MEng Thesis Proposal

Shidan Xu

**Modeling User Transition Behavior across Networks With a Probabilistic Programming Approach**

**Abstract**

This paper describes a thesis project to be carried out at the MIT Advanced Networks Architecture (ANA) group. This project involves predicting a user’s behavior in various networks. In particular, we are interested in how the user is likely to transition between networks. We plan to design various models to predict the user’s behavior, and use various evaluation criteria to understand which model most vividly captures the user’s behavior. As a learning experience for our team and me, we employ probabilistic programming (PP) techniques. PP is a way of programming that utilizes existing software packages that include encapsulated methods for the inference step of machine learning. By simplifying the implementation, it allows users to focus on models. This project has two main objectives. The first is to design several models to capture one or multiple aspects of a typical user, or a group of users’ behavior in a network. The second is to evaluate each model to decide which model is more feasible, and which model is applicable under what situations. Together this project aims to give a better understanding of how network user transitions.

**Introduction**

This project relates to the FIND (Future Internet Design) initiative [1], which asks the question of what the requirements should be for a global network 15 years from now, and how we can build such a network if we are not constrained by the current Internet. The current Internet has made some design choices with assumptions. One particular assumption the Internet is making is that network mobility is similar to mobility in geography. [2] In reality we found that to be not necessarily true. A user could transition from a 4G network to Wi-Fi by walking a few meters in real world, but the IP addresses in network topology is likely to be not adjacent, even entirely different. In extreme cases, the nearest common ancestor of nodes in network topology can be on opposite coasts of the country. Such design is costly for the user to transition between networks. In designing a new network, one of the focal points is to decrease such inconsistency, as packets of information need to be rerouted.

The first step in creating an efficient network is to know the users. One of the actions we can evaluate is user transition. The bulk of this proposed work hence relates to evaluating and predicting how the user is likely to transition between networks, given his/her information. The workflow of this project is to

1. Process Datasets
2. Build a model
3. Evaluate the model’s performance
4. Recurse steps 2 & 3

We examine each step in the next paragraphs.

**Datasets**

Multiple datasets exist for carrying out this research. We decided to first reproduce the Yang paper results as a baseline. Hence we start out by exploring the UMass dataset. The UMass dataset contains user email activity in the form of IMAP logs for residents of University of Massachusetts at Amherst (1). The dataset includes user information such as the identity, the time of action, and action in terms of attaching to network, detaching from network, IP address, and the device used to log in. With this information, we can decide whether a transition happens from the logs. Brian Copeland, a UROP student of the group, parsed the original log entries into SQL entries that contain each individual session, or duration when the user is on a particular network. We can easily overlay the durations to decide occurrences of transition. The details of transition criteria can be found in Yang [3] paper.

**Modeling**

Given the dataset, we want to model how the user is likely to transition. In general, there are two categories of models we can use, generative vs. discriminative. A discriminative algorithm learns what criteria separate the classes given the dataset.

For instance, logistic regression is a discriminative algorithm. A generative model, focuses on each class, and creates a model for what the underlying process for each class is [4]. For x the features, y the output, discriminative models learn Pr(y | x), whereas discriminative models learn Pr(x | y). Therefore, generative model allows researchers to synthesize new dataset by understanding what the underlying distribution is. Since we are predicting user behavior, we lean towards using generative models.

For preliminary baseline, we reproduce the UMass three-state hidden Markov model. We model the user in three states: not on a network, on one network, on multiple networks. A 3 by 3-state transition matrix can be calculated directly from the dataset. Given this transition probability matrix, we can generate estimations of what the user may behavior in the future. We do not necessarily believe that such modeling optimally captures user behavior, but it serves as a nice baseline.

|  |  |  |  |
| --- | --- | --- | --- |
| Transition Prob. | On 0 Network | On 1 Network | On >=1 Networks |
| On 0 Network | 0.9 | 0.1 | 0 |
| On 1 Network | 0.3 | 0.5 | 0.2 |
| On >=1 Networks | 0 | 0.4 | 0.6 |

Table 1: Empirical estimate of transition probabilities (Data fake).

The more complex version involves using more features. Weather, location, time of day can all be factors that affect how the user transitions. For instance, when it snows, the user is more likely to stay indoors; the number of transitions may be fewer. We plan to learn these features through machine learning. First we separate the original dataset into training and validation datasets by time. Based on the training dataset, we build a model to extrapolate how the user will behave, and compare with the validation dataset.

The UMass dataset is rather simplistic with minimal useful information, and suffers from information loss, as some entries of device id are not available. We have the option of extending the work to a broader, more complex dataset [What is the name of the dataset?], if the legal process can proceed.

**Evaluation**

The performance of a model needs to be assessed using some evaluation criteria. One such evaluation criterion can be the duration that we predicted the user to be on a particular network vs. actual duration. Another type of evaluation can be whether we correctly predicted the sequence of network transitions. This would be useful if the user follows a daily routine. Various evaluations explore various aspects of the models, and the models may beat each other under different evaluation metrics.

For baseline, we evaluate the distribution of number of transitions for each user on a daily basis as noted in Yang. We compute one distribution from the training dataset, and another for the randomly generated dataset based on HMM runs. We make the naive assumption that all users behave similarly and the transition distributions are approximately normal.

[Include Image 1 of Training data distribution]

[Include Image 2 of testing data distribution]

This is a minimal evaluation. For more sophisticated evaluation criteria, we may consider the proportion of duration that we correctly predicted a user in the correct state; or we may be interested in correctly estimating the sequence of transitions that a user takes. Different evaluation criteria target different aspects of prediction. The latter criterion may be useful in detecting the user’s daily network usage habits.

**Recurse & Compromise**

We repeat the approach on a new set of models and evaluations. Given the timeframe of the project, we might need to make a compromise between a rich modeling approach vs. a rich evaluation approach. Ideally, for x models and y evaluations, we can fill out the entire x by y matrix. However, certain evaluations on particular models may be irrelevant. The advantage of probabilistic programming is that it grants the ability to easily implement different models. In a traditional programming language, the user implements the optimization function, for instance gradient descent. The user also needs to implement the neural network forward and backward iterations. Using a probabilistic programming approach, we can make use of readily available inference methods. We’d like to have a framework that can outperform the current research, as well as have better evaluation of the current models.

**Toolkit**

We decided to use Python PYMC3 for our inference methods. Many probabilistic programming languages exist. While some are based on existing coding languages, others are completely new languages. We decided to stay away from alpha version, new programming languages as the learning barrier may hinder the progress. The advantage of using a probabilistic programming language is that the inference is readily implemented in the API, as opposed to having to implement it from scratch. PYMC3, in particular, is a python API that is easy to use. It offers many general discrete and continuous distributions, and the option to specify our own distribution [5]. While PYMC3 only supports a set of common inference methods, we argue that although other inference methods could be faster, optimization of speed is of minimal importance in training our relatively small dataset.

**Timeline**

12/14 Simple working 3-state HMM model, with baseline metrics

1/15 Multiple working models

2/01 Related Works

2/08 Draft Introductory

2/15 Multiple evaluations

2/22 Discussion of evaluation techniques

3/15 Analysis of pros and cons of each model; possible extension to new datasets

3/22 Draft Modeling and Evaluation

4/15 Draft Conclusion

4/22 Thesis

**Related Works**

There are relatively few related works in the field. One major related paper is Yang’s Measurement and Modeling of User Transition Among Networks. This paper created one particular 3-state hidden Markov model that’s evaluated by measuring the distribution of cost function of signaling a network attachment / detachment. We’d like to expand the work in both the modeling and evaluation space.

**Acknowledgement**

The author would like to thank Dr. Karen Sollins and Dr. Steven Bauer for introducing this amazing topic to the author, and for their continued advice and insight in conceiving this project proposal.

**References**

1. NSF NeTS FIND Initiative, <http://www.nets-find.net/>
2. Jacobson, Van. Networking Named Content. New York: Association for Computing Machinery, 2009.
3. Yang, Sookhyun. "Measurement and Modeling of User Transitioning Among Networks."
4. Ng, Andrew. “On Discriminative vs. Generative classifiers: A comparison of logistic regression and naïve Bayes”
5. PyMC3, <https://github.com/pymc-devs/pymc3>