User Modeling and Churn Prediction in Over-the-top Media Services

Vineeth Rakesh
Interdigital AI Lab, USA
vineeth.mohan@interdigital.com

Ajith Pudiyavitil*
Lowe's, USA
ajithkp12@gmail.com

Jaideep Chandrashekar
Interdigital AI Lab, USA
jaideep.chandrashekar@interdigital.com

ABSTRACT

We address the problem of customer retention (*churn*) in applications installed on over the top (OTT) streaming devices. In the first part of our work, we analyze various behavioral characteristics of users that drive application usage. By examining a variety of statistical measures, we answer the following questions: (1) *how do users allocate time across various applications?*, (2) *how consistently do users engage with their devices?* and (3) *how likely are dormant users liable to becoming active again?* In the second part, we leverage these insights to design interpretable churn prediction models that learn the latent characteristics of users by prioritizing the specifications of the users. Specifically, we propose the following models: (1) Attention LSTM (ALSTM), where churn prediction is done using a single level of attention by weighting on individual time frames (temporal-level attention) and (2) Neural Churn Prediction Model (NCPM), a more comprehensive model that uses two levels of attentions, one for measuring the temporality of each feature and another to measure the influence across features (feature-level attention). Using a series of experiments, we show that our models provide good churn prediction accuracy with interpretable reasoning. We believe that the data analysis, feature engineering and modeling techniques presented in this work can help organizations better understand the reason behind user churn on OTT devices.

Reference Format:
Vineeth Rakesh, Ajith Pudiyavitil, and Jaideep Chandrashekar. 2020. User Modeling and Churn Prediction in Over-the-top Media Services. In *3rd Workshop on Online Recommender Systems and User Modeling (ORSUM 2020), in conjunction with the 14th ACM Conference on Recommender Systems, September 25th, 2020, Virtual Event, Brazil.*

1 INTRODUCTION

In recent years, users have increasingly taken to consuming streaming video services via applications (i.e. Netflix, Hulu, Youtube, etc.) on so called over the top (OTT) platforms (i.e. AppleTV, Roku, Amazon FireTV, etc.). Given the very large (and still growing) number of streaming services, there is fierce competition to attract new customers, while maintaining customer satisfaction. Unfortunately, there is significant cost to attracting new users; thus, service providers are very invested in retaining end-users and keeping them engaged with their products. These customer retention efforts focus on providing exclusive and engaging content, personalized recommendations and intuitive user interfaces. When such efforts fail, the operators experience customer *churn*, wherein a subscriber stops using the service. By examining high level data collected on one such OTT hardware platform, we propose feature engineering techniques for modeling user behavior and leverage these features to develop *application-level churn prediction models*. Specifically, given a user u who installs an application a at a given time on their OTT device, our model predicts whether u will be engaged (or not engaged) with a after a particular time window.

Users may decide to abandon a streaming service for any number of reasons such as a limited time budget to consume content, an increasing affinity for a different application or a lack of compelling new content. Yet another reason may be that the user experiences more hardware faults (i.e. reboots, poor wifi, high memory usage, etc.) when a particular service is being used. In such cases, the user perceives these faults as being caused by the application and quits out of frustration. The key observation here is that both *application-level* and *device-level* behavior can influence user churn. Consequently, it is critical to model these heterogeneous factors along with temporally correlated features reflecting usage of different applications to accurately predict churn. To achieve this, we first analyze a dataset that captures high level *events* on OTT devices and examine the signals of application churn. These events could be user-initiated (i.e. opening or closing an application, restarting the device, putting the device to sleep, etc.) or device-specific events (i.e. automatic reboots, wifi drops, software/firmware updates, etc.) With this data, we examine questions such as: (1) *how users allocate time across applications on their OTT device*, (2) *how often and for how long users engage with the device (specific applications), and* (3) *how long do users go dormant, and how likely is it for a dormant user to become reactive.* In the second part of our work, we leverage these statistical insights to design interpretable models that are effective in predicting churn across a wide range of scenarios.

The naive approach to building a model that predicts churn would be to fix an observation time window T, extract a number of features of interest $|\boldsymbol{m}|$ from this window and then deploy a suitable classification algorithm that predicts whether a subscriber will quit a service after a period of time T. While this is entirely viable, there are two important drawbacks. First, the data is inherently noisy and high-dimensional; OTT devices send out periodic device-level and application level summaries (the start time and duration of an application session), and events observed on the box. This results in a feature space of size $T \times |\boldsymbol{m}|$. Second, there is significant inherent temporal correlation in the data; if a user spends a significant amount of time inside an application on successive days, there is a strong signal that he/she will engage with the same application on the next day. Flattening data over the entire window T into a single representation vector will lead to this information being lost. An alternative, more principled approach, is to learn latent attributes

*This work was done when the author was at Interdigital AI Lab.

ORSUM@ACM RecSys 2020, September 25th, 2020, Virtual Event, Brazil

of the data using time-series models such as recurrent neural networks (RNN) [3] and use them as features for churn prediction. A potential issue with this approach is that the compressed latent vector is inefficient at capturing all the necessary information that leads to churn. Furthermore, it is extremely difficult to interpret the results of vanilla RNNs.

In this paper, we propose a model that address the drawbacks of the more conventional approaches. First, we introduce Attention LSTM (ALSTM), where we modify the neural machine translation (NMT) model [2] for churn prediction; ALSTM models the local attention. Second, we propose Neural Churn Prediction Model (NCPM), which incorporates two levels of attention i.e., local and global. ALSTM uses temporal-level attention (local attention) where different (sub) observation windows contribute different weights towards predicting churn. For example, in a particular week we might observe the subscriber slowly starting to watch more and more content on a new application, and spending less and less time in another in which they were previously engaged (and eventually abandon). Thus, features collected in this week might require higher priority when compared to other time frames. NCPM on the other hand is a more comprehensive model that captures each feature using a separate ALSTM. The individual ALSTMs are then combined with a feature-level attention layer (global attention). Global attentions are much better at prioritizing weights across different temporal features. For example, churning could be more influenced by device-level issues such as periodic reboots rather than application engagement. Although attention-based RNNs have been extensively used in the field of natural language processing [7, 25], they have very rarely been applied to the problem of churn prediction. To our knowledge, the only work that appears to address this area is [26]; however, the attention mechanism used in their work is quite different than ours. We summarize the major contributions of our work as follows:

- **Understanding user behavior**: Through data engineering and statistical analysis, we provide several insights that explain the behavior of users in our OTT dataset.
- **Predicting Churn**: We propose attention-based RNN models that learn the characteristics of users in a weighted low-dimensional latent space.
- **State-of-the-art performance**: By conducting extensive experiments on a real-world dataset, we show that NCPM outperforms all other models over different test cases and achieves an accuracy of upto 89% and AUC of 92%. Additionally, NCPM interprets the reason for churning by emphasizing on features such as inter-arrival time between the apps and consistency in app usage.

We begin by introducing our dataset in Section 2 and in Section 3, we model the behavior of OTT customers as observed in this dataset. The churn prediction models ALSTM and NCPM are proposed in Section 4 followed by the results of our experiments in Section 5. Finally, we review related work in Section 6 and conclude our paper in Section 7.

2 DATASET

Our dataset consists of high level application session data (start time and durations) and device level events that spans from Sept 2018 to April 2019. The data was collected from a sample of 31k AndroidTV based OTT devices deployed in homes and operated by a large provider. Each of these devices comes pre-installed with a set of applications; apart from these, users (u) can download others from a large catalog on the app store. We observed over 3.7k distinct applications being used in the dataset. However, some devices and applications are used very sporadically and account for very little data. To remove these, we pre-processed the data to filter out devices that were active for less than 90 days in total, and removed applications that were used on fewer than 15 devices in our population. This filtering resulted in 14,082 unique users (i.e., OTT devices) [1] and 462 unique apps; this dataset is denoted by $\mathcal{D}$. Note that the churn prediction model builds features over the lifetime of the application on the device; this *starts from the time the application was installed, to when the user is deemed to have abandoned it.* Unfortunately, for the pre-installed applications such as Netflix, YouTube and SlingTV *there is no install date.* One can exclude such apps from the data; however, this leads to removing a significant number of users. This is because a *large proportion of users tend to confine themselves to using the pre-installed apps* (which also happens to be the popular streaming services). Alternatively, having all users in a single bin could lead to some serious bias in our modeling since for default apps, we do not have a clear signal on when it was downloaded. The user might have been using the app well before the onset of our study. Therefore, besides $\mathcal{D}$, we create a separate dataset $\mathcal{D}_e$ that completely excludes the default apps, while for $(u, a) \in \mathcal{D}$ that do not have a start date, *we simply take the first log entry of a by u as a proxy for the actual install date.* The subset $\mathcal{D}_e$ covers 8,223 unique devices using 397 distinct applications.

Clearly, the earlier a service provider is able to predict churn (of u), the more effectively they can take action and address the underlying reasons as to why a customer might be departing. Consequently, we divide D and D_e into different days of activities A, where $A = \{t | t \leq T, T \in \{5, 10, 15, 20\}\}$. For instance, $T = 5$ means we consider a maximum of five days of user activity to predict the churn. Table 1 shows the characteristics of dataset $\mathcal{D}$ and $\mathcal{D}_e$ across different activity days. In the upcoming section, we explain the methodology of determining the churners and non-churners (i.e., columns four and five in Table 1). Here, one can see that as T increases, the number of users decrease. This is because the number of users that continuously use the OTT for say 20 days is far less than than those who use for just 5 days.

3 ANALYZING USER CHARACTERISTICS

App and Device Usage: Figure 1 (a) shows the ten most popular apps seen in our dataset, based on the number of users that regularly use them. Sling Tv, Netflix, Youtube and Google games occupy the top four spots. Figure 1 (b) plots the distribution of daily time spent inside each of the applications. We find that users spend 3-4 hours on average with the OTT device, with a very small fraction of users that spend more than 8 hours. In fig. 1 (c), we break this daily spend into four different parts of a day, corrsponding to morning (5am-12pm), afternoon (12pm-5pm), evening (5pm-10pm) and night (10pm-5am). Unsurprisingly, we observe that users tend to spend more time in the evening when compared to other periods of a

[1] we use the words *devices* and *users* interchangeably

	Dataset $\mathcal{D}$				Dataset $\mathcal{D}_e$			
T	#Users	#Apps	#Churns	#NonChurns	#Users	#Apps	#Churns	#NonChurns
5	13082	402	9017	16044	6390	283	8712	9421
10	7954	347	6339	8596	5500	234	4661	7538
15	5440	256	4123	5932	4929	194	3088	6423
20	5009	179	3193	4067	4453	165	2231	5548

Table 1: Statistics of the churn prediction datasets $\mathcal{D}$ and $\mathcal{D}_e$ for different range of activity days T. For example, T = 5 implies for a given user-app (u, a) tuple, u actively used a upto 5 days before churning.

day (median of 2.2 hrs). However, it is not significantly higher than afternoon, which has a median of 1.8 hrs and morning with a median of 1.4 hrs. Another key statistic of interest is the inter arrival time between application sessions. Note that there may not be an explicit indicator of a user quitting an application. Very often, users just stop using the application that is installed, or deactivate their accounts but keep the application installed. Thus, churn must be detected implicitly, i.e., by the fact of the application not being started by the user for a sufficiently long time. We calculate the arrival time between successive start times of an application session on a device (across all applications) and plot the maximum values, across all the devices, in fig. 1 (d). We observe that users, after an absence, return to the OTT applications within a median time of 6 days (100-200 hours). The 75%-ile value of this distribution is about 10 days. Later in this section, we leverage this to establish an inactivity threshold when we define churn more precisely.

User Engagement Patterns: Here we try to understand how users spend time on their OTT device. We wish to explore the following aspects: (a) Are there users who consistently use the box for same number of hours every day? (b) Are there dormant users who don't use their device for a while, but then reactivate it? (c) Are there users that engage with their device, but only intermittently and for brief periods of time? To answer these, we carry out the following analysis. First, for each day that our dataset spans, we compute the cumulative time (in terms of cummulative distribution function - CDF) that the user engaged with the device. Specifically, we compute (i, c_i) for each user, where $i = 1, 2, .., |\mathcal{D}|$ represents each day in our dataset, and c_i is the total number of hours spent on the device upto day i. Next, we carry out a non-linear fit on this data for each user, recording the learned slope, intercept and standard error as derived features for each user. Subsequently, we cluster the derived features using K-means; the number of clusters is determined based on the silhouette score [23]. This analysis yields four main behavior patterns that cover the vast majority of users, and are depicted in fig. 2. Each plot is based on the original data of cumulative device engagement time (x-axis is days elapsed, y-axis is cumulative time spent). These four patterns can be labeled as follows: (1) *mid bloomers* (a), these are users who are initially silent and do not use the OTT box heavily, but then suddenly start using during the middle phase of their total period. (2) *late bloomers* (b), these users remain dormant for a longer duration with minimal activity, but suddenly start using the device towards the end. (3) *potential churners* (c), these are users who are of interest to us. As explained by the plot, these users start using the device heavily at first, but then stop using the box for various reasons. Please note that since y-axis is the CDF, the flat line here indicates minimal or no activity (also indicated by very low standard deviation, since there is no activity). (4) *consistent users* (d), finally, these are users who are consistent and regularly use their OTT boxes to watch different shows.

Understanding Churning Behavior: Before introducing our prediction models, we briefly explain how we label a user (or device) having churned, i.e., left the service. This is fundamentally a difficult task because there is no explicit signal for this behavior. Further complicating things, (a) some users don't use the app for a few days, but return back after a brief period of inactivity, and (b) some users simply download the app once (or spend a brief amount of time in it) and never use it again. Figure 3 depics, at a high level, all the information for a user (u) application (a). The dotted lines at either end capture the time data was collected and each of the green vertical lines in the middle indicate the start of application sessions (a_1 is the first session, a_n is the last). Here, we see that the application was downloaded after the start of the data collection and used several times, the last instance is at t_3. Somewhat infrequently, we see the device itself *disappears* from the dataset; we consider this a signal that the user has disconnected the device and is no longer using it. In this scenario, we capture this event having occured at t_4. With this depiction, we can now define churn in very specific terms by addressing the two challenges previously discussed. First, we require that the application not be used for a period of time after the last use. Following the example in fig. 3, we impose the condition $\Delta_2 \geq T_{3q}$. Here T_{3q} – an *inactivity threshold* – is the 3rd quartile of the distribution in fig. 1 (d) and turns out to be 10 days. Second, we require a minimum number of sessions to be recorded for an application and user. Specifically, u should have engaged with a at least K times, i.e., $n \geq K$ and we set $K = 3$. Figure 4 illustrates the characteristics of devices that are *exclusively labeled as churn*. We see that the median inter-arrival times for the top 5 apps is around 30 days (Figure 4 (b)), which is significantly higher than the generic inter-arrival characteristics shown in Figure 1 (d). In figure 4 (c) we notice that users who download more apps tend to have higher churning rate, we obtained a Pearson correlation coefficient of 0.67. It is also interesting to observe that as the churn increases, users tend to switch between apps more frequently, where the app switch is indicated by the session feature (y-axis).

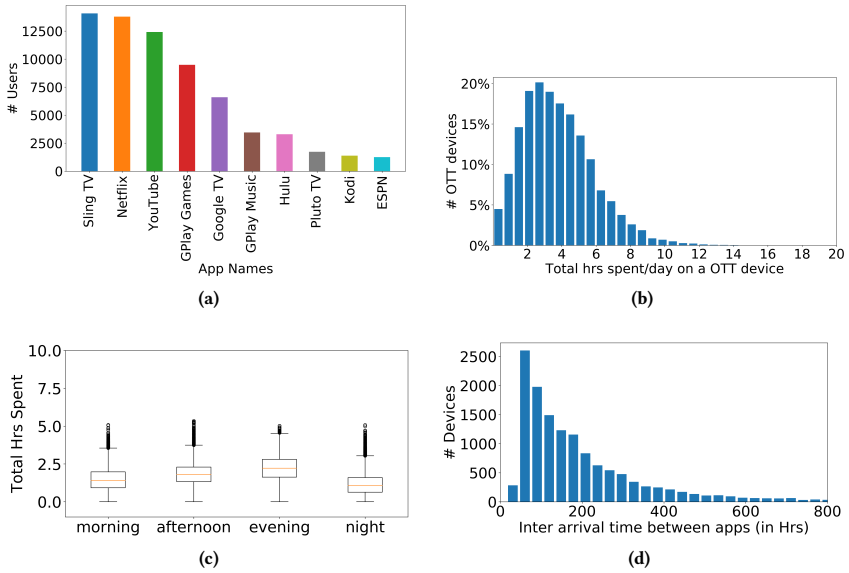


Figure 1: Generic characteristics of users in $\mathcal{D}$ (a) shows the top 10 most frequently used apps, (b) majority of the users spend 2-6hrs per day, (c) users tend to spend more time in the evening and (d) majority of the users tend to return back to their OTT device within a maximum of 8-9 days ($\approx$ 200 hrs)

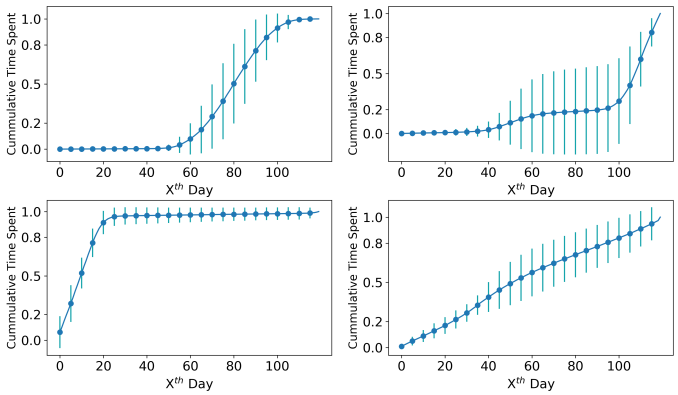


Figure 2: The four types of users captured by our clustering framework. Clockwise from top left, (a) mid-bloomers, (b) late-bloomers, (c) potential-churners and (d) consistent users. The error bars show the standard deviation based on usage.

4 PREDICTING CHURN

Given a user u and an app a, our objective is to predict if u will continue or stop using a. We realize each user-app entity as a tuple (X, M_x, Y) where $X = \{x_1, ..., x_t\}$ is a stream of events (or logs) that spans a time $t \in T$. Each event x comprises of M features, and $Y = \{y_1, ..., y_t\}$ are the binary labels that indicate churn (or non-churn) at t. When designing our churn prediction model we had two main objectives. First, since our data is highly temporal, it is important to learn the latent characteristics of churners (and non-churners) in such a way that it embeds the temporality of events. Second, not only should we predict the churn with good accuracy, but also produce highly interpretable results. In other words, we should reason out as to why a user is churning. To achieve this, we propose the following models: (1) attention LSTM (ALSTM), which is a simple modification of the neural machine translation (NMT) model [2] and (2) neural churn prediction model (NCPM): a more comprehensive model that incorporates temporal-level attention (or local attention) and feature-level attention (or global attention). Both the models are based on recurrent neural network (RNNs) that have shown to be effective in modeling time-series data [5, 8]. RNNs take a series of temporally dependent inputs and learn their

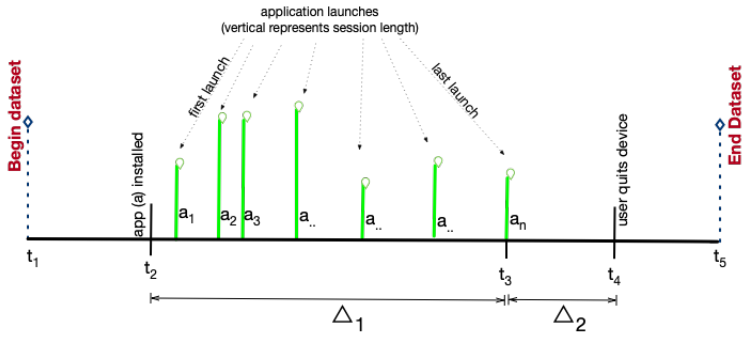


Figure 3: High level summary of user u interacting with application a

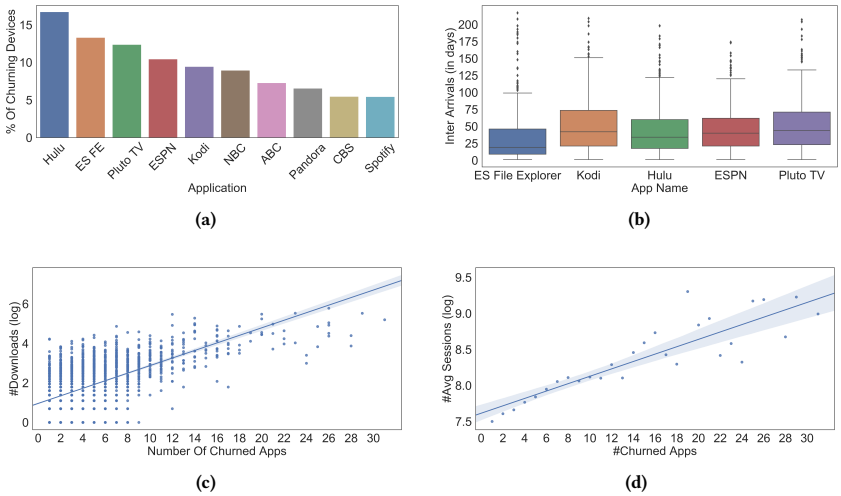


Figure 4: Characteristics of churning OTT devices (a) Most frequently churned applications for the dataset D_e (b) the median inter-arrival is about 25-30 days for top-5 churning apps, (c) users tend to churn more when they download more apps and (d) users who churn for more apps tend to switch between apps more frequently.

latent representation (or hidden state vector) using the following expression:

$$h_t = f(h_{t-1}, x_t) \tag{1}$$

where h_t is the hidden layer at time t and f is some non linear function. For our application, we model f using long short-term memory network (LSTM) [13]. LSTM has four states that are defined as follows:

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}; x_t] + b_i) \\ f_t &= \sigma(W_f \cdot [h_{t-1}; x_t] + b_f) \\ c_t &= f_t \times c_{t-1} + i_t \times tanh(W_c \cdot [h_{t-1}; x_t] + b_c) \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ h_t &= o_t \times tanh(c_t) \end{aligned} \tag{2}$$

where t is the time step (i.e, days) , h_t is the hidden state at t , c_t is the cell state at t , x_t is the hidden state of the previous layer at time t, i_t , f_t , o_t are the input, forget and out gates, respectively.

Attention LSTM (ALSTM): Obviously one can predict churn by simply providing the input X to a vanilla LSTM, get the latent vectors h from the final layer, and use them as features for prediction. A potential issue with this approach is that the compressed (or low dimensional) latent vector h is inefficient in capturing all the necessary information that attributes to churn. As explained in Section 1, in a particular week we might observe the subscriber slowly starting to navigate towards a new application and spend less and less time in one that they were previously engaged with (and eventually abandon). So, it is important to give high priority to these time windows when compared to other weeks. Modeling churn using vanilla LSTM networks fails to prioritize such key events. Inspired by recent developments in neural machine translation (NMT) [2], we incorporate attentions into LSTM to overcome this issue. Since our application is very different from natural language processing, we introduce two modifications over NMT. First, we replace the decoder part with a single layer neural network (NN) with sigmoid activation for churn prediction and second, we change

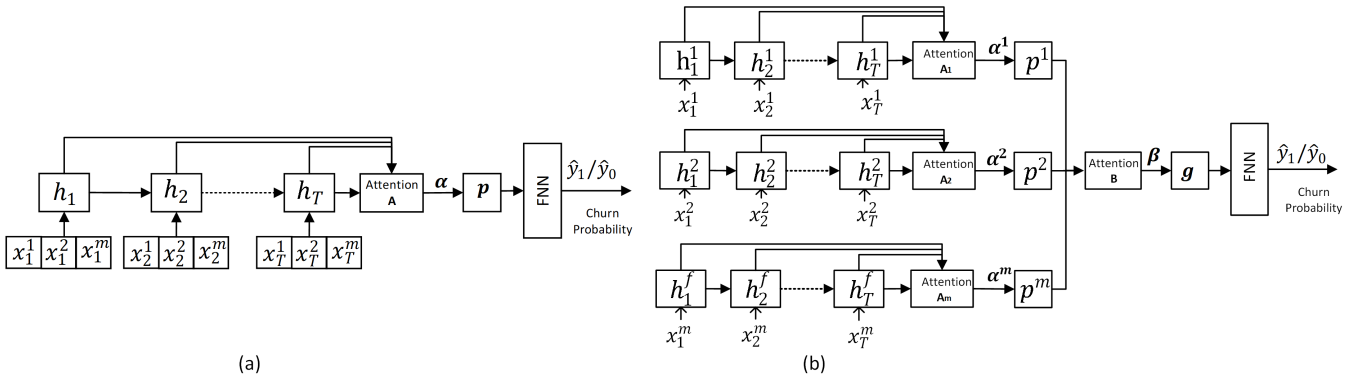


Figure 5: The end-to-end architecture of the proposed models (a) ALSTM that uses attentions on a single LSTM network to model churn (b) shows NCPM that uses separate LSTM networks for modeling individual features and predicts the churn using weighted attentions.

the attention mechanism to suit our problem. The proposed ALSTM model is shown in Figure 5 (a). Here, the attention block A outputs a vector of weights α that emphasizes the importance of the latent vector h for a given time frame t. The weighted latent vector p is defined as follows:

$$p = \sum_{t=1}^{T} \alpha_t h_t \tag{3}$$

where the weight α_j for a time instance t is defined by

$$\alpha_j = \frac{exp(s_j)}{\sum_{t=1}^{T} exp(s_t)} \tag{4}$$
$$s_j = \sum_{k=1}^{K} h_k^t \cdot W_{kj}^t$$

Neural churn prediction model (NCPM): One drawback of ALSTM is that it is unable to prioritize across features. For example, churning could be more influenced by the consistency of users (see Figure 2), while the number of downloads might have little importance. To overcome this problem, we incorporate both temporal-level attention and feature-level attention. As depicted in Figure 5, instead of treating the features as a single vector, we decouple the features and model them using individual ALSTMs. Similar to ALSTM, the attention block p^m of an a feature m captures the influence (or weight) of the latent features from different slices of time. On the other hand, the feature-level attention is capture by the block B, which is defined by the following expressions:

$$g = \sum_{m=1}^{M} \beta^m p^m \tag{5}$$

In the above expression, β denotes the individual attention weights that is defined as follows:

$$\beta^m = \frac{exp(c^m)}{\sum_{m=1}^{M} exp(c^m)} \tag{6}$$

$$c_j^m = \sum_{i=1}^{Z} z_i U_{ij} \tag{7}$$

where $z = p^1 \oplus \{p^i\}_2^m$ is the concatenation (indicated by $\oplus$) of the feature-level latent vectors . Finally, to predict the churn, a linear projection with a sigmoid function is connected to the output of the last layer to produce user churn prediction as follows:

$$\hat{y} = \sigma(W_g \cdot g + b_g) \tag{8}$$

The loss for both ALSTM and NCPM is computed using binary cross entropy, that is defined as follows:

$$\mathcal{L} = \sum_{i} -y_i log(\hat{y}_i) - (1 - y_i) log(1 - \hat{y}_i) \tag{9}$$

5 EXPERIMENTS

Obviously, from Table 1, one can notice that our dataset is biased towards negative samples (i.e., #non-churns). Therefore, to create a balanced dataset, for every positive data point (i.e, churns) for an app a, we randomly sample a corresponding negative data point. We test our models by varying the number of days in the training sample (explained in Section 2). This helps us to see how quickly can our models predict the churn. For all our experiments, we use 10 fold cross validation, where eight folds are used for training, one for validation, and one for testing. The deep learning models are implemented using Keras with Tensorflow as the back-end.

5.1 Baselines

We compare the performance of the proposed ALSTM and NCPM with three baseline methods. Unlike the proposed models (i.e., ALSTM and NCPM) the following baselines do not capture the temporality in data. Therefore, the inputs to these model are flattened vectors across the time frames.

Logistic Regression: the classic model for binary classification problem. Albeit simplistic, it helps us to understand if a linear decision boundary is sufficient to capture the churners and the non-churners. We use L2 norm as the regularizer and Stochastic Average Gradient (SAG) as the solver.
Multi-layer Perceptron (MLP): We consider a simple two layer neural network and a dropout layer to avoid overfitting. A linear projection with a sigmoid function is connected to the output of the last layer to produce user churn. Similar to logistic regression, MLP does not capture the temporal dependencies between the data. The number of neurons are set to 80 for each intermediate layer.
Random Forest (RF): Despite the rapid advancements in the field of deep learning, ensemble techniques such as RF [4] remain highly competitive in producing excellent results on data with several modalities. In our experiments, the number of decision trees are set as 50 and the maximum depth as 10.

5.2 Results

Classification accuracy: Tables 2-3 shows that NCPM outperforms all other models for both datasets $\mathcal{D}_e$ and $\mathcal{D}$, achieving an accuracy of upto 92%. As we increase the number of days the accuracy increases for all models (except logistic regression). Here, CA-5 implies the classification accuracy with just 5 days of data, while CA-20 implies 20 days of data. We can also see that the proposed ALSTM is not as good as NCPM which proves the following: (1) it is important to learn the latent attributes of each individual features separately and (2) incorporating both global and local attention is necessary. That being said, ALSTM clearly outperforms MLP, which emphasizes the necessity of learning the temporal actions of OTT users. The worst performing model is the logistic regression, which is just slightly better than a random selection. This illustrates the difficulty of our churn prediction task. The performance of RF is very close to that of ALSTM, which indicates that ensemble models are still a strong candidate for our problem.

In general, the performance of models over non-continuous data is much better than its continuous counterpart, this can be explained using the following example. let us say that u uses an app for six days before churning and we have the following data for u $\{m_1, m_4, m_6, m_7, m_8, m_{10}\}$, where m is some feature and the suffix indicates the day. Our objective is to predict the outcome on sixth day, using the first five days. Since the user does not use the OTT box for days two, three, and five, the continuous data that is fed to our models (i.e., both NCPM and ALSTM) is essentially a sparse vector $\{m_1, 0, 0, m_4, 0\}$, which has several missing values. On the contrary, for non-continuous dataset, we will have the actual usage values for five days. This obviously means that the model gets to train with more observed data points, which leads to better prediction accuracy. Another interesting observation is that the performance of all models (except logistic regression) is noticeably better on the all-apps dataset. One key reason for this outcome is the popularity of the default apps. Apps such as Sling TV, Netflix and Youtube are significantly popular than other non-default apps. Therefore, the models are able to effectively learn the churn patterns for such apps more effectively.
AUC and ROC characteristics: Figures 6 and 7 compare the ROC characteristics of the proposed models with other baselines. The corresponding AUC values are listed in Table 4, due to space constraints, only the non-default case is furnished. Similar to the accuracy scores, for most scenarios, NCPM remains dominant over other models. We also notice that RF tends to perform better than NCPM and ALSTM when the temporal length of data is low (i.e., with just five days). However, as we incorporate more days for training, there is a clear increase in the performance of our models. The outcome for dataset $\mathcal{D}$ is much different than $\mathcal{D}_e$ where we are able to achieve an AUC of almost 89% with just five days of data; additionally, ALSTM seems to perform very similar to NCPM.
Interpreting the churn prediction: One of the key strengths of our model is interpretability. As explained in Section 1, ALSTM provides single level of interpretability, which indicates which days are important when predicting churn. NCPM on the other hand, has two levels; besides telling the important days, it also tells us which features are important. We present the interpretability scores as heatmaps in Figure 8. Due to the lack of space, we only furnish the results of non-continuous dataset. Heat maps (a)-(d) explains that the influence of features are not uniform across apps; for instance, when we have less days for prediction, the churn is influenced by two main attributes namely, the number of downloads and the cluster types (Figures 8 (a) and (c)). As we incorporate more data for training (i.e., the number of days) the attention tends to get more focused towards a few key features. For non-default apps, there seems to be more attention on the inter-arrival time, while for all-apps the influence seems to be more towards the number of reboots. It could be possible that these apps experience a higher number of app crashes, which could lead to user rebooting the device. For non-default apps, Hulu, Plot Tv and Kodi is heavily influenced by the cluster id feature that we engineered in Section 3. When it comes to temporal attention (Figures 8 (e)-(h)), for dataset D, the influence is mainly concentrated on a few selective days, i.e., day 4 for non-continuous case, and day 2 for continuous case. Contrary to this, for D_e this influence is spread across almost all days.

In Section 3 we explained that the engagement pattern of users could have a strong impact on churn. To show this effect, for each user, we get the final attention score from individual RNNs of NCPM and plot the outcome in Figure 9 (a). Here, we can see that consistent users are the highest indicators of non-churn, while potential churners are the highest indicators of churn. Interestingly, mid bloomers seem to have higher attention over late bloomers when it comes to predicting non-churners, while the opposite is true for churners. Figure 9 (b) provides a more deeper look into this outcome by emphasizing on the importance of temporal progression on the user types. Unsurprisingly, during the initial phase (elapsed duration of 10-20%) almost all types have less attention weights. This is because, during the early phase, we do not have enough data about the user type. As the time progresses, around 20-50% of the elapsed duration, we see that potential churners have the strongest impact on the outcome followed by consistent users and late bloomers. Around 50-80% , the impact of potential churners drastically reduces, while mid and late bloomers increase. At the final stage (i.e., 80-100%) almost all user types have less importance (or attention). This is because, during the last phase, there is more available data in the form of other features such as number of

(a)

Model	CA-5	CA-10	CA-15	CA-20
Logistic	0.56	0.55	0.54	0.54
RF	0.65	0.73	0.79	0.80
MLP	0.6	0.66	0.72	0.78
ALSTM	0.65	0.77	0.83	0.86
NCPM	**0.67**	**0.79**	**0.84**	**0.88**

(b)

Model	CA-5	CA-10	CA-15	CA-20
Logistic	0.56	0.55	0.54	0.54
RF	0.66	0.70	0.73	0.75
MLP	0.58	0.63	0.70	0.74
ALSTM	0.60	0.67	0.74	0.79
NCPM	0.62	0.70	**0.76**	**0.81**

Table 2: Classification accuracy (CA) for (a) non-default and non-continuous data across 5-20 days and (b) non-default and continuous data across 5-20 days.

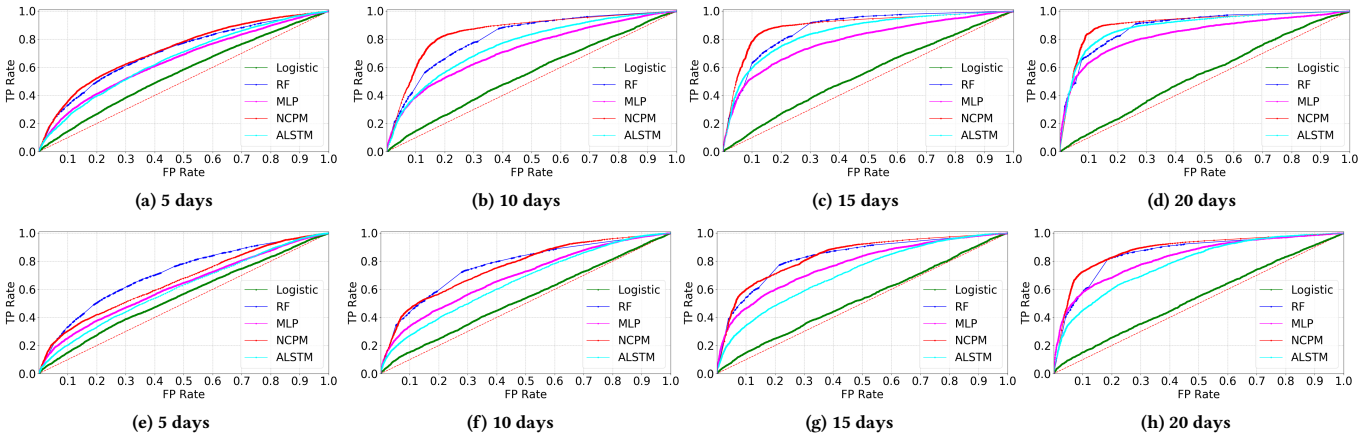
(a)

Model	CA-5	CA-10	CA-15	CA-20
Logistic	0.56	0.55	0.54	0.54
RF	0.73	0.78	0.85	0.89
MLP	0.7	0.72	0.73	0.74
ALSTM	0.78	0.83	0.87	0.91
NCPM	**0.78**	**0.84**	**0.89**	**0.92**

(b)

Model	CA-5	CA-10	CA-15	CA-20
Logistic	0.56	0.55	0.54	0.54
RF	0.72	0.78	0.78	0.83
MLP	0.62	0.70	0.71	0.74
ALSTM	0.74	0.79	0.83	0.86
NCPM	**0.76**	**0.82**	**0.84**	**0.89**

Table 3: Classification accuracy (CA) for (a) all apps and non-continuous data across 5-20 days and (b) all apps and continuous data across 5-20 days.



(a) 5 days (b) 10 days (c) 15 days (d) 20 days

(e) 5 days (f) 10 days (g) 15 days (h) 20 days

Figure 6: Receiver operating characteristic curve (ROC) of the proposed ALSTM and NCPM model along for non-default apps. Curves (a)-(d) represent the ROC for non-continuous dataset and (e)-(h) represents the continuous dataset.

downloads and inter-arrival time between apps. Consequentially, the model is able to rely on better indicators at the later stage.

6 RELATED WORK

The problem tackled in this paper is related to the following topics: (1) churn prediction (2) user behavior modeling and (3) interpretable neural networks. We now detail some existing research that correspond to these topics.

Churn Prediction: User retention (or churn) has been extensively studied in the field of social computing and human computer interaction (HCI) [9, 14, 27]. However, developing predictive models for churn is still at infancy. Au et. al. [1] adopt a rule based learning technique for early churn prediction. In [29], the authors tackle the problem of churn prediction in mobile apps. They find that application performance such as energy consumption and latency have a significant impact on retention. [16] use a social influence based approach for churn prediction. Recently, [26] develop an interpretable framework that constraints the objective of RNN with the outcome of K-means clustering to predict the retention of users in Snap Chat.

Modeling user behavior: There are numerous research on behavior modeling [10, 15, 21]. For example, [6] predict user intents

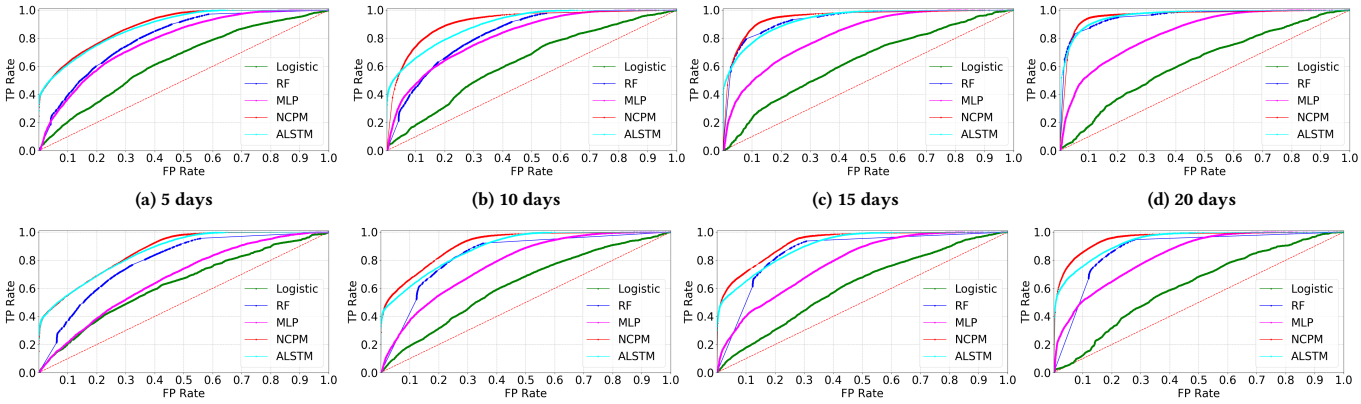
(a)

Model	AUC-5	AUC-10	AUC-15	AUC-20
Logistic	0.56	0.55	0.55	0.55
RF	0.71	0.81	0.87	0.89
MLP	0.65	0.72	0.79	0.84
ALSTM	0.66	0.76	0.85	0.89
NCPM	**0.72**	**0.85**	**0.9**	**0.91**

(b)

Model	AUC-5	AUC-10	AUC-15	AUC-20
Logistic	0.55	0.53	0.53	0.54
RF	0.71	0.77	0.83	0.86
MLP	0.62	0.69	0.77	0.83
ALSTM	0.6	0.66	0.71	0.78
NCPM	0.66	0.76	**0.84**	**0.88**

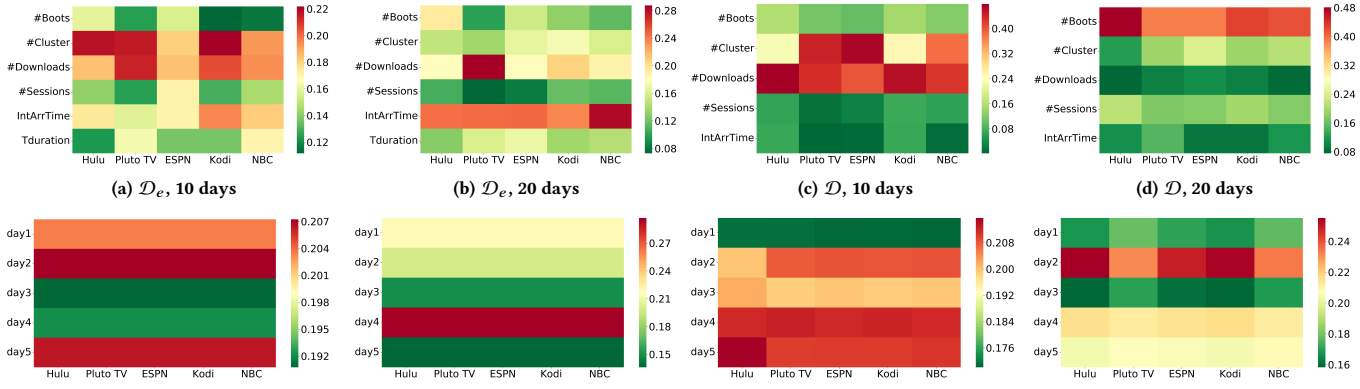
Table 4: Area under the ROC curve (AUC) for (a) non-default apps and non-continuous data across 5-20 days and (b) non-default apps and continuous data across 5-20 days.



(a) 5 days (b) 10 days (c) 15 days (d) 20 days

(e) 5 days (f) 10 days (g) 15 days (h) 20 days

Figure 7: Receiver operating characteristic curve (ROC) of the proposed ALSTM and NCPM model along for all apps. Curves (a)-(d) represent the ROC for non-continuous dataset and (e)-(h) represents the continuous dataset.



(a) $\mathcal{D}_e$, 10 days (b) $\mathcal{D}_e$, 20 days (c) $\mathcal{D}$, 10 days (d) $\mathcal{D}$, 20 days

(e) $\mathcal{D}_e$, 10 days (f) $\mathcal{D}$, 20 days (g) $\mathcal{D}_e$, 10 days (h) $\mathcal{D}$, 20 days

Figure 8: The feature-level (a-d) and the temporal-level (e-h) influences on churn prediction for non-continuous dataset. The gradient of colors denote the proabability scores, where red denotes the highest weight and green denotes the lowest influence.

by leveraging the activity logs in Pinterest. [12] predict the likelihood of a successful search in web search queries. They show that user behavior are more predictive of goal success than those using document relevance.[22] model the behavior of users in the Kickstarter crowdfunding domain using a heterogeneous combination of social communities, popularity of projects and the impact of reward categories. Studies such as [11, 24] and [21] model user behavior from sequencial actions such as click streams and social

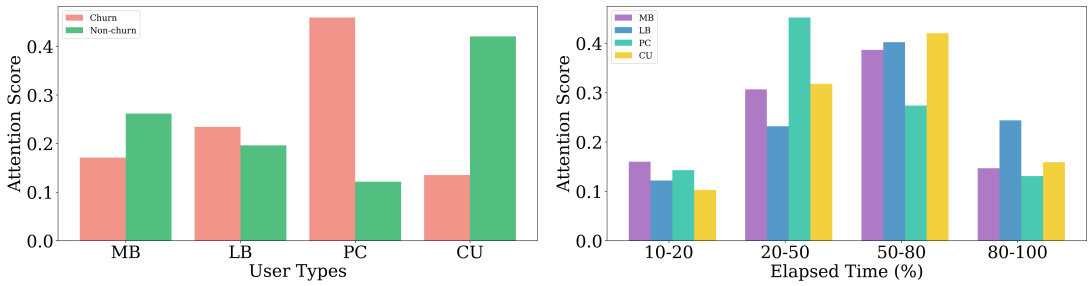


Figure 9: The attention weights of mid bloomers (MB), late bloomers (LB), potential churners (PC), and consistent users (CU): (a) indicates the overall attention-scores during the final phase of prediction and (b) indicates the attention scores at different stages of temporal progression.

network activities. [24] use a combination of Mahalanobis distance (for detecting outlines) and Markov Chains to model sessions in click streams, while [21] use a temporal LDA based approach for tour recommendation in Foursquare.

Interpretable Sequence Modeling: RNNs have become the state-of-the-art technique for sequential modeling [5, 13]. Albeit a plethora of research in the NLP domain [2, 19], extending interpretable RNNs for other real world applications is still an emerging field of research. In a recent work, [18] predict the engagement of users in the Snap Chat app by capturing the in-app action transition patterns as a temporally evolving action graph. [17] develop an interpretable LSTM to learn multi-level graph structures in a progressive and stochastic manner. [20] propose a dual stage attention model for medical diagnostics such as heart failure prediction. Zhou et. al. [28] propose an attention-based RNN that predicts the purchase probability of users for targeted ads. Albeit having a similar NN architecture as ours, their problem is quite different than churn prediction. Additionally, they modeling of local and global attention is quite different than ours. To the best of our knowledge, the only research that closely resembles our work is the churn prediction model proposed by Yang et. al. [26]. However, the attention mechanism used in their work is quite different than ours.

7 CONCLUSION

In this paper we proposed interpretable recurrent neural network based models for prediction churn in over the top media (OTT) devices. In the first part of the paper, we analyzed the behavioral characteristics of users and found that they can be categorized into four main types: mid bloomer, late bloomers, potential churners and consistent users. In the second part, we introduced two models for churn prediction, namely Attention LSTM (ALSTM) and Neural Churn Prediction Model (NCPM). In ALSTM, the prediction of churn was done by weighting on individual time frames (temporal-level attention) and (2) NCPM, we used two levels of attentions namely, feature-level and temporal-level. We showed that NCPM outperforms all other models over a wide range of test cases and achieves an accuracy of upto 89% and AUC of 92%.

REFERENCES

[1] Wai-Ho Au, Keith CC Chan, and Xin Yao. 2003. A novel evolutionary data mining algorithm with applications to churn prediction. *IEEE transactions on evolutionary computation* 7, 6 (2003), 532–545.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[3] Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks* 5, 2 (1994), 157–166.

[4] Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.

[5] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8, 1 (2018), 6085.

[6] Justin Cheng, Caroline Lo, and Jure Leskovec. 2017. Predicting intent using activity logs: How goal specificity and temporal range affect user behavior. In *Proceedings of the 26th International Conference on World Wide Web Companion*. International World Wide Web Conferences Steering Committee, 593–601.

[7] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[8] Edward Choi, Andy Schuetz, Walter F Stewart, and Jimeng Sun. 2016. Using recurrent neural network models for early detection of heart failure onset. *Journal of the American Medical Informatics Association* 24, 2 (2016), 361–370.

[9] Giovanni Luca Ciampaglia and Dario Taraborelli. 2015. MoodBar: Increasing new user retention in Wikipedia through lightweight socialization. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. ACM, 734–742.

[10] Gideon Dror, Dan Pelleg, Oleg Rokhlenko, and Idan Szpektor. 2012. Churn prediction in new users of Yahoo! answers. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 829–834.

[11] Şule Gündüz and M Tamer Özsu. 2003. A web page prediction model based on click-stream tree representation of user behavior. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 535–540.

[12] Ahmed Hassan, Rosie Jones, and Kristina Lisa Klinkner. 2010. Beyond DCG: user behavior as a predictor of a successful search. In *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 221–230.

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Selim Ickin, Katarzyna Wac, Markus Fiedler, Lucjan Janowski, Jin-Hyuk Hong, and Anind K Dey. 2012. Factors influencing quality of experience of commonly used mobile applications. *IEEE Communications Magazine* 50, 4 (2012), 48–56.

[15] Marcel Karnstedt, Matthew Rowe, Jeffrey Chan, Harith Alani, and Conor Hayes. 2011. The effect of user features on churn in social networks. In *Proceedings of the 3rd International Web Science Conference*. ACM, 23.

[16] Jaya Kawale, Aditya Pal, and Jaideep Srivastava. 2009. Churn prediction in MMORPGs: A social influence based approach. In *2009 International Conference on Computational Science and Engineering*, Vol. 4. IEEE, 423–428.

[17] Xiaodan Liang, Liang Lin, Xiaohui Shen, Jiashi Feng, Shuicheng Yan, and Eric P Xing. 2017. Interpretable structure-evolving LSTM. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1010–1019.

[18] Yozen Liu, Xiaolin Shi, Lucas Pierce, and Xiang Ren. 2019. Characterizing and Forecasting User Engagement with In-app Action Graph: A Case Study of Snapchat. *arXiv preprint arXiv:1906.00355* (2019).

[19] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association.*

[20] Yao Qin, Dongjin Song, Haifeng Chen, Wei Cheng, Guofei Jiang, and Garrison Cottrell. 2017. A dual-stage attention-based recurrent neural network for time series prediction. *arXiv preprint arXiv:1704.02971* (2017).

[21] Vineeth Rakesh, Niranjan Jadhav, Alexander Kotov, and Chandan K Reddy. 2017. Probabilistic social sequential model for tour recommendation. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 631–640.

[22] Vineeth Rakesh, Wang-Chien Lee, and Chandan K Reddy. 2016. Probabilistic group recommendation model for crowdfunding domains. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, 257–266.

[23] Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics* 20 (1987), 53–65.

[24] Narayanan Sadagopan and Jie Li. 2008. Characterizing typical and atypical user sessions in clickstreams. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 885–894.

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[26] Carl Yang, Xiaolin Shi, Luo Jie, and Jiawei Han. 2018. 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*. ACM, 914–922.

[27] Igor Zakhlebin, Em Horvát, et al. 2019. Investor Retention in Equity Crowdfunding. In *Proceedings of the 10th ACM Conference on Web Science*. ACM, 343–351.

[28] Yichao Zhou, Shaunak Mishra, Jelena Gligorijevic, Tarun Bhatia, and Narayan Bhamidipati. 2019. Understanding Consumer Journey using Attention based Recurrent Neural Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 3102–3111.

[29] Agustin Zuniga, Huber Flores, Eemil Lagerspetz, Petteri Nurmi, Sasu Tarkoma, Pan Hui, and Jukka Manner. 2019. Tortoise or Hare? Quantifying the Effects of Performance on Mobile App Retention. In *The World Wide Web Conference*. ACM, 2517–2528.