Evaluating Vector Representations from User's Reviews in a Recommendation Task

Vitor R. Tonon, Tiago P. da Silva, Vinícius Ferreira, Gean T. Pereira, Solange O. Rezende

[1]Institute of Mathematics and Computer Science, University of São Paulo,
São Carlos, SP, Brazil

{vitor.tonon,tpinho,vfsilva,geantrinpereira,solange}@usp.br

Abstract. *The recommendation task is a prominent and challenging area of study in Machine Learning. It aims to recommend items such as products, movies, and services to users according to what they have liked in the past. In general, most of the recommendation systems only consider structured information. For instance, in recommending movies to users they might use features such as genre, actors, and directors. However, unstructured data such as users' reviews may also be considered, since they can aggregate important information to the recommendation process, improving the performance of recommendation systems. Thus, in this work, we evaluate the use of text mining methods to extract and represent relevant information about user reviews, which were used alongside with rating data, as input of a content-based recommendation algorithm. We considered three different strategies for this purpose, which were: Topics, Embeddings and Relevant Embeddings. We hypothesized that using the considered strategies, it is possible to create more meaningful and concise representations than the traditional bag-of-words model, and yet, increase the performance of recommendation systems. In our experimental evaluation, we confirmed such a hypothesis, showing that the considered representations strategies are indeed very promising for representing user reviews in the recommendation process.*

1. Introduction

A great amount of data is generated every second on the web, which is originated from news, music, movies and all kinds of other products and services. Because of this abundant information, users usually suffer difficulties identifying and choosing the products or services that best meet their needs. In this context, recommendation systems seek to find and recommend relevant items to users, considering their history of interactions with the system, such as previous liked and/or disliked items and viewing information [Bobadilla et al. 2013, Aggarwal 2016, Shah et al. 2016]. These systems help users to find more easily what they are looking for and, therefore, bring several advantages to the user and sellers/content providers, such as increasing user satisfaction and the number of sales.

There are two main categories of recommendation systems [Ricci et al. 2011]: collaborative filtering and content-based filtering. Collaborative filtering seeks to recommend items by identifying patterns of relationship between items and/or users, considering this user's interaction data, also known as ratings. Content-based filtering considers user data or items' features to create the user and/or items profiles, and then uses these profiles to find similar items to recommend. One main disadvantage of using collaborative filtering is the difficulty in generating good recommendations when the amount of data is small (the cold-start problem) [Aggarwal 2016], because this kind of system relies only on historical rating data. Content-based filtering algorithms, on the other hand, allow considering external data in the recommendation process, such as item metadata (e.g. name and description) and user reviews, which can alleviate the cold-start problem. Therefore, in this work, we use a content-based filtering algorithm.

User reviews consist of natural language texts, written by users, containing their opinions about different items' features (e.g. if the items are movies, these features could be the actors, director, the total duration of the movie, and several other relevant entities that represent this movie). Since the ultimate goal of recommendation systems is to predict user preference and these texts contain such information, user's reviews are extremely valuable in these systems [McAuley and Leskovec 2013, Zheng et al. 2017, Guo et al. 2017, Baral et al. 2018, Sundermann et al. 2019]. Thus, in this work, our goal is to extract these relevant features of user reviews, creating items representations from these texts, which will then be aggregated in the recommendation process, alongside the rating data.

There are several ways to create vector representations from textual data. The simplest form is to use the bag-of-words model, in which the frequency of words in a document is considered as these vector representations [Aggarwal and Zhai 2012]. Although this method usually generates good results, it creates high dimensionality representations, which may further worsen the cold-start problem. One possibility to address this problem would be to extract the relevant topics from these texts and use only them in the bag-of-words model. Another possibility, which has been presenting very promising results, is to use word embeddings, which consists of high-quality word vectors trained on large datasets. The main advantage of using such embeddings is that words with similar meanings tend to have closer representations in the resulting embedded space, bringing text semantics to the created representations [Mikolov et al. 2013].

Given this scenario, we propose to extract and create vector representations from user reviews considering these different strategies, which are Topics, Embeddings and Relevant Embeddings, and to aggregate them in a content-based collaborative filtering algorithm. We hypothesize that it is possible to create more meaningful and concise representations from these texts, using topics modeling, document embeddings and/or document relevant embeddings, and that aggregating them in the recommendation process can increase the performance of recommendation systems, when compared to the traditional bag-of-words model. In summary, the contributions of this work are: (i) the proposal to use text mining algorithms in the recommendation task; and (ii) the evaluation of three different strategies to create item representations from textual data.

We carried out an experimental evaluation using the MovieLens benchmark dataset enriched with reviews collected from IMDb. An analysis of the results revealed that our proposal to use topics modeling and word embeddings in the recommendation task statistically outperformed the traditional bag-of-words model. In addition, the use of word embeddings may bring text semantics to the created representations, which raises a variety of future research regarding the interpretability of the recommendations.

The remaining of this work is structured as follows. In Section 2, we present the most related works to our proposal, that we could find in the literature. In Section 3, we present three different strategies for representing and extracting relevant information from user's reviews and how to aggregate them in the recommendation task. In Section 4, we present the experimental evaluation we performed and the results we obtained. Finally, in Section 5 we draw our main conclusions from this work and present some perspectives of future work.

2. Related Work

For the last few years, considering user reviews in recommendation systems has been a hot research topic, since several studies [Almahairi et al. 2015, Zheng et al. 2017, Baral et al. 2018, Sundermann et al. 2019] have been reporting an increase in the quality of recommendations when using such information. In order to aggregate user's reviews in the recommendation process, it's necessary to extract relevant information from these texts and several methods have been used for this purpose, such as the traditional bag-of-words method [Almahairi et al. 2015], aspects extraction [Baral et al. 2018], topics modelling [McAuley and Leskovec 2013] and word embeddings [Zheng et al. 2017].

[Almahairi et al. 2015] proposed two models that represent the text of the reviews as bag-of-words, named Bag-of-Words regularized Latent Factor (BoWLF), and Language Model regularized Latent Factor (LMLF). According to the authors, both methods achieved good performance over the literature's classic methods. However, their approach did not consider the semantic meaning of words due to the adopted bag-of-words representation. Similarly, [Raghavan et al. 2012], use bag-of-words features extracted from reviews text to train a regression model to predict the quality score of Amazon's products.

[Baral et al. 2018] extracted from the user's reviews points of interest (POIs) and their related aspects, which consist of characteristics of the item being reviewed. They used this information, alongside with the rating data, in a matrix factorization recommendation algorithm to generate recommendations. [Baral et al. 2018] reported that there was an increase in the performance recommendations when aggregating such information.

[McAuley and Leskovec 2013] noticed that traditional recommendations system often discard review texts, using only the user's interaction data. However, according to them, these reviews contain the explanations as to why users rate/view some items and, therefore, are extremely valuable to recommendations systems. Thus, they extracted the most important topics of these texts, using Latent Dirichlet Allocation (LDA), which is a topic modeling approach, and aggregated such information in a matrix factorization algorithm. According to them, their method was able to generate higher accuracy recommendations when using the extracted topics from reviews.

In [Zheng et al. 2017], a recommendation system was proposed using neural networks, user reviews, and word embeddings. In order to alleviate the sparsity problem and the cold-start problem, the authors focused on learning user behavior and items' features from user reviews, aggregating such information in the recommendation algorithm, which was based on neural networks. Besides this, word embeddings were considered in this process to bring the semantics of the texts to the recommendations. According to the authors, their proposal outperformed all the considered baselines.

All of this research considered user reviews in the recommendation process.

We consider that our proposal is more related to [McAuley and Leskovec 2013] and [Zheng et al. 2017] because we also used topics modeling and word embeddings to extract and represent these texts in the recommendation process. However, one main difference is that we want to evaluate these strategies separately to verify which one generates better recommendations when aggregated in a recommendation system. With this in mind, in the next section, we present our proposal for using these strategies in the recommendation task.

3. Generating Vector Representations from User's Reviews

In general, traditional recommendation systems only considers the user's data and the items ratings to generate recommendations. However, several studies [Baral et al. 2018, Guo et al. 2017, He et al. 2015] have demonstrated the importance of aggregating user reviews in this process, due to its relevant information about user's opinions on item features. Thus, in this work, we propose to use three different strategies from the Text Mining domain to extract and represent relevant terms from user reviews, aggregating them in a content-based recommendation algorithm. We believe that by bringing external information to the recommendation process, such as these relevant terms from user reviews, we could alleviate the sparsity and cold-start problem and also enhance the accuracy of recommendations.

In Figure 1, we present an overview of our approach for generating recommendations from user reviews. The main idea of our proposal is to extract relevant information from user's reviews using three methods: (i) Topics Modeling, which considers the Hierarchical Dirichlet Process (HDP) [Wang et al. 2011]; (ii) Document Embeddings and (iii) Document Relevant Embeddings, which considers the word embeddings of the documents [Mikolov et al. 2013]. Since these reviews are related to the items of the system, this extracted information is used to create item representations, which are then used alongside with rating data to generate recommendations by a content-based filtering algorithm. Our idea is to compare the accuracy of these generated recommendations to those generated by using only the traditional bag-of-words model as item representations (our considered baseline).

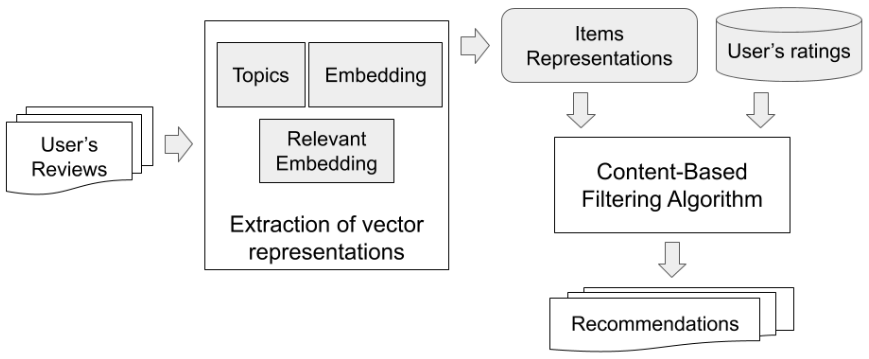

Figure 1. Overview of the proposed approach for generating recommendations from users' reviews.

In the following sections, we detail the baseline we used and each one of the three considered strategies: (i) Bag-of-Words Model (BOW) using Term Frequency-Inverse

Document Frequency Weighting (TF-IDF Weighting), which is our baseline approach, (ii) Topic Modeling using TF-IDF Weighting, (iii) Document Embeddings, and (iv) Document Relevant Embeddings.

3.1. Bag-of-Words Model using TF-IDF Weighting

To convert the text data into a numerical form, we use the TF-IDF weighting of the terms in the text [Baeza-Yates and Ribeiro-Neto 2008]. This numerical value represents the importance of a term to a document in a corpus and has two parts: td and idf. Considering a term t and a document d, the first part $tf_{t,d}$ is the term frequency, i.e., the frequency a term occurs in a document. It can be calculated by:

$$tf_{t,d} = \frac{n_{t,d}}{n_d}, \tag{1}$$

considering $n_{t,d}$ as the number of times that t appears in the document d and n_d as the number of terms in d.

The second part is idf, which represents the weight of rare words across all the documents in the corpus, i.e., terms that occur rarely in the corpus have a high idf score. The idf can be calculated as:

$$idf_t = log\frac{N}{df_t + 1}, \tag{2}$$

where N represents the total number of documents in the corpus and df_t the number of documents where $tf_{t,d} \neq 0$. The 1 is added to avoid divisions by zero when a term t is not present in the corpus. Using tf and idf, the value of $tf - idf$ $w_{i,j}$ for a term i in a document j in the corpus can be calculated by:

$$w_{i,j} = tf_{i,j} \times log\frac{N}{df_i + 1}. \tag{3}$$

The TF-IDF weighing of a corpus is a matrix $W_{m \times n}$ where m represents the number of terms and n represents the number of documents in those documents:

$$W = \begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} & \cdots & w_{1,n} \\ w_{2,1} & w_{2,2} & w_{2,3} & \cdots & w_{2,n} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ w_{m,1} & w_{m,2} & w_{m,3} & \cdots & w_{m,n} \end{bmatrix}_{m \times n}. \tag{4}$$

In this work, we calculated the bag-of-word model using TF-IDF Weighting for each one of the user's reviews, creating item representations. These representations were then used as input of the recommendation algorithm, alongside with the rating data. We used this method as the baseline in our experiments.

3.2. Topic Modeling using TF-IDF Weighting

The traditional bag-of-words model, described in the previous section, considers all words to create its text representation, which may cause dimentionality problems when the number of words is high. Therefore, one possibility to address this problem is to use topic modeling methods [Blei et al. 2003], which seeks to identify and extract only the relevant terms of a collection of documents. For this purpose, we considered the Hierarchical

Dirichlet Process (HDP), which is a topic modeling algorithm that extracts several topics that are learned from the textual data.

The HDP is an extension of LDA, designed to address situations where the number of topics in document-modeling terms is not known *a priori*, which is our case. In this model, the words with the highest probabilities in each topic usually provide good insights into what the topic is related to. Thus, using the HDP algorithm, we may extract the relevant terms from user's reviews and, based only on this subset of words, we may create vector representations for each review using TF-IDF Weighting, as described in Section 3.1. These representations are then used as input of the recommendation system. This approach is referred to as **Topics** in the remaining sections.

3.3. Document Embedding

One possible way to create vector representations from textual data is to use word embeddings, which consists of high-quality word vectors trained on large datasets. One main advantage of using such embeddings is that two similar words tend to have closer representations in the resulting embedded space, which may bring some semantics to the created representations. Considering the benefits of using such an approach, our goal was to create a vector representation for each user's review using these embeddings. For this, we considered the Google's pre-trained word and phrase embeddings[1], which was trained on a Google News dataset, with about 100 billion words, and consisted of vectors of size three hundred.

For each review in our dataset, we considered the embeddings for all of its words and aggregated them to get a unique representation for each document:

$$vector(D_i) = \frac{1}{|D_i|} \sum_{j=1}^{|D_i|} vector(w_j), \tag{5}$$

such that $|D_i|$ represents the number of words in document D_i and *vector*(w_j) represents the word embedding of word w_j. As each review represents a given item, these documents representations were then used as item profiles in the content-based recommendation system. This approach is referred to as **Embeddings** in the remaining of this paper.

3.4. Document Relevant Embeddings

As presented in Equation 5, the previous approach considers every word of a document to calculate its vector representation. However, several words occur only a few times in the document, and therefore may not be very representative of the document as a whole, which could then worsen the performance of our method. To alleviate this problem, we considered only the most relevant words (according to the TF-IDF measure) to calculate the document embeddings:

$$vector(D_i) = \frac{1}{|W_{relevant}|} \sum_{w_j \in W_{relevant}} vector(w_j), \tag{6}$$

where $W_{relevant}$ is the set of document words with the highest TF-IDF values. This approach is referred as **Relevant Embeddings** in the remaining sections.

[1]https://code.google.com/archive/p/word2vec/

In the next section, we present the experimental evaluation we performed to validate our approach.

4. Experimental Evaluation

In this work, our goal was to extract relevant information from user reviews to enrich the recommendation process, alleviating the sparsity and the cold-start problem. Thus, we considered different strategies of representing texts in a format suitable to recommendation algorithms. We identified that using the bag-of-words model for this purpose is the most common and popular alternative, usually generating good results. However, the resulting representations have high dimentionality, which could worsen the accuracy of recommendations. Therefore, we applied three alternatives that create more concise representations: Topics, Embeddings, and Relevant Embeddings, as described in the previous section. We hypothesized that using these three strategies generates more meaningful and concise representations from user reviews, which when aggregated in the recommendation process can increase the performance of recommendation systems when compared to the traditional bag-of-words model. In this section, we present the experiments we made to validate our hypothesis and the results we obtained.

4.1. Experimental Settings

In our experiments, we used the Natural Language ToolKit (NLTK) [2], a python library for Natural Language Processing (NLP), to manipulate the user's reviews. We also used the scikit-learn [3] library and Pandas [4], since it provides fast and effective data structures and a full set of tools to manipulate these data. We adopted an extension of the *MovieLens*[5] dataset. The original dataset, which consisted of ratings made by users to movies, was enriched using IMDb[6] reviews. These reviews are natural language texts written by users and contain their opinions about the different features of the movies. The dataset consists of 535,784 ratings, 3,564 movies, 3,974 users and 24,880 reviews.

According to [Aggarwal 2016], the recommendation task may be seen as a rating prediction task, which consists of predicting the user's preference for the items of the system. Therefore, to evaluate the performance of a recommendation system, we can calculate the error of the predicted ratings against a set of real ratings (i.e. the ratings in the test set). The most common error metric is the Root Mean Squared Error (RMSE), which is defined in Equation 7:

$$RMSE = \left(\frac{1}{|S|} \sum_{(i,j) \in S} (r_{user_i,item_j} - \hat{r}_{user_i,item_j})^2 \right)^{1/2}, \tag{7}$$

such that S consists of the entries in the test set, $r_{user_i,item_j}$ represents the real rating of $user_i$ to $item_j$ and $\hat{r}_{user_i,item_j}$ represents the predicted rating by the recommendation algorithm for this user and item.

[2]https://www.nltk.org/

[3]https://scikit-learn.org/stable/

[4]https://pandas.pydata.org/

[5]https://grouplens.org/datasets/movielens/

[6]https://www.imdb.com/

In our experiments, we wanted to compare if the error of the recommendations were lower by using Topics, Embeddings, and Relevant Embeddings approaches than the error by using the tradition bag-of-words model. In order to do so, we performed 10-fold cross-validation and the Wilcoxon statistical test to verify if the results are statistically different. For generating recommendations, we considered the FBC-KNN algorithm, in which the generated representations were used as item profiles. These profiles were then used to calculate the similarities among items. The predicted rating of a user for an item is given by the weighted average of the ratings this user made on similar items. We varied the number of the considered neighbors (parameter K) in our experiments. From past empirical experimentation, we have noticed that $K = \{50, 100, 150\}$ yield good results, thus, in the next section, we present the recommendation results obtained from these values.

Due to the high number of words present in the user's review (> 75,000), the resulting representations of our experiments had high dimensionality when using the traditional bag-of-words model. Given our limited computational power, we were unable to consider all of them in our experiments and, thus, we defined a sparsity threshold, which filtered out the words considered too sparse. After this filtering, our bag-of-words representation considered around 370 words.

4.2. Results and Discussion

In Table 1, we present the mean of the RMSE we obtained for these experiments, considering the results obtained in each fold. In each experiment, we used a different strategy for representing the user's reviews, as described in Sections 3.1, 3.2, 3.2. We highlight in bold the best results obtained from each strategy, given the number of neighbors considered (parameter K) in the recommendation algorithm.

Table 1. RMSE for the recommendations using the vector representations obtained by the extraction methods described in Section 3

Number of Neighbors (K)	Vector Representation Methods			
	Baseline (BOW)	Topics	Embeddings	Relevant Embeddings
50	**1.047540**	1.056690	**1.047175**	1.046862
100	1.048690	1.054230	1.048358	1.046089
150	1.052201	**1.053612**	1.050174	**1.044808**

Considering the created representations to each item by these different strategies, both the baseline and Embeddings were approximate of the same size ($\approx$ 300), in the same way as Topics and Relevant Embeddings ($\approx$ 50). From Table 1, we can see that the baseline and Embeddings strategies presented better results with 50 neighbors, while Topics and Relevant Embeddings had better results with 150 neighbors. This may be an indicator of the impact of the dimensionality of the created representations on the recommendation algorithm. When the algorithm had more external information (in the case of the baseline and Embeddings), it needed to consider fewer neighbors to make a good prediction. In the same way, when it had less information (Topics and Relevant Embeddings), the algorithm needed more neighbors to predict user preference.

As we can see, Topics strategy was significantly the worst approach for generating vector representations for user reviews, because its results were worse than those obtained by the baseline, Embeddings and Relevant Embeddings. This may be caused because we extracted the most relevant topics considering the whole set of documents, instead of each document at a turn, which may result in topics to broad and not very representative of the documents. In future works, we intend to extract the topics for each document separately and then analyze the impact of the created representations on the recommendations.

We may also observe that using word embeddings to represent textual data may be indeed a good representation strategy, because both Embeddings and Relevant Embeddings obtained significantly better results than the baseline. This may indicate that considering the semantics of the reviews has a high impact on the created representations and brings perspectives of creating explanations for the recommendations using these embeddings, which were extracted from user's reviews.

Furthermore, we may conclude that Relevant Embeddings is the best representation strategy among all of the considered strategies, because it also obtained better results than Embeddings. This may confirm that some words are not that important when representing a document, since filtering some of them yielded better results than considering all of them.

5. Conclusion

Recommendation systems seek to predict and recommend relevant items to users, considering their history of interactions with the system (i.e. ratings data). However, considering only this information is not enough to make good predictions, especially when the number of ratings is small. This problem is known as the cold-start problem and may impair the accuracy of recommendations. There are several ways to alleviate such a problem, such as using external information in the recommendation systems, as user reviews.

User reviews are natural language texts, in which users write about their opinions regarding different items' features. Several research [Almahairi et al. 2015, Zheng et al. 2017, Baral et al. 2018, Sundermann et al. 2019] aggregates this information into the recommendation process, using different strategies to represent them in this process. The most common representation model is the bag-of-words model, but it creates high dimensionality representations, which may impair the performance of the recommendations. In this work, we explored three alternative strategies, which were: Topics, Embeddings, and Relevant Embeddings.

Our goal was to create lower dimensionality representations using these strategies, to use as the input of the recommendation algorithms. We hypothesized that it is possible to create more meaningful and concise representations from user reviews, using topics extraction and/or word embeddings and that aggregating them in the recommendation process can increase the performance of recommendation systems, when compared to the traditional bag-of-words model.

The experimental evaluation we performed showed that creating vector representations from these texts using word embeddings is indeed a good strategy, because both Embeddings and Relevant Embeddings performed better when generating recommendations (lower error) than using the traditional bag-of-words model. However, from our experiments, we noticed that the topics extraction method (the Topics strategy) was not good at creating vector representations for the recommendation process. Perhaps this was caused by considering all of the documents when extracting the most relevant terms, instead of considering each of them individually.

As future work, we intend to explore the interpretability of the created word embeddings representations, to verify if it is possible to generate explanations for the recommendations using this information. Furthermore, we want to propose a model for extracting topics from each review separately and considering only these topics to represent each review in the recommendation algorithm. Finally, we want to explore the computational complexity and scalability of our proposed approach, as well as its performance on other benchmark datasets.

Acknowledgments

This work was supported by grants #2016/17078-0 and #2018/04651-0, São Paulo Research Foundation (FAPESP). This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

References

Aggarwal, C. C. (2016). *Recommender Systems*. Springer, 1 edition.

Aggarwal, C. C. and Zhai, C. (2012). *Mining text data*. Springer Science & Business Media.

Almahairi, A., Kastner, K., Cho, K., and Courville, A. (2015). Learning distributed representations from reviews for collaborative filtering. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 147–154. ACM.

Baeza-Yates, R. and Ribeiro-Neto, B. (2008). *Modern Information Retrieval: The Concepts and Technology Behind Search*. Addison-Wesley Publishing Company, USA, 2nd edition.

Baral, R., Zhu, X., Iyengar, S., and Li, T. (2018). Reel: R eview aware explanation of location recommendation. In *Proceedings of the 26th Conference on User Modeling, Adaptation and Personalization*, pages 23–32. ACM.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of machine Learning research*, 3:993–1022.

Bobadilla, J., Ortega, F., Hernando, A., and Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, 46:109–132.

Guo, Q., Sun, Z., Zhang, J., Chen, Q., and Theng, Y.-L. (2017). Aspect-aware point-of-interest recommendation with geo-social influence. In *Adjunct Publication of the 25th Conference on User Modeling, Adaptation and Personalization*, pages 17–22. ACM.

He, X., Chen, T., Kan, M.-Y., and Chen, X. (2015). Trirank: Review-aware explainable recommendation by modeling aspects. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1661–1670. ACM.

McAuley, J. and Leskovec, J. (2013). Hidden factors and hidden topics: understanding rating dimensions with review text. In *Proceedings of the 7th ACM conference on Recommender systems*, pages 165–172. ACM.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Raghavan, S., Gunasekar, S., and Ghosh, J. (2012). Review quality aware collaborative filtering. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 123–130. ACM.

Ricci, F., Rokach, L., and Shapira, B. (2011). Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer.

Shah, L., Gaudani, H., and Balani, P. (2016). Survey on recommendation system. *International Journal of Computer Applications*, 137(7).

Sundermann, C. V., Domingues, M. A., Sinoara, R. A., Marcacini, R. M., and Rezende, S. O. (2019). Using opinion mining in context-aware recommender systems: A systematic review. *Information*, 10(2):42.

Wang, C., Paisley, J., and Blei, D. (2011). Online variational inference for the hierarchical dirichlet process. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 752–760.

Zheng, L., Noroozi, V., and Yu, P. S. (2017). Joint deep modeling of users and items using reviews for recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 425–434. ACM.