2019 7th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)

Study Area Recommendation via Network Log Analytics

Anupama Upadhayula[1], Avinash Ravilla[1], Ishwarya Varadarajan[1],
Sowmya Viswanathan[1], and David C. Anastasiu[1,*]

***Abstract*—With the current surge in the number of devices connected to the Internet all over the world, usage of network bandwidth has also seen a huge increase. This has led to network traffic congestion and slowdowns. It is increasingly important to anticipate future network usage, which would help prepare and provide adequate infrastructure or service support. In this paper, we propose a model that uses machine learning techniques to analyze network logs of access points installed throughout the campus of San José State University (SJSU) to predict its network usage at a future point in time. We also propose to build a classification model to classify user's browsing behavior as either study or leisure, based on the content of the pages they view. Hence, the final model will be able to predict network throughput of access points and the type of activity they will be used for. We utilize these models in a novel mobile phone application we designed that helps students choose an appropriate place to study or hang out at a chosen time in the near future. Our application combines the use of mobile phone sensors for localization with the power of machine learning models executed in the cloud to provide a seamless user experience for students and faculty alike.**

***Index Terms*—mobile phone, machine learning, cloud, network log analytics, classification, regression.**

I. Introduction

It is difficult to find an access point (AP) which is highly available and suitable for a user's nature of work at SJSU. Existing solutions revolve around improving the range of APs and using better routers. Our solution is to design an algorithm that considers the usage of various APs located on the SJSU campus and predicts a better AP location for a user. Network data is collected from the APs across the university on a daily basis. Our method analyzes more than 15 GB of domain name service (DNS) log data spread over 283 APs equipped with wireless sensors to predict the throughput of an AP on campus. Additionally, It identifies a location as 'work' or as 'leisure' based on the type of activity the users connected to the AP engage in.

Wireless communication is one of the most successful means of communication. Network congestion is a major problem for wireless networks. It leads to loss of packets resulting in poor quality of service. Performance degrades when the load is not anticipated and necessary measures are not deployed to handle sudden crises. An algorithm that can analyze past data and find patterns in them to anticipate future events can be used to potentially mitigate congestion problems. Mining network data comes with several challenges, however. Many factors, such as large amounts of data, network speed, varied dynamic network topology, resource constraints, and others, need to be considered while performing data mining.

There are four different techniques that can be considered while mining network data – frequent pattern mining, sequential pattern mining, classification and clustering. There have been many studies which explored the use of time series as a general network traffic prediction model. Data is recorded over hours, days, weeks, months, years, etc. Models such as SARIMA, and back propagation neural network (BPNN), have been used and compared for analyzing such data. Results have shown that neural network models comparatively give better prediction accuracy. Several linear and non linear models can also be adopted to predict the rate of traffic. Some linear models, such as the Holt Winters and the ARMA models, are combined with clustering techniques to yield better results. Prediction models can be built to mine the network data or to forecast the network data by enhancing existing models.

In our project we consider several methods for predicting future utilization of an AP, including ARIMA, Facebook Prophet and Recurrent Neural Network Long Short Term Model (RNN-LSTM). We developed a mobile and web based application that puts the best of the above models to use and delivers a recommendation where a student may find a study area or a place to relax. Analyzing and tracing the mobility characteristics of mobile users is not addressed in our current model, but our data has the scope to build such a model. Our applications are hosted in the cloud. Mobile cloud deployment of the applications has numerous advantages compared to a centralized or a decentralized system. A centralized system, where a single server serves a large number of clients, involves high support and maintenance costs and is vulnerable to single points of failure. In a decentralized system, single points of failure do not occur since data, hardware, and software are distributed over the network. In both cases, having our own data centers would be a huge overhead due to high build and maintenance costs. Hence, a mobile cloud deployment is an effective solution to provide high application availability, elastic scalability, and data security at a low cost.

II. Related Work

A. Network traffic Classification

Network architecture has become complicated with the rapid development of the Internet and network services. Classification of network traffic has become one of the most important tasks for network administrators these days. Shafiq et al. [1]

[1]Computer Engineering, San José State University, San José, CA
*Corresponding Author, david.anastasiu at sjsu.edu

2573-7562/19/$31.00 ©2019 IEEE
DOI 10.1109/MobileCloud.2019.00014

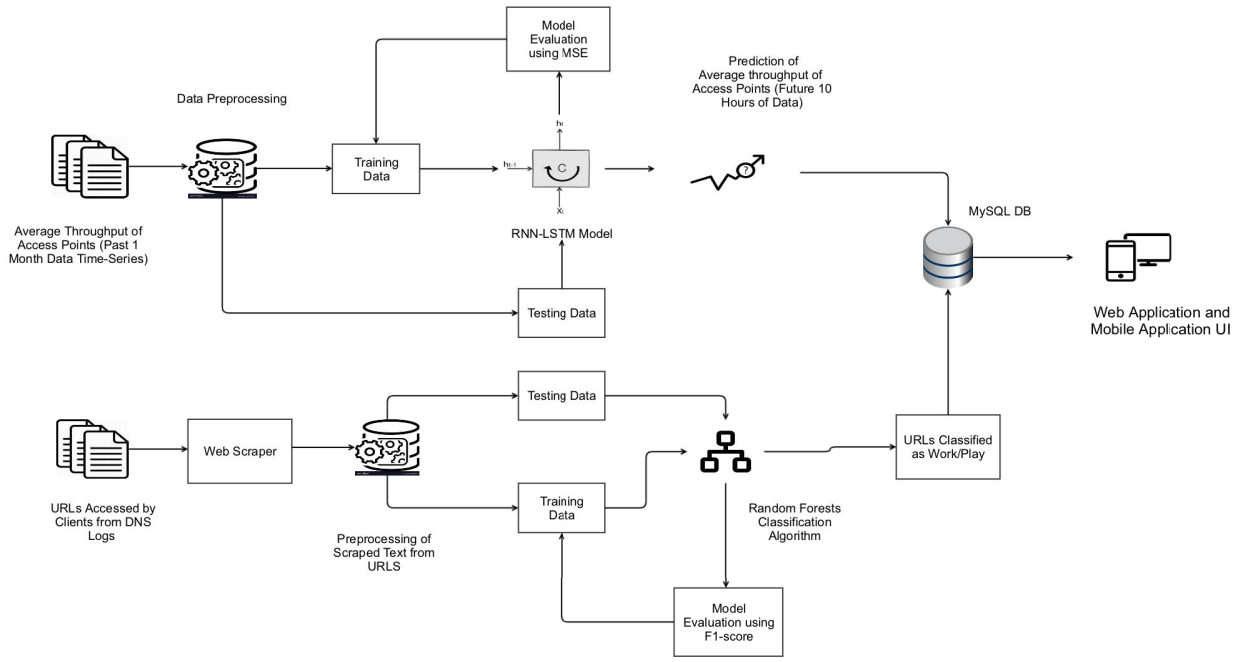

Fig. 1. System architecture.

discuss various types of network traffic classification methods. They find that making real-time decisions based on the available networks, the amount of communication between them, or the threshold limits of each of those devices is a challenging task. Such resource constrained computing and resource retention prompts us to analyze the network data. Miao et al. [2] provide a comprehensive discussion about classification of network data. They showed that random forest and neural networks are two of the top machine learning algorithms that work well to classify network traffic. These models can help us understand the bottlenecks and avoid congestion in the network.

B. Traffic Prediction

Traffic control centers manage and collect huge amounts of data regarding the traffic and usage of networks. Sometimes, a prediction model can be used to provide information on when the network will be busy. Prangchumpol [3] proposed an unconventional model which uses association rules to find a relationship between the network traffic, semester, and time. It predicted the network traffic for the next day in the semester which was useful in network routing and improved the network performance. Wen and Lee [4] proposed a hybrid grey-based recurrent neural network (G-RNN) which helped in traffic prediction. This technique has been proven to be accurate when the data are random, and have spatial and time series correlations. The grey preprocessing approach is used to decrease the randomness in the data. Traffic prediction is also useful when trying to forecast congestion in networks.

Aldhyani and Joshi [5] proposed using an integrated model of different time series models with soft clustering techniques to predict network usage. They hypothesized that linear time series models like Holt Winters, exponential smoothing, ARMA, and autoregressive neural network, when integrated with a clustering approach, provide better results in network traffic forecasting. Eterovic et al. [6] compared the performance of ARIMA and artificial neural network (ANN) models. The ARIMA model assumes the network traffic to be stationary in time. They showed that ARIMA worked for short term predictions but failed for long term predictions. ANN gives better prediction results in this study when taking long term forecasts into consideration. Feng and Shu [7] also compared ARIMA and ANN models and concluded that ANN has better prediction accuracy than ARIMA. ARIMA was proved once again not to be useful for long range dependent predictions. Ang et al. [8] employed long short-term memory (LSTM) recurrent neural network (RNN) to analyze the traffic flow. They concluded that traffic flow together with speed as inputs to the model yields good results. LSTM is a machine learning technique which analyzes and learns information from data which spans over a long period of time. It is very useful for time series prediction. Zhuo et al. [9] found that better accuracy is obtained when using LSTM for predictions in large granularity data set.

III. System Architecture

In this section, we describe the architecture of the system we have devised, which can also be viewed in Fig. 1.

We developed a web based and a mobile based application for users to view the predicted average throughput and type of activities that are performed using a particular AP.

The back-end system of our application consists of Random Forest, a classification model to categorize APs as being used for work and leisure classes, and RNN-LSTM, a time-series model to predict the availability of APs.

a) Random forest: Random forest is a supervised classification algorithm which belongs to the ensemble family of algorithms. A random forest is a collection of decision tress formed from randomly split input data. They use an important and powerful technique, called bootstrapping, which is based on descriptive statistics. Each tree is formed through rules that are automatically discovered using the Gini index.

b) Long Short Term Memory Model (LSTM): Recurrent Neural Network (RNN) is one of the deep learning models extensively used in time-series analysis. When an RNN contains LSTM units as its cells, it is called an LSTM neural network. RNNs can predict the output by learning the most recent data from the past, but they do not have the capability to remember data further back in time. An LSTM network is capable of remembering data for longer periods and predict the output in the future. Hence, LSTM networks are more reliable for problems with long-term dependencies.

In our work, we used the same LSTM model architecture as that in Olah [10], which consists of two neural network layers with each layer containing 4 LSTM units, for each time step. Fig. 2 depicts this model. Each LSTM cell has 20 units that learn data and predict the output for the next time step. The LSTM predicts average throughput for the next 4 hours from the current hour by learning hourly throughput data from the past month.

IV. Project Design

The paper deals with two problems and proposes to solve them using machine learning. We predict the average throughput (Kbps) using time-series models and also categorize APs into work and leisure types using classification models. The tool we propose to develop can only recommend locations to its users based on historical data and does not force the users to comply to the aspect of the location. If a large enough number of users perform leisure activities in a location recommended for work, the system adapts based on the new usage characteristics and changes the location status to leisure. Future students seeking a location to study will be recommended alternate places to study. Users can see the availability and type of activity for each AP on a Google map.

A. System Design

1) Regression Problem: This problem involves analyzing time-series data containing average throughput from the past month for each AP in the University and predicting the throughput for the next 4 hours from the current hour. Three time-series models were used and their predictions were evaluated using Mean Absolute Error (MAE). The models will be trained and tested with different sets of parameters. On the basis of evaluation, the predictions of the most efficient model will be displayed on the front-end applications.

2) Classification Problem: The classification problem involves using three classification models to categorize network access data to multiple categories. The effectiveness of the models was evaluated using the F1 score measure to determine the best model to be used in our application. The input data to the models contain texts scraped from corresponding URLs accessed by the users of the university's network using a web scraper. The scraped text of each URL was manually labeled as *work* or *leisure* to aide the learning of supervised classification algorithms.

B. Client Design

We developed a Web application and a mobile app for the user to check the availability of APs on the campus network. The user interfaces of the applications prompt the user to enter a preferred date and time. Once the request is submitted, the page displays the APs at respective locations in the Engineering and MLK library buildings using Google maps. The mobile app can use the phone's GPS sensor, if available, to localize the user within the campus map. APs are denoted using markers on the maps. When the users click on the marker, they can see the details of the corresponding AP, such as its name, the predicted throughput for the selected hour, and the type of activity for which the AP is being or will be used.

V. Methods

A. Classification

We used supervised classification models to categorize the nature of work each AP is being used for in two categories, namely *work* and *leisure*. This section explains the different algorithms we used for the classification problem and their results.

1) Input Data: We used the network access logs as input to our prediction methods, which contain the following information: date, client IP, client ID, URL, query type, query DNS IP, answer code, answer DNS IP, record type, and error. To categorize the nature of work in APs, we considered the columns date, client IP, URL and record type from the network log data as input to our application. The column URL contains the address of websites browsed by the network users.

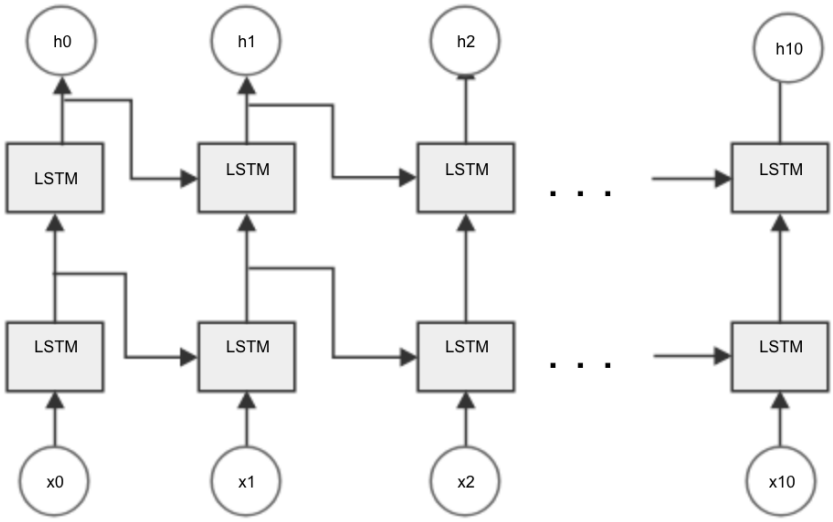

Fig. 2. RNN-LSTM Architecture Olah C. [10]

Duplicate URLs were removed and a web scraper script was used to scrape the text of the corresponding websites. The scraped text was then manually classified into two categories: leisure and education. In some cases, the web scraper directly assigned classes to URLs based on words in the URL. For example, facebook and youtube URLs were automatically designated as leisure.

We manually labeled 12,358 scraped texts to prepare the training data for the classification algorithms. We removed stop words (commonly used words such as 'the', 'in', 'a', 'an', 'for', etc.) and retained the base form of words, also called as lemmatization. The texts were then converted to sparse vectors using a count vectorizer in a format readable by machine learning algorithms. Directly categorized URLs and those categorized using classification models were uploaded to a MYSQL database table.

2) Models: A brief description to the classification models is given below.

a) Random forest: Random forest is a classification models that belongs to the class of ensemble algorithms. An ensemble method is a technique that combines the result from different machine learning algorithms to produce more accurate results. In the case of random forest, different random subsets of features are used to train decision tree classification models, and the final classification result is given by aggregating the predictions of the individual tree models for a given test sample.

b) Gaussian naïve Bayes: Naïve Bayes is a collection of classification algorithms that are purely based on the Bayes theorem. It is a linear classifier. The common principle shared by this collection of algorithms is the strong independence between the features. One of the algorithms in the collection is the Gaussian naïve Bayes, which assumes that continuous features are normally distributed [11].

c) XGBoost classifier: XGBoost, like random forest, belongs to the family of ensemble algorithms [12]. It combines the learning rate of multiple algorithms. The result is an aggregated value of several algorithms.

3) Evaluation Methodology:

a) Training: Training involves providing a labeled dataset as input to an algorithm to learn a model, or function, that associates the sample features with the class label. The results of a classification algorithm depend on how well it is trained. We used different values for the meta parameters of each algorithm we tested. For random forest, we used different values for number of estimators used to formulate the rules, i.e., the number of trees. Since the amount of data we are working with is large, small number of trees yielded results with very low accuracy. When the number of trees was increased to 1000, it produced considerably better results. Increasing the number of trees above 1000 did not have much impact on reducing the error. We tested both the Gini index and entropy as potential criteria for building the decision trees within the forest. We found that the Gini index produced better results when compared to entropy.

b) Evaluation Metric: As suggested by P. Tao [13], we considered weighted F1 score as a metric for evaluating the effectiveness of the classification algorithms. F1 score conveys a balance between precision and recall. Since, the data had imbalanced distribution of classes, we calculated F1 scores for the three models to compare the performance of each model and determine the model that produced the best results, in general.

c) Cross validation: To ensure the learned models are generalizable, we used k-fold cross validation when evaluating each method. In k-fold cross validation, the data are split into k sections and each section, in turn, is used for testing a model that is trained using data from all the other sections. Results from each tested fold are then averaged to produce the final F1 score for the experiment. In our experiments, we used a value of 4 for k, i.e., in each experiment we used 25% of the data for testing and 75% for training, and the final result is the average of the results for each of the 4 test folds.

4) Evaluation Results: Tables I and II show the classification results of different models we learned using different criteria and numbers of estimators. Results show that random forests tend to perform better, capturing the important features in the data, compared to the XGBoost models. Also, random forests do not require much specific tuning when compared to some boosting algorithms and avoid over-fitting the model, whereas boosting algorithms like xgboost over-fit the model in some cases. We have 11000 records for training. We found that random forests perform better with smaller chunks of data and xgboost works better with large amounts of data.

TABLE I
RANDOM FOREST RESULTS FOR DIFFERENT PARAMETERS

Estimators	Criterion	F1 Score
100	Entropy	0.592
500	Entropy	0.621
500	Gini	0.677
1000	Gini	0.718

TABLE II
XGBOOST RESULTS FOR DIFFERENT PARAMETERS

Learning rate	Max Depth	Estimators	F1 Score
0.1	2	100	0.43
0.1	4	500	0.49
0.3	5	1000	0.56

B. Regression

1) Input Data: We used network DHCP logs to derive network throughput at different APs. These logs have the following information: client User name, client IP address, client MAC address, association time, vendor, AP name, radio type, device name, map Location, SSID, profile, VLAN ID, protocol, session duration, policy type, and average session throughput (Kbps). We used the average session throughput (Kbps), which was further aggregated over each hour, for training our regression models. Every AP thus had 24×30 average throughput values, corresponding to each hour in a day for a month. The data were then split into train and test sets, and models were learned from the training set and used to make predictions on the test data.

2) Models: We used three machine learning models in our project, and their performances were compared based on their mean absolute error (MAE) scores to find the most effective algorithm suited for solving the regression problem.

a) ARIMA: The Auto Regressive Integrated Moving Average (ARIMA) method is very useful for analyzing and predicting time series data. Its input is a univariate time series. The AR in the ARIMA acronym stands for auto-regression, denoting the dependency between an observation and a number of lagged observations. The integrated (I) component refers to making the time series data stationary by differencing of observations, which is a neccesary component for accurate predictions. Finally, MA refers to using a moving average function to model the dependency between an observation and the error for observations that are in lag [14]. Parameters p, q, and d constitute the standard notation of ARIMA and can be defined as the number of lags (p), the degree of differencing (q), and the size of moving average window (d).

We leveraged the code for ARIMA from the Machine Learning website [14] and trained the model by tuning teh p, q, and d parameters. Parameter p is deduced by looking at the autocorrelation function of the time series data. Parameter q is deduced by looking at the partial correlation function. Finally, the number of lags going beyond the critical range was used to assign an appropriate value for q [15].

For different values of parameters p, q, and d, their respective mean absolute error values for the AP with ID KNG-M-M30-1 are listed in Table III. Additionally, Fig. 3 shows an example of actual values vs. predicted values using the ARIMA model for a randomly chosen location, the AP ID KNG-M-M30-1.

TABLE III
ARIMA RESULTS

p	q	d	MAE
5	1	1	324.423
4	1	1	326.857
6	1	3	323.839

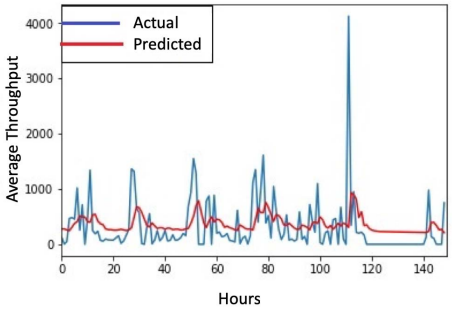

Fig. 3. ARIMA prediction results.

b) Facebook prophet: Prophet is used for forecasting time series data based on an additive model. Holidays, weekly, monthly, and yearly trends can all be incorporated in this model. It automatically removes outliers and fills missing data.

It can be easily integrated into R and Python projects, which are the most commonly used programming languages in data science projects. The Facebook prophet model is single variate, using as input only the network throughput values and their associated time stamps.

The MAE value for AP KNG-M-M30-1 using the best prophet model we trained was 332.20057. Additionally, Fig. 4 shows the actual vs. predicted values for the prophet model.

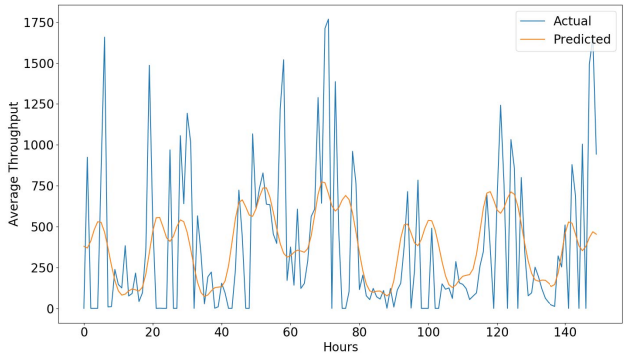

Fig. 4. Facebook Prophet prediction results.

c) RNN-LSTM: RNN-LSTM is a machine learning model extensively used for time-series prediction problems. To train this model, we aggregated average throughput values for each AP for a month on an hourly basis. The aggregated values were then scaled in the range $[0, 1]$ to nullify the effect of large values on the prediction.

We split the data for each AP as 0.1% of total length of the dataset for testing and 0.9% of total length of the dataset for training. For training the model, we used a window size of 4 time steps, with the independent variable containing the previous 4 hours of average throughput values and the dependent variable containing the next 4 hours, including the 5th hour. We used 32 for the batch size when training this model. Hence, the shape of each batch fed to the model is 32 records, 4 time steps and 1 feature (average throughput).

We varied a number of the model parameters, such as the number of LSTM layers, the number of units in each cell of the LSTM layer, and the number of iterations and epochs, and evaluated the models using MAE for each time step. Table IV shows results from our experiments with the RNN-LSTM model. Additionally, Fig. 5 depicts the actual and predicted values of average throughput measured in Kbps for a randomly chosen location.

TABLE IV
RNN-LSTM RESULTS

LSTM units	Epochs	Iterations	MAE
100,100	40	200	420.06
50,50	40	215	347.60
20,25	40	200	289.44

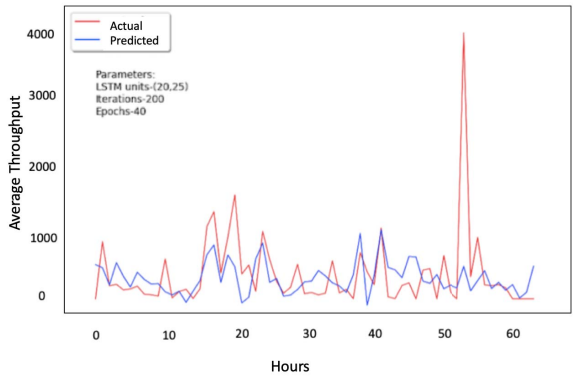

Fig. 5. RNN-LSTM prediction results.

C. Client Implementation

The mobile application was designed using Android studio. It is currently available in the Google Play store to be installed on android devices. The web application was developed using technologies such as Python Flask, HTML, jQuery, JavaScript and CSS to provide users a geographical visualization of APs, their availability, and their work or leisure classification.

1) Client Components: Below are the components of the user interface of our mobile and web based applications.

a) Date and time pickers: The launch page of both the web and mobile based applications provide users with a date-time picker widget to view availability and type of usage of APs for the selected date and time.

b) Map: APs in the MLK library and engineering building are displayed on a map using the Google Maps API. Coordinates of APs in the respective buildings were recorded and are fed to the Maps API to place markers at appropriate locations.

c) Markers: Each AP is represented with a marker on the map. When clicking each marker, details about the AP, such as its name, predicted throughput, and type of activity it was used for by the majority of users are displayed. Color codes are used to differentiate highly available and busy APs used for work and leisure activities. A legend on the map describes the color codes used for differentiating intensity and type of activity performed using each AP.

2) Mobile Application: The mobile application contains an interactive date and time picker widget. After selecting a given date and time, the user is shown a set of markers on a Google map. Each AP is color coded based on their network availability and type of usage. When selecting a marker, details of the AP are shown in the marker info window.

3) Web Application: In this section, we explain the design of our project's Web application.

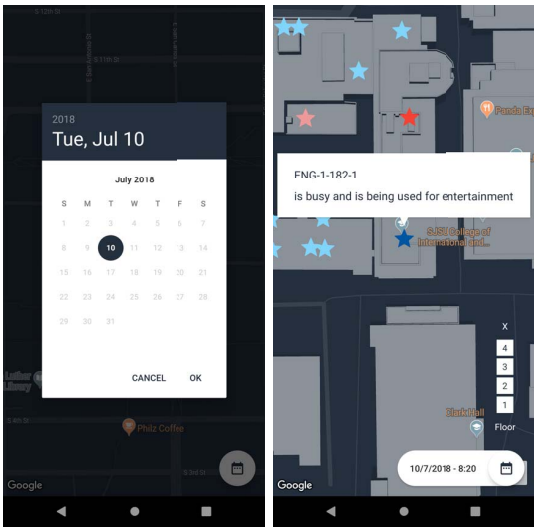

Fig. 6. Mobile application – date picker and busy marker in the leisure category.

a) Launch page: The user can select a preferred date and time for which he/she wants to know the availability of the APs suitable to perform work or leisure activities.

b) Maps page: After the user submits the request on the launch page, the user is directed to a page which displays the type of activity performed and availability of APs in SJSU's MLK Library and Charles Davidson Engineering buildings using Google Maps, as shown in Fig. 7. If the chosen date is in the future, the application uses the output from the classification and regression models to display predicted usage. Past actual values are stored in a MySQL database and queried as necessary to obtain the number of users using APs for work and leisure activities. Queried or predicted results are converted to a JSON string and sent to the application front-end. There, results are grouped and displayed accordingly on the user interface.

VI. Benchmarks and Performance

A. Benchmarks

1) Classification Models: In his study, George Forman [16] took several algorithms into consideration for text classification, including Naïve Bayes and random forest. He found that the random forest models performed better than alternatives, resulting in higher precision.

2) Regression Models: Zhuo et al. [9] showed that LSTM provides better accuracy for a time series data. Additionally, Feng and Shu [7] showed that Neural Network predictor performs better than other popular network traffic predictors such as ARIMA, FARIMA.

B. Performance

In this section, we describe the performance of the classification and regression models we learned and draw conclusions for their usefulness for the task of study location recommendation.

Table V shows the best F1-score performance of each of the classification models we compared. Moreover, Table VI shows the average MAE of the regression models we compared.

TABLE V
Evaluation Results of Classification Models

Models	F1 Score
Random forest	0.718
Gaussian naïve Bayes	0.673
XGBoost	0.565

TABLE VI
Evaluation Results of Regression Models

Models	Average Mean Absolute Error
RNN-LSTM	63.350
ARIMA	110.961
Facebook Prophet	122.137

We can deduce, from the comparative evaluation we performed, that the RNN-LSTM model performs better than the ARIMA and Facebook prophet models. Although RNN-LSTM performs the best among the 3 methods, the predictions are not accurate. Unfortunately we had only about 30 days of continuous data which we used for training and testing the models. We leave as future work executing the 3 models with more data, after appropriately tuning them.

VII. Conclusions

In this paper, we described a system we built for study area recommendation based on predicting the busyness and activity type of campus network APs. Based on the average MAE scores of the regression models, we found that the RNN-LSTM model was more effective than the ARIMA and Facebook prophet models for predicting AP average throughput values. F1-scores of classification models showed that the random forest model outperforms the Gaussian naïve Bayes and the XGBoost classification models for deciding the activity type. Therefore, in our system, we used RNN-LSTM as our regression model to predict average throughput for up to 4 hours in the future and the random forest classification model to classify URLs as work or leisure. Based on these models, we designed a Web based application and a mobile app that recommends users a study area on SJSU's campus. Our application takes advantage of mobile phone features (for localization) as well as cloud-based processing for machine learning inference.

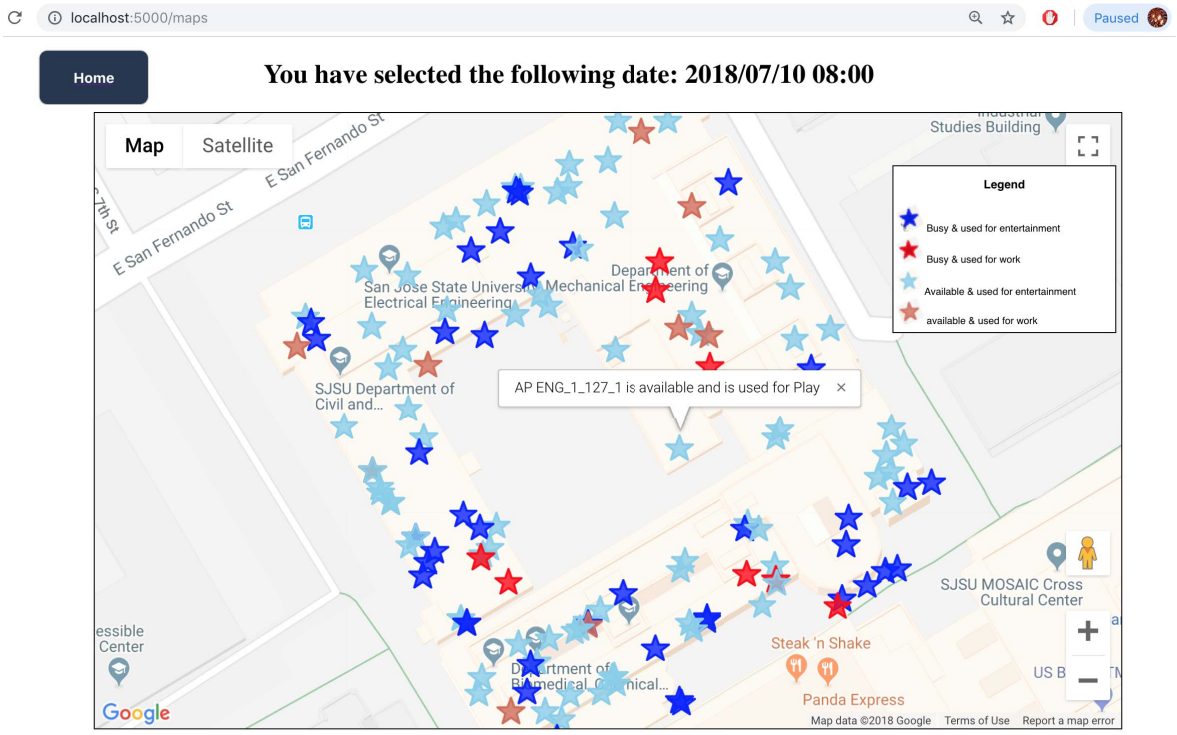

Fig. 7. Web Application – user interface – Engineering building access points.

REFERENCES

[1] M. Shafiq, X. Yu, A. A. Laghari, L. Yao, N. K. Karn, and F. Abdessamia, "Network traffic classification techniques and comparative analysis using machine learning algorithms," in *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, Oct 2016, pp. 2451–2455.

[2] M. Yuantian, R. Zichan, P. Lei, Z. Jun, and X. Yang, "Comprehensive analysis of network traffic data," *Concurrency and Computation: Practice and Experience*, vol. 30, no. 5, p. e4181, 2017, e4181 cpe.4181. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.4181

[3] D. Prangchumpol, "Improving the performance of network traffic prediction for academic organization by using association rule mining," in *Fourth edition of the International Conference on the Innovative Computing Technology (INTECH 2014)*, Aug 2014, pp. 93–96.

[4] Y.-H. Wen and T.-T. Lee, "Fuzzy data mining and grey recurrent neural network forecasting for traffic information systems," in *IRI -2005 IEEE International Conference on Information Reuse and Integration, Conf, 2005.*, Aug 2005, pp. 356–361.

[5] T. H. H. Aldhyani and M. R. Joshi, "Integration of time series models with soft clustering to enhance network traffic forecasting," in *2016 Second International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, Sept 2016, pp. 212–214.

[6] T. Eterovic, S. Mrdovic, D. Donko, and Z. Juric, "Data mining meets network analysis: Traffic prediction models," in *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, May 2014, pp. 1479–1484.

[7] H. Feng and Y. Shu, "Study on network traffic prediction techniques," in *Proceedings. 2005 International Conference on Wireless Communications, Networking and Mobile Computing, 2005.*, vol. 2, Sept 2005, pp. 1041–1044.

[8] D. Kang, Y. Lv, and Y. y. Chen, "Short-term traffic flow prediction with lstm recurrent neural network," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Oct 2017, pp. 1–6.

[9] Q. Zhuo, Q. Li, H. Yan, and Y. Qi, "Long short-term memory neural network for network traffic prediction," in *2017 12th International Conference on Intelligent Systems and Knowledge Engineering (ISKE)*, Nov 2017, pp. 1–6.

[10] C. Olah. Understanding lstm networks. [Online]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[11] R. W. Talbot, C. Acheampong, and R. H. Wicentowski, "Swash: A naive bayes classifier for tweet sentiment identification," in *2013 25th Chinese Control and Decision Conference (CCDC)*, 2015, pp. 4287–4290.

[12] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[13] P. Tao, H. Yi, C. Wei, L. Y. Ge, and L. Xu, "A method based on weighted f-score and svm for feature selection," in *2013 25th Chinese Control and Decision Conference (CCDC)*, May 2013, pp. 4287–4290.

[14] J. Brownlee. How to create an arima model for time series forecasting in python. [Online]. Available: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python

[15] Abishek. How to identify arima p d and q parameters and fit the model in python. [Online]. Available: https://www.youtube.com/watch?v=bqvZL8Ww3aA

[16] G. Forman, "An extensive empirical study of feature selection metrics for text classification," *Journal of Machine Learning Research*, vol. 3, pp. 1289–1305, 2003.