# Keyword-Based TV Program Recommendation

**Christian Wartena, Wout Slakhorst, Martin Wibbels, Zeno Gantner, Christoph Freudenthaler, Chris Newell, Lars Schmidt-Thieme**
Novay, Enschede, the Netherlands; {christian.wartena,wout.slakhorst,martin.wibbels}@novay.nl
University of Hildesheim, Hildesheim, Germany {gantner,freudenthaler,schmidt-thieme}@ismll.de
BBC R&D, London, UK, chris.newell@rd.bbc.co.uk

## Abstract

Notwithstanding the success of collaborative filtering algorithms for item recommendation there are still situations in which there is a need for content-based recommendation, especially in new-item scenarios, e.g. in streaming broadcasting. Since video content is hard to analyze we use documents describing the videos to compute item similarities. We do not use the descriptions directly, but use their keywords as an intermediate level of representation. We argue that a nearest-neighbor approach relying on unrestricted keywords deserves a special definition of similarity that also takes word similarities into account. We define such a similarity measure as a divergence measure of smoothed keyword distributions. The smoothing is done on the basis of co-occurrence probabilities of the present keywords. Thus co-occurrence similarity of words is also taken into account. We have evaluated keyboard-based recommendations with a dataset collected by the BBC and on a subset of the MovieLens dataset augmented with plot descriptions from IMDB. Our main conclusions are (1) that keyword-based rating predictions can be very effective for some types of items, and (2) that rating predictions are significantly better if we do not only take into account the overlap of keywords between two documents, but also the mutual similarities between keywords.

## 1 Introduction

Notwithstanding the success of collaborative filtering algorithms for item recommendation there is still situations in which there is a need for content-based recommendation, especially in new-item scenarios, e.g. in streaming broadcasting. Since video content is hard to analyze we use context documents to compute item similarities. We do not use the documents directly, but use keywords as an intermediate level of representation. The representation by keywords has the advantage that the two tasks of text analysis and recommendation are clearly separated. Moreover, this offers the possibility to integrate information from different sources, including human classification and allows correction of faulty analyses, which might be important for many organizations.

Content-based recommendation relies on the ability to compute similarities between items based on their content. Classical methods use the overlap of words (either keywords are all words in the documents/descriptions), expressed by a correlation coefficient, like the Jaccard coefficient, or by the cosine similarity, to define the similarity between items. However, two items might have very similar content but use a different vocabulary to describe it. If we restrict the description of an item to a few keywords, the problem will become even more severe. Especially when keywords are not restricted to a set of standardized terms, it might be the case that two items have a considerable overlap in content but are described by completely disjoint sets of keywords. Thus we expect that recommendations could be improved if we are able to include keyword similarities in the definition of item similarities.

We compute similarities between keywords by comparing their co-occurrence distributions. For words in texts it is a well-studied phenomenon that semantic and syntactic similarities can be computed by comparing the contexts in which they appear. Stated in other words: appearing in a similar context is a better indication for similarity than direct co-occurrence. For keywords we expect the same behavior since they are extracted from the (rather short) texts. In each text one synonym of a word is likely to be dominant and selected as a keyword. In other documents different synonyms of the keyword will appear in similar contexts.

Since we can use the same collection of keyword annotated items as we use for recommendation, the keyword-to-keyword similarities can be integrated easily into the item-item similarities. We consider a Markov chain on items and keywords, with transitions from items to keywords, representing the probabilities of terms to be a keyword for a given item and transitions from keywords to items, representing the probabilities for each document to be annotated with a given tag. Now the co-occurrence distribution of a keyword is obtained by a two-step Markov chain evolution starting with a keyword. Keyword similarities are determined by comparing their co-occurrence distributions. Item similarities are obtained by comparing the keyword distribution that arises from a one-step Markov chain evolution. By a three-step evolution starting with a document we incorporate the co-occurrence distributions of the keywords into a kind of smoothed keyword distribution of the item. When these smoothed distributions are compared, the co-occurrence similarity of keywords is included in the item-item similarity.

We have evaluated recommendations based on the keywords with a dataset collected by the BBC and with viewing data from MovieLens combined with plot descriptions from IMDB. For the BBC dataset we have the original editorial synopsis and a collection of related web pages. From both sets of texts we have extracted keywords by two different methods. For all set of keywords in the BBC dataset we see a clear improvement of recommendation results when keyword similarities are included in the computation of item-item similarities. Moreover, we see that keyword-based recommendation gives very good results, comparable or slightly better than those obtained by state-of-the-art collaborative filtering recommenders. Further observations from the experiments with this dataset are that the keywords extracted using a co-occurrence-based technique introduced in [20] give better results than the keywords extracted on the basis of their tf.idf value and that the related websites give rise to better keywords than the original descriptions.

In contrast to the BBC data, for the MovieLens dataset keyword-based recommendation is not able to predict useful ratings at all. This might be explained by the fact that keywords try to define the topic of an item. In a homogeneous database of movies it is likely that topic is not a key factor determining the users appreciation of the movie.

Our main conclusions are that it matters how the keywords are extracted and which texts are used and in the second place that the similarity measure is very important: recommendation results are significantly better if we do not only take into account the overlap of keywords between two documents, but also the mutual similarities between keywords.

## 2 Related Work

### 2.1 Co-occurrence-Based Similarity

The idea that words can be described in terms of the context in which they appear and hence the idea that word similarities can be derived by comparing these contexts has a long tradition in linguistics and is stated e.g. by Zelig Harris [5]. The concept has become known as the distributional hypothesis. Various formalizations of the idea differ considerably in the way a context of a word is defined. Co-occurrence distributions arise from approaches that do not use grammatical structure. Schütze and Pederson [16] suggest that one could construct a vector of co-occurrence probabilities from a complete word co-occurrence matrix, where co-occurrences are counted in a fixed size window. The cosine similarity of these vectors then provides a similarity measure. However, they do not pursue this approach because it was computationally too expensive. The approach that is most similar to the approach we will use is that of Linden and Piitulainen [10], who take all words in any dependency relation to the word under consideration as its context. Then the probability distribution over the words in the context is computed. Finally, the Jensen-Shannon divergence is used to compare these distributions.

This approach is very much the same as the query language models used in pseudo-relevance methods in information retrieval as formulated e.g. by [8] and [21]. In these approaches first, all documents containing the query term are retrieved. Then the average distribution of words in the documents is computed which in this approach is called the query language model. Finally, documents are ranked according to the similarity of the document distribution to the query language model.

### 2.2 Keyword Extraction

Extracting keywords from a text is closely related to ranking words in the text by their relevance for the text. To a first approximation, the best keywords are the most relevant words in the text. Determining the right weight structure for words in a text is a central area of research since the late 1960's ([15]). In 1972 Spärck Jones (reprinted as [17]) proposed a weighting for specificity of a term that has become known as *tf.idf*. This measure is still dominant in determining the relevance of potential keywords for a text. However, keywords are not simply the most specific words of a text and other factors may also play a role in keyword selection. Frank et al. [4] and Turney [19] and subsequently many others have used machine learning approaches to keyword extraction to integrate other features.

The relevance measure used below was introduced by Wartena et al. [20] and it was shown there that this measure gives good results for keyword extraction.

### 2.3 Keyword-Based Recommendation

As noted e.g. by [2] popular collaborative filtering algorithms are not suited for TV program recommendation, as the new-item problem is very prevalent here. For new items content-based recommendation has to be used. In content-based recommendation approaches it is common to base recommendations on the words found in textual descriptions of the items. Here usually tf.idf weights or information gain is used ([12]) to determine the relevance of words. Words with low weights are usually removed, but still a relatively large number of words (100 or more [12]) is used for representation of the text. Furthermore, not all highly relevant words usually can serve as keywords that often are required to be noun phrases. Thus this approach differs significantly from a keyword-based approach.

Recently, there is a considerable interest in using social tags for recommendation. Tags are in many respects similar to keywords, but also have a lot of different characteristics. In most tagged collections the assigners of the tags are the same people that we want to compute recommendations for. Thus most approaches try to capture the tagging behavior of users to improve recommendations. One of the first papers that integrates tag-based similarities in a nearest-neighbors recommender is by Tso-Sutter et al. [18]. Liang et al. [9] also use a nearest-neighbor approach for tag-based recommendation. Most other approaches like the one of Firan et al. [3] build user profiles from tags and base recommendations on these profiles.

## 3 Markov Chains on Items and (Key)words

We use the distributions of terms over items for two different purposes: first we consider the distribution of all terms occurring in the texts to select a few key terms to represent each document. In a second stage we consider the distribution of keywords over items. We have to keep in mind that we talk about different sets of terms in both cases. The concepts and techniques used are however the same.

Consider a set of $n$ term occurrences (e.g. words or multi-words) each being an instance of a term $t$ in $\mathcal{T} = \{t_1, \ldots, t_m\}$, and each occurring in a source document $d$ in a corpus $\mathcal{D} = \{d_1, \ldots, d_M\}$. Let $n(d,t)$ be the number of occurrences of term $t$ in $d$, $n(t) = \sum_d n(d,t)$ be the number of occurrences of term $t$, $N(d) = \sum_t n(d,t)$ the number of term occurrences in $d$ and $n$ the total number of term occurrences in the entire collection.

We define three (conditional) probability distributions

$$q(t) = \frac{n(t)}{n} \qquad \text{on } \mathcal{T} \tag{1}$$

$$Q(d|t) = \frac{n(d,t)}{n(t)} \qquad \text{on } \mathcal{D} \tag{2}$$

$$q(t|d) = \frac{n(d,t)}{N(d)} \qquad \text{on } \mathcal{T}. \tag{3}$$

Probability distributions on $\mathcal{D}$ and $\mathcal{T}$ will be denoted by $P$, $p$ with various sub- and superscripts.

Consider a Markov chain on $\mathcal{T} \cup \mathcal{D}$ having transitions $\mathcal{T} \to \mathcal{D}$ with transition probabilities $Q(d|t)$ and transitions $\mathcal{D} \to \mathcal{T}$ with transition probabilities $q(t|d)$ only. Given a term distribution $p(t)$ we compute the one-step Markov chain evolution. This gives us a document distribution $P_p(d)$:

$$P_p(d) = \sum_t Q(d|t)p(t). \tag{4}$$

Likewise given a document distribution $P(d)$, the one-step Markov chain evolution yields the term distribution

$$p_P(t) = \sum_d q(t|d)P(d). \tag{5}$$

Since $P(d)$ gives the probability to find a term occurrence in document $d$, $p_P$ is the weighted average of the term distributions in the documents. Combining these, i.e. running the Markov chain twice, every term distribution gives rise to a new term distribution

$$\bar{p}(t) = p_{P_p}(t) = \sum_{t',d} q(t|d)Q(d|t')p(t'). \tag{6}$$

For some term $z$, starting from the degenerate term distribution $p_z(t) = \delta_{tz}$ (1 if $t = z$ and 0 otherwise), we get the *distribution of co-occurring terms* or *co-occurrence distribution* $\bar{p}_z$

$$\bar{p}_z(t) = \sum_{d,t'} q(t|d)Q(d|t')p_z(t') = \sum_d q(t|d)Q(d|z). \tag{7}$$

This distribution is the weighted average of the term distributions of documents containing $z$ where the weight is the probability $Q(d|z)$ that an instance of term $z$ has source $d$. If we compute term similarities by comparing their co-occurrence distribution – rather than the source distributions $Q(d|z)$ – we base the similarity on the context in which a word occurs as intended in the distributional hypothesis.

Likewise we obtain a term distribution if we run a Markov chain three times starting from the degenerated document distribution $P_d(i)\delta_{id}$:

$$\bar{p}_d(t) = p_{P_{p_{P_d}}}(t) = \sum_{d',t',d''} q(t|d')Q(d'|t')q(t'|d'')P(d''|d) \tag{8}$$

$$= \sum_{d',t'} q(t|d')Q(d'|t')q(t'|d) = \sum_z q(z|d)\bar{p}_z(t). \tag{9}$$

The distribution $\bar{P}_d$ can be seen as a smoothed version of the document distribution $P_d$ in which co-occurrence information of the words is integrated. Thus, if we compare documents using these smoothed distributions we also take into account co-occurrence-based word similarities.

## 4 Keyword Extraction

For all items in our datasets a short textual description is available. We extract words from these texts to represent them as a vector in a word space. We can either use all words (after removing stop words) or only a small selection.

For keyword extraction we compare two different extraction methods. Both methods are based on ranking words and selecting the $k$ top-ranked words. The first method uses standard tf.idf ranking. The tf.idf value of a term $t$ in a document $d$ is defined as

$$tf.idf(t,d) = \frac{n(d,t)}{\log df(t)}, \tag{10}$$

where $n(d,t)$ is the number of occurrences of $w$ in $d$, and $df$ is the number of documents $d'$ for which $n(d',t) > 0$.

The second method uses the hypothesis that the co-occurrence distribution of a good keyword is a good estimator of the term distribution of the document. Thus the suitability of a word as a keyword can be predicted by comparing the co-occurrence distribution of the word and the term distribution. There are various options to compute the similarity between two distributions. In [20] it was shown that the following correlation coefficient gives the best results:

$$r(z,d) = \frac{\sum_t (\bar{P}_d(t) - q(t))(\bar{p}_z(t) - q(t))}{\sqrt{\sum_t (\bar{P}_d(t) - q(t))^2}\sqrt{\sum_t (\bar{p}_z(t) - q(t))^2}}. \tag{11}$$

This coefficient captures the idea that two distributions are similar if they diverge in the same way from the background distribution $q$. The coefficient is in fact the cosine of the residual co-occurrence distribution of the term and the smoothed term distribution of the document after subtracting the background term distribution. Note that the "residual" probabilities can be negative and hence $r(z,d)$ also can become negative. For keyword extraction we will not only use the coefficient for ranking, but we will also require that the correlation coefficient defined in equation 11 is positive.

The different keyword extraction strategies are implemented in a UIMA[1] text analysis pipeline. All words in the text are stemmed using the tagger/lemmatizer from [6] and annotated by the Stanford part of speech tagger ([1]). To compute co-occurrence distributions all open class words are taken into account.

## 5 Keyword-Based Recommendation

The recommendation strategy we use is a straightforward $k$-nearest-neighbor approach for recommendation ([13]). Content-based $k$-nearest-neighbor approaches are similar to classical collaborative filtering algorithms, but the similarity measure between items is based on the content of the items and not on the ratings. The rating we predict for a user and an item is the weighted average of all items rated by the user, where more similar items get greater weights. To be precise, let $I_u$ be the set of all items rated by user $u$, then the predicted rating $R(u,i)$ of $u$ for item $i$ is defined by

$$R(u,i) = \frac{\Sigma_{j \in I_u} \mathrm{sim}(i,j) R(u,j)}{\Sigma_{j \in I_u} \mathrm{sim}(i,j)}. \tag{12}$$

We use two different keyword based similarity measures for items. The first measure is the Jaccard coefficient:

$$\mathrm{sim}(i,j) = \alpha + \frac{|K_i \cap K_j|}{|K_i \cup K_j|}, \tag{13}$$

where $K_i$ is the set of keywords of item i. The additional parameter $\alpha$ ensures that each item is taken into account, even if the set of keywords is disjoint from the item for which a rating has to be predicted. Thus, items which do not overlap with any other items rated by the user get the user average as the prediction. If a very large value is taken for $\alpha$, the predicted rating will always be the user average. Some initial experiments suggest that a value of about 0.1 yields the best results.

[1]http://incubator.apache.org/uima/

Table 1: Characteristics of the two datasets

| | BBC | MovieLens subset |
|---|---|---|
| users | 84 581 | 4 805 |
| items | 2 487 | 704 |
| ratings | 130 262 | 361 961 |

Since all keywords are drawn from an unrestricted vocabulary it might be the case that two texts are tagged with similar or strongly related words but not with exactly the same words. Thus we should not only check whether the same keywords are used, but also how strongly the keywords are related. As argued before, this can be done by comparing co-occurrence distributions: the co-occurrence distribution can be seen as a proxy for the semantics of a word. The whole text now has to be represented by the average of all co-occurrence distributions of all its keywords. This new distribution is in fact a smoothed version of the original keyword distribution of the document. The similarity between two items $i$ and $j$ is now given by

$$\mathrm{sim}(i,j) = \alpha + 1 - \mathrm{JSD}(\bar{p}_i \| \bar{p}_j). \tag{14}$$

Again we use $\alpha = 0.1$, and JSD is the Jensen-Shannon divergence.

## 6 Evaluation

### 6.1 Data Sets

#### BBC Broadcast Data

As a first dataset to test our hypothesis that kNN-based rating prediction will benefit from including co-occurrence into the computation of item similarity was collected in a user study at the BBC. BBC programming provides a very interesting use case for keyword based recommendation. Since the BBC does not have a static database of items, like the movie databases on which much of the research on recommendation was done, but a stream of items. Here in fact each item that we want to predict ratings for is a new item. Content-based recommendation might be very useful in this situation. For all items an editorial description and one or more web pages are available.

The BBC data was collected during field trials of the MyMedia project[2] concerning recommender systems. An audience research panel was asked to rate all content items they watched during the field trial. In parallel, media server logs were analyzed to determine the viewing behavior of a larger superset of users. The characteristics of the dataset are described in Table 1.

[2]http://www.mymediaproject.org

Table 2: Number of unique keywords and average keywords per item

| | Unique KWs | | KWs per item | |
|---|---|---|---|---|
| | tf.idf | co-occ. | tf.idf | co-occ. |
| BBC original descr. | 7 136 | 5 631 | 9.39 | 8.44 |
| BBC web descr. | 3 950 | 2 770 | 10.00 | 9.98 |
| IMDB plots | 6 651 | 4 827 | 10.00 | 10.00 |
| IMDB keywords | 14 177 | | 73.23 | |

Every content item in the BBC dataset has a related web page or website. This meant that two descriptions were available for each item:

1. Original editorial descriptions typically 30 to 200 words in length.
2. Website text typically 200 to 4000 words in length.

The website text was obtained automatically using some knowledge about the rough HTML structure of the web sites. Note that some content items have very brief descriptions and a simple, single web page associated with them whereas other items have longer descriptions and a substantial website. Where items were part of an ongoing series the web site frequently includes information about the complete series, rather than information about an individual episode.

We have extracted keywords from all texts by stemming and the two weighting schemes discussed above. Since we only extract nouns and verbs as keywords and we also exclude person names, as far as properly identified, less than ten keywords were found for a number of items. For all texts that are long enough 10 keywords were extracted. When extracting keywords using the correlation defined in 11 we also restrict the set of possible keywords to those term that have a positive correlation. Thus the number of keywords extracted here sometimes is lower than 10 even if 10 nouns are present in the text. The average number of keywords assigned and the total number of unique keywords used are given in Table 2.

**MovieLens Dataset**

The second dataset we have used is derived from the 10 Million rating dataset from MovieLens ([11]). We have augmented this dataset with the plot descriptions of the movies from IMDB ([7]). For a lot of movies the available plots are very short and uninformative. Thus we restricted the dataset to the movies having plots of at least 200 words. The characteristics of the dataset are described in Table 1. The number of keywords per item and the total number of unique keywords are given in Table 2.

As compared to the BBC dataset we see that the dataset is much denser: the number of users and items is smaller whereas there are many more ratings.

## 6.2 Experimental Setup

The goal of the experiment is twofold. First we want to know whether extracted keywords provide a viable resource on which to base recommendations. In the second place we want to test whether the similarity measure defined in (14) gives better rating predictions than the Jaccard coefficient (13). To test the latter hypothesis for each set of keywords we compute predictions using both measures. In order to test the first hypothesis we compare the keyword-based rating predictions to predictions from other algorithms. We use the following baselines:

1. user average,
2. item average,
3. collaborative filtering, and
4. genre- and series-based prediction.

Item average (i.e. for a user-item pair we predict the average rating other users have assigned to that item) provides a nice baseline in the experiment but is not an alternative to content-based recommendations in real scenarios, since it cannot be applied for new items. User average (i.e. for a item user pair we predict the average rating the user has given to other items) also is a good baseline but not useful in real life since it does not help a user to make any choices. Collaborative filtering provides a very strong baseline and is some sense gives the limit we want to reach. However, it is only applicable in the static experiment and not in the streaming broadcast scenario as discussed above. For collaborative filtering we have used a state-of-the-art matrix factorization implementation.[3] For the genre-based recommendation we use the same algorithm as for the keyword-based recommendation. To do so we simply treat the genre labels as keywords. In the experiment with the BBC dataset there are a lot of series. We expect that series-based recommendation might give very good results, since it is likely that someone who likes some episodes of a series will also like the remaining episodes. Series can easily be identified, since in almost all cases all items of a series have the same title. By using the title of each item as a keyword we get a series-based recommender. Since we use $\alpha = 1$ for all items that do not belong to a series already rated by the user we predict the user average. Given the good results of genre-based recommendation in earlier experiments we also use genres and the combination of genres and title for content-based recommendation.

For evaluation we have done a leave-one-out experiment: each rating is predicted using all ratings except the one that has to be predicted. Since the recommender does not need any training of a model (except the co-occurrence distributions of the keywords) this is a very feasible approach. For the collaborative filtering we use a different protocol, since for each split a new model has to be trained. The result given here is obtained using a 10-fold cross-validation. Interpreting the results requires

[3] Biased matrix factorization from the MyMediaLite package: `http://ismll.de/mymedialite` [14]

Table 3: Results of content-based recommendation on BBC dataset

| Data | Distance | RMSE |
|---|---|---|
| web – tf.idf | Jaccard | 0.302 |
| web – co-occ | Jaccard | 0.290 |
| original – tf.idf | Jaccard | 0.335 |
| original – co-occ | Jaccard | 0.329 |
| genres | Jaccard | 0.312 |
| title | Jaccard | 0.287 |
| genres + title | Jaccard | 0.293 |
| web – tf.idf | JSD | 0.285 |
| web – co-occ | JSD | 0.283 |
| original – tf.idf | JSD | 0.332 |
| original – co-occ | JSD | 0.312 |
| genres | JSD | 0.339 |
| genres + title | JSD | 0.285 |
| user average | | 0.348 |
| item average | | 0.464 |
| MF | | 0.291 |

Table 4: Results of content-based recommendation on MovieLens/IMDB dataset

| Data | Distance | RMSE |
|---|---|---|
| plot - tf.idf | Jaccard | 0.414 |
| plot - co-occ | Jaccard | 0.412 |
| original keywords | Jaccard | 0.409 |
| genres | Jaccard | 0.406 |
| plot - tf.idf | JSD | 0.414 |
| plot - co-occ | JSD | 0.413 |
| original keywords | JSD | 0.413 |
| genres | JSD | 0.410 |
| user average | | 0.415 |

some caution because the matrix factorization models were trained using roughly 10 % smaller datasets.

## 6.3 Results

As it is common for rating prediction, we use the root mean square error (RMSE) as evaluation measure. The results in terms of RMSE are given in Table 3 and Table 4 for the BBC and MovieLens datasets, respectively.

The first remarkable fact is that keyword-based rating prediction gives very good results on the BBC dataset but cannot improve on the item average baseline in the case of the MovieLens/IMDB data. This result is not very surprising. Keywords mainly give the topic of the program or the movie plot. Whether someone likes a movie might depend on the genre, the director, the actors, etc. but probably not on the topic of the plot. Nevertheless we see that keyword-based recommendation indeed can be very useful since it clearly outperforms simple baselines like user or item average. As expected the series (title) and genre-based recommenders perform very well. However, the best keyword-based recommenders perform equally well. Surprisingly, the content-based recommenders perform equal well as the matrix factorization. The conclusion for our first hypothesis therefore is that keyword-based recommendation can be very useful for a dataset in which the topic of the item matters and for which no other suitable meta-data, such as genre or series information is available.

With regard to our second question, whether the inclusion of keyword co-occurrence information in the definition of item similarity is useful, we see that in almost all cases our new distance measure gives better results than the standard measure. Only the genre-based results are poorer. We have however to say that the measure was not intended for use with such clearly defined concepts such as genres. It should solve problems with (near) synonyms in a set of freely selected keywords.

Furthermore we observe that the co-occurrence-based keywords perform better than tf.idf-based keywords. Thus the results also provide more evidence to support the conclusions of a comparison between the two methods in previous work ([20]). Finally, we see that the keywords extracted from the related material perform better than the keywords extracted from the original descriptions. When we look into more detail, on the contrary one gets the impression that the keywords extracted from the original descriptions contain less mistakes and noise. However, the main effect seems to be, that there are a lot of items for which the original descriptions are too short and give too few keywords.

## 7 Conclusion

In this paper we have investigated keyword-based rating prediction. Keywords constitute a useful level of description of an item since keywords can be assigned by humans or extracted automatically from one or more texts. We have shown that for some datasets keyword-based rating predictions give very good results, comparable to state-of-the art collaborative filtering methods. We have hypothesized that the reason lies in the nature of the dataset and the relevance of the topic of the item for the appreciation of the item. It remains a question for future research to apply keyword-based rating prediction to more datasets to verify this hypothesis.

We have argued that a nearest-neighbor approach relying on unrestricted keywords deserves a special definition of nearness taking word similarities also into account. We have defined such a similarity measure as a divergence measure of smoothed keyword distributions where the smoothing is done on the basis of the co-occurrence probabilities of the keywords. In the experiments we see that for various sets of keywords this measure always gives better results than the Jaccard coefficient.

Other findings are that the keywords extracted from the related web pages lead to better recommendation results than the keywords extracted from the original abstracts. The main reason seems to be that the abstracts are in many cases too short to extract an optimal number of relevant keywords. Finally we see that the keywords obtained by comparison of co-occurrence distributions lead to better recommendation results than the keywords extracted using a standard tf.idf relevance measure.

## 8 Acknowledgments

This work was funded by the European Commission FP7 project MyMedia[4] under the grant agreement no. 215006. We thank the anonymous reviewers for their valuable feedback.

## References

[1] Stanford part of speech tagger. http://nlp.stanford.edu/software/tagger.shtml.

[2] P. Cotter and B. Smyth. Ptv: Intelligent personalised tv guides. In *AAAI/IAAI*, pages 957–964. AAAI Press / The MIT Press, 2000.

[3] C. S. Firan, W. Nejdl, and R. Paiu. The benefit of using tag-based profiles. In V. A. F. Almeida and R. A. Baeza-Yates, editors, *LA-WEB*, pages 32–41. IEEE Computer Society, 2007.

[4] E. Frank, G. W. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning. Domain-specific keyphrase extraction. In T. Dean, editor, *IJCAI*, pages 668–673. Morgan Kaufmann, 1999.

[5] Z. Harris. Distributional structure. *Word*, 10(23):146–162, 1954.

[6] M. Hepple. Independence and commitment: Assumptions for rapid training and execution of rule-based pos taggers. In *ACL*, 2000.

[7] http://www.imdb.com.

[8] J. D. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In W. B. Croft, D. J. Harper, D. H. Kraft, and J. Zobel, editors, *SIGIR*, pages 111–119. ACM, 2001.

[9] H. Liang, Y. Xu, Y. Li, R. Nayak, and L.-T. Weng. Personalized recommender systems integrating social tags and item taxonomy. In *Web Intelligence*, pages 540–547. IEEE, 2009.

[10] K. Lindén and J. Piitulainen. Discovering synonyms and other related words. *CompuTerm 2004*, pages 63–70, 2004.

[11] http://www.grouplens.org/system/files/-README_10M100K.html.

[12] M. J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13(5-6):393–408, 1999.

[13] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The Adaptive Web: Methods and strategies of web personalization. Volume 4321 oF Lecture Notes in Computer Science*, pages 325–341. Springer-Verlag, 2007.

[14] S. Rendle and L. Schmidt-Thieme. Online-updating regularized kernel matrix factorization models for large-scale recommender systems. In *RecSys '08: Proceedings of the 2008 ACM Conference on Recommender Systems*. ACM, 2008.

[15] G. Salton and C. Buckley. Term weighting approaches in automatic text retrieval. Technical report, Cornell University, 1987.

[16] H. Schütze and J. Pederson. A cooccurrence-based thesaurus and two applications to information retrieval. In *Proceedings of RIA Conference*, pages 266–274, 1994.

[17] K. Spärck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 60:493–502, 2004.

[18] K. H. L. Tso-Sutter, L. Balby Marinho, and L. Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In R. L. Wainwright and H. Haddad, editors, *SAC*, pages 1995–1999. ACM, 2008.

[19] P. D. Turney. Learning algorithms for keyphrase extraction. *Inf. Retr.*, 2(4):303–336, 2000.

[20] C. Wartena, R. Brussee, and W. Slakhorst. Keyword extraction using word co-occurrence. In *DEXA Workshops*, pages 54–58. IEEE Computer Society, 2010.

[21] C. Zhai and J. D. Lafferty. Model-based feedback in the language modeling approach to information retrieval. In *CIKM*, pages 403–410. ACM, 2001.

[4] http://www.mymediaproject.org/