# Keep You from Leaving: Churn Prediction in Online Games

Angyu Zheng[1,2], Liang Chen*[1,2], Fenfang Xie[1,2], Jianrong Tao[3], Changjie Fan[3], and Zibin Zheng[1,2]

[1] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
[2] National Engineering Research Center of Digital Life, Sun Yat-sen University, Guangzhou, China
[3] NetEase Fuxi AI Lab, Hangzhou, China
{zhengangy,xieff5}@mail2.sysu.edu.cn,{chenliang6,zhzibin}@mail.sysu.edu.cn,{hztaojianrong,fanchangjie}@corp.netease.com

**Abstract.** Customer retention is a crucial problem for game companies since the revenue is heavily influenced by the size of their user bases. Previous studies have reached a consensus that the cost of attracting a new player can be six times than retaining the players, which indicates an accurate churn prediction model is essential and critical for the strategy making of customer retention. Existing works more focus on studying login information(e.g. login activity traits of users) ignoring the rich in-game behaviors(e.g. upgrading, trading supplies) which could implicitly reflect user's preference from their inter-dependencies. In this paper, we propose a novel end-to-end neural network, named *ChurnPred*, for churn prediction problem. In particular, we not only consider the login behaviors but also in-game behaviors to model user behavior patterns more comprehensively. For time series of login activities, we leverage a LSTM-based structure to learn intrinsic temporal dependencies so as to capture the evolution of activity sequences. For in-game behaviors, we develop a time-aware filtering component to better distinguish the behavior patterns occurring in a specific period and a multi-view mechanism to automatically extract the multiple combinations of these behaviors from various perspectives. Comprehensive experiments conducted on real-world data demonstrate the effectiveness of the proposed model compared with state-of-the-art methods.

**Keywords:** Churn prediction · Online games · neural network · in-game behaviors · login activities.

## 1 Introduction

The huge revenue generated by online games including massively multiplayer online role-playing games (MMORPGs) has attracted many game companies, which results in increasingly intense competition in the game market. Customer retention is becoming a major concern, since: 1) *the cost of attracting a new player can be six times than retaining the players* [23]; 2) long-term players

usually generate higher profits than the new ones. As an important part of the user retention, it is crucial to know early on whether players will choose to stay or leave the game in the early stage, which is also called as churn prediction problem.

An accurate churn prediction model is essential and critical for the strategy making of the customer retention. Once the churners are identified by the prediction model, game managers can take some measures to prevent those from leaving the game such as providing some reward tasks to stimulate the user's interest or pushing notifications with fresh play strategies that the user interests. Moreover, the prediction results of the churn prediction model can provide the game platform with a reference to understand the overall preferences of the game players and accordingly make appropriate strategies. An increasing number of churners may become a strong signal for game operators to adjust game strategy in advance.

Previous research for the churn prediction problem in online games (e.g. MMORPGs) focuse on mining salient features to indicate whether a user is about to leave the game. They prefer to exploit handcraft features after a comprehensive analysis on multiple characteristics and complete the churn prediction task by using traditional machine-learning-based methods [3, 4, 7, 8, 10]. The limitations of previous investigations are mainly two-fold: 1) Heavily depending on domain-specific knowledge and artificial features, which is not widespread to different application scenarios. For example, some features are not universal or difficult to collect for online games such as "click count" and "rests used" in [10], "sum of inter-session length per week" in [3], "social activity" and "item upgrade" in [8], "rate of group interactions" in [2]. 2) Mainly utilizing features derived from the statistics of login information, while ignoring users' behavior information in the game. These information is important for the churn prediction since it could further indicate the users' preference for the game. Players put many efforts into perfecting their roles, such as constantly performing tasks to upgrade or trading supplies to enhance their equipment, which demonstrates a kind of preference for their characters of the game. Tao et al. have proven the importance of users' in-game behavioral information in bot detection [16].

There exist several challenges of churn prediction in online games. As mentioned above, users' behaviors in online games are mainly classified in two aspects: login information (e.g. session statistic, login frequency) and in-game behavioral information (e.g. a series of in-game behaviors such as upgrading, trading supplies). First, these data are in different types since the former is often expressed as real-value vectors, while each element of the latter data is a discrete value representing a specific action. It is challenging to model these data together to capture user-game interactions and inherent behavior patterns. Second, since each player has her/his own lifetime, short- and long-term modeling is required for capturing the evolution of users' preferences and temporal patterns better. Third, users' behaviors are closely related to their daily life (details will be given in Section 3). For example, the length of users' engagement with the games on weekdays is different from that on weekends, or some events such as trading

specific items or fighting battles can only take place on some special days (e.g., festivals). Therefore, it is essential to additionally consider the influence of these information when modeling.

To alleviate the above mentioned challenges, in this paper, we propose a novel end-to-end neural network approach, named *ChurnPred*, for churn prediction in online games. We consider login information and in-game information together to model user behaviors more comprehensively, from which potential behavior patterns are automatically learned without manually extracting features. Considering the impact of lifetime of users on login behaviors, we leverage LSTM models to learn the short- and long-term users' preferences. As we find in Section 3 that some behaviors are closely related to the day of occurrence, we propose a time-aware filtering component to better distinguish these characteristic behaviors based on the period of events. Besides, we propose a multi-view mechanism to automatically extract the multiple combinations of in-game behaviors from various perspectives which would lead to the departure of users.

To summarize, main contributions of this paper are listed as follows:

- We develop a novel end-to-end neural network approach, named *ChurnPred*, via considering login behaviors and in-game behaviors for churn prediction in online games. Additionally, we propose time-aware filtering mechanism to better distinguish the behavior patterns occurring in that period and a multi-view mechanism to extract the multiple combinations of in-game behaviors from various perspectives which would imply the departure of users.
- We conduct comprehensive experiments on a real-world dataset of three different periods to verify the effectiveness of the proposed model. Experimental results shows the superiority of our model for churn prediction in online games.

## 2 Related Work

Customer churn behaviors have been consistently analyzed across a wide range of industries, since most companies are convinced the number and stickiness of users play an importance role on their competitiveness and vitality in the market. Most of these works [10,14,23] focus more on extracting outstanding features and exploring the classification performance among the traditional classifiers such as logistic regression [11], random forests [21]. They model churn prediction as a binary classification problem and tend to summarize the difference of the samples by using statistical techniques for better identification. These studies depend much on domain-specific knowledge and artificial features, which is not universal to different application scenarios. Recently, some studies have suggested more advanced models on the churn prediction. Some works [13, 18] propose survival analysis model by modeling the playtime of players. Since deep learning has achieve great success in various domain such as detection [19] and recommender [5, 20], some researches [1, 9, 17] focus on leveraging the deep neural network for churn prediction problems, which motivates us to employ deep neural network models.

Only a few papers are directly related to the online games including MMOPRGs. Borbora et al. design a lifecycle-based approach for modeling churn behavior and propose three dimensions to construct derived features for a distance-based schema *wClusterDist* for better classification [3]. However, the lifecycle-based approach ignores the time of users' registration and time consumption on the game. Those loyal customers who have been playing for a long time tend to be less active in observation and thus may be easily mistaken as churners. Runge et al. focus on predicting churn for high-value players of casual social games since they find that the top 7% of paying players contribute around 50% of the total revenue and acquire a series of features for classification [14]. Castro et al. propose a frequency analysis approach for feature representation from login records. The approach converts the login records into a fixed-length arrays as the inputs and use probabilistic classifiers with the k-nearest neighbors algorithm for classification [4]. The above investigations mainly focus on login information(i.e. login frequency), but do not consider users' in-game behavioral information of online games (e.g.MMORPGs). In-game behaviors are crucial for user behavioral modeling, since it contains rich information about the user including the players' specific events in the game and the chronological order of these events. These data will contribute a lot to fully reflect or accurately capture the tendency of players to leave the game.

## 3 Dataset Description

In this section, we give some detailed information of the real-world dataset from a MMOPRG released by the NetEase Games[4]. This dataset is collected from a server including user logs from 22 June, 2018 to 20 September, 2018 and over 880,000 users with hundreds of millions of behavioral sequences. In this dataset, 485 regular events are defined based on the game content and user logs has been automatically established for each player to record the events as well as the times when the player trigger them.

In this dataset, we define two classes of users: churners and non-churners. Usually, churners represent the users who leave the game permanently. To be less ambiguous, we define churners as those who are consistently inactive for over 7 days [12, 23]. Let *leave_day* denote the day that user leave the game. To compare two types of users at the same stage, we mainly focus on the users who have left the game during a specific period and those who haven't. We define the period as a churn window which is denoted as $[observed\_day_1, observed\_day_2]$. The users whose *leave_day* fall in this window will be considered as "churners", and those whose *leave_day* are after the $observed\_day_2$ are defined as "non-churners". In the following section, we adopt this setting for both "churners" and "non-churners".

[4] NetEase Games is the one of China's largest MMORPG developer companies, which has published dozens of popular games including Ghost II, Tianxia 3 and Fantasy Westward Journey Online.

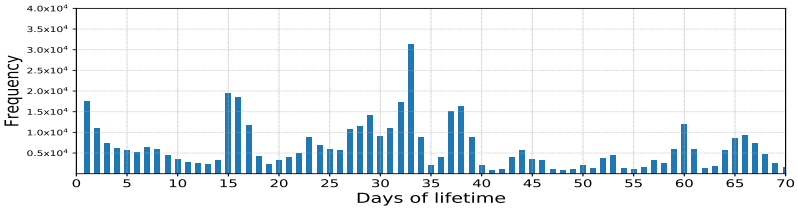


Fig. 1: The distribution of the lifetime of users in a real-world dataset.

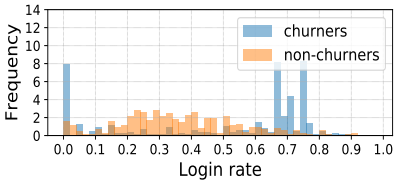


Fig. 2: The login rate of churners and non-churners

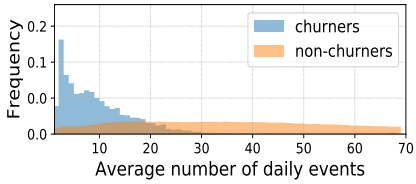


Fig. 3: The average number of daily events of churners and non-churners

In order to find out how long the user has been playing the game before they leave, we examine the distribution of the lifetime of all users in this dataset shown in Fig 1. We observe that the number of churners fluctuates periodically. This indicates that users will leave the game with relatively high probability after they have played for a certain number of days (15/16/32/33). The predictive model needs to capture this characteristic by considering users' lifetime when predicting the probability of the user departure.

To better understand what motivates users continue to play the game, we examine the difference of login information and in-game behavioral information between churners and non-churners. We observe several striking features.

Fig 2 shows the percentage of login days to the whole lifetime between churners and non-churners. We can see that the non-churners are concentrated in the range of 0.2 to 0.5 while churners have high distribution at both ends. Some churners have a low login rate due to various reasons such as lack of interest in the game. But interestingly, the figure also reveals that users with high login rates have a higher probability of leaving. This phenomenon that users log into the game frequently before they leave is helpful for game operators to take some churn preventive measures, such as pushing notification. In Fig 3, the average daily events of churners is relatively small and the distribution of non-churners performs stable. Intuitively, the number of events reflects the duration of playing. Churners always have a fewer events per days because they have fewer times to stay engaging with the game due to low motivation while non-churners are more willing to spend their time to play the game and thus have more events.

We then investigate the in-game behaviors of users. The frequency of each events for churners and non-churners is plotted in the Fig 4. We can see that there are some differences on the event frequency between these two types of users. Some events occur more often in certain types of users or with a relatively

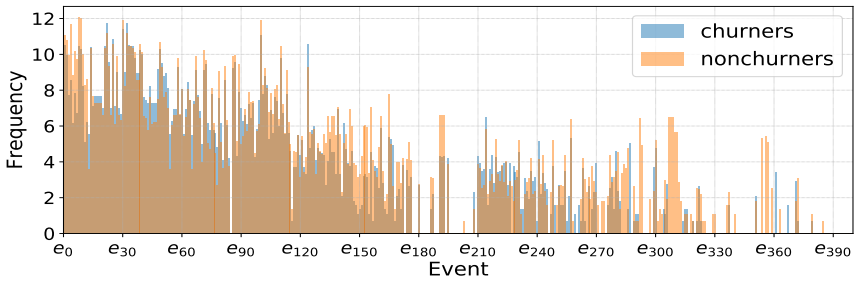


Fig. 4: The occurrence frequency of each event of churners and non-churners

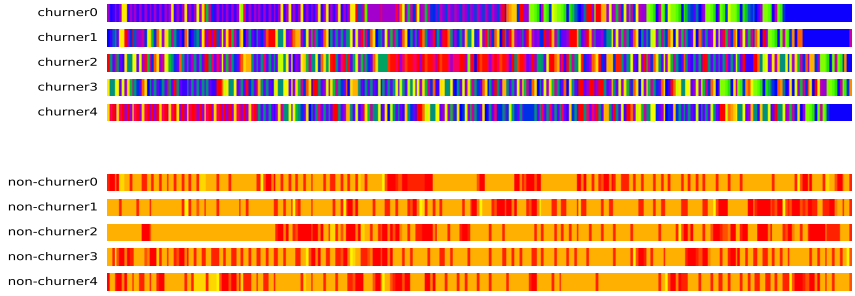


Fig. 5: The behavioral sequences of churners and non-churners. Each color block represents the user's behavior in the game.

high frequency. For example, $e_{150}$ has a higher frequency for non-churners and $e_{310}$ occurs commonly in non-churners but rarely in churners. For the sequence of events, we randomly sample 5 churners and 5 non-churners in churn windows, and extract the sequences of their last 200 events prior to the *leave_day* and the $oberseved\_day_1$, respectively. The result is shown in Fig 5. During this period, the behaviors of non-churners are usually diverse and each is of short duration while the behavior of churners is monotonous and each last for a long time. It can be clearly seen that there are significant differences in the behaviors between churners and non-churners, which further indicates that short-term behaviors reflect whether the user stays in the game or not. Some studies focus on long-term behavior modeling, which not only faces lengthy behavior information, but also increases the complexity of the model and training time.

## 4 Model Architecture

In this section, we present the details of the proposed model. The architecture is illustrated in Fig 6. The model can be divided into three main components: 1) an in-game behavior encoder that models the in-game behavioral information of each user as a context embedding vector. 2) a login behavior encoder that models

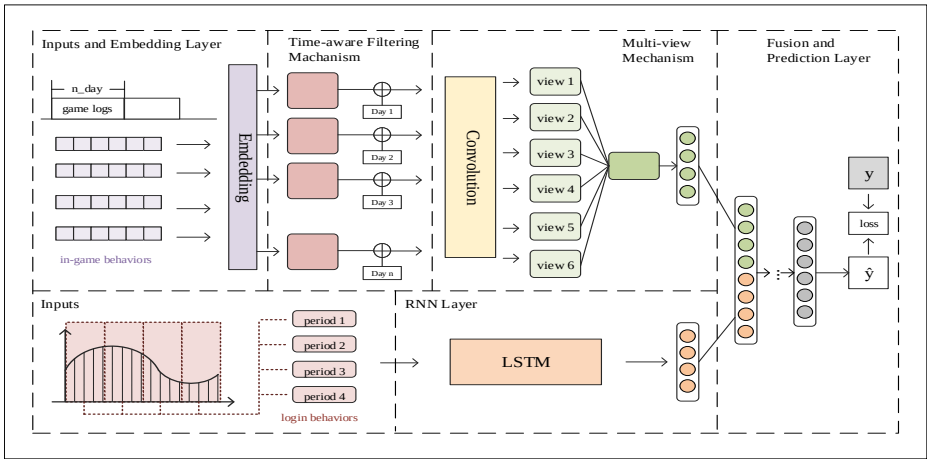


Fig. 6: The architecture of ChurnPred.

the login information of each user in online game as a context embedding vector. 3) the fusion and prediction layer that aggregates above two kinds of embedding vectors and outputs the final possibility of whether the user leaves the game.

### 4.1 In-game Behavior Encoder

**Inputs.** In terms of in-game information, we collect the daily events $e_{ut}^{(d)} \in E$ for user $u$ and arrange them in chronological order which denoted as in-game behavioral sequences $S_u^{(d)} = \{e_{u1}^{(d)}, ..., e_{ut}^{(d)}, ..., e_{uB}^{(d)}\}$ where $E$ is the set of events, $d$ represents the day of events, $B$ denotes the length of $S_u^{(d)}$. As we discussed above, users' total historical behaviors are lengthy and massive which may greatly increase the complexity of the model and training time. In this paper, we use the data of $T_1$ days before the day $observed\_day_1$ as our input which is denoted as $S_u = \{S_u^{(1)}, S_u^{(2)}, .., S_u^{(T_1)}\}$.

**Embedding Layer.** Given $S_u^{(d)} = \{e_{u1}^{(d)}, ..., e_{ut}^{(d)}, ..., e_{uB}^{(d)}\}$, events are embedded into content vectors in a latent space through an embedding layer. In the discretization process, each event identity $e_{ut}^{(d)} \in E$ are encoded into an one-hot vector $o_{ut}^{(d)}$ with $|E|$-dimension. As the inputs are high-dimensional binary vectors, we use the embedding layer to transform them into dense representations. The event embedding vector $x_{ut}^{(d)}$ can be obtained as follows:

$$x_{ut}^{(d)} = W_e^T o_{ut}^{(d)}, \quad t \in \{1, 2, ..., B\} \tag{1}$$

where $W_e \in \mathbb{R}^{|E| \times L}$ denotes the latent factor matrix (embedding matrix) and $L$ is the predefined value used to set the dimension of latent vectors.

**Time-aware Filtering Mechanism.** In the analysis above, we find that user's behavior is closely related to the day it occurs. Hence, we propose a time-aware gating mechanism to capture these characteristic behaviors based

on the time period. Dauphin et al. in [6] propose the gated linear unit (GLU) where they use this gating mechanism for language modeling to allow the model selects related words or features for the next word. Inspired by their work, we make some changes based on this structure where we additionally consider the effect of period on the inputs. We introduce a time matrix $W^{(d)}$ for each day in order to select what features will be propagated to the downstream layers. The formula is shown as follows:

$$D_u^{(d)} = X_u^{(d)} \odot \sigma(X_u^{(d)} * W^{(d)} + b^{(d)}), d \in \{1, 2, ..., T_1\} \tag{2}$$

where $X_u^{(d)} = \{x_{ut}^{(1)}, x_{ut}^{(2)}, .., x_{ut}^{(B)}\} \in \mathbb{R}^{B \times L}$, $\sigma$ is a sigmoid function, $\odot$ is Hadamard (element-wise) product and $W^{(d)}, b^{(d)}$ are parameters to be learned.

**Multi-view Mechanism.** With the success of convolution filters of Convolutional Neural Networks (CNN) in capturing local features for image recognition, we adopt CNN units for multi-view generation where we regard $V_u^{(d)}$ as an "image" of behavioral information and the sequential patterns as local features of this "image". Convolutional filters represented as $kh \times kw$ matrices, which they slide over the "images" and then summary the multiple combinations of behaviors in various views. We use $B$ filters to encode the in-game behaviors of each day respectively. Each filters $F^{(d)} \in \mathbb{R}^{kh \times kw}$ slide over $D_u^{(d)}$ as follows:

$$v_u^{lk(d)} = D_{u,\{l:l+kh-1,k:k+kw-1\}}^{(d)} \oplus F^{(d)} \tag{3}$$

$$V_u^{(d)} = \{v_u^{lk(d)} | 1 \leq l \leq B - kh + 1, 1 \leq k \leq L - kw + 1\} \tag{4}$$

where $\oplus$ is the sum of element-wise product, $D_{u,\{l:l+kh-1,k:k+kw-1\}}$ denotes the convolutional area of $D_u^{(d)}$ with rows from $l$ to $(l+th-1)$ and the columns from $k$ to $(k+tw-1)$.

Note that the obtained views from the above formulas do not contribute equally to the final results. We introduce an attention mechanism [24] to address this problem. Traditional attention mechanism is used to learn the attentive weights for multiple vectors. In this paper, we modify the formulas in order to learn the weights from multiple matrices (i.e. views). The representation of final view $V_u$ is formed by a weighted sum of these generated views which is calculated as follows:

$$H_u^{(d)} = tanh(V_u^{(d)}) \tag{5}$$

$$\alpha_u^{(d)} = \frac{exp(H_u^{(d)} \oplus W_a)}{\sum_{i=1}^{T_1} exp(H_u^{(i)} \oplus W_a)} \tag{6}$$

$$V_u = \sum_{i=1}^{T_1} \alpha^{(d)} V_u^{(d)}, \tag{7}$$

where $V_u^{(d)}, H_u^{(d)}, W_a, V_u \in \mathbb{R}^{(B-kh+1) \times (L-kw+1)}$, $\alpha_u^{(d)}$ is the attentive weight of the view $V_u^{(d)}$ and $W_a, b_a$ are training parameters. To make the final view into a

latent vector, We use max-pooling to summarize the characteristics of the final view. The formula is as follows:

$$c_u^{in} = max - pooling(V_u) \tag{8}$$

where $c_u^1 \in \mathbb{R}^{B-kh+1}$ is the context representation for in-game behavioral information.

### 4.2 Login Behavior Encoder

**Inputs.** login behaviors in online games can be expressed as login frequency, play time etc. In this paper, we use daily play time for each user to describe the login information. We define a time window with the size of $T_1$ days and consider $T_2$ consecutive time windows before the $observed\_day_1$. The input can be expressed as a sequence $M_u = \{m_1, m_2, .., m_{T_2}\}$ where $m_t$ is a $|T_1|$-dimensional vector representing the duration of each day in the $t-$th windows.

**Recurrent Neural Network (RNN) Layer.** We apply a multi-layer LSTM (Long Short-Term Memory) for long- and short-term modeling since it takes various periods of daily data as time series and has strong ability in learning intrinsic temporal dependencies so as to capture the variation of activity sequences. Each layer of LSTM computes as follows:

$$i_t = \sigma(W_i * h_{t-1} + W_i * m_t + b_i) \tag{9}$$
$$f_t = \sigma(W_f * h_{t-1} + W_i * m_t + b_f) \tag{10}$$
$$\tilde{c}_t = \tanh(W_c * h_{t-1} W_i * m_t + b_c) \tag{11}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{12}$$
$$g_t = \sigma(W_g * h_{t-1} + W_g * m_t + b_g) \tag{13}$$
$$h_t = g_t \odot \tanh(c_t) \tag{14}$$

where $i_t, f_t, g_t$ and $c_t$ are an input gate, a forget gate, an output gate and memory state at time t, respectively. $h_t$ represents the hidden state vector and we set $h_0 = \mathbf{0}$ by default. $W_i, W_f, W_c, W_g, b_i, b_f, b_c, b_g$ are training parameters. The output of the last LSTM will be considered as the context representation of login information $c_u^2 = h_{T_2}$.

### 4.3 Fusion and Prediction Layer

After obtaining two kinds of context embedding vectors, i.e. $c_u^{in}$ and $c_u^{out}$, we concatenate these vectors into a unified vector $c_u$ which will be considered as high-level representation of behavioral features. We feed it into a fully connected feed forward neural network and output the final probability for churn prediction. The unified embedding via fusion can be denoted as:

$$c_u = [c_u^1, c_u^2] \tag{15}$$
$$y_u = \sigma(Wp * c_u + b_p)) \tag{16}$$

where [.] is a concatenate operation, $W_p, b_p$ are parameters in this layer.

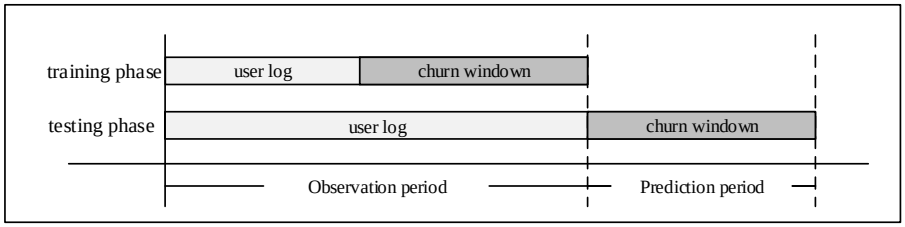


Fig. 7: Label Generation Process. We use different churn windows in the training and testing phase where the churn window of the testing set is behind the training set.

### 4.4 Loss Function and Optimization

At last, we adopt cross-entropy as our loss function for model optimization. To prevent over-fitting, we adopt $l_2$ regularization on the parameters in our loss function. The objective function is defined as follows:

$$\mathcal{L} = -\sum [y_i \log \hat{y_i} + (1 - y_i) \log(1 - \hat{y_i})] + \lambda \|\Theta\|^2 \tag{17}$$

where $\hat{y_i}$ is the probability of becoming churners for user $u_i$ and $y_i$ is the corresponding truth score. If user $u$ is a churner, then we have $y_i = 1$; otherwise, $y_i = 0$. $\Theta$ represents all of model parameters that will be learned in the training phase and $\lambda$ is the regularization weight. We use Adam optimizer to learn the model.

## 5 Experiment

In this section, we aim to answer the following research questions:

- **RQ1**: How does *ChurnPred* perform as compared with widely used methods and the state-of-the-art ones in churn prediction?
- **RQ2**: What are the effects of the in-game behavior encoder and login behavior encoder in our proposed method?
- **RQ3**: How do different hyper-parameter settings (e.g. dimension of embedding vectors) affect the performance of *ChurnPred*?

### 5.1 Dataset and Experimental Setup

We conduct the experiments on a real-world dataset described in Sec. 3. In particular, we extract the daily events of the users in the game and arrange them in chronological order serve as the features of in-game behavioral information (i.e. behavior sequences). The length of each daily behavior sequence will be considered as the features of login information (i.e. the daily play time) preprocessed by normalization. When constructing the training and testing samples, inspired by the paper [9], we adopt a similar splitting process to eliminate the

Table 1: Detailed statistics of the three periodic dataset. We use three subset of the raw dataset with different churn windows for testing.

| Dataset | Phase | Churn Window | Churners | Non-churners | Total Users |
|---|---|---|---|---|---|
| MMORPG_1 | Train | 2018-07-20∼2018-07-26 | 40907 | 81814 | 122721 |
| | Test | 2018-07-27∼2018-08-02 | 103395 | 206790 | 310185 |
| MMORPG_2 | Train | 2018-07-27∼2018-08-02 | 103395 | 206790 | 310185 |
| | Test | 2018-08-03∼2018-08-09 | 28157 | 56314 | 84471 |
| MMORPG_3 | Train | 2018-08-03∼2018-08-09 | 28157 | 56314 | 84471 |
| | Test | 2018-08-10∼2018-08-16 | 50550 | 101100 | 151650 |

problem of data leakage. Besides, a down-sampling approach is applied to avoid a skewed distribution [3, 10], i.e. the ratio of churners to non-churners is 1:2. In order to better evaluate the performance of our model, we divide the dataset into three subsets where the churn windows in the testing phase are three consecutive weeks. The description of these datasets are illustrated in Table 1.

### 5.2 Evaluation Metrics and Baselines

Three widely used evaluation metrics, i.e. Precision, F1-Score and Accuracy, are adopted as metrics and performance is recorded when achieves the best F1-Score. Each experiment is run 5 times to take the best F1-Score as the final performance. To verify the effectiveness of the proposed *ChurnPred* model, we compare it with the following baselines:

- **Logistic Regression (LR)** [10, 14]: This is a popular linear classification algorithm with login information. It analyzes the relationship between one or more existing independent variables.
- **Random Forest (RF)** [10]: This is a traditional classifier based on ensemble learning containing a multitude of decision trees and make the decisions together for classification. The inputs are as same as LR.
- **Multi-layer Perceptron (MLP)** [14]: Multi-layer perceptron is an artificial neural network which maps a set of input vectors into a low-dimensional space. We implement MLP with 2 fully-connected layers with the inputs of login information.
- **wClusterDist** [3]: wClusterDist is a distance-based classification schema conducted on login information as well as the derived features in three semantic dimensions of engagement, enthusiasm and persistence.
- **LSTM+Attention (ATT-LSTM)** [15]: This is an attention-based LSTM model for classifying early churn users whose input is the user behavior event sequence binned at constant intervals. The inputs are sequences of user in-game behaviors after registration.
- **PLSTM+** [22]: This is a two-step framework involving interpretable clustering and churn prediction. The prediction model is based on LSTM by

Table 2: The overall performance on MMOPRG_1, MMORPG_2 and MMOPRG_3.

| Method | MMORPG_1 | | | MMORPG_2 | | | MMORPG_3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | F1-Score | Accuracy | Precision | F1-Score | Accuracy | Precision | F1-Score | Accuracy |
| wClusterDist | 0.3395 | 0.5068 | 0.3515 | 0.3458 | 0.5139 | 0.3696 | 0.3424 | 0.5101 | 0.3599 |
| ATT-LSTM | 0.3334 | 0.4999 | 0.3340 | 0.3333 | 0.4997 | 0.3338 | 0.3334 | 0.5000 | 0.3339 |
| PLSTM+ | 0.3959 | 0.3503 | 0.6116 | 0.4275 | 0.4651 | 0.6090 | 0.3225 | 0.1450 | 0.6324 |
| RF | 0.6024 | 0.2456 | 0.6841 | 0.4306 | 0.5098 | 0.5995 | 0.4270 | 0.3515 | 0.6326 |
| LR | 0.5027 | 0.5589 | 0.6690 | 0.4072 | 0.5426 | 0.5432 | 0.4109 | 0.3803 | 0.6155 |
| MLP | 0.5923 | 0.6140 | 0.7329 | 0.3826 | 0.5142 | 0.5062 | 0.4032 | 0.5363 | 0.5386 |
| ChurnPred | **0.6043** | **0.7022** | **0.7631** | **0.4577** | **0.5459** | **0.6250** | **0.4807** | **0.5720** | **0.6478** |

leveraging the correlations among users' multidimensional activities and the underlying user type is derived from the interpretable clustering. Similar to its original inputs, we take the daily occurrences of the 10 most frequent in-game behaviors as inputs for prediction.

### 5.3 Parameter Settings.

Neural network-based models are all implemented in Pytorch[5] including ChurnPred, ATT-LSTM, PLSTM+ and MLP. These models are optimized with the Adam optimizer, and the batch size is set as 512 by default. In terms of hyperparameters, we apply a grid search for hyperparameters on neural networks: the learning rate is tuned among {0.0001, 0.001, 0.01}, the size of hidden layer and the embedding matrix is in {8, 16, 32, 64} and the threshold is in {0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9}. For ChurnPred, the convolution is done by convolution kernels with the width of 3 and the height equals to the size of embedding matrix. For PLSTM+, we set the $\lambda$ in the loss function as 1 and use 2 hidden layers in each LSTM. For wClusterDist, we set the number of clusters as 5. For ATT-LSTM, we use 2-layer LSTMs and set the dropout as 0.5.

### 5.4 Performance Comparison (RQ1)

Table 2 shows the performance comparison of the proposed model and the state-of-the-art ones. Key observations from the experimental results are listed as follows:

– Among the conventional methods LR, RF and MLP, RF has the best average Precision while MLP has the best average F1-score on three datasets, which shows that RF is relatively correct on predicted churners but fails to find out more true samples since RF mistakes the churners as non-churners in most cases. MLP have a higher F1-score among these methods. The possible reason

[5] https://pytorch.org/

is that the model predicts non-churners as churners as much as possible, which recalls more and more true samples and thus increases the F1-score with the decline in precision.
- In the comparison models, both of PLSTM+ and ATT-LSTM use the in-game behavioral information of users as inputs but show different performance. PLSTM+ performs poorly, showing that the frequency of users' behavioral events is unable to fully describe the recent behavioral information of users. Instead, ATT-LSTM uses behavioral sequences as inputs and achieves better performance which implies the potential behavior patterns in the user's behavior sequences have ability of indicating whether users leave the game.
- RF, LR, MLP and wClusterDist all use login information as inputs. In these models, MLP performs the best on average F1-score metrics followed by wClusterDist. The result shows that MLP can capture these dynamic changes in login sequences while RF and LR lack the ability to encode this information. wClusterDist benefits from the derived features in three semantic dimensions of engagement, enthusiasm and persistence which describes changes in the users' login status, resulting in better performance.
- ChurnPred generally outperforms all baselines. This is largely due to considerations on login information and in-game behavioral information in online games. For in-game behaviors, it leverages the multi-view mechanism to learn the potential behavioral patterns. For login information, it is sensitive to the changes on daily play time which indicates the model capture the dynamic characteristics in login information. By integrating the two kinds of information, the model has been greatly improved.

### 5.5 Component Analysis (RQ2)

In order to evaluate the performance between login behavior encoder and in-game behavior encoder in our proposed method, we design three different models: ChurnPred-$\alpha$ retains login behavior encoder, ChurnPred-$\beta$ uses only in-game behavior encoder and ChurnPred adopts the above two kinds of components. These three models are conducted on $MMORPG\_1$ dataset and keep the same model parameters when training. The results are shown in Fig 8. We can see that ChurnPred achieves the best performance, ChurnPred-$\beta$ is the second and ChurnPred-$\alpha$ is the third. It shows that in-game behavioral sequences implies the potential behavior pattern, which contains more information about the user's intention of leaving when compared with login information in online games. Further, the results demonstrate the effectiveness of our proposed ChurnPred in encoding the intrinsic sequential patterns and login patterns, both of which contributes a lot in the decision-making process.

### 5.6 Parameter Sensitivity (RQ3)

To investigate the robustness of the $ChurnPred$ model, we study how the different choices of parameters affect the performance. Except for the parameter

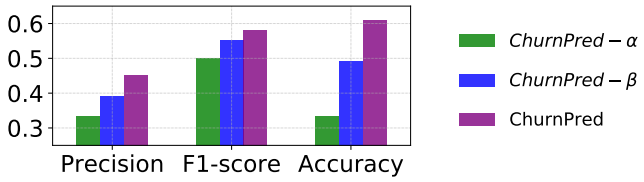


Fig. 8: Effect of in-game behavior and login information encoders.

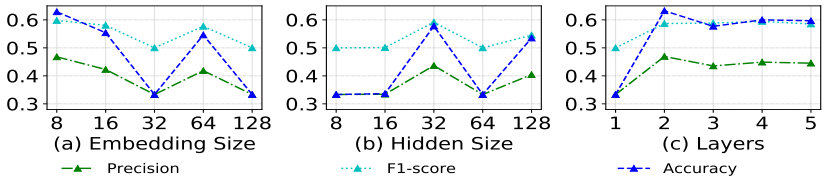


Fig. 9: Effect of the hyper-parameters.

being tested, we set other parameters to default values. The experiments are conducted on $MMORPG\_1$ dataset.

**Effect of Embedding Size.** Figure 9(a) shows the performance in different dimension size of embedding matrix. We can observe that model performance generally declines with the increase of dimensions, which shows that the proposed model is sensitive to the dimension of the embedding matrix. choosing a reasonable dimension length gives the model superior performance and vice versa.

**Effect of Hidden State Dimension.** We keep the same hyper-parameters and vary the dimension of hidden state *hidden_dim* in LSTM component in the range of {8, 16, 32, 64, 128} to investigate whether ChurnPred can benefit from the dimension size. The experimental results are illustrated in Figure 9(b). As we can see, the model have the best performance when *hidden_dim*=2 and perform poorly in most cases, which means that the dimension size of the hidden state needs to be selected appropriately and otherwise it will get worse.

**Effect of Layer Numbers.** Figure 9(c) shows the performance in various number of hidden layers *n_layer* in LSTM component. The best performance is obtained when *n_layer*=2. After that, as the number of layers increases, the performance begin to descend slowly and become stable. This suggests that two layers are enough for the model to achieve significant performance and more layers will not contribute to better performance.

## 6 Conclusion and Future Work

In this paper, we investigate the problem of churn prediction in online games. We first explore and analyze user behaviors of a real-world MMORPG including engagement, days of lifetimes, in-game behaviors etc. According to the analysis insights, we develop a churn prediction model named *ChurnPred* by leveraging in-game behaviors and login behaviors of online games. We propose a time-

aware filtering mechanism and a multi-view mechanism for behavior modeling. Comprehensive experiments conducted on a real-world dataset demonstrate the effectiveness of the proposed model by comparing with state-of-the-art methods. As future work, we will consider the social influence on the in-game behaviors. We argue that richer information could help the model make better decisions on the churn prediction. Further, the scalability problem of the proposed model will be considered in the future.

## Acknowledgments

The paper was supported by the National Natural Science Foundation of China (61702568, U1711267), the Program for Guangdong Introducing Innovative and Entrepreneurial Teams (2017ZT07X355) and the Fundamental Research Funds for the Central Universities under Grant (17lgpy117). Liang Chen is the corresponding author.

## References

1. Bertens, P., Guitart, A., Periáñez, Á.: Games and big data: A scalable multi-dimensional churn prediction model. In: 2017 IEEE Conference on Computational Intelligence and Games (CIG). pp. 33–36. IEEE (2017)
2. Borbora, Z., Srivastava, J., Hsu, K.W., Williams, D.: Churn prediction in mmorpgs using player motivation theories and an ensemble approach. In: 2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing. pp. 157–164. IEEE (2011)
3. Borbora, Z.H., Srivastava, J.: User behavior modelling approach for churn prediction in online games. In: 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing. pp. 51–60. IEEE (2012)
4. Castro, E.G., Tsuzuki, M.S.: Churn prediction in online games using players' login records: A frequency analysis approach. IEEE Transactions on Computational Intelligence and AI in Games **7**(3), 255–265 (2015)
5. Chen, L., Liu, Y., He, X., Gao, L., Zheng, Z.: Matching user with item set: collaborative bundle recommendation with deep attention network. In: Proceedings of the 28th International Joint Conference on Artificial Intelligence. pp. 2095–2101. AAAI Press (2019)
6. Dauphin, Y.N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 933–941. JMLR. org (2017)
7. Kawale, J., Pal, A., Srivastava, J.: Churn prediction in mmorpgs: A social influence based approach. In: 2009 International Conference on Computational Science and Engineering. vol. 4, pp. 423–428. IEEE (2009)
8. Kwon, H., Jeong, W., Kim, D.W., Yang, S.I.: Clustering player behavioral data and improving performance of churn prediction from mobile game. In: 2018 International Conference on Information and Communication Technology Convergence (ICTC). pp. 1252–1254. IEEE (2018)

9. Liu, X., Xie, M., Wen, X., Chen, R., Ge, Y., Duffield, N., Wang, N.: A semi-supervised and inductive embedding model for churn prediction of large-scale mobile games. In: 2018 IEEE International Conference on Data Mining (ICDM). pp. 277–286. IEEE (2018)
10. Milošević, M., Živić, N., Andjelković, I.: Early churn prediction with personalized targeting in mobile social games. Expert Systems with Applications **83**, 326–332 (2017)
11. Nie, G., Wang, G., Zhang, P., Tian, Y., Shi, Y.: Finding the hidden pattern of credit card holder's churn: A case of china. In: International Conference on Computational Science. pp. 561–569. Springer (2009)
12. Pudipeddi, J.S., Akoglu, L., Tong, H.: User churn in focused question answering sites: characterizations and prediction. In: Proceedings of the 23rd International Conference on World Wide Web. pp. 469–474. ACM (2014)
13. Ren, K., Qin, J., Zheng, L., Yang, Z., Zhang, W., Qiu, L., Yu, Y.: Deep Recurrent Survival Analysis. arXiv e-prints arXiv:1809.02403 (Sep 2018)
14. Runge, J., Gao, P., Garcin, F., Faltings, B.: Churn prediction for high-value players in casual social games. In: 2014 IEEE conference on Computational Intelligence and Games. pp. 1–8. IEEE (2014)
15. Sato, K., Oka, M., Kato, K.: Early churn user classification in social networking service using attention-based long short-term memory. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining. pp. 45–56. Springer (2019)
16. Tao, J., Xu, J., Gong, L., Li, Y., Fan, C., Zhao, Z.: Nguard: A game bot detection framework for netease mmorpgs. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 811–820. ACM (2018)
17. Umayaparvathi, V., Iyakutti, K.: Automated feature selection and churn prediction using deep learning models. International Research Journal of Engineering and Technology (IRJET) **4**(3), 1846–1854 (2017)
18. Viljanen, M., Airola, A., Heikkonen, J., Pahikkala, T.: Playtime measurement with survival analysis. IEEE Transactions on Games **10**(2), 128–138 (2017)
19. Wu, J., Yuan, Q., Lin, D., You, W., Chen, W., Chen, C., Zheng, Z.: Who are the phishers? phishing scam detection on ethereum via network embedding. arXiv preprint arXiv:1911.09259 (2019)
20. Xie, F., Chen, L., Ye, Y., Zheng, Z., Lin, X.: Factorization machine based service recommendation on heterogeneous information networks. In: 2018 IEEE International Conference on Web Services (ICWS). pp. 115–122. IEEE (2018)
21. Xie, Y., Li, X., Ngai, E., Ying, W.: Customer churn prediction using improved balanced random forests. Expert Systems with Applications **36**(3), 5445–5449 (2009)
22. Yang, C., Shi, X., Jie, L., Han, J.: I know you'll be back: Interpretable new user clustering and churn prediction on a mobile social application. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 914–922. ACM (2018)
23. Yuan, S., Bai, S., Song, M., Zhou, Z.: Customer churn prediction in the online new media platform: A case study on juzi entertainment. In: 2017 International Conference on Platform Technology and Service (PlatCon). pp. 1–5. IEEE (2017)
24. Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H., Xu, B.: Attention-based bidirectional long short-term memory networks for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 207–212 (2016)