2021 IEEE 23rd Int Conf on High Performance Computing & Communications; 7th Int Conf on Data Science & Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application

A Filter is Better Than None: Improving Deep Learning-Based Product Recommendation Models by Using a User Preference Filter

Miguel Alves Gomes*, Hasan Tercan*, Todd Bodnar†, Philipp Meisen† and Tobias Meisen*
*Chair for Technologies and Management of Digital Transformation, University of Wuppertal, Wuppertal, Germany
Email: {alvesgomes, tercan, meisen}@uni-wuppertal.de
†Breinify Inc., San Francisco, USA
Email: {todd.bodnar, philipp.meisen}@breinify.com

***Abstract*—Nowadays, recommendation systems are a central component in a wide range of online services. One major challenge for them is to generate suitable recommendations for individual users who differ, among other things, in their interests and preferences. In this paper, we propose a deep learning-based product-to-product recommendation system that uses product embeddings to incorporate content-based product information and user data in form of historical checkouts. On top of that, we add a filter that rearranges the models' recommendations based on the individual user preferences to a product category that is derived from past activities. In order to evaluate and show the impact of our approach, we conduct tests on historical data and an A/B-test in a real online service that offers thousands of different customer packaged goods (CPG) to millions of users. The results show that our approach delivers suitable product recommendations to users and outperforms the current system in use.**

I. Introduction

In our modern world it is possible to purchase nearly everything online. Due to the Covid-19 pandemic, the sales numbers of e-commerce businesses and websites have increased even more over the past year [1]. As the number of users and product portfolios continue to grow, recommendation systems are also playing an increasingly important role in such systems. Traditionally, recommender systems are developed to find frequently occurring user-product relationships and to recommend interesting products which a user may not even know to search for. Thereby, major challenges are to include personal user preferences into the recommendation and to deal with a vast amount of different products and user activity data.

Recommender systems are often based on statistics from historical user interactions such as product purchases. Often used approaches are based on collaborative filtering methods that consider similarities of products and user activities [2]. More recently, the research focuses on deep learning-based approaches trained on historical user and transaction data [3]. One popular approach in this area is the use of product embeddings (also known as prod2vec) [4], [5]. Product embeddings are a vector representation of products where the distance of two products in the vector space represents their similarity. With a nearest neighbor approach, it is possible to compute product similarities out of the product vectors and generate recommendations.

One drawback of such approaches is that they are designed to address the majority of the users and do not take into account user-specific information. Since they are solely based on the statistical behavior of all users, they may not cover the needs and preferences of the individual users. Another point to consider when making a recommendation is the company's intention to maximize its profit. The goal of recommendation should therefore be to recommend not only products that are interesting to the user but also products that are likely to maximize profits.

In this paper, we address these issues in a real world product-to-product recommendation problem for an online retailing service with several thousands of products and millions of user activities per week. Our proposed approach is based on product embeddings that encodes both user purchase histories and product categories. Based on the derived recommendations from the product embeddings, we apply a filter in a second step to further increase the result quality of the recommendation. The filter makes use of the users' product category preferences that are calculated on historical user activities.

In order to make a reliable and valid evaluation, we conduct an A/B-test in which we compare two different variants of our recommender system together with a baseline recommender that is already in use. The A/B test is conducted as part of several email campaigns, each of which includes more than three million customers. We evaluate the models by comparing their click-through rates (CTR) and subsequent real purchases of the users. We show that using a filter on top of the product embedding improves the performances.

The paper is structured as follows. In the next section, we present related work in the area of recommender systems with a focus on approaches based on deep learning and product embeddings. In section III, we describe the use case and the data basis for our work. Section IV provides in detail our proposed approach combining product embeddings with

978-1-6654-9457-1/21/$31.00 ©2021 IEEE
DOI 10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00195

user-related filters. The methodology for model training and evaluation is provided in section V. In section VI, we present and discuss the results of the A/B-test. Finally, we conclude our paper with a summary of the findings and outlook of future work in section VII.

II. Related Work

A. Recommender Systems

Modern algorithms for recommender systems can basically be divided into five types of methods [6]: content-based, collaborative filtering, demography-based, knowledge-based, and hybrid recommender systems. Content-based filtering is based on the similarity between the items to be recommended. On the other hand, collaborative filtering considers the similarity between user activities. Here, it is assumed that there exist users with similar interests and preferences and thus similar behavior [6], [2], [7]. Matrix factorization [8], [9] represents a commonly used method in this context. Demographic-based filtering extends the consideration of the user with information such as age, gender, employment status, and location. Knowledge-based recommender systems are mainly, but not only, used when there is not enough information available to provide a recommendation. Examples are expensive purchases such as buying a house or a car. Nevertheless, recommendations can also be made based on all the knowledge that can be found in the use case [6]. More recent approaches involve the use of knowledge graphs [10]. Lastly, there exist hybrid recommender systems that combine several of the above approaches [6], [7].

B. Deep Learning-Based Recommender Systems

The ongoing development in the field of deep learning has also had a huge impact on the field of recommender systems. Deep learning-based recommendation systems can be distinguished based on the underlying architecture [3]. Since there is a variety of deep learning approaches the focus is set to recommendations with multilayer perceptrons, convolutional neural networks, recurrent neural networks, neural attention networks, and deep learning embeddings.

Multilayer perceptrons (MLP) are the basis for neural collaborative filtering in which deep neural networks and collaborative filtering are combined to large recommender frameworks [11], [12], [13], [14], [15], [16]. Recommendation with deep structured semantic models (DSSM) is another way to use MLPs for recommendations, where the networks have to learn semantic representations of items [17], [18].

Convolutional neural networks (CNN) are mainly utilized for feature extraction in unstructured multimedia data and it has been shown that recommendations can be made based on them [3], [19]. In addition, CNN-based collaborative filtering [20] or graph-CNNs [21], [22] can be used for recommendation. Recurrent neural networks are in contrast mainly used to predict the next item in a sequence of user interactions [23], [24], [25], [26]. More recently, approaches based on neural attention-based recommenders are developed. Chen et al [27] propose a collaborative filtering model with item and component attention. Tay et al [28] focus more on collaborative metric learning with memory-based attention.

Another popular branch of recommendation systems is based on product embeddings. Their concept is inspired by word2vec and word embeddings in the field of natural language processing. Mikolov et al. thereby introduced the skip-gram approach to efficiently train the word embeddings [29]. An important property of the word embeddings is that word vectors that are close to each other are also similar in terms of their meaning. Taking the properties of word embedding into consideration, it is reasonable to transfer the concept of word embeddings to recommender systems by training product embeddings (called prod2vec), as proposed by Ozsoy [30] for venue recommendation and Grbovic et al. [4] for email advertisement. Ozsoy considers venues as items in his work and trains embeddings accordingly. The recommendations are based on the cosine similarity of the embeddings. In contrast, the recommendation in the work of Grbovic et al. is based on a clustering approach. Vasile et al. [31] extend their product embedding by including additional product information in the embedding input. They call their proposed approach meta-prod2vec.

C. Recency, Frequency, and Monetary Analysis

In marketing, Recency, Frequency, and Monetary (RFM) analysis is a widely used method for customer segmentation and profiling [32], [33], [34], [35]. Thereby, a customer value is calculated for each customer, which is based on past behavior and includes three metrics. Recency is a value that describes "when" was the last time a customer has bought a product or service. Frequency describes "how often" the customer buys a product or a service in a given time range. Monetary value represents "how much" the customer spends in a given period [36]. Guney et al. successfully use RFM analysis to identify customer groups in a video on demand service for targeted marketing [35]. Frasquet et al. propose a RFM-based customer segmentation in a brick-and-mortar multichannel scenario in order to enhance the adoption of new channels [34]. In the work of Hsu et al., the authors propose a methodology based on RFM to identify critical products that are preferred by a special group of costumer but not by the average customer [37]. In summary, it is apparent from these works, among others, that RFM analysis is a successful method for segmenting the purchasing behavior of customers and, based on this, determining individual user preferences.

III. Use Case

In this paper, we evaluate our proposed recommendation approach in a real use case that deals with recommendations for consumer package goods (CPG), offered by a retailer located in the US with thousands of employees and hundreds of local stores. The company offers several thousand products in its portfolio. The products differ not only in their various categories but also in the frequency with which they are purchased by users (from once a year to several times a week) as well as in their price (ranging from low-price to premium

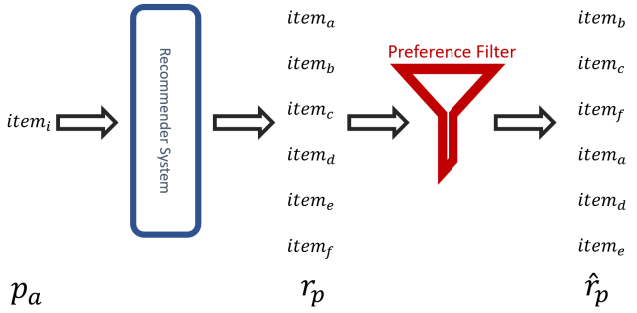

Fig. 1. Combination of the prod2vec-based recommender system with the pre-calculated preference filter. An recommendation for a product p_a leads to a product ranking r_p. Then, the preference filter of a user is applied to the ranked products and yields in a new ranking $\hat{r}_p$. The k products with the best ranking are recommended to the user.

segment). The retailer provides a website with an overview of all purchasable products in stock and the possibility to buy the products online.

For analysis and advertising purposes some user activities on the website are recorded and stored. This includes activities like product views, add to carts, and purchases. For most of the activities, users are uniquely identifiable via their IDs. In our experiments, we utilize historical data over a time span of one and a half years. From this data, we extract all interactions which indicate the visit of a product page, changes of a shopping cart, or product purchases.

The goal in this use case is to provide product recommendations for personalized email marketing campaigns. Millions of users are messaged as part of these campaigns. In an email, several products are promoted that could be interesting for the customer. The goal of the campaigns is to increase the user's binding to the company and its website. Therefore, it is important to make personalized recommendations that are relevant for the user. The advantage of email recommendations is that we exactly know which user we are targeting with a recommendation, which allows the recommendation to incorporate the past history. The success of a recommendation is measured by two key performance indicators: clicks on the recommendations and actual product purchases following the campaign.

IV. Methodology

In this paper, we propose a combination of a prod2vec recommender approach with pre-calculated user preferences. Thereby the underlying product embedding is trained by adapting the skip-gram technique on product purchases (i.e. checkouts). Figure 1 provides an overview of the components' interplay. First, a recommendation with a product ranking r_p is made by the prod2vec recommender. After that, a user-specific preference filter is applied to the product ranking r_p, which leads to a new product ranking $\hat{r}_p$. In the following, we will describe the individual components of our approach.

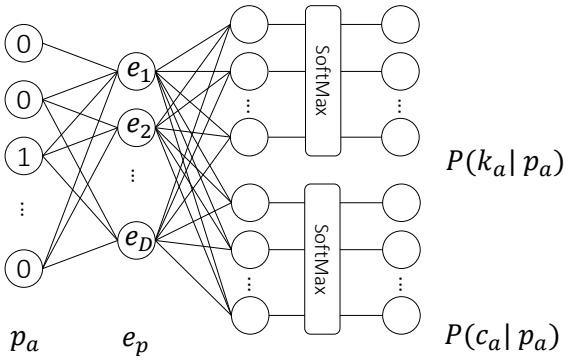

Fig. 2. The adapted extended skip-gram model with a one-hot encoded input layer for a product (p_a), an embedding layer (e_p) with D nodes, and two softmax output layers (k_a and c_a). The goal of the neural network is to predict the probabilities $P(k_a|p_a)$ and $(P(c_a|p_a)$ in which k_a is the category of the input product p_a and c_a is the checkout with the products that are bought together with the input p_a.

A. Extended Skip-gram Model

Our proposed product embedding is an extended version of the skip-gram model proposed by Tercan et al. [5] and is similar to meta-prod2vec by Vasile et al. [31]. However, we do not include the product category information as input to the neural network but as an output. Figure 2 illustrates this extended skip-gram model. Particularly, given two sets in which C is a set of checkouts $c \in C$ and P is a set of products $p \in P$ with $c \subseteq P$, $\emptyset \notin c$, and p is not necessarily unique in c. Also, given a set K of product categories $k \in K$ and a mapping function $m : p \mapsto k$. The objective is to train a D-dimensional product embedding representation $e_p \in \mathbb{R}^D$ in which similar products are closer to each other in the given embedding space. Thereby, we define the similarity between a pair of products (p_a, p_b) as follows: (p_a, p_b) are similar to each other if $m(p_a) = m(p_b)$ and there are checkouts $c' \subseteq C$ with the same products that contain p_a or p_b. In order to train a product embedding that fulfills that definition of similarity, we make use of the skip-gram approach. The learning task for the neural network is to predict a checkout c_a with n products p and simultaneously predict the category $k = m(p_a)$ of the a given input product p_a. With the simultaneous prediction of other purchased items and the product category, content-based and collaborative filtering-based information is encoded in the product embedding. Based on initial pre-tests that we've conducted prior to the main evaluation, we have observed that the extended skip-gram approach is superior to the normal skip-gram model.

B. Recommendation based on Product Embedding

The extended skip-gram approach provides a product embedding e_p that represents all products P. A recommendation for a target product is done by using the nearest neighbor approach in the embedding space. For this purpose, we use the cosine similarity as a distance metric:

$$Cosine\ Similarity = cos(\theta) = \frac{a \cdot b}{\|a\|\|b\|} \qquad (1)$$

in which θ describes the angel of two vectors a and b. $|cos(\theta)| \in [0, 1]$ for two vectors is 1 if both vectors are identical and 0 if the vectors are orthogonal in the vector space. The more similar the two vectors are the closer the cosine similarity is equal to 1 [38]. In order to recommend j products for a given target product, we calculate the cosine similarity of all products to it and retrieve the j products with the highest similarity.

C. Filtering based on User RFM Analysis

As presented in section II, RFM analysis can be used to generate individual customer information from purchase behavior. In our filtering approach, we use the RFM metrics to calculate user preferences scores. Specifically, we compute the RFM values for each user and product category k in a given time period. The RFM values are normalized over the categories k such that each user has a frequency value f_k and monetary value m_k with $f_k, m_k \in (0, 1]$. The user preference up_k for each category k is computed as $up_k = \frac{m_k + f_k}{2}$. When making product recommendations for a user, we use the computed product preference up_k to rearrange the ranked recommendations r_p that come from the skip-gram model. That yields a new product ranking $r'_p = \frac{r_p + up_k}{2}$ with $k = m(p)$.

In addition to this re-ranking, we use a monetary filter to recommend products whose price is equal to or more expensive than the price the user normally pays for the corresponding product category. From the RFM analysis of the user, we have the monetary information m'_k for each category which is the non-normalized value. Applying it to the product ranking r'_p yields a new product ranking $\hat{r}_p = r'_p + 1$, if $price \geq m'_k$ for a product p of category k else $\hat{r}_p = r'_p$. Then the products are sorted based on $\hat{r}_p$ and finally, the top j products are recommended.

V. Experiments

A. The Baseline Model

The model that is currently in use for the recommendations is based on the users' histories. The model's recommendations are computed by using the co-occurrence of the products that a user bought in the past. The calculation is based on the DIM-SUM algorithm [39], [40]. The model is updated weekly. However, the model does not account for products that are added within a week. For these newly added products, a fallback strategy is provided. This fallback recommendation is based on product metadata such as name, type, and description. Specifically, these recommendations are computed by the cosine similarity of the products calculated from a case-insensitive TF-IDF vector from a pre-determined set of a product's text-based attributes.

B. Experimental Setup

In our experiments, we train and test different variants of our recommendation approach to investigate the benefit of individual components such as the RFM-filter. For this purpose, we first conduct initial grid search based tests on the historical data to identify the best performing hyperparameters

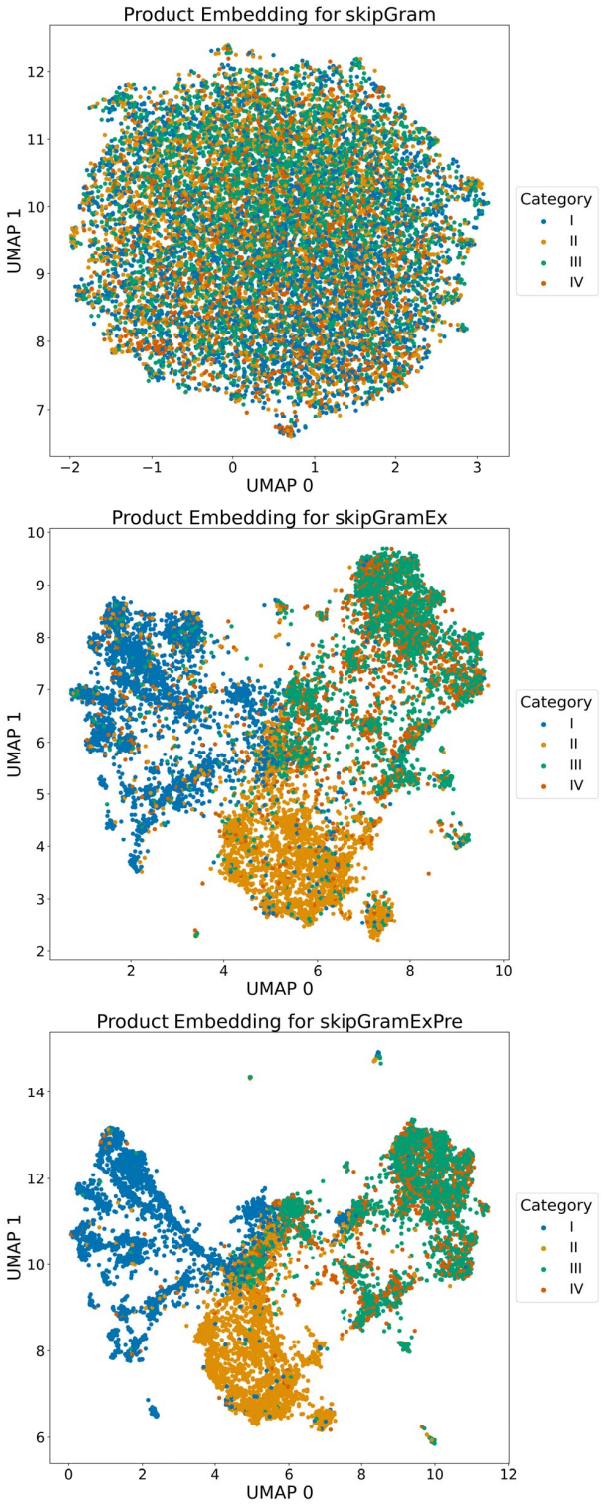

Fig. 3. Trained embeddings of all products in the set P for the three trained models skipGram, skipGramEx, and skipGramExPre. The embeddings are transformed into a two-dimensional space using UMAP and colored according to the (anonymized) product categories. Note that, for the model training, we use a finer level of product categories than in the visualization.

of the underlying neural network and the skip-gram method, resulting in a model with an embedding dimension $D = 128$ and a checkout size $n = 4$, which is equivalent to predicting four items. The number of different products p is in the ten-thousands and the products can be mapped to around a dozen categories k. We train the models for 150 epochs using the Adam optimization function [41] and the cross-entropy loss function.

Based on these results, we train the following different variants of the recommender approach. Note that all variants are trained on three months of historical user activity data before the A/B-test (unless otherwise noted).

- **skipGram**: skipGram model without product category extension in the model output
- **skipGramEx**: extended skipGram model with product category extension in the model output
- **skipGramExPre**: skipGramEx model which is pre-trained on data from a whole year and then finetuned on the same data as skipGramEx.
- **skipGramExPreFilter**: skipGramExPre model in combination with the RFM-filter for recommendations

Figure 3 shows a visualization of the product embeddings of the three different variants skipgram, skipGramEx and skipGramExPre by using the dimensionality reduction method UMAP [42]. The visualization clearly shows that skipGramEx and skipGramExPre allow a clearer separation of the different categories in the embedding. Interestingly, product categories III (green) and IV (red) are related product types. The plot also shows that the products are most distinguishable for the SkipGramExPre model. Therefore, we select the SkipGramExPre version for further experiments in the A/B-test.

As part of a two-week-long A/B test, both the trained skipGramExPre and the skipGramExPreFilter approaches are applied and compared to the baseline model. The A/B-test consists of several email campaigns in which millions of users are addressed. Each recommender is in charge of approximately the same number of users. At recommendation time, both the user ID and the most recent user activities are available to a recommender. The target product, which is used to make a recommendation, is calculated by taking the four most recent user activities and computing the mean product embedding out of the four corresponding products. The recommendation is then conducted as described in section IV. Thereby, the only difference between skipGramExPre and skipGramExPreFilter is that the latter uses the described user preference filter for rearranging the recommendations.

C. Evaluation Metrics

Our evaluation is based on both offline tests on historical campaign data as well as the online A/B-test. For the offline evaluation, our metric of choice is the hit rate HR, which is defined as the ratio if a recommended product has been clicked by the user considering historical data. For the computation, the number of hits is divided by the number of provided recommendation:

$$HR = \frac{\#hits}{\#provided\_recommendations} \tag{2}$$

After completion of the A/B-test, we calculated the click-through rates CTR of all tested recommenders:

$$CTR = \frac{\#clicked\_recommendation}{\#provided\_recommendations} \tag{3}$$

The CTR is a strong indicator if a recommended product is relevant for the user. However, it does not consider the value of a product. There are products in different price ranges and some products increase the revenue more than others. Therefore, we use as a second online evaluation metric the theoretical sales. The assumption is that the user will buy the product after clicking on it. This metric is a weighted CTR in which the clicks are weighted by the product price. Another metric we consider in our test is the real product purchases of users after having received an email. We, therefore, calculate the number as well as the total spending of purchases after three and six days. Additionally, we consider the sales uplift by considering the purchase behavior of the users before and after the test in order to estimate the impact of a recommender on the user groups.

VI. Results

Note that all results of the A/B-test (online evaluation) are represented as a percentual ratio to the performance of the baseline model.

A. Offline Evaluation

Table I shows the hit rate for the top-4 and top-14 product recommendation. We exclude the baseline model from this computation because it is the model in use and thus the clicked recommendations in the historical data are provided from it. For the top-4 recommendations, the hit rate of the skipGramExPre and skipGramExPreFilter models is not significantly different. In the case of the top-14 recommendation, the hit rate of the skipGramExPre model is better. From this, it could be deduced that the filter we use is not that useful for a good recommendation. The question is whether this also holds true for the real tests.

TABLE I
HIT RATE ON HISTORICAL CAMPAIGN DATA FOR THE SKIP-GRAM MODEL WITH AND WITHOUT THE RFM-FILTER FOR TOP-4 AND TOP-14 RECOMMENDATIONS.

Recommender	skipGramExPre		skipGramExPreFilter	
top-k	4	14	4	14
hit rate (in %)	**0.73**	**2.07**	0.71	1.84

B. Online Evaluation

Table II shows the ratio of the CTR for the three models of the A/B-Test. Every recommender has made millions of email recommendations. The table is split into two parts. The first row represents the CTR for all provided recommendations, whereby a single user may click on several recommended products in an email. The second row represents the number of unique users who clicked at least once on a recommended product in the email. The results show that the skipGramExPreFilter model achieves the best performance for both the regular and the unique CTR and performs with a 37% higher CTR than the baseline model. The other skip-gram model without the filter performs similarly to the baseline model. Table III shows the resulting theoretical sales of the A/B-test. The first row represents the ratio of the total value of all recommended products that were clicked. The second row represents the mean values of the products. In both cases, our models outperform the baseline model. The skipGramExPre model has the possibility to sell products that are 4.25% more expensive than the products of the baseline model. Our skipGramExPreFilter model can sell products that are on average 11.9% more expensive. The question that arises is whether this theoretical aspect will actually come into effect. Table IV shows the actual purchases of the users three and six days after they clicked on a recommendation. We, therefore, compute the number of purchases as wells as the mean, the standard deviation, and the total spending prices. In terms of total sales, our two proposed models perform better than the baseline model. However, the extended skip-gram model realizes the fewest purchases. After three days, customers that received a recommendation from the skipGramExPre model buy 23.81% fewer times than customers that got a recommendation from the baseline model. After six days, it is only 4.76% higher than the baseline. The highest count, mean, and total sales for both three and six days after recommendation are achieved by the skipGramExPrefilter model. On average the products are 122.91% more expensive for three days and 88.42% for six days. Although the users were randomly assigned to a recommender, it is possible that the results are caused by the fact that the assigned user groups generally have different purchasing behavior (e.g. buying more frequently and more expensive products than others). In order to validate the results, we calculate the sales uplift for every user group by comparing the buying behavior before and after the A/B-test. We, therefore, compute the average user spendings a few weeks before (pre-test) and after (post-test) the campaigns and take the mean uplift over all users' spending. Table V lists the ratios of the average spending for the pre-test phase and the post-test phase in relation to the baseline model. In addition, the last row shows the uplift, which is computed by subtracting the pre-test ratio from the post-test ratio. The results show that both user groups for the skip-gram models had higher pre-test spending (18.4% and 56.98%) than those of the baseline model. However, they also show that these gaps are growing after the A/B-test. Both our models have led the users to buy more products. The skipGramExPreFilter model thereby achieves the highest uplift of 52.83%.

TABLE II
CLICK-THROUGH RATE FOR THE THREE MODELS IN THE A/B-TEST. THE RATIO IS CALCULATED IN COMPARISON TO THE BASELINE MODEL.

Recommender	baseline	skipGramExPre	skipGramExPreFilter
ratio (in %)	0.0	-4.3	**37.0**
unique user ratio (in %)	0.0	1.5	**23.1**

TABLE III
ACCUMULATED TOTAL AND MEAN PRODUCT PRICE RATIO FOR THE RECOMMENDED PRODUCTS THAT WERE CLICKED BY THE USERS DURING THE TWO-WEEK A/B-TEST. ALL VALUES ARE A RATIO WITH RESPECT TO THE VALUES OF THE BASELINE MODEL.

Recommender	baseline	skipGramExPre	skipGramExPreFilter
total value ratio (in %)	0.0	9.41	**64.41**
mean value ratio (in %)	0.0	4.25	**11.9**

TABLE IV
CALCULATED AMOUNT AND PRICES OF PURCHASED PRODUCTS FOR EACH RECOMMENDER WITHIN THREE AND SIX DAYS AFTER USERS HAVE CLICKED ON A RECOMMENDATION. ALL VALUES ARE A RATIO WITH RESPECT TO THE VALUES OF THE BASELINE MODEL THREE DAYS AFTER A CLICKED RECOMMENDATION.

Recommender	baseline		skipGramExPre		skipGramExPreFilter	
days after	3	6	3	6	3	6
count ratio (in %)	0.0	23.81	-23.81	4.76	**23.81**	**85.71**
mean ratio (in %)	0.0	11.66	44.61	60.78	**122.91**	**88.42**
std ratio (in %)	0.0	17.87	30.76	63.78	**167.9**	**143.79**
total ratio (in %)	0.0	38.25	10.18	68.43	**162.84**	**249.93**

TABLE V
RATIO OF MEAN SPENDING OVER ALL USERS PER RECOMMENDER BEFORE AND AFTER THE A/B-TEST. THE RATIO IS COMPUTED WITH RESPECT TO THE BASELINE MODEL.

Recommender	baseline	skipGramExPre	skipGramExPreFilter
pre-test ratio (in %)	0.0	18.40	35.98
post-test ratio (in %)	0.0	56.98	88.81
uplift (in %)	0.0	38.58	**52.83**

C. Further Discussion

The obtained results provide important insights regarding evaluation methods of recommender systems. In our case, the offline evaluation showed that using the RFM-filter does not lead to better performances (i.e. hit rate). A big issue is that an evaluation based on historical data cannot capture how a customer would react if another product is recommended. Additionally, the data is based on the clicked recommendations of the baseline model. On the historical data, the non-filter model (skipGramExPre) performs better than the filter-based model. One reason might be that both the baseline and the skipGramExPre model are built to make recommendations based on average customer behavior. The filter model instead takes into account the individual user behavior from past activities.

It is obvious that an online evaluation with an A/B-test is a better evaluation method for recommendation systems. Unlike offline evaluation, we are able to observe the direct effect of the recommendation on the users. However, the online evaluation is not flawless either. This is due to the individual purchase behavior of the users. There exist many external factors that could play a role and that are not considered in the recommender, such as seasonal behavior, holidays, or special occasions of users. To compensate for different unknown effects, we have used a composition of different metrics.

Taking all evaluation results into account, we show that our two proposed models outperform the baseline model that is currently in use. In addition, it can be concluded that the skipGramExPreFilter model performs significantly better than the skipGramExPre model. We believe that this is due to several factors. First, our proposed extended skip-gram model encodes two different types of information in the embedding: the product similarity representing products that are bought together and product similarity based on additional meta-information such as the product category. With this extended skip-gram model, we have a solid product representation. Second, our proposed RFM-filter incorporates additional user information into the recommendation. This leads to better recommendations for the users and additional profits for the company by recommending more appropriate as well as more expensive products. Recommending more expensive products had no measurable negative effect on the click rate or sales in general. Here the question arises whether customers are discouraged by too expensive products or, on the contrary, are attracted to the site by expensive products.

VII. Summary and Outlook

In this paper, we propose a prod2vec-based product-to-product recommendation model in which we incorporate product and purchase information. Our main contribution is two-folded. First, we propose an extension of a skip-gram architecture to include additional product information such as the product category into the learning task. Second, we use a filter that rearranges the model's prediction based on the RFM analysis of user data. We conducted an extensive evaluation of our models using offline and online evaluations based on an A/B-test. An important finding is that evaluation on historical data can lead to conflicting results comparing to real-world tests and therefore has only a limited validity with regard to the performance of a recommender system in real use. The online evaluation results show that, on the one hand, our recommender achieves an improvement with regard to the baseline model that is currently in use and, on the other hand, using a filter is better than none.

A major challenge for recommender systems is to recommend the right product to the right person at the right time. In the future, we want to continue investigating in recommending the right product to the right person. Therefore, we pursue to improve the proposed RFM-filter by incorporating additional metrics as well as learning user-specific filters by means of machine learning. We also pursue to investigate state-of-the-art customer segmentation methods for better recommendations. In addition, future work will deal with the time aspect in the recommendation, such as by detecting and predicting recurring purchase behavior of users.

References

[1] "Estimates of global e-commerce 2019 and preliminary assessment of covid-19 impact on online retail 2020," *UNCTAD Technical Notes on ICT for Development*, no. 18. [Online]. Available: https://unctad.org/system/files/official-document/tn_unctad_ict4d18_en.pdf

[2] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *IEEE transactions on knowledge and data engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[3] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, pp. 1–38, 2019.

[4] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp, "E-commerce in your inbox: Product recommendations at scale," in *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, 2015, pp. 1809–1818.

[5] H. Tercan., C. Bitter., T. Bodnar., P. Meisen., and T. Meisen., "Evaluating a session-based recommender system using prod2vec in a commercial application," in *Proceedings of the 23rd International Conference on Enterprise Information Systems - Volume 1: ICEIS,*, INSTICC. SciTePress, 2021, pp. 610–617.

[6] D. Das, L. Sahoo, and S. Datta, "A survey on recommendation system," *International Journal of Computer Applications*, vol. 160, no. 7, 2017.

[7] G. Adomavicius and A. Tuzhilin, "Context-aware recommender systems," in *Recommender systems handbook*. Springer, 2011, pp. 217–253.

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

[9] Y. Zhou, D. Wilkinson, R. Schreiber, and R. Pan, "Large-scale parallel collaborative filtering for the netflix prize," in *International conference on algorithmic applications in management*. Springer, 2008, pp. 337–348.

[10] B. Hui, L. Zhang, X. Zhou, X. Wen, and Y. Nian, "Personalized recommendation system based on knowledge embedding and historical behavior," *Applied Intelligence*, pp. 1–13, 2021.

[11] G. K. Dziugaite and D. M. Roy, "Neural network matrix factorization," *arXiv preprint arXiv:1511.06443*, 2015.

[12] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*, 2017, pp. 173–182.

[13] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "Deepfm: a factorization-machine based neural network for ctr prediction," *arXiv preprint arXiv:1703.04247*, 2017.

[14] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, "Collaborative knowledge base embedding for recommender systems," in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 353–362.

[15] X. Li and J. She, "Collaborative variational autoencoder for recommender systems," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 305–314.

[16] Y. Wu, C. DuBois, A. X. Zheng, and M. Ester, "Collaborative denoising auto-encoders for top-n recommender systems," in *Proceedings of the ninth ACM international conference on web search and data mining*, 2016, pp. 153–162.

[17] A. M. Elkahky, Y. Song, and X. He, "A multi-view deep learning approach for cross domain user modeling in recommendation systems," in *Proceedings of the 24th international conference on world wide web*, 2015, pp. 278–288.

[18] J. Serra and A. Karatzoglou, "Getting deep recommenders fit: Bloom embeddings for sparse binary input/output networks," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, 2017, pp. 279–287.

[19] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, 2018, pp. 565–573.
[20] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," *arXiv preprint arXiv:1808.03912*, 2018.
[21] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.
[22] R. v. d. Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," *arXiv preprint arXiv:1706.02263*, 2017.
[23] C.-Y. Wu, A. Ahmed, A. Beutel, A. J. Smola, and H. Jing, "Recurrent recommender networks," in *Proceedings of the tenth ACM international conference on web search and data mining*, 2017, pp. 495–503.
[24] S. Wu, W. Ren, C. Yu, G. Chen, D. Zhang, and J. Zhu, "Personal recommendation using deep recurrent neural networks in netease," in *2016 IEEE 32nd international conference on data engineering (ICDE)*. IEEE, 2016, pp. 1218–1229.
[25] A. Tamhane, S. Arora, and D. Warrier, "Modeling contextual changes in user behaviour in fashion e-commerce," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2017, pp. 539–550.
[26] Y. Gui and Z. Xu, "Training recurrent neural network on distributed representation space for session-based recommendation," in *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–6.
[27] J. Chen, H. Zhang, X. He, L. Nie, W. Liu, and T.-S. Chua, "Attentive collaborative filtering: Multimedia recommendation with item-and component-level attention," in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, 2017, pp. 335–344.
[28] Y. Tay, L. Anh Tuan, and S. C. Hui, "Latent relational metric learning via memory-based attention for collaborative ranking," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 729–739.
[29] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.
[30] M. G. Ozsoy, "From word embeddings to item recommendation," *arXiv preprint arXiv:1601.01356*, 2016.
[31] F. Vasile, E. Smirnova, and A. Conneau, "Meta-prod2vec: Product embeddings using side-information for recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, 2016, pp. 225–232.
[32] F. Liu, S. Zhao, and Y. Li, "How many, how often, and how new? a multivariate profiling of mobile app users," *Journal of Retailing and Consumer Services*, vol. 38, pp. 71–80, 2017.
[33] S. Shokouhyar, S. Shokoohyar, and S. Safari, "Research on the influence of after-sales service quality factors on customer satisfaction," *Journal of Retailing and Consumer Services*, vol. 56, p. 102139, 2020.
[34] M. Frasquet, M. Ieva, and C. Ziliani, "Online channel adoption in supermarket retailing," *Journal of Retailing and Consumer Services*, vol. 59, p. 102374, 2021.
[35] S. Guney, S. Peker, and C. Turhan, "A combined approach for customer profiling in video on demand services using clustering and association rule mining," *IEEE Access*, vol. 8, pp. 84 326–84 335, 2020.
[36] A. M. Hughes, "Boosting response with rfm." *Marketing Tools*, pp. 4–8, 1996.
[37] P.-Y. Hsu and C.-W. Huang, "Iect: A methodology for identifying critical products using purchase transactions," *Applied Soft Computing*, vol. 94, p. 106420, 2020.
[38] A. Singhal *et al.*, "Modern information retrieval: A brief overview," *IEEE Data Eng. Bull.*, vol. 24, no. 4, pp. 35–43, 2001.
[39] R. B. Zadeh and G. Carlsson, "Dimension independent matrix square using mapreduce," *arXiv preprint arXiv:1304.1467*, 2013.
[40] R. B. Zadeh and A. Goel, "Dimension independent similarity computation." *Journal of Machine Learning Research*, vol. 14, no. 6, 2013.
[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[42] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.