A Latent Feelings-aware RNN Model for User Churn Prediction with only Behaviour data

Meng Xi*, **Zhiling Luo***, **Naibo Wang***, **Jianrong Tao**†, **Ying Li*** and **#Jianwei Yin***
*Zhejiang University, Hangzhou, China
†User Persona Group, NetEase Fuxi AI Lab, Hangzhou, China
#corresponding author
{ximeng, luozhiling, wangnaibo, cnliying, zjuyjw}@zju.edu.cn
, hztaojianrong@corp.netease.com

***Abstract*—User Churn Prediction is a cutting-edge research area in the web service industry, it is the key for managing the user in the virtual world and provide feedback information for improving the corresponding web service. At present, most of the relevant work is to design a questionnaire to collect data of users' characteristics and feelings and then develop a general model by finding relevance. However, that kind of methods requires quite a time and manpower, and most web services can only obtain logs of users' behaviours and have no access to users' feature data. Therefore, it is a big challenge to conduct user churn prediction with only behavior data and get users' latent feelings from their action data in order to improve the accuracy of churn prediction. In this paper, a novel Latent Feelings-aware RNN model, namely LaFee, has been proposed to solve the user churn prediction problem by using only behaviour data. The latent feelings, proven to be satisfaction and aspiration, can be estimated through the intermediate variable of the trained LaFee. We also designed experiments on a real dataset and the results show that our methods outperform the baselines.**

***Index Terms*—web services; churn prediction; machine learning; latent feeling; user satisfaction; aspiration;**

I. Introduction

The term *user churn* is used in the information and communication technology industry to indicate the customers who are going to end their subscriptions and find a new competitor [1], [2]. The problem of user churn could be different for web services. Because the users can obtain the web services directly on corresponding web pages, the owner of the service cannot flag the user churn through events like cancel account or uninstall the application [3]. In the field of web services, user churn is always judged by the logout-login interval. For instance, one could be flagged "churn" if the user has not logged in for a week.

The traditional methods to churn prediction always consists of feature engineering, data labelling, and prediction model training. These kinds of methods largely rely on the features which are already existed or well-organized in the raw data. However, the soundness of those methods depend heavily on the qualities of the features selected and can hardly cope with the common cases where only user behaviour data are accessible. Besides, quite an amount of researches regard machine learning methods as black boxes and the models of those methods usually lack interpretation.

We mainly concentrate on two problems here:

1. How to conduct user churn prediction with only behavior data?

2. How to get users' latent feelings from their action data in order to improve the accuracy of churn prediction?

In this work, we transform the churn prediction from a classification problem into a regression problem. That is to say, what we predict is not whether the users will be lost, but the time interval between this logout and the next login. We propose a novel latent feelings-aware RNN model, namely LaFee, to solve the user churn prediction problem through only behaviour data. In addition to predict user churn, we also extract the latent feelings, include satisfaction and aspiration, of the users through LaFee. The satisfaction indicates the pleasure a user gets from every part of the web service or the fulfilment degree of his or her expectations or needs. The aspiration indicates the wish or desire to perform an operation in the service. The satisfaction and the aspiration are used to express the latent feelings from two different aspects. For instance, after using a service continuously for a long time, the aspiration of the user may decrease due to fatigue. Nevertheless, satisfaction may increase because the service works well and fulfils the user needs. By using these latent feelings, operators can take measures against lost users in a more timely and targeted manner.

The main contributions are as follows:

- A novel latent feelings-aware RNN model, called LaFee, is proposed. It can transform the churn prediction from a classification problem into a regression problem. And LaFee is able to predict the user churn in web service scenarios where only behaviour data is available and the churn events are inexplicit. Meanwhile, the latent feelings learned by Lafee can help to improve the accuracy of existing methods.
- The users' latent feelings are introduced, learned and discussed. In this work, the latent feelings learnt by LaFee are named as satisfaction and aspiration, not only for that we designed so, but also for that they are proven own same properties as users' realistic feelings.
- Some innovative and useful information such as user patterns and psychological states are reflected through statistics and analysis of satisfaction and aspiration.

978-1-7281-8777-8/20/$31.00 ©2020 IEEE
DOI 10.1109/SMDS49396.2020.00011

The rest of this paper is organized as follows. Section II briefly reviews the related work. Section III introduces our proposed method in detail. Section IV introduces the dataset we used. Section V reports our experimental evaluation, and Section VI discussed the latent feelings learned. Section VII concludes the paper.

II. Related Works

A. User satisfaction

As far as we know, most of the researches on user satisfaction are focused on the field of the computer-human interaction. There are mainly two kinds of approaches to satisfaction modeling: the qualitative approaches and the quantitative approaches [4]. In the qualitative approaches, the researchers focused on modeling the incorporating flow to evaluate and analyze user satisfaction through clustering methods [5], [6]. Since we want to quantify the latent feelings of the users, we mainly investigate the quantitative approaches.

There are a lot of good researches in this field. One of the most common research routes is to propose hypothesizes or research questions, collecting interaction data, investigating user satisfaction, designing features (feature engineering), predicting satisfaction through the features, and verify the hypothesizes proposed at the beginning [7]–[9]. User participation is necessary for these kinds of researches. Otherwise, the researchers cannot get the ground truth of the proposed prediction model or verify the validity of their features. The ground truth of satisfaction can be obtained by letting the participants score directly or analyzing their physiologic data. It is hard to be put into practice in situations like web services where the users are unavailable. A model geared to Skype overcomes this problem with an objective source- and network-level metrics like bit rate, bit rate jitter and round-trip time to evaluate user satisfaction [10]. However, that kind of research is confined to a specific situation. In this work, we are committed to proposing a general method to estimate users' latent feelings through the state of the hidden layer in the process of predicting user churns.

B. Churn prediction

Churn prediction is always a hotspot in both academia and industry. When we need to model user churn, we need to take into account the user's use history, including state sequences, behaviour sequences, and these reasons may come from external interference and accidents.

Usually, classification algorithms are used to solve the user churn detection and prediction problem. AC Bahnsen et al. constructed a cost-sensitive customer churn prediction model framework by introducing a new financial-based approach, which enables classification algorithms to better serve business objectives [11]. Z Kasiran et al. applied and compared the Elman recurrent neural network with reinforcement learning and Jordan recurrent neural network to the prediction of the loss of the subscribers of the abnormal telephones [1]. A Hudaib et al. mixed K-means algorithm, Multilayer Perceptron Artificial Neural Networks (MLP-ANN) and self-organizing maps (SOM) to establish a two-stage loss prediction model, which was tested on real datasets [2].

The problem of user churn has also been taken seriously in the game area. Since the game manufacturer only got user behaviour data, EG Castro et al. used four different methods to convert them into a fixed length of the data array, and then took these items as input, trained probabilistic classifiers using k-nearest neighbour machine learning algorithm, which made better use of the information in the data and getting better results [3].

However, in practical applications, due to the different criteria of churn, we need to train and run several models under different criteria to get satisfying results. Therefore, we propose a method which can transform the churn prediction from a classification problem into a regression problem. It can predict the time that may last after a user's logging out. And this time can be used to calculate various user churn rates.

C. Recurrent neural network

The recurrent neural network realizes the mapping from the input sequence to the output sequence by adding a self-looping edge in the neural network and has a wide range of applications in the problems of recognition, prediction, and generation. However, the performance on long-term dependencies is poor, and the difficulty of convergence increases with the length of the sequence [12].

Therefore, S Hochreiter et al. proposed a long short-term memory (LSTM) algorithm to solve this problem [13]. The design of the LSTM neural unit is shown in Figure 1. Its calculation Equation is as follows:

$$f_t = \sigma(W_f \cdot [H_{t-1}, I_t] + b_f) \tag{1}$$

$$i_t = \sigma(W_i \cdot [H_{t-1}, I_t] + b_i) \tag{2}$$

$$\widetilde{C}_t = \tanh(W_c \cdot [H_{t-1}, I_t] + b_c) \tag{3}$$

$$o_t = \sigma(W_o \cdot [H_{t-1}, I_t] + b_o) \tag{4}$$

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C}_t \tag{5}$$

$$H_t = o_t * \tanh(C_t) \tag{6}$$

There are many improvements and variants based on RNN or LSTM. FA Gers et al. enhanced LSTM expression and experimental results [14] by adding "peephole connections" between internal cells and multiplication gates. J Koutnk et al. improved the performance and training speed of the network by reducing the number of RNN parameters, dividing the hidden layer into independent modules and inputting their respective time granularities, enabling them to perform iterative calculations at their respective clock rates [15]. J Bayer et al. used the advantages of variational reasoning to enhance RNN through latent variables and built Stochastic Recurrent Networks [16]. N Kalchbrenner et al. extended the LSTM to apply it to multidimensional grid spaces to enhance the performance of LSTM on high-dimensional data such as images [17]. Due to the extensive use of RNN and LSTM and a large number of variants, it is difficult to judge whether

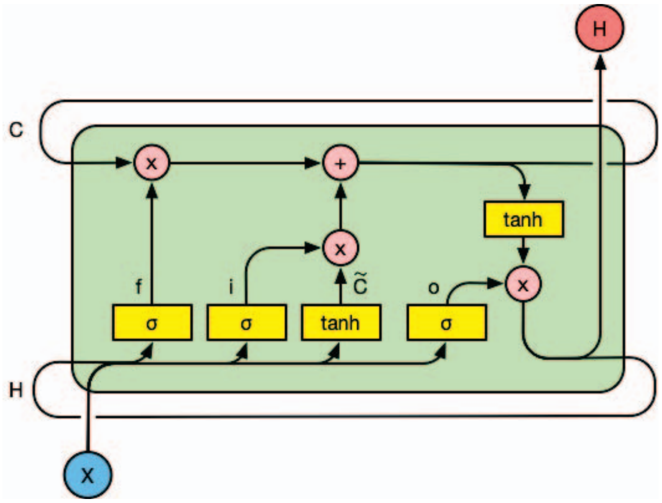


Fig. 1. Sample graph of LSTM cell.

the RNN structure used in the current scenario is optimal. Therefore, some researchers have analyzed and evaluated RNN, LSTM and its variants in various tasks such as speech recognition and handwriting recognition to eliminate people's doubts in this area. At the same time, they found and proved that the forgetting gate and output activation function is the key component [18], [19].

RNN and LSTM are widely used and studied in various fields. Y Fan et al. used a recurrent neural network with two-way long-term memory cells to capture correlation or co-occurrence information between any two moments in speech, parametric text-to-speech (TTS) synthesis [20]. K Cho et al. encodes and decodes the symbol sequence [21] by combining two RNNs into one RNN codec. K Gregor et al. Constructed a deep loop attention writer for image generation by combining a novel spatial attention mechanism that simulates the concave of the human eye with a sequential variational automatic coding framework that allows iterative construction of complex images. PLD) Neural Network Structure [22]. A Ando et al. achieved a dialogue-level customer satisfaction modeling method [23] by jointly modeling two long-term short-term regression neural networks (LSTM-RNNS).

These methods usually do not perform well on long-term problems, so our method is designed to introduce users' latent feelings and improve performance.

III. LaFee

In this section, we will introduce LaFee in detail. LaFee's innovation is mainly reflected in two aspects. For one thing, it converts the classification problem of the churn prediction into a regression problem. LaFee predicts the duration of the user's offline status based on the behaviour data and calculates the classification accuracy through setting the churn time threshold. Lafee solves the problem of low quantity of logout actions, which leads to difficulty in training. For another, LaFee contains a dual-output RNN model, which modified the calculation logic of LSTM cell, to obtain the user's latent feelings while predicting user churn.

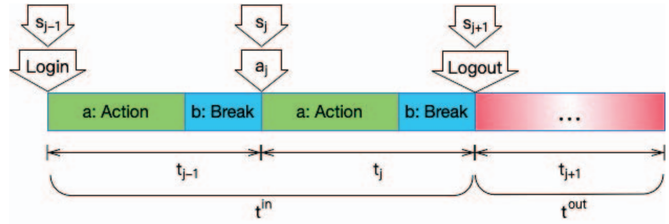


Fig. 2. Illustration of Time Slice in This Work.

A. Problem Formalization

This section is the behaviour-based modelling method for user churn prediction used in LaFee. Traditionally, the churn prediction was treated as a classification problem. In the web service scenario, it's hard to tell if a user is churned due to the lack of churn events. Therefore, we define user churn by offline time threshold of τ. If a user did not log in during time τ, he or she would be judged as churn. For example, it can be considered that a user has been lost if one has not been online for 7 consecutive days. By this way, we transfer the churn prediction problem into a regression problem of offline time t prediction.

In order to make full use of the log data, we inferred the users' sequences of the states and calculated the time intervals through the sequence of the behaviours. The relationship among the three sequences is: in a time slice, the user performs an action a under state s, and waits for the time interval t before executing the next action. Based on the above analyses, we remodel the problem as below:

Defintion 1. *Observing a set of user log files. Each file can be transferred into a sequence ζ. By transformming all log files, we can get $\mathcal{D} = \{\zeta_1, \zeta_2, ..., \zeta_N\}$ where $\zeta_i = \{(s_j, a_j, t_j)_{j\in 1...T}\}$. Each ζ_i is a composition of state, action, and time sequence of a user. Within a time slice, the relation of (s_j, a_j, t_j) is that player takes the action a_j under the state s_j and cost the time t_j (see Figure 2). If a_j is logout, the type of t_j is offline time and belong to t^{out}. Otherwise, t_j is online time and belong to t^{in}. Given a time τ as a user churn threshold, train a model $\mathbb{M}$ that minimizes the variance between t_j (of t^{out}) and $t_j^* = \mathbb{M}(s_j, a_j)$, and can maximize the churn prediction accuracy (see Equation 7) under threshold τ at the same time, where $N_{t^{out}}$ means the total number of a user's log out actions.*

$$accuracy = \frac{\Sigma_i^{N_{t^{out}}} 1((t_i^{out} \geq \tau \wedge t_i^{out*} \geq \tau) \vee (t_i^{out} < \tau \wedge t_i^{out*} < \tau))}{N_{t^{out}}} \tag{7}$$

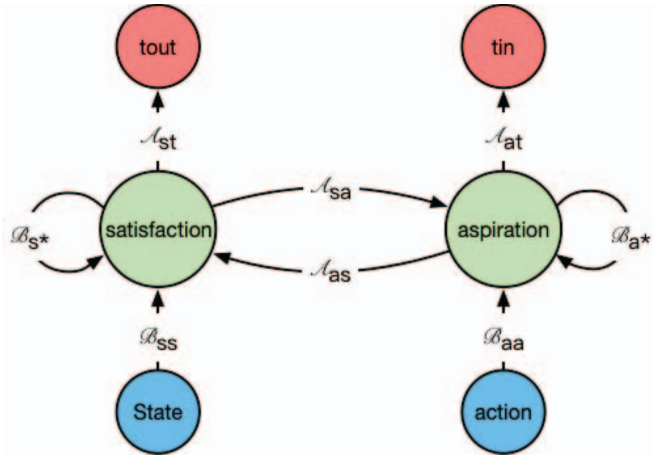


Fig. 3. Graphic model of LaFee.

B. Latent Feelings-aware RNN model

In cooperation with industry, we learned that it is as important to understand the reasons for the user churn to predict the user loss. The web service operators wish to have a way, in addition to predicting users' churn but also can estimate users' real-time subjective feelings, which could help them to respond in advance. We use the satisfaction and aspiration to describe users' latent feelings from two different aspects. The satisfaction indicates the pleasure a user gets from every part of the web service or the fulfilment degree of his or her expectations while the aspiration indicates the wish or desire to perform an operation in the service. We designed a dual-output latent feelings-aware RNN model to get users' latent feelings here.

Here we will illustrate our model by introducing its graphic structure first and then explain the sample graph of Lafee Cell.

We suppose that the user experience will determine the time interval among the users' actions. And t^{in} is primarily determined by the user aspiration, and t^{out} is primarily determined by user satisfaction. So, we designed a dual-output RNN models to predict t^{in} and t^{out} separately (see Figure 3). The loss functions of the two output are independent but former layers' parameters are shared. We take $\mathscr{A}$ as a linear transformation and $\mathscr{B}$ as a non-linear transformation.

Here we will talk about the process of calculating t^{in} first, this process occurs when the user is online and before logging out. Firstly, we calculate the tensor of satisfaction, which depends on the current state and previous satisfaction (see Equation 8). Then we can calculate the aspiration by the satisfaction we just got, the action of this time slice, and the previous aspiration (see Equation 9). Finally, as t^{in} is primarily determined by the user aspiration, t^{in} could be obtained by aspiration through Equation 10.

$$\begin{aligned} satisfaction_{t^{in}} &= \mathscr{B}_{ss}(state_{t^{in}}) \\ &+\mathscr{B}_{s*}(satisfaction_{t^{in}-1}) \end{aligned} \tag{8}$$

$$\begin{aligned} aspiration_{t^{in}} &= \mathscr{A}_{sa}(satisfaction_{t^{in}}) \\ +\mathscr{B}_{aa}(action_{t^{in}}) &+ \mathscr{B}_{a*}(aspiration_{t^{in}-1}) \end{aligned} \tag{9}$$

$$t^{in} = \mathscr{A}_{at}(aspiration_{t^{in}}) \tag{10}$$

Then we talk about the process of calculating t^{out}, which occurs when the user is logging out. Firstly, we calculate the tensor of aspiration, which depends on the current state and previous aspiration (see Equation 12). Then we can calculate the satisfaction by the aspiration we just got, the action of this time slice, and the previous satisfaction (see Equation 11). Finally, as t^{out} is primarily determined by the user satisfaction, t^{out} could be obtained through Equation 13. It is clear that $\mathscr{B}_{ss}, \mathscr{B}_{s*}, \mathscr{B}_{aa}, \mathscr{B}_{a*}$ are shared by two processes. The rest transformations are owned by each.

$$\begin{aligned} aspiration_{t^{out}} &= \mathscr{B}_{aa}(action_{t^{out}}) \\ &+\mathscr{B}_{a*}(aspiration_{t^{out}-1}) \end{aligned} \tag{11}$$

$$\begin{aligned} satisfaction_{t^{out}} &= \mathscr{A}_{as}(aspiration_{t^{out}}) \\ +\mathscr{B}_{ss}(state_{t^{out}}) &+ \mathscr{B}_{s*}(satisfaction_{t^{out}-1}) \end{aligned} \tag{12}$$

$$t^{out} = \mathscr{A}_{st}(satisfaction_{t^{out}}) \tag{13}$$

Note that Equation 11 is different from Equation 9 since they are calculated when the user is online and offline respectively, which can be seen from t_{in} and t_{out}. Similarly, Equation 12 and 8 are different too.

After having a general understanding of the model, we then elaborate the calculation methods of all transformations and tensors in LaFee.

As both satisfaction and aspiration have sequential dependencies, we redesign the method of RNN to establish the time sequence relationship between the value of satisfaction or aspiration. We build a variant of LSTM to calculate and store the values of satisfaction and aspiration. We hide the output layer, merge the cell state and the hidden layer, and simplify the design of the input gate and the forget gate (see Figure 4).

First is case of calculating t^{in}. The first step is to calculate satisfaction. We calculate $Us_{t^{in}}$ by Equation 14 using sigmoid function as activation function, current state and previous satisfaction as input. As $Us_{t^{in}}$ indicates what to remember, $1 - Us_{t^{in}}$ indicates what to forget. Then we calculate a candidate satisfaction $\widetilde{Cs}_{t^{in}}$ that used $\tanh$ function and same input as $Us_{t^{in}}$ (Equation 15). Finally, new satisfaction is an addition of the values to be remembered (calculated by multipling $Us_{t^{in}}$ and $\widetilde{Cs}_{t^{in}}$) and memories left by previous satisfaction (gained by multipling $1 - Us_{t^{in}}$ and $sat_{t^{in}-1}$). The Equations 14 - 16 correspond to Equation 8.

$$Us_{t^{in}} = \sigma(W_{us} \cdot [sat_{t^{in}-1}, state_{t^{in}}] + b_{us}) \tag{14}$$

$$\widetilde{Cs}_{t^{in}} = \tanh(W_{cs} \cdot [sat_{t^{in}-1}, state_{t^{in}}] + b_{cs}) \tag{15}$$

$$sat_{t^{in}} = (1 - Us_{t^{in}}) * sat_{t^{in}-1} + Us_{t^{in}} * \widetilde{Cs}_{t^{in}} \tag{16}$$

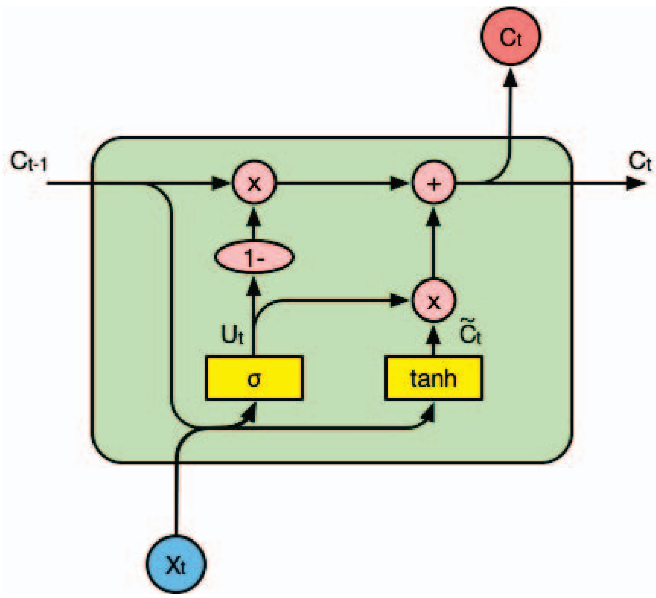


Fig. 4. Sample graph of LaFee cell.

The second step is to calculate the aspiration. On one hand, aspiration depends on the action the player has taken recently. For instance, one could be tired and lose some aspiration if dozens of works were operated in the past few time slices. On the other hand, the previous aspiration can affect that of this time slice because aspiration is consecutive to some extent. Current satisfaction also has an impact on aspiration. The calculation of aspiration is the same as satisfaction except for the influence of satisfaction is introduced in the last step. Detailed steps are shown in Equations 17 - 19, which correspond to Equation 9.

$$Ua_{t^{in}} = \sigma(W_{ua} \cdot [asp_{t^{in}-1}, action_{t^{in}}] + b_{ua}) \quad (17)$$

$$\widetilde{Ca}_{t^{in}} = \tanh(W_{ca} \cdot [asp_{t^{in}-1}, action_{t^{in}}] + b_{ca}) \quad (18)$$

$$asp_{t^{in}} = \begin{array}{l}(1 - Ua_{t^{in}}) * asp_{t^{in}-1} + Uat^{in} * \\ \widetilde{Ca}_{t^{in}} + W_{sa} \cdot sat_{t^{in}} + b_{sa}\end{array} \quad (19)$$

The final step is to calculate t^{in}. As shown in Figure 2, a player executes action a(not logout) under state s, the corresponding t^{in} consists of two parts: action part and break part. The action part time is mainly determined by the type of the action, while break part is affected by aspiration. t^{in} can be obtained through Equation 20, which corresponds to Equation 10.

$$t^{in} = W_{at^{in}} \cdot asp_{t^{in}} + b_{at^{in}} \quad (20)$$

Equation 21-23 (corresponding to Equation 11) are used to calculate aspiration first. The next step is to use the aspiration just obtained, combined with the state and the previous satisfaction to calculate the current satisfaction (Equation 24-26, corresponding to 12). And t^{out} is calculated at the end by Equation 27 (corresponding to 13). As the two processes

TABLE I
STATES AND ACTIONS OBTAINED AFTER PREPROCESSING.

state	action	
Gold	LoginRole	RewardAchievement
Experience	LogoutRole	InviteLog
EmojisSent	ReplaceRole	ShareLog
GiftsSent	PrivateGame	FollowLog
AchievementGot	QuickMatch1V1	PraisePlayRound
ItemNum	QuickMatch2V2	RoomModeCreate
GradeUp	DailyTaskFinish	ConsumeItem
OnlineDuration	DailyTaskReward	GuideInfo
	DailySign	AdsLog
	DailySignReward	

of calculating t^{in} and t^{out} are relatively symmetrical, detailed description will not be repeated. The calculation process of t^{out} can be clearly seen and understood through Equations below.

$$Ua_{t^{out}} = \sigma(W_{ua} \cdot [asp_{t^{out}-1}, action_{t^{out}}] + b_{ua}) \quad (21)$$

$$\widetilde{Ca}_{t^{out}} = \tanh(W_{ca} \cdot [asp_{t^{out}-1}, action_{t^{out}}] + b_{ca}) \quad (22)$$

$$asp_{t^{out}} = (1 - Ua_{t^{out}}) * asp_{t^{out}-1} + Ua_{t^{out}} * \widetilde{Ca}_{t^{out}} \quad (23)$$

$$Us_{t^{out}} = \sigma(W_{us} \cdot [sat_{t^{out}-1}, state_{t^{out}}] + b_{us}) \quad (24)$$

$$\widetilde{Cs}_{t^{out}} = \tanh(W_{cs} \cdot [sat_{t^{out}-1}, state_{t^{out}}] + b_{cs}) \quad (25)$$

$$sat_{t^{out}} = \begin{array}{l}(1 - Us_{t^{out}}) * sat_{t^{out}-1} + Us_{t^{out}} * \\ \widetilde{Cs}_{t^{out}} + W_{as} \cdot asp_{t^{out}} + b_{as}\end{array} \quad (26)$$

$$t^{out} = W_{st} \cdot sat_{t^{out}} + b_{st} \quad (27)$$

As for loss function, whether calculating t^{in} or t^{out}, our optimization direction is the same. The loss function is to minimize variance between time intervals predicted and the real ones (see 28).

$$loss = \frac{\Sigma_{j=1}^{n}(t_j - t_j^*)^2}{n} \quad (28)$$

IV. Data Overview & Preprocessing

The web game is a common web service. Different from general web services, web games usually have more intensive user behaviour data. So in this work, we use the data of web games to experiment.

The dataset used in this work is from a web card game operated by our cooperative company. The dataset consists of JSON log files from 89,379 users, containing 5,887,094 pieces of log data. The log files record users behaviours except for those of how they play the cards. Therefore, the basic rules of the game are hidden in this work. And we can regard playing a card game as a function of the web game service. Each piece of log consists of three types of data:

- log_id: the identity of the log. The event that generates the log is indicated, like LoginRole, InviteLog, ConsumeItem, etc.
- raw_info: the detailed information recorded in it, including login platform, IP address, server ID, etc. The information type and amount vary depending on the log_id.
- timestamp: the time when the event occurred.

An example of a raw log is shown below (some private information is hidden):


```
[
...
{
"log_id": "LoginRole",
"raw_info": {
"account_id": "22*********",
"ip": "4*.***.**.*81",
...
},
"timestamp": "2018-01-25 07:34:02"
},
...
{
"log_id": "QuickMatch2V2",
"raw_info": {
"total1": 43,
"total2": 43,
"total3": -37,
"total4": -49,
...
},
"timestamp": "2018-01-25 07:38:34"
},
...
{
"log_id": "LogoutRole",
"raw_info": {
"account_id": "22*********",
"ip": "4*.***.**.*81",
...
},
"timestamp": "2018-01-25 08:08:05"
},
...
]
```


JSON files are appropriate for data transmission but not friendly for machine learning. The event recorded in a log could be an action taken by a user, an event triggered by the game system, or a change of the users' states like gold, experience, etc. Sometimes we need to analyze the log_id and raw_info together to get the actual user behaviour. There are errors appeared in the log files. We summarize these errors to provide a reference for the related works in the future. In a web service log data, there are mainly four types of log error: event name(log_id) error, the incompleteness of paired data, duplicate records of a same event, chaotic sequence of the events. Below is our processing method.

Firstly, we corrected logs with log_id wrong through information reasoning and quantity verification. For instance, because the "LoginRole" amount is almost half more than "LogoutRole", we think that there are behaviours that are mistakenly recorded as "LoginRole". By reasoning about the detailed information, we found that some of the " LoginRole " are actually "PrivateGame". Secondly, the logs with repeated meanings were merged. For instance, both "Trade" with 18 gold coins deducted and "MatchInfo" is used to record the beginning of a match. Thirdly, we reordered all the logs through the order specified by the game mechanism. For instance, there are occasions that several events take place at the same time (with the same timestamp), like "LoginRole" and "DailySign", where the logs would be arranged randomly. But the game mechanics stipulate that "LoginRole" must occur before "DailySign". Finally, we inferred the users' sequences of the states and calculated the time intervals through the sequence of the behaviours.

After all these processes, we turned logs into 28-dimensional vectors to facilitate the training of neural networks. There are 8 dimensions of states, 19 dimensions of actions, and 1 dimension of the time intervals. The states and actions obtained are listed in Table I. All states are represented by the specific cumulative quantity, such as "Gold" is the user's current game gold coins, "GiftsSent" is the number of gifts sent by the user so far. For the actions, we use one-hot encoding to generate 19-dimensional behaviour data. The relationship among the three sequences is: in a time slice, the user performs an action a under state s, and waits for the time t before executing the next action.

V. Experiments

Our experiment is mainly divided into three parts. The first part is used to validate the effectiveness of LaFee. The second part and the third part are used to analyze and verify the physical meaning and effectiveness of satisfaction and aspiration through statistical analysis and data mining methods.

Before the experiment started, we extracted four most representative groups of users from the huge mount of log data. First, we sorted the data by users' match-winning percentage, dividing the highest 20% into the high winning-rate group, and the lowest 20% into the low winning-rate group. The winning percentages of the two groups were 62.66% and 19.33%, respectively. Besides, according to the different proportions of users' behaviours, we regard the 20% user with the highest proportion of match behaviours as the battle player, and the user with the least 20% of the match behaviours as the social player. The match behaviours accounted for 18.47% and 5.39%, respectively.

A. Performance Evaluation

In this part, we study the effectiveness of our model. We apply data sets to 4 baseline methods. There are two off-the-shelf classifiers, including Random Forest (RF) [24] and Gradient Boosting Decision Tree (GBDT) [25]. Both of them are implemented by Scikit-learn and the number of trees of RF is set as 500. The other two are deep learning approaches, including Long Short Term Memory(LSTM) [13], Deep Neural Network(DNN) [26]. They are implemented by Tensorflow and Keras, respectively. The time step of LSTM is sixteen and the DNN has six hidden layers.

At the same time, we apply LaFee to all the baselines. We use the suffix of _lf to identify satisfaction and aspiration calculated by the LaFee model as input to each baseline to prove LaFee's validity.

TABLE II
COMPARISON AMONG DIFFERENT APPROACHES

Method	High Winning-rate			Low Winning-rate			Battle Player			Social Player			ALL		
	1 day	3 days	7 days	1 day	3 days	7 days	1 day	3 days	7 days	1 day	3 days	7 days	1 day	3 days	7 days
RF	74.09	81.11	82.32	82.83	86.87	87.54	72.03	81.67	83.60	80.47	86.69	87.57	81.18	89.46	90.87
RF_lf	**81.10**	**86.58**	**87.81**	**88.36**	**92.81**	**93.49**	**72.40**	**82.14**	**84.42**	**85.67**	**92.24**	**93.43**	**84.75**	**92.58**	**93.49**
GBDT	74.32	93.92	97.97	89.18	98.80	99.80	82.37	96.13	98.49	90.53	98.52	99.77	87.56	97.85	99.55
GBDT_lf	**75.69**	**94.44**	**98.61**	**90.15**	**99.18**	**99.90**	**83.30**	**96.48**	**99.12**	**91.04**	**98.81**	**99.96**	**88.31**	**98.18**	**99.62**
DNN	83.81	91.38	92.59	91.25	96.63	97.31	77.63	90.54	92.90	85.53	94.03	95.60	85.45	97.11	99.02
DNN_lf	**87.44**	**95.10**	**96.63**	**93.15**	**97.60**	**98.29**	**80.22**	**91.65**	**93.85**	**87.38**	**94.82**	**95.79**	**86.57**	**97.16**	**99.13**
LSTM	82.32	95.36	98.55	86.85	97.41	99.68	78.78	95.03	98.47	82.13	97.17	99.55	80.67	95.73	99.04
LSTM_lf	**95.01**	**99.21**	**99.74**	**90.15**	**99.18**	**99.90**	**83.30**	**96.48**	**99.12**	**88.51**	**98.63**	**99.86**	**88.31**	**98.18**	**99.62**

All programs are executed on the same computer with 32G RAM, 2.8GHz CPU, and NVidia GeForce TitanX GPU. 70% of data are randomly extracted and used for training and the rest are used as the test set. It was ensured that all approaches are compared on the same test set. We repeated 10 times for each experiment to get the average accuracy.

In Table II, the prediction accuracies of the four groups and all users under three different criteria are accurate to two decimal places. We can see that the accuracies of models with Lafee are always better than the original ones, which proves Lafee do provide useful information to the model, i.e, the latent feelings calculated by Lafee do work. Note that with the increase of τ, the regularity of user churn and the correlation with online behaviour are greatly improved. That is why LaFee can always be the best in long term prediction. We note that the accuracies of 7-day is better than 3-day, while 3-day is better than 1-day, this is because that the longer the user is away, the more fixed their behavior becomes. In fact, the users who do not log in for 7 consecutive days are most likely to be the users who are actually lost. Therefore, under the threshold of 7 days, the models with LaFee involved show the highest accuracies.

As for LSTM, the accuracies increase as the τ increases, because LSTM has introduced timing information. However, we observed that the result of introducing only regression modeling was worse than LSTM, while the result of introducing LaFee was better than LSTM. This is because the regression modeling introduced the online data, which means that the latent feelings within and out the game are learned together, eventually leading to overfitting of the model. LaFee learned satisfaction and aspiration semi-separately. And the performances of LSTM_lf are always the best, which show that LaFee does learn the latent feelings of users. For all the methods and data groups, LaFee involved methods' results are the best in 7-day churn prediction, which also proves the effectiveness of the satisfaction and aspiration we have learned.

B. Satisfaction Analysis

We first studied the relationship between user satisfaction and match-winning rate (see Figure 5). We reduce the tensors of the satisfactions to 1 dimension by averaging. As the user satisfaction is mostly affected by the recent game experiences,

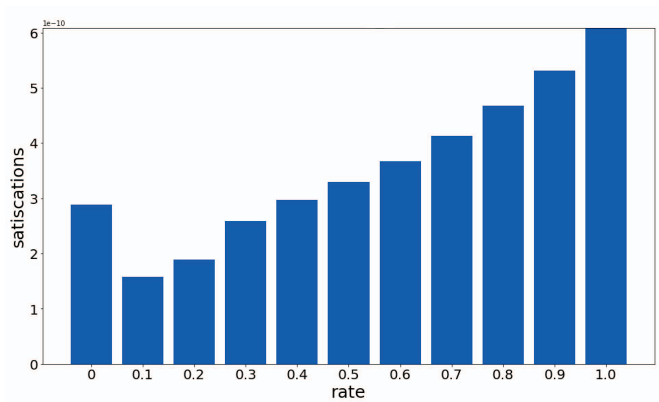


Fig. 5. Satisfaction and winning rate correlation statistics.

we study the relationship between the satisfaction and the winning rate of the users' last 10 match games. The x-axis represents different winning rate and the Y-axis is the average of the satisfaction under the corresponding winning rate.

It can be observed that in the last 10 games, the user satisfactions are mostly increased with the increase of the winning rate. Different from what we have known in the past, it might be the best choice to control the user match winning rate to around 50%, *the user satisfaction can be improved with the short-term victory.* An interesting phenomenon is: when the user loses all 10 games, the satisfactions are higher than winning once. This shows that *compared to the losing streak, occasionally winning one or two games makes the user feel more dissatisfied and painful.*

Then we studied the relationship between user satisfaction and user logout time (see Figure 6). The x-axis is the average of satisfaction in a time domain and Y-axis is the corresponding logout time (t^{out}) domain whose unit is second. Since the value range of t^{out} is very large, to be able to perform a detailed analysis, we conducted our analysis of that under the condition that the login time was within one day.

First, let's look at the green bar at the bottom with the highest satisfaction. The corresponding time interval is about 3 hours. In other words, *the user re-login once every three hours or so means that the user is satisfied with the game.* This is understandable in combination with our daily experience. 3

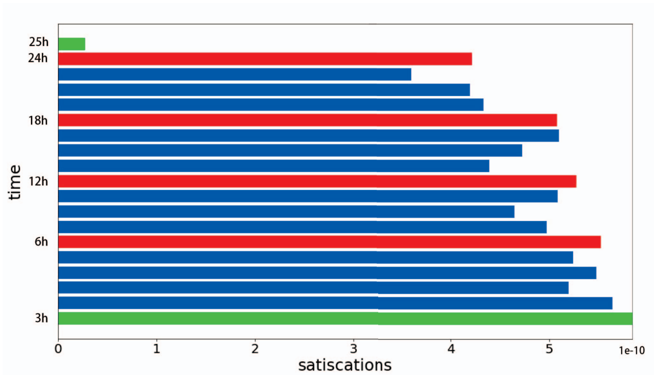


Fig. 6. Satisfaction and logout time correlation statistics.

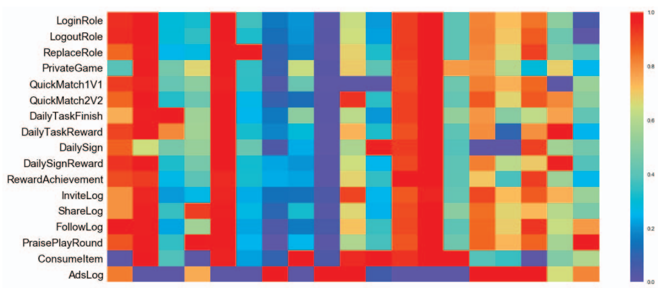


Fig. 7. Colormap of average aspiration to each action.

hours is just effective working time for most people in the morning, afternoon or evening. Whenever the user finishes their work, they will remember to login the game, and it does indicate that they are satisfied with the game and willing to play again and again.

Then we look at the red bars in the middle. The time intervals corresponding to the four red bars from bottom to top are about 6, 12, 18, and 24 hours, respectively. As the value of t^{out} increases, the average user satisfaction wavily decreases. This shows that *when users are not satisfied with the game, they tend to log out for a longer period of time.* And 6, 12, 18, 24 hours and the 3 hours analyzed are just on the peak. That is to say, *when the users' logout time is more consistent with their daily schedule, they are more satisfied with the game.* We also observed that when the logout time reached over 86,400 seconds (over one day), user satisfaction would fall off sharply. In other words, *one day can indeed be used as a criterion for a user's estimation of game satisfaction and the possibility of loss.* Moreover, *if the value of satisfaction is equal to or lower than that of 1-day-average, the user may not return in the short term.*

C. Aspiration Analysis

Each aspiration is a 19-dimension vector. We group the aspirations according to the corresponding behaviour and average the vectors within the group. Thus, we can get a 19-dimensional average aspiration vector for each type of behaviour. For all 19 types of behaviour, we can form a matrix of average aspiration vectors. In order to achieve a better visualization, we normalize the matrix by min-max and then show it through the colormap (see Figure 7).

From Figure 7 we can see that the aspiration of different behaviours is obviously different. The service operators can speculate on what users might do next through their current aspirations.

Then we conducted further analysis on the aspirations. We randomly extracted four users, named BP1, BP2, SP1, and SP2, from the battle players and the social players. In Figure 8, all red lines or points are of BP1. And the blue, yellow and green ones are of BP2, SP1, and SP2, respectively. Through our model, we extracted the aspiration of all the behaviours of the four users. We averaged each user aspiration, get the user's average aspiration and make two radar maps (see Figure 8 (a)&(c)). It can be clearly seen that BP1 and BP2 are very similar in the case that the two users are of the same type (see Figure 8 (a)). This situation also exists between SP1 and SP2 (see Figure 8 (c)). However, BP1/BP2 is very different from the radar map of SP1/SP2 who are not battle players. In other words, *users with similar behavioural habits will have similar aspirations.*

Furthermore, we reduce the tensors of the aspirations to 1 dimension by averaging, and then randomly extract aspiration sequences of the four users along with the corresponding actions (see Figure 8(b),(d)). In Figure 8(d), the sequences randomly taken by BP1 and BP2 are similar (mostly QuickMatch1V1, shown as 4 in Y-axis), their corresponding aspiration sequences are also very similar. So are the SP1 and SP2. In summary, *by observing and analyzing the distribution and trend of aspiration, one can estimate the behaviour distribution and action strategy of a user.*

Also, in Table II, the social group is always easier to predict than the battle group. As you can see from figure 8 (b), the aspiration of the social group is more differentiated.

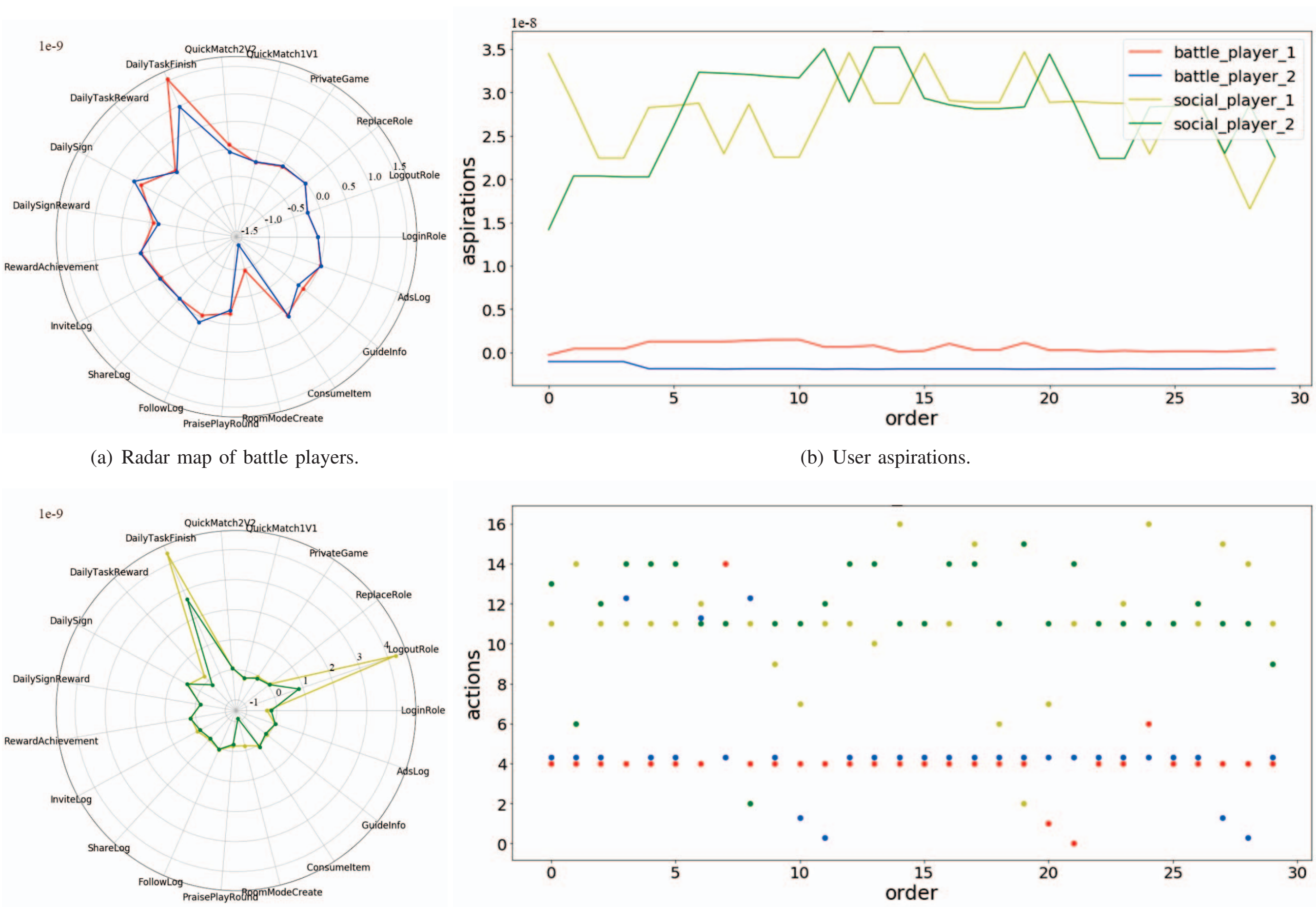
VI. Discussion

In this section, we mainly discuss why satisfaction and aspiration could be obtained by predicting the time sequence through state and action sequence.

Firstly, t^{in} is mainly determined by the user's current aspiration. As shown in Figure 2, the time interval of t^{in} is mainly determined by the two parts: the action part and break part. The action part is mainly related to the type of behaviour. And the break part is mainly determined by the user's current aspiration to play. When the aspiration to play is strong, the user may quickly proceed to the next action, resulting in break part time is very short, vice versa.

Secondly, user satisfaction is mainly stored in user states and reflected by the length of t^{out}. Incompetence, immersion, flow, tension, challenge, etc. are usually used to express satisfaction [27]. Ultimately, these factors will be converted into data stored in the player's state sequence. It often leads to the infinite extension of the t^{out} which could be the ultimate churn, i.e., a user is not satisfied with the game.

Thirdly, user satisfaction and game aspiration will interact with each other and have a sequential relationship. And the



(a) Radar map of battle players.

(b) User aspirations.

(c) Radar map of social players.

(d) User actions.

Fig. 8. Comparison among users on aspiration.

relationship has been well embedded in LaFee's RNN model. When the game satisfaction is high, the players' aspiration to play is easier to maintain at a higher level. Conversely, when game satisfaction is relatively low, players will lose their aspiration to play faster. At the same time, no matter satisfaction or aspiration, the value of the present time slice depends on the last one.

Here is a simple example. When a player plays a game, his satisfaction with the game may increase or decrease. But after a certain length of time in the game, he will certainly logout, that is, the aspiration to play the results of the declines. In the case of more satisfied with the game, after a period of offline rest, the aspiration for the game will be rebounded, leading to the user logs in again for the next round of the game. In general, the offline break time may be longer as the decrease of game satisfaction. More extreme condition is that the game is very unsatisfactory, making user completely lose the aspiration to play this game and never login.

Our method and model are of great value to both academia and industry. On one hand, the regression modeling method proposed can not only remodel the churn prediction problem but also help solve the problem of underfitting of the model using only the behaviour sequences. User churn is a problem for almost all web services, but most services even do not accumulate to the data amount that can effectively analyze the causes of user churn using traditional methods. Our approach helps developers and operators improve their services better and faster by making full use of the existing data in the system.

On the other hand, our LaFee model can estimate user satisfaction and aspiration in real-time while predicting user churn. The service operators can adjust their services by analyzing the performance of satisfactions to provide better service experience. Secondly, the service operators can also use satisfaction to better estimate the user's next possible logout time and dynamically allocate server resources. Finally, the real-time feedback from satisfaction can help the service operators better understand where user satisfaction rises or falls and analyze them. The aspirations can help to segment the user groups and quickly classify users in the condition of the absence of user characteristic data. At the same time, according to the real-time changes of aspiration, the operators can estimate the user's next actions and then provide more targeted service recommendations and advertisement pop-up services.

VII. Conclusion

In this work, we propose a dual-output latent feelings-aware RNN model called LaFee and a regression modeling method to help predict user churn to make full use of online data. We then can estimate users latent feelings and improve the accuracy of user churn prediction by only behaviour data with them. Then we carry out a statistical quantitative analysis of the satisfaction and the aspiration while expounding and proving the physical meaning of them. Finally, the significance of our model and the usefulness of the latent feelings are discussed.

Acknowledgment

This work is supported by the National Key Research and Development Program of China(No.2017YFB1400601), National Natural Science Foundation of China under Grant (No.61825205, No.61772459), National Science and Technology Major Project of China(No.50-D36B02-9002-16/19).

References

[1] Z. Kasiran, Z. Ibrahim, and M. S. M. Ribuan, "Customer churn prediction using recurrent neural network with reinforcement learning algorithm in mobile phone users," *International Journal of Intelligent Information Processing*, vol. 5, no. 1, p. 1, 2014.

[2] A. Hudaib, R. Dannoun, O. Harfoushi, R. Obiedat, and H. Faris, "Hybrid data mining models for predicting customer churn," *International Journal of Communications Network & System Sciences*, vol. 8, no. 5, pp. 91–96, 2015.

[3] E. G. Castro and M. S. G. Tsuzuki, "Churn prediction in online games using players' login records: A frequency analysis approach," *IEEE Transactions on Computational Intelligence & Ai in Games*, vol. 7, no. 3, pp. 255–265, 2015.

[4] G. N. Yannakakis, "How to model and augment player satisfaction: a review," in *First Workshop on Child, Computer and Interaction*, 2008.

[5] B. Cowley, D. Charles, M. Black, and R. Hickey, "Toward an understanding of flow in video games," *Computers in Entertainment (CIE)*, vol. 6, no. 2, p. 20, 2008.

[6] N. Lazzaro, "Why we play games: Four keys to more emotion without story," 2004.

[7] C. Klimmt, C. Blake, P. Vorderer, and C. Roth, "Player performance, satisfaction, and video game enjoyment," in *International Conference on Entertainment Computing*, 2009, pp. 1–12.

[8] J. Wang, J. Yin, S. Deng, Y. Li, C. Pu, Y. Tang, and Z. Luo, "Evaluating user satisfaction with typography designs via mining touch interaction data in mobile reading," in *CHI Conference*, 2018, pp. 1–12.

[9] T. W. AlHanai and M. M. Ghassemi, "Predicting latent narrative mood using audio and physiologic data." in *AAAI*, 2017, pp. 948–954.

[10] K. T. Chen, C. Y. Huang, P. Huang, and C. L. Lei, "Quantifying skype user satisfaction," in *Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, 2006, pp. 399–410.

[11] A. C. Bahnsen, D. Aouada, and B. Ottersten, "A novel cost-sensitive framework for customer churn predictive modeling," *Decision Analytics*, vol. 2, no. 1, p. 5, 2015.

[12] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult." *IEEE Trans Neural Netw*, vol. 5, no. 2, pp. 157–166, 1994.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] F. A. Gers and J. Schmidhuber, "Recurrent nets that time and count," in *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium*, vol. 3. IEEE, 2000, pp. 189–194.

[15] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A clockwork rnn," *Computer Science*, pp. 1863–1871, 2014.

[16] J. Bayer and C. Osendorfer, "Learning stochastic recurrent networks," *arXiv preprint arXiv:1411.7610*, 2014.

[17] N. Kalchbrenner, I. Danihelka, and A. Graves, "Grid long short-term memory," *Computer Science*, 2015.

[18] R. Jozefowicz, W. Zaremba, and I. Sutskever, "An empirical exploration of recurrent network architectures," in *International Conference on International Conference on Machine Learning*, 2015, pp. 2342–2350.

[19] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE Transactions on Neural Networks & Learning Systems*, vol. 28, no. 10, pp. 2222–2232, 2015.

[20] Y. Fan, Y. Qian, F.-L. Xie, and F. K. Soong, "Tts synthesis with bidirectional lstm based recurrent neural networks," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[21] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *Computer Science*, 2014.

[22] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: a recurrent neural network for image generation," *Computer Science*, pp. 1462–1471, 2015.

[23] A. Ando, R. Masumura, H. Kamiyama, S. Kobashikawa, and Y. Aono, "Hierarchical lstms with joint learning for estimating customer satisfaction from contact center calls," in *INTERSPEECH*, 2017, pp. 1716–1720.

[24] A. Liaw, M. Wiener *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[25] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics & Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[26] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. N. Sainath *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal processing magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[27] M. H. Phan, J. R. Keebler, and B. S. Chaparro, "The development and validation of the game user experience satisfaction scale (guess)," *Human factors*, vol. 58, no. 8, pp. 1217–1247, 2016.