Buying or Browsing? : Predicting Real-time Purchasing Intent using Attention-based Deep Network with Multiple Behavior

Long Guo, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, Bin Cui †
Alibaba Group, Beijing & Hangzhou, China
†School of EECS Key Laboratory of High Confidence Software Technologies (MOE), Peking University
{leo.gl,issac.hlf,rongfei.jrf,binqiang.zhao,yongshu.wxb}@alibaba-inc.com,bin.cui@pku.edu.cn

ABSTRACT

E-commerce platforms are becoming a primary place for people to find, compare and ultimately purchase products. One of the fundamental questions that arises in e-commerce is to predict user purchasing intent, which is an important part of user understanding and allows for providing better services for both sellers and customers. However, previous work cannot predict real-time user purchasing intent with a high accuracy, limited by the representation capability of traditional browse-interactive behavior adopted. In this paper, we propose a novel end-to-end deep network, named Deep Intent Prediction Network (DIPN), to predict real-time user purchasing intent. In particular, besides the traditional browse-interactive behavior, we collect a new type of user interactive behavior, called touch-interactive behavior, which can capture more fine-grained real-time user features. To combine these behavior effectively, we propose a hierarchical attention mechanism, where the bottom attention layer focuses on the inner parts of each behavior sequence while the top attention layer learns the inter-view relations between different behavior sequences. In addition, we propose to train DIPN with multi-task learning to better distinguish user behavior patterns. In the experiments conducted on a large-scale industrial dataset, DIPN significantly outperforms the baseline solutions. Notably, DIPN gains about 18.96% improvement on AUC than the state-of-the-art solution only using traditional browse-interactive behavior sequences. Moreover, DIPN has been deployed in the operational system of Taobao. Online A/B testing results with more than 12.9 millions of users reveal the potential of knowing users' real-time purchasing intent.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Applied computing** → **Online shopping**.

KEYWORDS

e-commerce, recommendation system, purchasing intent prediction, hierarchical attention mechanism, multiple behavior

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330670

ACM Reference Format:
Long Guo, Lifeng Hua, Rongfei Jia, Binqiang Zhao, Xiaobo Wang, Bin Cui †. 2019. Buying or Browsing? : Predicting Real-time Purchasing Intent using Attention-based Deep Network with Multiple Behavior. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330670

1 INTRODUCTION

In the internet era, large e-commerce platforms such as Taobao and Amazon are becoming a primary place for people to find, compare and ultimately purchase products. As an important part of user understanding, it is crucial to know whether a customer is buying or just browsing on an e-commerce application, as it allows for providing better services for both sellers and customers. From the perspective of the sellers, knowing users' current purchasing intent can increase their sales volume and profit margin. When the e-commerce platform has increased confidence that a subset of users are more likely to purchase, it can perform some proactive actions to maximize conversion based on this information. The platform may offer time-limited coupons or create bundles of complementary products to push the users to complete their purchases. From the perspective of the customers, recognizing users' current buying or browsing intent is vital for the e-commerce platform to set appropriate strategies for the recommendation system and search engine to improve user experience.

Previous studies focus on leveraging traditional user behavior, which we call browse-interactive behavior, to predict users' purchasing intent [15, 18, 19, 23]. However, limited by the representation capability and frequency of occurrence of the browse-interactive behavior, e.g., browse, search or collect a product, it is hard to predict users' real-time purchasing intent depending solely on these actions. In other words, these actions contain insufficient information about user behavior patterns that would lead to a purchase in a short time. The purchase intent of a customer may slowly build over time and may not instantaneously lead to a purchase. As a result, it is challenging to identify the moment when the customer finally places the order. To this end, we need some more fine-grained behavior data to model user purchasing behavior patterns.

Table 1: Average number of actions per user per day.

	Browse	Tap	Swipe
Number of actions	42	391	1583

In order to capture users' real-time purchasing intent, we collect a new type of interactive behavior data, which we call touch-interactive behavior. With the rapid development of hardware and

software on mobile devices, we take advantage of the sensors and accelerometers of the mobile devices to automatically glean the real-time context of user interactions, such as the swipe and tap actions. Compared with the browse-interactive actions, the touch-interactive actions occur more frequently. As shown in Table 1, the number of swipe actions and tap actions generated per user per day are 37.7 times and 9.3 times more than that of the browse-interactive actions, respectively. As a result, the touch-interactive behavior contains more rich information about user behavior patterns. For example, we find that some customers would browse the product comments for a long time before they place the order. Such typical patterns can be easily captured by using the touch-interactive behavior. By combing the traditional browse-interactive behavior and the new touch-interactive behavior, we are able to model the user behavior patterns more comprehensively.

However, there exist several challenges in predicting users' real-time purchasing intent. First, the touch-interactive behavior contains less semantic information than the browse-interactive behavior. Therefore, it is challenging to extract useful features from these data to improve the prediction performance. Second, it is necessary to figure out an effective fusion mechanism to combine the browse-interactive behavior and the touch-interactive behavior in order to bring their advantages into full play. Third, due to the complexity of the browsing behavior where the customers with different purchasing intent can appear to be very similar, it is essential to capture common features that can well depict the customers and unique features that would lead to different purchasing behavior.

In this paper, we propose a novel end-to-end deep network, named Deep Intent Prediction Network (DIPN), for the real-time purchasing intent prediction. In DIPN, the user behavior features are automatically learned from the raw data without the need of extensive feature engineering. In particular, we propose a hierarchical attention mechanism to fuse the views extracted from different interactive behavior sources. In the bottom attention layer, we design an intra-view attention mechanism which focuses on the inner parts of the behavior sequence. In the top attention layer, we propose an inter-view attention mechanism that learns the inter-view relations between different behavior sequences. In addition, we propose to train the real-time and long-term purchasing intent simultaneously with the same model. With the multi-task learning, DIPN can capture common features that well depict the customers and unique features that would lead to different purchasing behavior. The contribution of the paper can be summarized as follows:

- We collect a new type of user behavior, the touch-interactive behavior, which contains rich information about user behavior patterns. Together with the traditional browse-interactive behavior, we are able to depict a user from different views for better performance of purchasing intent prediction.
- We propose a deep network DIPN for real-time purchasing intent prediction. A novel hierarchical attention mechanism is proposed to fuse multiple views extracted from different interactive behavior sources. In addition, multi-task learning is introduced to better distinguish user behavior patterns.
- We conduct extensive experiments to evaluate the performance of DIPN in both offline and online settings. Experimental results on a large-scale industrial dataset shows the superiority of DIPN in predicting purchasing intent. In particular, DIPN has been deployed in the operational system of Taobao and adopted in the coupon allocation task at a shopping festival. Online A/B testing shows the benefits of knowing users' real-time purchasing intent.

The rest of the paper is organized as follows. We discuss related work in Section 2, followed by the data description in Section 3. We describe the design of DIPN model in Section 4 and give an overview of the deployment of DIPN in Section 5. We present experiments in Section 6 and conclude the paper in Section 7.

2 RELATED WORK

2.1 Purchasing Intent Prediction

The problem of purchasing intent prediction has been heavily studied, with a variety of classic machine learning and deep learning modelling techniques employed. The earliest work come from the RecSys 2015 challenge [2], which provides a public dataset consisting of 9.2 million user-item click sessions. Given a session, the goal of the challenge is to predict whether the user is going to buy something or not within this session. Romov et al. [15] won the competition using GBM with extensive feature engineering on session summarizing. The other feature-based work includes the ensemble model with neural net and GBM used by [23] and the deep belief networks and stacked denoising auto-encoders by [22]. To reduce the feature engineering work, several work [19, 20, 25] adopt the recurrent neural network (RNN) to model the sequence nature of sessions, where a bi-directional LSTM is used in [19, 25] and a mixture of LSTM is used in [20].

Our work is distinguished from previous work in the following aspects. First, given a history session, our goal is to predict a user's subsequent purchasing behavior within a given time, while the goal of previous work is to predict the purchasing behavior within the session. Our setting is more realistic because in reality we should predict the future behavior based on the current incomplete session. Second, the key difference of our work is that we collect touch-interactive actions to capture the real-time user behavior patterns. As a result, we need to handle several data sources in our model while previous work only deal with a single source.

2.2 Sequence Classification

The task of purchasing intent prediction is closely related to sequence classification. A brief survey by [26] categorizes the sequence classification methods into three groups: feature based methods [1, 13, 29], sequence distance based methods [10, 17, 24], and model based methods [4, 27, 31]. Our work is related to the model based approach, where we use an end-to-end deep network to model the sequences and save extensive feature engineering work. Our work is also related to sentence classification in natural language processing [9, 11, 30]. Text sentences and time series data are similar to each other in that they are both ordered sequences in nature. However, the semantic information contained in these two kinds of sequences are definitely different. Our work differs from the previous work in that we need to handle several different data sources with different formats while in traditional sequence classification, the data usually comes from a single source.

Type	Timestamp	Page	Position	Start Position	End Position	Duration
Open Page	1531640194337	Page_Home	/	/	/	/
Tap	1531640195116	Page_Home	x:336.5, y:473.0	/	/	/
Swipe	1531640196199	Page_Home	/	x:242.5, y:573.5	x:234.0, y:558.5	34
Leave Page	1531640197067	Page_Home	/	/	/	/

(a) Swipe-interactive behavior

Event	Page	Button	Timestamp
2101	Page_Detail	Page_SearchItem_Button-Status	1531640197187
2001	Page_Detail	/	1531640197199
2101	Page_Detail	Page_Detail_Button-Comments	1531640197219

(b) Tap-interactive behavior

Table 2: An example of raw data in the touch-interactive behavior dataset.

2.3 Multi-task Learning

Multi-task learning has been used successfully across various applications of machine learning, from natural language processing [3, 5] and speech recognition [6] to computer vision [8] and recommender systems [14]. By sharing representations between related tasks, multi-task learning can enable the model to capture more underlying factors and generalize better on its original task. Ruder [16] presents an overview of multi-task learning in deep learning, where multi-task learning is typically done with either hard or soft parameter sharing of hidden layers. The hard parameter sharing method is the most commonly used multi-task learning approach, which shares the hidden layers between all tasks and keeps several task-specific output layers. Collobert et al. [5] simultaneously learn several NLP tasks using a language model with embedding lookup table sharing. In [8], multi-task learning is adopted to improve the performance of classifying object proposals using deep convolutional networks. Ni et al. [14] use deep multi-task representation learning to generate user representations for personalization in e-commerce portal. In soft parameter sharing, each task has its own model with its own parameters where the distance between the parameters is regularized. Duong et al. [7] uses l_2 distance for regularization while Yang et al. [28] use the trace norm. Our model is related to the hard parameter sharing method. We propose a novel way by partitioning a user's purchasing intent into three different phases and use multi-task learning to learn the unique behavior that would lead to different purchasing intent.

To the best of our knowledge, our work is the first study that uses the attention-based deep network with multi-learning on multiple user behavior sequences for real-time purchasing intent prediction.

3 DATASET

We build two types of user interactive behavior dataset, i.e., the new touch-interactive behavior and the traditional browse-interactive behavior. In the following, we describe each dataset in details.

3.1 Touch-interactive Behavior

The touch-interactive behavior dataset contains normal users' daily touch-interactive information when using the Taobao app, which is composed of the swipe-interactive and tap-interactive behavior.

The swipe-interactive behavior. This behavior includes four types of basic actions, i.e., *Open Page*, *Leave Page*, *Swipe* and *Tap*. Table 2a shows an example of raw data in the swipe-interactive behavior. A user's swipe-interactive track is a time sequence of actions, consisting of these four basic types of actions. Each action has a timestamp and a page index to identify when and where the action occurs. In addition, the positional coordinates of the action on the touch screen are also recorded. The duration presents how long the action lasts. As shown in Table 3a, for each action at a data point, we extract 14 raw features. The time duration of a swipe, time gap between two actions and positional coordinates of actions are continuous variables. Page indices, action indices and swipe directions (i.e., left/right and up/down) are categorical variables. We conduct discretization on all the raw features to ensure unified inputs for DIPN. The discretization of the continuous variables is described as follows:

- **Position.** The positional coordinates of actions are continuous values, and are discretized according to the resolution of the touch screen. We divide the width of the screen into 17 uniform segments, and the length into 25 segments for one-hot vectors encoding.
- **Swipe Length.** The length of a swipe is encoded into a one-hot vector, the length of which is as twice as the length of the one-hot vectors of position encoding. The reason of applying twice length is that, for a swipe track, we consider the direction of the swipe.
- **Time Gap and Duration.** We apply a step function to encode time gaps between actions and swipe duration as follow:

$$y = \begin{cases} \lfloor x/f_s \rfloor, & x < f_b \\ \lfloor x/f_b + 9 \rfloor, & f_b \le x < 10 \times f_b \\ 19, & x \ge 10 \times f_b \end{cases}$$

where $\{f_s = 100, f_b = 1000\}$ are used for time gap, and $\{f_s = 25, f_b = 250\}$ are used for time duration.

The tap-interactive behavior. This behavior records the information associated with the tap actions, as shown in Table 2b. A user's tap-interactive track is a time sequence of tap actions. Each action has a timestamp and page index to identify when and where the action occurs. There is also an event id to identify whether a user taps on a page or a button. If a button is tapped, the button name is also recorded. As shown in Table 3b, we extract 3 raw features, all of which are categorical variables.

Feature	Dictionary Dim	Embedding Dim
Page Index	224	32
Action Index	4	4
Time Gap	20	8
Tap Position X	17	8
Tap Position Y	25	8
Swipe Start Position X	17	8
Swipe Start Position Y	25	8
Swipe End Position X	17	8
Swipe End Position Y	25	8
Swipe Length on X	34	8
Swipe Length on Y	50	16
Swipe Right/Left	2	2
Swipe Up/Down	2	2
Swipe Duration	20	8

(a) Swipe-interactive behavior

Feature	Dictionary Dim	Embedding Dim
Event Index	2	2
Page Index	200	16
Button Index	500	32

(b) Tap-interactive behavior

Feature	Dictionary Dim	Embedding Dim
Type Index	6	4
Leaf Category Index	19011	32
Top Category Index	102	16
Page Index	181	16
Page Stay Time	179	16
Timestamp	25	4

(c) Browse-interactive behavior

Feature	Dictionary Dim	Embedding Dim
C.F. within one week	20	8
...	...	...
C.F. within one year	100	16
A.F. within one week	20	8
...	...	...
A.F. within one year	100	16
P.F. within one week	20	8
...	...	...
P.F. within one year	100	16

(d) User history feature

Feature	Dictionary Dim	Embedding Dim
Age Level	9	4
Gender	3	2
Buyer Star	17	8
Tm Level	6	4
VIP Level	8	4
Phone Price	11	4
...	....	....

(e) User profile feature

Table 3: Statistics of feature sets used in DIPN.

3.2 Browse-interactive Behavior

The browse-interactive behavior represents the typical behavior users conduct on products when browsing an e-commerce application. It includes five types of actions, i.e., browse a product, search a product, collect a product, add a product to cart and purchase a product. A user's browse-interactive track is a time sequence of these actions. As shown in Table 3c, we extract 6 raw features for each action. The type index represents the type of an action. For the continuous variables, i.e., page stay time and timestamp, we perform similar discretization operation as for the time gap in the swipe-interactive behavior.

A user's purchase behavior has a high correlation with her historical behavior, which can represent the activeness of the user. Active users who behave more frequently in the history may keep this trend in the future, while those who are inactive may stay quiet for a long time. Therefore, we also collect some statistics for the historical behavior based on the browse-interactive behavior. In more details, we count up the frequency of three types of actions in different time windows. Table 3d shows the features extracted for each action, where C.F., A.F. and P.F. represent the frequency of collecting a product, adding a product to cart and purchasing a product, respectively. For each action, we count the frequency of the action within one week, two weeks, one month, three months, six months and one year. To improve personalization, we also collect the user profile dataset containing various users' basic information, such as the age level and gender, as shown in Table 3e.

4 MODEL ARCHITECTURE

We propose a novel deep network, named Deep Intent Prediction Network (DIPN), for the real-time purchasing intent prediction. Figure 1 shows the model architecture of DIPN. It includes an embedding lookup layer which embeds the one-hot vectors of the raw action features into dense vectors, followed by a fully-connected layer. After that, a bidirectional recurrent layer is applied to each user behavior sequence to model the long-term dependencies between complex user actions. Then a hierarchical attention layer is applied to fuse the outputs from the recurrent layer. Finally, DIPN is trained with multi-task learning. In the following, we will introduce each component of DIPN in details.

4.1 Embedding Layer

As introduced in Section 3, we use five groups of features in DIPN, i.e., *User Swipe-interactive Behavior*, *User Touch-interactive Behavior*, *User Browse-interactive Behavior*, *User History* and *User Profile*. In the discretization process, the raw values of every feature are encoded into one-hot vectors, the length of which is shown in the *Dictionary Dim* column of Table 3. Then the one-hot vectors are used as the inputs of DIPN. As the inputs are high dimensional binary vectors, we use the embedding layer to transform them into low dimensional dense representations. The embedding operation follows the table lookup mechanism. In more details, each feature is corresponding to one embedding matrix. For example, the embedding matrix of the *Button Index* feature in Table 3b is represented as $\mathrm{E}_{button} = [e_1; e_2; ...; e_n] \in \mathbb{R}^{n_e \times n_b}$, where $e_i \in \mathbb{R}^{n_e}$ represents an embedding vector with dimension n_e, and n_b represents the number of buttons that a user can tap. The embedding vector of the *Button Index*

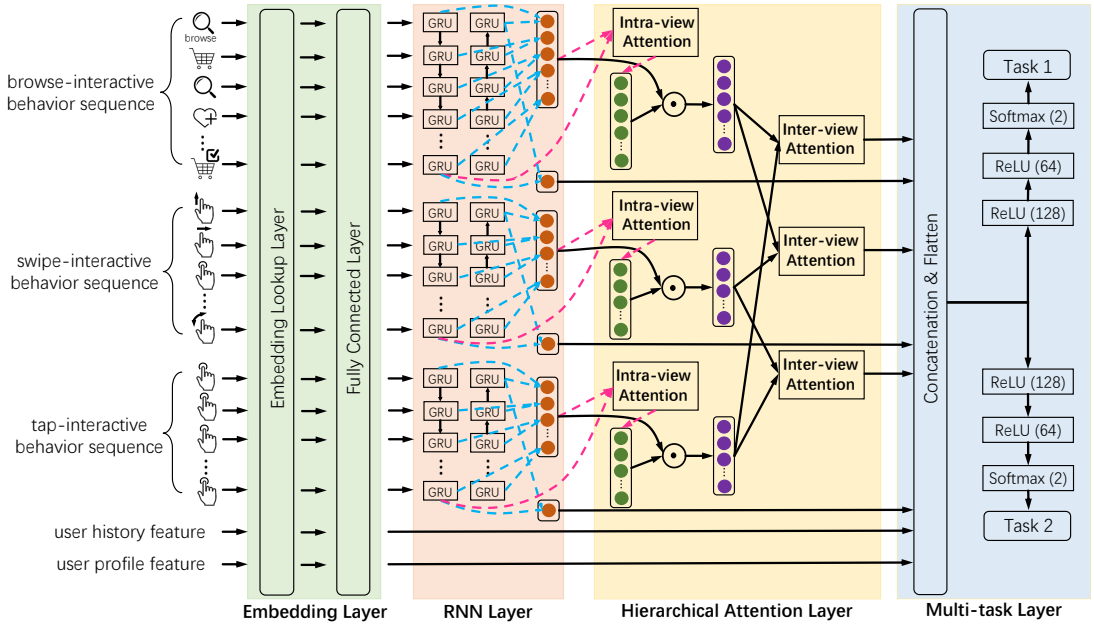

Figure 1: The model architecture of DIPN.

feature can then be obtained as $\mathrm{E}_{button} \cdot \mathrm{B}_{button} \in \mathbb{R}^{n_e}$, where $B_{button} \in \mathbb{R}^{n_b}$ is the one-hot vector of the *Button Index* feature. The length of every embedded feature is shown in the *Embedding Dim* column of Table 3. At last, for each feature group, all the embedded features are concatenated into a vector and fed into a fully-connected layer for reshape. During the training process, the embedding layer is trained at the same time with the model.

4.2 RNN Layer

The user interactive behavior used in DIPN are all time sequence of actions. Therefore, we use RNN to model the long-term dependencies between actions. The adoption of RNN can eliminate the need for extensive feature engineering, which is very helpful because it is difficult to extract features from the touch-interactive behavior composed of swipe or tap actions with little semantic information. To avoid the vanishing gradient problem suffered by the standard RNN, LSTM and GRU are proposed to control the update of the information via gates. We take GRU to model the dependency because GRU is faster than LSTM and more suitable for e-commerce system. The formulations of GRU are listed as follows:

$$\begin{aligned} r_t &= \sigma(\mathbf{W}_{er} e_t + \mathbf{W}_{hr} h_{t-1} + b_r) \\ z_t &= \sigma(\mathbf{W}_{ez} e_t + \mathbf{W}_{hz} h_{t-1} + b_z) \\ \tilde{h}_t &= \tanh(\mathbf{W}_{eh} e_t + \mathbf{W}_{hh}(r_t \odot h_{t-1}) + b_h) \\ h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t. \end{aligned} \tag{1}$$

where e_t is embedding vector of the t-th action, h_t is the t-th hidden states, σ is the sigmoid function and $\odot$ is the element-wise product operator. To better capture the global information of the behavior sequences, we adopt a bidirectional recurrent layer composed of two GRU layers working in opposite directions. We obtain the representation of the t-th action by concatenating the forward hidden state $\overrightarrow{h}_t$ and backward hidden states $\overleftarrow{h}_t$, i.e., $h_t = [\overrightarrow{h}_t, \overleftarrow{h}_t]$. In this way, a behavior sequence is represented as $\mathrm{h} = \{h_1, h_2, ..., h_n\} \in \mathbb{R}^{n \times 2d}$, where d is the dimension of the hidden state .

In DIPN, we need to handle three types of behavior sequences, i.e., the swipe-interactive sequence, the tap-interactive sequence and the browse-interactive sequence. There are two ways to fuse these sequences: early fusion and late fusion. The early fusion refers to aligning the three sequences by timestamp before feeding them into a single GRU model, while the late fusion refers to first feeding each sequence to a separate GRU model and then concatenating the output hidden features. One disadvantage of the early fusion method is that the behavior sequences usually have different densities, as shown in Table 1. When aligning the sequences by timestamp, dense sequence could dominate the concatenated feature space and override the effects of sparse but important sequence. In addition, since the length of GRU model is limited, the early fusion method would result in information loss of the dense sequence when truncating the sessions. Therefore, we propose to use the late fusion method and feed the three behavior sequences to separate Bi-GRU models, as shown in Figure 1. After the RNN layer, we get three hidden outputs, i.e., $\mathrm{h_s} = \{h_{s1}, h_{s2}, ..., h_{sn}\} \in \mathbb{R}^{n \times 2d}$, $\mathrm{h_t} = \{h_{t1}, h_{t2}, ..., h_{tn}\} \in \mathbb{R}^{n \times 2d}$ and $\mathrm{h_b} = \{h_{b1}, h_{b2}, ..., h_{bn}\} \in \mathbb{R}^{n \times 2d}$, corresponding to the swipe-interactive, tag-interactive and browse-interactive sequence, respectively.

4.3 Hierarchical Attention Layer

To better fuse the views extracted from different behavior sequences, we propose a hierarchical attention mechanism, where the bottom attention layer focuses on the inner parts of each behavior sequence while the top attention layer learns the inter-view relations between different behavior sequences, as shown in Fig. 1. In the following, we introduce the hierarchical attention mechanism in details.

Intra-view Attention. The intra-view attention mechanism at the bottom tries to identify the important actions within each sequence that contribute more to the purchasing intent prediction. Intuitively, the current behavior conducted by a user is most predictive of purchasing intent. To capitalize on this, we calculate the attention score between each action in a sequence and the current action, which is formulated as:

$$a_t = \frac{\exp(h_t \mathbf{W_a} \overrightarrow{h}_n)}{\sum_{i=1}^{n} \exp(h_i \mathbf{W_a} \overrightarrow{h}_n)}, \tag{2}$$

where $\overrightarrow{h}_n \in \mathbb{R}^d$ is the final output state of the forward GRU model, $h_t \in \mathbb{R}^{2d}$ is t-th output state of the Bi-GRU model, $\mathbf{W_a} \in \mathbb{R}^{2d \times d}$, and a_t is the attention score calculated for h_t. Attention score can reflect the relationship between h_t and the current action $\overrightarrow{h}_n$. Therefore, the action that is more related to the current action can get a larger attention score.

Different from traditional attention mechanism which conducts a weighted sum pooling operation to calculate the final output, the intra-view attention mechanism applies the element-wise product on the outputs of Bi-GRU and its corresponding attention score vector as follows:

$$\mathsf{v_s} = \mathsf{h_s} \odot \mathsf{a_s},\ \mathsf{v_t} = \mathsf{h_t} \odot \mathsf{a_t},\ \mathsf{v_b} = \mathsf{h_b} \odot \mathsf{a_b}, \tag{3}$$

where $\mathsf{v_s} \in \mathbb{R}^{n \times 2d}$, $\mathsf{v_t} \in \mathbb{R}^{n \times 2d}$ and $\mathsf{v_b} \in \mathbb{R}^{n \times 2d}$ are the outputs of the intra-view attention layer, corresponding to the swipe-interactive sequence, tag-interactive sequence and browse-interactive sequence, respectively.

Inter-view Attention. The swipe-interactive, tap-interactive and browse-interactive behavior depict a user from different views simultaneously. For example, a user interested in a product would browse some comments about the product and other similar products for comparison before she finally places the order. This process would generate some browse, swipe and tap actions. It is important to utilize these relationships to model the purchasing intent of a user. However, the actions from different views are usually asynchronous. The reasons are twofold. First, the synchronous actions become asynchronous due to the different densities of each behavior. Second, the related actions are originally asynchronous, e.g., some actions result in the occurrence of other actions. To this end, we propose a novel inter-view attention mechanism to better discover the asynchronous interactions between different views.

The inter-view attention mechanism takes two views as inputs, as shown in Figure 2. In particular, for each action in one view, we calculate its distance with all the actions in the other view. We borrow this idea from the self attention mechanism in Transformer [21]. In this way, the asynchronous interactions between actions can be captured effectively. The inter-view attention mechanism $\mathrm{IA}(\mathsf{v_s}, \mathsf{v_b})$ is formulated as follows, where we take the swipe-interactive view $\mathsf{v_s}$ and the browse-interactive view $\mathsf{v_b}$ for example:

$$\begin{aligned} \mathrm{IA}(\mathsf{v_s}, \mathsf{v_b}) &= \mathrm{A_s}(\mathsf{v_b}, \mathsf{v_s}, \mathsf{v_s}) \odot \mathrm{A_b}(\mathsf{v_s}, \mathsf{v_b}, \mathsf{v_b}) \\ \mathrm{A_s}(\mathsf{v_b}, \mathsf{v_s}, \mathsf{v_s}) &= \mathrm{softmax}(\frac{\mathsf{v_b}\mathsf{v_s}^T}{\sqrt{2d}})\mathsf{v_s} \\ \mathrm{A_b}(\mathsf{v_s}, \mathsf{v_b}, \mathsf{v_b}) &= \mathrm{softmax}(\frac{\mathsf{v_s}\mathsf{v_b}^T}{\sqrt{2d}})\mathsf{v_b}, \end{aligned} \tag{4}$$

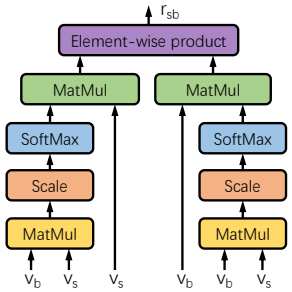

Figure 2: Inter-view attention mechanism.

where $\mathrm{A_s} \in \mathbb{R}^{n \times 2d}$ and $\mathrm{A_b} \in \mathbb{R}^{n \times 2d}$ are attentive representations of $\mathsf{v_s}$ and $\mathsf{v_b}$, respectively, and the element-wise product $\odot$ is used to model the interactions between $\mathrm{A_s}$ and $\mathrm{A_b}$. Note that $\mathrm{IA}(\mathsf{v_s}, \mathsf{v_b})$ is a symmetric operation and returns the interaction representation $\mathsf{r_{sb}}$ between $\mathsf{v_s}$ and $\mathsf{v_b}$. We can calculate $\mathsf{r_{st}}$ and $\mathsf{r_{tb}}$ following the same procedure. At last, we can get three representations $\mathsf{r_{sb}} \in \mathbb{R}^{n \times 2d}$, $\mathsf{r_{st}} \in \mathbb{R}^{n \times 2d}$ and $\mathsf{r_{tb}} \in \mathbb{R}^{n \times 2d}$ after the inter-view attention layer.

4.4 Multi-task Layer

In this work, we propose to train DIPN with multi-task learning. The reasons are twofold. First, with multi-task learning, DIPN can capture common features that well depict the customers and unique features that would lead to different purchasing behavior. Second, a model that learns multiple tasks simultaneously is able to learn a more robust representation and improve the generalization.

As shown in Fig. 1, we use two tasks in this layer, i.e., the real-time purchasing intent prediction and the long-term purchasing intent prediction, which are defined as a predictive measure of subsequent purchasing behavior within one hour and one day, respectively. The reason that we define the period of the long-term purchasing intent as one day is because the purchasing behavior conducted one day later has a relative low correlation with the current behavior sequence. As a result, we partition user purchasing intent into three phases: real-time phase, long-term phase and irrelevant phase. Note that we use multi-task learning to handle the multi-class learning problem. By training the real-time intent and long-term intent with two separate tasks, we are able to distinguish between the subtle differences of user behavior.

The outputs of the hierarchical attention layer are first flattened and then concatenated with the user history feature, user profile feature and the concatenated last forward and backward state of the Bi-GRU models. The concatenated feature vectors are fed into two different branches. In each branch, fully connected layers are used to learn the combination of features automatically. The loss functions of the real-time purchasing intent prediction task and long-term purchasing intent prediction task are defined as follows:

$$\begin{aligned} \mathcal{L}_{short} &= -\frac{1}{N} \sum_{(x,y) \in \mathcal{D}}^{N} (y \log p_s(x) + (1-y) \log(1 - p_s(x))) \\ \mathcal{L}_{long} &= -\frac{1}{N} \sum_{(x,y) \in \mathcal{D}}^{N} (y \log p_l(x) + (1-y) \log(1 - p_l(x))) \end{aligned} \tag{5}$$

where $\mathcal{D}$ is the training set with size N, x is the input of the network and y is the label, $p_s(x)$ and $p_l(x)$ represents the predicted

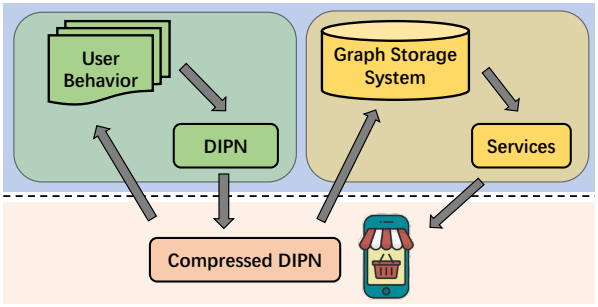

Figure 3: System overview under edge computing.

probability of sample x being purchased within one hour and one day, respectively. The global loss used in our DIPN model is:

$$\mathcal{L}_{global} = \mathcal{L}_{short} + \mathcal{L}_{long}. \tag{6}$$

5 SYSTEM OVERVIEW

Benefit from the rapid development of hardware and software on mobile devices, we are able to collect real-time behavior features to improve the performance of purchasing intent prediction. However, this benefit comes with a price that the high frequency of occurrence of the real-time features hinders DIPN from deploying in the industrial environment. If DIPN is deployed at the server side following traditional cloud-based computing architecture, the high frequency of features when used for prediction would result in a high communication cost unbearable to both the servers and the smartphones. To solve this problem, we propose to deploy our DIPN model on the mobile devices following the idea of edge computing [1], which is defined as a distributed computing paradigm in which computation is largely or completely performed on distributed device nodes known as smart devices or edge devices.

The overall structure of our prediction system deployed in a large-scale e-commerce platform, namely Taobao, is illustrated in Figure 3. The procedure is as follows. We first train our DIPN model in the cloud on powerful servers, and then use AliNN, which is Alibaba's solution for deploying machine learning models on mobile devices, to compress DIPN (with a size of 2MB) and deploy the compressed model on the devices. After that, the compressed DIPN can directly use the collected real-time features on a mobile device for prediction. Only the prediction results are sent to the cloud, which are stored in an online graph storage system and can be used to provide services for customers later. If we need to update DIPN , we only need to collect the training data from the devices and re-deploy the updated model to the devices.

There are several advantages to deploy DIPN on the mobile devices. First, it can reduce the communication cost between the cloud and the devices significantly. During the Alibaba 2018 Double 11 Shopping Festival, DIPN serves more than 10 million customers without suffering from the traditional peak traffic problem of the platform on that day. Second, it can greatly increase the response speed of DIPN by moving the model to the data instead of the data to the model. The response time of DIPN is between 20 $\sim$ 50 ms on different devices, which is immune to the influence of the network traffic and improves application performance. Third, it can well protect the user privacy, because only the prediction scores, rather than the features capturing behavior patterns, are sent to the cloud.

[1] https://en.wikipedia.org/wiki/Edge_computing

6 EXPERIMENTS

In this section, we present a comprehensive evaluation of the performance for DIPN. We first introduce the experimental setup and then present the experimental results under various settings. Finally, we share a case study for online serving.

6.1 Experimental Setup

Dataset Statistics. We conduct the experiments on a large-scale industrial dataset collected from Taobao. The dataset contains normal users' daily interaction information when using our app, which consists of four subsets including the swipe-interactive behavior, tap-interactive behavior, browse-interactive behavior and also the user profiles. We collect 800,000 users' behavior for two weeks. For each user, we randomly truncate about 400 groups of samples on average. In total, we obtain 300 million groups of samples. Each group contains the user profile feature, the user history feature, and the three sequences with length 256 (padding 0 for short ones). Note that within each group, the timestamps of the last actions in the three sequences are the same. The real-time label and long-term label are then tagged for each group based on the timestamp of the last action. We use samples in the first 13 days for training and samples in the last day for evaluation.

Compared Methods. We compare DIPN to the state-of-the-art approaches in purchasing intent prediction. Besides, we conduct experiments to verify the effect of each component in DIPN. In the following, we introduce the compared methods briefly.

- **GBDT** [15]: A competitive gradient boosting model widely used in industrial environment. For a fair comparison, besides the session features used in [15], we also add the user history feature and user profile feature to the input of the model. Our goal is to see the benefit of using the new touch-interactive behavior in predicting user purchasing intent.
- **RNN+DNN** [19]: A bidirectional RNN is used to model the dependency between the browse-interactive actions. Similar with **GBDT**, we modify RNN by adding the user history feature and user profile feature with DNN.
- **DIPN-early-fusion**: Early fusion is applied in DIPN by first aligning the three sequences by timestamp and then feeding them into a single Bi-GRU layer. As a result, only intra-view attention mechanism is adopted.
- **DIPN-no-attention**: A sub model of DIPN without using the hierarchical attention mechanism. The outputs from the RNN layer are simply concatenated.
- **DIPN-no-inter-view-attention**: A sub model of DIPN by removing the inter-view attention mechanism.
- **DIPN-no-intra-view-attention** A sub model of DIPN by removing the intra-view attention mechanism.
- **DIPN-no-multi-task** A sub model of DIPN without using multi-task learning.

Experimental Details. DIPN is trained with SGD, using the Adam optimizer [12] with initial hyper-parameters of $\epsilon = 10^{-3}$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The dimension of the hidden state in the Bi-GRU model is set to $d = 32$. We train DIPN using a distributed

Table 4: Comparison of different models.

Model	AUC
GBDT	0.7871
RNN+DNN	0.7902
DIPN-early-fusion	0.7708
DIPN-no-attention	0.8345
DIPN-no-inter-view-attention	0.8367
DIPN-no-intra-view-attention	0.8401
DIPN-no-multi-task	0.8371
DIPN	**0.8429**

Table 5: Impact of multi-task learning.

	AUC(real-time)	AUC(long-term)
DIPN-no-multi-task	0.8371	0.8204
DIPN	**0.8429**	**0.8276**

Table 6: Impact of different sources.

Task1	AUC	Task2	AUC
DIPN w/o profile.	0.8381	DIPN w/ profile.	0.5419
DIPN w/o history.	**0.7862**	DIPN w/ history.	**0.7335**
DIPN w/o browse.	0.8303	DIPN w/ browse.	0.6533
DIPN w/o swipe.	0.8287	DIPN w/ swipe.	0.6742
DIPN w/o tap.	**0.7978**	DIPN w/ tap.	**0.7418**

TensorFlow with 1 parameter server and 100 workers. The metric used in our experiments is Area Under the Curve (AUC), which is insensitive to class imbalance and suitable to our experiments.

6.2 Experimental Results

Results of different models. Table 4 shows the performance of the evaluated models. We have the following observations. (1) DIPN outperforms the baseline methods GBDT and RNN by a significant margin about 5.6% and 5.3% in terms of AUC, respectively. The improvement of DIPN over GBDT and RNN reveals the value of adopting the touch-interactive behavior to depict users from different views. (2) The early fusion manner is not appropriate for fusing views from different data sources. We can see that DIPN-early-fusion performs worst among the compared models. The reason is that the early fusion method could result in the imbalance of different views and information loss. (3) The hierarchical attention mechanism plays an important role in DIPN. As shown, DIPN-no-inter-view-attention and DIPN-no-intra-view-attention are superior to DIPN-no-attention but inferior to DIPN. This proves that the intra-view attention and inter-view attention mechanism are effective in identifying important actions within a view and discovering useful asynchronous interactions between views, respectively. (4) Prediction performance can be further improved by utilizing multi-task learning. Table 5 shows the results of DIPN with or without multi-task learning. As shown, the performance of real-time and long-term purchasing intent prediction can be improved by 0.6% and 0.7% when using multi-task learning, respectively.

Impact of different sources. In this paper, DIPN predicts the real-time purchasing intent by utilizing multiple data sources simultaneously. To better understand the role that different data sources play, we conduct two types of tasks using DIPN. The first task is to predict the purchasing intent without each data source, while the second task only uses one single data source for prediction. The results are shown in Table 6. We can see that each data source gives a positive impact on the performance of DIPN. The user profile feature performs worst in the second task, which is as expected because it only provides basic information about a user. However, it increases AUC by about 0.5% when used together with other data sources, because it improves personalization in DIPN. The tap-interactive behavior plays a more significant role than the other behavior. The reason is that it captures more real-time behavior patterns compared with the browse-interactive behavior and contains more rich semantic information compared with the swipe-interactive behavior. It should be noted that the user history feature also contributes a lot to the performance of DIPN, demonstrating that the activeness of users has a great impact on their purchasing behavior. As shown, by utilizing all the data sources listed in our paper, DIPN gains about 18.96% improvement on AUC than the baseline only using traditional user behavior sequences.

6.3 Online A/B Testing

Coupon allocation is an important strategy for improving the Gross Merchandise Volume (GMV) on e-commerce platforms. In this section, we introduce a new coupon allocation strategy based on the real-time purchasing intent predicted by DIPN in online traffic of Taobao. The online A/B testing was conducted at "Double 11" in 2018, which is a shopping festival in China, similar as the "Black Friday" in America.

We choose a coupon with 10 RMB nominal value for our testing and set three coupon allocation strategies to compare the performance, defined as follows:

- **All-allocation Strategy** where everyone in this bucket is selected to get this coupon.
- **Non-allocation Strategy** where everyone in this bucket is not selected to get this coupon.
- **Model-allocation Strategy** which uses the score predicted by DIPN and the fixed thresholds to decide the allocation.

The users selected to get this coupon will be pushed a popup in Taobao's mobile application.

We use the usage rate of coupons R_c, and the GMV improvement per coupon I_{gmv} as evaluation metrics, defined as follows:

$$R_c = \frac{N_{wb}}{N_b}, \tag{7}$$

where N_{wb} is the number of users who have used this coupon to buy something in a bucket b, and N_b is the total number of users who have got this coupon in b.

$$I_{gmv} = \frac{(G_b - \frac{N_b}{N_{non}} G_{non})}{N_{wb}} = \frac{N_{non} G_b - N_b G_{non}}{N_{non} N_{wb}}, \tag{8}$$

where G_b is the total GMV of users in a bucket b, N_b is the number of users in b, G_{non} and N_{non} are the total GMV of users and the number of users in the non-allocation strategy bucket, respectively.

Table 7: The results of different coupon allocation strategies.

	Num. of Users	R_c	I_{gmv}
None-allocation	$1.38M$	/	0
All-allocation	$10.35M$	40.4%	I^a
Model-allocation	$1.22M$	57.0%(**+41.1%**)	I^m(**+39.8%**)

We hypothesize that the users with a very low purchasing intent are hard to change their mind because of this coupon, while the users with a high purchasing intent are not needed to be given this promotion. Therefore, we set the lowest threshold $t_l = 0.2$ and the uppermost threshold $t_u = 0.4$. The users whose real-time purchasing intent score given by DIPN is between t_l and t_u are selected to get this coupon in the model-allocation strategy bucket.

As shown in Table 7 [2], there are more than 12.95 millions of users in this online A/B testing. It is notable that the model-allocation strategy contributes up to **41.1%** R_c and **39.8%** I_{gmv} promotion compared with the all-allocation strategy in this large scale online traffic. The reason is that DIPN can help allocation system to understand user's real-time purchasing intent and allocate the coupon to the right person at the right time. Compared with the all-allocation strategy, a reasonable allocation strategy relied on DIPN would result in a significant GMV improvement.

7 CONCLUSION

In this paper, we propose DIPN, a novel attention-based deep network with multi-task learning, for real-time purchasing intent prediction. Different from previous work, we collect a new type of user interactive behavior, i.e., the touch-interactive behavior, to capture comprehensive user behavior patterns. In order to fuse multiple user interactive behavior effectively, we propose a hierarchical attention mechanism including intra-view attention and inter-view attention. In addition, we use multi-task learning to train DIPN to better distinguish user behavior patterns. We conduct extensive experiments on a large-scale industrial dataset to evaluate the performance of DIPN. Experimental results show the superiority of DIPN under various settings. In particular, online A/B testing results reveal the potential of knowing users' real-time purchasing intent, which would result in a significant GMV improvement in the e-commerce platforms.

8 ACKNOWLEDGMENTS

This work is supported in part by the National Natural Science Foundation of China under Grant No. 61702016, 61832001, 61572039, the National Key Research and Development Program of China (No. 2018YFB1004403).

REFERENCES

[1] Charu C. Aggarwal. 2002. On Effective Classification of Strings with Wavelets. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 163–172.
[2] David Ben-Shimon, Alexander Tsikinovsky, Michael Friedmann, Bracha Shapira, Lior Rokach, and Johannes Hoerle. 2015. RecSys Challenge 2015 and the YOOCHOOSE Dataset. In *Proceedings of the 9th ACM Conference on Recommender Systems*. 357–358.
[3] Yangbin Chen, Yun Ma, Xudong Mao, and Qing Li. 2019. Multi-Task Learning for Abstractive and Extractive Summarization. *Data Science and Engineering* 4, 1 (01 Mar 2019), 14–23.
[4] Betty Yee Man Cheng, Jaime G. Carbonell, and Judith Klein-Seetharaman. 2005. Protein classification based on text document classification techniques. *Proteins: Structure, Function, and Bioinformatics* 58, 4 (2005), 955–970.
[5] Ronan Collobert and Jason Weston. 2008. A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning. In *Proceedings of the 25th International Conference on Machine Learning*. 160–167.
[6] L. Deng, G. Hinton, and B. Kingsbury. 2013. New types of deep neural network learning for speech recognition and related applications: an overview. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*.
[7] Long Duong, Trevor Cohn, Steven Bird, and Paul Cook. 2015. Low Resource Dependency Parsing: Cross-lingual Parameter Sharing in a Neural Network Parser. In *ACL-IJCNLP*. 845–850.
[8] Ross Girshick. 2015. Fast R-CNN. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. 1440–1448.
[9] Long Guo, Dongxiang Zhang, Lei Wang, Han Wang, and Bin Cui. 2018. CRAN: A Hybrid CNN-RNN Attention-Based Model for Text Classification. In *37th International Conferencel on Conceptual Modeling*. 571–585.
[10] Eamonn J. Keogh and Michael J. Pazzani. 2000. Scaling Up Dynamic Time Warping for Datamining Applications. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 285–289.
[11] Yoon Kim. 2014. Convolutional Neural Networks for Sentence Classification. *CoRR* abs/1408.5882 (2014).
[12] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR* abs/1412.6980 (2014).
[13] Neal Lesh, Mohammed J. Zaki, and Mitsunori Ogihara. 1999. Mining Features for Sequence Classification. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 342–346.
[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. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 596–605.
[15] Peter Romov and Evgeny Sokolov. 2015. RecSys Challenge 2015: Ensemble Learning with Categorical Features. In *RecSys '15 Challenge*. Article 1, 4 pages.
[16] Sebastian Ruder. 2017. An Overview of Multi-Task Learning in Deep Neural Networks. *CoRR* abs/1706.05098 (2017).
[17] Rong She, Fei Chen, Ke Wang, Martin Ester, Jennifer L. Gardy, and Fiona S. L. Brinkman. 2003. Frequent-subsequence-based Prediction of Outer Membrane Proteins. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 436–445.
[18] Humphrey Sheil and Omer Rana. 2018. Classifying and Recommending Using Gradient Boosted Machines and Vector Space Models. In *Advances in Computational Intelligence Systems*. 214–221.
[19] Humphrey Sheil, Omer Rana, and Ronan Reilly. 2018. Predicting purchasing intent: Automatic Feature Learning using Recurrent Neural Networks. *CoRR* abs/1807.08207 (2018).
[20] Arthur Toth, Louis Tan, Giuseppe Di Fabbrizio, and Ankur Datta. 2017. Predicting Shopping Behavior with Mixture of RNNs. In *ACM SIGIR Forum*.
[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems 30*. 5998–6008.
[22] Armando Vieira. 2015. Predicting online user behaviour using deep learning algorithms. *CoRR* abs/1511.06247 (2015).
[23] Maksims Volkovs. 2015. Two-Stage Approach to Item Recommendation from User Sessions. In *RecSys '15 Challenge*. Article 3, 4 pages.
[24] Li Wei and Eamonn Keogh. 2006. Semi-supervised Time Series Classification. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 748–753.
[25] 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 *RecSys '15 Challenge*. Article 12, 4 pages.
[26] Zhengzheng Xing, Jian Pei, and Eamonn Keogh. 2010. A Brief Survey on Sequence Classification. *SIGKDD Explor. Newsl.* 12, 1 (Nov. 2010), 40–48.
[27] Oksana Yakhnenko, Adrian Silvescu, and Vasant Honavar. 2005. Discriminatively Trained Markov Model for Sequence Classification. In *Proceedings of the Fifth IEEE International Conference on Data Mining*. 498–505.
[28] Yongxin Yang and Timothy M. Hospedales. 2016. Trace Norm Regularised Deep Multi-Task Learning. *CoRR* abs/1606.04038 (2016).
[29] Lexiang Ye and Eamonn Keogh. 2009. Time Series Shapelets: A New Primitive for Data Mining. In *KDD*. 947–956.
[30] Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level Convolutional Networks for Text Classification. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. 649–657.
[31] Yi Zhao, Yanyan Shen, and Yong Huang. 2019. DMDP: A Dynamic Multi-source Default Probability Prediction Framework. *Data Science and Engineering* 4, 1 (01 Mar 2019), 3–13.

[2] As the sensitive data policy, the I_{gmv} of the all-allocation strategy and the model-allocation strategy have been replace as I^a and I^m.