You Should Read This! Let Me Explain You Why

Explaining News Recommendations to Users

Roi Blanco[‡] Diego Ceccarelli [†§] Claudio Lucchese [†] Raffale Perego[†]
Fabrizio Silvestri[†]
‡ Yahoo! Research. Barcelona, Spain – roi@yahoo-inc.com
†ISTI-CNR, Pisa, Italy – {firstname.lastname}@isti.cnr.it
§ Dipartimento di Informatica, Università di Pisa

ABSTRACT

Recommender systems have become ubiquitous in content-based web applications, from news to shopping sites. Nonetheless, an aspect that has been largely overlooked so far in the recommender system literature is that of automatically building explanations for a particular recommendation. This paper focuses on the news domain, and proposes to enhance effectiveness of news recommender systems by adding, to each recommendation, an explanatory statement to help the user to better understand if, and why, the item can be her interest. We consider the news recommender system as a black-box, and generate different types of explanations employing pieces of information associated with the news. In particular, we engineer text-based, entity-based, and usage-based explanations, and make use of a Markov Logic Networks to rank the explanations on the basis of their effectiveness. The assessment of the model is conducted via a user study on a dataset of news read consecutively by actual users. Experiments show that news recommender systems can greatly benefit from our explanation module as it allows users to discriminate between interesting and not interesting news in the majority of the cases.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information Filtering, Search Process*

General Terms

Algorithms, Design, Experimentation

Keywords

Query log analysis, News recommendation, Recommendation Snippets, Markov Logic Networks

CIKM'12, October 29–November 2, 2012, Maui, HI, USA.

1. INTRODUCTION

The large amount of research work done in the last decade in the area of recommender systems was mainly motivated by the great success these kinds of systems have, and are still gaining, in real-life Web applications. Amazon is usually credited among the first ones to exploit the potential of these tools to enhance user engagement. Recommender systems, however, are becoming increasingly popular in diverse application domains. Originally thought for products, recommender systems are now popular also for other types of data, such as music, videos, queries, friends on social networks, news, among others.

News systems in particular, benefit from recommendations given the fact that online news systems are exploratory by nature. People browse through the list of daily news usually driven by personal interest, curiosity, or both. Due to the amount of news items available, online news services deploy recommender systems that help the users find potentially interesting news. To the best of our knowledge, these recommendations are displayed with a very shallow explanation of why a given news has been suggested for reading (e.g. a snippet of text from the news, or the number of views of the news). However, even in the case that a relevant news item has been recommended for a user, the action of accessing the item will largely depend on how well this interestingness is assessed by the user *before* clicking on the item. Not generating an interesting explanation might downgrade the performance of recommender systems, their applicability and, consequently, their value for monetization.

Explaining news recommendations is the goal *at-large* of the research presented in this paper. In particular, we aim at enhancing the users experience on news platforms, like Google News, Yahoo! News, and the alike by motivating the recommended news shown by means of a tool that automatically generate brief, yet significant, explanations.

Choosing the right explanation models to be used by a news recommender system is one of the challenges we face in this research. Notably, these explanations do not aim at increasing the click-through ratio of recommendations. Rather, they try to help users to realize whether a news item can be of her interest or not, by providing her with additional information on the recommendations being proposed. The ultimate goal is thus increasing the engagement of users with news platforms, by presenting a correct explanation on *why* the recommendation has been shown.

Particularly, in this paper we introduce different ways of generating explanations for the recommended news, and we develop a machine learning based method whose goal is to

rank these explanations in order to maximize the usefulness of the recommendation itself. Given a pair of news items, the one actually read by a user and the one suggested by a news recommender considered as a *black-box*, we generate a set of possible explanations for the latter.

The paper presents the following contributions:

- Automatic explanation of news recommendations is, to the best of our knowledge, a new problem. Previous studies either focused on different media recommended or analyzed different visualization strategies for groups of news.
- The techniques to automatically generate candidate explanations on the basis of features extracted from the content and the context of the last news read and the recommended one.

Related Work. The problem of recommending items to users is well-known in information retrieval research community and has received a lot of attention in the last decade [1]. The problem of explaining recommendations is, of course, tightly coupled.

Herlocker *et al.* [3] propose a taxonomy categorizing different approaches for explaining recommendations. Authors support the choice of a black box model where explanations are produced independently of the recommendation algorithm, rather than a white box approach, where the explanations are generated on the basis of the complex rules and technique exploited buy the algorithm, thus being too complex to help the user in understanding the recommendations. On the same line, Vig *et al.* [5] discuss the difference between transparency and justification: the former allows the disclosure of how the system really works, while the latter can be decoupled from the recommendation algorithm. Often justifications are preferred because i) the real algorithm is difficult to explain (e.g., matrix factorizations), ii) the algorithm is secret, iii) more flexible design of the explanations. For these reasons, in this work we considered the recommender system as a black box. The problem of retrieving explanations has also been studied for entity search [2].

In this work, we make use of the Markov Logic Networks (MLNs) of Richardson and Domingos 2006 [4], a joint model that combines first order logic (FOL) and Markov networks. The model is able to capture contextual information and long-range dependencies between features, which are critical to the task addressed here. Markov Logic Netwoks have been used elsewhere for a plethora of natural language and data mining applications. A key advantage of MLNs is that they allow to express semantically-rich formulas to capture a variety of long-term dependencies between features in a seamless fashion. In essence, they act as an interface layer between the learning process and the domain knowledge engineering.

2. RANKING EXPLANATIONS

News systems usually exploit some recommendation algorithm able to suggest one or more interesting news being related to the news a user is currently reading. Explaining news recommendations consists in building and ranking a predefined list of explanation templates.

Let $I = \{i_1, i_2, \ldots, i_n\}$ be the set of news items, and let $E = \{e_1, e_2, \ldots, e_m\}$ the set of possible explanations. Given a *source news item* i_s, i.e., the news a user was reading, and a *target news item* i_t, i.e., the recommended news, we aim at ordering the elements in E according to their *relevance* or *elucidatory value*. The ordering shall reflect how good the explanation is in helping a user that read i_s in the past to take the decision of reading or not news i_t. The idea is that the first explanation in the ordering has to be the most *explicative* for the user.

Given a pair $(i_s, i_t) \in \mathcal{I}$, with $\mathcal{I} = (I \times I)$, such that at least a user has been suggested a news i_t after reading i_s, we denote with $Y = \{y_1, \ldots, y_{|E|}\}$, with $y_j \in E$, a permutation of E where the explanations are sorted in decreasing order of relevance. If Y^{st} denotes the output for (i_s, i_t), then y_1^{st} is the best explanation for i_t given i_s. The problem is to learn a function $f : \mathcal{I} \rightarrow Y$, over a class of functions H, such that it minimizes a loss function $\Delta(Y, \hat{Y})$ measuring the penalty for making a prediction Y if the correct output is $\hat{Y}$. If $P(\mathcal{I}, Y)$ denotes the data generation distribution, then the goal is to minimize the risk:

$$f = \arg\min_{f \in H} \int \Delta(f(i_s, i_t), Y^{st}) \; d\, P(\mathcal{I}, \mathcal{Y}) \qquad (1)$$

Our goal is to find such f that, for a given pair of news items, is able to predict the optimal ranking of the explanations in the set E. Note that the problem formulation does not make any assumptions on what algorithms are used by the news recommender system. This is a carefully informed choice as we aim at giving a system that is oblivious with respect to the recommendation algorithm. This assumption has two major benefits. The first one is that we can take our explanation method and adapt it to any news site using any recommendation method and, being decoupled, any update on the recommender system would not imply an update on the explanation system. Secondly, we do not disclose any information about the recommendation algorithm that is usually a, well-kept, trade secret.

As usual in this cases, at recommendation time the learned function f is applied to the features extracted on-the-fly from the news items and explanations are ranked accordingly.

3. GENERATING EXPLANATIONS

Our goal is to generate for each pair of read and recommended news a small set of explanations that can possibly capture the interest of the reader and make the news more appealing. Therefore, one of the first problems we deal with is to select what kind of explanations we have to generate and how. We adopted a very pragmatic approach and we come up with a set of 16 different explanations grouped into three different classes according to the types of features used to generate them. In particular we distinguish between *text*-based, *entity*-based, and *usage*-based explanations. In Table 1 we report the list of the 16 different kinds of explanations grouped by their class:

Text-based The explanations in this class have the common characteristic to be generated by exploiting the textual content in either the read or recommended news.

Entity-based Explanations using entities, or images associated with them, as a mean to convey the reason why a news has been recommended to a user.

Usage-based Explanations of this class are extracted by considering how news are accessed by users.

Observe that for each pair of news, it is not always possible to produce all the 16 explanations. The snippets generated for the explanations contain two sentences and do not exceed 200 characters.

	Explanation	Description
text based	SIMILARITY	Considers the cosine similarity between the content of the source and target news. News are recommended by showing the title and the first sentence
	SIMILARITYSNIPPET	The explanation consists in providing a snippet containing the two most similar sentences extracted from the read and recommended news.
	TARGETSNIPPET	This explanation is constructed by simply taking the first two sentences appearing in the recommended news.
	TAGSPLANATION	[5] The explanation exploits the common tags X associated with the read and recommended news, if any. Tags are generated by taking the 15 terms with the highest TF-IDF from each news.
entity based	SHAREDENTITY	The explanation tells to the user that a given named entity X is shared between the read and recommended news.
	TARGETENTITY	The explanation presents to the user the named entity X extracted from the recommended news if it does not appear in the read news.
	DISTINCTENTITIES	The explanation presents the two main distinct entities extracted from the read and recommended news.
	TARGETIMAGE	The explanation shows an image X associated with the main named entity represented in the recommended news. The image shown is taken from *Freebase*, using the provided API[1]
	IMAGES	The explanation shows two images X and Y, associated with entities represented in the read and recommended news, respectively
	SHAREDPLACES	The explanation proposes the geo-names shared between the source and target news, if any
	TARGETPLACES	This explanation suggests that the recommended news refers to some geo-names
	CATEGORIES	The explanation exploits the common categories associated with the read and recommended news, if any. Categories are provided by the news provider.
usage based	POPULARITY	The explanation highlights the popularity of the recommended news
	QUERIES	The explanation is given by the set S of terms shared among the past queries for which the recommended news was retrieved and clicked
	TARGETBIASEDSNIPPET	The explanation consists in providing a query-biased snippet X for the recommended news.
	SOURCEBIASEDSNIPPET	Similar to TARGETBIASEDSNIPPET except for the fact that the terms used are those of the queries associated with the read news and not the recommended one.

Table 1: Different types of explanations

4. MODELING AND LEARNING

This section introduces the main features of news and recommendations that are being engineered in the system, and a brief introduction to the learning model employed to rank explanations.

4.1 Features Used

Given a quadruple (i_s, i_t, e_i, y) composed by source and target news, one of the explanations introduced in Section 3 we extract the set of features described in the list below. The features are used for learning an explanation ranking, and exploit both the entities and the geo-names extracted from both the source and the target news.

COS the cosine similarity $cosine(i_s, i_t)$ between the bag-of-word representations of both the read and the recommended news;

E1 a binary value stating the presence or absence in the target news i_t of entities. This feature is 1 when there exists an entity in the text of i_t.

E2 a binary value stating if ($\mathbf{E2} = 1$) target and source news share some common entity;

E3 a binary value stating if ($\mathbf{E3} = 1$) target and source news share the main entity, i.e. the entity appearing most frequently within the news.

SE1 the number of terms shared between the snippet in e_i (if any) and the labels of the entities recognized in the source news i_s;

SE2 the number of terms shared between the snippet in e_i (if any) and the labels of the entities recognized in the target news i_t;

ST1 the number of terms shared between the snippet in e_i (if any) and the title of the source news i_s;

ST2 the number of terms shared between the snippet in e_i (if any) and the title of the target news i_t;

SG1 the number of terms shared between the snippet in e_i (if any) and the geo-names in the source news i_s;

SG2 the number of terms shared between the snippet in e_i (if any) and the geo-names in the target news i_t.

Note that features SE1, SE2, ST1, ST2, SG1, SG2, are only generated for explanations based on the snippets, i.e., SIMILARITYSNIPPET, TARGETSNIPPET, TARGETBIASEDSNIPPET and SOURCEBIASEDSNIPPET.

4.2 Learning relevant explanations with Markov Logic Networks

Given the variety of features embodied in the model, we would like to build a class of functions f that are able to account for structured dependencies in input features. We make use of the Markov Logic Networks (MLNs) of Richardson and Domingos 2006 [4], a joint model that combines first order logic (FOL) and Markov networks. The model is able to capture contextual information and long-range dependencies between features, which are critical to the task addressed here.

MLNs have further interest in that most common probabilistic models can be succinctly formulated as MLNs, including HMMs, CRFs, logistic regression, Bayesian networks, and so on.

To train the MLN model we exploited some ad-hoc rules involving predicates modeling relevance, relatedness, etc. As

a rule of thumb, each signal described in Section 3 will have a predicate to incorporate it into the model.

We make use of *Alchemy*,[2] a software toolkit that provides a series of algorithms for statistical relational learning and probabilistic logic inference, based on the Markov Logic Networks representation.

In the following we illustrate some of the logic formulas that we used to model the problem and to train the learning model. The domain developed includes a number of ground terms, for instance:

```
Relevant(rec, expl)
Rank(rec, expl, rank)
HasExpl(rec,expl)
Similar(rec, similarity!)
Related(rec)
ShareEntities(rec)
TargetHasEntities(rec)
HasSourceEntityTerms(rec,expl)
```

We are interested in learning the probability that an explanation is relevant for a particular couple of news (i.e., `Relevant`), and the probability that an explanation will have a certain rank for a recommendation (`Rank`). There are several features we can exploit, for example: the possibility to produce a certain explanation for the given pair of news (`HasExpl`), the cosine similarity between the news (`Similar`), if the news are related or not (`Related`), if they share some entities (`ShareEntities`), etc. As a rule of thumb, each signal described in Section 3 will have a predicate to incorporate it into the model.

The system is described by means of rules in first order logic; the following two rules that define two straightforward facts:

```
!HasExpl(r,e) => !Relevant(r,e).
HasExpl(r,+e) => Relevant(r,+e)
```

The former means that when an explanation `e` cannot be computed for a pair `r`, then the explanation is not relevant for `r` (the dot at the end of the rule specifies an *hard constraint*), the latter denotes a set of rules (one for each explanation). The model learns, for each instantiated rule (e.g., `HasExpl(r,IMAGES) => Relevant(r,IMAGES)`), how much a particular explanation type is relevant for an explanation.

Given the expressiveness of FOL, deriving rules is straightforward to produce. To reach a reasonably performing system, we devise several other rules involving different types of features, like:

```
ShareMainEntity(r) => Relevant(r,+e)
TargetHasEntities(r) => Relevant(r,+e)
Related(r) => Relevant(r,+e)
```

The MLN will learn a weight for each rule, out of a training file with a set of ground atoms. Then, given a set of test instances, the system performs inference and estimates the rank of a given set of explanations.

5. EXPERIMENTS

Dataset. Since there is no benchmark dataset for the problem at hand, we generated an evaluation dataset as follows. The dataset contains quadruples in the form (i_s, i_t, e, r), this is, a pair of source and target news items, an explanation e and a score r for e. This allows to evaluate the goodness of a given algorithm at predicting r, or to derive the ranked list of explanations and measure the ranking performance using standard information retrieval metrics.

[2] `http://alchemy.cs.washington.edu/`

Explanations Evaluation Dataset	
Number of Evaluators	5
Number of Evaluations quadruples (i_s, i_t, e, r)	1,632
Distinct Pairs (i_s, i_t)	120
Avg. Explanations per Recommendation	8
Distinct i_s	29
Distinct i_t	98
Related pairs (i_s, i_t)	36
Not Related pairs (i_s, i_t)	84
Avg. Pairwise Cohen's κ (scores)	0.42
Avg. Pairwise % agreement (scores)	68%
Avg. Pairwise Cohen's κ (relatedness)	0.65
Avg. Pairwise % agreement (relatedness)	83%

Table 2: Dataset and Evaluation statistics

The first step in creating the dataset consists in obtaining valid news items pairs. Each pair should consist of the news currently read and a recommended news. To avoid overfitting with recommendations generated using a particular system we resort to exploit the information recorded by a popular search engine toolbar. The toolbar service logs the pages visited by the users over time. We filtered only the pages from the Yahoo! news portal producing for each user a chronologically ordered sequence of news items. The hypothesis is that a perfect news recommender system should be able to recommend for a given news the immediately successive read news item. On the basis of this assumption, we consider a pair of consecutive news in the logs as an evidence that some recommender system suggested the second news on the basis of the first one, and that the user actually read both of them.

The dataset we are considering is made up of 121 distinct news items forming 120 news pairs. The dwell time on the news pair is about 7 minutes on average, meaning that both the news were likely to be of interest to the user. The pairs were selected randomly from the daily activity of over 100K users. For each pair we generated the explanations as described in Section 3. Human assessors, via an ad-hoc web-based assessing platform scored the explanations according to their capability at helping the user in deciding whether or not the target news is worth of being read, on a scale from 1 (bad) to 5 (perfect), with a *don't know* option. This means that the explanation should make it clear the relation between news i_s, and i_t. Nonetheless, it might happen that there is not a clear relation between the pair of news items. This is very typical when the user is reading the news of the day and switching between a variety of topics. In this case, we asked the reviews to rank higher the explanations that better help the user in understanding that the news are not content related.

Table 2 reports the statistics on the dataset obtained. On average 8 different explanations could be produced for each news item pair, resulting in a total of 1,632 evaluations, due to the overlapping evaluations on the first 20 news item pairs.

Effectiveness of explanations. We assess in this section the effectiveness of the following explanation strategies:

	Measure	POP	MLN
	NDCG	0.7458	**0.7860**
	MAP	0.4366	**0.4917**
@1	Precision	**0.7729**	0.7432
	Recall	**0.2325**	0.2178
	F-Measure	**0.3471**	0.3259
@2	Precision	**0.7279**	0.7131
	Recall	**0.4296**	0.4196
	F-Measure	**0.5195**	0.5038
@3	Precision	0.7135	**0.7353**
	Recall	0.6372	**0.6632**
	F-Measure	0.6437	**0.6634**

Table 3: Average performance of the different explanation ranking strategies computed using 10-fold cross-validation. Best results in boldface.

- *Popular explanation first* [POP]. For each recommended item we rank explanations only on the basis of their popularity and relevance in the training set, placing the most popular explanation first.
- *MLN-Based* [MLN]. We generate a Markov Logic Network-based model.

We report the results of the different methods using traditional performance metrics such as: Precision@k, Recall@k, F-measure@k with cut-offs $k = 1, 2, 3$, Mean Average Precision (MAP), and Normalized Discounted Cumulative Gain (NDCG). We make the following two assumptions: (i) we consider an explanation relevant if and only if an evaluator marked it as good, excellent or perfect, (ii) in case of more than a judgement, we consider the average rate. Results achieved via 10-fold cross validation are presented in Table 3. Results show that POP and MLN have similar performance in terms of precision, recall, and F-measure for cut offs equal to 1 and 2, being the MLN superior in terms of MAP and NDCG. POP is able to rank well only the first two explanations. On the other hand, MLN is more effective in ranking all the available explanations (according to F-measure@3, to MAP and NDCG), thus matching the complex relationships between news and recommendations.

The goodness of MLN would also make it possible to provide more complex explanations resulting from the combination of those previously described in Section 3.

6. CONCLUSIONS AND FUTURE WORK

In this paper we investigated the novel problem of devising an effective way to automatically explain news recommendations to enhance user experience on online news platforms. First, we created a dataset of news and related recommendations from the data recorded by the browser toolbar of a popular Web search service. Given a pair of news, the one actually read by a user and the one suggested by a black-box recommender, our two-steps goal was to generate a set of suitable explanations by exploiting pieces of information from content and context of the news, and then to rank these explanations on the basis of their expected usefulness. Regarding the generation of candidate explanations, we engineered the methods for building automatically 16 different types of suitable explanations. The techniques devised exploit text-based, entity-based, and usage-based features. Most of the features used to describe news and recommendations rely on textual similarity between sentences or entities, but the most interesting are extracted or enriched by exploiting external knowledge bases such as *Freebase*, and browser toolbar data that record how the news items have been searched for and consumed. We pursued two distinct strategies to address the problem of ranking candidate explanations: (*i*) a static (explanation popularity-based) approach, and (ii) a matching learning approach. Specifically, for the matching learning approach we employed Markov Logic Networks for learning to rank the set of explanations generated.

The model was trained with a human-assessed dataset, where we asked a set of assessors to score the explanations generated for 120 pairs of news items. We are confident this dataset can be of help for future studies about this topic.

Experimental results showed that the method provides high-quality explanations, and it is able to outperform state-of-the-art structured learning to rank approaches, by a percentage ranging from 43% (POP strategy) to 51% (MLN strategy) in the case of NDCG measure. We also get similar figures in the case of MAP.

The work introduced here represents a first step towards generating meaningful explanations of recommender systems. Future research will try to learn the interplay between explanations and other types of factors in a fully fledged recommender system, like user engagement, dwelling time or perception on system quality. An interesting direction to investigate would be that of learning non-trivial combinations of different explanations, and measuring their direct impact on users. In the case, the Markov Logic Network machinery could be useful to learn the right way to merge them, in order to devise a readable friendly personalized explanation.

7. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *Knowledge and Data Engineering, IEEE Transactions on*, 17(6):734 – 749, june 2005.

[2] R. Blanco and H. Zaragoza. Finding support sentences for entities. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 339–346, New York, NY, USA, 2010. ACM.

[3] J. L. Herlocker, J. A. Konstan, and J. Riedl. Explaining collaborative filtering recommendations. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, CSCW '00, pages 241–250, New York, NY, USA, 2000. ACM.

[4] M. Richardson and P. Domingos. Markov logic networks. *Mach. Learn.*, 62(1-2):107–136, Feb. 2006.

[5] J. Vig, S. Sen, and J. Riedl. Tagsplanations: explaining recommendations using tags. In *Proceedings of the 14th international conference on Intelligent user interfaces*, pages 47–56. ACM, 2009.