Will This Online Shopping Session Succeed? Predicting Customer's Purchase Intention Using Embeddings

Miguel Alves Gomes
Chair for Technologies and Management of Digital Transformation, University of Wuppertal
Wuppertal, North Rhine-Westphalia, Germany
alvesgomes@uni-wuppertal.de

Richard Meyes
Chair for Technologies and Management of Digital Transformation, University of Wuppertal
Wuppertal, North Rhine-Westphalia, Germany
meyes@uni-wuppertal.de

Philipp Meisen
Breinify Inc.
San Francisco, California, USA
philipp.meisen@breinify.com

Tobias Meisen
Chair for Technologies and Management of Digital Transformation, University of Wuppertal
Wuppertal, North Rhine-Westphalia, Germany
meisen@uni-wuppertal.de

ABSTRACT

Customers are increasingly using online channels to buy products. For e-commerce companies, this offers new opportunities to tailor the shopping experience to customers' needs. Therefore, it is of great importance for a company to know their customers' intentions while browsing their webpage. A major challenge is the real-time analysis of a customer's intention during browsing sessions. To this end, a representation of the customer's browsing behavior must be retrieved from their live interactions on the webpage. Typically, characteristic behavioral features are extracted manually based on the knowledge of marketing experts. In this paper, we propose a customer embedding representation that is based on the customer's click-events recorded during browsing sessions. Thus, our approach does not use manually extracted features and is not based on marketing expert domain knowledge, which makes it transferable to different webpages and different online markets. We demonstrate our approach using three different e-commerce datasets to successfully predict whether a customer is going to purchase a specific product. For the prediction, we utilize the customer embedding representations as input for different machine learning models. We compare our approach with existing state-of-the-art approaches for real-time purchase prediction and show that our proposed customer representation with an LSTM predictor outperforms the state-of-the-art approach on all three datasets. Additionally, the creation process of our customers' representation is on average 235 times faster than the creation process of the baseline.

CCS CONCEPTS

• **Information systems → Data mining**; **Computational advertising**; **Expert systems**; **Data analytics**; **Personalization**; Task models; Online advertising; • **Applied computing → Online shopping**.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs International 4.0 License.

CIKM '22, October 17–21, 2022, Atlanta, GA, USA
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9236-5/22/10.
https://doi.org/10.1145/3511808.3557127

KEYWORDS

e-commerce, purchase prediction, customer representation, embedding, skip-gram, machine learning

ACM Reference Format:
Miguel Alves Gomes, Richard Meyes, Philipp Meisen, and Tobias Meisen. 2022. Will This Online Shopping Session Succeed? Predicting Customer's Purchase Intention Using Embeddings. In *Proceedings of the 31st ACM Int'l Conference on Information and Knowledge Management (CIKM '22), Oct. 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3511808.3557127

1 INTRODUCTION

Nowadays, many products and services are offered online making the online market the preferred choice purchase for many customers. In 2018, the online sales share exceeded in-store sales in the U.S. for the first time [20]. Furthermore, the COVID pandemic strengthened this customer behavior and leads to further growth in e-commerce sales [22]. In contrast to brick-and-mortar retail, e-commerce offers far-reaching opportunities to tailor the shopping experience to customer needs [12]. A basic prerequisite for this customization is that customer behavior must be observed and analyzed in a sensible manner to derive meaningful recommendations for actions suggested to the customer during their shopping experience [7, 8]. In this regard, seemingly simple classifications, such as whether or not a customer will purchase a product or service offered, pose non-trivial challenges [14, 15]. Nevertheless, such information is central for companies when planning resources and inventories, among other things [11, 21, 35]. Typically, the number of customers with mere browsing intentions is far greater than customers with purchase intentions and it has been shown that it is more effective to target customers with a purchase intention [3, 24]. Additionally, Esmeli et al. [6] state that it is more useful to target customers with purchase intention in real-time to not miss the chance to target them with appropriated marketing strategies. However, knowing that browsing customers do not intend to buy anything allows real-time counteracting to engage with the customer and for example, encourage purchases.

In our work, we address the problem of predicting the customer's intention in real-time. More specifically, we investigate how sequential customer interactions on an e-commerce platform can be used to predict the probability of a purchase. To this end, our proposed solution consists of several components. In order to characterize a customer, we first need to represent a customer's behavior digitally. In the literature, this digital representation consists of customer features that are derived manually after an extensive data analysis process by experts. Another approach, which we also pursue in the following, is the automatic derivation of features. Thereby, our approach is based on learning embeddings as representations. More precisely, we use an embedding to encode customers' interactions acquired from online sessions. We utilize the resulting embedding representations to train different machine learning (ML) models to predict if an ongoing session is going to lead to a purchase. We evaluate our approach on three different datasets and show that our embedding approach with a long short-term memory (LSTM) model outperforms existing state-of-the-art approaches on all three datasets. In order to engage the customer in the ongoing session, it is essential to predict future behavior in real-time. In this context, real-time means that the customer gets the feeling that the content is delivered immediately. As several authors state, this means that a website responds within 0.1 seconds [4, 17]. We show that our approach is capable to extract the embedded customer representation and make a purchase prediction within this timeframe. Furthermore, our embedding approach creates the customer representation on average 235 times faster compared to the actual state-of-the-art approach.

The remainder of this paper is structured as follows: In the next section, we present related work for customer representation and purchase prediction. In section 3, we introduce the use case and describe the datasets. In section 4, we provide theoretical details of our approach. Section 5 describes all necessary steps of our experiments to solve the stated problem. Thereafter, in section 6, we present the results and discuss the outcomes. Finally, we summarize our work and discuss its limits and future research opportunities.

2 RELATED WORK

2.1 Customer Representation

For making a purchase prediction a customer representation is necessary. Hence, features are needed that represent the customer. A large body of work has been devoted to determining features to extract from browser clickstreams and use to represent customer behavior [13, 18, 23, 25–27, 31, 33, 34]. For example, Martinez et al. [15] utilize four different groups of features for a representation: "characteristics related to purchase time", "characteristics related to purchase value", "further customer information", and "additional variables". Esmeli et al. [6] represent the customer with twelve manually selected features which are "Total Viewed Items in a Session", "Total Unique Items", "Total Session Duration", "Click Rate" (How many clicks in a session duration), "Max. Popularity" (Popularity Rate of Items), "Min. Popularity", "Duration Spent on a Product" (in minutes), "Number of Unique Categories", "Hour of Session", "Day of Week", "Weekend" (boolean), "Day of the Year" and utilize these to train four different models. Sheil et al. [26] manually selected six features namely "Timestamp", "Item ID", "Price", "Price Variance", "Item Category ID", and "Item quantity" and insert them into an embedding layer which is trained end-to-end for purchase prediction.

Lin et al. [14] represent the customers with a different approach. It is based on the "Five-Stage Sequential Consumer Purchase Decision Model" (PDM) [10]. Based on the actual customer stage in the PDM, they assign an encoded value to the customer sequence. The issue is that their approach requires a lot of knowledge about the customer and dataset. For one dataset, they utilize an encoding with either eight or 14 different values but for the other dataset, they only can apply three values due to the lack of knowledge. Another customer representation approach is proposed by Bauman et al. [2]. They utilize a graph representation which is built based on the customer's clickstream data from the webpages. These different customer representation approaches require domain knowledge over the process to create an encoding for each customer and the customers need to be known to aggregate the information for manually selected features. In order to handle these problems and to have a more generalizable customer representation that is applicable to more use cases without much adjustment, we propose an embedding approach, which requires far less domain knowledge and works on unknown users. Examples of successful applications of embeddings in e-commerce are recommender systems as shown by Vasile et al. [32], Alves Gomes et al. [1], Tercan et al. [30], and Srilakshmi et al. [28].

2.2 Purchase Prediction

Different machine learning models are used to predict if a customer will purchase a product or not. For example, Park et al. [23] and Romov et al. [25] use a gradient boosting (GB) method or Li et al. [13] utilize a linear regression model for purchase prediction. Martinez et al. [15] apply different models like logistic lasso regression, extreme learning machine (single-hidden layer neural network), and GB to make the classification if a customer will purchase or not in a given month. In their studies, GB shows the best performance. Thereby, in all the mentioned works, the purchase prediction is made on historical records, i.e., after the actual customer interaction.

As Esmeli et al. [6] already mentioned in their publication, a major issue they identified is that most publications make purchase predictions on past events and do not try to predict the customers' purchase intention while they are browsing. However, in our literature research, we have found two publications that tackle this problem. Lin et al. [14] and Esmeli et al. [6] use their models to make a just-in-time purchase prediction. Lin et al. use a logistic regression (LR) model and a long-short term memory (LSTM) model for the prediction. Their experiments show that the LSTM model performs slightly better than the LR model. Esmeli et al. apply decision tree (DT), random forest (RF), Bagging, K-nearest-neighbour (KNN), and Naive Bayes (NB) to the customer interaction data for the purchase prediction. They show that DT has the best performance.

Table 1 summarizes the related publications for purchase prediction and gives an overview of the used customer representation approaches, predicting models, used datasets, and if approaches can be used in real-time. We noticed that most authors select features manually for the customer representation. Further, the most used dataset is the yoochoose dataset which is also referred as RecSys2015 dataset.

Table 1: Overview of publications that predict a future purchase with information about the utilized customer representation, predicting models, data sets, and whether it can be used in real-time.

Author	Year	Customer Representation	Model	Dataset	Real-Time
Li et al. [13]	2015	Manual Feature Selection	Gradient Boosting with Logistic Regression	Alibaba (closed)	No
Park et al. [23]	2015	Manual Feature Selection	Gradient Boosting	yoochoose	No
Romov et al. [25]	2015	Manual Feature Selection	Gradient Boosting	yoochoose	No
Wu et al. [33]	2015	Manual Feature Selection	Gradient Boosting; Multilayer Perceptron; Bi-LSTM	yoochoose	No
Bauman et al. [2]	2018	Graph	Logistic Regression; Random Forest; Gradient Boosting	Online Retailer 1 (closed); Online Retailer 2 (closed)	No
Martinez et al. [15]	2018	Manual Feature Selection	Lasso Regression; Gradient Boosting; Extreme Learning Machine	B2B data (closed)	No
Sheil et al. [26]	2018	Manual Feature Selection and Embedding	Gradient Boosting; LSTM	yoochoose; RetailRocket	No
Lin et al. [14]	2019	Encoding	Logistic Regression; LSTM	yoochoose; JD.com 1 (closed)	Yes
Mokryn et al. [19]	2019	Manual Feature Selection	Bagging; NBTree; Logistic Regression; Gradient Boosting	yoochoose; Zalando	No
Zeng et al. [36]	2019	Manual Feature Selection	Logistic Regression	JD.com 2 (closed)	No
Esmeli et al. [6]	2020	Manual Feature Selection	Naive Bayes; Decision Tree; Random Forest; Bagging; K-Nearest-Neighbor	yoochoose	Yes

3 USE CASE AND DATA DESCRIPTION

The task we address is to predict in real-time the likelihood of a browsing customer making a purchase. To this end, our approach needs to meet four requirements.

(1) The purchase prediction is at least as accurate as published prediction methods.
(2) The purchase prediction must take less than 0.1 seconds for a good as possible user experience while the customer is browsing.
(3) The approach is transferable to different use cases (data formats and kind of data) with as little manual effort as possible from a domain expert.
(4) The approach works for known and unknown customers.

In order to meet the aforementioned requirements, we define the minimum necessary information that the gathered data must contain. As a rule, each customer interaction is recorded separately for the events listed below. Each event is characterized by three values, which are a session identifier, a timestamp, and a touchpoint. Examples of touchpoints are URLs visited and products viewed. Usually, more information like activity types are tracked and we can combine multiple pieces of information to define the touchpoints. We show such a combination in section 5 when we explain the exact preprocessing of the data used. For reasons of transferability of the approach, touchpoints are intentionally kept abstract and have to be defined manually specific to the use case at hand. A sequence of events is called sessions and the session identifier is necessary to assign the events to a session. The timestamp is necessary to sort the events within a session chronologically and as aforementioned, the touchpoint describes the object the customer interacts with.

To demonstrate transferability between use cases, we evaluate our approach on three real-world datasets. We use two publicly available datasets (yoochoose and openCDP) as well as a proprietary (closed) dataset provided by a large US online retail group. Each dataset contains our defined necessary information above and also comes with more information. In our considered datasets, the records have either customers' interactions with products or the webpage URL.

The yoochoose dataset[1] was introduced for the RecSys Challenge 2015[2] and amongst others used by Lin et al. and Esmeli et al. to predict customer purchases. Due to its wide distribution, we use the yoochoose dataset as the baseline dataset for purchase prediction to compare our approach. The dataset provides information about click events and buy events of a European online retailer. Each event contains information about the session identifier, the event time, the product, and the product category. Additionally, the dataset provides information about in which session a product was purchased, the product price and the quantity. It contains records of around 9.2 million session with 33 million customer interactions in which 5.5% are sessions with purchases. The records are distributed over six month (April 2014 to September 2014).

[1]Download dataset at https://www.kaggle.com/datasets/chadgostopp/recsys-challenge-2015
[2]https://recsys.acm.org/recsys15/challenge/

The openCDP dataset[3] was used in 2020 RecSys tutorial[4] and is provided by the REES46 Marketing Platform[5]. The dataset contains customer behavior data for seven months (October 2019 to April 2020) from a large multi-category online store. Each entry in the dataset constitutes a customer event on the online platform and contains nine different values of information. These values are a session identifier, a user identifier, an event time, an event type, the event touchpoint that is a product in this use case, a product category id, the product brand, the product price, and a product category brand. Event types are either "product view", "add to cart", and "purchase". The dataset contains 386,299 different products and over 411 million events from around 89 million sessions of which 6.1% are purchase sessions.

The closed dataset contains browsing information of a consumer package good online retailer in the US. Each entry is an event that has information about the event time, event type, contains a session identifier, a user identifier (if known), and the touchpoint with additional information. In this dataset, the customer touchpoint is the URL the customer is visiting. The data was recorded over five months from January 2020 to May 2020. It consists of 53 million customer events with 66,883 different URLs. Each event can be of type "page visit", "product view", "add to cart", "remove from cart", or "purchase". The events were made in 6.2 million sessions of which 1.6% lead to a purchase.

4 METHODOLOGY

Typically, predicting whether a customer will buy something based on their behavior is usually done in several steps. First, a domain expert will extract the necessary information from the customer data to create a customer representation that allows comparing customers with each other. Next, the data is split into a training and test set, subsequently, a learning model is trained on the customer representation training data to predict the likelihood of a product purchase. In the last step, the performance of the model is evaluated.

Our approach differs mainly in how the customer representation is created, allowing us to pursue additional goals besides mere purchase prediction. We aim to generate a customer representation that is easily transferable to other use cases and requires as little domain expertise as possible. Additionally, we aim to only utilize real-time customer information. This means that we refrain from using information that only becomes available at the end of a customer session to construct our customer representation and we do not rely on historical customer information because this would not allow addressing new or unknown customers without a history.

Our underlying assumption is that the customer's intention is reflected in their behavior when browsing the webpage which is represented by the ongoing online session. Furthermore, we assume that customers with similar sessions have similar intentions. Following these assumptions, we create a customer representation based on the touchpoints of an ongoing session.

Our work is inspired by word embeddings by Mikolov et al. [16]. As previously mentioned, we utilize embeddings to represent the customer under consideration of the aforementioned constraints. In Natural Language Processing (NLP) words in a sentence or document are encoded in vectors. Similarly, we encode the customer's events recorded during online sessions. Figure 1 illustrates our proposed model inspired by the skip-gram model. Our adaption to the classic skip-gram model can be described as follows: Given are two sets S and T in which T is a set of all touchpoints $t \in T$ and S is a set of all customer sessions $s \in S$. Here, s is an ascended time-ordered sequence of touchpoints t with sequence length $n, n \in \mathbb{N}, n > 1$. The objective is to train a D-dimensional embedding representation $e_t \in \mathbb{R}^D$ in which similar touchpoints are closer to each other in the given embedding space. Thereby, we define the similarity between two touchpoints t_i and t_k as follows: t_i and t_k are similar to each other if they share the same context $m, m \in \mathbb{N}, m \leq n$. The context is given by t_i's previous touchpoints $\{t_{i-1}, t_{i-2}, ..., t_{i-a}\}$ and subsequent touchpoints $\{t_{i+1}, t_{i+2}, ..., t_{i+b}\}$ with context size $a+b+1 = m$. In order to train an embedding that fulfills that definition of similarity, the learning task for the embedding is to predict the context c_i of the touchpoint $t_i \in s_i$ with the goal to maximize the likelihood

$$L(\theta) = \prod_{i=1}^{n} \prod_{-m \leq j \leq +m; j \neq 0} P(t_{i+j}|t_i; \theta) \quad (1)$$

with θ being the optimizable variables (weigths) of the embedding.

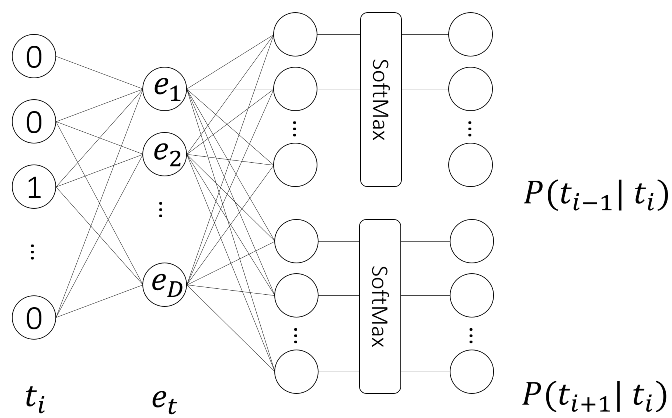


Figure 1: Adapted skip-gram model with context size $m = 2$. The first layer is the one-hot encoded input layer for a touchpoint (t_i). The second layer is the embedding layer (e_t), and the last layer a softmax activated output layer. The goal of the neural network is to predict the conditional probabilities $p(t_{i-1}|t_i)$ and $p(t_{i+1}|t_i)$ in which t_{i-1} is the previous and t_{i+1} subsequent customer touchpoint of t_i

5 EXPERIMENTS

Our approach consists of four steps. (1) Data preprocessing, (2) embedding training, (3) prediction model training, and (4) prediction model evaluation. In the second and third step, we used hyperparameter tuning. The experiments were implemented in Python 3.8.10 with NumPy and Pandas for the data preprocessing, PyTorch for embedding training, scikit-learn and PyTorch for purchase predicting model implementation and evaluation, Matplotlib and Seaborn

[3]Download dataset at https://www.kaggle.com/datasets/mkechinov/ecommerce-behavior-data-from-multi-category-store and https://drive.google.com/drive/folders/1Nan8X33H8xrXS5XhCKZmSpClFTCJsSpE
[4]https://recsys.acm.org/recsys20/tutorials/
[5]https://rees46.com/?locale=en

for visualization, and Optuna for hyperparameter tuning which uses different algorithms like grid search, random, bayesian, and evolutionary. The data preprocessing and model validation was computed on a Windows 10 machine with an Intel i9-10885H and 64GB RAM. Model training and hyperparameter search was computed on a Ubuntu 18.04 machine with 96xIntel Xeon Platinum 8168 CPU @ 2.70GHz, 756GB RAM, and 8xNvidia Tesla V100 GPUs.

5.1 Data Preprocessing

As a first step, we prepared the data for further usage. The yoochoose dataset consists of two files one with product click events and the other with product purchase events. We had to link the event with the help of the session identifier in order to know which session has led to a purchase. Regarding the remaining two datasets, first, we removed all session information about purchases e.g. checkout events, because it is the prediction objective, but kept the information on whether a session contains a purchase or not as a label. In a second step, we merged the event type information with the touchpoint events by concatenating either the event type with the product identifier for the openCDP dataset or the event type with the URL for our closed dataset, which leads to new touchpoints of the form "eventTypeString:touchpointString". For example, a "add to cart" event of the product with the ID "1" would results in "cart:1" and is distinguishable to the "view" event of product "1" which has the form "view:1". In a third step, we removed the query strings from the URLs of the closed dataset, which would over-specify the URLs leading to distinct and unique URLs, which makes generalization of the trained models difficult.

Thereafter, we aggregate the sessions and remove sessions with less than three events. The reason for this is that no useful prediction can be made based on just one or two customer interactions in a real-world setting. For example, usually, the first URL is the landing page and until a checkout event, multiple URLs are visited by the customer. Similarly, for products, a "product view" must first be followed by an "add to cart" and then a "purchase" event. Table 2 shows information about the number of events, number of sessions, average session length, and further information about the sessions containing or not containing purchases of the three datasets after the first preprocessing step.

Table 2: Properties of the three used datasets in our experiments after the first unified preprocessing.

	yoochoose	openCDP	closed
#events	24,628,059	348,906,538	19,740,317
#sessions	4,431,931	40,103,535	2,528,265
#purchase sessions	377,376	5,297,561	99,787
#no-purchase sessions	4,054,555	34,805,974	2,428,478
Øsession length	5.557	8.700	7.808
Øpurchase session length	8.117	9.109	17.859
Øno-purchase session length	5.318	8.638	7.395

For each dataset, we created a training set for the embedding, and a training and test set for the prediction model. We balanced the datasets in terms of their ratios of purchase to no-purchase sessions by undersampling the number of no-purchase sessions with random selection, similar to Esmeli et al. [6]. Afterward, we split the balanced datasets into a training and test set in the ratio 85 : 15. The test set was exclusively used for model validation and never used for any other processing step. Regarding the embedding training set, we created trigrams for all remaining sessions. Specifically, for each touchpoint t_i in all sessions a triplet is created in the form of (t_i, t_{i-1}, t_{i+1}). In order, not to neglect the first and last touchpoint in a sequence when creating the trigrams we have introduced "START" and "END" tokens. Similar approaches are widely used in NLP [5]. Table 3 summarizes the details of the training and test sets for each of the three datasets. Additionally, it shows the number of unique touchpoints that are included in the embedding training set and the number of unknown touchpoints which describes the number of touchpoints that are in the prediction model test set but not in the training sets. Unknown touchpoints can be considered as new touchpoints recently added to the webshop. For example, new products in the portfolio.

Table 3: Metainformation of the train and test set of the three considered datasets.

		yoochoose	openCDP	closed
train set	#sessions	641,539	9,005,853	169,637
	Øsession length	6.72	8.25	12.63
	Øpurchase session length	8.12	7.85	17.92
	Øno-purchase session length	5.32	8.65	7.34
test set	#sessions	113,213	1.589.269	29,937
	Øsession length	6.73	8.23	12.46
	Øpurchase session length	8.13	7.84	17.50
	Øno-purchase session length	5.34	8.62	7.47
embedding train set	#trigrams	10,247,216	129,318,153	3,952,501
	#touchpoints	48,012	582,082	72,759
	#unknown touchpoints	182	509	96

5.2 Embedding Training for Customer Representation

We implemented the embedding as described in section 4 and trained an embedding model for each of the three datasets. For training, we used the Cross-Entropy Loss and Adam as an optimizer algorithm [9]. Hyperparameters like the embedding dimension, batch size, and learning rate are selected by a hyperparameter search algorithm aiming to minimize the training loss. For the search, we used 100 trials that trained at most 50 epochs on previous randomly selected 50% of the embedding training set. Table 4 summarizes the empirically determined hyperparameters.

Table 4: Used hyperparameters to train the embeddings of each dataset.

Hyperparameter	yoochoose	openCDP	closed
Embedding dimension	32	64	32
Batch size	1024	1024	256
Learning rate	9.854×10^{-4}	1.921×10^{-3}	5.033×10^{-3}

Thereafter, we trained each embedding with the calculated hyperparameter until convergence which means, that within 10 epochs the loss was not improved by at least 0.001. As aforementioned, in a real-world setting unknown touchpoints can occur. In language models, this is known as out-of-vocabulary problem. A solution to this problem is to introduce an "UNKNOWN" token [29]. Based on this, we introduced a single new "UNKNOWN" touchpoint to handle all unknown touchpoints and increased the embedding input layer by one. To have training examples, we decided to replace the fraction between unknown and known touchpoints of the input touchpoints with the "UNKNOWN" touchpoint at each epoch.

5.3 Baseline Representation

In order to compare our embedding approach against the current state-of-the-art we chose an alternative approach as a baseline from the literature, which must fulfill three criteria: (1) "easy to adapt", (2) "top state-of-the-art performance" on the yoochoose dataset, and (3) real-time capability. "Easy to adapt" means the implementation does not require additional expertise about the dataset or feature selection process. In order to compare against state-of-the-art performances, we picked five related publications that utilize the yoochoose dataset and thus, could potentially serve as a baseline approach. Table 5 shows the archived performance for purchase intention prediction of the five publications. Four of the five authors used the area-under-curve (AUC) score which makes it comparable. Given the three requirements, our final decision for the baseline has fallen on the approach of Esmeli et al. [6] as (1) their customer representation is almost completely transferable to each of the three use cases, thus easy to adapt, (2) they have reported the highest AUC score on the yoochoose dataset, and (3) they claim the capability to perform real-time predictions.

Table 5: Overview of the models' performance on the yoochoose dataset for publications that use either F1 or AUC score.

Author	Model	Score
Romov et al. [25]	Gradient Boosting[i]	AUC: 0.85
Sheil et al. [26]	LSTM	AUC: 0.839
Lin et al. [14]	Logistic Regression	AUC: 0.72
Mokryn et al. [19]	Bagging	F1: 0.786
Esmeli et al. [6]	Decision Tree	AUC: 0.97

[i] Winner of the RecSys 2015 Challenge

For the baseline customer representation, we need to extracted twelve features from the customer interaction data of which eight are session and touchpoint-based and four are time-based. In the following the twelve features are briefly described: **total viewed items** describes how many touchpoints a customer had so far in the session. **number of unique items** describes the number of unique touchpoints in a session. **total session duration** describes the session's duration in minutes. **click rate** is defined by the ratio of **total viewed items** and **total session duration**. **max popularity** and **min popularity** are represented by the number of a touchpoint that is either interacted with the most or the least. **Duration spent on a product** is the average time in minutes on a touchpoint. **unique categories** is the number of unique touchpoint categories in a session. Although, the product category is not a requirement on the datasets, both open datasets provide category information of the products. In contrast, for the closed dataset, we used the event type as the category for the URL. **Hour of the day**, **day of the week**, **weekend** (0 or 1), and **day of the year** of the session start [6].

5.4 Purchase Prediction Models

We perform purchase predictions by a learning model trained on a binary classification task, i.e., to estimate if a given session will lead to a purchase. Based on the related literature, we tested six different ML models, i.e., **DT**, **RF**, **GB**, **LR**, **Multilayer Perceptron (MLP)**, and **LSTM**. The input of all models is the customer representation, more specifically in the case of our embedding approach, it is the embedded touchpoint sequence of size $D \times n$, with D being the dimension of the touchpoint embedding and n being the session length. All learning models with exception of the LSTM network require a fixed input size. Therefore, we trained the models for different session sizes m in a range of 3 to 20 to account for the majority of different session lengths. For sessions longer than m we cropped the end, while sessions shorter than m were padded with zeros. In the case of the baseline approach, it is not necessary to use different session lengths for training because the input size is constant. However, for the baseline approach, we did not utilize an LSTM predictor because it requires a sequential input that is not given. In general, we used the twelve features to train the models. Additionally, we determined the effect of the time features by training the models on the first eight features omitting the last four time features.

Similar to the embedding training, we performed 10-fold cross-validation and hyperparameter tuning to obtain the models' hyperparameters. The hyperparameter search aimed to maximize the F1 score and stopped after two hours or after 100 trials. Furthermore, in the hyperparameter search the session length m was randomly selected from the normal distribution with the mean and standard deviation of the session length for the associated dataset to reduce the computation amount. Table 6 summarizes the parameters determined by the tuning process for each model, approach, and dataset.

5.5 Experiment Evaluation

To evaluate our trained models and the overall approach, we use three evaluation metrics, two of which describe the performance of the purchase prediction model while the remaining one evaluates the real-time capability of the models. Additionally, we measure the time required for our embedding approach to compute the customer representation and compare it against the baseline approach. In order to achieve comparability, we measure the performance in the following with the AUC score, since most approaches from the literature use this same score. Additionally, we evaluate the models based on their F1 scores to compare them.

In a real-world scenario, it is not unusual to receive several thousands of requests per second at peak activity times. To evaluate our approach regarding its real-time capabilities, we mocked a simple stress test representing such a real-world scenario to measure (1) the prediction time of the learning models and (2) the computation

Table 6: Tuned hyperparameter of the predicting models for the three different dataset and our embedding and the baseline customer representation.

		yoochoose		openCDP		closed	
Model	Parameter	Embedding	Baseline	Embedding	Baseline	Embedding	Baseline
DT	Criterion	gini	entropy	entropy	gini	entropy	entropy
	Splitter	best	best	best	best	best	best
	Min samples split	2	9	7	20	13	26
	Max features	sqrt	auto	sqrt	None	None	sqrt
	Max depth	9	5	15	7	7	6
RF	N estimators	70	50	100	70	120	140
	Criterion	entropy	gini	entropy	ginig	entropy	gini
	Min samples split	15	32	2	11	6	6
	Max features	None	None	None	None	sqrt	None
	Max depth	16	14	16	17	20	12
GB	Learning rate	$1.063e^{-3}$	$1.488e^{-3}$	$5.043e^{-2}$	$2.988e^{-3}$	$3.767e^{-3}$	$1.724e^{-2}$
	N estimators	70	10	40	160	60	80
	Min samples split	6	2	4	2	6	10
	Max features	sqrt	sqrt	sqrt	None	sqrt	auto
	Max depth	14	8	10	14	8	7
LR	Solver	lbfgs	newton-cg	lbfgs	newton-cg	lbfgs	newton-cg
	Max iter	574	950	264	67	297	678
MLP	Num layers	2	3	1	3	2	1
	Layer sizes	1024;64	16;32;16	512	256;64;64	16,32	256
	Activations	tanh;sig	-;-;elu	sig	elu;elu;relu	elu,sig	tanh
	Batch size	128	256	512	512	512	128
	Learning rate	$6.911e^{-4}$	$1.951e^{-3}$	$4.14726e^{-5}$	$5.766e^{-5}$	$1.973e^{-3}$	$8.758e^{-5}$
LSTM	Num layers	2	-	1	-	1	-
	Hidden size	512	-	64	-	32	-
	Batch size	1024	-	512	-	256	-
	Learning rate	$2.834e^{-5}$	-	$2.753e^{-3}$	-	$8.044e^{-4}$	-

time to create the customer representation. Therefore, the following process was used. First, for n in 1 to 10^6 in power of ten steps, we randomly select n sessions from one of the three datasets. Then, our embedding and the baseline representation are created for the previously selected sessions and the time is measured for each. Thereafter, the prediction time for the six ML models is measured. This process is repeated 100 times.

6 RESULTS AND DISCUSSION

6.1 Prediction Evaluation

Table 7 shows the results of our experiments. We measured the models' performance of our embedding and the baseline approach with and without time features. Regarding the yoochoose dataset, the baseline approach performs better on five of the six learning models. In comparison, the embeddings are only convincing for the LSTM model. However, this model delivers the best overall results for both the F1 score and the AUC. When training other models with embeddings, however, slightly worse results are observed. In the case of the openCDP dataset, we observe a similar picture in which the baseline has a better performance on the same models but again our approach results in the best overall performance for F1 and AUC score with LSTMs. Generally, our embedded customer representation performs better with the LSTM predictor on the two publicly available datasets. However, regarding the closed dataset, our embedding approach outperforms the baseline approach independent of which ML model is used. Comparing the results of our embedding approach with an LSTM against the baseline approach with the best model, our approach has a 0.002 higher F1 score on the yoochoose data, a 0.017 higher F1 score on the openCDP data, and a 0.098 higher F score on our closed dataset.

The first thing we notice is that the baseline approach on the yoochoose dataset performs worse than reported in the publication. We presume two reasons. The first is that we removed all sessions shorter than three events. This eliminates many sessions that are easier to classify since most of them are not buying sessions. In numbers, 4,817,798 sessions were removed of which 4,685,478 (97%) are no-purchase sessions. The second reason for the difference is the way Esmeli et al. evaluate their approach. They propose an evaluation approach to evaluate the early purchase intention and then calculate the AUC score based on that. Considering the results of Lin et al. (table 5), we see that they reach an AUC score of 0.72 on the yoochoose dataset, which is comparable to the AUC score we get with the baseline approach.

Another observation is that regarding the baseline approach, the time features have a positive effect on the prediction performance. For each model and dataset, the F1 score is higher for customer representations with timing features. The same holds true for the AUC score, except once when the AUC score is 0.001 higher for the baseline model without time features with an LR predictor on the opdenCDP data.

One observation we made about the results is that our embedding approach with LSTMs performs particularly well and shows the best performance for the three datasets. The reason is, on the one hand, the ability of LSTMs to process sessions of different lengths and on the other hand, the necessary uniform input length for the remaining five learning models, which is disadvantageous for our embedding approach. For these five models, it is necessary to either lengthen sessions that are too short or shorten sessions that are too long, which both result in an information loss. In contrast, the baseline approach does not have this problem, since the length is always fixed.

Regarding the results of the four non-neural net learning models on the openCDP data. The results are lower in comparison to the MLP and LSTM models. The reason is that the tuning of the hyperparameters could not identify the best parameters because the computation time was too long due to the number of features and the size of the training dataset. Tree-based ML models are not designed to be trained partially like neural nets.

Another reason for the different results regarding approaches and the datasets is the information contained in the data. As our results show, the prediction accuracy on the yoochoose dataset is worse than the performance on the other two datasets. Based on this, we deduce that the information in the yoochoose dataset for individual purchase sessions and no-purchase sessions does not differ as much as in the other two. The yoochoose dataset contains only product view records without any other information, like as whether the product has been added to the shopping cart which can indicate a buying interest of the customer. Furthermore, a purchase event can not happen without a previous "add to cart" event which increases the odds of a purchase that can be learned from the ML model. Regarding the model scores on the openCDP dataset, this is either represented by successive product interactions that are beneficial for the baseline approach or event types that are encoded in the trained embeddings. Regarding the former, the embedding representation would also benefit if we did not encode the event type into the touchpoints because the ML model could learn that two successive equal touchpoints represent an "add to cart" event. To verify this thought we analyzed the false positive predicted sessions of the test set. The analysis shows that out of the 109,407 false positives 93,468 (85.4%) are sessions with "add to cart" events.

Table 7: AUC and F1 performance of each used ML model of our embedding approach against the baseline for each dataset. The best performing approach for each model is indicated by bolt values and the best approach and model, in general, is additionally underlined per dataset.

		DT		RF		GB		LR		MLP		LSTM	
datasets	Approache	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC	F1	AUC
yoochoose	Embedding	0.636	0.676	0.670	0.747	0.662	0.725	0.637	0.712	0.672	0.723	**0.697**	**0.772**
	Baseline	**0.680**	**0.723**	**0.694**	**0.757**	**0.695**	**0.746**	**0.641**	**0.725**	**0.695**	**0.746**	-	-
	Baseline^{i}	0.656	0.716	0.684	0.731	0.684	0.731	0.638	0.723	0.683	0.732	-	-
openCDP	Embedding	0.757	0.806	0.765	0.817	0.763	0.814	0.755	0.809	0.841	**0.912**	**0.872**	**0.931**
	Baseline	**0.843**	**0.901**	**0.855**	**0.916**	**0.852**	**0.914**	**0.811**	0.884	**0.851**	0.911	-	-
	Baseline^{i}	**0.843**	0.900	0.845	0.906	0.846	0.906	0.805	**0.885**	0.842	0.902	-	-
closed	Embedding	**0.851**	**0.895**	**0.882**	**0.929**	**0.865**	**0.916**	**0.830**	**0.891**	**0.870**	**0.915**	**0.890**	**0.940**
	Baseline	0.779	0.834	0.792	0.852	0.791	0.850	0.766	0.832	0.786	0.834	-	-
	Baseline^{i}	0.774	0.831	0.776	0.834	0.779	0.836	0.756	0.826	0.778	0.828	-	-

^{i} Baseline customer representation without time features.

In comparison, from the 75,556 false negatives, only 37,676 (49.9%) sessions contain "add to cart" events. This analysis supports our thought.

This also applies to our closed dataset. The touchpoints of the closed dataset contain even more information because the URLs cannot be selected arbitrarily, but are bound to the structure of the website. This is particularly advantageous for our embedding approach since the similarities are given by a structure that usually does not change. As words in a sentence are subject to certain rules, the URLs are also subject to a certain structure that is encoded in the embedding. Thus, similarities between touchpoints are more important since they lead to the same target (see similarity definition in section 4).

6.2 Real-Time Evaluation

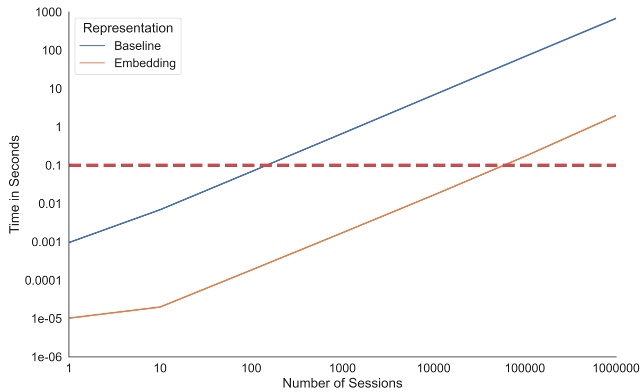
As we pointed out in our requirements, in addition to high prediction accuracy, speed in computing and accessibility of the prediction for a real scenario is of utmost importance. Figure 2 shows the required average time in seconds to compute the customer representation in plot 2a and to predict a purchase in plot 2b for 1 to 10^6 sessions. Regarding the customer representation creation time for our embedding and the baseline approach, we see in plot 2a that the computation time required to create our embedding representation is two orders of magnitude faster than the baseline approach. In numbers, extracting the features from 1,000 sessions takes on average 0.675 seconds. For compression, our approach takes on average 0.0017 seconds for the same amount of sessions. Starting with 100,000 sessions our approach needs a little longer than 0.1 seconds, specifically 0.1703 seconds on average. Break down the results into one session, our approach takes on average 3.058×10^{-6} seconds and the baseline approach 7.215×10^{-4} seconds, which makes our approach 235 times faster.

In plot 2b, we see the average inference time for the six ML models used. All six models are capable to make predictions for 10,000 sessions within 0.1 seconds. Only at 100,000 sessions do four models no longer manage to be faster than 0.1 seconds. Both, DT and LR are capable to make a prediction for over 100,000 sessions within 0.1 seconds. None of the six models manages to make a prediction for one million sessions within 0.1 seconds. DT and LR need a little more than 0.2 seconds and the other four models need about 2 seconds for this amount.

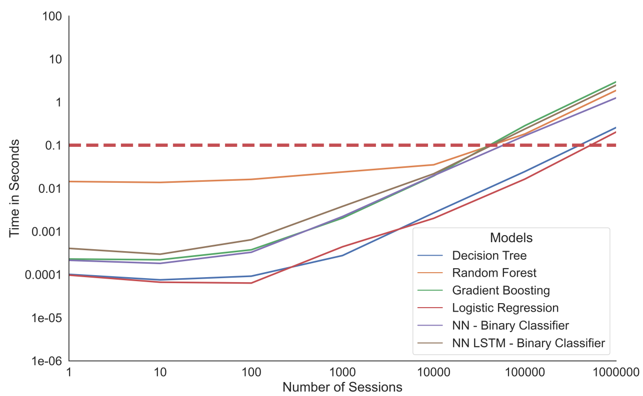
If we add the time needed for the representation calculation with the time needed by the ML model for the prediction, we get the complete time needed for the prediction. For our embedding approach and an LSTM, which has the best performance, a prediction takes 4.089×10^{-4} seconds to create the representation of an ongoing session and to predict the outcome. Even 10,000 sessions at the same time would not be a problem for our approach, which would need 0.038 seconds. Starting with 100,000 sessions our approach would need 0.408 seconds and therefore, not be able to predict customers' purchase intention within 0.1 seconds and by our definition not be real-time capable. Nevertheless, all evaluation experiments were computed on one process. That means that parallelization can enable real-time capability. Thus, we can conclude that our customer representation embedding with a learning model is truly real-time capable and can be used productively. In regard to our experiment, we left out the fact that our representation can be formed incrementally since only the new touchpoints ever need to be embedded. Additionally, we were quite strict about the timeframe for the real-time prediction and therefore, a question that remains unanswered is how adverse it would be if the time on peak activity times would last 1 second.

6.3 Customer Representation Evaluation

Summarizing the results shows that our embedding approach in combination with an LSTM predictor is in many respects superior to the baseline approach for predicting customers' purchase intention in real-time. Furthermore, it meets all set criteria. We set minimum requirements on the data and therefore, the approach requires less domain knowledge about the data and use case. It is only necessary to define the session's touchpoints and train an embedding appropriately. Hyperparameter and model selection are done completely automated and without the need for further action. The presented data analysis served for the purpose of evaluation and is not a required step. This property makes it also easily transferable to different use cases which have discretized touchpoints. Furthermore,



(a) Average computation time to create customer representation with our embedding approach or the baseline approach for 1 to 10^6 incoming sessions.



(b) Average inference time of the six learning models for 1 to 10^6 incoming sessions.

Figure 2: Results from the real-time evaluation. The red dashed line indicates the 0.1 seconds for real-time capability. Note that the x and y-aches are logarithmic scaled

additional information like event types is easily addable which we have done for the opdenCDP and closed dataset. Like the touch-point selection, this also requires minimal manual effort. Since our approach is based solely on session information, it works equally well for known and unknown customers. Hence, two of the four requirements are satisfied. With regard to the other two requirements, our embedding approach is real-time capable, scalable, and fast enough to be used on platforms with thousands of interactions simultaneously. Additionally, it outperforms on all three datasets the state-of-the-art approaches, which additionally, require far more manual tuning to transfer it to new use cases. One possible reason for this is that the embedding encodes information that is difficult to represent in a manually created fixed feature set that generalizes to a much smaller feature space than the embedding.

Besides all the advantages, our embedding approach has also its downside. Skip-gram embeddings are based on neural networks, therefore, interpretability and explainability are an issue. Unlike the embedding approach, manual feature extraction can be used to calculate feature importance and enable appropriate countermeasures based on that. For example, if you identify time as an important feature for not making a purchase, you can automatically start certain marketing campaigns at certain times to get these customers to buy.

7 SUMMARY AND OUTLOOK

For e-commerce companies, it is particularly important to know about their customers' intentions to plan marketing campaigns or engage in real-time during browsing sessions. We proposed an approach combining a customer embedding based on browsing behavior and an LSTM model to predict the likelihood of a customer's purchase in real-time. We empirically evaluated our approach on three different datasets and demonstrated that it outperforms current state-of-the-art approaches, both, in terms of predictive accuracy and computation time. Since the embedding is trained independently of the purchase prediction task, the approach can be transferred to other tasks such as predicting the next action during a browsing session or the time of the next visit.

We have three next steps, that we will approach in the future. (1) we will apply our approach to other e-commerce related tasks, like, which product will the customer purchase next, will the customer come back, will the customer recommend our site, etc, (2) we want to investigate the usability of recently proposed sequential deep learning models like transformer networks and other networks relying on the attention mechanism to perform the various prediction tasks, and (3) we want to transfer our proposed approach into a live environment to validate its applicability in a real-world scenario. Then, we are aiming to use the predictions to create appropriate marketing campaigns in real-time to drive customers with uncertain purchase intentions towards a purchase or to engage a customer toward a company's goal. In this regard, we aim to investigate which marketing strategies have the biggest impact on an individual customer.

In addition, we want to further improve the representation. In the baseline model, we saw that temporal features had a positive effect on the prediction and therefore we want to find a way to encode the time of the customer interaction into the embedding. In other publications e.g. Vasil et al. [32] or Alves Gomes et al. [1], additional information is encoded into the embedding like the product category. We also want to try these approaches and see whether the performance increases. For example, we assume that such an approach would increase the performance on the yoochoose dataset because more information is encoded in the embedding.

REFERENCES

[1] Miguel Alves Gomes, Hasan Tercan, Todd Bodnar, Tobias Meisen, and Philipp Meisen. 2021. A Filter is Better Than None: Improving Deep Learning-Based Product Recommendation Models by Using a User Preference Filter. In *2021 IEEE 23rd Int. Conf. on High Performance Computing and Communications; 7th Int. Conf. on Data Science and Systems; 19th Int. Conf. on Smart City; 7th Int. Conf. on Dependability in Sensor, Cloud and Big Data Systems and Application (HPCC/DSS/SmartCity/DependSys)*. IEEE, 1278–1285. https://doi.org/10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00195

[2] Annika Baumann, Johannes Haupt, Fabian Gebert, and Stefan Lessmann. 2018. Changing perspectives: Using graph metrics to predict purchase probabilities. *Expert Systems with Applications* 94 (2018), 137–148. https://doi.org/10.1016/j.eswa.2017.10.046

[3] Rajat Kumar Behera, Angappa Gunasekaran, Shivam Gupta, Shampy Kamboj, and Pradip Kumar Bala. 2020. Personalized digital marketing recommender engine. *Journal of Retailing and Consumer Services* 53 (2020), 101799. https://doi.org/10.1016/j.jretconser.2019.03.026

[4] Stuart K. Card, George G. Robertson, and Jock D. Mackinlay. 1991. The Information Visualizer, an Information Workspace. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New Orleans, Louisiana, USA) *(CHI '91)*. Association for Computing Machinery, New York, NY, USA, 181–186. https://doi.org/10.1145/108844.108874

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[6] Ramazan Esmeli, Mohamed Bader-El-Den, and Hassana Abdullahi. 2021. Towards early purchase intention prediction in online session based retailing systems. *Electronic Markets* 31, 3 (2021), 697–715.

[7] Taeho Hong and Eunmi Kim. 2012. Segmenting customers in online stores based on factors that affect the customer's intention to purchase. *EXPERT SYSTEMS WITH APPLICATIONS* 39, 2 (2012), 2127–2131. https://doi.org/10.1016/j.eswa.2011.07.114

[8] K. J. Kim and H. Ahn. 2004. Using a clustering genetic algorithm to support customer segmentation for personalized recommender systems. In *ARTIFICIAL INTELLIGENCE AND SIMULATION (Lecture Notes in Computer Science, Vol. 3397)*, T. G. Kim (Ed.). 409–415.

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Philip Kotler, Kevin Keller, Mairead Brady, Malcolm Goodman, and Torben Hansen. 2019. *Marketing Management: 4th European Edition.* Pearson UK.

[11] Venkatesan Kumar, Rajkumar Venkatesan, and Werner Reinartz. 2008. Performance implications of adopting a customer-focused sales campaign. *Journal of Marketing* 72, 5 (2008), 50–68.

[12] Peter S.H. Leeflang, Peter C. Verhoef, Peter Dahlström, and Tjark Freundt. 2014. Challenges and solutions for marketing in a digital era. *European Management Journal* 32, 1 (2014), 1–12. https://doi.org/10.1016/j.emj.2013.12.001

[13] Qiang Li, Maojie Gu, Keren Zhou, and Xiaoming Sun. 2015. Multi-Classes Feature Engineering with Sliding Window for Purchase Prediction in Mobile Commerce. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*. 1048–1054. https://doi.org/10.1109/ICDMW.2015.172

[14] Weiqiang Lin, Natasa Milic-Frayling, Ke Zhou, and Eugene Ch'ng. 2019. Predicting Outcomes of Active Sessions Using Multi-action Motifs. In *IEEE/WIC/ACM International Conference on Web Intelligence*. 9–17. https://doi.org/10.1145/3350546.3352495

[15] Andrés Martínez, Claudia Schmuck, Sergiy Pereverzyev, Clemens Pirker, and Markus Haltmeier. 2020. A machine learning framework for customer purchase prediction in the non-contractual setting. *European Journal of Operational Research* 281, 3 (2020), 588–596. https://doi.org/10.1016/j.ejor.2018.04.034

[16] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[17] Robert B. Miller. 1968. Response Time in Man-Computer Conversational Transactions *(AFIPS '68 (Fall, part I))*. Association for Computing Machinery, New York, NY, USA, 267–277. https://doi.org/10.1145/1476589.1476628

[18] Wendy W Moe. 2003. Buying, searching, or browsing: Differentiating between online shoppers using in-store navigational clickstream. *Journal of consumer psychology* 13, 1-2 (2003), 29–39.

[19] Osnat Mokryn, Veronika Bogina, and Tsvi Kuflik. 2019. Will this session end with a purchase? Inferring current purchase intent of anonymous visitors. *Electronic Commerce Research and Applications* 34 (2019), 100836. https://doi.org/10.1016/j.elerap.2019.100836

[20] Wenzhao Mu, Sharron J Lennon, and Wenqiao Liu. 2020. Top online luxury apparel and accessories retailers: what are they doing right? *Fashion and Textiles* 7, 1 (2020), 1–17. https://doi.org/10.1186/s40691-019-0197-x

[21] Francis J Mulhern. 1999. Customer profitability analysis: Measurement, concentration, and research directions. *Journal of interactive marketing* 13, 1 (1999), 25–40.

[22] United Nations. 2020. ESTIMATES OF GLOBAL E-COMMERCE 2019 AND PRELIMINARY ASSESSMENT OF COVID-19 IMPACT ON ONLINE RETAIL 2020. *UNCTAD Technical Notes on ICT for Development* 18 (2020). https://unctad.org/system/files/official-document/tn_unctad_ict4d18_en.pdf

[23] Chanyoung Park, Donghyun Kim, Jinoh Oh, and Hwanjo Yu. 2015. Predicting User Purchase in E-Commerce by Comprehensive Feature Engineering and Decision Boundary Focused Under-Sampling. In *Proceedings of the 2015 International ACM Recommender Systems Challenge* (Vienna, Austria) *(RecSys '15 Challenge)*. Association for Computing Machinery, New York, NY, USA, Article 8, 4 pages. https://doi.org/10.1145/2813448.2813517

[24] Werner J. Reinartz and V. Kumar. 2003. The Impact of Customer Relationship Characteristics on Profitable Lifetime Duration. *Journal of Marketing* 67, 1 (2003), 77–99. https://doi.org/10.1509/jmkg.67.1.77.18589

[25] Peter Romov and Evgeny Sokolov. 2015. RecSys Challenge 2015: Ensemble Learning with Categorical Features. In *Proceedings of the 2015 International ACM Recommender Systems Challenge* (Vienna, Austria) *(RecSys '15 Challenge)*. Association for Computing Machinery, New York, NY, USA, Article 1, 4 pages. https://doi.org/10.1145/2813448.2813510

[26] Humphrey Sheil, Omer Rana, and Ronan Reilly. 2018. Predicting purchasing intent: Automatic Feature Learning using Recurrent Neural Networks. *arXiv preprint arXiv:1807.08207* (2018). https://arxiv.org/pdf/1807.08207

[27] Catarina Sismeiro and Randolph E Bucklin. 2004. Modeling purchase behavior at an e-commerce web site: A task-completion approach. *Journal of marketing research* 41, 3 (2004), 306–323.

[28] Madiraju Srilakshmi, Gourab Chowdhury, and Sudeshna Sarkar. 2022. Two-stage system using item features for next-item recommendation. *INTELLIGENT SYSTEMS WITH APPLICATIONS* 14 (2022), 200070. https://doi.org/10.1016/j.iswa.2022.200070

[29] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 3104–3112.

[30] Hasan Tercan, Christian Bitter, Todd Bodnar, Philipp Meisen, and Tobias Meisen. 2021. Evaluating a Session-based Recommender System using Prod2vec in a Commercial Application. In *Proceedings of the 23rd International Conference on Enterprise Information Systems - Volume 1: ICEIS,*. INSTICC, SciTePress, 610–617. https://doi.org/10.5220/0010400706100617

[31] Dirk Van den Poel and Wouter Buckinx. 2005. Predicting online-purchasing behaviour. *European journal of operational research* 166, 2 (2005), 557–575.

[32] Flavian Vasile, Elena Smirnova, and Alexis Conneau. 2016. Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM Conference on Recommender Systems*. 225–232.

[33] Zhenzhou Wu, Bao Hong Tan, Rubing Duan, Yong Liu, and Rick Siow Mong Goh. 2015. Neural Modeling of Buying Behaviour for E-Commerce from Clicking Patterns. In *Proceedings of the 2015 International ACM Recommender Systems Challenge* (Vienna, Austria) *(RecSys '15 Challenge)*. Association for Computing Machinery, New York, NY, USA, Article 12, 4 pages. https://doi.org/10.1145/2813448.2813521

[34] Jinyoung Yeo, Sungchul Kim, Eunyee Koh, Seung-won Hwang, and Nedim Lipka. 2017. Predicting Online Purchase Conversion for Retargeting. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 591–600. https://doi.org/10.1145/3018661.3018715

[35] Valarie A Zeithaml, Roland T Rust, and Katherine N Lemon. 2001. The customer pyramid: creating and serving profitable customers. *California management review* 43, 4 (2001), 118–142.

[36] Ming Zeng, Hancheng Cao, Min Chen, and Yong Li. 2019. User behaviour modeling, recommendations, and purchase prediction during shopping festivals. *Electronic Markets* 29, 2 (2019), 263–274.