PSAC: Context-based Purchase Prediction Framework via User's Sequential Actions

Wei-Cheng Chen
Research Center for Information Technology Innovation, Academia Sinica,
Taipei, , Taiwan
jimmyweicc@citi.sinica.edu.tw

Chih-Yu Wang
Research Center for Information Technology Innovation, Academia Sinica,
Taipei, , Taiwan
cywang@citi.sinica.edu.tw

Su-Chen Lin
Verizon Media
Taipei, Taiwan
suchenl@verizonmedia.com

Alex Ou
Verizon Media
Taipei, Taiwan
oualex@verizonmedia.com

Tzu-Chiang Liou
Verizon Media
Taipei, Taiwan
tcliou@verizonmedia.com

ABSTRACT

Along with the daily operation of e-commerce web services, a significant quantity of data has been recorded. The research of user's behaviors based on the collected data has generated intense attention for accurately offering services that can match the customer's needs and predict the purchase actions. Traditionally, most of the researches utilize only the behavioral instances between users and products, i.e., browse or click history, and session status. However, these features provide only a fundamental knowledge of the given user rather than the rationale behind their actions. We find that query should play an important role as well as it is the main entry point for users when arriving e-commerce website. Since users utilize queries to decide the direction of succeeding event, the semantic meanings of these queries demonstrate a particular link with the action.

In this paper, we propose the **P**rediction framework that analyzes User's **S**equential **A**ctions via **C**ontext (PSAC) to exploit the connection between the user's searching keywords and behaviors to investigate their ultimate intention on an e-commerce website and improve the purchase prediction accuracy. We utilize the e-commerce dataset provided by Yahoo Taiwan, one of the largest web services provider in Taiwan. According to our preliminary analysis, we design a session-based structure to deal with the environment-shifting (influenced by coexisting fashion), and experience-shifting (changed through user's actions) issues which we observed in the dataset. In the simulation section, we apply two deep learning frameworks to perform the prediction task. Experimental results confirm that queries serve as a critical matter in perceiving a user's purchasing intention. Moreover, the proposed framework could significantly improve the prediction accuracy compared with baseline methods.

CCS CONCEPTS

• **Information systems → Electronic commerce**; **Recommender systems**; *Data analytics*; • **Applied computing → Online shopping**;

KEYWORDS

user behavior analysis, e-commerce purchase prediction, deep learning

1 INTRODUCTION

1.1 Background

Along with the Internet evolution, electronic commerce (e-commerce) has generated enormous interests in the past few years. Global retail e-commerce industry's market sales have recorded around 20.77% of compound annual growth rate (CAGR) from 2014 to 2018 and are looking forward to sustaining the growth pace until 2021 [1]. To cope with increasing competitions, companies and researchers race to develop strategies for matching customers' needs and for optimizing companies' profit. Some examples are personalized advertisement [17], upselling to existing customers [10], exclusive offers, and account-based marketing [16]. However, these solutions often require the ability to perceive the customer's intention to gain advantages in the competition from other platforms.

Recently, researchers explored the possibility to identify user's intention through the interactions between users and products, such as query terms and browsing/purchasing history. Query term, for instance, would roughly illustrate the needs the user would like to be satisfied. When the intent of the user could be identified, the system could redirect the users to proper products recommendations. For example, Adhikari *et al.* [2] employ Customer Interaction Networks (CIN) to investigate the relationship between queries and indicate that these relations can significantly improve searching quality. On the other hand, Kumar *et al.* [11] utilize user interactions data on a mobile application, such as activities properties, dwell time, and query contents, to learn the quality of search engine's feedback.

Despite query, researchers also studied other types of behaviors we can collect and analyze in e-commerce websites, such as clickstream, product quality, and user profiles. In particular, Huang *et al.* [6] examine mobile phones users' behavior across different e-commerce platforms. In their study, they claim that purchase decisions tend to happen promptly, and spatiotemporal factors such as time, regions, and platforms, influence shopping behaviors. Eventually, their simulation indicates that user's e-commerce platform preference is predictable. Similarly, Zhou *et al.* [20] offer that it is possible to improve prediction performance by utilizing micro behaviors such as comments, carting, and ordering. Ni *et al.* [14] aim at constructing a universal user representation via multiple e-commerce tasks. After modeling behaviors' and items' features into sequences of user behaviors, they employ deep Recurrent Neural Network (RNN) architecture and attention-based techniques for multi-task learning for ten days of data. Additionally, Lo *et al.* [12] utilize extended real-world data to build a cross-platform analysis of user behavior for more general purposes. They discover evidence that purchase intent will gradually increase through time and significantly soar three days before purchase. Remaining studies discuss topics including the sparseness in e-commerce context [8], exploration of user-item pairwise relationship [15], and online advertising [19], et cetera.

1.2 Rationale of Our Approach

From previous literature, we see that the researches in purchase intention prediction vary mainly with dataset's attributes. Nevertheless, the approaches that employ queries to perceive purchase intention is still scarce. In this study, we collaborate with Yahoo Taiwan which incorporates its e-commerce platform's data. In particular, we collect the dataset describing various behaviors made by its users, e.g., actions (click, view, buy), connection properties (timestamp, time spent), and the most critical piece of information - entered queries. Different from most existing researches, this paper focuses on the construction of user representation via their online actions to cope with e-commerce purchase prediction. Moreover, we utilize the relationship between users' textual inputs and their actions to realize his/her intention. Furthermore, we propose that users behavior are usually time-sensitive in two manners, that is, *environment-shifting* and *experience-shifting*. These phenomena illustrate the evolution of the user's actions through the purchase activity, which occurs in actual e-commerce platforms.

According to the above observations, we establish **P**rediction framework that analyzes User's **S**equential **A**ctions via **C**ontext-based information (PSAC), a prediction system based on session-based representation which records the behavioral sequence of each user. Meanwhile, we apply corpus embedding techniques to encode the latent information of entered queries and derive related attributes from the sequence of entered queries. Finally, the system utilizes two deep learning frameworks to analyze, store, and predict a user's intention under different scenarios. The simulation results indicate that the sequence of query contents is an essential feature in learning user's shopping intention and points out the difference between the two frameworks. Our main contributions in this paper include:

(1) We implement a **P**rediction framework that analyzes User's **S**equential **A**ctions via **C**ontext-based information, which predicts user's purchase intention via query-based information and their corresponding actions.
(2) We investigate the behavior of the real-world user on e-commerce activity that provides further information in predicting the user's purchase intention.
(3) We confirm that query factors are critical in intent and action predictions, which can be utilized in future works and industries application.

The remainder of the paper proceeds as follows. Section 2 consists of three parts: a description of the datasets, the investigation of data, and the preprocessing procedures. Section 3 describes the implementation of e-commerce purchasing prediction system via time-shifting scheme and the deep learning frameworks. Finally, Section 4 examines the simulation results, and Section 5 presents the conclusions.

2 DATASETS EXPLORATION

In this paper, we use a dataset provided by Yahoo Taiwan, which contains a variety of behaviors of its e-commerce platform users. Most importantly, the data not only document user's activities on the platform but also save their searching queries, a textual expression used by the customers to find the products they want.

2.1 E-commerce Dataset Preparation

Our sample dataset consists of real-world transaction records from the e-commerce platform which spans six months. Specifically, the dataset consists of three types of information: click, view, and buy, of over a million unique user's online actions, including 12M, 10M, and 1.2M records individually. Both click and view records specify the query terms the user entered before such actions. For further merging procedure, we label each record in the dataset with a session id s which specifies the belonging connection. The dataset contains the following information:

2.1.1 Click Records. Each click action represents the *click* action of a specific user after he/she enters an arbitrary searching query. Explicitly, numerous information will be stored, i.e., the user's id, entered queries, product's id, and the time that the action occurs. Because the click behavior demonstrates a one-to-one connection between entered queries and product, it can be seen as a strong assumption that the user has a great interest in the product.

2.1.2 View Records. Similar to click records, viewing data record the same set of features as clicks data. Nevertheless, there is a slight difference in the product's attributes. In each record, view actions stand for the operation of a user skimming through a shopping page. Compared with click actions which cover only a single product, a view record saves plural items on a single page after searching for arbitrary keywords.

2.1.3 Buy Records. Finally, buy dataset describes each purchase occurrence that happens during the targeting period. As for the data information, each buy record consists of all previous attributes except for the entered query. We list each dataset's attributes in Table 1.

Table 1: Features in each dataset

Property	Symbol	Explanation	Click	View	Buy
user id	u	user ID of current session	o	o	o
session id	s	session ID of current session	o	o	o
start time	t_s	starting time of current action	o	o	o
query	q	entered query or queries of current action	o	o	x
product id	p	product ID of current action	o	o	o

2.2 Exploratory Data Analysis

To obtain a brief knowledge of users' behavior on Yahoo Taiwan e-commerce platform, we conduct initial analysis on three topics: (i) query usage, (ii) action behavior, and (iii) session statistics. Furthermore, we examine the analysis under monthly-basis.

2.2.1 Query Usage. First of all, we find out that for each month in the dataset, the amount of unique queries ranges from 210K to 260K. Around 70% of queries were found unrepeated in the dataset. Beckmann *et al.* [3] also state this phenomenon that users tend to define personal keywords to describe the products. While an extensive data like this is possible for causing oversize dimension problems during the embedding process, we set two terms to tackle the issue, i.e., *Count Ratio* ($CR_{m,C}$) and *Space Ratio* ($SR_{m,C}$) of each month m. We represent the calculation in Eq. (1a) and (1b) and the bar charts in Figure 1(a) and 1(b).

$$CR_{m,C} = \frac{\text{number of } q_i \text{ in month } m \text{ with } c_{i,m} \geq C}{n_m}, \tag{1a}$$

$$SR_{m,C} = \frac{\sum_{i=1}^{n_m} \sigma_i * c_{i,m}}{\sum_{i=1}^{n_m} c_{i,m}}. \tag{1b}$$

where n_m is the amount of unique query in m, $Q_m = \{q_i | \ i = 1...n_m\}$ is the set of unique queries in m, $c_{i,m}$ is the number of occurance of q_i in m, $\sigma_i = 1$ if q_i in month m with $c_{i,m} \geq C$ else $\sigma_i = 0$, and C is a predefined constant ranges from one to five.

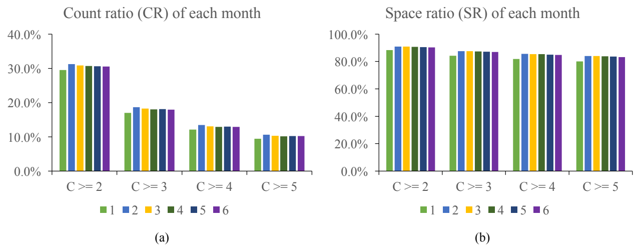

Figure 1: Count ratio $CR_{m,C}$ and Space ratio $SR_{m,C}$ under month m and Count C.

In 1(a), each segment in the chart represents the number of different query q_i occurs more than C times in month m. We can see that there is a similar distribution across six months. Around 30% of queries appear at least two times, 20% and 13% of queries occur at least three and four times, and lastly, 10% of queries appear five or more times. The figures are not only credible that most users use an independent query in searching actions but also demonstrate the diversity of e-commerce searching. Moreover, we define *Space Ratio* in 1(b) to illustrates the query's coverage ratio. For instance, $SR_{m,5}$ computes the coverage ratio of queries that appear at least five times in month m. We can see that across all months, although $CR_{m,C}$ diminishes when we increase C, the value of corresponding $SR_{m,C}$ retain 80% of the corpus space. The results indicate that users on e-commerce platforms tend to reuse queries during their searching events. Furthermore, it is conceivable and economical to learn the figuration of e-commerce queries data via a compact dataset by only preserving the queries with multiple counts.

2.2.2 Action Behavior. Customers' action exhibits various clues as well. Figure 2 describes the time difference between the start of action (click/view) with the action to add products into the cart. Specifically, the difference is computed via action's t_s and corresponding buy's t_s (after matching with the same u and p).

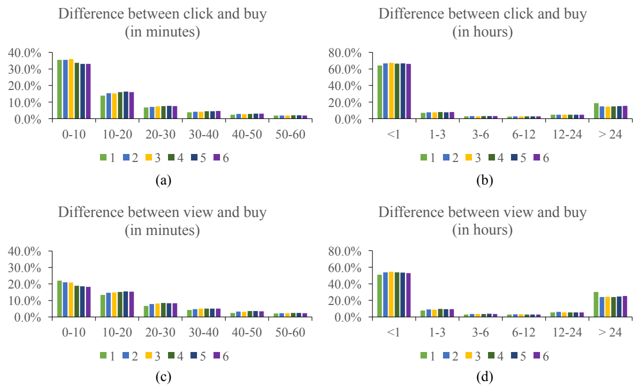

Figure 2: Different behavior of Click and View with the following carting.

Furthermore, for both click and view actions, we separate the figures into minutes and hours sections. Additionally, the first bars within 2(b) and 2(d) include all elements of 2(a) and 2(c). Comparing 2(a) with 2(c), we can see that around 35% of click actions have a 10 minutes gap with the following buy action. On the other hand, nearly 20% of view actions lead to a buy action within 10 minutes. Moreover, a significant difference between 2(b) and 2(d) locates at the last bars, which depicts the portion of users carting the products after twenty-four hours. For view actions, there is a slight excess in carting after a day compare to click actions. These observations suggest two arguments: (i) A click action is faster to induce a buy action than a view action; and (ii) An user who is still browsing the platform might require more time to make the purchase decision. These indications are useful for companies to design different strategies for drawing customers' intent.

2.2.3 Session Statistics. Lastly, we investigate the number of queries used by users having different intention in this section. To begin with, we combine our click and view records via session's id s and mark each action as purchased or non-purchased according to the buying dataset. Details for merging will be presented in Section 2.3. Afterward, we divide all constructed sessions into two parts: purchased sessions and non-purchased sessions. Figure 3 describes the related information.

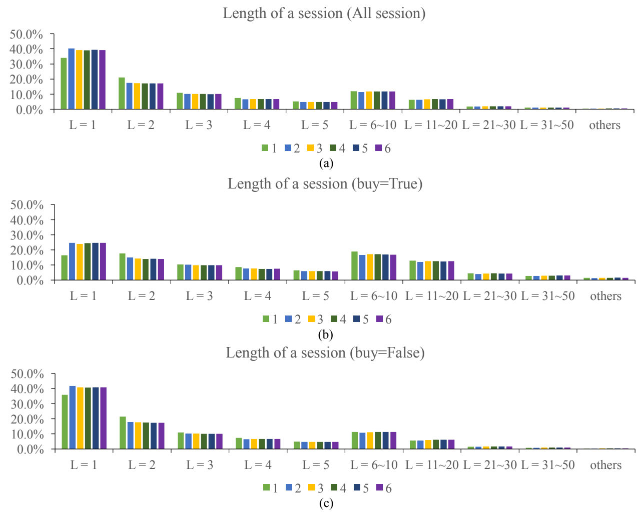

Figure 3: Length between different type of session. L stands for the number of actions in the given session.

It is obvious to see the difference between 3(b) and 3(c) that about 40% of non-purchased users use only one query during their session. On the contrary, only 15-25% of purchased users use one query through their session. Moreover, for users who purchase the product eventually, around 50% of the customers generate a medium or large sequence of actions ($L \geq 6$) within a single session. The appearance of session length demonstrates that customers who might make the deal at the end have more thoughts and reactions to achieve their goal in given operation time.

Like most financial issues, one of the primary problem in e-commerce purchase prediction is the existence of imbalance data [4]. Typically, most of the platform users have no absolute intention before they start searching for products. It is more probable that they were either entertained by other websites' commercial or kept pondering over the purchasing decision. As a result, a majority of users leave the website without making a purchase. For example, the size of non-purchased records in our dataset is 22 to 30 times the size of purchased records. Without balancing the training data, preliminary results are likely to be overrated. Therefore, we perform the random over-sampling approach on the training data, that is, randomly multiplying true cases until the number of both cases are similar. As for the testing set, we do not balance the data since it needs to fit the circumstance in the real world.

2.3 Data Preprocessing

To utilize the e-commerce data, we illustrate the procedure to merge separate datasets in this section. We first utilize the session id s to merge the action data into a unified user behavior dataset. In the Web industry, a session defines the login event after a user connects to the hosting server. Specifically, we record all of the activities during a given session to illustrate the user's behavior, which is beneficial for learning the development of their intentions dynamically. According to our dataset, Yahoo Taiwan specifies its session as a fixed period after any user creates a connection to their server. If an individual user browses the pages more than the given period, they will create multiple sessions according to the actual spending time.

The reason why we construct the data on a session-level is for the rationale behind searching conventions. In the real world, a user's searching habits and related actions are usually regarded as time-sensitive, closely connected with prior and latter instances. This phenomenon manifests in two aspects: (i) experience-shifting and (ii) environment-shifting. As for the first concept, it is convinced that human often generates interests as the time shift. For example, when a customer is entering a shop, he or she might initially search for the thing using a general description, such as shoes, bags, or computer. Typically, people take these actions as wandering and waiting; they either describe fewer details about the item needed in the first place or did not realize the subject they are searching. Furthermore, they might want to compare the same kind of products manufactured by a different company. Such actions are likely to last for a while until a suitable commodity is found. Usually, customers tend to check their favorite brand or look for specific functionality and appearance. In addition, some buyers will calculate their affordable budget. For example, a user who start the search of *bags* might end up checking *Saint Laurent leather shoulder bag* or *backpack 13' notebook waterproof.* For consumers already have a preference, they are likely to finish the previous process in lesser time. Also, the behavior exists not only in off-line business but also the online platform. With the advantage of technology, customers have more resources and information from the Internet, which makes the experience cycle more transient. Moreover, this process often accompanies with several clicking actions which help the user for getting more product details. As for the second concept, we will describe the rationale in Section 3.3.

To construct a session-level behavior sequence, we perform several preprocessing steps on the raw data. At first, we need to define two groups of features from our dataset: the first set describes user's actions after establishing a connection. The other group of features indicates the textual contents of users' queries, which requires Natural Language Preprocessing (NLP) techniques to constructs the latent representation.

2.3.1 User Behavior Sequence (UBS). To illustrate history movements after each user logs onto the server, we transform the session-based data into the User Behavior Sequence (UBS) to represent each session's properties. During the merging process, we first define the click and view records as primary actions and the buy records as supportive information. Then, we combine primary actions having the same session id s and sort all of the records by primary actions' start time t_s. As for the supportive information, we employ the time information to check whether each action leads to a purchase event. If any buy record's start time t_s is less than any primary action's start time, and they share the same user id u and product id p, we mark the primary activities afterward as purchased. Moreover, we ignore repeated primary actions, which defined as two consecutive actions having a time gap of fewer than 5 seconds. In general, this phenomenon might be a consequence of the network error or double-clicking rather than actual intent. For each logged user, we also sort his/her purchasing records by start time before the merging process.

2.3.2 Context-aware Latent Representation. After constructing the structure of UBS, we turn our attention to the query information in the dataset. To utilize the semantic features of each query, we perform several NLP techniques to construct queries' latent representation and integrate the features with UBS.

First, we would like to construct the query corpus from all sequences in UBS. For each sequence, we concatenate each primary action's entered queries into a sequence and tokenize the sequence using white space. Besides, there is no stop words replacement or term normalization (stemming and lemmatization) during the preprocessing since the existent of redundant inputs are typical behavior in e-commerce. After these operations, we build up a corpus consists of over 165M searching sequences in a more structured status. Nevertheless, the corpus space is too large to utilize which requires further processing. To generate the embedded representation, we trained through word2vec [13] to project the input corpus into a lower dimension to recognize the latency representation. Based on the conclusion in Section 2.2.1, we only take the word which occurs at least five times in the corpus into consideration. Furthermore, we empirically set the target dimension as 100, which is significantly smaller than the original vector spaces.

After generating the representation of current corpus, we extend the constituents of UBS and compute several context-related attributes. For each occurence of searching, we assign two query-related features (i) query length l_i^1 and (ii) query amount l_i^2. As for the query contents, we concatenate the embedded representation of linked q_i via two approaches. The first approach is to append the average of complete semantic embedding while the second approach considers the representation thoroughly, denoted as w_{avg} and w_{full} respectively. Since we like to investigate whether each action of the user gradually affects their final decision, we further define a parameter *pos* to state the query's position inside the linked session. For a latter position, it is plausible that the user has a more precise thought on the product he/she wants and a more thoroughgoing intention in making the orders or not. The definitions of advanced terms and context-related attributes are demonstrated in Table 2.

2.3.3 The Input Layers Settings. To understand the efforts of each feature, we construct the input layer under the following context: For the baseline in comparison, we only consider UBS's standard features, i.e., time gap t_d and the action type β_i. As for remaining settings, we concatenate the baseline feature vector with the following combination of advanced terms.

(1) UBS: each record consists of n of actions and the time the user spent (t_d + β_i);
(2) UBS$^+$: each record is the concatenation of UBS, l_i^1, and l_i^2;
(3) PSAC$_a$: each record is the concatenation of UBS$^+$, w_{avg}, w_s, and w_v;
(4) PSAC$_f$: each record is the concatenation of UBS$^+$, w_{full}, w_s, and w_v;
(5) PSAC$_f^+$: each record consists of PSAC$_f$ and *pos*.

Table 2: Advanced terms

Property	Symbol	Explanation
time gap	t_d	The difference between current action (i)' and previous action (j)'s starting time: t_s^i and t_s^j. For the last action in each UBS, we assign a dummy value τ as the difference.
action type	β_i	The type of current action, either click or view.
query length	l_i^1	The length of the entered query q_i.
query amount	l_i^2	The number of the terms after q_i been tokenized.
word2vec	w_{q_i}	A value represents the word2vec layer of given query q_i. Based on the computation approach, denoted as w_{avg} or w_{full}.
word2vec similarity	w_s	A value represents the similarity of word2vec layer between current action (i) and previous action (j): w_{q_i} and w_{q_j}. It is computed with the cosine similarity. If current action is the first action, set the value as one.
word2vec variance	w_v	A value represents the variance of a input sequence's all entered queries' word2vec layer. If the input sequence has one action only, set the value as zero.
position	pos	A value represents current action's position within the belonging session.

Since the main purpose of this paper is to probe the effectiveness of query-based dataset in perceiving user's purchase intention, we ignore context-related features in the baseline method. For the second and third combination, we consider several query-related features, including query length and amount, latency representation, similarity, and variance. Moreover, we apply different computation approach to incorporate the semantic values. Finally, the position attribute is added in the vector to examine the process that the user reshape their thoughts during the connection.

3 PURCHASING PREDICTION SYSTEMS VIA TIME SHIFTING STANDARD

In this section, we propose the Prediction framework that analyzes User's Sequential Actions via Context (PSAC). Firstly, we define several situations to apprehend human's behavior and the concept of time-shifting mentioned in previous sections, including (i) experience-shifting and (ii) environment-shifting. Specifically, we will consider the evolution of queries reshaping and a data-process that takes contemporary fashion into account. For the experiments, we apply two deep learning models: (i) Deep Neural Network (DNN) and (ii) Long Short-Term Memory Networks (LSTM), where we observed opposite views between two frameworks.

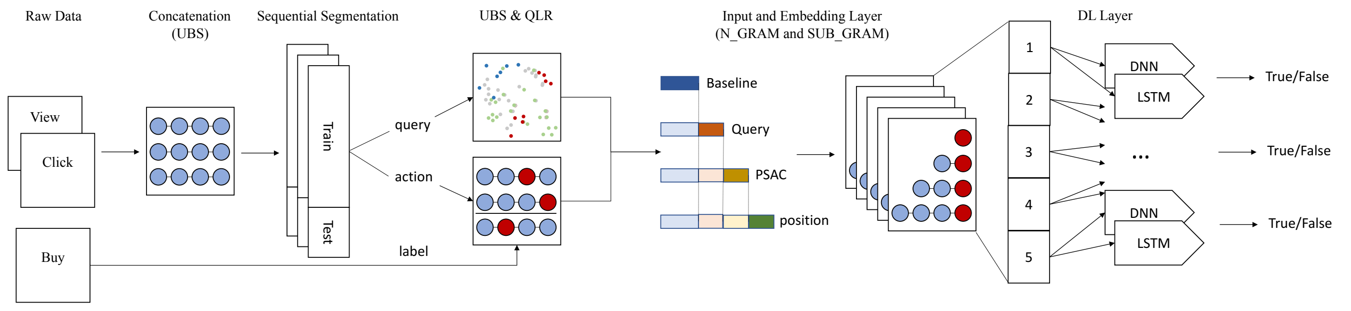

Figure 4: Framework of PSAC

3.1 Framework overview

In short, we illustrate the framework of PSAC in Figure 4. We first combine the view and click instances with session id s. The preliminary output will be used to construct the UBS. Next, to deal with environment-shifting, we apply sequential segmentation and divide the data into training and testing sets to simulate the actual condition. For each segment, we extract two parts of data, namely, query corpus and action sequences. As for the query, we apply context embedding techniques to project the corpus into lower dimensional representation. As for action, we label each action as purchased or non-purchased based on the buy dataset's start time t_1, user id u, and product id p. After merging each action in UBS with its corresponding textual feature, we compute several advanced terms, e.q., similarity, variance, and position attributes. Finally, we deal with the imbalance in e-commerce data and establish different scenarios to investigate the effect of query-based features. In the experiment, we implement the proposed system under two deep learning frameworks (DNN and LSTM) for purchase prediction.

3.2 Experience-shifting of UBS

Recall that in the real world, customers shape their idea through time and experience when searching on the internet. With the session-based data structure, we use the following settings to consider this property. After each construction of the input layer, we thus define two parameters: N_GRAM (M) and SUB_GRAM (μ) where $\mu = 1 \ldots M$. With a given M, we first gather UBS with a length of n where $n \geq M$. Then, each UBS will be separated into $n - M + 1$ of subsequences, and we denote the set of subsequences as $\text{UBS}_{n,M}$. Also, each subsequence owns n actions. Furthermore, we regard each instance in $\text{UBS}_{n,M}$ as an independent action series. Meanwhile, the final position in each series is the prediction target for purchase intention. We illustrate the construction of $\text{UBS}_{5,4}$ in Figure 5; we can see that a UBS with five actions separates into two subsequences if M is four (action_{1-4} and action_{2-5}).

Secondly, both subsequences generate four action series and each series consist of $\mu = 1, 2, 3, 4$ continuous actions. Such a design is capable for us to understand whether more actions benefit the proposed system in learning user's intention. Eventually, the number of subsequences built from our data is represented in Table 3.

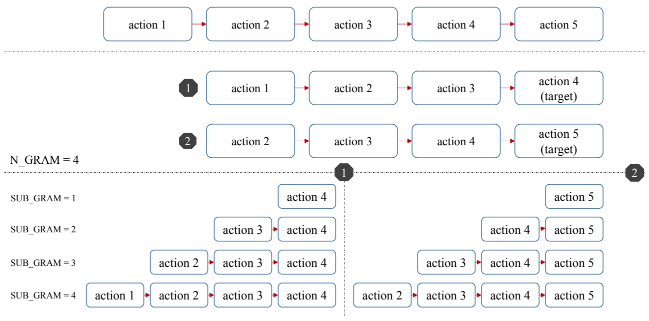

Figure 5: N_GRAM M and SUB_GRAM μ: A sketch of the experience-shifting when M equals four.

Table 3: Number of subsequences

M	Training	Testing
1	32,848,118	14,077,765
2	26,045,458	11,162,339
3	21,767,795	9,329,055
4	18,653,252	7,994,251
5	16,241,526	6,960,654

3.3 Environment-shifting of UBS

We then introduce the idea of environment-shifting, that is, the latest trend or craze toward certain products in the market. Practically, people are likely to put focuses on the hottest products or the latest fashion in the market. Furthermore, companies often enlarge the current by releasing commercials or discounts whenever they want to attract potential consumers. To simulate the real condition, we utilize the rolling-based approach to train the model separately. Explicitly, we segment the data into several pairs of subsets for the simulation: the first subset will cover x days of data to build UBS as the training set, and the other defines consecutive y days of data to construct UBS as the testing set. We execute the training process separately until all segments were done and aggregate the results to get the whole pictures. We demonstrate the process in Figure 6.

Moreover, we balance the training set to avoid overfitting and remain the imbalance in the testing set to fit reality. In sum, the advantages of sequential segmentation are 1) it is more reasonable for companies to use prior data as the training set of following

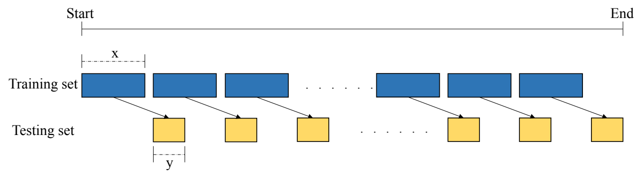

Figure 6: Sequential segmentation to capture contemporary trends

data because we can not foresee the behavior of future users in reality, and 2) it is more manageable since each training utilize a smaller dataset compared with the original one. In this paper, we empirically set the period of x, y as 28 and 14. Eventually, we divide our dataset into twelve pairs of subsets, and each pair has a month of data as the training set and the afterward half month of data as the testing set.

3.4 Deep Learning Frameworks

In this paper, we want to explore whether the usage of queries help us to distinguish potential buyers from other users. To achieve the goal, we adopted two types of deep learning structure: Deep Neural Network (DNN) and Long Short-Term Memory Networks (LSTM) to predict a user's purchasing intention.

3.4.1 Deep Neural Network, DNN. DNN is the most common model in the deep learning field. With the help of tuning parameters through gradient descent and a different activation function, it is possible to build up the optimal objective functions for targeting problems.

3.4.2 Long Short-Term Memory Networks, LSTM. LSTM is one of the extensions from vanilla RNN proposed by Hochreiter *et al.* [5] to deal with vanishing and exploding gradient problems. With the capacity of learning long-term dependencies, LSTM is widely used and demonstrates decent performance on many recent studies. To solve the problem of vanishing and exploding gradient, LSTM replaces the kernel of RNN structure with the idea of cell state, a representation of the state of input sequence transmitted information through the entire network. Moreover, LSTM demonstrates the capacity to add or remove information from the cell state by a mechanism called gates.

3.4.3 Hyper-parameter Setting and Implementation Details. To make sure that two models train under a similar condition, we construct both models with the structure having an approximate number of untrained nodes. For DNN, we build up a five layers network with one input layer of 1024 nodes, three hidden layers (1024, 256, and 256 nodes particularly), and one output layer of 2 nodes while our task is a binary classification. On the other hand, we construct the LSTM model as a four layers structure (one input layer of 256 nodes, two hidden layers of 256 nodes, and one output layer of 2 nodes). Moreover, layers in both models are fully connected, and the input layer's shape will match the size of the embedding layer given in Section 2.3.3. As for the naive setting (under $M = 1$ and use UBS$_{base}$), both models deploy roughly 1.3M untrained weights during the training process.

Table 4: Structure of DNN and LSTM models

Model	Layer structure	# untrained weights
DNN	1024-1024-256-256-2	1.3M
LSTM	256-256-256-2	1.3M

For specific model configuration of DNN, we apply ReLU as nonlinearity on the input layer and hidden layer. Because our targeting problem is a binary classification, we choose softmax function as the output layer's activation function. Furthermore, we train our experiments with Adam [9] using the following hyper-parameter settings: learning rate α of 0.001, the exponential decay rate β_1 and β_2 of 0.9, and 0.999 and implement categorical cross-entropy as the loss function. As for LSTM, we select RMSprop [18] as the optimizer with learning rate α of 0.001 and the exponential decay rate ρ of 0.9 and use the same loss function as DNN. We illustrate the architecture in Table 4.

Finally, in the training process, we use 30% of the data as the validation set, and the mini-batch size is set to be 500 across all data sets. The results reported were obtained after 50 epochs of training over the sample dataset. To avoid the simulation processes trapped in a local minimum, we implement a callback mechanism which interrupts the simulation if the loss was unchanged over the last ten epochs.

4 SIMULATION RESULTS

We evaluate the performance of PSAC with simulations utilizing the Yahoo! Taiwan dataset. For each simulation, we select the training and testing data sharing the same segment and extract the set of attributes under a given scenario. After constructing the input layer of each segment, we process the modeling on training data with 30% of instances as the validation set. Once finishing the training, we evaluate the model's ability to predict testing data in the following sections.

4.1 Performance Indicators

According to [7], as the increase in data skewness, ROC would be a misleading indicator. Since the imbalance in our data distribution is highly significant (22x-30x), we employ the F_1 score as the performance evaluation principle.

$$F_1 = 2 * \frac{\text{Pre} * \text{Rec}}{\text{Pre} + \text{Rec}}, \qquad (2)$$

where Pre stands for the precision and Rec is the recall.

4.2 Numerical Results

In the experiment, we analyze the effect of query contents with different environmental settings. Table 5 and 6 describe the numerical results of PSAC for purchase prediction under different scenarios and deep learning framework. Explicitly, the simulation results are computed by taking the average of all segments' outcome. Moreover, the average improvement illustrates the enhancement between current input settings with baseline approach (UBS). Based on the results, we made the following observations.

Table 5: Prediction Performance (DNN, F_1 score)

M	fset	$\mu = 1$	$\mu = 2$	$\mu = 3$	$\mu = 4$	$\mu = 5$	improvement (avg.)
1	UBS	0.1154					
	UBS^+	0.1181					+2.34%
	PSAC_a	0.1175					+1,82%
	PSAC_f	0.1420					+23.05%
	PSAC_f^+	0.1429					+23.83%
2	UBS	0.1132	0.1109				
	UBS^+	0.1139	0.1115				+0.58%
	PSAC_a	0.1133	0.1137				+1.31%
	PSAC_f	0.1346	0.1360				+20.77%
	PSAC_f^+	0.1346	0.1369				+21.17%
3	UBS	0.1103	0.0937	0.0930			
	UBS^+	0.1131	0.1093	0.1071			+11.45%
	PSAC_a	0.1108	0.1113	0.1105			+12.68%
	PSAC_f	0.1317	0.1317	0.1334			+34.47%
	PSAC_f^+	0.1310	0.1319	0.1344			+34.68%
4	UBS	0.1112	0.1061	0.1048	0.1010		
	UBS^+	0.1076	0.1072	0.1045	0.1028		-0.18%
	PSAC_a	0.1069	0.1081	0.1070	0.1077		+1.69%
	PSAC_f	0.1292	0.1296	0.1299	0.1319		+23.22%
	PSAC_f^+	0.1283	0.1292	0.1305	0.1317		+23.02%
5	UBS	0.1075	0.1048	0.1024	0.1006	0.1006	
	UBS^+	0.1067	0.1055	0.1031	0.1023	0.1012	+0.58%
	PSAC_a	0.1053	0.1062	0.1050	0.1059	0.1063	+2.55%
	PSAC_f	0.1271	0.1274	0.1278	0.1262	0.1282	+23.50%
	PSAC_f^+	0.1263	0.1257	0.1271	0.1283	0.1290	+23.46%

Table 6: Prediction Performance (LSTM, F_1 score)

M	fset	$\mu = 1$	$\mu = 2$	$\mu = 3$	$\mu = 4$	$\mu = 5$	improvement (avg.)
1	UBS	0.1158					
	UBS^+	0.1167					+0.78%
	PSAC_a	0.1163					+0.43%
	PSAC_f	0.1435					+23.92%
	PSAC_f^+	0.1439					+24.27%
2	UBS	0.1122	0.1092				
	UBS^+	0.1128	0.1103				+0.77%
	PSAC_a	0.1135	0.1119				+1.82%
	PSAC_f	0.1363	0.1322				+21.27%
	PSAC_f^+	0.1352	0.1345				+21.83%
3	UBS	0.1091	0.1068	0.1014			
	UBS^+	0.1106	0.1080	0.1004			+0.50%
	PSAC_a	0.1081	0.1082	0.1017			+0.23%
	PSAC_f	0.1329	0.1274	0.1253			+21.56%
	PSAC_f^+	0.1325	0.1303	0.1279			+23.20%
4	UBS	0.1065	0.1066	0.0990	0.0892		
	UBS^+	0.1080	0.1077	0.0978	0.0926		+1.26%
	PSAC_a	0.1068	0.1045	0.0991	0.0981		+2.10%
	PSAC_f	0.1302	0.1232	0.1201	0.1156		+22.18%
	PSAC_f^+	0.1299	0.1262	0.1226	0.1180		+24.12%
5	UBS	0.1045	0.1053	0.0968	0.0876	0.0822	
	UBS^+	0.1064	0.1045	0.0955	0.0904	0.0894	+2.33%
	PSAC_a	0.1034	0.1023	0.0967	0.0962	0.0924	+3.64%
	PSAC_f	0.1293	0.1199	0.1152	0.1108	0.1067	+22.58%
	PSAC_f^+	0.1274	0.1218	0.1178	0.1137	0.1089	+24.31%

First of all, we discover a slight increase when we add basic query-related features (query length and query number, UBS^+) to the baseline approach. As for the PSAC which utilizes context-related features (PSAC_a and PSAC_f), the performance of both methods outperform the baseline outputs; however, their results are entirely different. If we take averaged values as the input features, the improvement compared with the first approach is under 10%. On the other hand, if we extend the input layers with the complete queries information (PSAC_f), the performance consistently surpasses the baseline outcomes around 20% to 30%. We conclude that a vast amount of information is missing during the regularization procedure. Furthermore, we can see that query information plays a vital role in perceiving the user's intention. Next, we observe that the results demonstrate that the position attribute is another useful factor. We recognize one to two percent increase after considering the position attribute on most of the session sets with different N_GRAM M.

If we consider only the baseline approach and the enhanced version (UBS, UBS^+), the prediction performance declines as the length of the input sequence (μ) grow. After we take query-related features into account, the influences of μ on these two deep learning frameworks are different. As for DNN model, once we construct the input layer with query-related features (PSAC_a, PSAC_f, and PSAC_f^+), the model's performance gradually raises as μ increases. Since DNN recognizes each action in the input sequence equally during the training process, it is possible for us to comprehend the evolution of the user's query contents globally and to capture their intention more appropriately in an attention-like structure. Additionally, this phenomenon occurs in all M settings. Contrary to DNN, the score of LSTM decreases when applying query-related features if we utilize more μ in training. While the rationale behind RNN-based models pushes itself to put more focuses on closer steps, it is plausible that a more extended sequence might impair the performance. For example, a user might determine his/her shopping list in the middle of the searching operations and continues to

Table 7: Prediction Performance (all models, F_1 score)

M	fset	model	$\mu = 1$	$\mu = 2$	$\mu = 3$	$\mu = 4$	$\mu = 5$
1	PSAC_f^+	DNN	0.1429				
	PSAC_f^+	LSTM	**0.1439**				
2	PSAC_f^+	DNN	0.1346	**0.1369**			
	PSAC_f^+	LSTM	**0.1352**	0.1345			
3	PSAC_f^+	DNN	0.1310	**0.1319**	**0.1344**		
	PSAC_f^+	LSTM	**0.1325**	0.1303	0.1279		
4	PSAC_f^+	DNN	0.1283	**0.1292**	**0.1305**	**0.1317**	
	PSAC_f^+	LSTM	**0.1299**	0.1262	0.1226	0.1180	
5	PSAC_f^+	DNN	0.1263	**0.1257**	**0.1271**	**0.1283**	**0.1290**
	PSAC_f^+	LSTM	**0.1274**	0.1218	0.1178	0.1137	0.1089

browse other commodities afterward. If the user did not pass the thought onto the next action, it is hard for LSTM to perceive the conversion. Finally, we provide the comparison of DNN and LSTM with the PSAC$_f^+$ in Table 7. We can see that LSTM outperforms DNN for shorter action sequences. However, as the users begin to reshape their thoughts and to start a new searching action, DNN is more capable in realizing the connection between each action and query.

5 CONCLUSIONS

With the growing usage of e-commerce platforms, the analysis of user's purchase behavior has attracted attention in the field. Through an integrated prediction framework, the numerical results in our study indicate that query-related features are essential in making purchase prediction on e-commerce platforms. To utilize query-related features correctly, we need to consider several query's attributes, such as user's behavior in reshaping their ideas, the relationship between continuous queries, and the approach to represent query's embedded components. These results also support that both deep learning frameworks have their advantages; however, DNN's performance is more robust in capturing customer's purchasing intention as the searching sequence grow. Furthermore, the construction of session-based data structure is useful in storing customers' behavior sequences, and we provide several mechanisms in explaining the phenomenon of time-shifting in the real world. Such evidence should be of importance in e-commerce purchase prediction.

ACKNOWLEDGMENTS

This work was supported by the Ministry of Science and Technology under Grant MOST 105-2221-E-001-003-MY3 and the Academia Sinica under Grand Challenge Seed Project AS-GC-108-01.

REFERENCES

[1] [n.d.]. Global retail e-commerce market size 2014-2021. https://www.statista.com/statistics/379046/worldwide-retail-e-commerce-sales/

[2] Bijaya Adhikari, Parikshit Sondhi, Wenke Zhang, Mohit Sharma, and B. Aditya Prakash. 2018. Mining E-Commerce Query Relations using Customer Interaction Networks. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*. ACM Press, 1805–1814. https://doi.org/10.1145/3178876.3186174

[3] J.L. Beckmann, A. Halverson, R. Krishnamurthy, and J.F. Naughton. 2006. Extending RDBMSs To Support Sparse Datasets Using An Interpreted Attribute Storage Format. *(:unav)* (2006). https://doi.org/10.1109/icde.2006.67

[4] Liliya Besaleva and Alfred C. Weaver. 2017. Classification of imbalanced data in E-commerce. In *2017 Intelligent Systems Conference (IntelliSys)*. IEEE, 744–750. https://doi.org/10.1109/IntelliSys.2017.8324212

[5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (Nov 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[6] Hong Huang, Bo Zhao, Hao Zhao, Zhou Zhuang, Zhenxuan Wang, Xiaoming Yao, Xinggang Wang, Hai Jin, and Xiaoming Fu. 2018. A Cross-Platform Consumer Behavior Analysis of Large-Scale Mobile Shopping Data. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*. ACM Press, 1785–1794. https://doi.org/10.1145/3178876.3186169

[7] Laszlo A. Jeni, Jeffrey F. Cohn, and Fernando De La Torre. 2013. Facing Imbalanced Data–Recommendations for the Use of Performance Metrics. *(:unav)* (Sep 2013). https://doi.org/10.1109/acii.2013.47

[8] Ru Jia, Ru Li, Meiju Yu, and Shanshan Wang. 2017. E-commerce purchase prediction approach by user behavior data. (Jul 2017), 1–5. https://doi.org/10.1109/CITS.2017.8035294

[9] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]* (Dec 2014). http://arxiv.org/abs/1412.6980 arXiv: 1412.6980.

[10] Bernard Kubiak and Paweł Weichbroth. 2010. Cross- And Up-selling Techniques In E-Commerce Activities. *Journal of Internet Banking and Commerce* 15 (12 2010).

[11] Rohan Kumar, Mohit Kumar, Neil Shah, and Christos Faloutsos. 2018. Did We Get It Right? Predicting Query Performance in E-commerce Search. *arXiv:1808.00239 [cs]* (Aug 2018). http://arxiv.org/abs/1808.00239 arXiv: 1808.00239.

[12] Caroline Lo, Dan Frankowski, and Jure Leskovec. 2016. Understanding Behaviors that Lead to Purchasing: A Case Study of Pinterest. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*. ACM Press, 531–540. https://doi.org/10.1145/2939672.2939729

[13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]* (Jan 2013). http://arxiv.org/abs/1301.3781 arXiv: 1301.3781.

[14] Yabo Ni, Dan Ou, Shichen Liu, Xiang Li, Wenwu Ou, Anxiang Zeng, and Luo Si. 2018. Perceive Your Users in Depth: Learning Universal User Representations from Multiple E-commerce Tasks. *arXiv:1805.10727 [cs, stat]* (May 2018). http://arxiv.org/abs/1805.10727 arXiv: 1805.10727.

[15] Chanyoung Park, Donghyun Kim, Min-Chul Yang, Jung-Tae Lee, and Hwanjo Yu. 2017. Your Click Knows It: Predicting User Purchase through Improved User-Item Pairwise Relationship. *arXiv:1706.06716 [cs]* (Jun 2017). http://arxiv.org/abs/1706.06716 arXiv: 1706.06716.

[16] Michael Rose. [n.d.]. What Is Account-Based Marketing? https://www.forbes.com/sites/forbesagencycouncil/2017/11/01/what-is-account-based-marketing/

[17] Timo Saari, Niklas Ravaja, Jari Laarni, Marko Turpeinen, and Kari Kallinen. 2004. Psychologically targeted persuasive advertising and product information in e-commerce. In *Proceedings of the 6th international conference on Electronic commerce - ICEC '04*. ACM Press, 245. https://doi.org/10.1145/1052220.1052252

[18] Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural Networks for Machine Learning*. (2012).

[19] Shuangfei Zhai, Keng-hao Chang, Ruofei Zhang, and Zhongfei Mark Zhang. 2016. DeepIntent: Learning Attentions for Online Advertising with Recurrent Neural Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*. ACM Press, 1295–1304. https://doi.org/10.1145/2939672.2939759

[20] Meizi Zhou, Zhuoye Ding, Jiliang Tang, and Dawei Yin. 2018. Micro Behaviors: A New Perspective in E-commerce Recommender Systems. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM '18*. ACM Press, 727–735. https://doi.org/10.1145/3159652.3159671