Text-based collaborative filtering for cold-start soothing and recommendation enrichment

Charles-Emmanuel Dias, Vincent Guigue and Patrick Gallinari
Sorbonne Université, UPMC Univ Paris 6, UMR 7606, LIP6, F-75005, Paris, France
First.Last@lip6.fr

Abstract—**The difficulty to deal with new users, items and the poor explainability of predictions are well-known weaknesses of collaborative filtering. Classically, the cold-start issue is tackled either by asking for user interaction or exploiting side information while additional explanations are often extracted afterwards in a standalone process. Here, we propose a text-based collaborative filtering recommender system which provides a framework to solve both issues. Our method extends a scalable text embedding technique to build a unified vector space where users and items are mapped. We show how to use this space in a collaborative filtering scheme and demonstrate the interest of our text-based method for cold start soothing and recommendation explanation. The suitability of our approach is backed by competitive results on rating prediction task.**

I. Introduction

RECOMMENDER systems were developed to cherry-pick interesting content in an always growing environment to help users overcome the information overload. These systems can provide individually tailored advice such as which product to buy on Amazon, where to eat out with TripAdvisor, or what to watch on Netflix. One way of gaining insight into one's tastes is to use opinions left online; most of the time as a star rating. Collaborative Filtering (CF) engines extract information from past user behaviors to match consumers with appropriate items.

CF suffers from two well-known weaknesses: cold-start and opaqueness. By relying on past behaviors to predict preferences, recommenders tend to need a long warming period before being operational. This is a problem when no previous records exists, for a new user or for a new item. Moreover, Sometimes called black-boxes, recommender systems solely offer suggestions without any further explanations, rendering them less appealing [1]. Most systems tackling those issues tend to provide separate solutions [1], [2], [3], [4] resulting in scattered systems.

Classically, recommendation engines use ratings as their main source of information to build user and item profiles; yet, text reviews are often associated with those marks. Recent work on recommendation suggests that taking these associated texts into account leads to significant improvements in rating prediction [5], [6] by helping matrix factorization to cope with rating sparsity and extract relevant aspects from items. However, these models use text as a side feature and the textual component is usually extremely basic, not allowing to leverage the richness of review texts for rating prediction.

We investigate here the potential of word embedding techniques [7], [8]. in the context of recommendation. These recent models have foreseen an important success in different applications for natural language processing. They have not been used so far in the context of recommendation.

We then propose a novel approach to recommendation derived from state-of-the-art word embedding technique *word2vec* [7]. We aim at making full use of the richness contained in review text to build a latent space of words, users and items that can be used in a collaborative filtering scheme. Since our goal here is to demonstrate the value of review text for rating prediction, we focus on textual information, not considering ratings, to infer the similarities between users or between items. Ratings could easily be incorporated in an extension of this model, but this is not discussed here.. Text-centered latent space provides a natural framework to solve both cold-start and recommendation explanation problems in one embedded solution. In fact, providing the existence of side textual information, new user and item profiles can be instantly derived. This information is often readily available for new users (blogs, tweets, status updates) and new items (description, tags, critiques). Also, textual information such as related words or reviews can be extracted with this word-space in order to provide explicit pointers to support each suggestion made by the system.

Our text-based model shows competitive results on the classical rating prediction task and a good performance when used in a cold-start setting. Furthermore, we show how to extract words and how to build a reviews summary for a target item and illustrate why this enrichment is valuable for recommendation interpretation.

This paper is organized as follows: section II describes our text-based collaborative filtering model. Experimental results are presented in section III while concluding remarks are in section IV.

II. A text-based collaborative filtering model

Here, in opposition to other classical collaborative filtering models, we do not model users and items with respect to their ratings but according to their words. Specifically, we encode each user and item into a word vector space such that similar items and users are mapped close together. Embeddings with these properties can then seamlessly be used in a classical neighborhood-based collaborative filtering scheme [9].

Our model - **Conceptual Skip-Gram (CSG)** - extends *word2vec* word embeddings skip-gram method [7] and is inspired by one of its extension to build paragraph vectors, the distributed bag of words model [10].

Conceptual skip-gram - Negative sampling

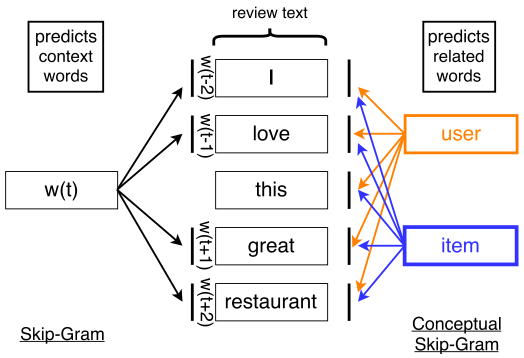


Fig. 1. (left) Classical Skip-Gram model over text. (right) Conceptual Skip-Gram binding users and items to the word vector space.

As mentioned above, this model derives from the ”distributed bag of words” introduced by [10] which considers a paragraph as a set of words and tries to predict them, thus creating an embedding for it. Each user and each item, which we call concepts, can be defined by a set of words from their associated review texts. We use them similarly as paragraphs in the above-mentioned algorithm: the goal is to find the best embedding for each concept in order to predict its associated words (Fig 1).

The distributional hypothesis states that words that are used and occur in the same contexts tend to have similar meanings. With this underlying assumption [7] optimizes the $P(context|word)$ (eq. 1) over a corpus to create word embeddings. The context being defined as a fixed window around the current word.

$$\operatorname*{argmax}_{\theta} \prod_{w \in corpus} \left[\prod_{w_c \in context(w)} P(w_c|w;\theta) \right] \tag{1}$$

Here, in addition, we also optimize $P(word|concept)$ (eq. 2) to form users and items embeddings. We speculate that related concepts are defined by analogous words.

$$\operatorname*{argmax}_{\theta} \prod_{c \in corpus} \left[\prod_{w \in concept(c)} P(w|c;\theta) \right] \tag{2}$$

For training, we use the negative-sampling approach [7] which is thoughtfully described in [11]; The goal is to train the model to differentiate noise from actual data, leading to good embeddings.

Words and concepts embeddings representing every concept of a review are learnt in parallel with two separate but similar objectives:

- Words co-occurring in the reviews should be close to each other while those not co-occurring should be far (equation 3)
- Concepts (users and items) should be close to their words and far from the others (equation 4).

Formally, with $\sigma(x) = \frac{1}{1+e^{-x}}$, the objective functions are the following:

$$\log \sigma(v_a'^{\intercal} v_w) + \sum_{i=1}^{k} \mathbb{E}_{v_b \sim P_n(\mathbf{w})} \log \sigma(-v_b'^{\intercal} v_w) \tag{3}$$

$$\log \sigma(v_a'^{\intercal} v_c) + \sum_{i=1}^{k} \mathbb{E}_{v_b \sim P_n(\mathbf{w})} \log \sigma(-v_b'^{\intercal} v_c) \tag{4}$$

Where v_w and v_c are the words (resp. concept) and their associated words v_a (co-occurring or concept-related). The v_b are the noise, drawn using a unigram law raised to the $\frac{3}{4}$ power[1] as in [7].

Collaborative filtering with embeddings

After training, we have a common embedding space for words and concepts. Vocabulary size is $|words| + |concepts|$. As in *word2vec*, we can use cosine similarity to find related elements within our latent space. For sake of simplicity we shift our similarity measure to $[0, 1]$ so that we can use it directly in different weighting schemes:

$$sim(u,v) = \alpha_{uv} = \frac{\frac{<u,v>}{\|u\| \times \|v\|} + 1}{2} \in [0,1] \tag{5}$$

By considering each user and item as concepts defined by their words, similar users and similar items should be mapped closely if they use similar words. We can now use this space as a similarity space between items and users in a classical neighborhood-based collaborative filtering [9] scheme: A predicted rating $\hat{r_{ui}}$ of a user u on a target item i is the similarity-weighted sum of either the user's ratings on a set $\mathcal{N}_i$ of similar items (to the target) or the item's ratings of a set $\mathcal{N}_u$ of similar users. Here, we also mean-center the ratings (eq 6) to take the rating bias into account.

$$\hat{r_{ui}} = \mu_i + \frac{\sum\limits_{j \in \mathcal{N}_i} \alpha_{ij}(r_{uj} - \mu_j)}{\sum\limits_{j \in \mathcal{N}_i} \alpha_{ij}}, \hat{r_{ui}} = \mu_u + \frac{\sum\limits_{v \in \mathcal{N}_u} \alpha_{uv}(r_{vi} - \mu_v)}{\sum\limits_{v \in \mathcal{N}_u} \alpha_{uv}} \tag{6}$$

Besides predicting ratings using learnt concepts representations, our system presents several benefits. In a cold-start setting, any textual content can be used to create a profile by simply optimizing equation 4 for each word from this side data. This cold representation can then be used in a simple, not mean-centered, neighborhood CF. This is a major benefit given that third party text-content such as blogs, tweets or descriptions are readily available via numerous web APIs.

Moreover, each user or item being bound to its closest words it is easy to extract them to describe a product or a user (eq. 5), as a wordcloud for example.

Finally, we can also extract full reviews or sentences by considering them as the sum of their words [13] from similar users. Then, they can be used to build a personalized summary of a suggested product reviews in order to explain the recommendation.

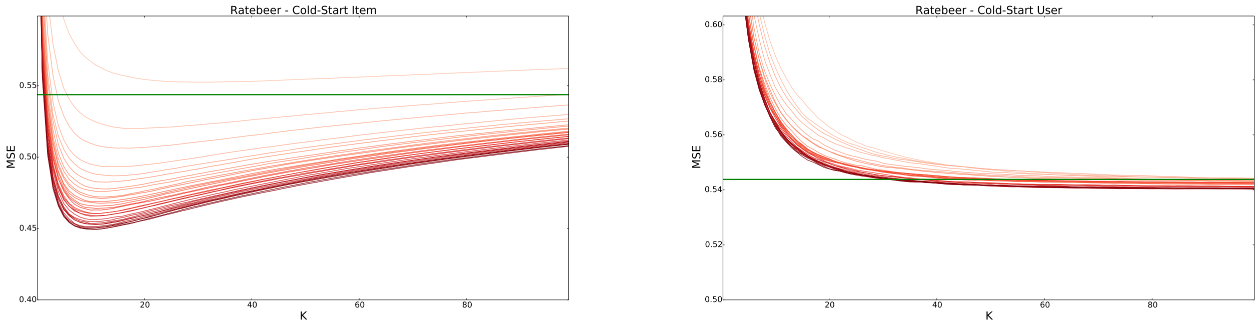


Fig. 2. Item (left) and User (right) cold-start scenario on the ratebeer dataset. Color intensity represents the number of used reviews to build a profile (from 1 to 20 reviews, from top to bottom). The green line is the performance of respectively the user and the item bias models.
x axis: size of the neighborhood $\mathcal{N}$, y axis: rating prediction error in MSE.

III. Experiments

In this section, we present several experiments to demonstrate the interest and the advantages of our text-based collaborative filtering system for recommendation by evaluating different tasks:

- **Rating prediction**: This is the classical task of rating prediction evaluated in MSE. The goal is to find out whether or not our text-based CF system is competitive with state-of-the-art methods.
- **Cold-start user/item rating**: If we assume to have textual info about the new user or the new item, how does MSE evolve in a cold-start scenario ?
- **Recommendation enrichment**: How can our text system explain recommendations?

Datasets and baselines

In order to evaluate our model we rely on three types of datasets from [5]: Beer reviews from *Beeradvocate* and *Ratebeer*, movies and music reviews from Amazon and *Yelp*-places reviews. Review duplicates and less frequent words have been removed in each of these datasets (table **??**.). We compare our model to [14] matrix-factorization bias model and to [5] Hidden Factor & Topic model (HFT) which aligns latent factors to text-extracted themes.

Rating prediction

We evaluate our model using MSE; results are presented in table I. To compare with the state-of-the-art results from [5] in all fairness, we follow their settings: Datasets were randomly split 80% for training and 20% for validation and testing.

We can see that our system, when using item-item similarity, is competitive with respect to state-of-the-art models on beer and Yelp reviews despite not directly optimizing rating prediction. Yet, the text seems to penalize HFT & CSG on both the movie and the music dataset where both models are less accurate than classical matrix factorization.

Dataset	μ	MF	HFT [5]	CSG-kNN		k
				user	item	
Ratebeer	0.701	0.306	0.301	0.336	**0.286**	23
Beeradv.	0.521	0.371	0.366	0.382	**0.357**	29
Movies	1.678	**1.118**	1.119	1.39	1.304	33
Music	1.261	**0.957**	0.969	-	1.201	26
Yelp	1.890	1.49	-	1.591	**1.407**	27

TABLE I
Rating prediction in MSE. HFT is the best "Hidden Factor & Topic model" reported from [5]. CSG-kNN: best k-neigh results in the text-space using k nearest neighbors.

Cold-start

We argue that our text-based CF method can soothe cold start by using third party text. To simulate third party texts we used reviews stripped out everything but text. This is similar to not mean-centering our data. Despite the obvious data bias, we claim that these experiments still give a good idea of text usefulness for cold-start.

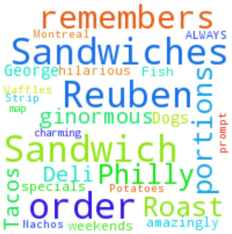
Two different experiments were conducted. First, using the same splitting as for rating prediction, by just removing mean-normalization from the prediction rule (equation 6). Results are shown in table II. We compare obtained results to [14] bias model behavior when facing cold start: predicting the item's mean in case of a new user and the user's mean in case of a cold item. Mean ratings are known to be a strong baseline despite its simplicity.

Dataset	μ	New User		New Item	
		μ_i	CSG-kNN	μ_u	CSG-kNN
Ratebeer	0.701	0.341	**0.333**	0.599	**0.371**
Beeradv.	0.518	0.397	**0.386**	0.490	**0.419**

TABLE II
Rating prediction in MSE without mean-normalization in the prediction rule to simulate cold-start. μ, μ_i, μ_u: mean, mean item,user: respectively predicts the global/item/user mean.

Our system is always better than the mean ratings. The most significant improvement is when dealing with item cold-start. This first experiment shows the interest of text profiles in a cold-start setting and confirms the superiority of item similarity over user similarity.

[1] raising the unigram distribution at the 3/4 power smooths it to make rare words appear more [12]



Predicted rating: 4.70

Extracted personalized summary: The staff is extremely friendly. On top of being extremely large portions it was incredibly affordable. Most of girls are good, one is very slow, one is amazing. The fish was very good but the Reuben was to die for. Both dishes were massive and could very easily be shared between two people.

Fig. 3. Example of recommendation enrichment on the Yelp corpus.

A second experiment was conducted to see how many reviews were necessary to obtain better results than these mean baselines. Active users and items were randomly extracted from the dataset. To simulate cold-start, we selected 40 reviews from each. Half of them for testing and the other half is for training. This way, we can gradually enlarge the data to learn our cold-start profiles while not reducing the test set.

Results are shown in figure 2. We can see that for item cold-start (left), even with a small amount of text (from two reviews and up), our system is more effective than the user mean, even with a low neighbors count. However, for user cold-start (right) at least ten reviews are needed to obtain satisfying results. The outcome of those experiments is quite promising for item cold-start where the word embedding technique is clearly effective. Also, given the difference between the item and the user graph, we can infer that review data is much more informative about item features than about user tastes. This also explains why item similarity is much more effective than the user one for collaborative filtering.

Recommendation enrichment

Another built-in feature of our text-based system is the ability to justify each recommendation. On one side it is possible to extract the nearest words of an item and offer a sort of description of it, as a wordcloud for example, helping the user to grasp quickly what the product offers. On another side, it's also possible to extract some existing sentences (following [13]) to build a personalized summary of existing reviews on a target item. Figure 3 shows an example of enriched recommendation. Which illustrates how beneficial written information is for the user.

IV. Conclusion and Discussion

In this paper, we proposed a text-based collaborative filtering model relying on a state-of-the-art text embedding technique. Contrary to classical CF techniques mainly using past ratings to build latent representations, we construct them using the text. We demonstrate that our technique has the advantage to provide a natural framework to solve both cold-start and opaqueness by leveraging the richness of review text.

Moreover, given that no substantial filtering was made on review text used to build latent profiles, it is possible that a more sophisticated preprocessing yields even better results.

Acknowledgment

This work was done with partial support from FUI ITER-RH Project.

References

[1] N. Tintarev and J. Masthoff, "A survey of explanations in recommender systems," *Proceedings - International Conference on Data Engineering*, pp. 801–810, 2007.

[2] Z. Gantner, L. Drumond, C. Freudenthaler, S. Rendle, and L. Schmidt-Thieme, "Learning attribute-to-feature mappings for cold-start recommendations," *Proceedings - IEEE International Conference on Data Mining, ICDM*, pp. 176–185, 2010.

[3] N. N. Liu, X. Meng, C. Liu, and Q. Yang, "Wisdom of the better few: cold start recommendation via representative based rating elicitation," *Proceedings of the 2011 ACM Conference on Recommender Systems, RecSys 2011, Chicago, IL, USA, October 23-27, 2011*, pp. 37–44, 2011.

[4] M. Saveski and a. Mantrach, "Item cold-start recommendations: learning local collective embeddings," *RecSys '14 Proceedings of the 8th ACM Conference on Recommender systems*, pp. 89–96, 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2645751

[5] J. McAuley and J. Leskovec, "Hidden factors and hidden topics: understanding rating dimensions with review text," *Proceedings of the 7th ACM conference on Recommender systems - RecSys '13*, pp. 165–172, 2013. [Online]. Available: http://dl.acm.org/citation.cfm?id=2507163

[6] G. Ling, M. R. Lyu, and I. King, "Ratings Meet Reviews , a Combined Approach to Recommend," pp. 105–112, 2014.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," *Nips*, pp. 1–9, 2013.

[8] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global Vectors for Word Representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, 2014.

[9] C. Desrosiers and G. Karypis, "A comprehensive survey of neighborhood-based recommendation methods," *Recommender Systems Handbook*, vol. 69, no. 11, pp. 107–144, 2011. [Online]. Available: http://www.springerlink.com/index/N3JQ77686228781N.pdf

[10] Q. Le and T. Mikolov, "Distributed Representations of Sentences and Documents," *International Conference on Machine Learning - ICML 2014*, vol. 32, p. 11881196, 2014. [Online]. Available: http://arxiv.org/abs/1405.4053

[11] Y. Goldberg and O. Levy, "word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method," *arXiv preprint arXiv:1402.3722*, no. 2, pp. 1–5, 2014. [Online]. Available: http://arxiv.org/abs/1402.3722

[12] O. Levy, Y. Goldberg, and I. Dagan, "Improving Distributional Similarity with Lessons Learned from Word Embeddings," *Transactions of the Association for Computational Linguistics*, vol. 3, pp. 211–225, 2015. [Online]. Available: https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/570

[13] M. Kageback, O. Mogren, N. Tahmasebi, and D. Dubhashi, "Extractive Summarization using Continuous Vector Space Models," *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality (CVSC)@EACL 2014*, pp. 31–39, 2014.

[14] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, pp. 42–49, 2009.