first evaluate our approach on multiple communities catering to different topics from a popular online Community Question Answer (CQA) platform, *StackExchange*[4]. The platform divides the communities into five different categories, i.e. Technology (**T**), Culture/Recreation (**C**), Life/Arts (**L**), Science (**S**) and Professional (**P**). For our analysis, we collect data from the ten largest communities from each of the five categories until March 2019, resulting in a total of 50 StackExchange communities. The list of 50 StackExchange communities per category are;

- Technology: AskUbuntu, Server Fault, Unix, TEX, Electronics, Gis, Apple, Wordpress, Drupal, DBA
- Culture/Recreation: English, Travel, RPG, Judaism, Puzzling, Bicycles, German, Christianity, BoardGames, History
- Life/Arts: Scifi, DIY, Academia, Graphic Design, Money, Photo, WorldBuilding, Movies, Music, Law
- Science: Stat, Physics, MathOverflow, CS, Chemistry, Biology, Philosophy, CS Theory, Economics, Astronomy
- Professional/Business: Workplace, Aviation, Writers, Open source, Freelancing, CS Educators, Quant, PM, Parenting

In StackExchange, each questioner can mark a candidate answer as an "accepted" answer. We only consider questions with an accepted answer. Table 6.1 shows the final dataset statistics.

For each (q, a) tuple, we compute the following basic features:
Activity features : View count of the question, number of comments for both question and answer, the difference between posting time of question and answer, arrival rank of answer

[4]https://stackexchange.com/

	Technology			Culture/Recreation			Life/Arts		
	ServerFault	AskUbuntu	Unix	English	Games	Travel	SciFi	Home	Academia
$\|Q\|$	61,873	41,192	9,207	30,616	12,946	6,782	14,974	8,022	6,442
$\|\mathcal{A}\|$	181,974	119,248	33,980	110,235	45,243	20,766	49,651	23,956	23,837
$\|U\|$	140,676	200,208	84,026	74,592	14,038	23,304	33,754	30,698	19,088
$\mu(\|\mathcal{A}_q\|)$	2.94	2.89	3.69	3.6	3.49	3.06	3.31	2.99	3.7

	Science			Professional/Business		
	Physics	Maths	Statistics	Workplace	Aviation	Writing
$\|Q\|$	23,932	18,464	13,773	8,118	4,663	2,932
$\|\mathcal{A}\|$	65,800	53,772	36,022	33,220	14,137	12,009
$\|U\|$	52,505	28,181	54,581	19,713	7,519	6,918
$\mu(\|\mathcal{A}_q\|)$	2.75	2.91	2.62	4.09	3.03	4.10

Table 6.1: Dataset statistics for the top three Stack Exchange communities from five different categories. $|Q|$: number of questions; $|\mathcal{A}|$: number of answers; $|U|$: number of users; $\mu(|\mathcal{A}_q|)$: mean number of answers per question. Professional/Business communities have slightly more answers per question on average than others. Technology communities are the largest in terms of number of question out of the five categories.

(we assign rank 1 to the first posted answer) [53].

Text features : Paragraph and word count of question and answer, presence of code snippet in question and answer (useful for programming based forums), word count in the question title.

User features : Word count in user profile's Aboutme section for both users; one posting the question and other posting the answer.

Time-dependent features like upvotes/downvotes of the answer and user feature like reputation or badges used in earlier studies on StackExchange [51] are problematic for two reasons. First, we only know the aggregate values, not how these values change with time. Second, since these values typically increase over time, it is unclear if an accepted answer received the votes *prior* to or *after* an answer was accepted. Thus, we do not use such time-dependent features for our model and the baselines in our experiments.

Reddit[5] is another popular CQA platform with subreddits similar to StackExchange communities. In particular, we focus on Ask* subreddits as they are primarily used to seek help from a community of experts and non-experts. In particular, we crawled data from /r/askscience (science forum), /r/AskHistorians (history forum), and /r/AskDocs (medi-

[5]https://www.reddit.com/

cal forum) until October 2017. We performed basic preprocessing and removed posts or comments with single word/URLs or missing author/title information. We also removed infrequent users who posted less than two comments. Reddit has a hierarchical comment structure. For this paper, we treat first-level comments as potential answers to the question. Users in these subreddits can get verified by providing anonymized verification documents including certification numbers, contact information, etc. to the moderators. We denote these verified users as experts. We treat an expert's comment as equivalent to an accepted answer and only consider posts which have an expert answer for our experiments. We discard posts with multiple experts' comment as it is hard to objectively choose a winner.

Dataset	$\lvert\mathcal{Q}\rvert$	$\lvert\mathcal{A}\rvert$	$\lvert\mathcal{U}\rvert$	$\mu(\lvert\mathcal{A}_q\rvert)$
AskDocs	11189	29207	4530	2.61
AskHistorians	15425	45586	11761	2.96
AskScience	37990	121278	32117	3.19

Table 6.2: Dataset statistics for the Ask* Reddit communities. $|Q|$: number of questions; $|\mathcal{A}|$: number of answers; $|U|$: number of users; $\mu(|\mathcal{A}_q|)$: mean number of answers per question.

We employ 12 basic features for the Reddit dataset:
Activity features : ArrivalRank of the answer, Number of subsequent comments on the answer, Number of other answers to the question, Upvotes and downvotes for both, question and answer.
Text features : Word count of the question and answer

We employ post-vote features here as [170] showed that there is widespread under-provision of voting on Reddit, partially due to long comment threads. It can act as a weak signal for answer quality. Unlike the StackExchange, Reddit voting is not biased by publicly visible acceptance of answers to a question. Thus, votes ideally represent the independent judgment of the crowd.

6.6.2 Experimental Setup

Baselines

We compare against state-of-the-art feature-based baselines for answer selection and competing aggregation approaches to fuse diverse relational views of the dataset [105, 95].
Random Forest (RF) [51, 53] model trains on the feature set mentioned earlier for each dataset. This model is shown to be the most effective feature-based model for Answer

Selection.

Feed-Forward network (FF) [52] is used as a deep learning baseline to learn non-linear transformations of the feature vectors for each (q, a) tuple. This model is equivalent to our Reflexive GCN model in isolation.

Dual GCN (DGCN) [105] trains a separate GCN for each view. In addition to the supervised loss computed using training labels, they introduce a regularizer to minimize mean squared error (MSE) between vertex representations of two views, thus aligning the learned latent spaces. Formally, For instance,

$$\mathcal{L}_{reg}(Z_c, Z_{ts}) = \frac{1}{n} \sum_{j \in 1,n} \|Z_c^j - Z_{ts}^j\|$$

computes the MSE loss between Contrastive and TrueSkill Similarity GCN. [105] proposed the model for two GCN representations. We extend the original two GCN model [105] to four GCN, each representing our relational views between the nodes. We minimize the supervised loss with respect to the best performing GCN model (Contrastive), and its node representation's alignment with respect to all other GCN's node representations. The Contrastive view is seen to exhibit the best performance in isolation. Thus, the DualGCN loss can be given by:

$$\mathcal{L} = \mathcal{L}_0 + \lambda(t) \left(\sum_{S_i \in \mathbf{S}, S_i \neq c} \|\mathbf{Z}_c^K - \mathbf{Z}_i^K\| \right)$$

where $\mathcal{L}_0$ represents the supervised loss and $\mathbf{Z}_c^K$ is the node representations of the Contrastive GCN. The regularizer loss is similar to our intra-relation aggregation approach but assumes label and feature sharing across *all* the views.

Relational GCN (RGCN) [95] combines the output representations of previous layer of each view to compute an aggregated input to the current layer. $\mathbf{Z}_i^{k-1}$ of layer $k-1$ of each view to compute an aggregated input to layer k. Formally,

$$\mathbf{Z}_{rgcn}^k = \sigma \left(\sum_{S_i \in \mathbf{S}} \mathbf{Z}_i^{k-1} \right)$$

where Z_{rgcn} is final output of this model at layer k and σ is the activation function.

We also report results for each view individually: Contrastive (C-GCN), Arrival Similarity (AS-GCN), TrueSkill Similarity (TS-GCN) and Reflexive (R-GCN) with our proposed IR-GCN model. We do not compare with other structure-based approaches to compute vertex representations [97, 98, 99, 100] as GCN is shown to outperform them [9]. We also compare with common aggregation strategies to merge neural representations discussed earlier in

section 6.5 later.

Evaluation Metric

We randomly select 20% of the questions, $\mathbf{T}_q \subset \mathcal{Q}$ to be in the test set. Then, subsequently all (q, a) tuples such that $q \in \mathbf{T}_q$ comprise the set of test tuples or vertices, $\mathbf{T}$. The rest of the vertices, along with their label information, is used for training the model. We evaluate our model on two metrics, Accuracy and Mean Reciprocal Rank (MRR). Accuracy metric is widely used in vertex classification literature while MRR is popular for ranking problems like answer selection. Formally,

$$Acc = \frac{1}{|\mathbf{T}|} \sum_{(q,a)\in\mathbf{T}} \mathbb{1}\left(y_{(q,a)} \cdot h_b((q,a)) > 0\right)$$

with $\cdot$ as the product and $\mathbb{1}$ as the indicator function. The product is positive if the accepted label and predicted label match and negative otherwise.

$$MRR = \frac{1}{|\mathbf{T}_q|} \sum_{q\in\mathbf{T}_q} \frac{1}{\sum_{a'\in\mathcal{A}(q)} \mathbb{1}\left(L_{(q,a)} < L_{(q,a')}\right)}$$

where $L_{(q,a)}$ refers to the position of accepted answer a in the ranked list for question q [171].
Implementation Details We implemented our model and the baselines in Pytorch. We use ADAM optimizer [172] for training with 50% dropout to avoid overfitting. We use four hidden layers in each GCN with hidden dimensions 50, 10, 10, 5, respectively, and ReLU activation. The coefficients of $\mathcal{L}_1$ and $\mathcal{L}_2$ regularizers are set to $\gamma_1 = 0.05$ and $\gamma_2 = 0.01$ respectively. For TrueSkill Similarity, we use margin $\delta = 4$ to create links, while for Arrival similarity, we use $\delta = 0.95$. We implement a mini-batch version of training for large graphs where each batch contains a set of questions and their associated answers. This is equivalent to training on the whole graph as we have disconnected cliques.

6.6.3 Performance Analysis

Table 6.4 shows impressive gains over state-of-the-art baselines for all the five categories of StackExchange. We report mean results for each category obtained after 5-fold cross-validation on each of the communities. Our induced-relational GCN model beats best performing baseline by 4-5% on average in accuracy. The improvement in MRR values is around 2.5-3% across all categories. Note that MRR is based only on the rank of the accepted answer

while accuracy is based on correct labeling of *both* accepted and non-accepted answers.

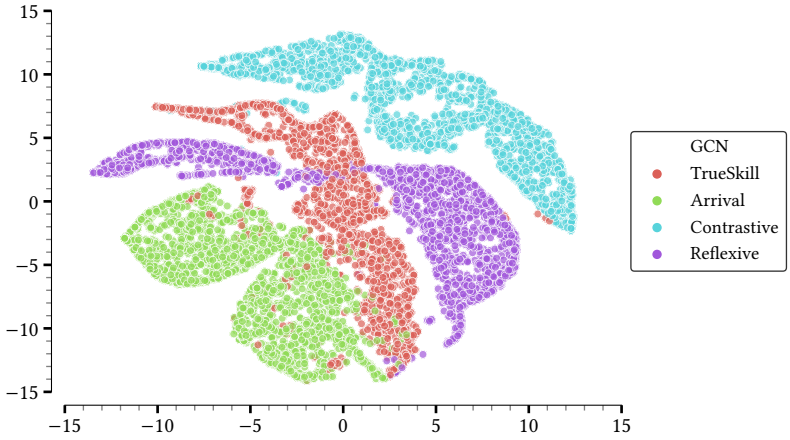


Figure 6.5: t-stochastic neighbor embedding (t-SNE) [173] distributions of the learned vertex representations by our model for Chemistry StackExchange. Each view learns a distinct vertex representation. Best viewed in color.

Among individual views, Contrastive GCN performs best on all the communities. It even beats the best performing baseline DualGCN that uses all the relational views. Note that contrastive view compares between the candidate answers to a question and uses our proposed contrastive modification to the convolution operation. Arrival Similarity follows Contrastive and then Reflexive. The superior performance of Arrival Similarity view shows that early answers tend to get accepted and vice versa. It indicates that users primarily use CQA forums for quick answers to their queries. Also, recall that Reflexive predicts each vertex's label independent of other answers to the same question. Thus, the competitive performance of Reflexive strategy indicates that vertex's features itself are well predictive of the label. TrueSkill Similarity performs at par or slightly worse than Reflexive. Figure 6.5 presents t-SNE distributions [173] of the learned vertex representations ($\mathbf{Z}_i^K$) of our model applied to Chemistry StackExchange from Science category. Note that each view, including two views under Similar Contrast relation, learns a distinct vertex representation. Hence, all views are essential and contribute to our final performance.

Out of the baseline graph ensemble approaches, DualGCN performs significantly better than RelationalGCN by an average of around 26% for all categories. Recall that in RelationalGCN model, the convolution output of each view is linearly combined to compute the final output. Linear combination works well for knowledge graphs as each view can be thought of as a feature, and then it accumulates information from each feature. DualGCN is similar to our approach and trains different GCN for each view and later merges their results. However, it enforces similarity in vertex representations learned by each view. This

restriction is not suitable for our induced-relationships as they are semantically different (contrastive captures contrast in features vs. similarity enforces label sharing).

Table 6.5 shows performance gains over the state-of-art baselines for the Reddit dataset. All results are reported after 5-fold cross validation. Our model improves by 16% on average in accuracy over the baselines for Reddit. The improvement in MRR is again higher for Reddit at an average increase of 7% than the baseline.

Among individual views, for Reddit, there is a huge difference in performance for each induced-relational GCN. TrueSkill Similarity performs much better followed by Arrival Similarity and Contrastive. Reflexive GCN performs the worst for Reddit as it predicts each node's label independent of answers to the same question.

Out of the baseline graph ensemble approaches, DualGCN and RelationalGCN, similar to StackExchange, DualGCN consistently performs better than RelationalGCN by an average of around 3% for Reddit.

6.7 DISCUSSION

In this section, we first evaluate importance of each relational view for our boosted model. We then compare with approaches proposed to merge neural networks in general in other domains. We then illustrate *discriminative magnification effect* in detail and study the robustness of our model to training label sparsity. We also extend our proposed approach to include textual features and compare it with text-based baseline. Finally, we provide a theoretical analysis of performance gains of our Contrastive GCN model. We also provide limitations of our approach.

6.7.1 Ablation Study on Relation Types

We present results of an ablation study with different combination of relation types (Contrastive, Similar and Reflexive) used for IR-GCN model in Table 6.6. We conducted this study on the biggest community from each of the five categories, i.e., ServerFault (Technology), English (Culture), Science Fiction (Life), Physics (Science), Workplace (Business). We also report results for AskDocs Reddit. Similar Contrast relation (TrueSkill and Arrival) used in isolation perform the worst among all the variants. Training Contrastive and Similar Contrast relation together in our boosted framework performs similar to our final model. Reflexive GCN contributes the least as it does not consider any neighbors.

6.7.2 Aggregator Architecture Variants

We compare our gradient boosting based aggregation approach with other popular methods used in literature to merge different neural networks discussed in section 6.5.

Table 6.7 reports the accuracy results for these aggregator variants as compared to our model. Our method outperforms all the variants with Fusion performing the best. This reaffirms that existing aggregation models are not suitable for our problem. Note that these approaches perform worse than even Contrastive GCN except Fusion. Fusion approach performs similar to our approach but is computationally expensive as the input size for each IR-GCN is linear in the number of all views in the model.

6.7.3 Discriminative Magnification effect

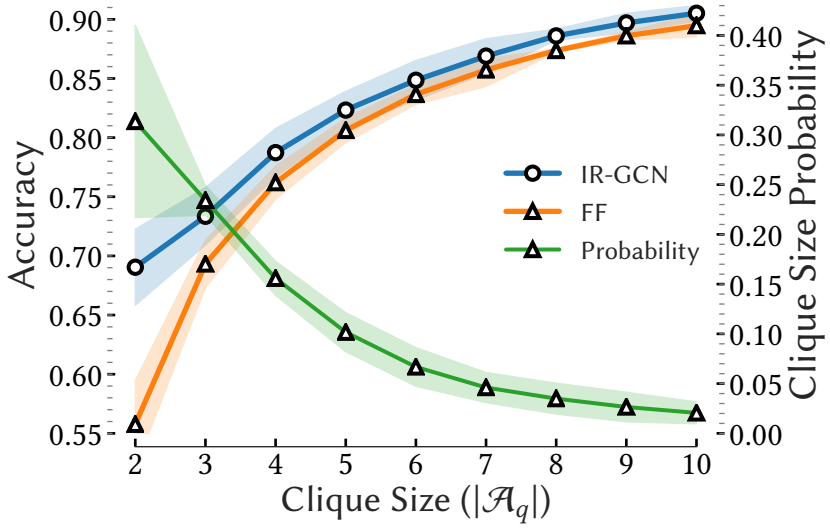


Figure 6.6: Accuracy of our IR-GCN model compared to the FF model with varying clique size (i.e. number of answers to a question, $|\mathcal{A}_q|$) for Contrastive view . We report averaged results over the largest community of all categories. Our model performs much better for smaller cliques, and the effect diminishes for larger cliques (eq. (6.7)). 80% of the questions have < 4 answers.

We show that due to our proposed modification to the convolution operation for contrastive view, we achieve *Discriminative Magnification effect* (eq. (6.7)). Note that the difference is scaled by Clique size $(1+1/n-1)$, i.e. number of answers to a question, $|\mathcal{A}_q|$. Figure 6.6 shows the accuracy of our IR-GCN model as compared to the FeedForward model with varying clique size. Recall that FeedForward model predict node labels independent of other nodes and is not affected by clique size. We report average results over the same five communities

as above. We can observe that increase in accuracy is much more for lower clique sizes (13% improvement for $|\mathcal{A}_q| = 2$ and 4% for $|\mathcal{A}_q| = 3$ on average). The results are almost similar for larger clique sizes. In other words, our model significantly outperforms the FeedForward model for questions with fewer candidate answers. However, around 80% of the questions have very few answers(< 4) and thus this gain over FF is significant.

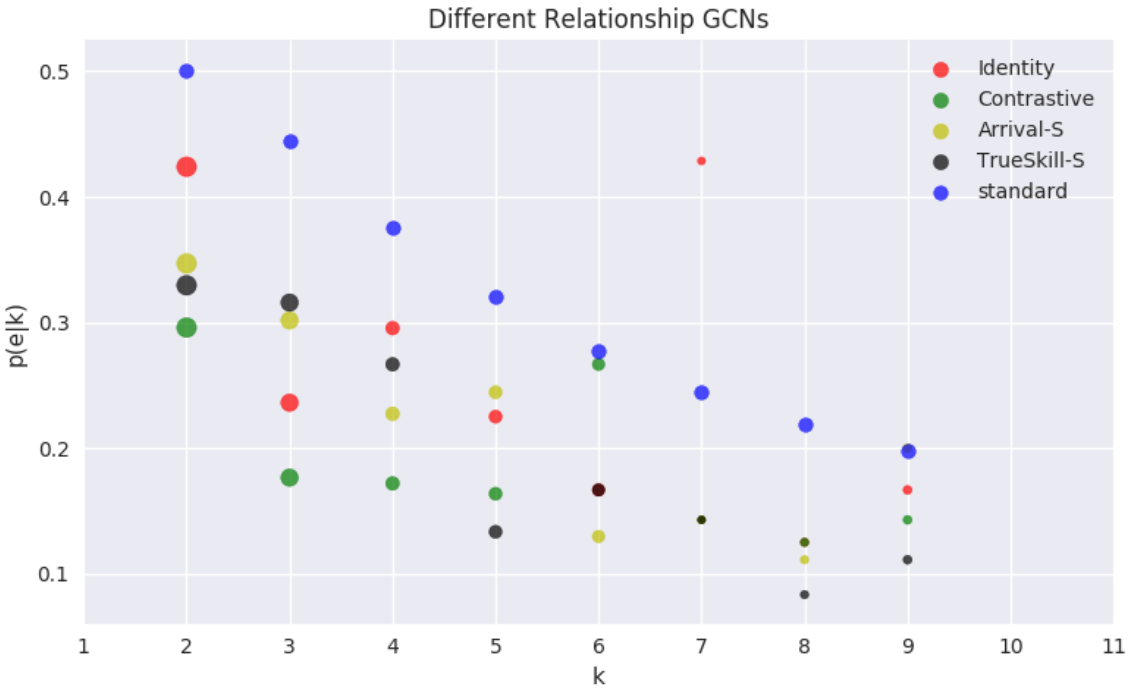


Figure 6.7: Probability of error with varying clique size for movie StackExchange. Standard represents random selection. Contrastive view outperforms other views for smaller clique sizes.

Alternatively, we also plot the probability of error per tuple given each clique size ($p(e|k)$) for movie StackExchange in Figure 6.7. The *standard* corresponds to a naive baseline of randomly selecting an accepted answer within each clique. For this standard baseline, error probability per clique can be denoted as,

$$p(e|k) = (1 - 1/k) * \frac{2}{k} = \frac{2(k-1)}{k^2}$$

$(1 - 1/k)$ denotes the probability of choosing the wrong accepted answer, while $2/k$ is the actual error rate in these scenarios. The error rate is such because even in cases where the baseline chose the wrong accepted answer, remaining answers are still correctly classified as not accepted.

The standard baseline performs the worst as the error probability is highest than the other baselines for each clique. The Contrastive view has the least error probability for smaller cliques ($k < 5$). This is analogous to the performance gain illustrated above due to the *Discriminative Magnification effect*. For larger cliques, similar contrast views (ArrivalSkill and TrueSkill) have the least error probability. As both of these views connects similar tuples across different questions, they are thus more useful for questions with larger number of competing answers.

6.7.4 Label Sparsity

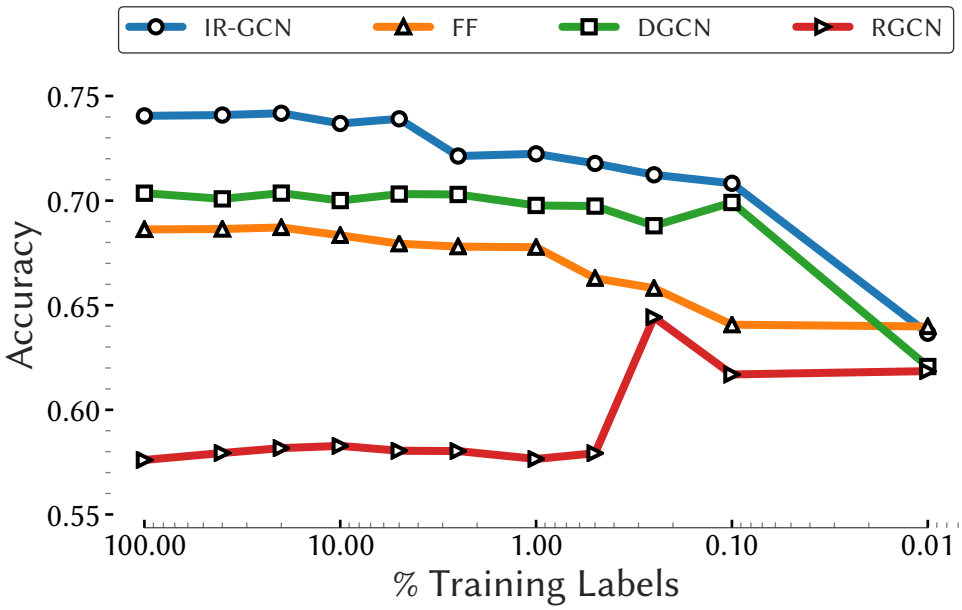


Figure 6.8: Change in accuracy with varying training label rates for Physics StackExchange. Our model is more robust to label sparsity than other relation ensemble approaches. RGCN works better with fewer labels as contrastive relation introduces noise in the model. At extreme sparsity, all approaches converge to the same value indicating random selection.

Graph Convolution Networks are robust to label sparsity as they exploit graph structure and are thus heavily used for semi-supervised settings. Figure 6.8 shows the change in accuracy for Physics StackExchange from Science category at different training label rates. Even though our graph contains disconnected cliques, IR-GCN still preserves robustness to label sparsity. In contrast, the accuracy of FeedForward model declines sharply with less label information. Performance of DualGCN remains relatively stable while Relational GCN's performance increases with a decrease in label rate. Relational GCN assumes each view to be of similarity relation and thus, adding contrastive relation introduces noise in the model.

However, as the training labels become extremely sparse, the training noise decreases that leads to a marked improvement in the model. In case of extremely low label rate of 0.01%, all approaches converge to the same value, which is the expectation of theoretically random selection. We also obtained similar results for other four StackExchange communities.

6.7.5 Including Textual Features

Most of the current literature focuses on using text similarity for Answer Selection. In this section, we compare our proposed IR-GCN model to a popular text-based model [174] for answer selection. For this experiment, we first preprocessed the text of both question and answers.

Text Preprocessing: We first removed all code snippets, HTML tags, stopwords and URLs from the text of all questions and answers. We then tokenized the text using NLTK tokenizer followed by lemmatization using WordNetLemmatizer and finally converted it into lowercase.

We use torchtext to create vocabulary and limit the text of each question and answer to be 250 words long. We initialized the words in the vocabulary using 300-dimensional pre-trained embeddings from Word2vec (`https://code.google.com/archive/p/word2vec/`). We randomly intiailized words present in the vocabulary but not in word2vec.

We evaluate multiple approaches to test the effectiveness of including textual features. **QA-LSTM/CNN [174]** uses a stacked bidirectional LSTM model followed by convolution filters to extract embeddings for question and answer text separately. Answers are then classified according to the cosine similarity of learned question and answer embedding.

Specifically, in this baseline, we use a biLSTM model with hidden dimension = 300 followed by 50 1D convolutional filters with a kernel size of 3. We then compute the final embeddings by applying 1D maxpooling on the output of the convolution layer. We also used Tanh nonlinearity and dropout of 0.3 on the final embeddings. We finally use these embeddings to compute a cosine simlarity score between a question and its answers. This score is used to rank the candidate answers for evaluation. We implemented the baseline in Pytorch.

Textual Similarity (T-GCN) We create a *SimilarContrast* view that connects answers authored by a user where her answer is significantly similar (dissimilar) to the question than other competing answers. We used cosine similarity on the learned question and answer embedding from the QA-LSTM/CNN approach as the similarity function.

Specifically, we extract the updated embeddings of the question and answer text from the learnt QA-LSTM model. We then compute cosine similarity between the embeddings of each question and its answers. We then connect answers authored by a specific user, where

the difference in cosine similarity of the answer with the other competing answers is greater than margin λ. Specifically, if the user authors answers a, a' to questions q, q', we create a link between a and a' if

$$|C_{q,a} - C_{q,b}| > \lambda; \forall b \in \mathcal{A}_(q)$$
$$|C_{q,a'} - C_{q,c}| > \lambda; \forall c \in \mathcal{A}_(q')$$

where $C_{q,a}$ is the cosine similarity of the answer a with respect to question q. Similarly, a link is created for the opposite case when difference is less than $-\lambda$. In our experiments, we assign $\lambda = 0.4$. The hypothesis is that irrelevant(dissimilar) answers will more likely be rejected and vice versa.

IR-GCN + T-GCN extends our proposed model to also include the Textual Similarity as the third *SimilarContrast* view in addition to Arrival and TrueSkill Similarity.

In general, the text-based baseline, QA-LSTM peforms worse than even reflexive GCN as shown in Table 6.8. Note that reflexive GCN employs a feedforward model on the activity and user features used in our experiments. This is a surprising result as most of the current literature focus on textual features for the task. Our results indicate that non-textual features are useful too for answer selection task on StackExchange communities.

Textual Similarity GCN performs better than QA-LSTM and Reflexive GCN. Even though we use the output of QA-LSTM to construct the graph for T-GCN, the graph improves performance as it connects answers across questions. However, adding T-GCN view in our proposed IR-GCN model decreases the performance slightly. One possible explanation could be that similar contrast views based on user features (Arrival similarity and TrueSkill similarity) are not compatible with views based on textual features.

We further replaced our activity based features with the learned embeddings obtained after training the QA-LSTM/CNN [174] model as the node features. We observed that performance of all approaches went down slightly when using textual features only (Table 6.9). As we noted before, GCNs aggregate features among the neighbors. In our similar contrast views, it is not favorable to aggregate textual features among the neighbors as it connects answers catering to different questions. Thus, aggregating textual features creates noise in the model leading to worse performance [6].

[6]We also experimented with concatenating textual features with the original features used in the previous experiments. However, the performance was still little worse than the results with only original features.

6.7.6 Contrastive GCN Analysis

The ability of neural networks to perform classification in sparse high-dimensional manifolds has been studied in past work, especially in the context of adversarial learning [175]. We employ the ReLU activation function in our convolution layers and study the outputs of the kth layer, i.e. embeddings with k-order locality. This transformation breaks the input space into cells with smooth gradients within each cell, at whose boundaries the piecewise linear function changes (i.e. the likelihood of the two classes of answers).

We ask a specific question in the context of our Contrastive IR-GCN. *What is the impact of the layerwise discriminative magnification induced by our formulation?* Discriminative magnifications results in improved separability of the two classes in the later convolving layers, an effect we earlier demonstrated with a sample network in fig. 6.3. This positively impacts the ability of the model to explain the observed data points (i.e create p-domains that are well aligned with the contrastive samples provided) and improve the generalizability of the learned model to unseen data points. However, it is important to maintain sufficient regularization with weight decay to prevent sparse regions exhibiting sharp gradients which could affect model performance.

The capacity of our model can also be quantified in terms of the VC dimension of the aggregated classifier against the individual learners. Gradient boosting with multiple relation learners (each of which captures a specific aspect of node locality via graph convolution on the induced relations) could boost the capacity of the joint model, enabling better generalization and a more accurate fit in the data manifold (i.e. higher capacity to fit regions to fine distinctions).

Let us denote the upper bound of the VC dimension or capacity of each individual learner as D (If the individual learners do not have identical capacity, the minimum can be used to compute a lower bound on the aggregated learner capacity). Then the gradient boosted learner with T classifiers has a bound on it's capacity [176] given by,

$$\mathcal{VC}_{Agg} = T \times (D+1) \times (3\log(T.(D+1)) + 2)$$

Thus we identify two potential reasons for our performance gains, first the discriminative magnification effect which also supports the strong individual performance of the contrast view, and second the gain in capacity from boosting, which could explain it's advantage over competing aggregation methods.

6.7.7 Limitations

We do recognize certain limitations of our work. First, we focus on equivalence relations that induce a graph comprising cliques. While cliques are useful graph objects for answer selection, equivalence relations may be too restrictive for other problems (e.g., the relation is not transitive). However, our modular framework does apply to arbitrary graphs, except that Equation (6.3) will no longer be an *exact* convolution but be an approximation. Second, we assume no evolution in author skills. This assumption is not true as users evolve with experience. We aim to address this in future work.

In summary, our model showed significant gains over state-of-the-art baselines for combining information from semantically different relational links in a graph. Our model is also more robust to training label sparsity as compared to other aggregator GCN approaches. We reasoned that the performance gains achieved by our aggregation strategy can be attributed in part to the enhanced learning capacity of the boosted model and the effect of discriminative feature magnification. We showed that content can also be used to induce graphs and performs better than using content features in isolation. Finally, we presented a few limitations and possible future extensions.

Method	**Technology**		**Culture/Recreation**		**Life/Arts**	
	Acc(%)	MRR	Acc(%)	MRR	Acc(%)	MRR
RF [51, 53]	66.78±0.023	0.683±0.043	72.5±0.018	0.626±0.050	72.71±0.049	0.628±0.089
FF [52]	67.31±0.027	0.786±0.022	72.22±0.020	0.782±0.023*	73.58±0.049	0.780±0.034
DGCN [105]	70.70±0.022	0.782±0.017	75.22±0.017	0.771±0.028	76.73±0.034	0.784±0.038
RGCN [95]	54.40±0.045	0.673±0.045	60.39±0.016	0.645±0.042	59.97±0.043	0.654±0.054
AS-GCN	67.76±0.032	0.775 ±0.015	73.05 ±0.021	0.763±0.025	73.79 ±0.048	0.776±0.042
TS-GCN	66.87±0.032	0.779±0.018	72.16±0.023	0.764±0.023	72.02±0.061	0.765±0.048
C-GCN	71.64±0.022*	0.790±0.015*	76.18±0.017*	0.781±0.024	77.37±0.034*	0.788±0.040*
IR-GCN	**73.96±0.023**	**0.794±0.014**	**78.61±0.018**	**0.790±0.025**	**79.21±0.032**	**0.800±0.037**

Method	**Science**		**Professional/Business**	
	Acc(%)	MRR	Acc(%)	MRR
RF [51, 53]	68.09±0.024	0.692±0.049	74.72±0.044	0.5951±0.081
FF [52]	67.87±0.024	0.800± 0.028	74.63±0.040	0.759±0.049
DGCN [105]	71.45±0.023*	0.791±0.035	76.86±0.031	0.751±0.046
RGCN [95]	58.65±0.054	0.682±0.042	63.02±0.038	0.657±0.061
AS-GCN	66.93±0.045	0.788 ±0.028	74.99±0.045	0.742 ±0.047
TS-GCN	65.90±0.042	0.790±0.031	74.17±0.046	0.747±0.044
C-GCN	70.81±0.042	0.800±0.032*	77.57±0.038*	0.768±0.034*
IR-GCN	**74.98±0.021**	**0.808±0.028**	**80.17±0.026**	**0.785±0.032**

* DGCN stands for DualGCN, RGCN stands for RelationalGCN, and IR-GCN stands for Induced Relational GCN.

Table 6.4: Accuracy and MRR values for StackExchange with state-of-the-art baselines. Our model outperforms by at least 4% in Accuracy and 2.5% in MRR. Contrastive GCN performs best among individual views. The model with $*$ symbol has the second-best performance among all other models. Our model shows statistical significance at level 0.01 overall second best model on single tail paired t-test.

Method	AskDocs		AskHistorians		AskScience	
	Acc (%)	MRR	Acc(%)	MRR	Acc(%)	MRR
RF [51, 53]	59.35	0.698	65.62	0.709	65.87	0.706
FF [52]	62.30	0.715	67.89	0.7302	68.99	0.713
DGCN [105]	77.54	0.790	80.49	0.805	75.57	0.821
RGCN [95]	57.98	0.667	64.56	0.684	62.42	0.642
AS-GCN	76.53	0.794	80.70	0.781	78.14	0.797
TS-GCN	84.44	0.861	90.95	0.829	87.61	0.822
C-GCN	67.39	0.753	70.57	0.744	71.11	0.769
IR-GCN	**87.60**	**0.896**	**93.81**	**0.851**	**89.11**	**0.837**

Table 6.5: Accuracy and MRR values for Ask Reddits. Our model significantly outperforms by 16% in Accuracy and 7% in MRR. TrueSkill Similarity performs best among individual IR-GCNs.

{ Relation Type}	**Tech**	**Culture**	**Life**	**Sci**	**Business**	**AskDocs**
C	71.23	75.90	78.71	72.99	76.85	67.39
{ TS, AS }	67.86	74.15	75.75	65.80	76.13	84.57
R	68.30	73.35	76.57	67.40	75.76	62.30
{TS, AS } + R	69.28	75.50	76.41	70.11	77.90	86.34
C + R	73.04	77.66	80.25	73.72	80.04	70.02
C + { TS, AS }	72.81	78.04	81.41	72.19	80.15	86.99
C + { TS, AS } + R	**73.87**	**78.74**	**81.60**	**74.68**	**80.56**	**87.60**

Table 6.6: 5-fold Accuracy (in %) comparison for different combination of relation types for our boosted model. Contrastive and Similar Contrast relations together performs similar to the final model.

Method	**Tech**	**Culture**	**Life**	**Sci**	**Business**	**AskDocs**
Stacking [165]	68.58	74.44	79.19	70.29	75.50	85.40
Fusion [166]	72.30	77.25	80.79	73.91	79.01	86.33
NeighborAgg [104, 95]	69.29	74.28	77.94	68.42	78.64	86.00
IR-GCN	**73.87**	**78.74**	**81.60**	**74.78**	**80.56**	**87.60**

Table 6.7: 5-fold Accuracy (in %) comparison of different aggregator architectures. These architectures perform worse than Contrastive GCN for StackExchange. Fusion performs similarly but is computationally expensive.

Method	**Tech**	**Culture**	**Life**	**Sci**	**Business**
QA-LSTM/CNN[174]	66.49	71.70	69.42	62.91	72.55
FF [52]	68.30	73.35	76.57	67.40	75.76
C-GCN	71.23	75.90	78.71	72.99	76.85
T-GCN	69.25	73.77	76.39	67.79	77.08
IR-GCN	73.87	78.74	81.60	74.68	80.56
IR-GCN + T-GCN	73.89	78.00	81.07	74.49	78.86

Table 6.8: 5-fold Accuracy comparison of text-based baseline and textual similarity GCN with IR-GCN.

Method	**Tech**	**Culture**	**Life**	**Sci**	**Business**
QA-LSTM/CNN[174]	66.49	71.70	69.42	62.91	72.55
FF [52]	66.00	72.22	69.85	63.63	75.57
C-GCN	66.19	72.45	70.23	63.89	75.71
CT-GCN	66.06	72.35	71.88	64.14	75.69
IR-GCN	66.56	72.92	72.54	65.11	75.95
IR-GCN + T-GCN	66.49	73.17	72.85	65.29	75.86

Table 6.9: 5-fold Accuracy comparison of text-based baseline and textual similarity GCN with learnt text embeddings as features in the GCN.

6.8 CONCLUSION

This paper addressed the question of identifying the accepted answer to a question in CQA forums. We developed a novel induced relational graph convolutional (IR-GCN) framework to address this question. We made three contributions. First, we introduced a novel idea of using strategies to induce different views on (q, a) tuples in CQA forums. Each view consists of cliques and encodes—reflexive, similar, contrastive—relation types. Second, we encoded label sharing and label contrast mechanisms within each clique through a GCN architecture. Our novel contrastive architecture achieves *Discriminative Magnification* between nodes. Finally, we show through extensive empirical results on StackExchange that boosting techniques improved learning in our convolutional model. This was a surprising result since much of the work on neural architecture that are strong learners focuses on stacking, fusion or aggregator architectures. However, boosting is traditionally shown to be most effective with weak learners. Our ablation studies show that the contrastive relation is most effective individually in StackExchange.

CHAPTER 7: MODELING USER LATENT BEHAVIOR WITH SOCIAL INFLUENCE

In this chapter, we jointly estimate user latent behavior with the effect of social influence on their behavior. Specifically, we estimate the abusive behavior of users, i.e., likelihood of posting an offensive content, on Twitter. Different from text representation approaches used in Chapter 4 and Chapter 6, we learn an improved text representation that is suitable to capture nuanced hate speech in text. We further propagate user's abusive behavior through their social connections on Twitter to capture homophily in abusive user accounts. We finally show that combining the text and user representations can greatly improve offensive language prediction on future tweets [177].

7.1 OVERVIEW

Abusive language usage in online social media is a grave issue that is affecting the interactions of users online. In a study conducted by Pew Research Center[1], 40% of adult Internet users have personally experienced harassment online, and 60% have witnessed offensive name-calling. Social media websites, like Twitter and Facebook, allow users to report harassing content. However, due to the sheer volume of data, timely human curation of all reported content is not possible. Besides, there is also a need to filter these hateful content proactively. Therefore, there is an increased interest in automatic detection and moderation of hate speech in natural language processing [78].

A typical definition of hate speech is as an *attack* targeted towards a particular *individual* or *entity* belonging to a protected group (protected group may include, but are not always limited to, religious, gender or racial minorities) [178]. Thus, hate speech identification can be cast as a relation extraction problem in which the goal is to detect a "hate" or "attack" relation that links the speaker to a protected group (the object of the attack).

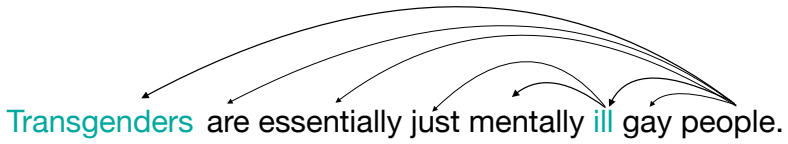


Figure 7.1: Dependency parse for a sample hate tweet. The target *Transgenders* is closer to the attack word, *ill* in the parse tree.

Current state-of-the-art methods in hate speech identification use either character or word

[1] http://www.pewinternet.org/2014/10/22/online-harassment/

n-gram features [78, 79] or employ sequential deep learning models like CNN or LSTM [87, 85]. However, these methods do not work well to capture long-range dependencies, for instance, in longer sentences with long clauses or complex scoping. Large pre-trained language models [1] achieve very high accuracy after fine-tuning on supervised tasks. However, they typically learn pairwise attention between all words in the text and are thus computationally expensive. This makes them unfit to be used efficiently for real-time detection.

Recent work by Clark et al. [179] analyzed attention mechanisms of the pre-trained BERT model and found that some of the attention heads are learning syntactic dependencies between words like direct objects of verbs, determiners of nouns, etc. A dependency parser also analyzes the grammatical structure of the sentence and returns the structure of syntactic dependence between words in the sentence. Recently, Zhang et al. [180] used words in the dependency parse path between the subject and object of the sentence for extracting relations between them. They encoded the dependency parse graph using efficient graph convolution operations. Graph convolutional networks [181] have been proposed for efficient convolutions over graph structured data. They are easy to parallelize and are computationally efficient.

However, a direct usage of Zhang et al. [180]'s method is not straightforward because of the complexity of the possibilities for expressing the attack in text. An attack may be expressed using explicit slurs or curses, as in: *RT @USER: Stupid f*cking n*gger LeBron. You flipping jungle bunny monkey f*ggot*, or not, as in: *Transgenders are essentially just mentally ill gay people*. In other instances, the attack can be *implicit* like *RT @USER: @USER every guy knows that the only thing that will make a woman happy is making any man a sandwich #notsexist.*. Moreover, the online text does not have annotated subject and object as in the benchmark datasets [180] and also often contains noisy text. However, parse structures can still be useful for capturing longer-range dependencies than sequential models (for instance, long clauses or complex scoping shown in these tweets). For instance, in Figure 7.1, the sequential distance between attack *mentally ill* and target *Transgenders* is five tokens, but while the distance in the parse tree is only two. Thus, we propose an adaption of the [180] model that learns a unified representation of text by encoding the whole dependency parse of the sentence. We later build a classifier based on this representation for identifying offensive language.

According to social influence theory [182], users get influenced by their friends' behavior leading to user homophily [183] (similar behavior) among connected users. Observable signals from social media can be taken as indicators of shared community membership. [93] showed that most of the abusive behavior in their version of Waseem and Hovy [78]'s dataset comes from a small set of densely connected users. They showed that incorporating user features along with linguistic features, can improve the hate classification task. We propose

an extended version of our model that uses a user social graph in addition to the parser graph to learn extended embeddings.

In this work, we propose a classifier based on dependency graphical convolutional networks (DepGCN) to detect offensive language online. We tested our method on the benchmark Twitter hate speech datasets. Our model outperformed the current state-of-the-art [78, 79] for offensive language detection and even strong baselines like fine-tuned BERT [1]. We further propose a UserGCN model to incorporate the effect of social influence on user's abusive behavior. The UserGCN model learns a user's abusive behavior through a class prior and propagates this behavior through their social network. After merging the DepGCN with the UserGCN model, we achieve a new state-of-the-art on these benchmark datasets.

7.2 PROPOSED METHOD

In this section, we first describe how we represent text as a graph using the dependency parse tree. We then convolve over this dependency graph using a graph convolutional framework (DepGCN) to compute a text embedding used for offensive language detection task. Further, we describe our novel GCN based architecture that exploits the user social graph to augment the text embeddings.

7.2.1 Graph representation of Text

We use the dependency parse tree to induce a graph on a sentence. Specifically, a graph $G =< V, E >$ is represented as a collection of vertices V and as a set of edges E between these vertices. Thus, to compute the graphical representation of the sentence, we treat each word as a vertex, with syntactic dependencies between words corresponding to an edge. Now, for this graph G, $\mathbf{A}$ represents the Adjacency matrix where $A_{ij} = 1$ if there is a dependency relation between word i and j and 0 otherwise. We also connect each word to itself such that $A_{i,i} = 1$; $\forall i \in V$. Although syntactic dependencies are directed, we treat these dependency edges as undirected, resulting in a symmetric matrix[2].

Graph Convolution Networks (GCN) are recently proposed to compute vertex embeddings in a graph by convolving over each vertex's local neighborhood [181].

The convolution operation for vertex i in layer k in GCN is defined as follows,

[2]Similar to [180], we observed a performance dip when using each edge direction as a separate graph.

$$h_i^{k+1} = \sigma\left(\sum_{j=1}^{|V|} \tilde{A}_{ij} W^k h_j^k + b^k\right) \tag{7.1}$$

$$= \sigma\left(\frac{\sum_{j \in \mathcal{N}(i) \cup i} W^k h_j^k}{d_i} + b^k\right) \tag{7.2}$$

where $\tilde{\mathbf{A}} = D^{-1/2}\mathbf{A}D^{1/2}$ is the normalized Adjacency matrix with D being the degree matrix. h_i^{k+1} represents the vertex embeddings at layer $k+1$, with h_i^0 being initialized with the vertex features. In our case, we use pretrained word embeddings as the initial features. W^k, b^k are learnable weight and bias parameters of layer k and σ represents the ReLU function. $\mathcal{N}(i)$ represents the vertex i's neighborhood while $d_i = \sum D_i$ represents the vertex degree.

Now, assume that $W^k = \mathbf{I}$ with $b^k = 0$ and $\sigma(.)$ as an identity function. The updated vertex embedding at layer $k+1$ will be,

$$h_i^{k+1} = \frac{\sum_{j \in \mathcal{N}(i) \cup i} h_j^k}{d_i}$$

It is thus easy to verify that the convolution operation updates the vertex embeddings at layer $k+1$ to be the average embeddings of the vertex's neighborhood and the vertex itself from the previous layer, k. In our dependency graph, applying graph convolution operation will augment each word's embedding with its syntactic neighbors. Thus, convolution helps to contextualize the word embeddings, where the word's syntactic relationships define the context. Notice that it is different from the sequential models (like LSTM or CNN), where the adjacent words in the sentence define the context.

Consider the sample tweet in Figure 7.1, first, *ill* will be augmented with its surrounding adverbs *mentally* and *just* (eq. (7.3)). In turn, these updated embeddings of *ill* will be propagated when computing embeddings of the noun *people* in addition to the subject *Transgenders*.

$$h_{ill} = f_{gcn}(h_{mentally}, h_{just}, h_{ill}) \tag{7.3}$$

$$h_{people} = f_{gcn}(h_{ill}, h_{transgenders}, h_{people}) \tag{7.4}$$

However, in sequential models with a fixed window, the attack *ill* will be too far from the subject *Transgenders*.

Further, by stacking such k convolution layers, we can propagate the vertex embeddings to its k-hop neighborhood [181]. For our experiments, we did not see any further improvements after two layers. This could be because, as we are dealing with a short text, the resulting parse tree is shallow.

7.2.2 Sentence representation

In the previous section, we computed contextualized word embeddings using syntactic relationships. However, we still need to aggregate these node embeddings to compute a graph-level embedding (sentence in our case). In particular, we perform masked pooling over the learned word embeddings from the last layer (K) to compute a sentence embedding. We only pooled over non-terminal words or intermediary nodes in the dependency parse tree (i.e. $|A_i| > 2$). We ignored the leaf words or words linked to only one other word as their word embeddings are relatively unchanged (because of less number of neighbors) after the convolution as compared to other intermediary nodes with more neighbors. Thus, when we perform pooling over all the words, leaf words will skew the final result even though they are not always important. We tried different variants of pooling (average and min), but max-pooling performed the best (eq. (7.5)) for our case.

$$h_G = \max_{i \in V'} (h_i^K) \; s.t. |A_i| = 2 \tag{7.5}$$

Further, these sentence embeddings are fed through fully connected layers followed by a sigmoid (σ) to compute final class score (eq. (7.6)) for the sentence.

$$c_G = \sigma(f_{MLP}(h_G)); c_G \in \mathbf{R}^C \tag{7.6}$$

Here, C represents the total number of classes.

7.2.3 Embedding variants

As we deal with noisy text, there can be ill-formed words and grammatically incorrect sentences that can lead to incorrect parse trees. Thus, to overcome these potential errors, we feed the initial word embeddings (h_i^0) to a BiLSTM module. The BiLSTM module helps to aid in word disambiguation by encoding adjacent words in the sentence.

The final architecture of our dependency graph convolution network (DepGCN) is depicted in Figure 7.2.

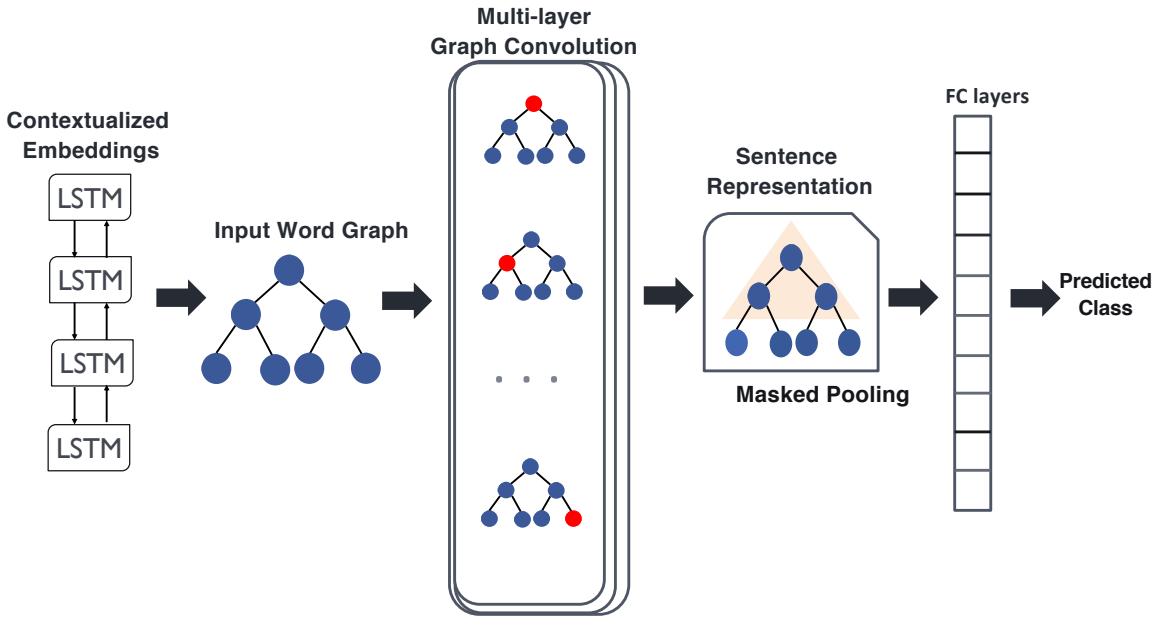


Figure 7.2: Overview of our proposed model

7.2.4 User Features

Mishra et al. [93] observed that only a small set of users were responsible for most of the offensive tweets in the Waseem and Hovy [78] dataset. In a similar analysis on tweets posted in response to President Obama's re-election, Zook [184] found that most of the racist tweets came from only a group of states. Thus, it shows that determining the author of the tweet will be beneficial for automated offensive language detection.

Further, prior studies have shown that user behavior is influenced by their friends in online social networks leading to similar behavior(user homophily) of connected users [183, 182]. To empirically verify this in Waseem and Hovy [78] dataset, we computed the average cosine similarity between the class distribution of a user's tweets and their friends' tweets. We observed a high similarity value of *0.80*, indicating similar tweeting behavior between connected users. In other words, this high value shows that users connected to offensive users tend to follow suit. Thus, modeling the influence of user's social connections can potentially improve our understanding of user behavior. We thus extend our proposed model to augment the sentence embeddings obtained from DepGCN with the social embeddings capturing user and her friends' tweeting behavior.

To this end, we use the follower followee relationship of each user in our dataset from Twitter[3]. Similar to the dependency graph, we create a social graph and represent each user as a vertex, and the edges represent the follower relationship. For our experiments,

[3]Thanks to authors of [93] for providing the user social relationship data.

we treat the follower-followee relationship as equivalent. We believe that it is a reasonable assumption, as in general, follower-followee relationships are often reciprocal, and our dataset does not contain any celebrities (skewed ratio of followers vs. followee).

To capture the user's tweeting behavior, we first compute user embeddings (e_u) as the average of the sentence embeddings obtained from DepGCN, for all the tweets authored by the user (eq. (7.7)).

$$e_u = \frac{\sum_T h_T}{|\mathcal{A}(u)|}; \forall T \in \mathcal{A}(u) \tag{7.7}$$

$\mathcal{A}(u)$ represents all the tweets authored by the user u.

To capture the effect of homophily, we perform graph convolution operation on the user's social graph with these user embeddings being used as initial vertex features (h_u^0).

In the first layer of the graph convolution, we project these user embeddings to a C dimensional vector to learn the user's prior distribution per class. It is not correct to classify each user or all her tweets to be abusive or not. Therefore, we learn a class probability for each user, which indicates her likelihood of posting an offensive tweet.

In the subsequent layers, we propagate these class priors through the user's social network.

$$h_u^k = \sigma\left(\frac{\sum_{v \in \mathcal{N}(u) \cup u} W^k h_v^k}{d_u} + b^k\right) \tag{7.8}$$

where $\mathcal{N}(u)$ denotes the friends of user u.

$$h_S = \sigma(h_u^K) \tag{7.9}$$

$$c_F = f_{MLP}([h_G, h_S]) \tag{7.10}$$

where $h_S \in R^C$ are social embeddings for user who tweeted the tweet. Finally, we concatenate the text and social embeddings for each tweet and compute the final class probability. The final architecture of UserGCN with our DepGCN is depicted in Figure 7.3.

7.3 EXPERIMENTS

In this section, we first describe our experimental setup, followed by the performance results. We then present a detailed error analysis of our dependency-based model vs. a widely-used sequential model for sentence classification. Finally, we present results for an

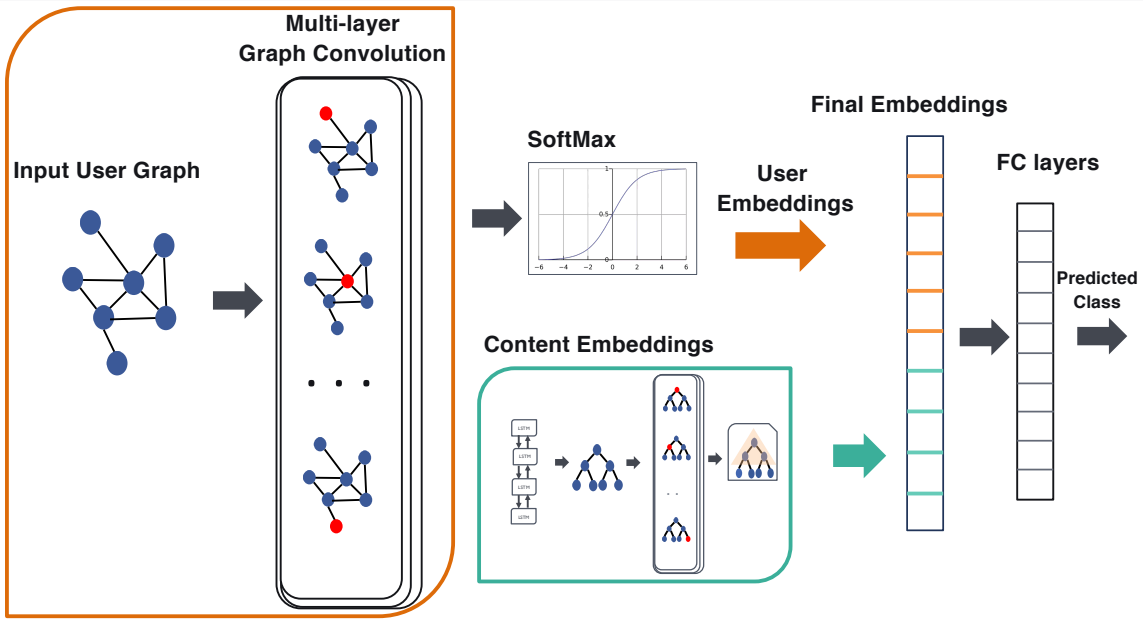


Figure 7.3: Overview of our proposed model with user social graph

approach that augments the textual embeddings with the user's social information.

7.3.1 Experimental Setup

We first describe our datasets, followed by comparative baselines.

Datasets: Wiegand et al. [185] emphasizes the difficulty of selecting representative datasets for studying abusive language. At the heart of this difficulty is the relative rarity of hate speech in the large-scale user-generated text. It is not unusual for $> 99\%$ of text to be benign.

To make experiments manageable Waseem and Hovy [78] bootstrap data collection with queries that are indicators of possible hate speech. As a result of this bootstrapping, the data collected is *not* representative of the underlying text distribution. This *data bias* applies to both the benign and the offensive categories.

Davidson et al. [79] provides an alternative dataset, but this, too, is affected by pre-filtering, with the result that the dataset also fails to represent the underlying text distribution. Thus, caution is advised when interpreting the results of studies on these datasets. Wiegand et al. [185] suggests several mitigations for the deficiencies of these datasets, as well as cross-classification methods that can sometimes diagnose over-optimism about classification results.

For comparability, we experiment with both the datasets from Davidson et al. [79] and

Waseem and Hovy [78]. We do not claim that our results will necessarily transfer to more naturalistic settings. Table 7.1 lists the per class distribution in the collected dataset. Note that as previously noted, Davidson et al. [79] dataset is highly skewed, with the majority of tweets being offensive. We thus also create a custom dataset (Davidson ext.) to mimic the real-world settings, by adding benign tweets from Waseem and Hovy [78] to Davidson et al. [79]'s benign tweets.

Dataset	**Categories**		
Davidson et al. [79]	Hate	Offensive	Benign
	1,430	19,190	4,163
Waseem and Hovy [78]	Racism	Sexism	Benign
	1,939	3,148	11,115
Davidson extended	Hate	Offensive	Benign
	1,430	19,190	15,278

Table 7.1: Dataset Statistics.

Baselines: We compare against a variety of state-of-the-art approaches proposed for computing sentence embeddings. We use these embeddings to classify the tweets into offensive or not.

- **N-grams** Current state-of-the-art approach for hate speech classification [78] extracts N-grams of the tweets and feed them into logistic regression along with Twitter-specific features per tweet.

- **BERT** BERT, Devlin et al. [1] provides a variety of text embeddings, which have achieved new state-of-the-art results on multiple natural language tasks such as question answering and language inference. For our experiments, we initially fine-tune the $BERT_{base}$ model on our training dataset. We then extract the embeddings of each tweet from the trained model and feed them to a feedforward network to compute the final class score.

- **BiLSTM** is a sequential model to compute sentence embeddings that is useful for many downstream classification tasks [186]. We use the output of the final hidden layer in the BiLSTM as the sentence embeddings. We follow BERT in feeding the sentence embeddings to a feed-forward network to compute the final class-wise score.

Implementation Details: We initialize h_i^0 of each word with its Glove embeddings [82] combined with its POS tag, NER tag, and dependency relation given by the Stanford NLP API. We use the Stanford parser[4] for extracting the dependency parse relationship between

[4] `https://nlp.stanford.edu/software/lex-parser.shtml`

the words in a tweet. We perform stratified sampling on the dataset to create an 80-10-10 split between training, development and test sets. The development set is used for hyperparameter tuning while the results are reported on the test set. We report the class-wise F1 score for each dataset. We implement our model and the baselines in PyTorch and run the experiments on an Nvidia Tesla V100 GPU. We use a two-layer GCN for both DepGCN and UserGCN. We use a *weighted* cross-entropy loss to counter the effect of class imbalance. We do for all the baselines and our proposed approach.

7.3.2 Performance Analysis:

Table 7.2 reports class-wise F1 score with weighted F1 for the Davidson et al. [79] extended dataset. As expected, the bag-of-words based N-gram approach is not competitive with the best approaches. This is reasonable, since that approach does not take any advantage of semantic similarities between different words. More surprisingly, the state-of-the-art BERT model, even after fine-tuning, still performs slightly worse than our DepGCN model.

Approach	Hate	Offensive	Benign	Overall
N-grams	0.35	0.88	0.88	0.85
BERT	0.45	0.94	0.96	0.91
BiLSTM	0.31	0.93	0.94	0.90
DepGCN	0.47	0.94	0.96	0.92
BiLSTM + DepGCN	**0.49**	**0.95**	**0.97**	**0.93**

Table 7.2: Class-wise F1 score with the overall weighted F1 score for different approaches on the Davidson et al. [79] extended dataset.

The sequential model, i.e., BiLSTM, also performs worse than our dependency-based model. As argued before, sequential models often struggle to capture long term dependencies between words while DepGCN alleviates this issue by encoding syntactic dependencies. Further, if we use BiLSTM to contextualize the embeddings before feeding it to our DepGCN model, the results are slightly improved. Note that even a slight improvement in the hate class is significant as the dataset contains limited training examples for this class (Table 7.1) as compared to the other classes.

We obtain a similar trend in the results when evaluating performance on the original Davidson et al. [79] dataset, as shown in Table 7.3. BERT becomes more competitive on the original dataset. The Benign class of the original dataset has systematically lower figures than the corresponding class in the extended dataset, presumably because the extended data set has a better representation of the space of possible benign examples. The Hate class is

Approach	Hate	Offensive	Benign	Overall
N-grams	0.46	0.94	0.84	0.89
BERT	0.42	**0.95**	**0.88**	**0.91**
BiLSTM	0.52	0.94	0.86	0.90
DepGCN	0.50	0.94	0.86	0.90
BiLSTM + DepGCN	**0.53**	0.94	0.87	**0.91**

Table 7.3: Class-wise F1 score with the overall weighted F1 score for different approaches on the Davidson et al. [79] dataset.

slightly easier to detect in the original dataset, even though it contains the same examples as the corresponding class in the extended dataset, presumably because the classifiers expend more of their modeling capacity on the benign set. The same pattern is present to a lesser degree for the Offensive class. BERT becomes more competitive with BiLSTM on the original dataset. BiLSTM retains a substantial ($0.53 > 0.48$) advantage over BERT on the Hate class, and is close on Offensive and Benign.

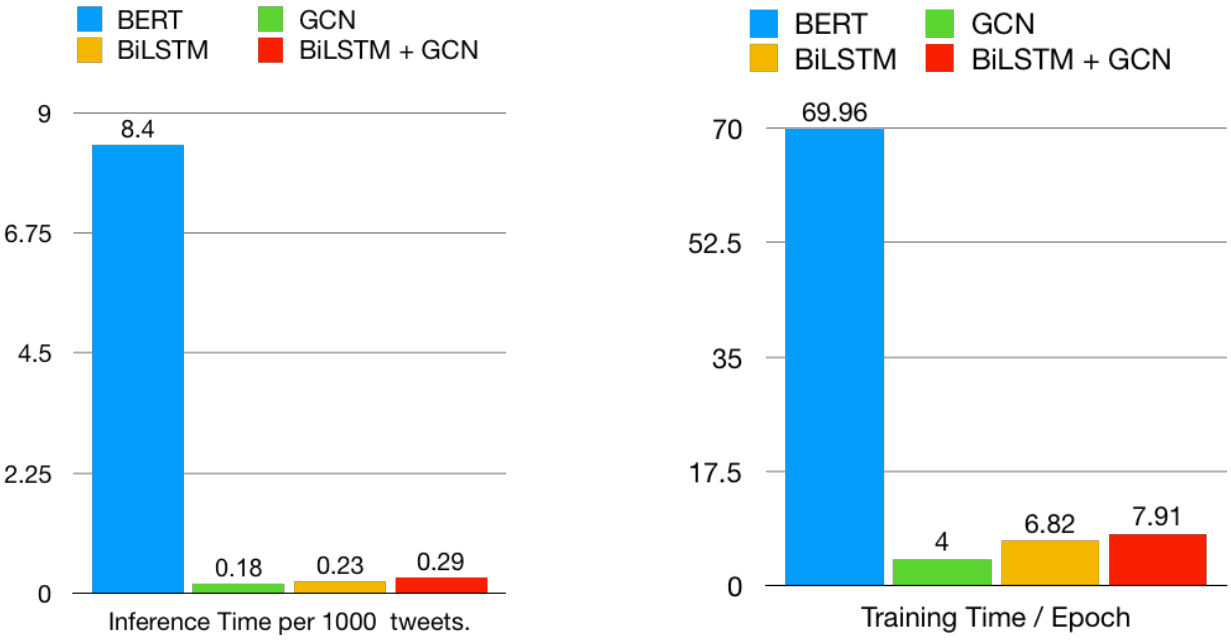
The sequential model, BiLSTM, performs slightly better than our DepGCN model. One possible explanation can be that the Davidson et al. [79] dataset is full of *slurs* and *direct* hate attacks on entities. These *direct* attacks do not exhibit long-range dependencies and thus, are well captured by the sequential models. Also, due to the heavy usage of *slurs*, BERT performs worse as there are many OOV tokens in the dataset.

Approach	Racist	Sexist	Benign	Overall
N-grams	0.75	0.71	0.88	0.83
BERT	0.78	**0.81**	**0.91**	**0.88**
BiLSTM	0.72	0.71	0.89	0.84
DepGCN	0.76	0.72	0.88	0.83
BiLSTM + DepGCN	**0.78**	0.74	0.90	0.85

Table 7.4: Class-wise F1 score with the overall weighted F1 score for different approaches on the Waseem and Hovy [78] dataset.

On a more nuanced dataset collected by Waseem and Hovy [78], BERT performs the best out of the competing methods, as shown in Table 7.4. Our model performs competitively for racist and benign tweets while it performs worse for sexist tweets. This dataset is more nuanced as it contains more indirect or implied hate attacks (discussed in section 7.3.3) with the usage of fewer slurs. Thus, the powerful language model BERT can better capture the meanings of these tweets.

Time Analysis : We further compare the running time analysis of all the baseline ap-



(a) Inference time

(b) Training time

Figure 7.4: Time analysis of variants of our model with respect to the popular BERT [1] language model.

proaches. Figure 7.4 shows the comparison for both training time and inference time. First, in Figure 7.4a, we plot the inference time (in secs) required by each approach per 1000 tweets. Our proposed DepGCN is the most efficient approach at inference time closely followed by BiLSTM. Adding the BiLSTM module before the DepGCN only increases the inference time slightly. However, BERT, on the other hand, takes an order of magnitude longer than any of these approaches. Note that the inference time does not take into account the time taken to extract the parse tree for the tweets. However, as we are looking at a short text, this time is negligible. The same trend can be observed for training time too in Figure 7.4b. However, the jump from DepGCN to BiLSTM training time is a little higher than during inference.

Thus, our parser-based DepGCN approach is much more efficient than the BERT model. Also, including BiLSTM module to the DepGCN model leads to only a slight drop in efficiency.

7.3.3 Error Analysis of Sequential vs. Dependency model

In this section, we present a detailed analysis of the errors of the sequential (BiLSTM) vs. Dependency (DepGCN) model. Table 7.5 shows the confusion matrix of BiLSTM vs DepGCN model on the Waseem dataset. The parser-based approach is more conservative in

labeling tweets as benign than the sequential approach. Specifically, sexist tweets are more probable to be misclassified as racist and vice versa. Alternatively, DepGCN tags much more benign tweets as offensive (Sexist/Racist), thus creating more false positives. However, as there is a higher cost involved in missing an offensive tweet, DepGCNs will be more effective in real-world scenarios.

	Racism	Sexism	Benign
Racism	7	11	7
Sexism	7	11	8
Benign	35	70	33

(a) Dependency Parser model

	Racism	Sexism	Benign
Racism	4	3	18
Sexism	1	7	18
Benign	9	25	104

(b) Sequential Model

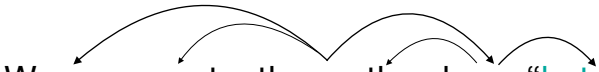
Table 7.5: Confusion matrix for Sequential (BiLSTM only) vs Dependency Parser (GCN only) approach for Waseem and Hovy [78] dataset.

We also examined some sample tweets from the Waseem and Hovy [78] dataset, which were erroneously classified as benign by BiLSTM but not by DepGCN and vice versa to understand the difference between these two approaches in depth.

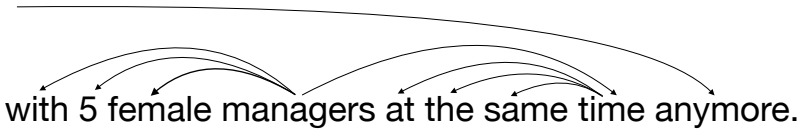
Sexist tweet missed by LSTM: Following is a sample sexist tweet that is correctly classified by the DepGCN approach but missed by the BiLSTM. *"I'm not sexist but women get upset with other women for stupid reasons. Women constantly say they have "haters"."* Figure 7.5a shows the parse tree of the tweet by the Stanford parser. It is a difficult sample to classify as the author of the tweet says that he is *not* sexist but is writing offensive remarks against women. The dependency tree can capture this long-range dependency and establish negative relation of "upset," "stupid," and "haters" with the "women" subject.

Sexist tweet missed by DepGCN: However, DepGCN fails to capture similar nuanced sexism in another sample tweet, *"And when they're all PMSing at the same time LOL I'm not sexist, but I can't work with 5 female managers at the same time anymore.""*. Note that the sentence contains punctuation error as it is missing punctuation between the two sentences in the tweet (after *time* and before *I'm not*). This error leads to a wrong parse tree, as shown in Figure 7.5b. Thus, our parser-based model is sensitive to these parsing errors.

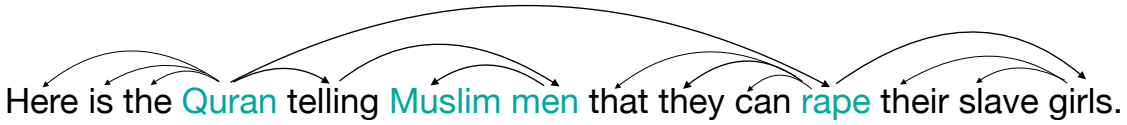
Racist tweet missed by DepGCN: However, even if the parse tree is correct, just establishing dependency relationships may not be sufficient to capture nuanced relationships in the text. For instance, the parse tree of the racist tweet, *"Here is the Quran telling Muslim men that they can rape their slave girls."* shown in Section 7.3.3 is correct. However, the parse tree misses the coreference of pronouns *they* and *their* to belong to *Muslim men*. In these



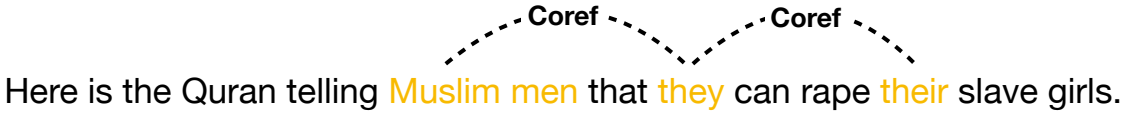
(a) Parse tree of the sexist tweet missed by LSTM.



(b) Parse tree of the sexist tweet missed by DepGCN



(c) Parse tree of the racist tweet missed by DepGCN.



(d) Coreference resolution of the tweet.

Figure 7.5: Parse Tree of the two sample tweets from Waseem dataset.

cases, powerful language models like BERT will be able to extract these relationships.

7.3.4 Effect of User Social Graph

Table 7.6 shows the class-wise F1 scores after adding the user social graph with the DepGCN model on the Waseem and Hovy [78] dataset. We do not report results on the other datasets as we have do not have any information about their users. UserGCN used in isolation performs worse than the other approaches. This poor performance is expected as it ignores the textual information entirely. Note that the UserGCN uses the user's class prior only for prediction for all of the user's tweets. However, even with that naive methodology, this model achieves similar performance to the previous state-of-the-art linguistic approach, N-gram (Table 7.4), and our DepGCN model for the sexist class.

Approach	Racist	Sexist	Clean	Overall
BERT	0.78	0.81	0.91	0.88
BiLSTM + DepGCN	**0.78**	0.74	0.90	0.85
UserGCN	0.61	0.72	0.79	0.74
BiLSTM+DepGCN +UserGCN	**0.79**	**0.82**	**0.92**	**0.88**

Table 7.6: Comparison of class-wise F1 score with weighted F1 score for different approaches with our proposed approach after adding the user social features on Waseem and Hovy [78] dataset.

After adding the user features, our model improves by 0.03 F1 points and slightly outperforms the previously best performing BERT model. The most significant improvement comes from the tweets annotated as sexist (similar trend seen in UserGCN). This improvement indicates that there is a strong homophily effect in users authoring sexist tweets in the dataset. As noted by Mishra et al. [93], racist tweets in this dataset are contributed by only five users who also tweet other benign and sexist tweets. Thus, incorporating additional user information does not give us higher gains for the racist class.

7.4 CONCLUSION

In this work, we propose a sentence encoder that extends the graph convolutional network (GCN) to an induced graph built from syntactic dependencies in the text for offensive language detection. Our model achieved state-of-the-art performance on public hate speech twitter datasets without the use of user features. The overall performance matches or exceeds that of strong baselines such as fine-tuned BERT. Our framework is interpretable and allows efficient parallel batch processing. Finally, we show that addition of user features is able to gain benefit from strong user homophily in the abusive user accounts that we worked with. This is achieved using a network architecture that allows behavioral priors to propagate from user to user.

CHAPTER 8: CONCLUSION

In this chapter, we will first summarize our contributions and then discuss avenues of future work.

8.1 SUMMARY

User behavior modeling lies at the heart of the current intelligent systems to effectively cater to user's preferences and needs. In this dissertation, we tackled two important challenges related to user behavior modeling. First is to develop deep understanding of user-generated content to estimate user's latent behavioral characteristics. The second is to model different types of user-to-user influence; either through established relationships or implicit social influences; different relationship semantics either similarity or contrast between the behavior. In particular, we developed approaches that modeled latent user reliability (chapter 4), incorporated social influence on user preferences (chapter 5), modeled the effect of semantically diverse social influences measured through similarity and contrast in user behavior (chapter 6), jointly modeled social influence with user's latent behavior estimation (chapter 7) and finally, modeled implicit influence based on similarity in behavior evolution (chapter 3). We applied our approaches to a multitude of tasks, predicting future purchases in recommender systems and research interests in scholarly data, predicting accepted answers in StackExchange communities, predicting offensive language on Twitter and inferring trustworthiness of comments and user reliability in Reddit.

Firstly, in Chapter 4, we modeled latent aspect-level user reliability in Reddit to estimate the most trustworthy comment to a post. In particular, our model estimated user-aspect reliability and semantic representation of each comment simultaneously in a unified optimization framework. We learned a trustworthy comment embedding for each post, such that it is semantically similar to comments of reliable users on the post and also similar to the post's context. The learned embeddings are further used to rank the comments for that post. We showed through experiments that modeling user-aspect reliability improves the prediction performance compared to the non-aspect version of our model. We also showed that the estimated user-post reliability can further be used to identify trustworthy users for particular post categories.

Secondly, we modeled the influence of user's friends on her reviewing preferences in the recommender systems in Chapter 5. Recommender Systems have also previously exploited the user homophily (similar behavior) between connected users to provide improved recom-

mendations to their users [110, 37, 32]. In this work, we exploited the effect of homophily in the user and item space *both* on user's evolving preferences. In the user space, we experimented with both established social connections, and induced connections based on similar purchasing history. Similarly, in the item space, we constructed item similarity graphs based on frequent co-occurence and feature similarity. To capture the effect of a user's friends' recent history, we developed a novel graph attention network based social aggregation model. Different from existing works, it aggregates a user's friends' recent item history in a weighted manner. We learn attention weights separately for each pair of a user and her friend. We developed a novel graph attention network based aggregation model for the item similarity graphs too. In contrast to existing work, we learn an attention weight for each similar item and later aggregate information of neighboring items in a weighted manner. Modeling homophily in both user and item space outperforms other approaches which exploit homophily of the either graphs.

Next in Chapter 6, we relaxed the assumption of similarity in the connected nodes used in previous chapter. This is especially useful in platforms with no or limited social structure. We induced semantically diverse graphs based on contrast and similarity in user's behavior. The contrastive relation is especially useful for ranking scenarios. Thus, we evaluated our model on answer selection task in CQA platforms. We also introduced an extension to the original GCN architecture to estimate contrast instead of similarity between connected nodes. Finally, we proposed a boosted architecture to merge diverse and potentially noisy graphs together. Our architecture beats other GCN based baselines proposed for multi-relational graphs.

Next, we worked on joint modeling of the latent user behavior with social influence in Chapter 7. We evaluated our approach on offensive language prediction task on Twitter. We first induced a graph on the words in a tweet using syntactic dependencies between them. These syntactic dependencies are better suited to capture long range dependencies present in offensive language than sequential models used currently. We proposed a graph convolution (GCN) based classifier that learns effective text representations using the induced content graph. We further estimate latent user behavior–abusive behavior i.e. their likelihood of posting offensive text online, from the improved text representations. Further, to capture user homophily in abusive user accounts, we proposed another GCN-based model that propagates these behaviors in their social circle on Twitter. We showed that leveraging both text and user representations is a more effective approach for detecting offensive language online rather than the current approaches that only use text.

Finally in Chapter 3, we proposed a model to implicitly identify similar users based on their latent behavior evolution. This work extended the work described in previous chapters

which induce connections based on aggregated user behavior. We specifically proposed a G-HMM architecture that clusters users with similar evolutionary archetypes. Our model identified four different type of archetypes of change in research interests for computer science researchers. These archetypes tend to differ in the their gender distribution and amount of grants awarded. We also observed different evolutionary archetypes among StackExchange users.

The main contributions of the dissertation are:

- Learned latent user behavior characterisitcs such as aspect-level user reliability and abusive language usage by leveraging user-generated content in the platform.
- Models implicit user influence (similar or contrastive) by inducing edges between the users. None of the previous works estimate influence through edges. Edges are constructed based on behavioral similarity with the other users, contrast in the behavior and for contrasting similarly from other users.
- Proposed modification to GCN architecture to model contrast relationship in the neighborhood. Also, proposed multiple ways to counter noise in induced edges; such as attention coupled with edge sampling; and boosting architecture where each induced graph acts as a weak learner.
- Proposed joint modeling of social influence with latent user behavioral characteristics.
- Proposed an interpretable model to identify similar users based on similar patterns of change in behavior.

8.2 FUTURE WORK

There are multiple avenues of future work which can improve upon the approaches discussed in this dissertation for user behavioral modeling.

Induced graph with user's latent behavior estimation: In chapter 6, we used multiple induced graphs based on contrast and similarity in user behavior to rank the answers in CQA forums. While, in chapter 4, we estimated topic-based reliability of the user, and in turn, used that to estimate the best answer in the forum. However, we did not exploit any explicit or implicit relationships between the users in the platform. One natural extension is then to incorporate modeling of user's topic-based reliability in the induced graphs (IR-GCN) approach to improve upon the estimation of best answer. For instance, in the contrastive graph, the current model establishes contrast between different answers based on the static

aggregated user or answer features. However, the updated model will also take into account the reliability of the answering user on the question's topic to establish the contrast between different answers. This context-based contrast can improve the answer selection task greatly.

Context-specific edge semantics: In Chapter 6, we treated different semantic edges (contrastive, similar contrast, reflexive) independent of each other. Specifically, more often than not, there will be only one kind of edges between two nodes. However, semantics of the connections between two nodes can depend on the context. For instance, for a recommender system, the key goal could be to show diverse range of potential items to a user. Thus, we first need to establish contrast between the items to create a ranked list of potential items the user may be interested in based on her past purchases. However, we may further need to establish similarity between these items based on some context such as price range, quality, aesthetics etc. to show diverse range of products to the user instead of showing near similar items. Similarly in Reddit, when choosing multiple correct comments for a post, we need to contrast between the different comments to rank them. Later, in the ranked comments itself, we may need to cluster these responses based on some attribute similarity like opinions, author demographics etc.

Evolving relationships: In all the works discussed in the dissertation, we assume static nature of the connections for both established and induced connections. However, this may not be true everywhere. Users' behavior change over time, such as their preferences towards a product in an e-commerce platform or their expertise in the CQA platform. Hence, inducing these behavioral connections based on aggregated history is not the optimal solution. These connections should evolve over time to give an accurate description of the social influence at a given time. Similarly, when estimating user's latent behavior characteristics such as reliability (chapter 4) or abusive behavior (chapter 7), more weightage should be given to their recent activity in the platform rather than the distant past.

Automated edge detection: The induced connections used in this dissertation are predefined, either based on prior knowledge or domain expertise. However, this could be a limiting factor when working on a novel domain with limited prior knowledge or for problems where establishing relationship (similarity or contrastive) is time-consuming or computationally expensive. Thus, another interesting future work is to allow the model to automatically detect these connections itself. However, there are chances of increased noise in the these automated edges. Thus, the model needs to be updated to counter the increased noise in the graph. In this dissertation, we handled noisy induced connections by treating them as weak learners. Yet another way to model these noisy edges can be as soft edges, i.e. each edge has a certain probability of being a valid edge. These probability values should be embedded in the model that will help to counter the strong signal created through these

induced connections currently.

Higher-order relationships Even though, we considered relationships with different semantics (contrastive, similarity by contrast etc.), they are still limited to pairwise relationships. However, it could be helpful to exploit higher-order relationships like motifs ([187, 188]) or transitive relationships (such as friend of a friend is a friend or friend of an enemy is a enemy) ([103]).

REFERENCES

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[2] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[3] Ronald R Yager. Targeted e-commerce marketing using fuzzy intelligent agents. *IEEE Intelligent Systems and their Applications*, 15(6):42–45, 2000.

[4] Nicholas B La Thangue and David J Kerr. Predictive biomarkers: a paradigm shift towards personalized cancer medicine. *Nature reviews Clinical oncology*, 8(10):587–596, 2011.

[5] Jon Kleinberg, Himabindu Lakkaraju, Jure Leskovec, Jens Ludwig, and Sendhil Mullainathan. Human decisions and machine predictions. *The quarterly journal of economics*, 133(1):237–293, 2018.

[6] Felix Wong, Chee Tan, Soumya Sen, and Mung Chiang. Quantifying political leaning from tweets, retweets, and retweeters. *IEEE Transactions on Knowledge and Data Engineering*, 28:1–1, 08 2016. doi: 10.1109/TKDE.2016.2553667.

[7] Munmun De Choudhury, Emre Kiciman, Mark Dredze, Glen Coppersmith, and Mrinal Kumar. Discovering shifts to suicidal ideation from mental health content in social media. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, page 2098–2110, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450333627. doi: 10.1145/2858036.2858207. URL `https://doi.org/10.1145/2858036.2858207`.

[8] Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. Social influence analysis in large-scale networks. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 807–816, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-495-9. doi: 10.1145/1557019.1557108.

[9] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016.

[10] Marcelo Maia, Jussara Almeida, and Virgilio Almeida. Identifying user behavior in online social networks. SocialNets'08, 2008. ISBN 978-1-60558-124-8. doi: 10.1145/1435497.1435498. URL `http://doi.acm.org/10.1145/1435497.1435498`.

[11] Lena Mamykina, Bella Manoim, Manas Mittal, George Hripcsak, and Björn Hartmann. Design lessons from the fastest q&a site in the west. CHI'11, 2011. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979366. URL http://doi.acm.org/10.1145/1978942.1979366.

[12] Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. Knowledge sharing and yahoo answers: Everyone knows something. WWW'08, 2008. ISBN 978-1-60558-085-2. doi: 10.1145/1367497.1367587. URL http://doi.acm.org/10.1145/1367497.1367587.

[13] Adabriand Furtado, Nazareno Andrade, Nigini Oliveira, and Francisco Brasileiro. Contributor profiles, their dynamics, and their importance in five q&a sites. CSCW'13, 2013. ISBN 978-1-4503-1331-5. doi: 10.1145/2441776.2441916. URL http://doi.acm.org/10.1145/2441776.2441916.

[14] Fabricio Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgilio Almeida. Characterizing user behavior in online social networks. IMC'09, 2009. ISBN 978-1-60558-771-4. doi: 10.1145/1644893.1644900. URL http://doi.acm.org/10.1145/1644893.1644900.

[15] Jaewon Yang, Julian McAuley, Jure Leskovec, Paea LePendu, and Nigam Shah. Finding progression stages in time-evolving event sequences. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW'14, pages 783–794. ACM, 2014. ISBN 978-1-4503-2744-2. doi: 10.1145/2566486.2568044. URL http://doi.acm.org/10.1145/2566486.2568044.

[16] Bernhard Knab, Alexander Schliep, Barthel Steckemetz, and Bernd Wichern. *Model-Based Clustering With Hidden Markov Models and its Application to Financial Time-Series Data*. Springer Berlin Heidelberg, 2003. ISBN 978-3-642-18991-3. doi: 10.1007/978-3-642-18991-3_64. URL https://doi.org/10.1007/978-3-642-18991-3_64.

[17] Sofia Angeletou, Matthew Rowe, and Harith Alani. Modelling and analysis of user behaviour in online communities. In *The Semantic Web-ISWC'11*, 2011. ISBN 978-3-642-25073-6.

[18] Tiago Santos, Simon Walk, Roman Kern, Markus Strohmaier, and Denis Helic. Activity archetypes in question-and-answer (q8a) websites—a study of 50 stack exchange instances. *Trans. Soc. Comput.*, 2(1):4:1–4:23, February 2019. ISSN 2469-7818.

[19] Padhraic Smyth. Clustering sequences with hidden markov models. In *Advances in Neural Information Processing Systems*, pages 648–654. MIT Press, 1997.

[20] Manuele Bicego, Vittorio Murino, and Mário A. T. Figueiredo. Similarity-based clustering of sequences using hidden markov models. In *Proceedings of the 3rd International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 86–95. Springer-Verlag, 2003. ISBN 3-540-40504-6. URL http://dl.acm.org/citation.cfm?id=1759548.1759559.

[21] Emanuele Coviello, Antoni B. Chan, and Gert R. G. Lanckriet. Clustering hidden markov models with variational hem. *J. Mach. Learn. Res.*, 15:697–747, 2014. ISSN 1532-4435. URL http://dl.acm.org/citation.cfm?id=2627435.2627457.

[22] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, pages 452–461, Arlington, Virginia, United States, 2009. AUAI Press. ISBN 978-0-9749039-5-8. URL http://dl.acm.org/citation.cfm?id=1795114.1795167.

[23] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 285–295, New York, NY, USA, 2001. ACM. ISBN 1-58113-348-0. doi: 10.1145/371920.372071. URL http://doi.acm.org/10.1145/371920.372071.

[24] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 173–182, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee. ISBN 978-1-4503-4913-0. doi: 10.1145/3038912.3052569. URL https://doi.org/10.1145/3038912.3052569.

[25] Yao Wu, Christopher DuBois, Alice X. Zheng, and Martin Ester. Collaborative denoising auto-encoders for top-n recommender systems. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, pages 153–162, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3716-8. doi: 10.1145/2835776.2835837. URL http://doi.acm.org/10.1145/2835776.2835837.

[26] Chen Cheng, Haiqin Yang, Michael R. Lyu, and Irwin King. Where you like to go next: Successive point-of-interest recommendation. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 2605–2611. AAAI Press, 2013. ISBN 978-1-57735-633-2.

[27] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 811–820, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-799-8. doi: 10.1145/1772690.1772773. URL http://doi.acm.org/10.1145/1772690.1772773.

[28] Peijie Sun, Le Wu, and Meng Wang. Attentive recurrent social recommendation. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, pages 185–194, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5657-2. doi: 10.1145/3209978.3210023. URL http://doi.acm.org/10.1145/3209978.3210023.

[29] Chenwei Cai, Ruining He, and Julian McAuley. SPMC: socially-aware personalized markov chains for sparse sequential recommendation. *CoRR*, abs/1708.04497, 2017. URL http://arxiv.org/abs/1708.04497.

[30] Wang-Cheng Kang and Julian McAuley. Self-attentive sequential recommendation. In *ICDM*, pages 197–206. IEEE Computer Society, 2018.

[31] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, pages 565–573, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5581-0. doi: 10.1145/3159652.3159656. URL http://doi.acm.org/10.1145/3159652.3159656.

[32] Dietmar Jannach and Malte Ludewig. When recurrent neural networks meet the neighborhood for session-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, pages 306–310, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4652-8. doi: 10.1145/3109859.3109872. URL http://doi.acm.org/10.1145/3109859.3109872.

[33] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. Recurrent recommender networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, WSDM '17, pages 495–503, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4675-7. doi: 10.1145/3018661.3018689. URL http://doi.acm.org/10.1145/3018661.3018689.

[34] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, pages 135–142, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-906-0. doi: 10.1145/1864708.1864736. URL http://doi.acm.org/10.1145/1864708.1864736.

[35] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, WSDM '11, pages 287–296, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0493-1. doi: 10.1145/1935826.1935877. URL http://doi.acm.org/10.1145/1935826.1935877.

[36] Menghan Wang, Xiaolin Zheng, Yang Yang, and Kun Zhang. Collaborative filtering with social exposure: A modular approach to social recommendation, 2017.

[37] Tong Zhao, Julian McAuley, and Irwin King. Leveraging social connections to improve personalized ranking for collaborative filtering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '14, pages 261–270, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-2598-1. doi: 10.1145/2661829.2661998. URL http://doi.acm.org/10.1145/2661829.2661998.

[38] Xin Wang, Wei Lu, Martin Ester, Can Wang, and Chun Chen. Social recommendation with strong and weak ties. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 5–14, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4073-1. doi: 10.1145/2983323.2983701.

[39] Weiping Song, Zhiping Xiao, Yifan Wang, Laurent Charlin, Ming Zhang, and Jian Tang. Session-based social recommendation via dynamic graph attention networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, WSDM '19, pages 555–563, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-5940-5. doi: 10.1145/3289600.3290989. URL `http://doi.acm.org/10.1145/3289600.3290989`.

[40] Pierre Deville, Dashun Wang, Roberta Sinatra, Chaoming Song, Vincent D Blondel, and Albert-László Barabási. Career on the move: Geography, stratification, and scientific impact. *Scientific reports*, 4:4770, 2014.

[41] Aaron Clauset, Samuel Arbesman, and Daniel B Larremore. Systematic inequality and hierarchy in faculty hiring networks. *Science advances*, 1(1):e1400005, 2015.

[42] Tara Safavi, Maryam Davoodi, and Danai Koutra. Career transitions and trajectories: A case study in computing. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '18, pages 675–684. ACM, 2018. ISBN 978-1-4503-5552-0. doi: 10.1145/3219819.3219863. URL `http://doi.acm.org/10.1145/3219819.3219863`.

[43] Dashun Wang, Chaoming Song, and Albert-László Barabási. Quantifying long-term scientific impact. *Science*, 342(6154):127–132, 2013.

[44] Samuel F. Way, Daniel B. Larremore, and Aaron Clauset. Gender, productivity, and prestige in computer science faculty hiring networks. In *Proceedings of the 25th International Conference on World Wide Web*, 2016. ISBN 978-1-4503-4143-1. doi: 10.1145/2872427.2883073. URL `https://doi.org/10.1145/2872427.2883073`.

[45] Samuel F. Way, Allison C. Morgan, Aaron Clauset, and Daniel B. Larremore. The misleading narrative of the canonical faculty productivity trajectory. *Proceedings of the National Academy of Sciences*, 114(44):E9216–E9223, 2017. ISSN 0027-8424. doi: 10.1073/pnas.1702121114. URL `http://www.pnas.org/content/114/44/E9216`.

[46] Shulamit Kahn. Gender differences in academic career paths of economists. *The American Economic Review*, 83(2):52–56, 1993. ISSN 00028282. URL `http://www.jstor.org/stable/2117639`.

[47] Melanie Ward. The gender salary gap in british academia. *Applied Economics*, 33(13): 1669–1681, 2001. doi: 10.1080/00036840010014445.

[48] Yong Liu, Jorge Goncalves, Denzil Ferreira, Bei Xiao, Simo Hosio, and Vassilis Kostakos. Chi 1994-2013: mapping two decades of intellectual progress through co-word analysis. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*, pages 3553–3562. ACM, 2014.

[49] Maria Biryukov and Cailing Dong. Analysis of computer science communities based on dblp. In *Proceedings of the 14th European Conference on Research and Advanced Technology for Digital Libraries*. Springer-Verlag, 2010. ISBN 3-642-15463-8, 978-3-642-15463-8. URL `http://dl.acm.org/citation.cfm?id=1887759.1887792`.

[50] Tanmoy Chakraborty and Subrata Nandi. Universal trajectories of scientific success. *Knowl. Inf. Syst.*, 54(2):487–509, February 2018. ISSN 0219-1377. doi: 10.1007/s10115-017-1080-y. URL `https://doi.org/10.1007/s10115-017-1080-y`.

[51] Grégoire Burel, Paul Mulholland, and Harith Alani. Structural normalisation methods for improving best answer identification in question answering communities. In *International Conference on World Wide Web, WWW*, 2016.

[52] Maximilian Jenders, Ralf Krestel, and Felix Naumann. Which answer is best?: Predicting accepted answers in MOOC forums. In *International Conference on World Wide Web*, 2016.

[53] Qiongjie Tian, Peng Zhang, and Baoxin Li. Towards predicting the best answers in community-based question-answering services. In *International Conference on Weblogs and Social Media, ICWSM*, 2013.

[54] Liu Yang, Minghui Qiu, Swapna Gottipati, Feida Zhu, Jing Jiang, Huiping Sun, and Zhong Chen. Cqarank: jointly model topics and expertise in community question answering. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 99–108. ACM, 2013.

[55] Alberto Barrón-Cedeno, Simone Filice, Giovanni Da San Martino, Shafiq R Joty, Lluís Màrquez, Preslav Nakov, and Alessandro Moschitti. Thread-level information for comment classification in community question answering. In *ACL (2)*, pages 687–693, 2015.

[56] Jiahui Wen, Jingwei Ma, Yiliu Feng, and Mingyang Zhong. Hybrid attentive answer selection in cqa with deep users modelling. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[57] Tsvetomila Mihaylova, Preslav Nakov, Lluis Marquez, Alberto Barron-Cedeno, Mitra Mohtarami, Georgi Karadzhov, and James Glass. Fact checking in community forums. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[58] Hyo-Jung Oh, Yeo-Chan Yoon, and Hyun Ki Kim. Finding more trustworthy answers: Various trustworthiness factors in question answering. *Journal of Information Science*, 39(4):509–522, 2013.

[59] Eric Gilbert. Widespread underprovision on reddit. In *Proceedings of the 2013 conference on Computer supported cooperative work*, pages 803–808. ACM, 2013.

[60] Xiaodong Zhang, Sujian Li, Lei Sha, and Houfeng Wang. Attentive interactive neural networks for answer selection in community question answering. In *AAAI Conference on Artificial Intelligence*, 2017.

[61] Wei Wu, Houfeng Wang, and Xu Sun. Question condensing networks for answer selection in community question answering. In *Association for Computational Linguistics*, 2018.

[62] Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In *ACL*, 2015.

[63] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. Semeval-2017 task 3: Community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 27–48, 2017.

[64] Tsvetomila Mihaylova, Gerogi Karadzhov, Pepa Atanasova, Ramy Baly, Mitra Mohtarami, and Preslav Nakov. Semeval-2019 task 8: Fact checking in community question answering. In *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)*, pages 856–865, 2019.

[65] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. Applying deep learning to answer selection: A study and an open task. In *IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU*, 2015.

[66] Sainbayar Sukhbaatar, Arthur Szlam, Jason Weston, and Rob Fergus. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, 2015.

[67] Hengtong Zhang, Yaliang Li, Fenglong Ma, Jing Gao, and Lu Su. Texttruth: An unsupervised approach to discover trustworthy information from multi-sourced text data. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2729–2737. ACM, 2018.

[68] Yaliang Li, Qi Li, Jing Gao, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. On the discovery of evolving truth. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 675–684. ACM, 2015.

[69] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552, 2017.

[70] Qi Li, Fenglong Ma, Jing Gao, Lu Su, and Christopher J Quinn. Crowdsourcing high quality labels with a tight budget. In *Proceedings of the ninth acm international conference on web search and data mining*, pages 237–246. ACM, 2016.

[71] Tathagata Mukherjee, Biswas Parajuli, Piyush Kumar, and Eduardo Pasiliao. Truthcore: Non-parametric estimation of truth from a collection of authoritative sources. In *Big Data (Big Data), 2016 IEEE International Conference on*, pages 976–983. IEEE, 2016.

[72] VG Vydiswaran, ChengXiang Zhai, and Dan Roth. Content-driven trust propagation framework. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 974–982. ACM, 2011.

[73] Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. A survey on truth discovery. *ACM Sigkdd Explorations Newsletter*, 17(2):1–16, 2016.

[74] Fenglong Ma, Yaliang Li, Qi Li, Minghui Qiu, Jing Gao, Shi Zhi, Lu Su, Bo Zhao, Heng Ji, and Jiawei Han. Faitcrowd: Fine grained truth discovery for crowdsourced data aggregation. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 745–754. ACM, 2015.

[75] Mengting Wan, Xiangyu Chen, Lance Kaplan, Jiawei Han, Jing Gao, and Bo Zhao. From truth discovery to trustworthy opinion discovery: An uncertainty-aware quantitative modeling approach. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1885–1894. ACM, 2016.

[76] Yaliang Li, Nan Du, Chaochun Liu, Yusheng Xie, Wei Fan, Qi Li, Jing Gao, and Huan Sun. Reliable medical diagnosis from crowdsourcing: Discover trustworthy answers from non-experts. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 253–261. ACM, 2017.

[77] Ellery Wulczyn, Nithum Thain, and Lucas Dixon. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1391–1399, 2017.

[78] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California, June 2016. Association for Computational Linguistics. doi: 10.18653/v1/N16-2013. URL `https://www.aclweb.org/anthology/N16-2013`.

[79] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*, 2017.

[80] Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 145–153, Republic and Canton of Geneva, CHE, 2016. International World Wide Web Conferences Steering Committee. ISBN 9781450341431. doi: 10.1145/2872427.2883062. URL `https://doi.org/10.1145/2872427.2883062`.

[81] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[82] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014. URL `http://www.aclweb.org/anthology/D14-1162`.

[83] Björn Gambäck and Utpal Kumar Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada, August 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-3013. URL `https://www.aclweb.org/anthology/W17-3013`.

[84] Ji Ho Park and Pascale Fung. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45, 2017.

[85] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760. International World Wide Web Conferences Steering Committee, 2017.

[86] Sweta Agrawal and Amit Awekar. Deep learning for detecting cyberbullying across multiple social media platforms. In *European Conference on Information Retrieval*, pages 141–153. Springer, 2018.

[87] Ziqi Zhang, David Robinson, and Jonathan Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web*, pages 745–760, Cham, 2018. Springer International Publishing. ISBN 978-3-319-93417-4.

[88] William Warner and Julia Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the second workshop on language in social media*, pages 19–26. Association for Computational Linguistics, 2012.

[89] Leandro Silva, Mainack Mondal, Denzil Correa, Fabrício Benevenuto, and Ingmar Weber. Analyzing the targets of hate in online social media. In *Tenth International AAAI Conference on Web and Social Media*, 2016.

[90] John Pavlopoulos, Prodromos Malakasiotis, Juli Bakagianni, and Ion Androutsopoulos. Improved abusive comment moderation with user embeddings. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 51–55, 2017.

[91] Jing Qian, Mai ElSherief, Elizabeth Belding, and William Yang Wang. Leveraging intra-user and inter-user representation learning for automated hate speech detection. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 118–123, 2018.

[92] Pushkar Mishra, Marco Del Tredici, Helen Yannakoudakis, and Ekaterina Shutova. Author profiling for abuse detection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1088–1098, Santa Fe, New Mexico, USA,

August 2018. Association for Computational Linguistics. URL `https://www.aclweb.org/anthology/C18-1093`.

[93] Pushkar Mishra, Marco Del Tredici, Helen Yannakoudakis, and Ekaterina Shutova. Abusive language detection with graph convolutional networks. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2145–2150, 2019.

[94] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *CoRR*, abs/1609.02907, 2016. URL `http://arxiv.org/abs/1609.02907`.

[95] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *European Semantic Web Conference*. Springer, 2018.

[96] Aravind Sankar, Adit Krishnan, Zongjian He, and Carl Yang. Rase: Relationship aware social embedding. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[97] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In *International Conference on Knowledge Discovery and Data Mining*, 2014.

[98] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *International Conference on Knowledge Discovery and Data Mining*, 2016.

[99] Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Revisiting semi-supervised learning with graph embeddings. In *International Conference on Machine Learning, ICML*, pages 40–48, 2016.

[100] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. LINE: large-scale information network embedding. In *International Conference on World Wide Web, WWW*, 2015.

[101] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, 2016.

[102] David K. Duvenaud, Dougal Maclaurin, Jorge Aguilera-Iparraguirre, Rafael Gómez-Bombarelli, Timothy Hirzel, Alán Aspuru-Guzik, and Ryan P. Adams. Convolutional networks on graphs for learning molecular fingerprints. In *Advances in Neural Information Processing Systems*, 2015.

[103] Tyler Derr, Yao Ma, and Jiliang Tang. Signed graph convolutional network. *CoRR*, 2018.

[104] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, 2017.

[105] Chenyi Zhuang and Qiang Ma. Dual graph convolutional networks for graph-based semi-supervised classification. In *World Wide Web Conference*, 2018.

[106] Aravind Sankar, Yanhong Wu, Liang Gou, Wei Zhang, and Hao Yang. Dynamic graph representation learning via self-attention networks. *arXiv preprint arXiv:1812.09430*, 2018.

[107] Rianne van den Berg, Thomas N. Kipf, and Max Welling. Graph convolutional matrix completion. *CoRR*, abs/1706.02263, 2017.

[108] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18*, 2018. doi: 10.1145/3219819.3219890. URL `http://dx.doi.org/10.1145/3219819.3219890`.

[109] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Eric Zhao, Jiliang Tang, and Dawei Yin. Graph neural networks for social recommendation, 2019.

[110] Le Wu, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. A neural influence diffusion model for social recommendation. In *Proceedings of the 42Nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR'19, pages 235–244, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6172-9. doi: 10.1145/3331184.3331214. URL `http://doi.acm.org/10.1145/3331184.3331214`.

[111] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks. *International Conference on Learning Representations*, 2018. URL `https://openreview.net/forum?id=rJXMpikCZ`.

[112] Kanika Narang, Austin Chung, Hari Sundaram, and Snigdha Chaturvedi. Discovering archetypes to interpret evolution of individual behavior. *arXiv preprint arXiv:1902.05567*, 2019.

[113] Jaewon Yang and Jure Leskovec. Patterns of temporal variation in online media. In *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*, New York, NY, USA, 2011. ISBN 978-1-4503-0493-1. doi: 10.1145/1935826.1935863. URL `http://doi.acm.org/10.1145/1935826.1935863`.

[114] Julian John McAuley and Jure Leskovec. From amateurs to connoisseurs: Modeling the evolution of user expertise through online reviews. In *Proceedings of the 22Nd International Conference on World Wide Web*. ACM, 2013. ISBN 978-1-4503-2035-1. doi: 10.1145/2488388.2488466. URL `http://doi.acm.org/10.1145/2488388.2488466`.

[115] Cristian Danescu-Niculescu-Mizil, Robert West, Dan Jurafsky, Jure Leskovec, and Christopher Potts. No country for old members: User lifecycle and linguistic change in online communities. In *Proceedings of the 22Nd International Conference on World Wide Web*. ACM, 2013. ISBN 978-1-4503-2035-1. doi: 10.1145/2488388.2488416. URL `http://doi.acm.org/10.1145/2488388.2488416`.

[116] Lei Li and B. Aditya Prakash. Time series clustering: Complex is simpler! In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, USA, 2011. Omnipress. ISBN 978-1-4503-0619-5. URL `http://dl.acm.org/citation.cfm?id=3104482.3104506`.

[117] T. Chakraborty, S. Kumar, M. D. Reddy, S. Kumar, N. Ganguly, and A. Mukherjee. Automatic classification and analysis of interdisciplinary fields in computer sciences. In *2013 International Conference on Social Computing*, pages 180–187, Sep. 2013. doi: 10.1109/SocialCom.2013.34.

[118] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15 Companion, pages 243–246. ACM, 2015. ISBN 978-1-4503-3473-0. doi: 10.1145/2740908.2742839. URL `http://doi.acm.org/10.1145/2740908.2742839`.

[119] The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, November 1999. URL `http://ilpubs.stanford.edu:8090/422/`.

[120] Kanika Narang, Susan T. Dumais, Nick Craswell, Dan Liebling, and Qingyao Ai. Large-scale analysis of email search and organizational strategies. In *Proceedings of the 2017 Conference on Conference Human Information Interaction and Retrieval*, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4677-1. doi: 10.1145/3020165.3020175. URL `http://doi.acm.org/10.1145/3020165.3020175`.

[121] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B*, 39:1–38, 1977.

[122] Lawrence R. Rabiner. Readings in speech recognition. chapter A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pages 267–296. Morgan Kaufmann Publishers Inc., 1990. ISBN 1-55860-124-4. URL `http://dl.acm.org/citation.cfm?id=108235.108253`.

[123] Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(2):411–423, 2001.

[124] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Statistics*, 22:79–86, 1951. ISSN 0003-4851. doi: 10.1214/aoms/1177729694. URL `https://doi.org/10.1214/aoms/1177729694`.

[125] Cyril H Goulden et al. Methods of statistical analysis. *Methods of statistical analysis.*, 1949.

[126] William H. Kruskal and W. Allen Wallis. Use of ranks in one-criterion variance analysis. volume 47, pages 583–621. Taylor & Francis, 1952.

[127] B. L. WELCH. The generalization of student's problem when several different population varlances are involved. *Biometrika*, 34(1-2):28–35, 01 1947. ISSN 0006-3444. doi: 10.1093/biomet/34.1-2.28. URL `https://dx.doi.org/10.1093/biomet/34.1-2.28`.

[128] Shima Ghassempour, Federico Girosi, and Anthony Maeder. Clustering multivariate time series using hidden markov models. In *International journal of environmental research and public health*, 2014.

[129] B.H. Juang and Lawrence R. Rabiner. A probabilistic distance measure for hidden markov models. 64, 02 1985.

[130] Helmut Ltkepohl. *New Introduction to Multiple Time Series Analysis*. Springer Publishing Company, 2007. ISBN 3540262393, 9783540262398.

[131] Alex Morales, Kanika Narang, Chengxiang Zhai, and Hari Sundaram. Crowdqm: Learning user aspect-reliability and comment trustworthiness in discussion forums. In *24th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, 2020.

[132] Guoliang Li, Jiannan Wang, Yudian Zheng, and Michael Franklin. Crowdsourced data management: A survey. In *Data Engineering (ICDE), 2017 IEEE 33rd International Conference on*, pages 39–40. IEEE, 2017.

[133] Bo Zhao and Jiawei Han. A probabilistic model for estimating real-valued truth from conflicting sources. 2012.

[134] Xiaoxin Yin, Jiawei Han, and Philip S. Yu. Truth discovery with multiple conflicting information providers on the web. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, pages 1048–1052, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-609-7. doi: 10.1145/1281192.1281309. URL `http://doi.acm.org/10.1145/1281192.1281309`.

[135] Alban Galland, Serge Abiteboul, Amélie Marian, and Pierre Senellart. Corroborating information from disagreeing views. In *Proceedings of the Third ACM International Conference on Web Search and Data Mining*, WSDM '10, pages 131–140, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-889-6. doi: 10.1145/1718487.1718504. URL `http://doi.acm.org/10.1145/1718487.1718504`.

[136] Xin Luna Dong, Laure Berti-Equille, and Divesh Srivastava. Integrating conflicting data: The role of source dependence. *Proc. VLDB Endow.*, 2(1):550–561, August 2009. ISSN 2150-8097. doi: 10.14778/1687627.1687690. URL `https://doi.org/10.14778/1687627.1687690`.

[137] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *WSDM*, 2008.

[138] Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. *arXiv preprint arXiv:1602.07019*, 2016.

[139] Dimitri P Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999.

[140] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[141] Shanshan Lyu, Wentao Ouyang, Yongqing Wang, Huawei Shen, and Xueqi Cheng. What we vote for? answer selection from user expertise view in community question answering. In *The World Wide Web Conference*, WWW '19, pages 1198–1209, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6674-8. doi: 10.1145/3308558.3313510. URL `http://doi.acm.org/10.1145/3308558.3313510`.

[142] Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1187–1198. ACM, 2014.

[143] Qi Li, Yaliang Li, Jing Gao, Lu Su, Bo Zhao, Murat Demirbas, Wei Fan, and Jiawei Han. A confidence-aware approach for truth discovery on long-tail data. *Proceedings of the VLDB Endowment*, 8(4):425–436, 2014.

[144] Jeff Pasternack and Dan Roth. Making better informed trust decisions with generalized fact-finding. In *IJCAI*, 2011.

[145] Kanika Narang, Yitong Song, Alexander Schwing, and Hari Sundaram. Fuserec: Fusing user and item homophily modeling with temporal recommender systems. In *Under review at ECML-PKDD*, 2020.

[146] L. Wu, P. Sun, R. Hong, Y. Ge, and M. Wang. Collaborative neural social recommendation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pages 1–13, 2018. ISSN 2168-2216. doi: 10.1109/TSMC.2018.2872842.

[147] Adit Krishnan, Ashish Sharma, Aravind Sankar, and Hari Sundaram. An adversarial approach to improve long-tail performance in neural collaborative filtering. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, CIKM '18, pages 1491–1494, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-6014-2. doi: 10.1145/3269206.3269264. URL `http://doi.acm.org/10.1145/3269206.3269264`.

[148] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. Challenging the long tail recommendation. *Proc. VLDB Endow.*, 5(9):896–907, May 2012. ISSN 2150-8097. doi: 10.14778/2311906.2311916.

[149] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[150] Weike Pan and Li Chen. Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering. In *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence*, IJCAI '13, pages 2691–2697. AAAI Press, 2013. ISBN 978-1-57735-633-2. URL `http://dl.acm.org/citation.cfm?id=2540128.2540516`.

[151] Jiliang Tang, Huiji Gao, Huan Liu, and Atish Das Sarma. eTrust: Understanding trust evolution in an online world. In *KDD*, 2012.

[152] Ruining He, Chen Fang, Zhaowen Wang, and Julian McAuley. Vista: A visually, socially, and temporally-aware model for artistic recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, pages 309–316, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4035-9. doi: 10.1145/2959100.2959152. URL `http://doi.acm.org/10.1145/2959100.2959152`.

[153] X. He, Z. He, J. Song, Z. Liu, Y. Jiang, and T. Chua. Nais: Neural attentive item similarity model for recommendation. In *IEEE TKDE'18*.

[154] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. Session-based recommendation with graph neural networks. In *AAAI'19*.

[155] Kanika Narang, Chaoqi Yang, Adit Krishnan, Junting Wang, Hari Sundaram, and Carolyn Sutter. An induced multi-relational framework for answer selection in community question answer platforms. *arXiv preprint arXiv:1911.06957*, 2019.

[156] Christopher J. Burges, Robert Ragno, and Quoc V. Le. Learning to rank with non-smooth cost functions. In *Advances in Neural Information Processing Systems*. MIT Press, 2007.

[157] Christopher J. C. Burges. From RankNet to LambdaRank to LambdaMART: An overview. Technical report, Microsoft Research, 2010.

[158] Qiongjie Tian and Baoxin Li. Weakly hierarchical lasso based learning to rank in best answer prediction. In *International Conference on Advances in Social Networks Analysis and Mining, ASONAM*, 2016.

[159] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv:1710.10903*, 2017.

[160] Ivan Brugere, Brian Gallagher, and Tanya Y. Berger-Wolf. Network structure inference, a survey: Motivations, methods, and applications. *ACM Comput. Surv.*, 2018. doi: 10.1145/3154524. URL `http://doi.acm.org/10.1145/3154524`.

[161] Lingfei Wu, Jacopo A. Baggio, and Marco A. Janssen. The role of diverse strategies in sustainable knowledge production. *PLoS ONE*, 2016.

[162] Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill™: A bayesian skill rating system. In *International Conference on Neural Information Processing Systems*, 2006.

[163] Fan R. K. Chung. *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society, 1997.

[164] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

[165] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional neural network: A deep learning framework for traffic forecasting. *CoRR*, abs/1709.04875, 2017.

[166] Golnoosh Farnadi, Jie Tang, Martine De Cock, and Marie-Francine Moens. User profiling through deep multimodal fusion. In *International Conference on Web Search and Data Mining*, WSDM '18. ACM, 2018.

[167] Holger Schwenk and Yoshua Bengio. Boosting neural networks. *Neural Computation*, 12(8):1869–1887, 2000.

[168] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*. Springer-Verlag, 1995.

[169] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *International Conference on Neural Information Processing Systems*, NIPS'16, 2016. ISBN 978-1-5108-3881-9.

[170] Eric Gilbert. Widespread underprovision on reddit. In *Conference on Computer Supported Cooperative Work*, CSCW '13, New York, NY, USA, 2013. ACM. URL `http://doi.acm.org/10.1145/2441776.2441866`.

[171] Xin-Jing Wang, Xudong Tu, Dan Feng, and Lei Zhang. Ranking community answers by modeling question-answer relationships via analogical reasoning. In *SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '09. ACM, 2009.

[172] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[173] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 2008.

[174] Ming Tan, Bing Xiang, and Bowen Zhou. Lstm-based deep learning models for non-factoid answer selection. *CoRR*, abs/1511.04108, 2015.

[175] Jiajun Lu, Theerasit Issaranon, and David A Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. In *ICCV*, pages 446–454, 2017.

[176] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[177] Kanika Narang and Chris Brew. Hate speech classification using syntactic dependency graphs. In *To be submitted*, 2020.

[178] Mai ElSherief, Vivek Kulkarni, Dana Nguyen, William Yang Wang, and Elizabeth Belding. Hate lingo: A target-based linguistic analysis of hate speech in social media. In *Twelfth International AAAI Conference on Web and Social Media*, 2018.

[179] Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does bert look at? an analysis of bert's attention. *arXiv preprint arXiv:1906.04341*, 2019.

[180] Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018. URL `https://www.aclweb.org/anthology/D18-1244`.

[181] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*, 2017.

[182] PETER V. MARSDEN and NOAH E. FRIEDKIN. Network studies of social influence. *Sociological Methods & Research*, 22(1):127–151, 1993. doi: 10.1177/0049124193022001006.

[183] Miller Mcpherson, Lynn Smith-Lovin, and James Cook. Birds of a feather: Homophily in social networks. *Annual Review of Sociology*, 27:415–, 01 2001. doi: 10.3410/f.725356294.793504070.

[184] Matthew Zook. Mapping racist tweets in response to president obama's re-election, 2012. URL `floatingsheep.org/2012/11/mapping-racist-tweets-in-response-to.html`.

[185] Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1060.

[186] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-2034. URL `https://www.aclweb.org/anthology/P16-2034`.

[187] Federico Monti, Karl Otness, and Michael M. Bronstein. Motifnet: a motif-based graph convolutional network for directed graphs, 2018.

[188] Aravind Sankar, Xinyang Zhang, and Kevin Chen-Chuan Chang. Motif-based convolutional neural network on graphs, 2017.