Prediction of User Opinion for Products
A Bag-of-Words and Collaborative Filtering based Approach

Esteban García-Cuesta[1], Daniel Gómez-Vergel[1], Luis Gracia-Expósito[1] and María Vela-Pérez[2]
[1]*Computer Science Department, Universidad Europea de Madrid, Calle Tajo S/N Villaviciosa de Odón 28670, Madrid, Spain*
[2]*Departamento de Estadística e Investigación Operativa II, Facultad de Ciencias Económicas y Empresariales, Universidad Complutense de Madrid, Campus de Somosaguas, Madrid, Spain*
{*esteban.garcia, daniel.gomez, luis.gracia*}*@universidadeuropea.es, mvelaper@ucm.es*

Keywords: User Opinion, Recommendation Systems, User Modeling, Prediction, Hyper-personalization.

Abstract: The rapid proliferation of social network services (SNS) gives people the opportunity to express their thoughts, opinions, and tastes on a wide variety of subjects such as movies or commercial items. Most item shopping websites currently provide SNS systems to collect users' opinions, including rating and text reviews. In this context, user modeling and hyper-personalization of contents reduce information overload and improve both the efficiency of the marketing process and the user's overall satisfaction. As is well known, users' behavior is usually subject to sparsity and their preferences remain hidden in a latent subspace. A majority of recommendation systems focus on ranking the items by describing this subspace appropriately but neglect to properly justify why they should be recommended based on the user's opinion. In this paper, we intend to extract the intrinsic opinion subspace from users' text reviews –by means of collaborative filtering techniques– in order to capture their tastes and predict their future opinions on items not yet reviewed. We will show how users' reviews can be predicted by using a set of words related to their opinions.

1 INTRODUCTION

The advent of the Internet and its social websites have made it possible for people to express their opinions with great ease. This is particularly true in e-commerce web sites –*e.g.*, Amazon– where users may read published opinions to gather a first impression on an item before purchasing it. This information may also be used to design better marketing strategies, to hyper-personalize the website, and to improve the user's experience. Recall that by hyper-personalization we doesn't only mean the process of adaptation to the user's needs and their characteristics but also to provide some insights about it.

In this sense, recommender systems have truly transformed the way users interact and discover products on the web. Whenever a user assesses any type of product there exists the need to model how the assessment is done to be able to recommend new products they may be interested in (McAuley and Leskovec, 2013a), or to identify users of similar taste (Sharma and Cosley., 2013). To model users and the way they evaluate and review products it becomes necessary to unveil the latent structure of their opinions. In (McAuley and Leskovec, 2013b), for instance, the authors present a hidden factor model to understand why any two users may agree when reviewing a movie yet disagree when reviewing another: The fact that users may have similar preferences towards one genre, but opposite preferences for another turns out to be of primary importance in this context.

Incorporating the latent factors associated with users is, therefore, a fundamental step in any recommendation system (Bennet and Lanning, 2007). Typically, these systems use plain-text reviews and/or numerical scores, along with machine learning algorithms, to predict the scores that users will give to items that remain still unreviewed (Y. Koren and Volinsky, 2009). In (McAuley and Leskovec, 2013b) authors also propose the use of these latent factors not for prediction, but to achieve a better understanding of the rating dimensions –to be connected to the intrinsic features of users and their likes–, hence improving the user modeling process.

Our starting hypothesis is that by assuming the existence of a latent space that accurately represents the users' interests and tastes (see (McAuley and Leskovec, 2013b)) we may be able to predict their opinions/reviews. Rather than using the latent space to predict ratings, we intend therefore to predict the

García-Cuesta, E., Gómez-Vergel, D., Expósito, L. and Vela-Pérez, M.
Prediction of User Opinion for Products - A Bag-of-Words and Collaborative Filtering based Approach.
DOI: 10.5220/0006209602330238
In *Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods (ICPRAM 2017)*, pages 233-238
ISBN: 978-989-758-222-6

sets of words that users will choose to express their opinions on not previously reviewed items. The approach is based on two stages. The first one sets up the opinion dictionary, not too large as to impede numerical computations, but rich enough as to characterize the users' opinions in this product reviewing context. Take the words 'expensive' and 'good_quality' for example,[1] the former being a purely subjective term which expresses a negative opinion about a product, the latter expressing a positive opinion instead. We would like these terms to be part of the dictionary since they convey relevant information on the user's opinion. We want to stress the fact, however, that this article does not address a sentiment analysis problem, that is, we do not try to find out whether the item's review is positive or negative. The second stage predicts the set of words a user would choose had they had the opportunity to review an item, based on the hidden dimensions that represent their tastes.

1.1 Contributions

Our main contribution is to propose and describe – for the very first time, to the best of the authors' knowledge– a model that combines the use of hidden dimensions (associated with users' tastes and product features) and a matrix factorization approach to predict the user's opinion on not reviewed items. The results show that the prediction of the set of words which best describes a review is possible and gives, at this early stage of development, an initial understanding of the main reasons why a user would like or dislike a product. This is important since this information can be used to complement the rating's value and provide extra information to the user whenever a new product is recommended. Thus, this approach can be used together with the current recommendation systems to provide further insight into the reasons why the product is recommended to a specific user, knowing that the very same product can be recommended to another user for completely different reasons. We applied this approach to the Amazon musical instrument dataset (J. McAuley and Leskovec, 2015), which contains a total of $85,405$ reviews for 1429 users and 900 products. We chose this dataset due to its ease of interpretability and reasonable size (notice that, since we insert a dictionary vector of size D into the matrix for each review, the overall size of the dataset increases by a factor 10^2).[2] The rest of the paper is organized as follows: Section 2 contains the state-of-the-art on recommendation systems based on user modeling and collaborative filtering approaches and explains the similarities and differences with our proposal. Section 3 describes the experiments we conducted to test the implemented model. We show our results in Section 4 and discuss the model's strengths and weaknesses. Finally, Section 5 presents the conclusions and some insights into future work.

2 USER MODELING BASED ON OPINIONS

In what follows, we introduce some terminology and the formal notation we use throughout the paper, as well as a brief review on the traditional user modeling approaches to recommendation. We then proceed to explain in full detail our new opinion prediction model based on tensor factorization.

2.1 Notation

A typical online shopping website with SNS capabilities provides, for the purposes of this article, N reviewers $A = \{u_1, \dots, u_N\}$ writing reviews on a set of M items $P = \{p_1, \dots, p_M\}$. Generally, a given user will have only scored and reviewed a subset of these M items, thus making the website's ranking matrix sparse. Let $S \subseteq A \times P$ denote the set of user-item pairs (u,i) for which a written review exists and let $\mathbf{t}_{ui}$ be the associated feature vector that represents the text contents of the u-th user's review on the i-th item in the dataset. More specifically, we use the bag-of-words features extracted from the text reviews to represent the user's review. The review bag-of-words vector is then defined by $\mathbf{t}_{ui} \in \mathbb{R}^D$, where D is the word vocabulary size. We will incorporate these reviews into a fundamental model that predicts users' opinions (*i.e.*, $\mathbf{t}_{ui}$ vectors) on items not included in S.

2.2 Basic Related Work on Recommendation Systems

Most existing recommendation systems fit into one of the following two categories: i) content based recommendation or ii) collaborative filtering (CF) systems.

The first approach addresses the recommendation problem by defining a user profile model U that repre-

[1] We work with concepts provided by the natural language analysis tool, so terms can be compositions of several words.

[2] We use 20 executors with 8 cores and 16GB RAM on a Hadoop cluster with a total of 695GB RAM, 336 cores, and 2TB HDFS. Our implemented algorithms are easily scalable, so any RAM limitation might be solved using a cluster with a sufficiently large number of nodes.

Table 1: Notation.

Symbol	Description
$\mathbf{t}_{ui}$	u-th user's review ('document') on i-th item
K	number of latent dimensions
D	number of words in the dictionary
t_{uij}	j-th word of the $\mathbf{t}_{ui}$ vector
f_{uij}	frequency of occurrence of the t_{uij} word
N_t	number of words in the text t
S	all (u,i) pairs of existing reviews
u	user
i	item
$\mathbf{R}$	text reviews input matrix in $\mathbb{R}^{N\times(M\times D)}$

sents all the information available on a user. In a basic problem setup, U includes the users' preferences for a set of items, later used to describe the users' likes and dislikes. One of the main drawbacks of this basic approach is the fact that it ignores users' opinions on different elements, taking only their preferences into consideration. Previous studies (W. Zhang and Li, 2010) proposed the use of sentiment analysis to find out the set of words that positively describes user preferences to be able to predict the sentiment value. This proposal was later extended in (Chen and Wang, 2013)(L. Fangtao and Zhu, 2011) to enhance the user profile's description by using linear combinations of the initial set or a subset of words. Both articles rely on a user profile which is built *a priori* and used later on to predict the recommendations. This methodology, however, does not attempt to reflect the intrinsic likes and dislikes of users on different items, focusing on a more general description of their preferences instead.

One way of resolving this limitation consists in including the text features of the user's reviews (more specifically, frequencies of occurrence of words) into the model. In (L. Fangtao and Zhu, 2011), the authors incorporate reviews, items, and text features into a three-dimensional tensor description to reveal the different sentiment effects that arise when the same word is used by different users in ranking different items. The authors showed an improvement on the ranking prediction when compared to previous models. In a similar fashion, (M. Terzi and Whittle, 2011) presents an extension of the user-kNN algorithm that measures the similarity between users in terms of the similarities between text reviews –instead of using numerical ratings only–, applying a collaborative filtering model to predict the ranking recommendations. The authors claim that their model outperforms the conventional algorithms that only use the ratings as inputs.

Despite the successful achievements of this last two proposals, notice that none of them attempt to predict the user's opinion and reveal the intrinsic features behind an item's recommendation.

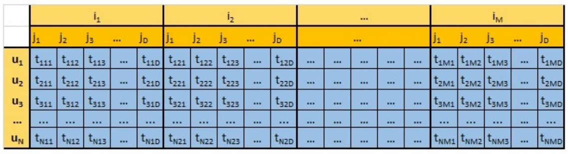

Figure 1: 2D representation of the input matrix **R**.

2.3 Explaining Recommendations by Predicting the Opinion

Next, we discuss how a careful interpretation of the prediction about a user's review can justify the recommendation, helping us achieve a better understanding of the reasons why the user may like/dislike the product. Table 2 shows two Amazon reviews to explain how this interpretation is done. The first column contains the original reviews and the second one includes the predicted sets of words obtained with our model. We can tell at a glance that the first reviewer complains about a bad headphones' ergonomic design, something which is reasonably predicted in the second column by the words 'bass' and 'vibration'. In the second example the review praises the good performance of the earphones, a fact that is predicted by the words 'purchase', 'happy', and 'better'.

Table 2: Example of Amazon review parts for a drum product and concepts associated in our dictionary.

Review Text	**Prediction**
The bass drum vibrates.	bass; vibration
Be happy if you purchase this, because you will never find a better deal for this.	purchase; happy; better

Our model makes use of a collaborative filtering model reviewed in the next section.

2.3.1 Collaborative Filtering

A rich collection of algorithms and recommender systems has been developed over the last two decades. The wide range of domains and applications shows that there is not a one-size-fits-all solution to the recommendation problem and that a careful analysis of prospective users and their goals is necessary to achieve good results.

Collaborative filtering, in particular, is a technique that generates automatic predictions for a user by collecting taste information from other people. The information domain for these systems consists of users who already expressed their preferences for various items, represented by $(user, item, rating)$ triples. The rating is typically a natural number between zero and five or a two-valued like/dislike variable. Usually, the associated rating matrix is subject to sparsity due to the existence of unrated items. The full evaluation process often requires the completion of two tasks: (i) predicting the unknown ratings and (ii) providing the best ranked list of n items for a given user (M. D. Ekstrand and Konstan, 2012).

2.3.2 Predicting the Opinion using Alternating Least Squares (ALS)

Our information domain consists of triples of the form $(u, i, \mathbf{t}_{ui})$, where u is a natural number that labels a user, i labels an item, and $\mathbf{t}_{ui}$ is the corresponding review vector (possibly empty). Let $\mathbf{R} \in \mathbb{R}^{N\times(M\times D)}$ be the 2-dimensional input matrix (typically subject to sparsity) with entries $f_{uij} \geq 0$ only for pairs $(u,i) \in S$. Here, f_{uij} denotes the frequency of occurence (if any) of the j-th word in the u-th user's review for the i-th item in the dataset. If we let $\mathbf{R}_i$ denote the $(N \times D)$-matrix containing all reviews for the i-th product, then $\mathbf{R} = [\mathbf{R}_0\ \mathbf{R}_1\ \cdots\ \mathbf{R}_M]$ is set up by concatenating all $\mathbf{R}_i$ matrices. This $\mathbf{R}$ matrix represents a high dimensional space where the users' opinions (either positive or negative) are latent and can be represented by a subset of new features in a lower dimensional space.

Matrix $\mathbf{R}$ can then be subjected to an ALS factorization (Y. Koren and Volinsky, 2009) of the form $\mathbf{R} \approx \mathbf{P}\mathbf{Q}^{\mathrm{T}}$ in order to estimate the missing reviews. Here, $\mathbf{P} \in \mathbb{R}^{N\times K}$ and $\mathbf{Q} \in \mathbb{R}^{(M\times D)\times K}$, where $K \in \mathbb{N}$ is the number of latent factors or features –in our model, a predefined constant typically in the range $2 \leq K \leq 10$. Any frequency f_{uij} can then be approximated by the usual scalar product $\hat{f}_{uij} = \mathbf{p}_u^{\mathrm{T}}\mathbf{q}_{ij}$, with $\mathbf{p}_u \in \mathbb{R}^{K\times 1}$ the u-th row of $\mathbf{P}$ and $\mathbf{q}_{ij} \in \mathbb{R}^{K\times 1}$ the $(iD+j)$-th row of $\mathbf{Q}$.

3 EXPERIMENTS

We test our model using the musical instruments Amazon dataset for experimentation, which contains user-product-rating-review quads for a total of 85,405 reviews for 1429 users and 900 products (J. McAuley and Leskovec, 2015).[3]

At a first step, we process the reviews using a natural language processing API graciously provided to us by Bitext corportation,[4] making all the basic tokenization, lemmatization, PoS (R. Benjamins and Gomez, 2014), and concept identification tasks straightforward. This enables us to syntactically analyze the texts in an efficient manner in order to extract the simple (*e.g.*, 'cheap') and compound (*e.g.*, 'digital_products') concepts to be part of the dictionary, including concepts related to sentiment.

[3] http://jmcauley.ucsd.edu/data/amazon/

[4] https://www.bitext.com

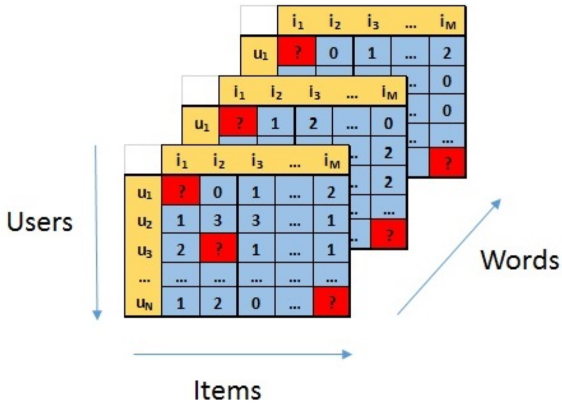

Figure 2: 3D matrix representation of the 'missing' values $\mathbf{R}_{test}$ to be predicted and used for validation.

At this stage, we keep track of the (usually different) sets of words used by each customer in their product reviews, along with their frequencies of occurrence. The final version of the dictionary -from now on referred to as *global*- is obtained by taking the union of the individual users' lists of words. To only retain the most relevant concepts and keep the size of the dictionary manageable for subsequent computations, we impose a minimum global frequency of occurrence on any given word to be included into the list ($f = 100$ in the first experimentation round and $f = 50$ in the second, collecting 116 and 256 concepts respectively). Table 3 shows a subset of the concepts that best represent the users' opinions after applying the aforementioned frequency filtering.

Table 3: Some of the concepts included in the opinion dictionary for the minimum frequency of occurrence of 50.

'good', 'music', 'inexpensive', 'standard', 'amazing', 'easy', 'good quality', 'boom', 'quick', 'cheap', 'decent', '5 star', 'durable', 'highly recommend', 'accurate', 'pretty good', 'would recommend', 'no problem', 'gig gag', 'best', 'good price', 'much better', 'amazing', 'very pleased', 'yes', 'very good', 'fun', 'great', 'standard', 'durable', 'recommend', 'strong', 'very happy', 'keep', 'love', 'really nice', 'not bad', 'sturdy', 'good product', 'handy', 'try','quality', ''can not', 'cool', 'comfortable', soft', 'excellent', 'much better', 'fantastic', 'quick', 'hard', 'low', 'great value', 'break', 'price', 'big', 'wow', 'great product', 'try','worth', 'problem', 'wrong', 'distortion', 'simple', 'soft', or 'inexpensive'.

Next, we randomly remove 30% of the word lists —that is, $\mathbf{t}_{ui}$ reviews — from the 2-dimensional $\mathbf{R}$ matrix to validate our model, giving raise to a train set $\mathbf{R}_{train}$ and a test set $\mathbf{R}_{test}$. The way to achieve this is straightforward: We pick 30% of all user-item coordinates $(u,i) \in S$ at random –a total of 3073 pairs– and replace their corresponding word lists with empty vectors. The resulting sparse matrix $\mathbf{R}_{train}$ is finally subjected to ALS factorization in order to reconstruct the removed vectors and compare them with the originals.

All our codes are implemented in Python 3.5 using the collaborative filtering RDD-based Apache Spark implementation of the ALS algorithm, [5] which is well known for its robustness and efficiency. This implementation, in turn, makes use of the MLlib library. [6]

4 RESULTS

We use the L2 norm to evaluate the training performance and the overall quality of the predictions, as shown in table 4.

Table 4: Results obtained for the different experiments.

#Concepts	K	Exec.Time (Min.)	Avg./Std. Jaccard	Avg./Std. L2 norm
116	5	16	0.17/0.14	2.01/1.17
256	5	33	0.12/0.10	2.20/1.20
116	7	16	0.11/0.11	2.25/1.23
256	7	33	0.12/0.11	2.25/1.24
116	10	16	0.16/0.13	2.01/1.19
256	10	33	0.11/0.11	2.23/1.25

We also use the Jaccard distance to evaluate whether a word appears or not in the prediction in the following sense: If the test set $\mathbf{R}_{test}$ contains a positive frequency for a t_{uij} word and this word also appears in the train set $\mathbf{R}_{train}$, then the Jaccard distance $t_{test,uij} - t_{train,uij}$ is zero; otherwise it is 1. No significant differences are observed in the results when varying the number of latent factors from 5 to 10. The number of concepts (the word vocabulary size) does not seem to significantly alter the results either, although the L2 distance values suggest that a smaller number of concepts yields a smaller error. The reason for this behavior can be found in the fact that, if the word vocabulary size remains small the selected concepts are really the most frequently used, and hence, they generate more easily recognizable patterns.

Figure 3 shows that the Jaccard distance tends to concentrate more around the mean and its lower values (exponential decay) than the Euclidean distance, a fact that may find explanation in the very definition of Jaccard distance we are using.

In table 5 we also show a qualitative comparison between original reviews and their predictions for the "best" and the "worst" case based on the L2 error. In both cases we observe a larger number of concepts in the prediction than in the original review. This is particularly true in the worst case (which has an error of 4.15 in L2); in this case, we are able to predict the concepts present in the original review, but also many others that were not originally part of it (*e.g.*, 'break', 'accurate', or 'awesome'). In the best case (which has an error of 1.68 in L2) the prediction is almost perfect but we were not able to generate the concept 'would recommend', obtaining 'problem' instead, which is not used in the original text.

[5]It is part of the MLlib Apache's Library.

[6]Apache Spark's scalable machine learning library.

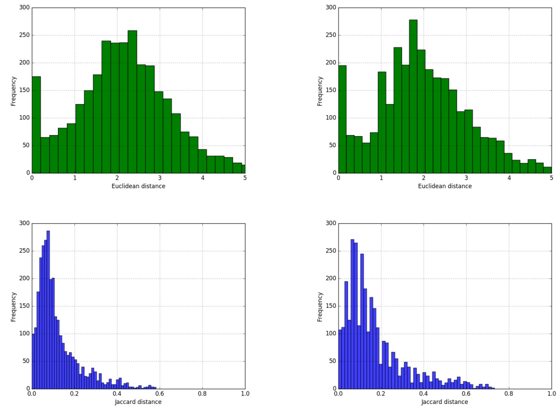

Figure 3: Euclidean (top) and Jaccard (bottom) distances histograms for K=5 (left) and K=10 (right) with a frequency filtering of 50.

We want to highlight however that, due to the large vocabulary size and sparsity of the data, many reviews are predicted as zero vectors even though they contain nonzero frequencies in the training set. This makes it advisable to use a variant of the ALS method specifically optimized for low-rank matrices, a problem that we attempt to address in the future.

Table 5: Example of a Amazon review text and prediction.

Review Text	Real	Prediction	Quality
Good little connector cable. Well constructed, durable, built to last. I would recommend this cable to connect stomp boxes. Worth it.	'cable', 'good', 'well','would recommend'	'cable','good' 'well','problem', 'perfect'	Best
I've been stringing my guitars with D'Addario for several years. While my jazz box is set up with heavy strings, my Les Paul goes lighter–horses for courses, you might say. I've been using D'Addario XL strings on this guitar for a while, although it's the EXL140 (slightly larger bottoms). But why not give these lighter strings a try? The quality is obviously high. Tone is good, as is durability. If you do much complex chording, you're going to miss the heavier string, but for shredders and pure rock players these are very nice. I'll be going back to the EXL 140s, since I prefer the more positive feel of the larger string. But fat or skinny, D'Addario makes some of the best strings out there.	'very nice','tone' 'quality','prefer','lighter' 'high','good' 'best'	'very nice, 'tone' 'lighter' 'high','good', 'best','great', 'awesome','finger','happy' 'highly recommend','like' 'price','would recommend' 'wow','worth','nice'	Worst

5 CONCLUSIONS AND FUTURE WORK

We have shown in this paper that it is possible to predict a user's review for a previously unreviewed product by means of a CF based model. This method uses a pre-built opinion dictionary that only contains the words that do represent concepts. For this purpose we used state-of-the-art NLP tools and implemented a new ALS-based algorithm to unveil the latent dimensions that best represent the user's expressiveness. The fundamental idea behind the model is that the different reasons that lead to a user's opinion (expressed in a review) may be captured by those latent factors, and hence, they can be predicted through a direct comparison with other users of similar taste.

The results show that the model, although still at a preliminary stage of development, is able to deduce the latent dimensions and that the predictions are meaningful enough as to provide a useful insight into the potential opinion of a user on a new product. These results are far from final, however. A better dictionary, more representative of the users' tastes, is necessary to obtain more accurate predictions. Our preliminary results indicate the presence of concepts of very little value that should better be avoided if this approach is really to be used to provide explanations in recommendation systems.

Finally, we mentioned the scalability of this new approach –implemented using a Hadoop based cluster and its distributed computational and storage resources. We intend to conduct further and more exhaustive analysis in larger datasets, with larger dictionaries, by enlarging these capabilities. It is our believe that a deeper analysis of the latent factors and their categorization will allow a better understanding of the conceptual parts of a language involved in the users' opinions.

ACKNOWLEDGEMENTS

The authors want to thank Bitext (http://bitext.com) for providing NLP services for research. They also acknowledge the support the Universidad Europea de Madrid through the E-Modelo research project. Special thanks to Hugo Seage for developing a significant part of the code used for experimentation, and to Jose M. López and Javier García-Blas for their insightful comments.

REFERENCES

Bennet, J. and Lanning, S. (2007). The netflix prize. In *KDD Cup and Workshop*.

Chen, L. and Wang, F. (2013). Preference-based clustering reviews for augmenting e-commerce recommendation. In *Knowledge-Based Systems (2013)*.

J. McAuley, R. P. and Leskovec, J. (2015). Inferring networks of substitutable and complementary products. In *Knowledge Discovery and Data Mining*.

L. Fangtao, L. Nathan, J. H. K. Z. Q. Y. and Zhu, X. (2011). Incorporating reviewer and item information for review rating prediction. In *Proc. of the 23rd IJCAI, AAAI Press (2011)*, pages 1820–1825.

M. D. Ekstrand, J. T. R. and Konstan, J. A. (2012). Collaborative filtering recommender systems. In *Foundations and Trends in Human-Computer Interaction*, pages 81–173.

M. Terzi, M. Rowe, M. F. and Whittle, J. (2011). Text-based user-knn: measuring user similarity based on text reviews. In *Adaptation and Personalization, Ed. Springer*, pages 195–206.

McAuley, J. and Leskovec, J. (2013a). From amateurs to connoisseurs:modeling the evolution of user expertise through online reviews. In *WWW, 2013*.

McAuley, J. and Leskovec, J. (2013b). Hidden factors and hidden topics: understanding rating dimension wih review text. In *Proceedings of the 7th ACM conference on Recommender Systems, RecSys, 2013*.

R. Benjamins, David Cadenas, P. A. A. V. and Gomez, J. (2014). The voice of the customer for digital telcos. In *Proceedings of the Industry Track at the International Semantic Web Conference 2014*.

Sharma, A. and Cosley., D. (2013). Do social explanations work? studying and modeling the effects of social explanations in recommender systems. In *WWW, 2013*.

W. Zhang, G. Ding, L. C. and Li, C. (2010). Augmenting chinese online video recommendations by using virtual ratings predicted by review sentiment classification. In *Proc. Of the IEEE ICDM Workshops. IEEE Computer Society, Washington, DC (2010)*.

Y. Koren, R. B. and Volinsky, C. (2009). Matrix factorization techniques for recommender systems. In *Journal Computer, IEEE Computer Society Press*, pages 30–37.