Twitter Analysis of Road Traffic Congestion Severity Estimation

MINOR PROJECT I

Submitted by:

Apurva Agarwal (9915103007)
Gaurav Singh (9915103252)
Kshitij Jain (9915103122)

Under the supervision of
Mr. Sanjeev Patel

Department of CSE/IT
Jaypee Institute of Information Technology University, Noida

September 2017

ACKNOWLEDGEMENT

I would like to place on record my deep sense of gratitude to Mr. Sanjeev Patel, Professor, Jaypee Institute of Information Technology, India for his/her generous guidance, help and useful suggestions.

I also wish to extend my thanks to classmates for their insightful comments and constructive suggestions to improve the quality of this project work.

Signature(s) of Students

Apurva Agarwal	**(9915103007)**
Kshitij Jain	**(9915103122)**
Gaurav Singh	**(9915103252)**

ABSTRACT

Road traffic congestion is one of the problems in large cities. While Twitter has become a major means for communication and message dissemination among Internet users, one question that arises is whether Twitter messages alone can suggest road traffic congestion condition. This paper proposes a method to predict traffic congestion severity level based on the analysis of Twitter messages. Different types of tweets data are used to construct a C4.5 decision tree model for prediction, including tweets from selected road-traffic Twitter accounts, tweets that contain road-traffic-related keywords, and geo-tagged tweets whose volume suggests large crowds in certain areas. In the training that used tweets data of one of the top congested roads in Bangkok, density of tweets has high information gain in classifying congestion severity level. Via a web application, the model can provide a traveler with an estimate of the congestion severity level of the road at every 30 minutes. Such advance congestion information can help the traveler to make a better travel plan. In an evaluation using 10-fold cross validation, the model also shows good performance.

TABLE OF CONTENT

List of Tables **Page**

List of Figures

Abbreviations

1)API: Application Programming Interface

2)OWL: Ordinary Wizarding Levels

3)SWRL: Semantic Web Rule Language

4)CCTV: Close Circuit Television

5)QGIS: Quantum GIS

6)JSON: Javascript Object Notation

Nomenclature

1)Weka Wrapper Library: *Python-weka-wrapper* allows you to use Weka from within Python. The library uses the Javabridge library for starting up, communicating with and shutting down the Java Virtual Machine in which the Weka processes get executed. Python-weka-wrapper provides a thin wrapper around the basic (non-GUI) functionality of Weka (some plots are available using Python functionality). You can automatically add all your Weka packages to the classpath. Additional jars can be added as well.

2)C4.5: It is an algorithm used to generate a decision tree developed by Ross Quinlan. C4.5 is an extension of Quinlan's earlier ID3 algorithm. The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. Authors of the Weka machine learning software described the C4.5 algorithm as "a landmark decision tree program that is probably the machine learning workhorse most widely used in practice to date".

3)J48: It allow you to predict the target variable of a new dataset record, Decision tree J48 is the implementation of algorithm ID3 (Iterative Dichotomiser 3) developed by the WEKA project team.

Chapter 1: Introduction

Nowadays social networking services are widely used by Internet users for communication, e.g. sharing information and personal status, publicizing events, news reporting etc. Twitter, Facebook, and Instagram are some of the popular social networking services, ach providing APIs for retrieving social network data. By nature, Twitter messages are more public and come in large volume with high velocity. Therefore, via the easy-to-access APIs, Twitter messages become a large data source for analysis of socially valuable information, such as social behavior, current public interests, public opinions on particular topics, occurrence of events etc.

As road traffic congestion is one of the main problems in large cities including Bangkok, road traffic information is made available through several authorized sources (e.g. government agencies, traffic reports on radio and television channels, traffic websites, and map services). Those mainly use traffic data that are captured at the locations from traffic cameras and report the current condition of the traffic. Since Twitter users tweet to spread information, we aim to answer one question, i.e. can Twitter messages alone suggest road traffic congestion condition? Our assumption is that different types of tweets data might be useful to predict congestion severity:

1) There are a number of Twitter accounts that report current road traffic condition, i.e. @js100radio, @fm91trafficpro, @traffy, @weather_th, and @longdotraffic.

2) Some users tweet to tell their followers about current road traffic congestion condition, either to share or even to complain about the condition.

3) Some Twitter user's check-in their current locations when they go to different places, and the volume of these geo-tagged tweets can indicate large crowds, and effectively current traffic volume, in certain areas.

Such Twitter usage motivates us to study whether tweets alone can be used to construct a prediction model for congestion severity. Specifically, if the model can predict congestion severity level of a road at some point in the future, rather than at current time, a traveler could better plan his/her travel. If the traveler knows that the traffic will be, for example, less congested in half an hour, he/she could wait until then before setting off on a drive. This paper proposes a method to predict traffic congestion severity in the next 30 minutes using tweets from selected accounts, tweets containing traffic-related keywords, and geotagged tweets. The prediction model is a C4.5 decision tree that was trained by tweets data of one of the top congested road in Bangkok (i.e. Pahonyothin Road). The attributes of the training data are:

1) Day of week

2) Hours of day

3) Minutes of hour

4) Tweets density

Via a web application, a user can view an estimate of the congestion severity level of the road at every 30 minutes.

Chapter 2: Congestion Severity Prediction Method

The overview of the traffic congestion severity prediction method is depicted in Fig. 1. The method consists of 3 steps

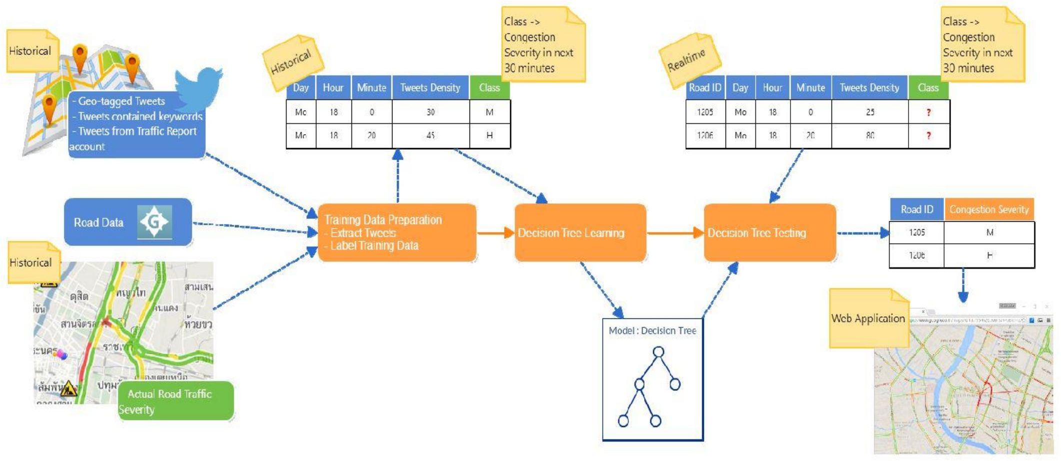


Figure 1. Overview of the method.

A) Data Collection

We collected data from three sources, i.e. road data, Twitter data, and actual road traffic congestion data. They were processed and stored in a MongoDB database as in Fig. 2.

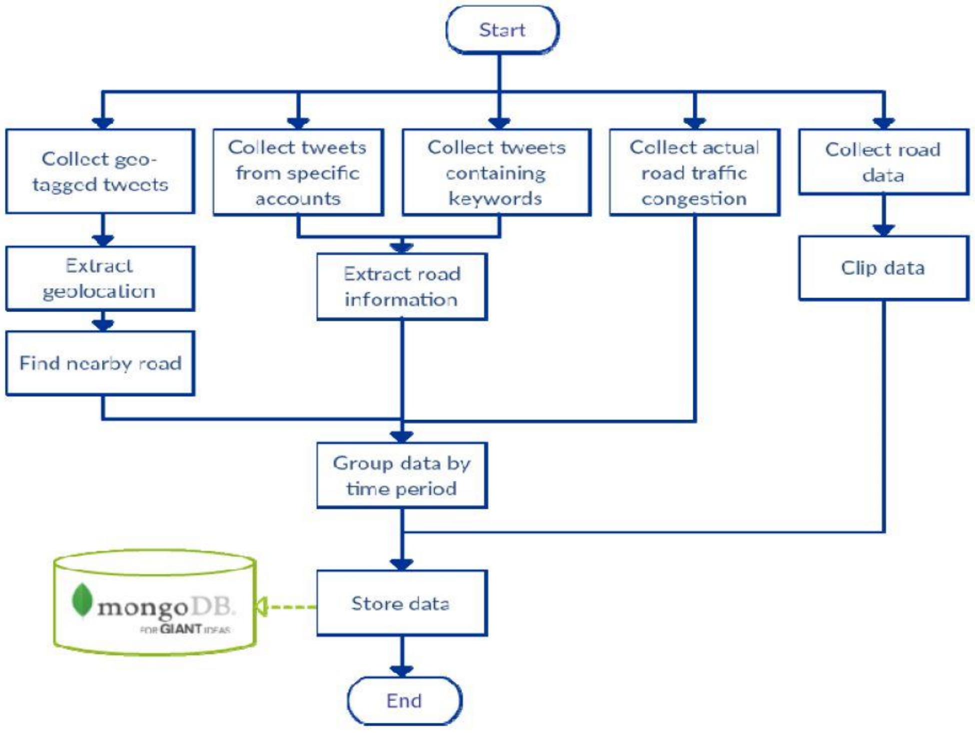

Start
Collect geo-tagged tweets
Collect tweets from specific accounts
Collect tweets containing keywords
Collect actual road traffic congestion
Collect road data
Extract geolocation
Find nearby road
Extract road information
Clip data
Group data by time period
Store data
mongoDB
End

Figure 2. Data collection.

1) Road Data

First, we collected road data from Geofabrik which provides Thailand OpenStreetMap data. Then, we used QGIS application to select (or clip) only primary, secondary, and motorway types of roads in Bangkok, using the spatial query
tool of QGIS, as these three road types are the types of the actual road traffic congestion data. We exported the selected data to the GeoJSON format and then stored them in a MongoDB collection. Finally, we created 2d sphere index in the geometry field of the collection. This collection is called the Road collection.

2) Twitter Data

We wrote three python scripts to collect tweets via Twitter Streaming APIs. We collected data for 30 days during 5 January – 5 February 2016, and then processed and stored them in MongoDB.

• To collect geo-tagged tweets in Bangkok boundary, first, we used the statuses/filter endpoint of the API. We passed the values '100.329071, 13.49214, 100.938637, 13.95495' which represent Bangkok boundary as the location parameter to the endpoint.

Second, we filtered out tweets that were automatically tagged by the devices because the tag is polygon and we could not find suitable reference centroid from that.

Third, we extracted geolocation from each tweet by getting the coordinates property of the tweet.

Next, we used the $near operation of MongoDB to find which road from the Road collection is within 1,000 meters of the tweet geolocation.
Then, we extracted the timestamp_ms property of each tweet to get its hour and minute. We put tweets whose minute value is in [00, 29] in one group and [30, 59] in another group.

• To collect tweets from the selected Twitter accounts, first, we used the filter endpoint of the API. We passed the Twitter account IDs of @js100radio, @fm91trafficpro, @longdotraffic, @weather_th and @traffy, which report traffic information to their followers, as the follow parameter to the endpoint.
Second, we used Lexto to tokenize texts in tweets in order to extract road information. Since Lexto uses a dictionary to perform its task, we appended road names from our Road collection to the default dictionary of Lexto. We used only the tweets that we could find related road names. Next, we grouped remaining tweets by their timestamps for every 30 minutes in the same way as we did with the geo-tagged tweets. Then, we stored these processed tweets in a MongoDB collection called the Follow collection.

• To collect tweets containing keywords related to traffic congestion, first, we used the filter endpoint of the API. We passed the values "Traffic jam, Accident, Road, Traffic, etc" as the track parameter to the endpoint. We collected 12,293 tweets in this step.

Next, we extracted road information and grouped these tweets by their timestamps for every 30 minutes in the same way as we did with the geo-tagged tweets. We used only the tweets that we could find related road names. Then, we stored these processed tweets in a MongoDB collection called the Keyword collection.

3) Actual Road Traffic Congestion Data

We got the actual traffic congestion severity data from The Intelligent Traffic Information Center Foundation (iTIC) and Longdo Traffic. (These data would be used later to label the classes of the training data.) Congestion severity is categorized into three levels, i.e. L (low), M (medium), and H (high). Note that the actual road data are finer-grained than the road data in our Road collection because the actual congestion data were collected for "links" of roads, whereas the road information in our Road collection were collected for "roads" since traffic information in tweets mostly were associated with roads, rather than links of roads. Also, the actual congestion data for the links were collected at every 30 minutes. We put these actual data in a MongoDB collection called the Congestion collection.

B) Model Construction

To construct the model, we prepared the training data from all the collected data above. An example of the training data to predict congestion severity level (H, M, L) are shown in Table I. There are four attributes, i.e. 1) day of week 2) hours of the day 3) minutes of the hour, and 4) tweets density.

DAY	HOUR	MINUTE	TWEETS DENSITY	CLASS
TUE	H8	M0	71	H
TUE	H6	M0	23	M
TUE	H2	M30	3	L

Table I

As described above, we extracted the timestamp_ms property of each tweet and put the tweets in groups by their timestamps. For example, the first group of the training data in Table I refers to the tweets that occurred on Tuesday during 8:00-8:29 (or H8:M0) whereas the third group refers to the tweets that occur on Tuesday during 2:30-2:59 (or H2:M30). Note that "H" was appended to hours of day and "M" to minutes of hour to make them nominal values. The tweets in all three collections (i.e. Geo-tagged, Follow, and Keyword collections) which are belonged to each group those were counted, as shown in Fig. 3, to obtain tweets density of the group.

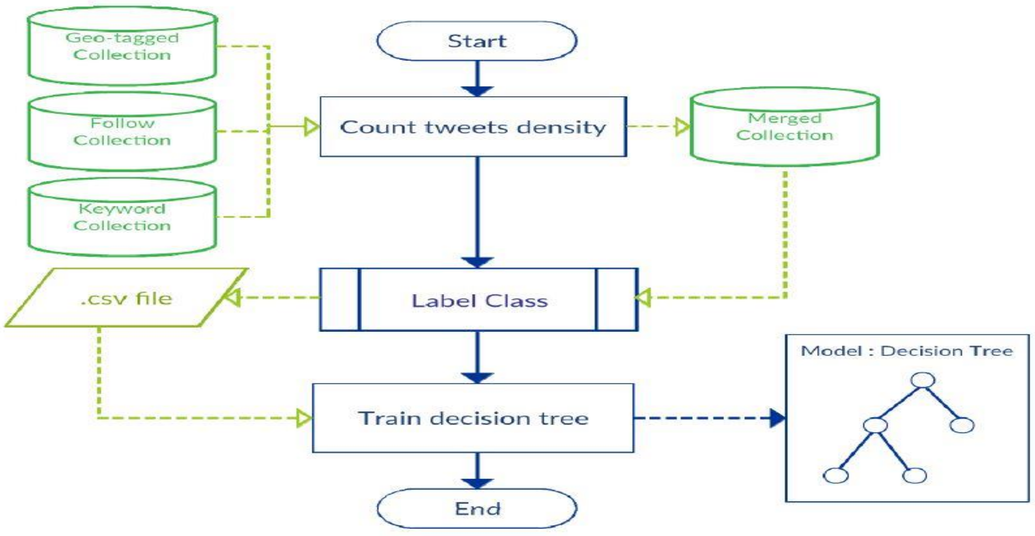


Figure 3. Training data preparation and model construction.

To do so, we assigned different weights for the tweets from different collections based on how strong they represent traffic information. That is, tweets from the selected Twitter accounts concern actual traffic condition, whereas the geo-tagged tweets are merely tweets with geo-tags and may not be about traffic condition. Additionally, tweets with traffic-related keywords may or may not give information about congestion condition. We hence assigned the maximum weight of 10 to each tweet in the Follow collection, medium weight of 4 to each tweet in the Keyword collection, and minimum weight of 1 to each tweet in the Geo-tagged collection. Then, we counted tweets density of each group by calculating the total weights of the tweets in the group. For example, if there was one tweet from each of the three collections which belong to the same group Mon H8:M0, tweets density for this group would be 15. We selected a road with the highest tweets density for training, i.e. Pahonyothin Road, in this case. All tweets in each group were merged and stored in the Merged collection.

Next, to label a class of congestion severity (H, M, L) to a group of tweets, the process is shown in Fig. 4.

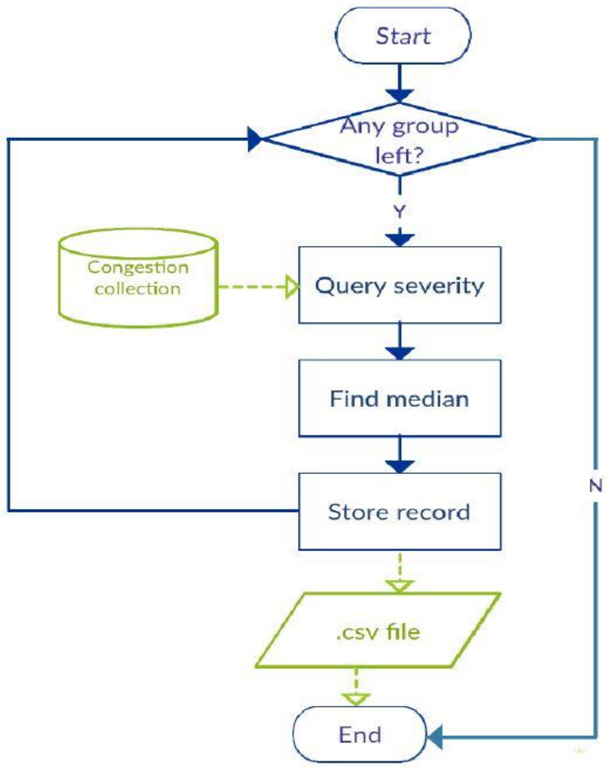


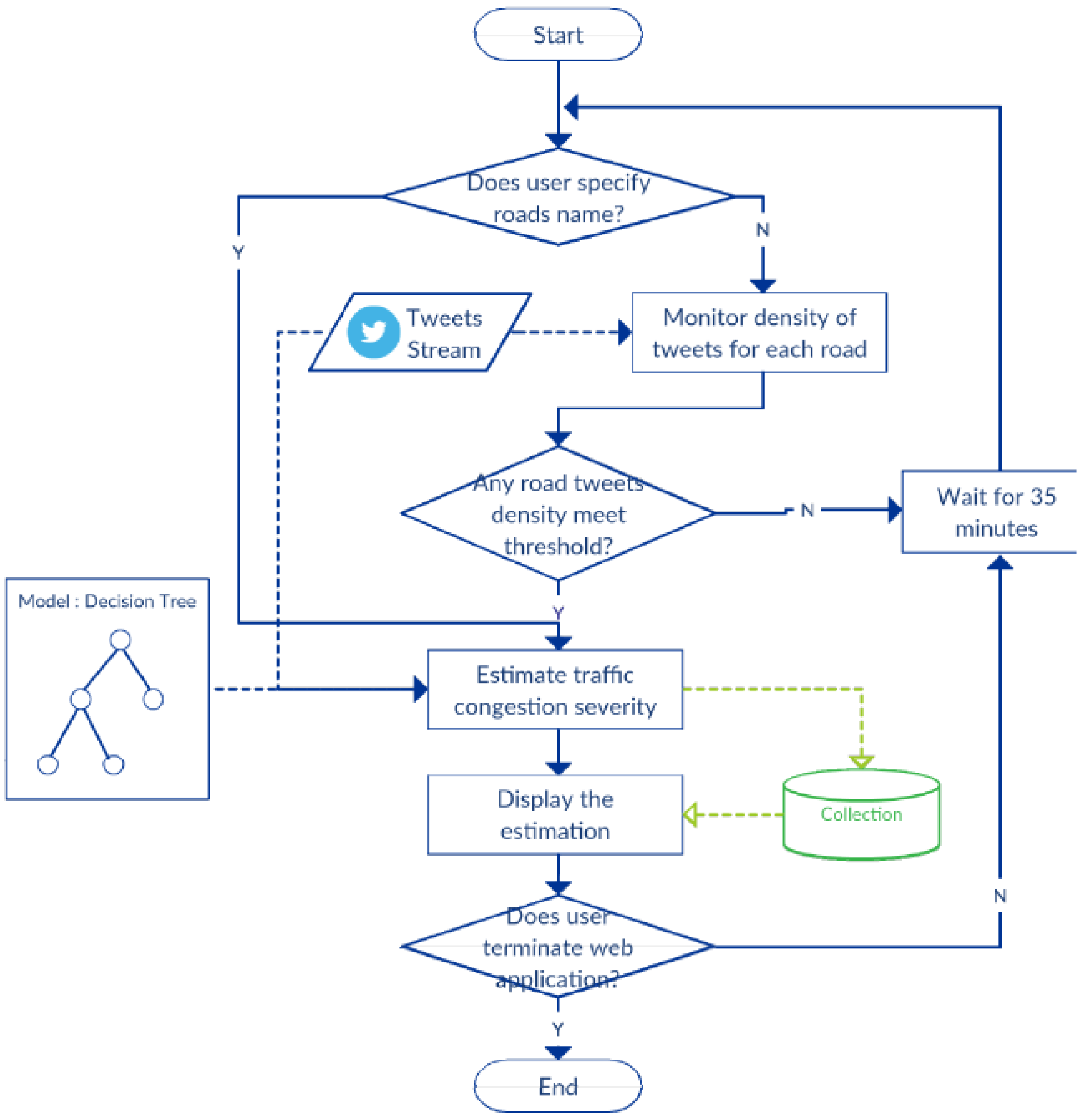
Figure 4. Labeling training data with congestion severity class.

For each group of tweets, we queried congestion severity of Pahonyothin Road from the Congestion collection using day of week, hours of day, and minutes of hour of the next 30 minutes after the timestamp of the group, as a query condition. For example, for the group Tue H8:M0 (8:00-8:29), we queried the congestion severity of Tue during 8:30-8:59. Since Congestion collection stored congestion severity level of each link of the road, so the query returned several congestion severity levels for different links of the road during 8:30-8:59. Therefore we had to find a representative value to represent the congestion severity level of the whole road. As the severity level is in a nominal scale, we chose the median value. After that, each group with a labeled class was stored in the .csv file of the training data.

After we got the .csv file, we used the Weka 3 tool to construct a C4.5 decision tree model. Since Weka 3 implements C4.5 in its J48 decision tree, so we used J48 pruned tree for our model. When Weka 3 finished training the data, we evaluated the model using 10-fold cross validation (see Section IV). After that, we exported the result as a model file to use with our web application.

C) Web Application

To use the model, we developed a web application that can estimate congestion severity level of the road. The Django web framework was used with the Python-Weka wrapper library to deal with the model file from the previous step. Fig. 5 describes the process of the web application which comprises two parts, i.e. backend and frontend.



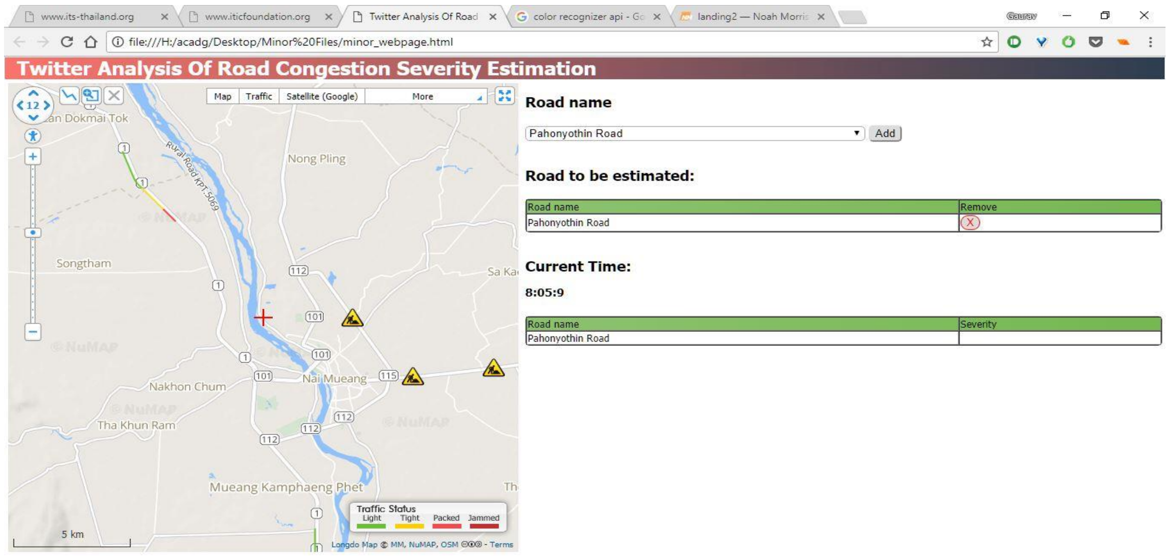
Start
Does user specify roads name?
Y
N
Tweets Stream
Monitor density of tweets for each road
Any road tweets density meet threshold?
N
Wait for 35 minutes
Y
Model : Decision Tree
Estimate traffic congestion severity
Collection
Display the estimation
Does user terminate web application?
N
Y
End

Figure 5. Use of model in web application.

• **Backend Design:** we create scheduled tasks to collect tweets from selected accounts, tweets containing traffic-related keywords, and geo-tagged tweets, and process them in the same way as the training data. While the scheduled tasks work, we inspect tweets density for each road. If any road has its tweets density equal to 10, we will test its attributes with our model and then store the predicted result in a MongoDB collection for the frontend to display. (Note that this tweets density threshold is defined to reduce tweets processing.) We use the Python-Weka-wrapper library to call necessary Weka methods for prediction. The prediction will be executed every 35 minutes because the model can predict the congestion severity in the next 30 minutes, plus 5 minutes for checking tweets density with the threshold.

• **Front End Design**: we offer an option for a user to specify a road name. If a road name is specified, the web application will display only the predicted congestion severity for that road. But if the user does not specify a road name, the application will display congestion severity of all roads whose tweets density values meet the defined threshold described earlier in the backend design part. Also, the web application will be refreshed every 35 minutes if the user does not terminate the application.

Fig. 6 shows the UI of the web application. Roads data and their congestion severity estimation results are displayed on map controls whose base map is Google Maps, in the Map section. Different colors represent different severity estimation results, i.e. red for “H”, yellow for “M” and green for “L”.



Twitter Analysis Of Road Congestion Severity Estimation

Road name

Pahonyothin Road ▾ Add

Road to be estimated:

Road name	Remove
Pahonyothin Road	ⓧ

Current Time:

8:05:9

Road name	Severity
Pahonyothin Road	

Requirement Analysis

Software:

Python, MongoDB, Weka-Wrapper Library, Twitter API, Twython library.

1. **Python**: Programming language that let you work quickly and integrate systems more efficiently.
2. **MongoDB**: It is an enterprise-wide data management solution. Designed for Big Data.
3. **Weka-Wrapper Library**: It makes easy to run Weka algorithms and filters from within Python. It offers access to Weka API using thin wrappers around JNI calls using Javabridge package.
4. **Twitter API**: Python package for accessing twitter's Rest API and streaming API. Supports OAuth 1.0, OAuth 2.0 authentication and it works with latest python 2.x and 3.x branches.
5. **Twython**: Python wrapper of the Twitter API 1.1. It provides an easy way to access twitter data.

Hardware:

PC (minimum requirements: CPU 2GHz, 4GB RAM)

References

1.Twitter
https://twitter.com/

2. L. Burks, M. Miller, and R. Zadeh,
"Rapid estimate of ground shaking intensity by combining simple earthquake characteristics with tweets,"

3. F. Lecue, R. Tucker, V. Bicer, P. Tommasi, S. Tallevi-Diotallevi, and M. Sbodio
"Predicting severity of road traffic congestion using semantic web technologies,"

4. N. Wanichayapong, W. Pruthipunyaskul, W. Pattara-Atikom, and P Chaovalit
"Social-based traffic information extraction and classification"

5. Geofabrik GmbH, OpenStreetMap Contributors, Thailand OpenStreetMap data,
http://download.geofabrik.de/asia/thailand.html/

6. MongoDB, Inc., MongoDB,
https://www.mongodb.org/

7. Twitter Inc., The Streaming APIs,
https://dev.twitter.com/

8. Weka 3
http://www.cs.waikato.ac.nz/ml/weka/