From Word Embeddings to Item Recommendation

Makbule Gulcin Ozsoy
Department of Computer Engineering
Middle East Technical University
Ankara, Turkey
Email: makbule.ozsoy@ceng.metu.edu.tr

***Abstract*—Social network platforms can archive data produced by their users and re-use the data to serve the users better. One of the services that these platforms provide is the recommendation service. Recommendation systems can predict the future preferences of the users using various different techniques. One of the most popular technique for recommendation is matrix-factorization, which uses low-rank approximation of input data. Similarly word embedding methods from natural language processing literature learn low-dimensional vector space representation of input elements. Noticing the similarities among word embedding and matrix factorization techniques and based on the previous works that apply techniques from text processing for recommendation, Word2Vec's skip-gram technique is employed to make recommendations. Unlike previous works that use Word2Vec for recommendation, non-textual features are used. The aim of this work is to make recommendation on next check-in venues and a Foursquare check-in dataset is used for this purpose. The results showed that use of vector space representations of items modelled by skip-gram technique is promising for making recommendations.**

***Keywords*—Recommendation systems, Location based social networks, Word embedding, Word2Vec, Skip-gram technique**

I. Introduction

The social network platforms (e.g. Twitter, Facebook, Foursquare) have many active users who produce vast amount of information by interacting with items/services and with other users on the platforms. For example, up to December 2015, 320 million monthly active users use Twitter, more than 55 million users use Foursquare and 1.01 billion daily active users use Facebook. These platforms are able to archive and use the produced information to better serve their users. One of the services that most of the social network platforms provide is the recommendation service.

Recommendation systems predict the future preferences of users' based on their previous interactions with the items. For example, information on previous Foursquare check-ins of users can be used to make recommendation on future check-ins. As already mentioned, there is vast amount of information on historical preferences of users. In the literature, this information is used in several different ways to make recommendation, e.g. by applying neighbourhood based, machine-learning based and matrix-factorization based methods. Recently, matrix factorization (MF) based approaches gained more attention by researchers, as these methods can efficiently deal with large datasets by using low-rank approximation of input data [15].

Similar to matrix factorization methods, word embedding methods learn low-dimensional vector space representation of input elements. They are used to learn linguistic regularities and semantic information from large text datasets and they are gaining more attention especially in natural language processing and text mining fields [19]. In this work, I aimed to use Word2Vec's [18] skip-gram word embedding technique for recommending next check-in locations.

Efficiency of using text processing techniques in recommendation systems is already exemplified in some of the previous works in the literature ([4], [22], [19]). [4] is one of the state-of-the-art methods for venue recommendation on Location Based Social Networks (LBSNs) and employs a language model based method. [22] aims to make recommendation to users about which blog to follow. It uses Word2Vec to model a word based feature, i;e. tags. [19] employs three different word embedding techniques, one of which is Word2Vec, to make recommendation on MovieLens and DBbook datasets. It uses textual data collected from Wikipedia about the items.

In this work, I employed Word2Vec's skip-gram technique to make recommendations on LBSNs. Unlike the previous works that use Word2Vec for recommendation ([22], [19]), I used a non-textual feature, namely the past check-ins of the users. For the evaluation I used a Foursquare check-in dataset, which is already used in previous works ([4], [20]).

The rest of the paper is structured as follows: I give information on the related work in the Section II. I explain Word2Vec and our proposed method in the Section III. I give the experimental results in the Section IV and I conclude the paper in the Section V.

II. Related Work

Recommendation systems make recommendation of items by estimating their preferences ([16], [23]). In the literature there are three base recommendation approaches: Content based, collaborative filtering and hybrid approaches. Content based approach uses item features and their similarities to make recommendations. Collaborative filtering approach uses past preferences of users to decide which items to recommend. Hybrid methods combine these approaches to make recommendations.

Besides the above-mentioned methods, matrix factorization based methods gain attention from recommendation systems researchers. These methods use low-rank approximation of input data and can handle large volume of data [15]. In [7], it is stated that matrix factorization can represent the items and the users as vectors where high correlation between vectors leads to recommendation. Also, in the same work

it is stated that these methods have good scalability, high accuracy and flexibility. Some example works that use the matrix factorization for recommendation belong to Ma et al. [14], Zheng et al. [27], Liu et al. [13], Cheng et al. [3] and [12]. Among these works [27] and [3] have similar purpose as ours and they make location/activity recommendations to the target users, however they do not employ Word2Vec for this purpose.

Similar to matrix factorization methods, word embedding methods from natural language processing field learn low-dimensional vector space representation of input elements. The word embeddings learn linguistic regularities and semantic information from the input text datasets and represent the the meaning of the words by a vector representation ([19], [1]). In [1] it is stated that word embeddings can be learnt by Latent Semantic Analysis (LSA), topic models and matrix factorization techniques. Techniques defined in Word2Vec [18], namely skip-gram and continuous bag of words (CBOW), are commonly used in the literature to represent the word vectors.

Some of the recommendation methods ([22], [19]) use techniques from Word2Vec to represent their text based features. [22] aims to make recommendation to users about which Tumblr blogs to follow. In this work inductive matrix completion (IMC) method is used for recommendation. This method uses side features (i.e. likes, re-blogs and tags) as well as previous preferences of users. It does not directly use techniques from natural language processing, but employ Word2Vec to compute vector representation of tags; which are word based features. [19] empirically evaluates three word embedding techniques, namely Latent Semantic Indexing, Random Indexing and Word2Vec, to make recommendation. They evaluate their proposed method on MovieLens and DBbook datasets. They mapped the items in the datasets to textual contents using Wikipedia and used the textual contents for the recommendation. Another recommendation method that uses techniques from natural language processing is Socio-Historical method proposed in [4]. It is one of the state-of-the-art methods for venue recommendation on Location Based Social Networks (LBSNs). Observing the similarities in text mining and social network datasets, it employs language models approach from natural language processing to make venue recommendations. It models users historical preferences and social interactions, which can be used separately or in combination.

Techniques in Word2Vec are generally considered as deep learning technique. There are few other methods that employ deep learning to make recommendations, e.g. [21], [5] and [24]. In [21] uses Restricted Boltzmann Machines (RBM's) to make movie recommendations. It models correlation among item ratings. [5] extends [21] by modelling both user-user and item-item correlations. [24] proposes a hierarchical Bayesian model thet learns models on both content informationon items and past preferences of users.

In this work, I employed Word2Vec's skip-gram technique to recommend check-in locations to the target users. Unlike the previous works that use Word2Vec for recommendation ([22], [19]), I used non-textual features, namely the past check-ins of the users.

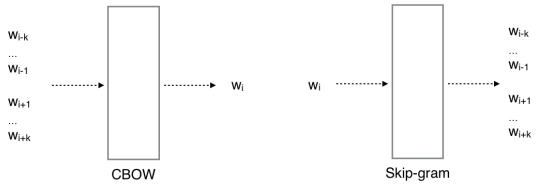

Fig. 1: Word2Vec techniques

III. Recommendation using Multiple Data Sources

Our aim is to list the top-k check-in locations (e.g. restaurant, cafe) that the target user will visit in the future. For this purpose I used a technique from Word2Vec toolbox, namely skip-gram. In this section, I give brief information on the techniques in Word2Vec toolbox and explain how skip-gram technique is used for check-in recommendation.

Word2Vec is a group of models which is introduced by Mikolov et al. ([18], [17]). It contains two different techniques, namely skip-grams and continuous bag of words (CBOW), which produce word embeddings, i.e. distributed word representations. The word embeddings represent the words in a low dimensional continuous space and carry the semantic and syntactic information of words [11]. While the CBOW technique uses the words around the current word to predict the current word, the skip-gram technique does the vice-versa, such that it uses the current word to predict the words around the current word (Figure 1). In both of the techniques, bag-of-words representation is used, i.e. order of the words in the input does not affect the result.

[10] states that CBOW combines words from the context window and cannot be easily expressed as a factorization. However, [9] shows that skip-gram performs a matrix factorization implicitly. The factorized word-context matrix is a co-occurence matrix that is known as pointwise mutual information (PMI) in the literature. Noticing the fact that both matrix factorization and Word2Vec techniques create low-rank approximation of the input data, I aimed to apply Word2Vec's skip-gram technique for recommending next check-in venues.

Our proposed recommendation method is composed of the following steps: First the input data is modelled using the skip-gram technique. Then this model is used to execute the recommendation process.

A. Modelling the input data using the skip-gram technique

The preferred technique to make recommendations is skip-gram technique: Firstly, use of text processing techniques for recommendation is already examplified in the llterature. Secondly, the skip-gram technique factorizes the input matrix implicitly and matrix factorization techniques are found to be effective in the recommendation systems literature. Lastly, skip-gram s preffered to CBOW, since it performs better than or equally well to CBOW technique ([18], [10])

I used the skip-gram technique implemented in the gensim

toolbox [1]. This implementation accepts a list of sentences which are theirselves are a list of words. These words are used to create the internal dictionary which holds the words and their frequncies. Afterwards the model is trained using the input data and the dictionary. The output of the technique is the word vectors, which can be used as features by different applications [25]. During the training various parameters can be tuned which affects the performance, in terms of time and quality. I detail how the paramters are tuned and the effects of different values in the evaluation section.

I noticed several similarities between the skip-gram technique and the recommendation process: First, the input data used in skip-gram technique is actually similar to what is used in the recommendation process. In the recommendation process I use a list of items that the user preferred/rated in the past and these lists can be divided into individual items. In other words, the sentences used in skip-gram can be mapped into past preferences of users in recommendation process and the words in skip-gram to individual items used in recommendation process. Second, the purpose of the skip-gram technique and the recommendation process are similar. Skip-gram model aims to predict to context words based on the current word, which can be mapped to predicting the items to be recommended based on already preffered/used items.

In the traditional recommendation process, the input data is composed of three base elements: user, item and rating. In most of the algorithms, these elements are represented by a *user* x *items* matrix, where the matrix entries indicate the ratings. For our check-in recommendation problem, the ratings are asssumed to be binary, such that the user is either checked in at a location(venue) or not. Then, each target user's past preferences can be represented as a list of items (the check-in venues).

Inspiring from [8], I used the item lists together with the users, i.e. not only items but all of the user and the items used by this user, as the input to skip-gram technique. As a result of the skip-gram, the vector representation of items and users are obtained, separately. These vector representations can be used to decide on which item is more similar to other item or which user is contextually closer to which items. These vectors and their similarities are used in the next step of our recommendation method.

B. Recommendation using vector representation

The skip-gram model provides the word vectors where the words with similar meaning are located closer in the vector space [8]. In the recommendation case, instead of words, there are items and users. The output of skip-gram provides the vector representation of the items and users in vector space where similar vectors are located closer to each other. In this report three different recommendation techniques that use the vector representation of items and users are proposed:

Recommendation by k-nearest items (KNI): In our recommendation by k-nearest items (KNI) approach, the similarity among user and item vectors are used. In this method, directly the most similar k items to the target user are found. For this purpose cosine similarity between the related vectors is used. The collected top-k items are recommended to the target user.

Recommendation by N-nearest users (NN): In recommendation by N-nearest users (NN) approach, the traditional user-based collaborative filtering method is applied on the vector representations. In the user-based collaborative filtering first the most similar users (neighbors) to the target users are selected, and then the items that are previously preffered by the neighbors are recommended to the target user. Similar to the traditional approach, in NN approach, first the top-N neighbors are decided using the similarity among the user vector representations. Then the items that are previously used/preffered by the top-N neighbors are collected. Summing up the votes of neighbors, the top-k items to recommend are decided. The collected top-k items are recommended to the target user.

Recommendation by N-nearest users and k-nearest items (KIU): This approach is a comabination of the previous two approaches. In this approach, first, the top-N neighbors are found by using the vector representation of the users. Then the top-k items that are most similar to the combination of target user and the neighbors are found by using the vector representations calculated in the first step. The collected top-k items are recommended to the target user.

IV. Evaluation

The aim is to recommend k-many check-in venues to each user based on their past check-ins. For this purpose the Checkin2011 dataset [2], previously used by [4] and [20], is used. The original dataset is collected from Foursquare website in between January 2011 - December 2011 and contains 11326 users, 187218 locations, 1385223 check-ins and 47164 friendship links. However, in [20] the researchers used a subset of this dataset by using the check-ins made in January as training set, and named it as CheckinsJan. The CheckinsJan dataset contains 8308 users, 49521 locations and 86375 check-ins. For the test step, the check-ins made in February are used and the target users are limited to the ones who checked in both January and February. The set of target users contains 7187 users. The evaluation results of the method presented in this report are compared to [4] and [20].

The performance is measured by Precision@k, Ndcg, Hi-trate and Coverage metrics. While giving the evaluation results, first, the performance metrics for each user are calculated separately and then their averages are presented.

Precision@k measures the relevance of items on the output list. the The Equations 1 presents how precision is calculated. In the equations, k indicates the output list size, tp indicates the true positives, i.e. recommended and actually used items, fp indicates the false positives, i.e. recommended but actually not used items.

$$Precision_k = \frac{tp_k}{tp_k + fp_k} \tag{1}$$

The Ndcg (Normalized discounted cumulative gain) metric decides the relevance of the listed items depending on their

[1] https://radimrehurek.com/gensim/models/word2vec.html

[2] http://www.public.asu.edu/ hgao16/dataset.html

rank. The Equations 2 and 3 presents how Ndcg (Normalized discounted cumulative gain) and Dcg (Discounted cumulative gain) are calculated, respectively. The Idcg (Ideal discounted cumulative gain) value refers to the best case where the output list is sorted by the relevance. In the equations, k is size of the output list, j is the position of the item on the output list and rel_j indicates if the item at rank j is relevant (true recommendation) or not. rel_j is 1.0 when the item at jth rank is relevant, and 0.0 otherwise.

$$Ndcg_k = \frac{Dcg_k}{Idcg_k} \tag{2}$$

$$Dcg_k = rel_1 + \sum_{j=2}^{k} \frac{rel_j}{log_2 j} \tag{3}$$

Hitrate measures the ratio of user who are given at least one true recommendation. In the Equation 4, m is one of the users, M is the total set of users, $|M|$ is the size of the users and $HitRate_m$ indicates if there is a hit for the target user m or not. $HitRate_m$ is equal to 1.0 if there is at least one true recommendation for that user and 0.0 otherwise.

$$HitRate = \frac{\sum_{m \in M} HitRate_m}{|M|} \tag{4}$$

Coverage measures the ratio of the users who are given any recommendation, independent from being relevant or not. In the recommendation systems literature, some of the methods may loose coverage in order to increase the accuracy [2]. [6] states that coverage and accuracy should be analysed together.

The methods presented in this report use two parameters: Number of neighbors (N) and output list size (k). [20] decided that best performing values are $N = 30$ and $k = 10$ for the CheckinsJan dataset, these values are used for the experiments. Also the upper-bound of the methods based on the decided parameters are presented in [20]: The upper-bound for Precision metric is found as 0.489 and for the rest of the metrics they are found as 1.0.

Several different settings are evaluated when the input data is modelled using skip-gram technique. The parameters used during the training affects the performance in terms of time and quality [26]. These parameters are based on the gensim toolbox implementation. I detail only the parameters that are set to a different value than the defult; for the rest of the parameters one can refer to gensim webpage. The details of the parameters and how they are tuned are as fallows:

- min_word_count: The technique ignores the items whose frequency is less than this parameter. Its default value is 5. In natural language processing, the items (words) that are seen only few times can be considered as garbage or typo, however in the recommendation systems the data is very sparse and having items that are observed only few times is normal. In order not to lose any item that is not used frequently, I set min_word_count parameter to 1.
- $size$: Size parameter represents the dimension of the feature vectors and its default value is 100. In [26] it is stated that bigger $size$ value can lead more accurate model, but requires more data. The suggested values for $size$ parameter is in tens to hundereds [26]. In our experiments this parameter is refferred as $feature\_count(F)$ and is set to different values in the range of $[10, 100]$ with 10 increments.
- $window$: Window parameter assigns the maximum distance between the current and the predicted words and its default value is 5. [26] states that it should be large enough to capture the semantic relationships among words. In our experiments this parameter is refferred as $context\_count(C)$ and is set to different values in the range of $[5, 20]$ with 5 increments.
- $iter$: Iter parameter represents the number of iterations on the input data and its default value is 1. In our experiments it is refferred as $epoch\_count(E)$ and is set to different values in the range of $[5, 25]$ with 5 increments.

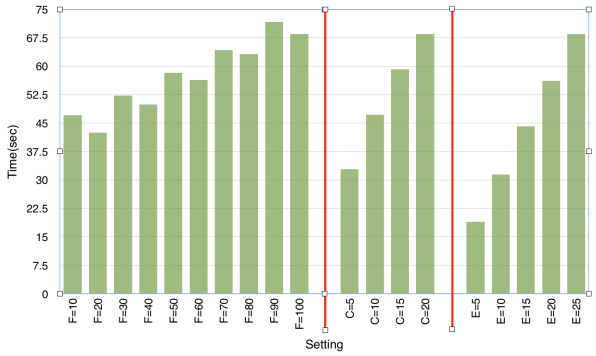

Fig. 2: The time needed to model the input data with skip-gram technique

While iterating on the different ranges of one parameter, the other parameters are fixed to a constant value. For example while iterating on $feature\_count$, the values of $context\_count$ and $epoch\_count$ are fixed. The constant values for the parameters are set to: $feature\_count(F) = 100$, $context\_count(C) = 20$ and $epoch\_count(E) = 25$.

The training time of the model for the CheckinsJan data is presented in the Figure 2. According to the figure, when the feature count increase, the time spent on the training increases from around 40 seconds to around 70 seconds. Similarly, for $context\_count$ and $epoch\_count$ the time spent on training increases as values of the parameters increase. Overall results show that training the model for CheckinsJan dataset takes less than 75 seconds.

Three different recommendation techniques that use the models trained by skip-gram are proposed. These techniques are 'recommendation by k-nearest items (KNI)', 'recommendation by N-nearest users (NN)' and 'recommendation by N-nearest users and k-nearest items (KIU)'. The time spent to make recommendation to all target users are presented on the Figure 3. The figure shows that the KNI method spends less time than other method, since it directly uses the output of the model without any further computations, such as finding

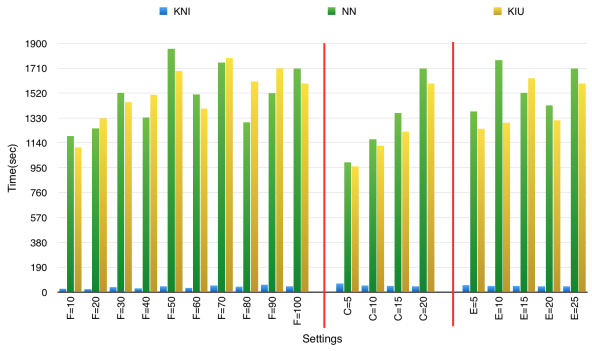

Fig. 3: The time needed to make recommendation using the model created by skip-gram

neighbors. Even the method and setting that spends the most time use less than 1900 seconds for all of the target users in CheckinsJan dataset, i.e. 7187 user. That means that the methods spend less than 0.26 seconds for each target user in the recommendation step.

In the the Figures 4 - 6, the recommendation performance of the proposed methods are shown. For all of the methods, increasing the $feature\_count$, i.e. the size of the vector representation, improves the performance. The effect of the increase for this parameter is less obvious as it is set to higher values than 50. When $context\_count(C) = 5$ the model is not able to capture the semantic similarity among the items and users. After setting this parameter to $C = 10$ and higher it performs better. For both $context\_count$ and $epoch\_count$, increase of the parameters slightly improves the performance.

According to the Figures 4 - 6, comparison of the performance of the recommendation techniques indicates that the best performing method is KNI, followed by KIU. Both of these methods use the vector representations of the items and their similarities to the target users. Both KNI and KIU uses similarity to item vectors, however KIU additionally uses the neighbors of the target users, which are decided by user vector similarities. Compared to KNI, the use of neighbors in KIU does not provide high performance gain. This may indicate that use of user vector similarities for CheckinsJan dataset is not effective. This may root from the fact that the users could not be differentiated in the vector space efficiently since the number of users in the dataset is not very high, i.e. only 8307 users.

In the Table I, I compare the proposed methods in this report, namely KNI, NN and KIU, to the methods from the literature. The first method from the recommendation literature is traditional collaborative filtering method (CF-C) which uses the past preferences of the users and their similarities. The next two methods are from [20], which are based on multi-objective optimization technique and combines past preferences of the users with other features, such as users' hometowns, friendship and their influence on each other. The methods are abbreviated as MO-CH when it uses past check-ins and hometowns of the users and MO-CFIH when it uses all of the above-mentioned features. The last two methods are from [4], which uses a language model based method from natural language processing

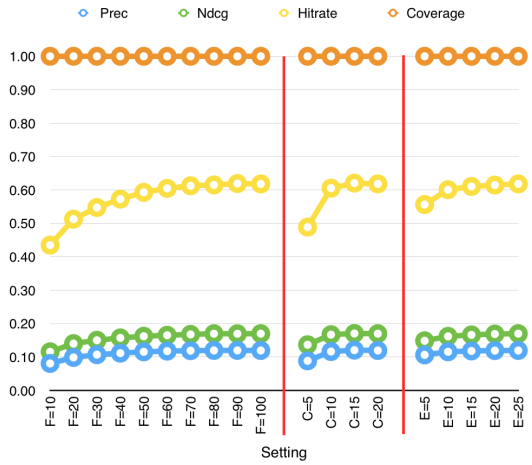

Fig. 4: Performance results of KNI

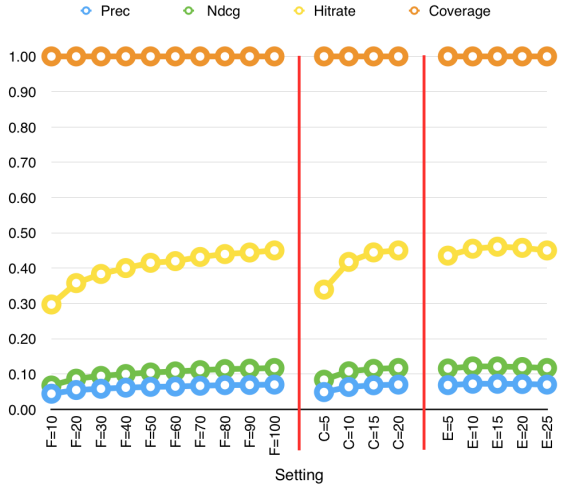

Fig. 5: Performance results of NN

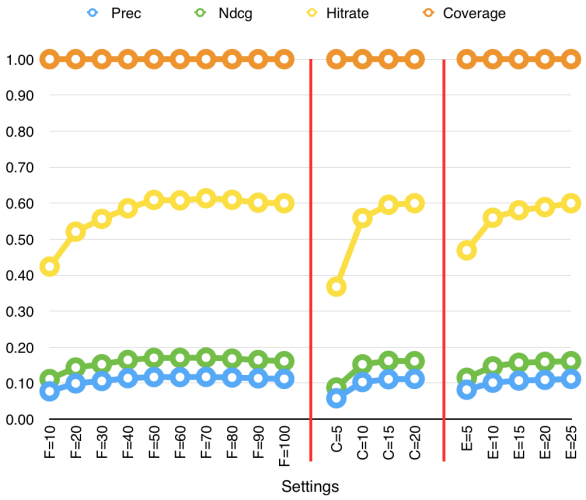

Fig. 6: Performance results of KIU

TABLE I: Comparison of methods

Method	Precision	Ndcg	HitRate	Coverage
KNI	0.119	0.169	0.618	**1.000**
NN	0.070	0.117	0.450	**1.000**
KIU	0.112	0.161	0.599	**1.000**
CF-C	0.114	0.242	0.621	0.955
MO-CFIH	0.105	0.218	0.596	0.999
MO-CH	0.112	0.227	0.616	0.996
Gao-H	**0.174**	**0.299**	0.696	0.952
Gao-SH	0.167	0.295	**0.721**	0.992

literature for recommendation. The method can combine past preferences and social ties of users. I used two versions of the methods, one of which uses only the past preferences of the users (Gao-H) and the other uses the combination of past preferences and social ties (Gao-SH).

According to the Table I, the methods proposed in this report, namely KNI, NN and KIU, are able to make recommendation to any users, i.e. having $Coverage = 1.0$. The best performing methods are Gao-H and Gao-SH, which shows that use of methods from natural language processing can be effective for making recommendations. In terms of Precision and Hitrate metrics, besides methods from Gao et al. [4], KNI method performs better compared to the other methods. For Ndcg metric, generally the methods that combine multiple features are better than the methods that use a single feature.

V. Conclusion

Recommendation systems predict the future preferences of users' based on their previous interactions with the items. In the literature, there are many different techniques to make recommendations, e.g. by applying neighbourhood based, machine-learning based and matrix-factorization based methods. One of the most popular methods is the matrix factorization based approaches which use low-rank approximation of input data. Similarly word embedding methods from natural language processing literature learn low-dimensional vector space representation of input elements.

Noticing the similarities among word embedding and matrix factorization techniques and inspring from the previous works that apply techniques from text processing for recommendation, Word2Vec's skip-gram techniqueis employed to make recommendations. In this work the aim is to recommend top-k venues to target users based on their past preferences. I used a dataset collected from Foursquare. The results showed that use of techniques from natural language processing is effective and use of skip-gram technique from Word2Vec is promising for making recommendations.

I want to apply the following improvements in the future: First, I want to use CBOW technique from Word2Vec for recommendation and compare its results to skip-gram technique. Second, I want to expand the dataset to observe the effect of data size. Third, I want to combine multiple features using the vector space representations, since experimental results showed that combining multiple features increases the recommendation performance.

References

[1] S. Arora, Y. Li, Y. Liang, T. Ma, and A. Risteski, "Random walks on context spaces: Towards an explanation of the mysteries of semantic word embeddings," *CoRR*, vol. abs/1502.03520, 2015.

[2] A. Bellogín, I. Cantador, F. Díez, P. Castells, and E. Chavarriaga, "An empirical comparison of social, collaborative filtering, and hybrid recommenders," *ACM TIST*, vol. 4, no. 1, p. 14, 2013.

[3] C. Cheng, H. Yang, M. R. Lyu, and I. King, "Where you like to go next: Successive point-of-interest recommendation," in *IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013*, 2013.

[4] H. Gao, J. Tang, and H. Liu, "Exploring social-historical ties on location-based social networks," in *Proceedings of the Sixth International Conference on Weblogs and Social Media, Dublin, Ireland, June 4-7, 2012*, 2012.

[5] K. Georgiev and P. Nakov, "A non-iid framework for collaborative filtering with restricted boltzmann machines," in *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, 2013, pp. 1148–1156.

[6] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. Riedl, "Evaluating collaborative filtering recommender systems," *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, 2004.

[7] Y. Koren, R. M. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *IEEE Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[8] Q. V. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, 2014, pp. 1188–1196.

[9] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 2014, pp. 2177–2185.

[10] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *TACL*, vol. 3, pp. 211–225, 2015.

[11] Y. Li, L. Xu, F. Tian, L. Jiang, X. Zhong, and E. Chen, "Word embedding revisited: A new representation learning and explicit matrix factorization perspective," in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, Buenos Aires, Argentina, July 25-31, 2015*, 2015, pp. 3650–3656.

[12] D. Lian, C. Zhao, X. Xie, G. Sun, E. Chen, and Y. Rui, "Geomf: Joint geographical modeling and matrix factorization for point-of-interest recommendation," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '14. New York, NY, USA: ACM, 2014, pp. 831–840.

[13] X. Liu and K. Aberer, "Soco: a social network aided context-aware recommender system," in *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, 2013, pp. 781–802.

[14] H. Ma, H. Yang, M. R. Lyu, and I. King, "Sorec: Social recommendation using probabilistic matrix factorization," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, ser. CIKM '08. New York, NY, USA: ACM, 2008, pp. 931–940.

[15] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, 2011, pp. 287–296.

[16] P. Massa and P. Avesani, "Trust-aware recommender systems," in *RecSys*, 2007, pp. 17–24.

[17] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *CoRR*, vol. abs/1301.3781, 2013.

[18] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, 2013, pp. 3111–3119.

[19] C. Musto, G. Semeraro, M. de Gemmis, and P. Lops, "Word embedding techniques for content-based recommender systems: An empirical evaluation." in *RecSys Posters*, ser. CEUR Workshop Proceedings, P. Castells, Ed., vol. 1441. CEUR-WS.org, 2015.

[20] M. G. Ozsoy, F. Polat, and R. Alhajj, "Multi-objective optimization based location and social network aware recommendation," in *10th IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom 2014, Miami, Florida, USA, October 22-25, 2014*, 2014, pp. 233–242.

[21] R. Salakhutdinov, A. Mnih, and G. E. Hinton, "Restricted boltzmann machines for collaborative filtering," in *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, 2007, pp. 791–798.

[22] D. Shin, S. Cetintas, and K.-C. Lee, "Recommending tumblr blogs to follow with inductive matrix completion." in *RecSys Posters*, ser. CEUR Workshop Proceedings, L. Chen and J. Mahmud, Eds., vol. 1247. CEUR-WS.org, 2014.

[23] M. Tavakolifard and K. C. Almeroth, "Social computing: an intersection of recommender systems, trust/reputation systems, and social networks," *IEEE Network*, vol. 26, no. 4, pp. 53–58, 2012.

[24] H. Wang, N. Wang, and D. Yeung, "Collaborative deep learning for recommender systems," *CoRR*, vol. abs/1409.2944, 2014.

[25] Word2Vec. Accessed: 2015-14-13. [Online]. Available: https://code.google.com/p/word2vec/

[26] Word2VecTutorial. Accessed: 2015-14-13. [Online]. Available: http://rare-technologies.com/word2vec-tutorial/

[27] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang, "Collaborative location and activity recommendations with GPS history data," in *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, 2010, pp. 1029–1038.