User Modeling and User-Adapted Interaction

Session-based Item Recommendation in E-Commerce

On Short-Term Intents, Reminders, Trends, and Discounts

Dietmar Jannach · Malte Ludewig · Lukas Lerche

August 2017

Abstract Many e-commerce sites present additional item recommendations to their visitors while they navigate the site, and ample evidence exists that such recommendations are valuable for both customers and providers. Academic research often focuses on the capability of recommender systems to help users discover items they presumably do not know yet and which match their long-term preference profiles. In reality, however, recommendations can be helpful for customers also for other reasons, for example, when they remind them of items they were recently interested in or when they point site visitors to items that are currently discounted.

In this work, we first adopt a systematic statistical approach to analyze what makes recommendations effective in practice and then propose ways of operationalizing these insights into novel recommendation algorithms. Our data analysis is based on log data of a large e-commerce site. It shows that various factors should be considered in parallel when selecting items for recommendation, including their match with the customer's shopping interests in the previous sessions, the general popularity of the items in the last few days, as well as information about discounts. Based on these analyses we propose a novel algorithm that combines a neighborhood-based scheme with a deep neural network to predict the relevance of items for a given shopping session.[1]

Keywords Recommender Systems · E-commerce

1 Introduction

Automated and in many cases personalized recommendations of the type "You may also be interested in . . . " are a common feature of modern e-commerce sites. Such recommendations can serve different purposes and create additional value for both customers and providers, e.g., by helping customers discover additional items of interest or by helping providers promote certain areas of their item spectrum.

The academic literature in the field mainly focuses on the recommendation utility for customers or the capability of algorithms to identify items that an individual user is presumably not aware of (Jannach and Adomavicius, 2016). In research settings the

D. Jannach, M. Ludewig, L. Lerche
Department of Computer Science, TU Dortmund, Germany
E-mail: firstname.lastname@tu-dortmund.de

[1] Parts of this work are based on (Jannach et al, 2015a), Lerche et al (2016) and (Jannach and Ludewig, 2017a). This paper or a similar version is not currently under review by a journal or conference, nor will it be submitted to such within the next three months. This paper is void of plagiarism or self-plagiarism as defined in Section 1 of ACM's Policy and Procedures on Plagiarism.

recommendation problem is often abstracted to one of matrix completion. The main algorithmic task given this problem formulation is then to reliably estimate the relevance of the unseen items based on the user's long-term behavior and preferences, and to create a ranked list of objects to be presented to the user.

However, in e-commerce environments, and in particular on retail websites that offer a wide range of different products, this research approach has a number of limitations and does not fully reflect all of the challenges of the domain.

- First, customers often visit e-commerce sites with a very specific shopping intent in mind (Moe, 2003). Relying solely on long-term preference models can be insufficient and algorithms have to adapt their recommendations to these short-term goals to be effective (Shani et al, 2005; Tavakol and Brefeld, 2014; Jannach et al, 2015a).
- Second, when relying on the matrix-completion problem formulation, the focus is on predicting the relevance of items for which no preference signal is given and which are presumably unknown to the user. However, many real-world systems also recommend items that the user has already seen or even purchased before, e.g., with the goal to remind users of things they were recently interested in or to provide convenient navigation shortcuts (Schnabel et al, 2016).
- Third, in many e-commerce domains, the purchase decision of a customer can depend on the current price of an item and some items might only be attractive if they are on sale. The relationship between prices and market demand has been extensively analysed in the economics literature for decades, e.g., based on statistical demand estimation approaches like the broadly-used one by Berry et al (1995), but it is not fully clear yet how price reductions impact the behavior of users in the context of recommendations.
- Fourth, an assumption in current research is that the recommendation of very popular items is of comparably little value, because the recommended items might be too obvious or of little novelty. Reports on real-world systems however indicate that recommending a mix of popular and lesser known items can be a good strategy (Garcin et al, 2014; Gomez-Uribe and Hunt, 2015).

Overall, whether or not an item should be recommended may depend on a number of factors other than its estimated match with the user's long-term preferences. These additional factors are however underexplored in the recommender systems literature. In this work, our aim is to systematically investigate the role of some of these factors – in particular those discussed above – and to design novel approaches to incorporate these factors into future recommender systems.

The structure and the contributions of this paper are as follows:

(i) In Section 2, we report the results of a systematic analysis of a large log dataset provided to us by Zalando[2], a European retailer of fashion products. A particular feature of the dataset is that it contains information about which items were recommended to users and which of these recommendations were successful, i.e., led to a purchase afterwards. This information allows us to systematically determine the characteristics of successful recommendations from log data, which to our knowledge has not been done before in the e-commerce domain.

(ii) Our analysis reveals that a larger fraction of the successful recommendations are actually *reminders*, i.e., items that the users have seen or purchased before. In Section 3 we therefore further elaborate on this topic and also report the results of a field test in a different e-commerce domain which shows that reminding users of known items can have a positive impact on the business.

[2] `http://www.zalando.com`

(iii) In Section 4 we finally show how the insights from the previous analyses can be operationalized within new recommendation algorithms. Specifically, we propose a hybrid method that combines a session-based nearest-neighbor scheme with a deep neural network that considers additional factors like discounts or recent sales trends in the ranking process. An experimental evaluation indicates that the proposed methods are more effective than previous approaches in terms of selecting and recommending items that are actually bought by customers on the site.

Generally, the work presented in this paper combines and extends a series of previous investigations on session-based recommendations in e-commerce environments. In (Jannach et al, 2015a) we mainly examined the relative importance of considering long-term preference models and short-term intents; in (Lerche et al, 2016) we focused on the value of placing reminders in recommendation lists; in (Jannach and Ludewig, 2017a) we finally designed a first recommendation method that used purchase prediction variables which were derived from a systematic analysis of log data. In this paper, we put these individual pieces together to provide a comprehensive picture of different phenomena that can be observed in practice but have not been explored in the literature to a large extent. Furthermore, we propose a novel method to combine the prediction features, which is more effective than our previous one and which is based on a deep learning technique.

2 Analyzing the Characteristics of Successful Recommendations

This section summarizes the main insights that were obtained through the analysis of Zalando's log data. The general goal of our analyses was to better understand in which cases the recommendations that were displayed to the users were successful (i.e., led to a purchase at the end).

2.1 Dataset Characteristics

The dataset provided to us contains a subset of the log of user interactions on the e-commerce site Zalando during about one year.[3] Actions were recorded for about 3.5 million users, which were identified through unique IDs based on cookies. The different types of actions that were made available to us include views of item detail pages, purchase actions, as well as add-to-cart and add-to-wishlist events. The recorded actions relate to more than 400,000 different shop items. For each item, we know various basic characteristics like the (anonymized) brand, the color, the item's catalog categorization (i.e., if it belongs to shoes or shirts), and the price level compared to other items of the same category. Furthermore, the log contains information about to which extent the item was discounted at the time of the user action.

Zalando's online shop also features a recommendation component, and additional items of interest are displayed to visitors when they navigate the site, for example, when they inspect the details page of an item.[4] Our log dataset contains the list of the top three recommended items that were displayed to the visitors on such pages. Click events on these recommended items were recorded as well, i.e., we know when a user visited an item through a link from the recommendation list. In our subsequent analysis, we consider a recommendation successful when the user purchased an item later on that he or she clicked earlier on a recommendation list.

[3] The dataset provided to us was fully anonymized and artificially distorted so that no inference on true sales numbers on the site can be made.

[4] Details about the used recommendation algorithm on the website were not disclosed to us.

Table 1: Characteristics of the Zalando datasets

	Raw dataset	*Frequent* users	*Regular* users	*Occasional* users
Users	3.5M	3,000	3,000	3,000
Items	460k	188k	121k	87k
Views	200M	3.1M	906k	452k
Purchases	3.9M	106k	47k	9.5k
Sessions	27.5M	338k	89k	64k
Sessions per user	7.79	113.11	29.91	21.31
Views per session	7.28	9.44	10.41	7.10
Purchases per session	0.14	0.31	0.52	0.15

The dataset is generally very sparse. Many users have visited the shop only once and a large majority of the users never made a purchase. Since one of our goals is to predict purchases based both on long-term and short-term user models, we focused on users that have interacted with the website several times in different shopping sessions and also made a minimum number of purchases. Removing non-buying users, which are not in the focus of our analysis, reduces the total number of unique customers by almost 80%, i.e., only about 760,000 of 3,5 million users remain.

For the various analyses and experiments made in this paper, we created a number of different data subsamples. Specifically, we varied the lower thresholds regarding the user's past activities, with the goal to assess if significant differences can be observed across different user groups, e.g., when applying different recommendation strategies.

- Our first sample contains 3,000 *frequent* (heavy) shoppers. We defined those customers as heavy users that had purchased at least 20 items during the year when the data was collected and who have interacted with the site within at least 40 sessions. On average, the randomly selected users of this group visited the online shop about two times a week, leading to an average of 114 sessions.
- The second sample covers a random selection of 3,000 *regular* shoppers, who purchased at least 10 items within at least 20 sessions. For the average customer in this group about 30 sessions were recorded, i.e., these users visited the site about every other week.
- The third sample comprises 3,000 randomly selected *occasional* site visitors, who interacted with the site at least in 10 sessions. No constraint on the number of purchases per user was applied for this dataset.

To further validate our novel prediction method presented in Section 4, we created the following additional datasets.

- A sample of 3,000 *random* users for which we only required that they purchased at least one single item. No constraint on the number of shop visits was applied.
- Two larger samples of *regular* users as described above, which comprise 5,000 and 10,000 users, respectively.

Table 1 shows some key characteristics of the three main datasets that we use for our analyses in the paper. The characteristics of the three additional validation datasets are given in Table 17 in Appendix D. Also in the appendix (Figure 6), we provide a visualization of the distribution of the purchase frequencies in the raw dataset. What we can for example observe in the data is that the frequent users visit the site more often, i.e., there are many more sessions per user, but their sessions are shorter on average and they make fewer purchases per session.

Generally, while the number of sampled users per dataset is comparably low, the resulting datasets still have a substantial size. The frequent user dataset for example

contains more than 3 million recorded item view events for almost 200,000 different items. We therefore see the dataset sizes not as a limiting factor of our research.[5]

2.2 Factors Influencing the Success of Recommendations

Based on our discussion of typical limitations of current research in the introduction of our paper, we investigated the impact of the following factors on the success of recommendations through different analyses. Each of these factors corresponds to one of the identified research limitations.

1. The importance of considering the users' short-term intents.
2. The effectiveness of recommending already known items.
3. The role of discounts.
4. The effects of considering (short-term) popularity trends.

We used the dataset of frequent shop visitors as a basis for the subsequent analyses. As a success measure for the recommendations, we calculated click-through rates and conversion rates. In our scenario, a *click-through* event happens whenever a user clicks on any of the items in a recommendation list and the click-through rate is computed as the number of click-through events divided by the number of page (recommendation list) impressions. We define a *conversion* as the situation when a user clicked on an item within a recommendation list and then actually purchased this item in the current or subsequent session. We determine the conversion rate by dividing the number of conversions by the number of clicks on any recommended item. Overall, the click-through rate therefore can be seen as an indicator to which extent the recommendations can attract clicks, whereas the conversion rate gives an indication of the effectiveness of a recommendation algorithm in terms of selecting items that match the users' preferences.[6]

In our sample of frequent users, we observed that about every 100^{th} displayed recommendation list received a click, i.e., a click-through rate of 1%. In general, the absolute value of the click-through rate can depend on various factors that are not related to the recommendation quality, including the visual layout of the web pages and the positioning of the recommendation list. Overall, a click-through rate of 1% shows that the displayed recommendations are not a central element for users to navigate the site.

The second and probably more interesting question is how often an item was added to the shopping cart and purchased after it was selected in a recommendation list, i.e., the conversion rate. In the dataset of frequent shop visitors, users placed an item into the shopping cart in one of the next two sessions in about 14 % of the cases when they had clicked on it within a recommendation list. In about 7 % of the cases when an item was clicked on in the list, it was actually purchased later on. This indicates that the recommender implemented on the site was in many cases able to select items that were truly interesting for the user. Note that this does not mean that only 7 % of the recommended items were interesting for the users, as we only know for the actually purchased items with certainty that they were relevant. In fact, other items shown in the recommendations might have been relevant as well, but not purchased at the end, e.g., because they were alternatives to the finally purchased item.

[5] Using much larger samples makes some of the experiments reported in later sections computationally challenging as some of the algorithms require extensive hyperparameter tuning.

[6] The conversion rate cannot directly be interpreted as the absolute amount of *additional* sales that is generated by a recommender since we cannot know if an item would have been bought by a user even when it was not recommended. Previous studies however indicate that recommenders in general can turn more visitors into buyers and be effective in terms of generating additional sales (Jannach and Hegelich, 2009).

2.2.1 The Importance of Considering the Users' Short-Term Intents

To assess how focused website visitors are when they arrive on the site, we analyzed their browsing behavior in terms of the diversity of the inspected items. An analysis based on the sample of frequent users showed that visitors on average look at the details of 9 different items per session and these items belong, again on average, to 2.7 different categories in the catalog. Given that there are more than 330 categories, this is a strong indicator that focused navigation behavior with a specific intent is common. Therefore, recommending mostly items that match the current shopping intent seems promising.

To quantify the importance of considering short-term intents when recommending, we calculated the recommend-to-purchase conversion rates for different situations: (a) when the recommended item was similar to the currently inspected one in terms of a certain feature (e.g., had the same color) and (b) when this was not the case. The results of this analysis are shown in Table 2.

Table 2: Recommend-to-Purchase Conversion Rates for Similar-Item and Different-Item Recommendations using the Dataset of Frequent Users

	Conversion rate when item feature has ...		
Item feature	... a different value	... the same value	Difference
Brand	0.950 %	4.227 %	345 %
Price level	1.403 %	3.624 %	158 %
Category	1.207 %	2.844 %	135 %
Color	1.521 %	2.701 %	77 %

The first row of the table for example shows that once a visitor arrived at the site and inspected an item of a certain brand, the conversion rate was 345 % higher when an item of the same brand was recommended than when items of other brands were recommended.[7] Much higher conversion rates are also achieved when items from the same price segment are recommended (158 %) and when the item belongs to the same category (135 %). Similarly, a substantial improvement (77 %) is observed when the recommendations have the same color as the last inspected item, even though the color scheme used by the shop is quite fine-grained.

Overall, the analysis shows that website users often have a comparably clear shopping intent when they visit the site and recommendations are correspondingly more successful when they relate to items that have characteristics that are similar to those of recently inspected ones. Therefore, recommending items from the most recent categories of interest might be a more effective strategy than making out-of-category recommendations. If the assumption holds that items of the same category are typically substitutes (alternatives) and items of other categories are complements (e.g., accessories), our observations would corroborate the findings of Diehl et al (2015) who also found a substitute-based store organization to be advantageous over a complement-based one for the fashion domain.

2.2.2 The Effectiveness of Recommending Already Known Items

When examining all three-item recommendations in the log, we noticed that about 10 % of these recommendations were items that the current user has inspected before at least

[7] Since brand loyalty is common in the fashion domain, a strong effect was generally expected.

once. The recommender system implemented on the site, like on other e-commerce sites such as Amazon.com, does therefore not limit itself to the recommendations of items that are presumably new to the user as is usually done in academic research.

The fraction of such *reminders* depends on the design and inner workings of the recommendation algorithm, which are unknown to us. The interesting part however is that nearly half (44 %) of the *successful* recommendations that finally led to a purchase were not new to the visitors. Obviously, we cannot know if the customers bought items *because* they were presented as reminders in the recommendation list. However, we can at least observe that visitors use the recommendations quite often as navigation shortcuts to items that they finally purchase (Plate et al, 2006; Schnabel et al, 2016). Including reminders in recommendation lists therefore seems to be promising in this environment.

On the other hand, the fact that more than half of the successful recommendations were actually unknown to the users shows that the implemented system is effective in helping users discover new items as well. From a practical perspective, this means that the most promising strategy could be to create recommendation lists that contain a well-balanced mix of already known items and items that are new to the visitor.

Generally, the selection of items that assumedly match the short-term interests can be made independently of the selection of the reminder items. However, once an estimate of the current user's shopping intent has been made, it can also be a plausible strategy to focus on reminder items that are a good match for the given shopping intent.

2.2.3 The Role of Discounts

In many domains, the price of an item has a direct impact on demand levels and sales. We could therefore hypothesize that items in recommendation lists that are marked as being discounted are more attractive to users and lead to higher conversion rates. On the other hand, there could also be a negative effect caused by recommending items that are labeled as being discounted. This could happen when visitors perceive the recommendations rather as advertisements than as unbiased hints by a benevolent recommendation system.

To analyze the effect in our dataset, we separately calculated the recommendation-to-purchase conversion rates for recommendations that were labeled as being on sale and items that were sold at the regular price. The observed differences were huge. While the conversion rates for non-discounted items were at only 0.45 %, the rate for items on sale was about 18 times higher and at 8.12 %. This is a clear indicator that on the analyzed website the recommendation of discounted items led to a higher effectiveness of the recommendation system.

Note that the majority of sales transactions during the data collection period were *not* involving discounted items. Furthermore, the existing recommender on the site on average only included one single discounted item in the recommendation list. Recommendations of discounted items were therefore disproportionately often successful.

2.2.4 The Effects of Considering (Short-Term) Popularity Trends

Recommending popular items is generally a "safe" strategy, even though it might not always lead to the highest business value as shown, e.g., in (Jannach and Hegelich, 2009). We made different analyses to identify a possible relationship between the popularity of a recommended item and the chances that the recommendation is adopted by a user.

A first analysis showed that whenever the visitor clicked on one of the three recommendations, the chances that it was the most popular one among the three were at 43 %, i.e., much higher than the theoretical 33 % random chance. For this particular measurement, we determined an item's general popularity by counting all view and purchase

events related to the item in the entire log dataset. However, in the fashion domain, seasonal trends are common and considering the item popularity over the period of one year is probably too coarse-grained. We therefore made an additional measurement in which we considered the popularity of the items in the recent past. As an example, we looked at how popular each item on a recommendation list was on the day on which the recommendation was made.

The results showed that recommendations were particularly successful when the recommended items were popular on that day and therefore probably represent recently trending items. Using a normalized popularity score, the average daily popularity of all recommended items was at 0.024, whereas the average of those which were actually selected afterwards was at 0.088, i.e., three times higher. The normalized score was computed by dividing the number of events (clicks and purchases) of an item on a specific day by the maximum number of events recorded for an item in the dataset on the same day.[8]

2.3 A Systematic Feature Importance Analysis

After having analyzed different potential success factors individually in the previous section, our next goal was to understand the *relative importance* of the different factors that can make a recommendation successful. To that purpose we used the available data to frame a classification problem where the task is to predict whether or not a displayed recommendation will later on lead to a purchase or not. Based on such a model, different feature weighting methods can be applied to numerically estimate the importance of the factors.

Correspondingly, each entry in the constructed classification dataset corresponds to an item recommendation recorded in the log data, which is labeled as being successful or not. We engineered a set of 95 different features as predictor variables. We included both comparably simple ones like the popularity of the item during the last n days as well as more complex ones that combine item characteristics with context-specific or user-specific aspects. An example for a more complex feature is the ratio of clicks by a user on items that have the same brand as the recommended one during the last n sessions. The full list of features can be found in Appendix C.

For the measurement, we again looked at the 3,000 *frequent* visitors and their successful recommendations from our dataset, leading to about 8,500 positive samples. Since the dataset is very imbalanced and there are comparably few successful recommendations, we applied random downsampling to end up with an equal number of samples for each class. Table 3 shows the 10 most relevant features using the *Gain Ratio* and the *Chi Squared* methods, respectively.[9] We used two alternative and popular feature selection methods to reduce the risk that our analysis is biased by the specifics of one single method. When looking at the most important features as listed in the table, we can observe that both methods lead to the exact same set of the 10 most relevant features (out of over 90). The order of the features is sometimes slightly different, which is caused by the specific ways the methods work.[10]

The results of both methods are therefore comparable and confirm the observations from the previous section. Every single feature in the top-10 list is either related to the recent popularity of the items, to current discounts, or related to the fact that a user has

[8] The popularity measurement only considered view and purchases related to the items up to the time point of the recommendation on that day.

[9] See, e.g., (Manning et al, 2008) for details about these feature selection methods.

[10] The results of the analysis for the set of 3,000 *occasional* users are similar and are reported in Appendix B. The detailed results of the feature analysis are provided at `http://ls13-www.cs.tu-dortmund.de/homepage/journal-cosr-2017`.

Table 3: Results of the statistical feature weight analysis.

Gain Ratio analysis	
Feature	Weight
Viewed before?	0.319
Any discount granted?	0.274
Discount level	0.274
Distance to last view (in days)	0.251
Current popularity (day)	0.249
Distance to last view (in sessions)	0.232
Distance to first view (in days)	0.199
Distance to first view (in sessions)	0.194
Number of previous views	0.181
Current popularity (week)	0.138

Chi Squared analysis (normalized)	
Feature	Weight
Current popularity (day)	1.000
Distance to last view (in sessions)	0.624
Distance to last view (in days)	0.619
Current popularity (week)	0.610
Number of previous views	0.603
Distance to first view (in sessions)	0.598
Distance to first view (in days)	0.595
Viewed before?	0.590
Any discount granted?	0.569
Discount level	0.569

recently viewed a given item in the past.[11] A correlation analysis of the relevant features over all positive and negative samples showed that while most groups of different features are not related, a measurable correlation between the discount level and the recent item popularity exists (0.47 for popularity/day, 0.34 for popularity/week). This could mean either that some items become particularly popular once they are discounted, or that the shop often reduces the prices for more popular items because discounts for popular items might help to attract more visitors to the site.

Overall, the analysis gives us a number of indications which factors contribute to the later success of a recommendation. In the following sections we will investigate how we can operationalize these insights when designing new recommendation algorithms.

3 A Detailed Analysis of Using Reminders Within Recommendations

The feature analysis results in Table 3 showed that there are in fact several features in the top ten list that are related to *reminders*, i.e., to items that the user has recently inspected. In this section we provide a more detailed analysis on the topic of reminders before we investigate the other features in Section 4.

Generally, reminders within recommendation lists have the unique characteristic that they do not help users discover so far unseen items, and it is therefore not fully clear to what extent they truly create business value. Our analyses so far only show that clicks on already known items in the recommendations lists comparably often lead to purchases afterwards. To assess the business value of reminders, we have performed an A/B test on an e-commerce site for electronics and gadgets in which we compared the effectiveness of a simple reminding strategy with other recommendation techniques. We will discuss the details in Section 3.1. Then, we will outline different alternative and more elaborate algorithmic approaches to select already known items for inclusion in the recommendation lists. The results of different offline experiments using the Zalando dataset will be summarized in Section 3.2.

3.1 Effectiveness Analysis: Results of a Field Test

To be able to conduct field tests on a real e-commerce website we collaborated with China-Gadgets[12], a German online retailer who specializes in consumer electronics and

[11] The entry "Distance to first/last view" in the table refers to the time that has passed since the user has viewed a recommended item the first or last time before the purchase.

[12] `http://china-gadgets.de`

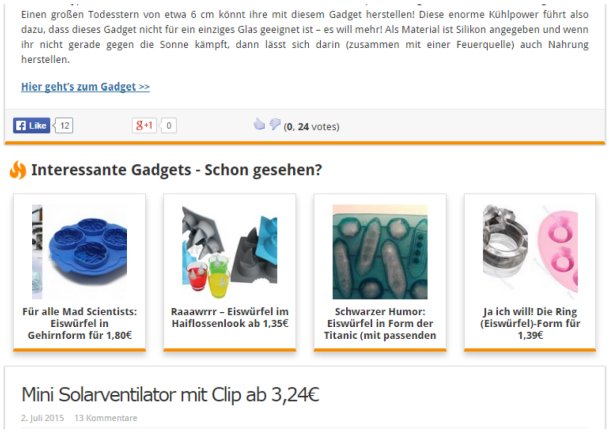

Fig. 1: Screenshot of four recommendations that were displayed below the most recent item of the China-Gadgets feed. In this example, the first item was a rubber ice tray to create ice "cubes" in the form of a Star Wars *Death Star* (not shown in the picture). The recommendations were created with the *Similarity* algorithm and are also rubber ice trays in the form of a brain, a shark fin, the Titanic, and a wedding band.

various types of gadgets. The front page of the China-Gadgets website is organized as a feed of product reviews, where the most recent item reviews are displayed at the top of the list. For each item, a details page exists which contains an in-depth review and a link to an external Chinese online shop.

3.1.1 Experiment Setup

For the purpose of the field study, we extended the existing website design and included recommendation lists at two places. First, we displayed four items below the most recent item, i.e., below the first entry of the feed, which can be seen in the screenshot in Figure 1. Second, we also showed recommendations on each item's detail page using the same layout and design as in Figure 1. Everything else on the website remained unchanged. We implemented the five strategies listed below in Table 5 to fill the recommendation lists.

Table 4: Dataset Characteristics

Characteristic	Value
Users	49k
Items	4k
Views	287k
Purchases	260k
Sessions	226k
Sessions per user	4.63
Views per session	1.94
Purchases per session	1.16
Popularity per item	134.96

During the experiment, which was conducted over the course of three months, the users of the website were randomly assigned to one of five conditions in an A/B test. Visitors were only once added to a group and the assignment was not changed when

Table 5: Brief descriptions of the algorithms used in the randomized field test.

Most Popular	A non-personalized baseline technique that recommends the most popular items of the categories to which the *reference item* belongs. The *reference item* is either the first item of the front page or the item of a detail page for which the recommendations are displayed.
Similarity	Another non-personalized method that recommends items that are "content-wise" similar to the reference item. The similarity between two items is determined by the angle between the TF-IDF-encoded representations of their plain-text item descriptions. We relied on the item descriptions because only a very limited set of item features was available to us.
Recently Viewed	A simple reminding strategy that displays the items that the current visitor has recently inspected in reverse chronological order.
Similarity Personalized	A personalized content-based method where the ranking of the items is determined by their distance to the current user's profile. The user profile is computed as the average TF-IDF vector of the items descriptions that the user has inspected in the past.
BPR	Bayesian Personalized Ranking (BPR) is a learning-to-rank collaborative filtering (CF) method for implicit feedback recommendation scenarios by Rendle et al (2009). To create a personalized item ranking for each user, a generic optimization criterion is optimized that is the maximum posterior estimator in a Bayesian analysis of the ranking task. Training of the model is done by utilizing a bootstrapped, stochastic gradient descent method. In the analysis of our previous work (Jannach et al, 2015a), BPR was the best-performing individual CF method. Contrary to the usual configuration of BPR, the algorithm is allowed to place items in the recommendation list that a user already knows. The BPR models were retrained every 30 minutes and the parameters of the method were optimized in an offline process beforehand.

a user repeatedly visited the site. All item view events and all clicks on links to the corresponding Chinese online shop were recorded.

To be able to measure the effects of personalization, we only considered users who visited the shop at least twice during the data collection period. The resulting dataset that we used in our analysis contained about 287,000 view events by about 49,000 users for over 4,100 products (see Table 4 for more details).

Since we also tested a popularity-based recommendation strategy, we analyzed the popularity distribution of the items (estimated by the number of view events). A distribution where a small set of popular items accumulates most of the view events would favor such a strategy. For the analysis, we grouped the items into 10 popularity levels with 400 items in each bin. Figure 2 shows the outcome of this grouping. The results reveal that the majority of the view events were indeed recorded for a comparably small set of items, and 10% of the items accounted for nearly 90% of all click events in the log data. Hence, there is a long tail of items that received very few clicks. The less popular half of all items, for example, were only involved in 1% of all interactions, with an average of 3.13 view events per item.

3.1.2 Observations

Looking at the general click-through rates across all conditions, about 2.6 % of the view events resulted from a click on an item in a recommendation list. The important business measure for China-Gadgets, however, is how often a visitor actually clicked on a link to the external Chinese retailer. In our analysis, we therefore consider a recommendation to be *successful* only when it was clicked by a user *and* when the user subsequently followed the link to the external website.

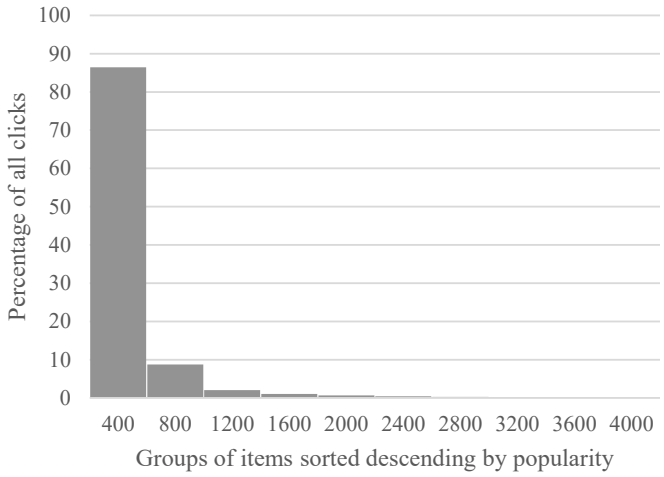

Fig. 2: Percentage of all item view events sorted descending by popularity, and grouped in bins of 400.

The resulting success rates are shown in Figure 3. The simple reminding strategy – to some surprise – actually worked best in terms of the business measure in this domain when compared to the other strategies. The success rate is at 0.34 %, which is significantly higher[13] than the next best strategy, which recommends items that are similar to those that the user has inspected in the past (*Similarity Personalized*). Recommending popular items was also comparably successful. This observation is not too surprising, given that the popularity of items follows a long-tail distribution, which means that it is comparably safe to recommend those view items that are liked by many users. The *BPR* strategy was not particularly successful in our tests, which can at least to some extent be the result of the sparsity of many user profiles. The non-personalized similarity-based approach led to the weakest performance in this comparison.

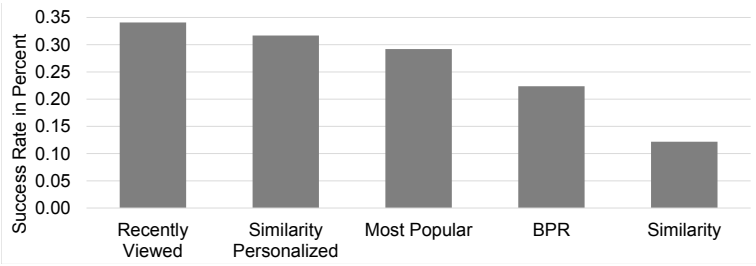

Fig. 3: Success rate of the different recommendation strategies in the China-Gadgets field experiment.

In contrast to typical recommender system evaluation setups, we configured all five algorithms in a way that they could also recommend items that the users already knew, i.e., for which we had observed item view events in the past. Across all configurations, we observed that 38 % of the *successful* recommendations were actually reminders, while only 20 % of *all* item suggestions were known to the users. In particular, the *Similarity*

[13] According to a four field Chi-squared test ($p < 0.05$) with Bonferroni correction.

Table 6: Percentage of reminders in the recommendations lists for all tested methods.

Method	Percentage of reminders
Recently Viewed	100.00 %
Similarity Personalized	82.56 %
BPR	24.95 %
Most Popular	3.33 %
Similarity	0.88 %

Personalized method tended to include a large number of known items in the recommendations (about 82 %) and these reminders are probably part of the success of the method in this experiment. Table 6 shows the percentage of reminders in the recommendations for all tested approaches.

Overall, our study suggests that including reminders in recommendation lists – besides serving as navigation shortcuts – can also be valuable from a business perspective. However, due to the site's structure, the results obtained in this field test have to be interpreted with care. One particularity of the China-Gadgets website, for example, is that new items are displayed in the form of a chronological news feed. For other online shops that follow a more traditional catalog and search-oriented website design, different effects may be observed.

3.2 Adaptive Reminders

The reminding method used in the field test proved to be quite effective despite its trivial nature. One can, however, imagine that simply recommending items in reverse chronological order might not be the best choice in all situations. Consider, for example, that a customer was searching for a pair of shoes and has – after inspecting a number of options – made a purchase in the last session. An intuitive approach when applying a reminding strategy would therefore be to *not* remind the user of shoes anymore in the next session. In the following sections, we will briefly summarize a number of such *adaptive* reminding strategies, which we originally introduced in (Lerche et al, 2016), and present results of offline experiments that were made using the Zalando *frequent* user dataset.

3.2.1 Proposed Strategies

In (Lerche et al, 2016), we proposed four general reminding strategies that have the goal to avoid too obvious recommendations. All strategies take a set of items as an input that the current user has recently interacted with, and re-rank and filter the items according to different goals. The strategies are briefly summarized in Table 7.[14]

Independent of the chosen reminding method, one possible additional strategy is to ignore an item when the user has recently made a purchase in the item's category. We call this the *Feature Filter (FF)* strategy because it filters out potential items to remind that have similar features as a recently purchased item. Technically, the *FF* strategy maintains a blacklist of categories for each user, and this blacklist is extended upon each purchase event. When the user, for example, has recently bought a pair of shoes, no reminders related to shoes will be displayed for some time. However, whenever a user continues to look for items that belong to a blacklisted category *after* the purchase, the category is removed from the blacklist again. Note that the *FF* strategy can be used in combination with any of the four re-ranking strategies proposed above.

[14] More information about the technical details of the algorithms are provided in (Lerche et al, 2016).

Table 7: Adaptive Reminding Strategies

Interaction Recency (IRec)	This method represents a generalized "Recently Viewed" strategy that considers also interactions other than item views (e.g., add-to-wishlist events). The items are ranked in reverse chronological order of the last interaction of the user with them.
Interaction Intensity (IInt)	In this method, the *amount* of recent interactions with a certain item is considered in addition to the last point in time of an interaction. More weight is given to items that the user has inspected more often.
Item Similarity (ISim)	The idea of this method is to remind the user of items that he or she inspected in a past session and which have similar features as the items that were inspected in the current session. If, for example, the user is inspecting shoes in the current session, the strategy ranks up other shoes that the user has inspected in the past. If the user has in the meantime looked up other types of products, e.g., scarves, those will be accordingly ranked lower, as they are from a different item category.
Session Similarity (SSim)	This method also considers the user's current session. In contrast to the *ISim* method, it does however not look for similar items, but for similar sessions. If a user in a past session was inspecting not only shoes but also matching belts, the *SSim* method will remind the user also of belts again, even though he or she has only started looking for shoes in the current session.

As a baseline to compare the reminding strategies with, we included a session-based recommendation method called *C-KNN* in our empirical evaluation. This nearest-neighbor algorithm compares the current session with all past sessions in the training data and then recommends items that appeared in past sessions that are most similar to the current one.

Technically, the algorithm works as follows. To speed up the similarity computations, we encode each session as a bit vector. Each vector element corresponds to an item and a "1" at a certain position means that an interaction with the item was recorded for the session. To quantify the similarity of the current session $\mathbf{c}$ with a historical session $\mathbf{h}$, we compute the cosine similarity $sim_{cos}(\mathbf{c}, \mathbf{h})$ between the vectors. Given the current session $\mathbf{c}$ we determine the k most similar sessions in the training data and denote this set as $mostSimilar(\mathbf{c})$. We then compute a ranking score $score(\mathbf{c}, i)$ for each item i given a current session $\mathbf{c}$ by summing up the similarity values of those sessions in $mostSimilar(\mathbf{c})$ that contain an interaction with i, i.e.,

$$score(\mathbf{c}, i) = \sum_{s \in \text{mostSimilar}(\mathbf{c})} sim_{cos}(\mathbf{c}, \mathbf{h}) \cdot \mathbf{1}_{\text{Interaction(s,i)}} \quad (1)$$

where $\mathbf{1}_{\text{Interaction(s,i)}}$ is an indicator function that returns true when item i appeared in session s. The recommendable items are finally ranked by the resulting scores in descending order.

The described method has proven to be very effective for session-based music recommendation in the past, see, e.g., (Hariri et al, 2012) or (Jannach et al, 2015b). According to additional experiments not reported here, *C-KNN* significantly outperforms more elaborate but session-agnostic methods like *BPR* for this problem setting.[15] The recommendations of the *C-KNN* method – in contrast to the other approaches tested in this setup – are not limited to already known items. This allows us to compare the effectiveness with non-reminding techniques on an absolute scale.

[15] We unfortunately were not yet able to benchmark *C-KNN* in a field study with our external partner.

3.2.2 Empirical Results

Using the generic evaluation protocol for session-based recommendations described in (Jannach et al, 2015a), we conducted a series of experiments on the Zalando and China-Gadgets datasets as well as on an additional dataset that was published in the context of the TMall recommendation competition[16].

According to our evaluation protocol, each recommendation algorithm is provided with a certain number of view events of the beginning of each session in which a purchase was made, and the goal is to predict the item that will be purchased. In addition to the views of the current session, the algorithm can be provided with information about the actions of the same user in the p preceding sessions.

We tested two different configurations of our protocol. The set of recommendable items consists of (a) either only those items that the user has interacted with in the p preceding session or (b) those that the user has interacted with in the p past sessions *plus* all item view events before the purchase in the current session. We call the latter configuration *reveal*, as the most recent views are revealed to the algorithm, and the other configuration *noreveal*. Since the algorithms are assumed to return ranked lists of items, we can use the standard information retrieval measures *hit rate* (i.e., recall given only one relevant item) and Mean Reciprocal Rank (MRR). Precision is proportional to recall in this setup.

Table 8 shows the main results of the empirical analysis for the Zalando frequent user dataset.[17] In this experiment, the possible items for recommendation (reminding) were taken from the last six user sessions ($p = 6$). In our previous work (Lerche et al, 2016), we experimented with different thresholds for the number of past user sessions from which reminders can be selected. Using too many previous sessions led to the inclusion of items that were already outdated. On the other hand, considering too few sessions makes the set of candidates that can be used as reminders too small. Overall, selecting $p = 6$ sessions led to the best results across the tested datasets including the Zalando dataset that is discussed here. In the *reveal* configuration in which the item view events of the current session were revealed to the algorithm, the absolute values, as expected, are generally higher. Users in almost all cases inspect an item before purchasing it. Purchases without item views in the same session only happen when the item was placed in the shopping basket already in a previous session.

Table 8: *Hit Rate@10* (HR) and *MRR@10* results for the Zalando frequent user subset. The best values for each configuration are highlighted in grey. The observations for the other datasets were similar, see also (Lerche et al, 2016).

Configuration	$p = 6$, reveal		$p = 6$, noreveal	
Metric@10	HR	MRR	HR	MRR
IRec	0.653	0.309	0.230	0.069
FF (*SSim*)	0.681	0.296	0.293	0.139
FF (*ISim*)	0.561	0.219	0.353	0.146
SSim ($k = 2$)	0.697	0.327	0.210	0.111
ISim ($k = 20$)	0.588	0.241	0.319	0.137
IInt	0.561	0.217	0.363	0.147
C-KNN	0.268	0.091	0.191	0.063

[16] The competition was organized in the context of IJCAI '15 and TMall is a leading Chinese online market place in the style of Amazon (`https://tianchi.aliyun.com/datalab/dataSet.htm?id=5`).

[17] For the sake of brevity we do not report the detailed results for the other datasets here. The additional results, which are in line with the observations for the Zalando dataset, can be found in (Lerche et al, 2016).

Overall, the results show that all reminding strategies are substantially and statistically significantly better in terms of the hit rate and the MRR than the best-performing baseline *C-KNN*:

- In the *reveal* condition, the "Recently Viewed" *IRec* configuration as expected is hard to beat as usually a view event is recorded some time before the purchase in the same session. Nonetheless, the results show that the *SSim* strategy, which looks for similar past sessions, can slightly outperform the *IRec* method.
- In the *noreveal* configuration, in contrast, the performance of the *IRec* and *SSim* strategies drops in comparison to the other techniques because in this condition they can only remind users of items that they have viewed in preceding sessions. The most effective strategy in this configuration is to consider how *often* an item was inspected by a user in the recent history (*IInt*). The feature filtering method *FF* also proved to be effective in this condition.

3.3 Discussion

The analyses in this section not only confirm the observation made earlier in (Jannach et al, 2015a) that including reminders in recommendations leads to higher accuracy values in offline experiments, but also that it can lead to increased business value in real-world applications. The analyses of the more elaborate strategies from the previous section furthermore indicate that better approaches than just recommending the most recently viewed items in reverse order exist.

How many items of a recommendation lists should be reminders and which strategy one should use to select the reminders largely depends on the specifics of the domain or application. In fact, our observations in Section 2 for the Zalando dataset showed that more than half of the successful recommendations were *not* reminders, so the problem exists to find a balance between the recommendation of already known items and novel item suggestions in practice.

4 Algorithmic Approaches to Improve Session-Based Recommendations

Having analyzed the different success factors for recommendations *ex post* in Section 2, our goal is now to operationalize these insights into new session-based recommendation algorithms that are better able to *predict* the next purchase than previous techniques.

4.1 General Approach – A Two-Phase Item Scoring Method

Our previous works in the field of session-based recommendations showed that considering both long-term preferences patterns and immediate short-term shopping intents can be key to high prediction accuracy in e-commerce scenarios (Jannach et al, 2015c). In the following, we will therefore adopt the same general approach and apply a two-phase item selection and ranking strategy:

- In the first phase, we use different baseline algorithms to compute an initial ranked set of recommendable items. The scoring is either based on long-term preferences only or based on a session-based scheme.

– In the second phase, we post-process the resulting sets and apply a number of re-ranking strategies, which consider the additional success factors for recommendations that were identified in Section 2, including discounts or trends in popularity.[18]

4.2 Baseline Scoring Techniques

We evaluated three baseline scoring techniques in our experiments.

– *BPR*, as it was the most effective context-agnostic baseline method according to the experiments in (Jannach et al, 2015c). Technically, the BPR model is learned on the long-term training data and the recommendations for a given session of the test dataset are only based on the ID of the current user without incorporating further contextual information.
– *C-KNN*, the session-based nearest-neighbor method mentioned above, because it led to the best accuracy values when additional factors like discounts were not considered.
– *C-CoOcc*, a session-based method that computes recommendations of the type "Customers who bought ... also bought ...". Technically, this method determines pairwise item co-occurrence patterns (association rules of size two) in the training data and then recommends those items that most frequently co-occur with the items in the current session. We include this method to assess to which extent our re-ranking methods lead to improvements when a simpler but commonly used baseline technique is employed.

4.3 Heuristic Re-Ranking Strategies

In (Jannach and Ludewig, 2017a) we proposed first approaches to incorporate additional relevance signals into the ranking process. The approaches were based on comparably simple heuristics for each of the potentially relevant success factors (short-term intents, reminders, trends, and discounts) identified earlier in this paper. A brief summary of these heuristic approaches is provided in Table 9.

Table 9: Heuristic Re-Ranking Strategies

Feature Matching (FM)	This strategy takes the user's short-term preferences into account by ranking those items up (given a baseline-ranked set of recommendations) that have common features with items that the user has inspected in the current session. Items with more matching features are ranked higher. This simple scheme proved to be effective according to the analyses in (Jannach et al, 2015a).
Interaction Recency (IRec)	This method, as described above, simply places those items in front of the list that the user has recently inspected. The items are sorted in reverse chronological order (most recent ones first). Items that were not viewed recently by the user are subsequently ranked according to the baseline score.

[18] In our empirical evaluation presented in Section 4.5 the baseline algorithms were used to create a set of 200 recommendations to be re-ranked, which on average led to the best results in terms of the hit rate.

Recent Popularity (RPOP-n)	This strategy considers recent sales trends in the ranking process. We compute a normalized popularity score for each recommendable item for the last n days. To compute the score we simply compute all interactions (views, purchases) of all users with each item in the last n days. For the last day, i.e., the one of the current session, we only count interactions up to the time point when the session started. In the following, we will report the results obtained for *RPOP-1*, i.e., we only consider the popularity of the current day, because this led to the best results according to our experiments.
Discount Promotion (DP)	This method ranks up items that are currently on sale. In our dataset, each purchase event can have one of four discrete *discount levels* assigned. Items with the highest (relative) discount are ranked up. Items with identical discount levels are ranked according to the baseline score.

Our systematic analysis in Section 2 has shown that most of the success factors are not correlated. We therefore implemented and evaluated a number of *weighted hybrid* strategies, which combine the outcomes of the four heuristic re-ranking strategies (*FM*, *IRec*, *RPOP-n*, *DP*). Through a series of experiments, as reported in (Jannach and Ludewig, 2017a), we determined the following mix of a cascading and weighted combination involving three heuristic strategies as the most effective one:

1. We apply the *FM* strategy to retain only items that share at least one feature with any of the items viewed in the most recent sessions.
2. We then calculate the *RPOP* and *DP* scores for these items.
3. The two scores are finally combined in a weighted approach and used for re-ranking.

We denote this combination as *WR(RPOP,DP,0.5)-FM*, where the numerical parameter indicates the relative importance of the two scores returned by *RPOP* and *DP*.

4.4 A Classification-based Approach Using Deep Neural Networks: DeepPredict

A main disadvantage of the presented heuristics-based approaches is that the re-ranking scheme is partially coarse-grained and that a lot of fine-tuning is required to find the right combination of heuristics and the corresponding weights when hybrids are used. We are therefore interested in developing a learning-based technique which is able to consider a variety of signals in parallel and which is able to determine optimal weights automatically.

In this section, we propose such a learning-based approach, which we call DeepPredict. The general approach is similar to the one applied in Section 2 and is based on using the available log data to frame a classification problem. In contrast to Section 2, however, the setup is slightly different. In Section 2, we were interested in determining the *general* importance of different features as predictors for the success of a given recommendation. The goal this time, however, is to predict which item will be purchased given the user's *specific and usually multiple* interactions in the current and previous sessions. Therefore, the problem encoding is different from the one used in Section 2 and is based on slightly different features as will be described below.

The outcomes of the classification task are numerical scores that express the probability that the user will click and purchase a recommended item. In contrast to traditional classification problems, we however do not use these scores to classify the recommendable items but use them to rank items that were identified as possible recommendations by a baseline algorithm, e.g., *C-KNN*.

4.4.1 Feature Engineering, Problem Encoding, and Sampling

Features. As a consequence of the slightly different problem setup, many of the features that we introduced in Section 2 cannot be used anymore. For example, we cannot any

Table 10: Features used in the learning-based approach

Feature	Description	Nb.	Type
Popularity	Normalized popularity of the item in the same day, week, or month	3	Float
Viewed before	True, if the item was viewed before by the user	1	Bool
Views count	Number of previous views of the item by the user	1	Integer
Distance to first view	Distance to the first item view by the user in days or sessions	2	Integer
Distance to last view	Distance to the last view of the item in days or sessions	2	Integer
Brand ratio	Fraction of items of the same brand in the last 1, 2, and 3 sessions	3	Float
Brand popularity	Overall popularity of the brand for the same day, week, or month	3	Float
Color ratio	Fraction of items with the same color in the last 1, 2, and 3 sessions	3	Float
Color popularity	Overall popularity of the color for the same day, week, or month	3	Float
Category ratio	Fraction of items of the same category in the last 1, 2, and 3 sessions	3	Float
Category popularity	Overall popularity of the category for the same day, week, or month	3	Float
Price level ratio	Fraction of items of the same price level in the last 1, 2, and 3 sessions	3	Float
Discount granted?	True, if the item is discounted	1	Bool
Discount level	Level of discount from 0 (no discount) to 3 (high discount)	1	Float

longer compare an item's color with the color of the *currently viewed item* as a binary feature. In fact, in the new problem situation we now have to consider a session-based context with *multiple items of different colors*. We therefore reduced the set of all features from Section 2 to 32 session-based features for the task of predicting the probability of whether or not a certain item will be purchased in the current session.

The set of features that were used in our experiments is shown in Table 10. A substantial number of these features is domain-specific, including for example those related to the color or the brand of an item, which are particularly important in the fashion domain. In general, however, the proposed classification-based approach can be applied for arbitrary domains and other domain-specific features can be included if needed.

Problem encoding and sampling. Differently from the encoding in Section 2, a positive training example is no longer based on the information if a certain item was shown in a recommendation list and then purchased. Instead, a positive training example is derived from the log data for each view event for which we later observed a purchase of the same item in the same or the next session. A view event that did not lead to a purchase is, in contrast, encoded as a negative example.

Since there are far more negative training examples than there are positive ones, for each positive example found in the user sessions we randomly sample exactly one negative example to end up with a balanced dataset.

4.4.2 Model Selection, Optimization, and Application

Model Optimization and Selection. After the feature engineering phase, a set of labeled training examples can be derived from the training part of the user log data.[19] Given

[19] In the experiments reported below we used the *frequent* user sample, as described in Section 2.1, as a basis for learning.

such a dataset, a variety of different supervised machine learning methods can in principle be applied. Using the RapidMiner software suite[20], we tested a number of different techniques, including classification using decision trees, random forests and logistic regression. To determine the best-performing algorithm and its parameters, we systematically searched for the best configuration by applying grid search in combination with ten-fold cross-validation, using again the functionality provided by the RapidMiner software. Furthermore, we applied a genetic feature selection strategy to find an optimal set of features for the given algorithm configurations.

At the end, the best model fit was achieved when testing an optimized artificial neural network with 2 hidden layers, each with 50 nodes, using the implementation of the H_2O open source artificial intelligence library[21], which is available as an add-on component for RapidMiner. We briefly explain the neural network in the next section. The feature selection process for the neural network based approach led to minor improvements when seven of the 32 features were excluded.[22]

A random forest classifier led to the second best accuracy scores (with 23 trees and a maximum depth of 49). For comparison purposes, we accordingly conducted experiments where we re-ranked the recommendations by the scores returned by this classifier. We call this alternative method RFPredict. In the case of random forests, feature selection did not lead to further improvements, which is probably caused by the inherent characteristic of the algorithm to implicitly include only important features in the trees, depending on the number of features and the configured depth of the trees.

Deep Neural Network Encoding The deep learning algorithm provided by H_2O is based on classic feedforward neural networks consisting of multiple layers of multiple nodes, each representing a human neuron as a weighted sum of inputs with a bias b as shown in Figure 4. All these nodes are layer-wise interconnected and the neurons' outputs are transformed with a nonlinear activation function f. In our network (see Figure 5), the generated session-based features form the first layer, and the last layer contains only one node for the probability of a purchase. The in-between layers are called hidden layers (2 layers, each with 50 neurons).

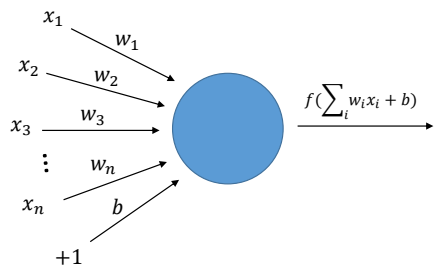

Fig. 4: Neuron

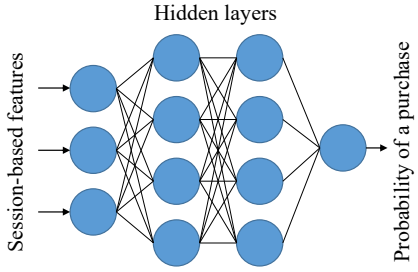

Fig. 5: Network

All weights and biases were optimized in a supervised learning process to minimize the difference between the predictions of the model and the real labels in our examples. The activation function that led to the best results in our model optimization process was the linear rectifier $f(x) = \max(0, x)$ combined with cross entropy

[20] `https://rapidminer.com`

[21] `https://h2o.ai`

[22] The features that were not considered by DeepPredict are listed in the appendix in Table 16. All non-considered features are related to the recent popularity of a certain brand or category.

($L(y,p) = -y\log(p) - (1-y)\log(1-p)$) as the loss function, which is commonly used for binary classification problems.[23] For the optimization of the network, the approach implemented in H_2O applies a parallelized stochastic gradient descent (SGD) method with back-propagation, see (Goodfellow et al, 2016; Candel et al, 2017). Furthermore, the SGD algorithm incorporates different optimizations like regularization, momentum, and an adaptive learning rate (ADADELTA) as discussed by Hinton et al (2012), Zeiler (2012), Sutskever et al (2013), and Wager et al (2013).

Model Application / Recommendation. As was done with the heuristic re-ranking approaches presented in Section 4.3, we applied the neural network-based method DeepPredict and the random forest-based method RFPredict in the form of a post-processing strategy in the recommendation phase.

1. Again, we use different baseline algorithms (e.g., *C-KNN*) to compute an initial ranked set of recommendable items.[24]
2. We consider each of these items as an unlabeled example and generate values for all features from Table 10 given the specific situation of the current session. Given an item to re-rank, we for example compute the fraction of items in the last two sessions that had the same color.
3. Finally, we apply our pre-trained model to all examples to predict if the corresponding item will be purchased in the current session by the user or not. The top items that were returned by the baseline algorithm are then (re-)ranked according to the probability score returned by the classifier.

4.5 Empirical Evaluation

We evaluated the different recommendation strategies on the three subsets of the Zalando dataset, i.e., the frequent, the regular, and the occasional user dataset, again using the evaluation protocol from (Jannach et al, 2015a), and hit rate and MRR as accuracy measures. The reported results are the average after applying a five-fold repeated-subsampling cross-validation scheme. As done in the previous experiments, we set the number of preceding user sessions to six ($p = 6$) and revealed all item views before the purchase in the current session.[25]

The obtained results for the datasets of frequent, regular, and occasional users are shown in Table 11, Table 12, and Table 13, respectively. The best accuracy values for each evaluated baseline are highlighted with a grey background and the overall best result per metric is emphasized in bold font.

4.5.1 Comparison of Baseline Strategies

The performance results of the individual baseline ranking schemes *BPR*, *C-KNN*, and *C-CoOcc* are shown in the first row of each table ("No post-processing")[26]. The *C-KNN* method, as already mentioned in Section 3.2.1, leads to the highest hit rate and MRR among all other baselines on all datasets.

[23] y is the correct label (1 or 0) and p the predicted probability.

[24] As in our previous experiments, we took the first 200 recommendations of the baseline algorithms as a basis for re-ranking as this cut-off value on average led to the best results in terms of the hit rate.

[25] Revealing fewer view items of the current session leads to accuracy values that are lower on an absolute scale; the ranking of the algorithms is however not changed.

[26] All reported differences are statistically significant according to a Student's t-test at $p < 0.01$ with Bonferroni correction.

Table 11: *Hit Rate@10* and *MRR@10* results for the Zalando *frequent* user subset.

Baseline	C-KNN		C-CoOcc		BPR	
Metric@10	HR	MRR	HR	MRR	HR	MRR
No post-processing	0.268	0.091	0.123	0.046	0.062	0.021
FM	0.281	0.093	0.145	0.052	0.119	0.046
IRec-FM	0.306	0.097	0.266	0.096	0.262	0.111
DR-FM	0.316	0.177	0.242	0.120	0.168	0.094
RPOP-FM	0.361	0.187	0.233	0.103	0.216	0.096
RFPredict	0.381	0.248	0.274	0.150	0.241	0.119
WR(RPOP,DR,0.5)-FM	0.382	0.220	0.262	0.121	0.225	0.100
DeepPredict	**0.405**	**0.284**	0.322	0.205	0.301	0.188

Table 12: *Hit Rate@10* and *MRR@10* results for the Zalando *regular* user subset.

Baseline	C-KNN		C-CoOcc		BPR	
Metric@10	HR	MRR	HR	MRR	HR	MRR
No post-processing	0.241	0.087	0.109	0.043	0.152	0.060
FM	0.257	0.091	0.129	0.049	0.197	0.077
IRec-FM	0.296	0.097	0.198	0.076	0.245	0.104
DR-FM	0.275	0.152	0.197	0.095	0.206	0.110
RPOP-FM	0.359	0.192	0.212	0.108	0.280	0.133
RFPredict	0.367	0.216	0.218	0.121	0.302	0.174
WR(RPOP,DR,0.5)-FM	0.373	0.213	0.225	0.115	0.316	0.155
DeepPredict	**0.389**	**0.253**	0.243	0.155	0.316	0.183

Table 13: *Hit Rate@10* and *MRR@10* results for the Zalando *occasional* user subset.

Baseline	C-KNN		C-CoOcc		BPR	
Metric@10	HR	MRR	HR	MRR	HR	MRR
No post-processing	0.405	0.197	0.134	0.053	0.142	0.055
FM	0.430	0.201	0.155	0.060	0.211	0.085
IRec-FM	0.501	0.217	0.235	0.090	0.296	0.125
DR-FM	0.472	0.226	0.209	0.094	0.261	0.137
RPOP-FM	0.501	0.241	0.240	0.118	0.321	0.149
RFPredict	0.500	0.295	0.264	0.159	0.356	0.206
WR(RPOP,DR,0.5)-FM	0.520	0.295	0.263	0.136	0.366	0.186
DeepPredict	**0.532**	**0.322**	0.285	0.177	0.356	0.212

On the dataset of frequent users, the context-aware but comparably simple *C-CoOcc* method significantly outperforms the context-agnostic *BPR* method, which only considers the user's long-term preferences. For the datasets consisting of regular and occasional users that in general have a shorter purchase history, in contrast, the *BPR* method works better than the *C-CoOcc* method.

In principle, one would expect *BPR* to work better for the frequent user dataset because more detailed profiles for each user are available. However, in the considered domain, some of the preference signals might be already outdated at the time of the recommendation. Since *BPR* has no built-in means to give less weight to older interactions, the learned model might focus too much on old interactions. For the dataset of regular or occasional users in contrast, the danger of overfitting to past interactions is less pronounced as the user profiles are less sparse in the first place. In such situations, *BPR*'s tendency to focus on popular items might in addition help to increase the obtained accuracy values, see also (Jannach et al, 2015c).

4.5.2 Considering Short-Term Intents, Reminders, Discounts, and Trends

In the next four rows of Table 11, Table 12, and Table 13, we show the effects of considering additional factors with the help of the heuristic re-ranking strategies presented in Section 4.3. The results show that each individual signal can measurably contribute to an accuracy increase.

(i) Applying Feature Matching (*FM*) as a post-processing strategy improves the accuracy results for every baseline and for all datasets. In the next set of measurements, we therefore always apply the *FM* method on top of the baseline ranking.
(ii) Adding reminders on top of the baselines and the *FM* strategy (*IRec-FM*) consistently leads to further increases in terms of the hit rate and the MRR. A similar effect was reported in (Jannach et al, 2015a) for different baselines. Nonetheless, while increases in accuracy can be achieved through reminders, focusing too much on reminders can be of limited business value, as such recommendations are not suited to point users to unknown but relevant items.
(iii) Next, we examine the effect of focusing the recommendations on items that are currently on sale (*DR-FM).* The results show that this strategy proves effective and significantly more accurate when compared to the *FM* baselines across all the configurations. The *DR-FM* strategy is however not consistently better than the *IRec-FM* reminding strategy for all datasets and accuracy measures.
(iv) The *RPOP-FM* strategy, which focuses on recent trends, finally leads to the best accuracy results so far. It significantly outperforms all other methods that combine Feature Matching with one additional signal on the *regular users* dataset. It is also better for the *frequent users* dataset, except when the *C-CoOcc* strategy is used as a baseline. In this measurement, we report the results of the *RPOP-1* strategy, i.e., we only consider the popularity of the items on the day when the recommendations are made. Considering longer-term trends is also helpful, but the best results in terms of the hit rate and MRR were achieved when only the last day was considered.

4.5.3 Leveraging Multiple Signals in Parallel

We tested various weighted combinations of our heuristic re-ranking approaches. Our analyses in Section 2 showed limited correlations between most success indicators, and we in fact achieved the best performance with the heuristic approaches when combining three of the factors in the method *WR(RPOP,DR,0.5).*

With the learning-based approaches DeepPredict and RFPredict we furthermore tried to consider all of these factors in a more elegant and automated way. And, in fact, applying a neural network in our DeepPredict approach in combination with the *C-KNN* method led to the overall best results in our experiments for all three datasets. The difference between the best-performing method and any other method was statistically significant at $p < 0.05$ according to a Student's t-test with Bonferroni correction.

The random forest based method RFPredict was in some test configurations slightly better than the fine-tuned weighted hybrid approach. In several other situations, we could however not find a parameter setting for the random forest classifier that led to an improvement over the heuristics-based hybrid. A possible reason for this observation might lie in the more coarse-grained predictions of the purchase probabilities of this method when compared with the neural network based method.

Overall, our results show that re-ranking the recommendations by considering a number of additional session-based factors can lead to significant accuracy gains. Furthermore, the experiments provide further evidence that modern deep learning approaches can help to obtain better results in this domain than we could achieve with more traditional methods like random forests or with manually tuned weight factors.

To further validate our observations, we have conducted additional experiments on different subsamples as described in Section 2. The datasets include a set of 3,000 *random* users who made at least one purchase, and two larger subsamples of *regular* users, involving 5,000 and 10,000 regular users, respectively. The results confirm the observations that were made in this section, i.e., that the best results can be obtained with the DeepPredict method, i.e., the effectiveness of the method is not limited to a certain type of users. The dataset characteristics and the detailed results are listed in the appendix (Tables 17 and 18).

4.6 Practical Implications

Our work has a number of practical implications. First, the work presented in this paper in general sheds light on the relative importance of considering long-term preference models and the short-term situational aspects in real-world e-commerce recommendation processes. Our results show that considering, for example, the user's short-term interests can lead to substantially more effective recommendations, i.e., to recommendations that ultimately lead to a purchase, than when considering only long-term models that are re-trained overnight. From a practical perspective it is interesting to see that already quite simple heuristic methods, which can be applied to re-rank the recommendation lists in real-time, are comparably effective. Furthermore, these heuristic methods can be used as a fallback method for anonymous or one-time users for which no long-term user profiles exist.

Second, our research works provide evidence that recommending items that the user has already seen can be a promising strategy in some domains, and we have proposed novel heuristics of how to select the items that the user should be reminded of. Practitioners should therefore consider and evaluate the inclusion of reminder items in their recommendations. In practical environments, the selection of the reminder items and how many of the items of a recommendation list should be reminders depends on the specifics of the domain and has to be determined based on business considerations or A/B tests. The analysis of the log data that was available to us indicates that at least some companies include a substantial amount of reminders in the recommendations.

Third, our analyses showed that customers of the analyzed shop click more often on recommendations when they relate to recently trending items or items that are currently discounted. Regarding the discounts, this means that users generally have no negative connotations when seeing recommendations that are on sale. They seemingly do not consider them as being less relevant, which might be the case for other advertisements on the shop website. Whether or not increased sales of already trending or discounted items are desirable in practice depends on the specifics of the business and the intended purpose of the recommendation service. If more revenue is the goal, selling more, even at reduced prices, is often better. If in contrast profit is the target business metric, promoting items with higher margins (including not-discounted ones), might be better. Similar considerations apply for the recommendation of already popular and trending items.

Fourth, our work shows that "reverse engineering" the most important features of successful recommendations from log data and using the features in a recommendation algorithm can be a promising approach in practice. Our work revealed some features that are probably relevant and useful in many e-commerce scenarios, like trends and discounts. In practice, often much more fine-grained information about the items or the contextual situation of the user is available, which can be used in the feature engineering phase. Considering a variety of such features in parallel can furthermore "automatically" result in diverse recommendation lists that represent a balanced mix of items that match

the user's long-term or short-term preferences, items the user has recently inspected, or items that sell well at the moment. The proposed prediction method is in general not limited to a certain set of features or specific learning method. In fact, our results showed that already the comparably simple weighted hybrid method can lead to good results. In practice, however, the optimal weights have to be fine-tuned based on offline experiments or through field tests in case there exists no suitable offline proxy for the target business metric. The proposed deep learning approach has a higher predictive power in offline experiments. In practice, however, one has to thoroughly analyze if the additional computational complexity of such a method justifies its usage.

5 Related Works

Historically, many papers in the field of recommender systems, like ours, target e-commerce domains, including the recommendation of books to purchase, movies to rent, or other types of goods to buy. Differently from many works in the literature, our work is not based on the matrix completion problem formulation, but addresses session-based recommendation scenarios, where the goal is to predict the users' next action(s) based on their most recent behavior. It therefore also falls into the category of *context-aware* recommender systems, where the context – in our case the user's short-term shopping intent – has to be inferred from the user's recent interactions with the website.

According to the categorization scheme for context-aware recommenders proposed by Adomavicius and Tuzhilin (2011), the techniques proposed in our work can be assigned to the category of *contextual post-filtering*. Post-filtering means that we initially create a list of items that are potentially relevant for a given user in general and then reorganize this list based on the currently given contextual situation.

Within the category of context-aware recommenders itself, our work falls into what can be called "session-based" or "session-aware" recommendation approaches. In session-based recommenders, the algorithm is typically given information about the last few interactions of the user with the system and the problem is to predict a certain future event, e.g., the next item view or purchase event, in the given session. "Next-basket recommendation", as discussed, e.g., in (Rendle et al, 2010), represents a specific form of this problem in e-commerce settings. Session-based next-item recommendations are however also common in the music domain in the form of playlist generation problems (Hariri et al, 2012; Bonnin and Jannach, 2014) or in the context of the provision of automated website navigation aids (Mobasher et al, 2002).

Compared to the huge amount of works on non-contextualized recommendations based on public datasets from MovieLens, Yahoo! or Netflix, works on session-based recommendation problems are comparably scarce. Early works in this area were published mostly in the field of *interactive* and *knowledge-based* recommender systems. In such approaches, the users are typically asked directly about their short-term preferences, e.g., using interactive queries or critiquing-based approaches (Ricci et al, 2003; Burke, 2000).

An example of an early *learning-based approach* for the problem of modeling short-term intents and predicting the next user action is the work by Mobasher et al (2002). Their goal was to predict the next navigation actions of users on a websites. Technically, sequential pattern mining was proposed as one solution to find patterns in past interaction logs. Later on, related works on the extraction of repeated navigation patterns from log data were proposed by Aghabozorgi and Wah (2009) and AlMurtadha et al (2010). The *C-CoOcc* method used in our experiments can be seen as a simple form of pattern mining. Sequential patterns for next-item recommendation were also evaluated for the music recommendation domain by Bonnin and Jannach (2014). In their work, it how-

ever turned out that sequential patterns were not as effective as a neighborhood-based method.

Session-based recommendations are also discussed in the domain of personalized news. Here, the recency of news items is crucial for successful recommendations, as the information gets irrelevant quickly. Therefore, focusing of the short-term interests of individual users or user groups becomes highly important. For example, in (Garcin et al, 2013) the authors create a news recommender with context trees that incrementally recommends news articles based on the current click stream of the users. The approach is therefore applicable for anonymous users that are not logged in and have no long-term profile. Similar to that work, Li et al (2010) utilize contextual bandits to generate recommendations for *Yahoo! News* based on the user's visited news article pages. For each page a user visits, the system sequentially sends new recommendations to the user and is therefore able to detect their short-term interests. A different news recommendation approach was proposed for the *Google News* platform by Liu et al (2010). The authors present a hybrid approach that uses content-based information about news items that match the user's interests, but also employs a collaborate filtering algorithm that detects the short-term trends based on the interactions of a user's neighborhood.

Another intuitive form of considering session-based next-item recommendation problems from a machine learning perspective is to view them as sequential optimization problems and to model them as Markov Decision Processes (Shani et al, 2005). Later papers that considered Markov processes or Markov chains for next-item recommendation problems include the works described in (Rendle et al, 2010) and (Tavakol and Brefeld, 2014) for the e-commerce and fashion domains or (Chen et al, 2012) for the music domain. In principle, these approaches can be used as alternative baseline techniques in our two-phase re-ranking scheme. In contrast to our approach, which considers several additional factors like reminders or recency aspects, the above-mentioned works mostly focus on determining those items that match the current intent of the session. In some cases, Markov process based approaches can suffer from scalability issues and evidence exists that in some domains neighborhood-based methods are at least equally effective in terms of their prediction accuracy (Bonnin and Jannach, 2014).

Recurrent neural networks (RNNs), as a special form of artificial neural networks, represent another machine-learning approach to model the next-item prediction problem. Such approaches recently gained popularity in the context of *deep learning* models. Examples of works that aim to learn the dynamic temporal behavior of users from log data with RNNs can be found, e.g., in (Romov and Sokolov, 2015) or (Hidasi et al, 2016). In contrast to our approaches, the mentioned works focus solely on short-term models, i.e., their predictions are only based on the actions of the current session. Again, these methods can in principle be used as alternatives baselines for our re-ranking models that are able to consider other aspects in the recommendation process. Recent work however revealed that despite the computational complexity of these models, they are not always favorable in terms of prediction accuracy over the *C-KNN* method used in our work (Jannach and Ludewig, 2017b). Another approach using RNNs for next-basket predictions was recently proposed in (Yu et al, 2016). Their method is capable of considering multiple sessions of a user over time, but only examines past checkout events and not a multitude of relevance signals as done in our approaches.

A number of alternative modeling techniques to deal with short-term and long-term interests was proposed in the literature. An early knowledge-based and "scenario-based" method was introduced in (Shen et al, 2007) to understand the user's immediate shopping needs in the fashion domain. Approaches to combine short-term interest models and long-term models were put forward, for example, by Anand and Mobasher (2007) or Nguyen and Ricci (2008). More recently, Hariri et al (2014) relied on a multi-arm bandit algorithm to consider the user's short-term goals and to discover possible interest shifts.

Finally, in the context of the ACM RecSys 2015 challenge, where the task was to predict whether or not a purchase will be made in a session and which item will be purchased, Romov and Sokolov (2015) used gradient boosting and decision trees to make predictions in a two-stage classification process. Similar to the DeepPredict method proposed in our work, Romov and Sokolov (2015) designed a number of features to predict the next item using a classification-based approach. In our work, we however consider additional types of features like reminders that were not in the focus of previous research. Moreover, the experiments on our datasets showed that artificial neural networks were able to outperform classifiers based on random forests.

Generally, considering *reminders* within recommendation lists has not been in the focus of the recommender systems research to a large extent. In an early work, Prassas et al (2001) proposed to remind users of online shopping sites of products during the "checkout" process ("Don't forget to buy"); Plate et al (2006) later on developed a mobile shopping assistant which suggests known items that are related to the products that the user is currently inspecting. Both mentioned articles are examples of works that discuss the potential value of reminding users of known things. However, no algorithmic approaches to select the items to recommend are proposed.

Recommending known items for *repeated consumption* was for example discussed by Anderson et al (2014) and Kapoor et al (2015). In the work of Anderson et al (2014), the recency of past interactions with a given product was considered as the best indicator for repeated consumption. Correspondingly, a recommendation model for known items was designed that incorporates the time of an item's last consumption with an exponential decay factor. Kapoor et al (2015) focus on music recommendations where repeated consumptions occur more often than in the online shopping domain. In their work the authors present a method that estimates, based on the current context, whether a user is in the mood for listening to new tracks or not. Overall, while the work of Anderson et al (2014) has similarities with our reminding strategies, we see the integration of the ideas of Kapoor et al (2015) as a potential future extension to our reminding approaches. Repeated purchases of the *exact* same item are however not very common in our dataset from the fashion domain and it is not fully clear yet if the findings of Kapoor et al (2015) that were made in the music domain generalize to this domain.

Reminders however may not only serve as a means to point users to items that were of interest in the past, they can also represent a form of *navigation shortcuts*, e.g., when users repeatedly inspect the different items of their current choice set. To which extent navigation shortcuts ("shortlists") are adopted by users was in the focus of a study by Schnabel et al (2016). Their work revealed that users in fact heavily relied on the functionality of a manual shortlist creation tool that was made available to them during the study. For cases in which such a functionality is not available, Close and Kukar-Kinney (2010) in an earlier work discovered that some online users tend to misuse the shopping basket for the purpose of managing the candidate products.

Finally, regarding the question of how to consider (short-term) popularity trends and whether or not to recommend discounted items, limited academic research exists so far. Recommending *generally popular* items is a common approach in cold-start situations, even though such recommendations often do not lead to the best results in terms of the business value (Adomavicius and Tuzhilin, 2005; Jannach and Hegelich, 2009). Considering items that are *currently popular* on the site was recently discussed by Padmanabhan et al (2015) and Gomez-Uribe and Hunt (2015) in the context of the movie and TV show recommendations provided by Netflix. Besides the consideration of recent trends, Gomez-Uribe and Hunt (2015) state that their algorithms generally rely on a "pretty healthy dose of (unpersonalized) popularity". In the academic environment, in contrast, the recommendation of popular items is often considered as being of limited value. More research is therefore required to understand in which domains and under

which circumstances the recommendation of known and generally popular items can contribute value to the business.

With respect to the recommendations of discounted items, to our knowledge very limited academic research has been published in the computer science literature until now. The topic of *dynamic* and *personalized pricing*, which is extensively discussed, e.g., by Choudhary et al (2005), is generally to some extent related. Werro et al (2005) for example propose a method for the generation of personalized discounts to retain promising customers and increase the willingness to purchase products. Personalized prices were also in the focus of the later work by Kamishima and Akaho (2011), who propose to dynamically adjust the prices in the context of a recommendation system. Research in this field is unfortunately hampered by the lack of real-world datasets, and works like the one by Kamishima and Akaho (2011) can currently only be based on synthetic datasets and simulations.

6 Summary, Research Limitations, and Future Works

The goal of our work was to contribute to a better understanding of factors that can make e-commerce recommendations successful in practice. Specifically, we could show that including already known, trending, and currently discounted items within the recommendations of an e-commerce site can be useful. These aspects, as well as the consideration of the user's short-term shopping intents, are so far little explored in the literature, partially due to the lack of publicly available datasets.

Our technical approach is based on the systematic analysis of characteristics of successful real-world recommendations, which to our knowledge has not been done in the e-commerce domain before. The obtained insights helped us engineer new methods that consider various signals in parallel and lead to higher prediction accuracy than previous methods. The relevance of specific features for the prediction task might be domain-specific, the general research approach is however generic and can be applied in other domains as well. We identified the features that we used in our analyses based on the research literature, based on observations from practical systems, and based on general considerations regarding the behavior of markets. Depending in particular on the application domain, certainly also other factors can exist that could be considered in addition to those discussed in our work.

So far, we could test our integrated deep learning model only in one application domain for which we had a dataset available that contained all relevant information including the pricing. Individual aspects of our approach were however already validated for different datasets and domains. The reminders were for example evaluated in a field test as described above. The importance of considering short-term intents was analyzed for a second e-commerce dataset (TMall) in (Jannach et al, 2015a). Using the TMall dataset, we could also validate that recommending items that were popular in the last few days is a better strategy than to recommend those items that were the most popular ones during the entire data collection period.

More research is however still required in different directions. First, while the general approach of deriving characteristics of successful recommendations from log data is applicable in domains other than e-commerce, it is not clear if the specific aspects investigated in this paper also play an important role in different scenarios. Recommending discounted items can for example be explored in other domains as well where the goal of the recommendations is to stimulate purchases. It is furthermore intuitive to assume that reminders and current trends should be factors to be considered for example in the music recommendation domain. However, determining a suitable moment to remind users of songs they heard in the past might be more challenging as short-term musi-

cal preferences can be influenced by a number of factors including the user's mood or current willingness to explore new things (Kapoor et al, 2015).

To address such issues, in general better methods are still needed to automatically assess the user's current intent and motivation to visit the site. Furthermore, a common challenge in practical applications also outside the e-commerce domain is to find the optimal balance between the recommendation of novel items, generally trending items, and reminders. Placing these different sets of items in separate recommendation lists is not uncommon on real-world e-commerce platforms. Barely any research on multiple recommendation list exists in the literature, even though this appears to be a problem that is relevant in practice (Gomez-Uribe and Hunt, 2015).

Finally, the question regarding the true business value of different recommendation strategies can probably only be answered in the context of a specific application. Our work shows at least for the case of reminders that they led not only to better results in the offline experiments, but also to a measurable increase in terms of the business metric on the e-commerce site on which we could run a field test.

References

Adomavicius G, Tuzhilin A (2005) Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. Transactions on Knowledge and Data Engineering 17(6):734–749

Adomavicius G, Tuzhilin A (2011) Context-aware recommender systems. In: Recommender Systems Handbook, Springer, pp 217–253

Aghabozorgi SR, Wah TY (2009) Recommender systems: Incremental clustering on web log data. In: Proceedings of the 2Nd International Conference on Interaction Sciences: Information Technology, Culture and Human, ICIS '09, pp 812–818

AlMurtadha Y, Sulaiman NB, Mustapha N, Udzir NI, Muda Z (2010) ARS: Web page recommendation system for anonymous users based on web usage mining. In: Proceedings of the European Conference of Systems, and European Conference of Circuits Technology and Devices, and European Conference of Communications, and European Conference on Computer Science, ECS'10/ECCTD'10/ECCOM'10/ECCS'10, pp 115–120

Anand S, Mobasher B (2007) Contextual recommendation. In: From Web to Social Web, Springer, pp 142–160

Anderson A, Kumar R, Tomkins A, Vassilvitskii S (2014) The dynamics of repeat consumption. In: Proceedings of the 23rd International Conference on World Wide Web, WWW '14, pp 419–430

Berry S, Levinsohn J, Pakes A (1995) Automobile prices in market equilibrium. Econometrica 63(4):841–890

Bonnin G, Jannach D (2014) Automated generation of music playlists: Survey and experiments. ACM Computing Surveys 47(2):26:1–26:35

Burke R (2000) Knowledge-based recommender systems. Encyclopedia of Library and Information Science 69(32):180–200

Candel A, Parmar V, LeDell E, Arora A (2017) Deep learning with H2O. `http://docs.h2o.ai/h2o/latest-stable/h2o-docs/booklets/DeepLearningBooklet.pdf`, accessed 17 August 2017

Chen S, Moore JL, Turnbull D, Joachims T (2012) Playlist prediction via metric embedding. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, pp 714–722

Choudhary V, Ghose A, Mukhopadhyay T, Rajan U (2005) Personalized pricing and quality differentiation. Management Science 51(7):1120–1130

Close AG, Kukar-Kinney M (2010) Beyond buying: Motivations behind consumers' online shopping cart use. Journal of Business Research: Advances in Internet Consumer Behavior & Marketing Strategy 63(9–10):986–992

Diehl K, van Herpen E, Lamberton C (2015) Organizing products with complements versus substitutes: Effects on store preferences as a function of effort and assortment perceptions. Journal of Retailing 91(1):1–18

Garcin F, Dimitrakakis C, Faltings B (2013) Personalized news recommendation with context trees. In: Proceedings of the 7th ACM Conference on Recommender Systems, RecSys '13, pp 105–112

Garcin F, Faltings B, Donatsch O, Alazzawi A, Bruttin C, Huber A (2014) Offline and online evaluation of news recommender systems at swissinfo.ch. In: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14, pp 169–176

Gomez-Uribe CA, Hunt N (2015) The Netflix recommender system: Algorithms, business value, and innovation. Transactions on Management Information Systems 6(4):13:1–13:19

Goodfellow I, Bengio Y, Courville A (2016) Deep Learning. MIT Press, `http://www.deeplearningbook.org`, accessed 17 August 2017

Hariri N, Mobasher B, Burke R (2012) Context-aware music recommendation based on latent topic sequential patterns. In: Proceedings of the Sixth ACM Conference on Recommender Systems, RecSys '12, pp 131–138

Hariri N, Mobasher B, Burke R (2014) Context adaptation in interactive recommender systems. In: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14, pp 41–48

Hidasi B, Karatzoglou A, Baltrunas L, Tikk D (2016) Session-based recommendations with recurrent neural networks. In: Proceedings of the International Conference on Learning Representations, ICLR '16

Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov R (2012) Improving neural networks by preventing co-adaptation of feature detectors. CoRR abs/1207.0580

Jannach D, Adomavicius G (2016) Recommendations with a purpose. In: Proceedings of the 10th ACM Conference on Recommender Systems, RecSys '16, pp 7–10

Jannach D, Hegelich K (2009) A case study on the effectiveness of recommendations in the mobile internet. In: Proceedings of the 3rd ACM Conference on Recommender Systems, RecSys '09, pp 205–208

Jannach D, Ludewig M (2017a) Determining characteristics of successful recommendations from log data – a case study. In: Proceedings of the Symposium on Applied Computing, SAC '17, pp 1643–1648

Jannach D, Ludewig M (2017b) When recurrent neural networks meet the neighborhood for session-based recommendation. In: Proceedings of the 11th ACM Conference on Recommender Systems, RecSys '17, p (forthcoming)

Jannach D, Lerche L, Jugovac M (2015a) Adaptation and evaluation of recommendations for short-term shopping goals. In: Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, pp 211–218

Jannach D, Lerche L, Kamehkhosch I (2015b) Beyond "hitting the hits" – generating coherent music playlist continuations with the right tracks. In: Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, pp 187–194

Jannach D, Lerche L, Kamehkhosh I, Jugovac M (2015c) What recommenders recommend: an analysis of recommendation biases and possible countermeasures. User Modeling and User-Adapted Interaction 25(5):427–491

Kamishima T, Akaho S (2011) Personalized pricing recommender system: Multi-stage epsilon-greedy approach. In: Proceedings of the 2Nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems, HetRec '11, pp 57–64

Kapoor K, Kumar V, Terveen L, Konstan JA, Schrater P (2015) "I like to explore sometimes": Adapting to dynamic user novelty preferences. In: Proceedings of the 9th ACM Conference on Recommender Systems, RecSys '15, pp 19–26

Lerche L, Jannach D, Ludewig M (2016) On the value of reminders within e-commerce recommendations. In: Proceedings of the 2016 Conference on User Modeling Adaptation and Personalization, UMAP '16, pp 27–25

Li L, Chu W, Langford J, Schapire RE (2010) A contextual-bandit approach to personalized news article recommendation. In: Proceedings of the 19th International Conference on World Wide Web, WWW '10, pp 661–670

Liu J, Dolan P, Pedersen ER (2010) Personalized news recommendation based on click behavior. In: Proceedings of the 15th International Conference on Intelligent User Interfaces, IUI '10, pp 31–40

Manning CD, Raghavan P, Schütze H (2008) Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA

Mobasher B, Dai H, Luo T, Nakagawa M (2002) Using sequential and non-sequential patterns in predictive web usage mining tasks. In: Proceedings of the 2002 IEEE International Conference on Data Mining, ICDM '02, pp 669–672

Moe WW (2003) Buying, searching, or browsing: Differentiating between online shoppers using in-store navigational clickstream. Journal of Consumer Psychology 13(1):29–39

Nguyen QN, Ricci F (2008) Long-term and session-specific user preferences in a mobile recommender system. In: Proceedings of the 13th International Conference on Intelligent User Interfaces, IUI '08, pp 381–384

Padmanabhan P, Sadekar K, Krishnan G (2015) What's trending on Netflix? URL `https://medium.com/netflix-techblog/whats-trending-on-netflix-f00b4b037f61`, accessed 17 August 2017

Plate C, Basselin N, Kröner A, Schneider M, Baldes S, Dimitrova V, Jameson A (2006) Recomindation: New functions for augmented memories. In: Proceedings of the 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems, AH '06, pp 141–150

Prassas G, Pramataris KC, Papaemmanouil O, Doukidis GJ (2001) A recommender system for online shopping based on past customer behaviour. In: Proceedings of the 14th BLED Electronic Commerce Conference, BLED '01, pp 766–782

Rendle S, Freudenthaler C, Gantner Z, Schmidt-Thieme L (2009) BPR: Bayesian personalized ranking from implicit feedback. In: Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence, UAI '09, pp 452–461

Rendle S, Freudenthaler C, Schmidt-Thieme L (2010) Factorizing personalized markov chains for next-basket recommendation. In: Proceedings of the 19th International Conference on World Wide Web, WWW '10, pp 811–820

Ricci F, Venturini A, Cavada D, Mirzadeh N, Blaas D, Nones M (2003) Product recommendation with interactive query management and twofold similarity. In: Proceedings of the 5th International Conference on Case-Based Reasoning, ICCBR '03, pp 479–493

Romov P, Sokolov E (2015) Recsys challenge 2015: Ensemble learning with categorical features. In: Proceedings of the 2015 International ACM Recommender Systems Challenge, RecSys '15 Challenge, pp 1:1–1:4

Schnabel T, Bennett PN, Dumais ST, Joachims T (2016) Using shortlists to support decision making and improve recommender system performance. In: Proceedings of the 25th International Conference on World Wide Web, WWW '16, pp 987–997

Shani G, Heckerman D, Brafman RI (2005) An MDP-Based Recommender System. Journal of Machine Learning Researh 6:1265–1295

Shen E, Lieberman H, Lam F (2007) What am I gonna wear?: Scenario-oriented recommendation. In: Proceedings of the 12th International Conference on Intelligent User Interfaces, IUI '07, pp 365–368

Sutskever I, Martens J, Dahl G, Hinton G (2013) On the importance of initialization and momentum in deep learning. In: Proceedings of the 30th International Conference on International Conference on Machine Learning, ICML '13, pp 1139–1147

Tavakol M, Brefeld U (2014) Factored MDPs for detecting topics of user sessions. In: Proceedings of the 8th ACM Conference on Recommender Systems, RecSys '14, pp 33–40

Wager S, Wang S, Liang P (2013) Dropout training as adaptive regularization. In: Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS'13, pp 351–359

Werro N, Stormer H, Meier A (2005) Personalized discount - a fuzzy logic approach. In: Proceedings of the 5th IFIP Conference on e-Commerce, e-Business, and e-Government, I3E '05, pp 375–387

Yu F, Liu Q, Wu S, Wang L, Tan T (2016) A dynamic recurrent model for next basket recommendation. In: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16, pp 729–732

Zeiler MD (2012) ADADELTA: an adaptive learning rate method. CoRR abs/1212.5701

Author Biographies

Dr. Dietmar Jannach is a Professor of Computer Science at TU Dortmund, Germany and head of the department's e-services research group. Dr. Jannach has worked on different areas of artificial intelligence, including recommender systems, model-based diagnosis, and knowledge-based systems. He is the leading author of a textbook on recommender systems and has authored more than hundred technical papers, focusing on the application of artificial intelligence technology to practical problems.

Malte Ludewig is a PhD candidate in Computer Science at TU Dortmund, Germany, from where he also received his MSc degree. His research interests lie in the field of recommender systems – with a focus on session-based recommendations – and personalization in e-commerce environments in general.

Dr. Lukas Lerche obtained his PhD and MSc degrees from TU Dortmund, Germany. During his doctoral studies Dr. Lerche worked on different research problems in the field of recommender systems. In his research publications, he mostly focused on recommendations based on implicit feedback and on session-based recommendation scenarios in e-commerce.

Appendix

A Numbers of Purchases per User

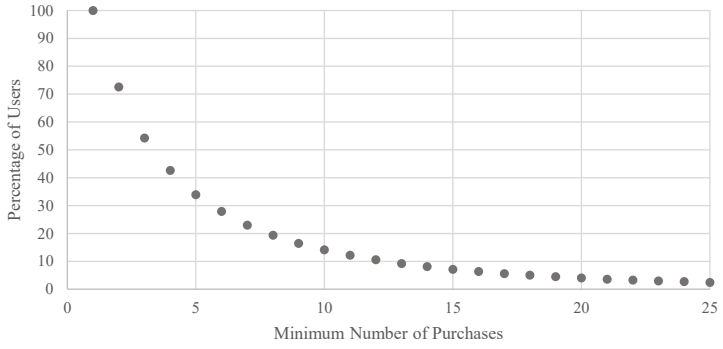

Fig. 6: The figure visualizes the purchase frequency distribution in the raw dataset, considering only users who ever made a purchase during the data collection period. The X-axis represents the minimum number of past purchases and on the Y-axis, the percentage of users in the dataset is shown that are above this threshold. For example, about one third of all users made 5 or more purchases.

B Feature Weights for the Occasional Users

Table 14: Results of the statistical feature weight analysis for the *occasional* user subset.

Gain Ratio analysis	
Feature	Weight
Discount level	0.453
Any discount granted?	0.325
Current popularity (day)	0.294
Current popularity (week)	0.231
Viewed before?	0.191
Distance to first view (in days)	0.191
Distance to first view (in sessions)	0.191
Current popularity (month)	0.174
Distance to last view (in sessions)	0.165
Distance to last view (in days)	0.157

Chi Squared analysis (normalized)	
Feature	Weight
Current popularity (day)	1.000
Current popularity (week)	0.788
Any discount granted?	0.762
Discount level	0.762
Current popularity (month)	0.544
Number of previous views	0.371
Distance to last view (in days)	0.370
Distance to last view (in sessions)	0.369
Viewed before?	0.364
Distance to first view (in days)	0.364

C Examined Features

Table 15: The full list of the examined features (see Section 2.3) along with their type and a short explanation.

Feature Name	Type	Explanation
clicked	Label	Recommended item was clicked
clicked_wished	Label	Recommended item was clicked and added to the wish list
clicked_cart	Label	Recommended item was clicked and added to the cart

clicked_bought *Successful Recommendation*	Label	Recommended item was clicked and bought in the same or the next session
relpop_{day,week,month} *Current popularity (day,week,month)*	Numerical	Popularity of the item on the same day, in the same week, or in the same month
samebrand	Bool	Same brand as the currently viewed item
brandratio_session{1,2,3}	Numerical	Ratio of the recommended brand regarding actions in the last 1, 2, or 3 sessions
brandpop	Numerical	Overall popularity of the brand
brandpop_{day,week,month}	Numerical	Popularity of the brand on the same day, in the same week, or in the same month
samecolor	Bool	Same color as the currently viewed item
colorratio_session{1,2,3}	Numerical	Ratio of the recommended color regarding actions in the last 1, 2, or 3 sessions
colorpop	Numerical	Overall popularity of the color
colorpop_{day,week,month}	Numerical	Popularity of the color on the same day, in the same week, or in the same month
samecat_{1,2,3,4}	Bool	Same category as the viewed item on breadcrumb navigation level 1, 2, 3, or 4
catratio_session{1,2,3}_{1,2,3,4}	Numerical	Ratio of the recommended category (breadcrumb navigation level 1, 2, 3, or 4) regarding actions in the last 1, 2, or 3 sessions
catpop_{1,2,3,4}	Numerical	Overall popularity of the category on breadcrumb navigation level 1, 2, 3, or 4
catpop_{day,week,month}_{1,2,3,4}	Numerical	Popularity of the category on the same day, in the same week, or in the same month for breadcrumb navigation level 1, 2, 3, or 4
sameprice	Bool	Same price level as the currently viewed item
priceratio_session{1,2,3}	Numerical	Ratio of the recommended price level regarding actions in the last 1, 2, or 3 sessions
similarity_viewed	Numerical	Ratio of features matched with the currently viewed item
similarity_session{1,2,3}	Numerical	Average ratio of features matched with items from the last 1, 2, or 3 sessions
neighbors_color	Numerical	Ratio of neighbor recommendations with the same color
neighbors_brand	Numerical	Ratio of neighbor recommendations with the same brand
neighbors_price	Numerical	Ratio of neighbor recommendations with the same price level
neighbors_category_{1,2,3,4}	Numerical	Ratio of neighbor recommendations with the same category (breadcrumb navigation level 1, 2, 3, or 4)
neighbors_distance	Numerical	Average ratio of item features matched with neighbor recommendations
prevrecclicks_sim	Numerical	Average ratio of item features matched with previously clicked recommended items in a session
prevrecclicks_color	Numerical	Average ratio of matching colors with previously clicked recommended items in a session
prevrecclicks_brand	Numerical	Average ratio of matching brands with previously clicked recommended items in a session
prevrecclicks_cat_{1,2,3,4}	Numerical	Average ratio of matching categories with previously clicked recommended items in a session (breadcrumb navigation level 1, 2, 3, or 4)
boughtbefore_sim	Numerical	Average ratio of item features matched with the last three previously bought items
boughtbefore_color	Numerical	Average ratio of matching colors with the last three previously bought items
boughtbefore_brand	Numerical	Average ratio of matching brands with the last three previously bought items
boughtbefore_cat_{1,2,3,4}	Numerical	Average ratio of matching categories with the last three previously bought items (breadcrumb navigation level 1, 2, 3, or 4)
discount *Any discount granted?*	Nominal	Knowledge about a discount: yes/no/unknown

discount_level *Discount level*	Numerical	Level of the discount (-1:unknown, 0:none, 1:low, 2:medium, 3:high)
viewed_before *Viewed before?*	Bool	Has the recommended item been viewed before
viewed_before_count *Number of previous views*	Numerical	Counter of previous item views
viewed_before_days_min *Distance to last view (in days)*	Numerical	Shortest distance to a previous item view event in days
viewed_before_days_max *Distance to first view (in days)*	Numerical	Longest distance to a previous item view event in days
viewed_before_sessions_min *Distance to last view (in sessions)*	Numerical	Shortest distance to a previous item view event in sessions
viewed_before_sessions_max *Distance to first view (in sessions)*	Numerical	Longest distance to a previous item view event in sessions
rec_before	Bool	Has the recommended item been recommended before
rec _before_count	Numerical	Counter of previous item recommendations
rec_before_days_min	Numerical	Shortest distance to a previous item recommendation event in days
rec _before_days_max	Numerical	Longest distance to a previous item recommendation event in days
rec _before_sessions_min	Numerical	Shortest distance to a previous item recommendation event in sessions
rec _before_sessions_max	Numerical	Longest distance to a previous item recommendation event in sessions
avg_colors_session{1,2,3}	Numerical	Average number of colors in the last 1, 2, or 3 sessions
avg_brands_session{1,2,3}	Numerical	Average number of brands in the last 1, 2, or 3 sessions
avg_price_session{1,2,3}	Numerical	Average number of price levels in the last 1, 2, or 3 sessions
avg_cat{1,2,3,4}_session{1,2,3}	Numerical	Average number of categories (breadcrumb navigation level 1, 2, 3, or 4) in the last 1, 2, or 3 sessions
user_avg_colors_session	Numerical	Session-wise average of different colors over all past user sessions
user_avg_brands_session	Numerical	Session-wise average of different brands over all past user sessions
user_avg_price_session	Numerical	Session-wise average of different price levels over all past user sessions
user_avg_cat{1,2,3,4}_session	Numerical	Session-wise average of different categories (breadcrumb navigation level 1, 2, 3, or 4) over all past user sessions
user_price	Numerical	Average price level of the user regarding all past actions
user_pricelevel_{view,buy}	Numerical	Average price level of the user regarding all past view or buy actions
user_discount	Numerical	Average discount level of the user regarding all past actions
user_pricereduction_{view,buy}	Numerical	Average discount level of the user regarding all past view or buy actions
user_viewedbefore_click	Numerical	Ratio of the user clicking on already known recommendations
user_viewedbefore_click_count	Numerical	Average number of previous item views before clicking on a recommendation
user_viewedbefore_success	Numerical	Ratio of the user clicking already known recommended items and buying them later on in the same session
user_viewedbefore_success_count	Numerical	Average number of previous item views before a sucessful recommendation (click and buy in the same session)

Table 16: List of features not considered by DeepPredict.

Feature name
brandpop
brandpop_day
brandpop_week
catpop
catpop_month
catpop_week
catpop_day

D Additional Experimental Results

Table 17: Characteristics of the additional Zalando datasets.

	Raw dataset	3k *Random* users	5k *Regular* users	10k *Regular* users
Users	3.5M	3,000	5,000	10,000
Items	460k	76k	150k	185k
Views	200M	358k	1.6M	3.2M
Purchases	3.9M	12k	67k	134k
Sessions	27.5M	41k	146k	293k
Sessions per user	7.79	13.77	29.22	29.29
Views per session	7.28	8.74	11.15	11.03
Purchases per session	0.14	0.29	0.45	0.46

Table 18: *Hit Rate@10* and *MRR@10* results for the additional subsets of *random* and *regular* Zalando users. Statistically significant differences (according to a Student's t-test with $p < 0.05$) between DeepPredict and the second-best method in the experiments are marked with a star.

Baseline	C-KNN					
Dataset	Zalando *random* 3k		Zalando *regular* 5k		Zalando *regular* 10k	
Metric@10	HR	MRR	HR	MRR	HR	MRR
No post-processing	0.341	0.187	0.392	0.178	0.430	0.189
FM	0.381	0.207	0.415	0.183	0.450	0.194
IRec-FM	0.458	0.299	0.484	0.198	0.499	0.204
DR-FM	0.403	0.258	0.461	0.196	0.476	0.222
RPOP-FM	0.458	0.270	0.491	0.228	0.497	0.217
RFPredict	0.458	0.294	0.497	0.281	0.490	0.271
WR(RPOP,DR,0.5)-FM	0.467	0.309	0.513	0.262	0.517	0.250
DeepPredict	**0.480***	**0.355***	**0.523**	**0.294***	**0.547***	**0.316***