Embeddings Methods for Next-Item and Last-Basket Session-Based Recommendations

Michail Salampasis, Theodosios Siomos, Alkiviadis Katsalis, Konstantinos Diamantaras, and Dimitrios Tektonidis

***Abstract*—In this paper we use two industry datasets and apply two embedding methods and their combination for developing and testing session-based recommenders. The first dataset corresponds to an e-commerce scenario demanding effective next-item recommendation. The second dataset represents a last-basket prediction setting. Experimental results show that in the next-item task, the content-based approach utilizing textual descriptors of the items extracted by Doc2Vec and collaborative item embedding approach Item2Vec, which is trained upon item sequences, have comparable performance. When combined, they produce the best next-item predictor. In the last-basket recommendation scenario, Item2Vec significantly outperforms the Doc2Vec embedding method. Finally, we report on experiments with reranking methods that demonstrate the effectiveness of simple and practical methods, using item categories, to improve the recommendations.**

***Index Terms*—Session-based recommenders, embeddings methods, next-item recommendations, last-basket prediction.**

I. Introduction

In academic research, most works focused on making automatic predictions about the interests of a user by collecting and modelling long-term preferences from many users [1], hence approaches that utilize long-term user profiles in Recommenders' research are predominant. However, in many applications (e.g. e-commerce), such long-term user models are often not available because users are first-time visitors, or the application does not require visitors to login for privacy reasons.

Consequently, effective recommendations have to be determined based on raw log data (i.e. the user's most recent interactions with the site or application) or other types of information that can be calculated or inferred such as real-time prediction of shopper's intent [2], how did they land on the site, time spent on pages, short-term popularity trends in the community [3], web browsing behavior [4]. Recommendation techniques which rely on the user's recent behavior and other session-specific data, and which adapt their recommendations to the user's actions, are called session-based recommendation approaches [5]. An extreme approach of such recommendation methods is to rely only on the current item under examination and the community observed patterns (i.e. recommendations of the type "users who viewed/bought this item also viewed/bought this item"). Other session-based techniques were proposed in the literature that consider not only the very last item (behavior object) viewed by the user, but also more user actions (behavior types such as search, click, view-cart, share etc.) as well, hence analyzing the current session from its very beginning and while it progresses.

The previous paragraph very briefly outlined techniques on *how* to recommend. All techniques are based on the basic session-based recommendation setting which can be summarized as: given a session context (e.g. current item, previous items visited, time spent, landing page and other potential features) find those items or events that are most likely to happen as the next user action. The other important factor of every session-based recommendation task is *what* to recommend. If there are clear session boundaries, then two potential tasks are next-item(s) and next-basket recommendation, depending on whether recommended items are for the current ongoing session or for the next one respectively. If there is not a clear session boundary (e.g. in song listening or next-POI applications), recommender systems recommend the next event or action (e.g. suggest the next movie to be watched).

In this paper we briefly discuss the concept of sequential recommendation and relevant prior work. Then we compare several embedding modelling methods for session-based recommendations conducting several experiments on two industry datasets. The first dataset has log data from a medium size e-commerce web application and the second dataset contains numerous baskets of purchased items from a pet-shop store, where each basket encapsulates the items purchased by one user in a period of time. We use two datasets because we want to test the methods under examination in the context of two basic tasks: next-item and next-basket recommendation. In next-item recommendation, each behavior (i.e. user action) involves only one item (e.g. a product, song, movie, or a location). In contrast, in next-basket recommendation, a behavior contains more than one object. However, despite the inputs they receive may be different, both tasks strive to predict the next item(s) for a session user, whilst the most popular output format is a top-N ranked item list usually determined by probabilities or a similarity function.

Distributed representation methods have attracted some interest in the last five years for developing recommender systems [6]-[8]. The key idea behind these methods is viewing items as words, and users' sessions as sentences. However, the study of embeddings methods for recommendation were overshadowed by the phenomenal

Manuscript received September 10, 2020; revised April 17, 2021. This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program of Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH/CREATE/ INNOVATE (project code: T1EDK-01776)

All authors are with the International Hellenic University, Thessaloniki, Greece (e-mail of the corresponding author: msa@ihu.gr).

doi: 10.18178/ijmlc.2022.12.4.1092

interest for RNN methods to recommender systems. We believe that these methods are worth further examination in the context of session-based recommendation, to investigate how they will perform under different settings. Especially, in the light of recent findings [1] that, in session-based recommendations, even simpler algorithms, often perform equally well when compared to more complex approaches based on deep neural networks, therefore there is space for improvement regarding the development of more sophisticated session-based recommendation algorithms. In that context, we wished to determine if combining embeddings modelling both content and item sequences for recommendation methods is feasible and effective.

The first method we tested is solely content-based and uses pre-processed text descriptors of each product (e.g. name, description, color) to calculate item vectors using the Doc2Vec model [9]. These vectors are later used to recommend the next item(s) given the current item and a similarity function. The second method is a variation of the Item2Vec method for item-based Collaborative Filtering (CF) [6]. Item2Vec is a pretty straightforward application of the Word2Vec [10] linguistic method in the context of CF. Item2Vec creates vectors of items using item sequences generated during a session, as input to learn items distributed semantic embeddings, in the same way as Word2Vec does for words in sentences. These embeddings are later utilized to infer item-to-item relations and provide session-based recommendations. Additionally, a framework is presented and tested on how to combine the Doc2Vec and Item2Vec methods, creating a hybrid method that utilizes both content and sequence patterns between items. Finally, we report on and discuss post-prediction re-ranking algorithms that are applied into the previous methods to further improve the recommendations that are initially produced.

The rest of the paper is organized in the following manner. Section II reports on prior work. In Section III we discuss the embeddings-based recommendation methods that we extended and tested. In Section IV we describe the setup of the experiments conducted, and we report and discuss the results obtained in both datasets. Section V concludes the paper, summarizing the findings and presenting future ideas for development.

II. Prior Work

Early recommendation methods to predict the next user actions were based on sequential pattern mining techniques using relatively simple techniques such as item-based collaborative filtering [11] or content-based methods [12]. The main advantage of techniques based on frequent patterns analysis is that they are easy to implement and lead to straightforward interpretable models. However, the mining process can be computationally demanding. At the same time, finding good algorithm parameters can be challenging. Moreover, in some application domains it seems that using frequent item sequences does not lead to better recommendations than when using simpler item co-occurrence patterns [13].

After these simpler sequential pattern mining techniques, more sophisticated methods based on Context Trees [14], Markov models [15], Reinforcement Learning [16] and Markov Decision Processes [17] were tested. Typical application domains of these methods are e-commerce, music/movie streaming services, web search analysis and several others. A parameter that should be tuned for this type of recommender models was how many of the previous interactions (i.e. short history) should be considered when predicting the next one. Possible solutions that were tested were to use a mixture of Variable-order Markov Models (VMMs) or context-trees to consider sequences of different lengths.

After the success of distributed representations of words and phrases (Word2Vec) that capture precise syntactic and semantic word relationships, recommender methods based on these methods that were originally devised for linguistic tasks, have been tested for CF. This group of methods labelled as Item2vec is capable of inferring item-item relations even when user information is not available. Word2Vec is the name for a pair of related models that are used to produce word embeddings and attempt to map words to a relatively low dimensional vector space that captures semantic relations between words. Doc2Vec is an unsupervised algorithm to generate vectors for sentences/paragraphs/documents. The algorithm is an adaptation of Word2Vec that generates vectors for words. The vectors generated by Doc2Vec can be used for tasks like finding similarity between sentences/paragraphs/documents.

Unlike sequence models such as RNN, where word sequence is captured in generated sentence vectors, Doc2Vec sentence vectors are word order independent. Similarly, the item-to-item (Item2vec) recommender system first is trained with the item sequences from previous site sessions. Then, in running mode, Item2Vec takes the current item as input, and outputs a set of similar items given the input. In order to select the candidates, similarity between items is calculated and the items are ranked from highest to lowest similarity score. Top-n items with the highest scores are recommended as the next item(s). In fact, several research works have shown that item-based CF produces competitive results when compared with SVD and other sequence-based CF methods [6], [7].

Finally, in the most recent years, the use of deep learning approaches based on artificial neural networks was another solution to the recommendation problem. Among these approaches, Recurrent Neural Networks (RNN), have been shown to be excellent models for sequential data and for data that is generated by users in a session-based manner. The main advantage of RNNs in comparison to traditional similarity-based methods for recommendation, is that they can dynamically model the entire session of user interactions (i.e. all types of events such as clicks, item views, etc.) in a very natural and effective way [18], [19].

III. Embeddings Methods for Session-Based Recommendations

A. Datasets

The raw data for the experiments related to the next-item task are taken from the web server logs of an e-commerce application (leather apparels) for a relatively long period of time (six months). The log data were analyzed to identify sessions, session length, user actions in each session, actions' related items, item categories, and time spent in each action.

The user-agent cookie of each log line is used to identify unique sessions. Each sequence identified by a unique cookie is a user's session and consists of the actions that the user has completed during a session. The above dataset is preprocessed to obtain only the sessions that contain at least 2 behavior sequences. As a result of this preprocess the dataset consists of 24111 sessions that altogether count 312912 user actions. The complete set of action types were retrieved for website user behavior analysis (click-stream analysis), but this work will be reported on another paper. The 728 sessions that ended in purchases means a conversion rate of 3%. In 91.2% of sessions (i.e., 22.008 sessions), users did not have any items in their shopping cart when they exited, which implies sessions that were pure browsing only. The rest of the sessions had items in their shopping cart when they finished, but never turned into purchases. Twelve different action types were identified out of the log files. For the next-item experiment which is presented in this paper, only the item-related user actions were considered (i.e. View Product) and only the sessions that include at least two different item views, finally pertaining to a number of 12128 applicable sessions consisting of 67101 user View Product actions. Fig. 1 shows the distribution of sessions' length for this subset.

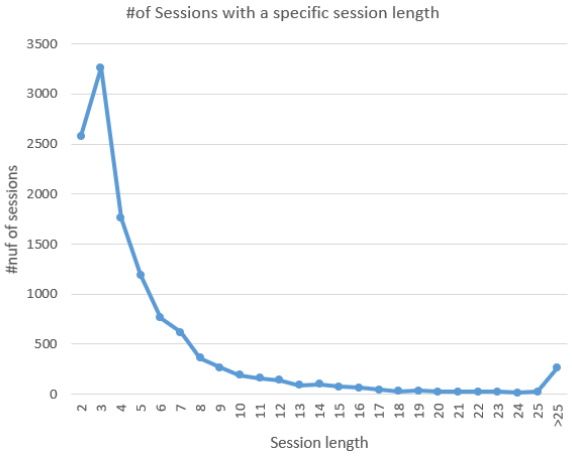

Fig. 1. Distribution of sessions' length (Next-item task/dataset).

The second testbed which implements the next-basket scenario is built also on a real-world dataset which contains numerous baskets of purchased items from a pet-shop store, where each basket encapsulates the items purchased by one user in a period of time. This is an industry dataset which contains 40203 transactions (baskets) belonging to 1493 users of which 1408 have more than one basket so they can be included in the test set. It contains 6626 items. The average length of all baskets is 2.26 (Fig. 2 shows the distribution of basket length in the dataset), therefore in all experiments that we report in this paper for the next-basket task, we set 2 as the cut-off level in our criterion (i.e. F1@2) in our results. Another parameter that we tested was the number of baskets (i.e. the history of this user purchases) that were considered in predicting the last-basket with a short history (1-7 previous purchases) performing better rather than considering the entire user baskets sequence.

B. Content-Based Embeddings: Doc2Vec

Our implementation of this method uses the content information that is associated with each item in the dataset. Specifically, name, color, description and several existing metadata (size, type and accessories of a product) were used to create vectors for each item. The Doc2Vec model has been used to extract the semantics, grammar and word order of the textual descriptions and transform it into a fixed dimension vector for each item. Subsequently, the similarity between these item vectors is applied to recommend other products. Standard pre-processing linguistic steps such as tokenization and stop-word removal have been applied, before feeding the textual descriptions to the training model. After training the Doc2Vec model with then remaining textual data, we obtained an n-dimensional vector for each item. After several tests, 500 dimensions were selected as the most effective parameter for the next-item dataset and 100 for the last-basket dataset. In the experiments reported in this paper we used both cosine and L2 norm similarity measures to compute the inter-item similarities. The cosine similarity measure produced slightly better results in both datasets.

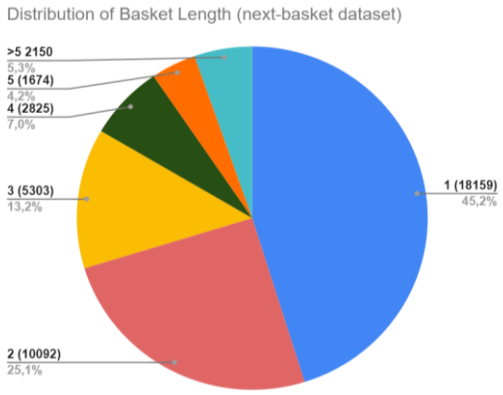

Fig. 2. Distribution of baskets' length (Last-basket task/dataset).

C. Collaborative Item Embeddings: Item2Vec

This method is an implementation of the Word2Vec algorithm that is used in the field of Natural Language Processing (NLP) to learn word associations from a large corpus of text. All these vectors belong to the same vector space. Thus, words with common content and semantic similarity have a small distance between their vectors in the vector space. In an analogous way, instead of assigning embeddings to words, Item2Vec [6] assigns embeddings to items. The underlying idea is that it can be used for collaborative filtering in order to provide recommendations even when user information is not available. The items used in the Item2Vec correspond to the sequence of words that are processed by Word2Vec and the users' sessions browsing an online store correspond to word sequences (sentences). Word2Vec uses two different model architectures to produce a distributed representation of words, namely, continuous bag-of-words (CBOW) and continuous skip-gram. With the first model, the context of a word (i.e. neighboring words) is used to predict a target word while in the latter a word is used to predict a target context (i.e. the neighboring words). Our implementation uses the skip-gram flavor in order to be trained.

The most effective size of embeddings for this method was 30 for the next-item dataset and 100 for the next-basket dataset. Better results are achieved when L2 norm is used as the chosen measure in the first dataset whereas cosine

similarity produces better results in the second dataset.

D. Embeddings Utilization

After the training of the item embeddings using either the Item2Vec or Doc2Vec methods, the steps that we applied -using each implementation separately- to produce the item recommendations are:

1) Fetch the embeddings of the last n items the user has visited in the ongoing session (or purchased in previous n baskets). Note that the value of n is an important parameter that affects the performance of the recommenders using embeddings therefore it is more extensively discussed later.
2) Compute the average vector u for these embeddings.
3) Compute the cosine similarity or the L2 norm between the vector u and all the item embeddings that are available in the dataset.
4) Sort the items based on the cosine similarity or the L2 norm (in descending and ascending order respectively) and produce the predictions table.
5) Select the top k most similar items to vector u which constitute the recommendations to the user.

E. Combining Item2Vec and Doc2Vec

Hybrid recommendation systems that combine both collaborative filtering and content-based methods are not unusual and were applied quite successfully [20]. Therefore, in our work we examine how this concept of hybrid utilization of text content and item sequences can be applied using embeddings methods. Our implementation of this "use both worlds" concept utilizes both the Item2Vec and the Doc2Vec methods, thus combining the pattern mining preferences of all sessions (community observed patterns) with the recommendation model which is based on textual/word vector features. The methodology that we formulated to combine the two methods is summarized in the next paragraph.

First, we acquire the predictions from each method separately. Each method produces its own predictions based on its corresponding vector space. One vector space refers to the vectors produced by the Item2Vec method and the other by the Doc2Vec method. The predictions are made based on the cosine similarity for both the methods. The value of the cosine similarity for each prediction shows how strong the prediction is. The range of cosine similarity is [0, 1]. The closer the value is to 1, the stronger the prediction is. In our method, we use v=[1-CosineSimilarity] to make the predictions. In this case, the range of the values remains the same (i.e. [0, 1]) but now, the closer the value is to 0, the stronger the prediction is.

For each method, we multiply the value of v of each prediction with the position of the item in the predictions table. This product depicts the confidence at which the corresponding method makes the specific recommendation. Then, we add the Item2Vec and Doc2Vec confidences for each corresponding product. Finally, we sort the products based on the summation of confidences in ascending order. In this way, we exploit the predictions with high confidence from each method. As we will discuss in the next Section, this fusion method improved the results in the next-item task -in comparison to the methods used independently- but it didn't have any positive effect on the next-basket recommendation task.

F. History Window Size

As we outlined in the Embeddings utilization sub-Section, the vector u is computed by averaging the embeddings of either the last n items the user has visited in an ongoing session or of all the items he has purchased in the n previous baskets. After several experiments, we found out that the better results are achieved when only the most recent items are averaged, i.e. the most recent behavior is significantly more influential regarding the next action, than the behavior at the beginning of the session. All the methods described in the preceding paragraphs perform better when a small history window size is taken. Specifically, in the first dataset, the best results are achieved when we take into consideration only the last item a user has visited whereas in the last-basket task best results are produced when we take into consideration the items of the (maximum) last 7 baskets.

IV. Experiments and Results

Before presenting the results of each method presented in Section III, we clarify some details about the experimental setup. More details to support the reproducibility of the experiment can be found at https://islab.iee.ihu.gr/. First, the datasets are those we discussed in the Dataset paragraph (Section III). In the first dataset we kept only the actions that correspond to view-item user actions. When running our experiments, for each test session we iterate the viewed items sequence and for every viewed item in the sequence we produce a different set of recommended items using all methods. If a method uses the entire sequence of viewed item(s) from the beginning of the session, then this sequence is provided to the method.

After obtaining a ranked list of n recommended item(s) we calculate the Reciprocal Rank for each item visited from the session sequence. When all items are considered (except the first one of each session), we calculate the Mean Reciprocal Rank (MRR) of the entire session and then again, we average the MRR of all tested sessions for each method as a whole. This is how we calculate the figures that are reported in this Section. MRR is a statistic measure for evaluating any process that produces a list of possible responses to a sample of recommendations. The reason we choose MRR as the evaluation measure is because it expresses the effectiveness to recommend the next-item as higher as possible in the recommendation list. The underlying assumption is that if a method attains an MRR between 0.20 to 0.25, then a system would require a list of 4-5 recommended items to effectively predict the next viewed item.

In the next-item task experiments, 90% of the dataset was used for the train set (60815 samples) and the remaining 10% for the test set (6286 samples). To test the validity of the models, where applicable, a random split was repeated 5 times to the set of all session ids and for each split 90% of the sessions were used for training and the remaining 10% for testing. The results reported are the mean of the results produced from each repetition. In the last-basket task all available purchases (baskets), were used for training and test purposes. Specifically, all purchase sequences except, the last

one, were used to train the prediction models. Finally, the last-basket of each user purchase (basket) sequence was predicted.

The post-prediction re-ranking method was based on the intuition that most of the sessions that users perform are focused on a few product categories. However, there are many ways such approach can be implemented. We preferred one that is biased towards the items belonging in the dominant category, as a result of observations favoring such bias. The precise method we applied this reranking process is explained in the next paragraph.

Every product in the store belongs to one or several product categories. To use categories for re-ranking, first we calculate the dominant category of all the products that the user has visited. The "dominant" category is defined as the category with the most "hits" from the beginning of the session until the current item. When this dominant category is computed (and this very simple computation is repeated after each user action), then the items that are recommended by each core method (e.g. Item2Vec), and are members of the dominant category, are re-ranked towards the top positions, shifting down all the other items that have been predicted higher but they do not belong to the dominant category. This general idea of reranking was also driven by the need to test how business logic requirements (e.g. promote products of specific categories) that would normally be very useful to the ecommerce store manager to implement marketing strategies, would affect the performance of a recommender. In the table summarizing all results, we present this condition marked as "with reranking".

Table I illustrates the result each method has produced for the next-item (MRR, columns 2 and 3) and the last-basket (F@2, columns 4 and 5) recommendation task. For the next-item task, the column indicated as "last" presents the results when only the last item was considered as the current context. The column marked as "all" presents the results when the entire user behavior sequence, from the beginning of the session until the currently viewed item, is considered.

TABLE I: MRR RESULTS OF ALL RECOMMENDER METHODS

	Next-item Recommendation task (ecommerce site)		Last-basket recommendation task (Almapet)	
Method	MRR/last	MRR/all	F1@2/7	F1@2 /all
Doc2Vec	0.101	0.062	0.154	0.114
Doc2Vec + reranking	0.123	0.079	0.143	0.105
Item2Vec	0.087	0.079	**0.221**	**0.167**
Item2Vec + reranking	0.111	**0.093**	0.182	0.148
Fusion (Doc/Item2Vec)	0.112	0.078	0.216	**0.167**
Fusion + reranking	**0.126**	0.089	0.184	0.151

Table I shows that the Doc2Vec method relying on items' textual representations is the best method from the two core embeddings methods (Doc2Vec/Item2Vec) that we tested attaining an MRR of 0.101 while Item2Vec achieves an MRR of 0.087. The combination (Fusion) of the content-based and the item sequence embeddings produced better results at 0.112. Finally, for the relatively static nature of the embedding methods, our experiments confirmed the expectation that category driven re-ranking will improve the results attaining the best overall MRR of 0.126.

Table I illustrates also the results of the last-basket prediction task. The first column presents the F1@2 results when the last 7 baskets are considered as history purchases. The second column (marked as all) illustrates the results when all purchases are applied as previous baskets. In this set of experiments the results are quite different when the same methods were used in the next-item task. First, the Item2Vec method performs significantly better than the Doc2Vec method. We believe this is mostly due to the very limited textual information which is available in the items of this dataset. Conversely to what was observed in the next-item task, the Fusion method in this dataset does not perform better than the individual methods. In fact, it performs slightly worse than Item2Vec. We believe that this is an indication that Fusing has potential only if all contributing methods have a reasonably good performance and, most importantly if all fused methods capture useful information that are not modeled by the other methods processed by the fusing process. In this specific task, textual representations of items in the dataset were very small, therefore adding Doc2Vec vectors does not add value in the overall prediction task.

Another interesting comment in the last-basket recommendation task is that the best results are achieved when methods use a shorter basket history (history window size=7). Generally, all methods produce worse results when the entire purchase history is used. It seems that a principle of locality holds which states that a user action/behavior is directly influenced more intensively by its immediate surroundings. Finally, another interesting finding is that when reranking is applied all methods have worse performance, something which we believe should be attributed to the dataset organization that has very few categories but also because the task is heavily precision-oriented (i.e. only two items are sought in the last-basket task).

V. CONCLUSIONS

Our work on comparing embedding methods for next-item and last-basket session-based recommendations was mainly driven by the less attention to that type of recommenders and the lack of systematic comparison in the session-based recommendation literature. Perhaps, this is because LSTMs architectures and methods have almost monopolized the interest for session-based recommenders in the last few years. In our work, we wished to determine if embeddings methods could also be used as an effective method for session-based recommenders. In particular, we wished to determine whether the specific task (next-item or last-basket) is an important factor when selecting an embeddings method for recommendations. Furthermore, we explored other important parameters (e.g. type of textual representations, history size, training parameters, re-ranking, similarity measures) that should be also considered when developing practical applications.

We believe that the experiments we present here provide useful insights into the utilization of embeddings methods for session-based recommenders in different settings. These insights can be summarized as follows:

1) In the next-item task where the user's interest is probably more focused around a product category and perhaps even to similar and/or related products, the items' textual

descriptions could be used to build a good estimator of the next-item to recommend. So, if there are rich textual descriptions, and the task is more focused around a specific category and product kind, probably Doc2Vec is the best recommendation method.

2) Item2Vec will perform better if the session involves more products that are not necessarily related, in terms of their textual information, and clearly this description resembles closer to the next-basket task. Of course, if there is very little textual data to use, then Item2Vec is the only feasible option.
3) Fusing different methods such as content-based (Doc2Vec) and item-based embeddings (Item2Vec) has not always a positive effect on the results, it largely depends on whether textual data (Doc2Vec) can contribute an extra value to item-based recommendations. Generally, this observation may be the starting point of an idea for a personalized and pluggable framework that could decide *at real-time* the best method to make a recommendation (e.g. Item2vec, Doc2Vec, Fuse). In fact, we believe, based on the work and results that we report in this paper, that a model can be trained to dynamically decide a recommender method as the session unfolds step by step. This is an area that we aim to explore further.
4) If recommenders have poor performance, practical methods such as category-based re-ranking will be beneficial, but only if the underlying assumption is valid and the items are sufficiently organized.

From the relevant literature we know that LSTMs produce very good results when applied in session-based recommendations. Their dynamic nature, which inherently models the behavior of explorative browsing that usually occurs in e-commerce applications, it may be better suited for this type of predictions. This is something that we plan to explore further in the specific datasets and applications. We plan to apply these deep learning methods in both datasets and recommendation tasks and examine their performance.

In conclusion then we feel that in this paper we have already presented some novel knowledge and useful results which will help engineers to use embeddings methods for producing effective recommender components for ecommerce and other session-based systems.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Authors A and E contributed to the preparation of data. Authors B, C and D participated in running the experiments. Author A wrote the paper. Authors A-D contributed to the analysis of the results. All authors had approved the final version.

References

[1] L. Malte and J. Dietmar, "Evaluation of session-based recommendation algorithms," *User Modeling and User-Adapted Interaction*, vol. 28, pp. 331-390, May 2018.

[2] C. O. Sakar, S. O. Polat, M. Katircioglu *et al*., "Real-time prediction of online shoppers' purchasing intention using multilayer perceptron and LSTM recurrent neural networks," *Neural Computing & Applications*, vol. 31, pp. 6893-6908, 2019.

[3] J. Dietmar, L. Malte, and L. Lerche, "Session-based item recommendation in e-commerce: On short-term intents, reminders, trends, and discounts," *User Modeling and User-Adapted Interaction*, vol. 27, pp. 351-392, 2017.

[4] M. A. Awad and I. Khalil, "Prediction of user's web-browsing behavior: Application of Markov model," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 4, pp. 1131-1142, Aug. 2012.

[5] M. Quadrana, P. Cremonesi, and J. Dietmar, "Sequence-aware recommender systems," *Computing Surveys*, vol. 54, pp. 1-36, 2018.

[6] O. Barkan and N. Koenigstein, "ITEM2VEC: Neural item embedding for collaborative filtering," in *Proc. IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP)*, Vietri sul Mare, 2016, pp. 1-6.

[7] V. Phi, L. Chen, and Y. Hirate, "Distributed representation-based recommender systems in E-commerce," in *Proc. DEIM Forum*, 2016.

[8] Z. Yang, J. He, and S. He, "A Collaborative filtering method based on forgetting theory and neural item embedding," in *Proc. IEEE 8th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*, 2019, pp. 1606-1610.

[9] L. Quoc and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. International Conference on Machine Learning*, 2014, pp. 1188-1196.

[10] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th International Conference on Neural Information Processing Systems*, 2013, pp. 3111-3119.

[11] J. B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative filtering recommender systems," *The Adaptive Web*, Springer, pp. 291-324, 2007.

[12] C. C. Aggarwal, "Content-based recommender systems," in *Recommender Systems*, Springer, pp. 139-166, 2016.

[13] B. Geoffray and J. Dietmar, "Automated generation of music playlists: Survey and experiments," *Computing Surveys*, vol. 47, no. 2, pp. 1-35, Nov. 2014.

[14] F. Garcin, C. Dimitrakakis, and B. Faltings, "Personalized news recommendation with context trees," in *Proc. the 7th ACM Conference on Recommender Systems*, 2013, pp. 105-112.

[15] M. Aghdam, N. Hariri, B. Mobasher, and R. Burke, "Adapting recommendations to contextual changes using hierarchical hidden Markov models," in *Proc. the 9th ACM Conference on Recommender Systems (RecSys '15)*, 2015, pp. 241-244.

[16] O. Moling, L. Baltrunas, and F. Ricci, "Optimal radio channel recommendations with explicit and implicit feedback," in *Proc. the sixth ACM conference on Recommender systems (RecSys '12)*, 2012, pp. 75-82.

[17] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-based recommender system," *Journal of Machine Learning Research*, vol. 6, pp. 1265-1295, 2005.

[18] R. Devooght and H. Bersini. (2017). *Collaborative Filtering with Recurrent Neural Networks*. [Online]. Available: https://arxiv.org/abs/1608.07400

[19] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D Tikk. (2016). *Session-based Recommendations with Recurrent Neural Networks*. [Online]. Available: https://arxiv.org/abs/1511.06939

[20] G. Liu and X. Wu, "Using collaborative filtering algorithms combined with Doc2Vec for movie recommendation," in *Proc. IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, 2019, pp. 1461-1464.

Michail Salampasis has a BSc in informatics and a PhD in computing. He is a full professor at the Information and Electronic Engineering Department of the International Hellenic University, Greece. His main research interests are in applied and interdisciplinary studies in information science, including models and experiments related to information-seeking behavior, information seeking

in large professional search systems, patent search, distributed information retrieval and systems evaluation. He has published more than 90 papers in refereed journals and international conferences. Also, he has large experience in systems development and has been the coordinator and lead scientist in many research & development projects. He has been a Marie Curie Fellow at the Institute of Software Technology and Interactive Systems, Vienna University of Technology.

Theodosios Siomos is a machine learning engineer. He holds a BSc in applied informatics and an MSc in data science. He currently works as a research assistant at the International Hellenic University in the field of recommender systems. His research interests are mainly focused on machine learning methods and parallel and distributed processing.

Alkiviadis Katsalis holds a BSc in mathematics and an MSc in data science. He worked as a data analyst assistant at Deloitte and now he is a natural language processing researcher. He currently is a PhD candidate in the area of NLP and NLG and more specifically his research interests focus on automatic text summarization and text generation.

Konstantinos Diamantaras received the diploma degree from the National Technical University of Athens, Greece, in 1987 and the Ph.D. degree in electrical engineering from Princeton University, Princeton, NJ, USA, in 1992. He is currently a professor at the Department of Information and Electronics Engineering, International Hellenic University, Greece. His research interests include machine learning, signal / image processing and parallel computing. Dr. Diamantaras has served as the chairman to the Machine Learning for Signal Processing (MLSP) Technical Committee (TC) of the IEEE Signal Processing Society and a member of the MLSP and Signal Processing Theory and Methods TCs as well. He has been the Chairman and a member of the TC for various machine learning, signal processing, and neural networks conferences. He has served as an associate editor for the IEEE Transactions on Signal Processing, the IEEE Signal Processing Letters, and the IEEE Transactions on Neural Networks.

Dimitrios Tektonidis holds a BSc and a PhD in computing. He has participated in more than 15 ICT research projects and has 20 years' experience in IT system development. His main fields of research are enterprise application integration (EAI), IS interoperability, service oriented architectures (SOA), big data and cloud computing.