Characterizing and Forecasting User Engagement with In-app Action Graph: A Case Study of Snapchat

Yozen Liu
University of Southern California
Los Angeles, California
yozenliu@usc.edu

Xiaolin Shi
Snap Inc.
Los Angeles, California
xiaolin@snap.com

Lucas Pierce
Snap Inc.
Los Angeles, California
lpierce@snap.com

Xiang Ren
University of Southern California
Los Angeles, California
xiangren@usc.edu

ABSTRACT

While mobile social apps have become increasingly important in people's daily life, we have limited understanding on what motivates users to engage with these apps. In this paper, we answer the question whether users' in-app activity patterns help inform their future app engagement (e.g., active days in a future time window)? Previous studies on predicting user app engagement mainly focus on various macroscopic features (e.g., time-series of activity frequency), while ignoring fine-grained inter-dependencies between different in-app actions at the microscopic level. Here we propose to formalize individual user's in-app action transition patterns as a temporally evolving *action graph*, and analyze its characteristics in terms of informing future user engagement. Our analysis suggested that action graphs are able to characterize user behavior patterns and inform future engagement. We derive a number of high-order graph features to capture in-app usage patterns and construct *interpretable models* for predicting trends of engagement changes and active rates. To further enhance predictive power, we design an *end-to-end, multi-channel neural model* to encode both temporal action graphs, activity sequences, and other macroscopic features. Experiments on predicting user engagement for 150k Snapchat new users over a 28-day period demonstrate the effectiveness of the proposed prediction models. The analysis and prediction framework is also *deployed at Snapchat* to deliver real world business insights. Our proposed framework is also general and can be applied to any online platform.

ACM Reference Format:
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. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3292500.3330750

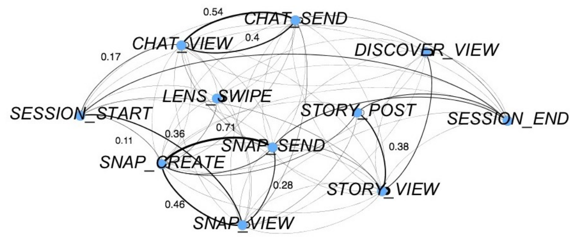


Figure 1: An example *action graph* derived from the in-app activity data of an individual Snapchat user. Nodes represent actions the user performs when using the app, and edges indicate transition probability from one action to the other within a session (i.e., from open to close the app).

1 INTRODUCTION

In recent years there has been a surge of interests on analyzing how users adopt online applications (Apps) on internet and mobile platforms [4] and understanding what retain user engagement on the Apps [9]. Despite many of such studies focus on profiling users [21] and building more sophisticated models towards surfacing more personalized content for users [17], however we still have limited understanding on what are the key factors that contribute to user engagement with the Apps—a question that has significant value for retaining and activating users. Therefore, it has become increasingly important to understand and characterize user behaviors and relate them to user engagement (e.g., prediction future engagement trends).

Prior efforts on analyzing user engagement focus on deriving *macroscopic* features from user profile (e.g., demographic information, session statistics)[1] and activity statistics (e.g., time-series of activity frequency) [16, 21]. A major line of existing work identify important characteristics of user behaviors via predicting if a user will return to the platform in a variety of ways. Return rate prediction is one of the most prevalent and significant task on social media platforms as companies are eagerly trying to find out the secret to their success. A few studies aim to predict churn rate, i.e cease activity on the platform, or user retention[2, 14]. Others come in from a different angle to predict when a user will return [13], if a user will return [16] or determine the lifespan of a new user [22]. Here we argue that such macroscopic features on user behaviors are (1) insufficient in *characterizing how how users interact with different functions within the Apps* (i.e., fine-grained App usage pattern); and

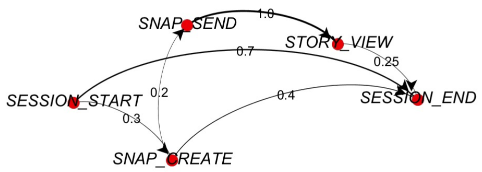


Figure 2: Another example of action graph with fewer number of actions and sparser action transitions (versus the graph in Fig. 1).

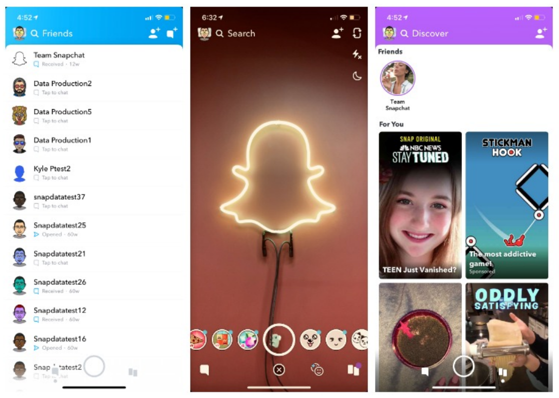
(2) *limited in terms of interpreting the user behaviors* (i.e., lack of explanation by black-box models).

In this paper, going beyond prior work, we propose a new angle to model user behavior with the concept of *Action Graph*—a weighted, directed graph for capturing individual user's in-app action transition patterns (e.g., how likely a user will view message right after she opens the app, and how likely she will create a new message after viewing some). Action graphs provide microscopic description of user's in-app usage behaviors and thus may contain additional signals to inform future user engagement. we take the anonymous data from Snapchat to study user behavior modeling and the connection between action graphs and user engagement. To motivate our study on capturing higher-order graph information, we perform a series of analysis. Through analyzing derivation of lower order features such as graph genes to higher order graph features like user paths, we find the intuition to model complete action graphs as a measure to capture higher-order graph information.

To enable interpretable user engagement prediction, we first propose a feature based model. Basic features such as macroscopic user level features as well as explainable graph features were modeled in a 2-step fashion. This feature-based model serves as a baseline for our deep neural network model. It also provides valuable insights and explanation to a deeper graph network.

To further improve user engagement prediction, we propose the following models. Activity sequence model with LSTM from previous work as baseline [21], static graph model with Graph Convolutional Network, and Temporal action graph model with GCN-LSTM. Temporal graph model best captures time dependencies of action graphs, and outperforms other models and previous baseline. To further improve performance, we propose a deep multi-channel end-to-end model combing activity sequence model, temporal action graph model and macroscopic features. Jointly training all models in an end-to-end manner enables better learning of combined effects and reaches best performance.

Contributions. In summary, we have made the following contributions: (1) We propose a new **data model**, *action graph*, for characterizing user in-app behaviors and apply it for forecasting user engagement. (2) We conduct **analyses** to understand how different user activity-related signals can help inform their future engagement and find plausible evidences from modeling high-order information in action graphs. (3) We propose a **GCN-LSTM model** for learning from temporal action graph and develop a **multi-channel end-to-end forecasting framework** for integrating with other useful signals. (4) Extensive experiments on both static and temporal action graphs show that modeling temporal action graphs provide notable performance improvement in user engagement prediction. Our ablation study demonstrate the effectiveness of simple interpretable graph features for predicting user engagement.



- Chat View
- Chat Send
- Snap View
- Snap Create*
- Snap Send*

Narrowcasting

- Snap Create*
- Lens Swipe
- Snap Save
- Snap Send*
- Story Post

Content Generation

- Story View
- Discover view

Broadcasting

* Actions can be performed in multiple screens

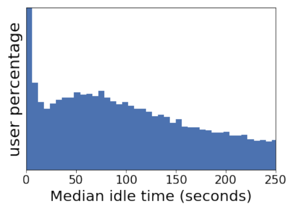
Figure 3: In-app screenshot of Snapchat and 10 actions to perform as the major functions of the app. Under each screenshot, we list possible actions users can perform when using the app.

2 ACTION GRAPH FOR CHARACTERIZING USER ENGAGEMENT

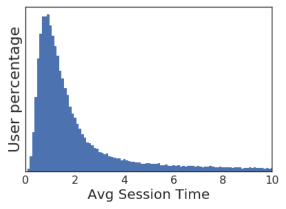
In this section we introduce the snapchat dataset used in our study, define the concept of action graph, and provide details on construction of action graphs from user activity data.

Data Collection. We collect the anonymous high level behavioral data of users that registered on the app between April 1st 2018 and September 30th in a specific region. A sample of the dataset from over 25 million new users in a specific country over the time period is further extracted. Four weeks of activity data right after registration is collected for each user, which consists of 10 core in-app functions (shown in Figure 3) that we find most effective in characterizing user behavior on Snapchat. We use the first two weeks of data for analysis and construction of user action graphs. The subsequent two weeks data is used for prediction evaluation of users' future engagement.

Sessions for In-app Activities. Ideally, an in-app session is a sequence of continuous actions separated clearly by users' disengagement. However, in real life mobile usage situations signals of starting engagement (i.e. App Open) can be triggered by both opening the app and returning from background mode. Similarly, signal of disengagement (i.e. App Close) can be triggered by disengaging the app or simply enters app-selection mode, checking notification, temporarily back-grounding the app, or replying to messages on another app but return shortly. As we can see, disengagement can be ambiguous as in above situations user will return to engagement from where they've previously left off and resume in a brief period of time. Splitting sessions solely based on disengagement signals recorded such as App Open and App Close will result in ineffectiveness of capturing accurate user action flow and user intention of using the app. Therefore, we need a more accurate definition for in-app sessions. As opposed to an entirely heuristic approach in splitting sessions, previous work has been done to identify sessions by fitting inter-activity time into multiple distributions [11]. By looking at the median idle time per user between start and end engagement signals in Figure 4a, we can observe that the distribution is a mixture of a long-tail and normal distributions. Median idle time was chosen for the reason that average idle time will subject to extreme outliers. The last 10 percentile of idle time distribution falls under the 25 seconds mark, which is coincidentally where two distributions intersect. We accordingly define 25 seconds as our threshold idle time to split sessions in addition to the disengagement signals (i.e. App Open/Close) in our experiments. Figure 4b shows the distribution of the time spent within each session on Snapchat following our definition of in-app sessions.



(a) Median idle time distribution



(b) Average session time

Figure 4: Distribution of user idle time between sessions and average time within sessions. Note that x-axis in (b) is re-scaled and both y-axes are masked to not show absolute value.

Definition of Action Graphs. In this study we define a new concept *Action Graph* to model user behavior. An action graph is a directed graph with 10 in-app actions (shown in Figure 3) plus Session Start and Session End as nodes. Similar to a Markov chain model, its edges are transition probabilities between actions nodes. Start and end nodes, i.e. Session Start and Session End, have only out-edges and in-edges respectively. Each user has his/her own unique action graph in the observed period. In the case of modeling temporal graphs, a unique action graph is derived at each time step per user. Examples of action graphs are shown in above figures, where Figure 1 serves as an example of a more engaged user and Figure 2 a less engaged user.

Action Graph Construction. In order to construct a meaningful action graph for each user, We first split a stream of core actions listed in Figure 3 into partitions of sessions that best captures the user's behavior on Snapchat. Each user has one or more session sequences. We remove sessions containing no actions but *Session Start/End* (i.e. invalid sessions). In order to make transition probability of action graphs more meaningful for each user, we only keep users with 5 or more valid sessions in our analysis, which results in around 150K sampled users for our study. Each user averages around 7 sessions a day and 98 sessions in the first two weeks of the observation period.

Specifically, each action graph is build with *all valid sessions of a user* using her/his activity data. Each *edge* in an action graph is calculated as the transitional probability between two actions aggregated in all sessions of a user. Action graphs can have a maximum of 12 nodes including Session Start and Session End.

3 LARGE-SCALE DATA ANALYSIS

3.1 Action Type

As we show in Figure 3, users' interactions with core functions on Snapchat are in three major categories: content generation, narrowcasting and broadcasting. Narrowcasting actions are triggered by one-to-one communication to interact directly with users and their friends, including Snap Send, Snap View, Chat Send and Chat View. Broadcasting behaviors are those that consumes and shares content to the entirety of a users' network. Such actions include story related activities and discover story activities such as Story Send, Story View and Discover View. The user interface of Snapchat is designed in a way that most narrowcasting functions are located on the left side of the start screen while most broadcasting functions on the right side.

The purpose of most social media studies is to understand user engagement, content consumption and to increase or retain the level of engagement and consumption. Preferably, a more engaged user will more likely to have more transitions between narrowcast and broadcast, consumption and creation activities. A transition from narrowcast actions to broadcast actions is favorable because consumption of broadcast contents bring in ad and revenue while users' use the app to communicate with friends. Observing transitions from broadcast to narrowcast can indicate that a user is more engaged with their friends and more likely to be retained. We find that if a session starts with narrowcasting activities, one third leads to broadcasting activities. If a session starts with broadcasting activities, one fifth of the time it leads to narrowcasting activities. It is fairly reasonable that a user only completes their intended communication or consumption activities and doesn't imply that users are not actively engaged. The subject matter here simply is how to increase engagement for those less engaged by encouraging them to navigate through cross functionality.

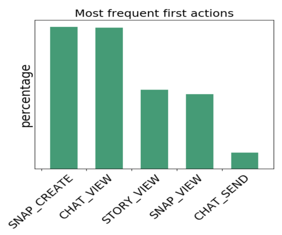


Figure 5: Most frequent first actions after Session Start. Figure shows snap create and chat view are two actions that most frequently triggers engagement. Y-axis is masked in order not to show absolute values.

In order to better understand what motivates users to start engaging with Snapchat, we examine the first action after *Session Start* in each session. Interestingly, we can see from Figure 5 that the most frequent first action for users are Snap Create (a content generation action) and Chat View (a narrowcasting action). Following by Snap View (which is another narrowcasting action) and Story View (a broadcasting consumption action). Additionally to actions that trigger users to engage, we also look into the duration of time a user spend on the app each session. In Figure 4b we find that the average session time per user shows a normal distribution and peaks within several minutes.

3.2 Session Gene

In order to have a deeper understanding of the sessions of action sequences and better connecting users intent of using the app (i.e. content generation, narrowcasting and broadcasting), we break sessions into session genes by applying soft clustering method Latent Dirichlet Allocation (LDA) [6]. Similar to topic modeling, we take each action as a word, each session as a document, and we name the topics learned from LDA "session genes". Session genes are fundamental elements that characterize the functions of a session at a finer level. To apply LDA to our data, we consider each session sequence as a single document and each action as a word. Five topics were derived from the LDA model and the topics are interpretable by summarizing the words with top weights in a topic. Each session contains a mixture of topics therefore we can view them as "session genes". The 5 genes derived are: Chat, Snap, Story View, Discover View and Content Creation. Figure 6 shows which actions comprise each gene. The content creation gene contains the widest span of actions, these actions can imply that users are sometimes playing around with the app and creating content while not necessarily sending them. Session genes can perfectly characterize the composition of each session that forms our action graphs. From Figure 7a we can observe that genes of communication eg. 'Chat' and 'Snap' are the most probable and common amid all 5 genes. We can conclude that narrowcasting communication features are the most used following with broadcasting (Story View). Next, we aggregate on a user level with all sessions of a user then take the mean of all users. In Figure 7b topic probabilities appears further normalized and this indicates that the majority of Snapchat users have a balanced usage across all functions in the app. Comparing session level aggregation Figure 7a and user level aggregation 7b, we can see that users that uses narrowcasting communication functions (Chat and Snap) more on Snapchat has a higher session count on average.

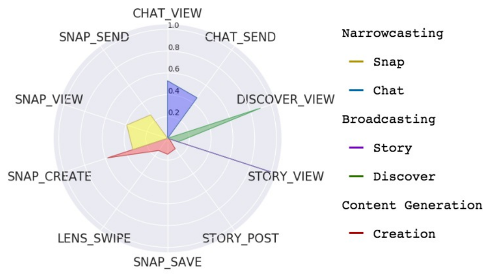


Figure 6: Session genes. We derive 5 session genes and chart shows the probability composition of each gene.

Common paths. Common paths are also the fundamentals of building action graphs, as the transition probabilities of graph edges are calculated from these paths. Common paths are derived from session sequences by extracting bi-grams, tri-grams and so on. Conversely to session genes which each session sequence are viewed as a bag of words in LDA model fit, common paths preserves the order of action sequences. As the fundamental building block of action graphs, common paths help us interpret action graphs and how it captures user behavior pattern.

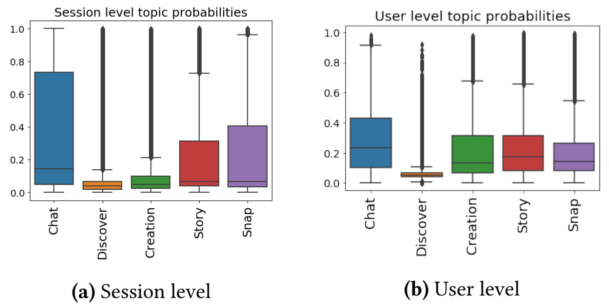


Figure 7: Session gene distributions on session and user level. Aggregation on different level is used to compare and provide analysis insight.

3.3 User Clusters and Engagement

In this section, we are trying to answer a simple question: How do the behavioral patterns described above correlate with user engagement? Enable to demonstrate the necessity to model higher order action graphs, we start from correlating lower level features to user engagement. We attempt user clustering with feature sets at each level. The number of clusters is determined by silhouette analysis. We pick the largest silhouette score and set cluster number to 4. Number of clusters are kept consistent to enforce comparability.

Session gene clustering. To cluster lower level graph features, we apply K-means clustering on probabilities of session genes. With 4 clusters we are able to separate users into groups with specific dominating genes. As a result groups of users with dominating story, creation, snap and chat genes can be separated as shown in Figure 8. However, As we can see in Figure 10a, these user clusters are able to separate users of different engagement level but a clear difference cannot be seen.

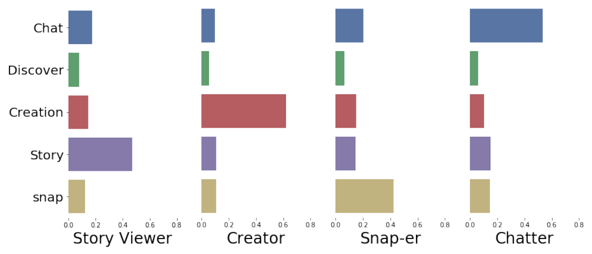


Figure 8: Session gene profiles. We derive 4 user clusters based on user's session gene constitution (i.e., probability of holding each session gene), and name them as: Story viewer, creator, snap-er, and chatter clusters.

Common paths clustering. With higher order features such as common paths, we repeat the same method and steps to cluster users. Using the probability normalized by session count and length per user, we derive 4 user clusters. Each cluster has a different distribution over the top paths with groups of users favoring particular paths . Nonetheless, the separation of engagement rate in Figure 10b for each user cluster are still not clear. Therefore, we move on to higher level features.

Graph metrics clustering. Once Again, we repeat previous clustering steps but replace with action graph level metrics, i.e. number of nodes, number of edges, and graph density, as features. Figure 9 shows the graph density distribution of users clustered by their

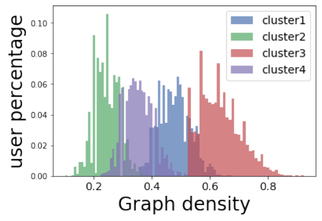


Figure 9: Density distribution of clusters by graph metrics

graph metrics. Different from lower level feature user clusters, we see a clear separation between user engagement rate in Figure 10c. Obviously, different user types are most efficiently captured with the use of action graphs and its simple features.

In summary, action graphs contain all the above information and is far more informative regarding user engagement. Therefore motivates the use of action graph modeling to predict future engagement.

4 FORECASTING USER ENGAGEMENT

In this section, we define two prediction tasks related to forecasting of user engagement, and propose a series of forecasting models for solving the two tasks, based on both *hand-built, explainable features* and *deep neural networks*. We start with a feature-based model that leverages features of different granularity, and move on to a more complex end-to-end prediction model to demonstrate action graphs as a good predictor of future user engagement level.

4.1 Task Formulation

We formulate a novel prediction task to predict the future engagement of a user. The prediction task leverages previous discovered insights and demonstrates that action graphs contributes to more accurate predictions. The prediction task can be formulated in two ways, as a classification problem, or as a regression problem.

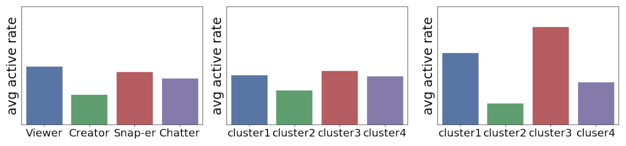
1. Classification of user engagement trend. First we define a 3-class classification task. In the following 2 weeks after observation period after registration, ie. week 3 - week 4, we record the days that a user is active. An active day is defined as a user having at least one valid session, which is a session containing at least one action we record. If a user has more active days in week 3 - week 4, we define that its engagement trend 'Increases' and vice versa 'Decreases'. If active day counts are the same from the first two weeks to the following two weeks, the user will be labeled as another class which stays the same. We also observe that in week 3 - week 4, active days appears to be more polarized than the first two weeks.

2. Prediction of user active rate. We formulate active rate prediction as a regression problem. Active rate is defined as "# of active days / # of total days" in the coming weeks. This task aims to predict the active rate of the coming weeks rather than only predicting the change of engagement trend between the two periods.

4.2 Feature-based Forecasting Model

We first propose a feature based model with macroscopic features and explainable graph features. The feature based model provides interpretability to deep neural graph models.

Macroscopic Features. Macroscopic features are extracted from user profiles with obvious features like session count and average



(a) Session gene **(b)** Common paths **(c)** Graph level

Figure 10: Average active rate of different clustering. Figures show that clustering with graph level features most correlates to active rate.

session time they are used to provide a baseline prediction. These features include (1) The average number of sessions per day for a user (2) The average time user spend on Snapchat per session (3) The gender of a user (4) The maximum age of a user during time period (5) The accumulated friend count of a user Macroscopic features are easy to understand and explain how it could affect the active rate.

Explainable Graph Features. An action graph can contain a vast amount of user behavior pattern information that is hard to see from a high level action graph. Therefore, we derive a few explainable graph features from action graphs to show that they are able to capture user behavior information and provide insights to user activity pattern and the way they interact with the app. (1) **Most likely first action**: The first action after session start. First actions often tells us what the user's most favorable interaction with Snapchat is, or what triggers the user to start using the app. (2) **Most likely action probabilities**: The most likely last actions before session end. (3) **K-hop paths**: A k-hop path starts from session start. K-hop paths are most likely action paths a user can take by calculating the joint probability $\Pi P(T_i)$ of transition probabilities T_i of edges along the path. (4) **End-to-end paths**: Here we utilizes BFS search to find paths from start node to end node. A path length cap is set to 6 to prevent indefinite search in loops and self-loops. We calculate the strength of each path (different length possible) as $P^{1/N}$ where P is the joint probability $\Pi P(T_i)$ of edges along path, and N the number of edges in path. (5) **Strongest cycles**: We utilize Johnson's algorithm to extract all elementary cycles from a graph. We then calculate cycle strength as $P^{1/N}$ where P is the joint probability $\Pi P(T_i)$ of edges and N is the number of edges in cycle. We will later show in section 5.6 selected graph features that are indicative to user engagement.

Predictive Models over Features. The feature-based model combines simple features vectors as input to the classifiers. The classifiers that we experimented on are SVM and Softmax classifier. As a regression problem we experimented on Linear regression and Ridge regression methods and select the optimal performance. SVM serves as a baseline where Softmax classifier reaches the best performance when combining all features.

4.3 Deep Neural Forecasting Models

To further improve user engagement prediction, we employ deeper neural models for various input signals. We utilize LSTM to model activity sequence time series data as baseline. We then propose static and temporal graph modeling methods and a multi-channel end-to-end training framework as our final best model for engagement prediction.

Activity Sequence model. Inspired by previous work that models user sequential behavior data and utilize LSTM sequence-to-sequence learning technique to encode and predict churn rate[21]. We implement 2 layer LSTMs with activity sequence containing actions in Figure 3 as a 10 dimensional time series input. The LSTM structure is able to model time series behavioral data and capturing evolvement of user activities. Each layer of the LSTM computes the following transformations:

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_t - 1, x_t] + b_f) \quad (1)\\ i_t &= \sigma(W_i \cdot [h_t - 1, x_t] + b_i)\\ c_t &= f_t * c_t - 1 + i_t * tanh(W_c \cdot [h_t - 1, x_t] + b_c)\\ o_t &= \sigma(W_o \cdot [h_t - 1, x_t] + b_o)\\ h_t &= o_t * tanh(C_t) \end{aligned}$$

where t is the time step in terms of days. h_t, c_t, x_t are the hidden state, cell state, and previous layer hidden state at time t. f_t, i_t, o_t are respectively the forget gate, input gate and output gate.

In our implementation of LSTM training, we apply softmax function $\text{softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$ to the linear projection of LSTM output o_T for our classification prediction $\hat{y}$ (2). Dropout is also applied to avoid over-fitting.

$$\hat{y} = \text{softmax}(W_c o_T + b_c). \quad (2)$$

Since LSTM can successfully capture time dependencies on user behavioral data, we will further combine its output embedding to other features and devise a more robust prediction model.

Static graph model. Static graphs are aggregated over the entire 2 weeks observation period, each user has a single unique graph. Although our derived explainable graph features serves well representing the graph, it does not capture all dependencies and pattern. Therefore, we employ a more powerful way to encode static graphs and conduct prediction. Graph Convolutional Network (GCN) [15] encodes each node as an embedding vector, can perform in a semi-supervised or supervised scenario to classify nodes. A GCN updates the node embedding using its neighbor information at each layer with message passing, and learn the representation of node. The layer-wise propagation rule in vector form is defined as

$$h_i^{l+1} = \sigma(\sum_j w_{ji} h_j^l W^l). \quad (3)$$

Where h_{v_i} is the feature of node i, j are the neighboring nodes of i, and W^l is the weight matrix of the l-th layer, σ can be a non-linear activation such as ReLU.

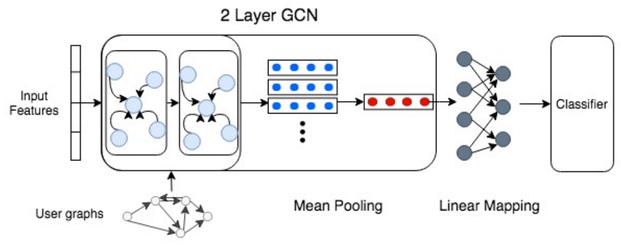


Figure 11: GCN model over static graph for engagement forecasting. User's static action graph (along with node features) are inputs to a two-layer GCN structure with mean-pooling after the last layer. Single graph embedding vector is then linearly mapped then feed to classifier.

We modify a two layer GCN structure to accommodate our case of graph classification on directed graphs as shown in Figure 11. To classify small graphs of many users, we train all small graphs together as a batched large graph. Utilizing the DGL library, graphs can easily be batched together. At each pass a mean-pooling transformation $v_G = \frac{1}{n} \sum_i^n v_i$ is applied on the output of two layer GCN to transform all node embeddings of a graph into a single graph representation. This will result in each graph having an unique embedding vector that characterize the structure of the graph. We then apply a softmax layer to linear projection of graph embedding vector v_G for our classification prediction $\hat{y}$.

$$\hat{y} = softmax(W_c v_G + b_c). \quad (4)$$

GCN graph embedding is a suitable representation of action graphs for the reason that it is trained and learned with our target engagement information and also incorporates neighboring information in the action graph at each pass. We can further combine such graph embedding with other features for stronger prediction.

Temporal Graph Model. In real world scenarios, action graphs evolve in a temporal fashion. Previously we model action graphs as static graphs ie. aggregate all sessions in observation period to derive a single graph. Here we introduce a time dependent variant of action graph modeling - temporal graph modeling.

While static graphs fail to capture the evolution of user behavior, temporal graphs account for the change in user pattern at each time step. We aggregate all sessions between each time step and computes an action graph. In our model, a day is used as the unit of time step. Thus, for a 14 day observation period we can derive 14 separate temporal graphs. A Long short-term memory network [eq 1] fits perfectly in this scenario for the reason that it recognizes temporal dependencies. In our model, each graph is transformed by Graph Convolutional Network(GCN) to a single graph embedding vector. We utilize a single GCN to encode all graphs at all time steps. Graph embedding produced by GCN at all time steps can be viewed as a sequence. This sequence is subsequently input to a single LSTM network for prediction. In simple terms, we compute an action graph at the end of each day with sessions in the day for each user, derive graph embedding of all graphs and feed to LSTM network as a timely sequence. A softmax layer is then applied to the last hidden state output of LSTM for classification task.

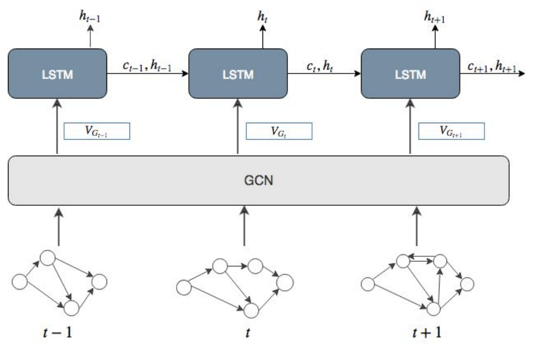


Figure 12: GCN-LSTM model over temporal action graph (i.e., a series of action graph snapshots) for engagement forecasting. At each time stamp we input the action graph snapshot to a GCN module and feed its output embedding (after mean pooling) to a LSTM for prediction.

Deep Multi-channel Forecasting Model. We propose a more intuitive and desired way to train the model by combining the LSTM for activity sequence and GCN action graph model along with macro features at the last stage to compute a single loss and back propagate to all models. Cross-entropy loss for each class c in softmax classifier can be calculated as

$$l_c = -\sum_c y_{i,c} \log(\hat{y_{i,c}}). \tag{5}$$

for each user i and its binary ground truth $y_{i,c}$ if y_i is class c. Alternatively, a linear SVM classifier can be used in place of softmax classifer. The loss will then be calculate as multi-class hinge loss

$$l_i = \frac{1}{N} \sum_{i \neq y_i}^{N} \max(0, 1 - w_{y_i} x + w_j x). \tag{6}$$

where w is model parameters. For our regression task we simply apply a linear regression layer and its is computed as Mean Squared error $l = \frac{1}{N} \sum_i^N (y_i - \hat{y_i})^2$.

Loss is then back propagated to the end-to-end framework into both models to complete the training process. By training in an end-to-end fashion the models learn together and reaches best performance.

Figure 13 illustrate a combination of macroscopic features, LSTM activity sequence and temporal GCN-LSTM model. We find that jointly training these models end-to-end result in best performance predicting user engagement.

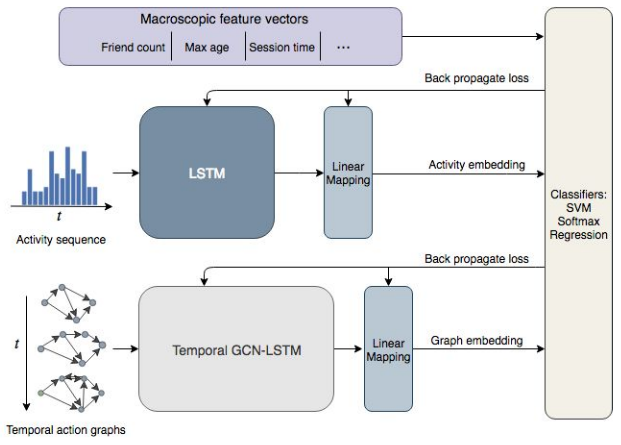


Figure 13: Deep Multi-channel Forecasting Framework. In our final forecasting model, macroscopic feature vector is combined with LSTM model (for activity sequences) and the proposed GCN-LSTM model (for temporal action graph). All these modules are jointly trained in an end-to-end manner for user engagement prediction.

5 EXPERIMENTS ON USER ENGAGEMENT PREDICTION

We conduct our experiments by adding features one by one to explore the effectiveness of each feature. We include 2 weeks activity sequence data modeling of our full observation period along with 2 weeks user graph data to ensure comparability. We aim to predict two variants of future engagement: (1) User engagement trend (2) Future active rate. We conduct experiments with different combinations of features as our model input, and predict a single class or value for the two tasks respectively. Intuitively, we would expect to see that graph embedding performs better than explainable graph features also giving a boost to existing activity sequence modeling method. Combining activity sequence embedding, temporal graph embedding and macroscopic features reaches best performance.

5.1 Datasets and Experimemt Setup

All experiments were done on the same sub-sampled data set of new users extracted from the Snapchat database. The original new user data to is sub-sampled into a 1:1:1 split for each of the three classes of engagement trend to avoid a skewed distribution. Labels of engagement trend were determined by the comparison of active days in first 2 weeks and the following two weeks; Labels of active rate for regression task is the active days of the following two weeks to the first two. Evaluation is done by randomly splitting data into 80:20 training and testing set for 10 times, then averaging the 10-fold cross validation score. The experiments were done on a single google cloud engine machine with only CPUs.

5.2 Model Training and Hyperparameters

Our experiments were done with SVM, softmax and linear regression as the final prediction layer in our model. All variables are also scaled before training using methods robust to outliers. For our GCN network we employ a two-layer configuration, applying mean pooling at the last layer to obtain a single vector representation for each action graph. Adjacency matrix for each graph is used as input feature. We configured our LSTM network as 2 layers with dropout (0.5) and embedding size of 32 which is empirically determined from previous work and further verified in this study. We utilize DGL library [1] and PyTorch [2] for implementation of GCN, LSTM models and our end-to-end framework. In our experiments, we did not pre-train the LSTM and GCN models as we found training them end-to-end from scratch yield similar (or slightly better) results.

To determine hyper parameters for our model, we conducted grid search using a held-out validate set in our experiments. We apply 0.5 dropout to LSTM and a L2 regularization $(1e-3)$ to the classifier layer to avoid model overfitting. Hidden state size of GCN was set to 20 to avoid overfitting as the action graph is relatively small in our case. Other input settings to our GCN network such as initializing with random Xavier normal/uniform initialization was also experimented but reaches similar but not better results.

5.3 Evaluation Metrics

Metrics for evaluating engagement trend classification. For our 3-class classification problem of engagement trend prediction, we use F1 score as our evaluation metric. In addition to accuracy, F1 score takes false positives and false negatives into account, hence a better evaluation metric. F1 score is the harmonic mean of precision and recall. Where precision = $tp/(tp+fp)$ and recall = $tp/(tp+fn)$. F1 score is then computed as $2*(precision*recall)/(precision+recall)$. All reported F1 score is Macro F1 score, which computes

[1]https://www.dgl.ai/
[2]https://pytorch.org/

the average of per-class F1 metrics in order to deal with inbalanced numbers of instances in each class.

Metrics for evaluating active rate regression. For our active rate regression problem, we compute Root mean square error RMSE as our evaluation metric. $RMSE = \sqrt{\frac{\Sigma_i(\hat{y_i}-y_i)^2}{n}}$. Comparing to another common metric for regression task, Mean absolute error, RMSE penalizes larger incorrect predictions more.

Table 1: Performance comparison on classification of user engagement trends. We performed 10-fold cross validation to compute the F1 score (based on precision and recall). We applied two different classifiers for feature-based models, end-to-end neural models, and their variants.

Models	**F1 score**	
Feature-based model	SVM	Softmax
Macroscopic features (Macro)	.372	.417
Graph features	.473	.473
Macro + Graph features	.479	.480
End-to-end neural model		
static GCN [15]	.490	.516
Macro + static GCN	.497	.515
activity LSTM [21]	.611	.614
Macro + activity LSTM	.613	.617
Macro + activity LSTM + static GCN	.616	.623
Temporal GCN-LSTM	.627	.629
Macro + activity LSTM + Temporal GCN-LSTM	**.629**	**.633**

5.4 Performance Study on Classification of User Engagement Trends

To compare the performance of graph features and graph embedding we consider macroscopic feature prediction as a weaker baseline and LSTM based activity sequence prediction as a stronger baseline, SVM as our baseline classifier and Softmax classifier to showcase the final performance of the classification task. Experiment results are recorded in Table 1.

1. Feature-based model vs. end-to-end neural model. Feature-based model can be trained with basic macroscopic features, graph features as well as different embedding features in a 2-step manner. End-to-end model on the other hand, jointly train all model parameters together. The end-to-end model in general learns a better fit for prediction, also more desirable and less complicated.

2. Modeling static vs. temporal graphs. Static graphs are aggregated over sessions in the whole 2 weeks observation period, whereas temporal graphs are aggregated per day for each user. While static graphs do not consider evolution of user behavior patterns, temporal graphs model action graphs with time dependency. As a result, we can see in Table 1 that when modeling the time variant of action graphs prediction performance improves. Meaning that we can successfully capture user behavior patterns with daily evolving action graphs.

3. End-to-End integration of different models. In our 3-way classification task for engagement trend prediction, we experiment different combinations of features. We use LSTM activity sequence embedding as a baseline from previous work [21]. Our temporal graph model alone is able to outperform LSTM activity sequence model. Adding static graph embedding on top of macroscopic features and activity embedding is able to increase performance. Combining macroscopic features, activity embedding and temporal graph embedding reaches best performance.

Table 2: Ablation study on different graph features for engagement trend classification. We constructed a base model using only macroscopic features and integrate it with different graph features.

Graph features	**F1 score**
Macroscopic features	.452
Macro + First actions	.473
Macro + 3-hop paths	.484
Macro + End-to-End paths	.482
Macro + Strongest elementary cycles	.482
Macro + All graph features	**.502**

4. Ablation study on graph features. We also conduct an ablation study of how our explainable graph features improve performance as shown in Table 2. Using macroscopic features as baseline, each graph feature adds on different effects and improvements of performance to baseline macroscopic features. A combination of all graph features show that graph features adds predictive strength to user engagement and also serves as an explanation to why a deeper graph network would work.

5.5 Results on Active Rate Prediction

Active rate prediction is a more fine-grained task than engagement level prediction, as it is a regression problem of 14 day range. The same combination of features in engagement trend prediction are experimented.

1. Temporal graph versus Activity sequence. In a more fine-grained task, we can see from Table 3 that modeling temporal graphs is able to provide a far superior prediction result than baseline activity sequence. The result that such a huge performance jump can be achieved by modeling temporal graphs shows its ability to predict future engagement rate.

2. Feature integration. We also integrate multiple feature combinations into our model. A combination of both baseline activity sequence and temporal action graph along with macroscopic features reaches best performance. With information of user engagement that action graph captures, business decisions can be made such as user retention and targeted incentives.

Table 3: Performance comparison on active rate prediction. We report the root mean square error (RMSE) of active rate prediction (regression). Smaller RMSE indicates a better prediction performance on the test set.

Features	**RMSE**
Macro features	5.30
Graph features	4.70
Macro + graph features	4.65
Static GCN	4.25
Activity LSTM	3.69
Macro + activity LSTM	3.32
Macro + activity LSTM + static GCN	3.23
Temporal GCN	2.95
Macro + activity LSTM + temporal GCN	**2.75**

5.6 Case Study on Graph Features

Amid many explainable graph features, we are eager to find features that most differentiates levels of user engagement. A simple L1 penalty term can be added to linear kervel SVM classifier to produce sparse coefficients and naturally acts as a feature selection tool. Therefore, variances of features that contributes most to prediction can be extracted after training all graph features with linear SVM. The most predictive graph features of higher order includes k-hop paths, end-to-end paths and cycle strength. Most top 3-hop paths from session start contains chat activities. It is evident that chat activities can imply communication with friends, which subsequently acts as a strong indicator of engagement increase. Top cycles on the other hand are mostly complex structures not simple cycles. This indicates that a well rounded user that is involved with more core functions on Snapchat is more engaged. Similarly, selected end-to-end paths are longer paths, not short simple paths. It implies that user spending more time, uses more functions in each session serves as a strong measure to the users' engagement. As we move on to deeper neural models, the interpretability of simple graph features are the premise to the reason why it works.

6 RELATED WORK

User behavior modeling and prediction. Many papers models user behavior and aim to predict if a user will return to the platform in a variety of ways. Return rate prediction is one of the most prevalent and significant task on social media platforms as companies are eagerly trying to find out the secret to their success. A few studies aim to predict churn rate, i.e cease activity on the platform, or user retention[2, 14, 21]. Others come in from a different angle to predict when a user will return [13], if a user will return [16] or determine the lifespan of a new user [22]. Another purpose of user behavior modeling leads to the discovery and prediction of user intention. Studies were done on predicting user intent and subsequent behavior, as well as prediction of a user's intention [8] and the users purchasing intent [17] on pinterest. Interesting to mention, some studies in the domain focus and models the consumption of a user [5, 19]. Many different methods were used in the task of prediction. Some uses simple logistic regression or gradient boosted tree methods with macroscopic features and reaches good result[1, 5, 16]. Others incorporate time information and utilizes Cox proportional hazard models [13, 22] or Long short-term memory structure [21] for prediction.

Action sequence modeling. There are plenty of publications modeling activity sequence as a graph in the field of search platform and social platforms. However, different from our work that models actions into activity graphs, related literature models Queries and Clickstreams into activity graphs(markov chain graphs). In literatures on search platforms, Queries can be modeled into a sequential graph to predict search success [12] or produce recommendation [7]. Queries graphs are also used to model user search behavior [3]. Clickstreams on the other hand were also used to model user behavior as a sequential graph/markov chain model on social and web platforms [4, 10, 18, 20]. There are studies done on activity sequence graphs with search queries and clickstream but we have yet to see activity sequence graphs modeled with user activities on a social media platform.

7 CONCLUSION

In this paper, we explore and analyze user behavior of Snapchat's new user through action graphs. Following analysis insights we developed a robust and predictive action graph modeling framework. In our analysis we provide valuable insights to understand new user patterns after initial registration and likelihood of future engagement. While our models and analysis focuses on Snapchat data, similar techniques and models can be applied to other real world social media or online platforms. Our action graph prediction framework is *deployed in Snap Inc.* to deliver analysis results and benefits according business and production decisions which includes but not limited to user modeling, growth and retention. Possible future work can be done on connecting users ego-network with its behavioral pattern.

REFERENCES

[1] Tim Althoff and Jure Leskovec. 2015. Donor retention in online crowdfunding communities: A case study of donorschoose. org. In *WWW*.
[2] 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.
[3] Ricardo Baeza-Yates, Carlos Hurtado, Marcelo Mendoza, and Georges Dupret. 2005. Modeling user search behavior. In *Web Congress*. IEEE.
[4] Fabrício Benevenuto, Tiago Rodrigues, Meeyoung Cha, and Virgílio Almeida. 2009. Characterizing user behavior in online social networks. In *SIGCOMM*.
[5] Austin R Benson, Ravi Kumar, and Andrew Tomkins. 2016. Modeling user consumption sequences. In *WWW*.
[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.* 3 (March 2003), 993–1022.
[7] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, Aristides Gionis, and Sebastiano Vigna. 2008. The query-flow graph: model and applications. In *CIKM*.
[8] Justin Cheng, Caroline Lo, and Jure Leskovec. 2017. Predicting intent using activity logs: How goal specificity and temporal range affect user behavior. In *WWW*.
[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] Şule Gündüz and M Tamer Özsu. 2003. A web page prediction model based on click-stream tree representation of user behavior. In *KDD*.
[11] Aaron Halfaker, Oliver Keyes, Daniel Kluver, Jacob Thebault-Spieker, Tien Nguyen, Kenneth Shores, Anuradha Uduwage, and Morten Warncke-Wang. 2015. User session identification based on strong regularities in inter-activity time. In *WWW*.
[12] Ahmed Hassan, Rosie Jones, and Kristina Lisa Klinkner. 2010. Beyond DCG: user behavior as a predictor of a successful search. In *WSDM*.
[13] Komal Kapoor, Mingxuan Sun, Jaideep Srivastava, and Tao Ye. 2014. A hazard based approach to user return time prediction. In *KDD*.
[14] Jaya Kawale, Aditya Pal, and Jaideep Srivastava. 2009. Churn prediction in MMORPGs: A social influence based approach. In *CSE*, Vol. 4. IEEE, 423–428.
[15] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv:1609.02907* (2016).
[16] Zhiyuan Lin, Tim Althoff, and Jure Leskovec. 2018. I'll Be Back: On the Multiple Lives of Users of a Mobile Activity Tracking Application. In *WWW*.
[17] Caroline Lo, Dan Frankowski, and Jure Leskovec. 2016. Understanding behaviors that lead to purchasing: A case study of pinterest. In *KDD*.
[18] Narayanan Sadagopan and Jie Li. 2008. Characterizing typical and atypical user sessions in clickstreams. In *WWW*.
[19] William Trouleau, Azin Ashkan, Weicong Ding, and Brian Eriksson. 2016. Just one more: Modeling binge watching behavior. In *KDD*.
[20] Gang Wang, Xinyi Zhang, Shiliang Tang, Christo Wilson, Haitao Zheng, and Ben Y Zhao. 2017. Clickstream user behavior models. *ACM Transactions on the Web* 11, 4 (2017), 21.
[21] 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 *KDD*.
[22] Jiang Yang, Xiao Wei, Mark S Ackerman, and Lada A Adamic. 2010. Activity Lifespan: An Analysis of User Survival Patterns in Online Knowledge Sharing Communities. *ICWSM* (2010).