Detection of Public Transport Problems using Location Extended Community Based Sensing

Bas Stottelaar
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
b.g.b.stottelaar@student.utwente.nl

ABSTRACT
Social media is getting a bigger part of our life. Not only is it important for our personal life, also companies use it to follow their customers as a part of their (defensive) marketing strategy. In this paper, a proof of concept is presented to retrieve and filter messages about public transport from Twitter and analyze them to get a view of problems, like delays, failures or overcrowded trains, per region and time. With this method, public transport companies can get an insight on what is happening on their tracks. During this research, 31526 messages were recorded from Twitter, where 3629 relevant messages were extracted. These messages were classified by sentiment, content and type of transport. A J48 decision tree classifier was used to generate a classifier for type of transport (91% accuracy), type of problems (68% accuracy) and sentiment (78% accuracy) for 726 labeled instances. This decision tree was used to classify the other instances in software. The conclusion, social media can be used to monitor tweets about public transport and automatically label them with acceptable accuracy, thus drastically reducing the number of incoming messages.

Keywords
Community based sensing, social media, public transport, problems, service grading, weka, j48, twitter

1. INTRODUCTION
People share a lot of data online, sometimes more than in real life[9]. When they share data, it is of course, recorded in a big database. You could consider people as all-round sensors and social networks as the communication medium. People collect data and share data via these networks with other people. Pervasive computing has entered the world of the streets in the form of mobile phones in pockets and backpacks. Cuff et al. call this transition urban sensing[4]. It is believed that all the recorded data from mobile phones are an excellent data source for research[9][1].

Not only do people use social media, companies also use social media to listen to costumers as part of their marketing strategy. Listening to customers can help making a company better. With the customer as a feedback sensor for a product, one can evaluate its service. But for public transportation companies with a lot of daily users, not only is the message relevant, also its location. For example: somebody shares the message ‘Bus X is too late again’, which is not very informative. But if we know it is about Bus X at location Y, we know a lot more, especially if Bus X seems to be late multiple times. This could be an indication for the company to inspect this incident further. Retrieving messages is simple, but not every message about public transport is relevant. An employees might spend a lot of time searching for relevant messages. Retrieving the location from a relevant message is not always possible. Fortunately, for a set of messages, it might be able to extract the location from the context.

The result of this research could be used to give public transport companies an insight in their service per region by gathering information from a wider, more specific and context aware approach. Not only could delays be detected, but also how people think about the transport service, as an alternative to time taking surveys and questionnaires. This way, companies can react quicker to problems.

1.1 Problem statement
With over 300.000 train travelers each day, the NS is one of the biggest transport companies in The Netherlands. However, not everything goes well. The NS claims that 94.7 percent[20] of the trains were on time in the year 2011. With between 4000 and 4500 trains per day, this means on average, around 200 trains per day are delayed,
do not ride or have other problems. Travelers do not only complain about delays. Many other problems could be the source of complaints, like overcrowded trains, no trains or other failures. And it is not only limited to train transport.

Knowing what is happening is very important to improve ones service. During the snow period in February 2012, the NS and Prorail received many complaints because of the snow, a high number of trains passed out. During the first day, no one knew what was happening and where the problems were and when they first occurred[19]. In this case, a lot of feedback is required to evaluate what happened. Getting feedback from customers is not easy. From time to time, questionnaires are executed to get insight in the current complaints from customers. However, this only gives a view of the current situation and about the topic of the questions. It is also very likely that when someone takes a survey, there are no problems or complaints at the time of questioning. It is also expensive to design, execute and evaluate questionnaires, even if they are automated.

A solution may be to use social media, because it is an excellent tool to get insight in all complaints from travelers. People share what (and how) they think, whenever they want, wherever they are. This idea is not a new idea, and has been used in other researches. For example, Davies et al. investigated the sentiment of people using Twitter[5] and presented a application that colors the states of America by happiness. With mobile devices, GPS can also be used to track Twitter messages. Figure 1 shows how the Dutch train net is visualized using geographically enabled tweets containing the word ‘trein’. This tool was created by Tjong Kim Sang[21]. The big problem remaining, is that for a given set of messages, how do you get the relevant messages and from a given relevant message, how does it know if it is about public transport? Somebody may only share a message that he or she has to wait again, which could give a clue on he or she is waiting for the bus or train, but it could be about something else too. If it knows the person is waiting at or near a train station, it might be more certain the person is waiting for the train. Secondly, if it knows it is about public transportation, one can map it to a location to make it relevant.

Two approaches that are possible:

- Some users share messages via Twitter and check-in via Foursquare. If a reasonable link can be found between these two (for example, within 5 minutes between the messages), it may be possible to determine the Twitter message location with the check-in from Foursquare.
- Derive location from the message itself, like the mentioned train station, train number and look up the location.

The objectives of this research will be threefold. First, it aims at the filtering of data to retrieve the relevant messages from all messages. Secondly, an attempt is made to retrieve all relevant meta data like location, time, nearest station or track etc. At last, it will classify data and different statistics are gathered for possible visualization and analyses.

1.2 Research questions

From the above introduction and problem statement, the following research questions are derived:

1. Can a social media message be used to determine a person’s position?
2. Is it possible to use social media to give an insight in the quality of public transport?
3. What kind of categories can be found in tweets about public transport?
4. Can an automated tool be created to monitor social media about public transport?

2. METHOD OF RESEARCH

The main source of messages is Twitter because it is free to use, widely used and offers an easy streaming API to interface. With Twitter, users can share in 140 characters what is up their minds. It has been shown that data from micro blogging sites like Twitter can be used as an early warning and outbreak detection system[17]. Another service that will be used is Foursquare. This is a social-driven check-in service?? that allows users to earn points and a rank in return for their current location. These check-ins contain geographical data. Currently, there are over 1.0 million Twitter users in The Netherlands sending over 5.0 million tweets per day, from which approximately one percent of the tweets are recorded with geographical data[10]. The number of Foursquare users in The Netherlands is unknown.

The first step is to fetch data from Twitter and Foursquare. Both websites offer an API to retrieve data easily and real time. As as the second step, the data should be filtered. First, non-dutch messages will be filtered via an approach similar to Tjong Kim Sang’s method[21]. Keyword match and location match will be used to filter relevant messages from irrelevant messages. Lampos et al. [11] propose a way to pre process each message: tokenisation, stemming, stop-word and name removal. They also used Google Sets way to pre process each message: tokenisation, stemming, stop-word and name removal. The list of categories can be found in section 3.1.4. The list has been derived from a small trial-and-error pre-research using Twitter. Again, the resulting data set must be verified against a manually labeled data set. For the types of transportation, it is expected that most data will address train or bus, which is the main transportation in The Netherlands. Therefore, the focus will be at bus and train transportation.

The next step to do is to extract location information from the data. This will be easy for the tweets and check-ins who are appended with geographical data. For tweets without location data, we should deduce it. With all data available, it should be possible to design a heuristic to give an insight in the quality of a station or bus stop. The messages will then be saved to a database and can then be queried by time, region, sentiment, type of problem and type of transport. All the programming will be done in Python, since it is a very flexible language.

Special attention should be paid to the privacy of users. One can state that gathering location data of an user is not allowed, others could state that the users itself shared his or her location. For this research, only data that is publicly available will be used. Any references to a user will be removed.
3. THE CONCEPT

3.1 Approach

3.1.1 Literature
Before any work was done, literature was consulted to conquer the hardest problems.

Sentiment analyses
In the search for problems in Twitter messages, sentiment analyses is involved to determine if a message is negative, positive or neutral. Davies et al. presented a language independent method[5] by using Naive Bayes models. They show two key ingredients to implement in a model for sentiment analyses. The first one is to incorporate geographic information. For a general topic, it is more likely that two or more users post a message that may be of the same sentiment. The second ingredient one is to also look at emotion icons. This is also confirmed by Read[18].

Unfortunately, both studies focused on random data, while this research focuses on specific, filtered data. Also, the geographic information is not always available. It was decided to focus on keyword matching extended with emoticons and keywords specifically about public transport (for example '#ns' refers to the railroad company NS, but it is mostly used in a negative context). A basic list of high potential keywords was fetched from Gietelink[8], who conducted an experiment with blog posts back in 2007.

Text mining
Text mining is not easy, because in contrast to normal data mining, the features (or attributes) are not yet determined. Also, text can be interpreted in several ways. For example:

1. 'Het duuuuuuurt lang voordat de trein komt!!!!'
2. 'Amsterdam was egt lagge'

Take the first item. The word ‘duuuuuuurt’ is being extended by the letter ‘u’ to emphasize the word. Keyword matching would fail if we would search for ‘duurt lang’². Therefore, words like above are trimmed by removing obsolete letters.

With item two, more understanding of the Dutch ‘SMS’ language is required. Almost 70% of the messages (see section ??) is posted from a mobile device, which ‘automatically’ implies spelling mistakes and shorter alternatives for longer words to stay below the 140 character limit[2]. The correct sentence for case two would be: ‘Amsterdam was echt lachen’. Because the complexity of this problem, it was decided to only take care of the basic spelling mistakes in the keywords used to track messages from Twitter and to extract features.

As the last step, the text is tokenized for interpretation. One of the methods is proposed by O’Connor et al. and tokenizes messages by hashtags, @-replies, abbreviations, strings of punctuation, emoticons and unicode glyphs[16]. In this concept, only hashtag, @-replies, emoticons are used, extended by companies mentioned and railroad stations.

3.1.2 Data gathering
The data is acquired via Twitter’s streaming API. This allows real-time access to posted Twitter messages matching specified keywords. With this API, one can track up to 400 keywords. If one or more words are found in a tweet, the tweet will be returned. For example:

- ‘storing’ will match ‘storing’, ‘STORING’, ‘%ToR-\textquotesingle ng’, ‘#storing’ or ‘@storing’
- ‘storing’ will match ‘stroomstoring’, ‘seinstoring’ but also ‘internetstoring’
- ‘te laat’ will match ‘De trein is te laat’, but not ‘Ik ben laat’

It can be seen, words are not context sensitive and may even return unrelated results like ‘internetstoring’³. Therefore, the list of keywords had to be designed carefully.

1. List of all possible tracking words, including flaming words - 40 words in total
2. As list one, but without the flaming words - 26 words in total
3. As list two, but only focusing on words directly related to public transport - 16 words in total

In appendix A the keywords for the three lists can be found. The lists were constructed by hand, based on a trial-and-error approach with the use of Twitter Search. The three lists were all tested between 8:00 AM and 12:00 AM of working days to see which list of keywords delivered the most accurate and useful keywords. The average results per 10 minutes are below in the table 1.

<table>
<thead>
<tr>
<th>Date</th>
<th>List 1</th>
<th>List 2</th>
<th>List 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>May, 8th</td>
<td>3469</td>
<td>250</td>
<td>133</td>
</tr>
<tr>
<td>May, 9th</td>
<td>3021</td>
<td>178</td>
<td>99</td>
</tr>
<tr>
<td>May 10th</td>
<td>155</td>
<td>73</td>
<td>29</td>
</tr>
<tr>
<td>Percentage in/out</td>
<td>0.011</td>
<td>0.068</td>
<td>0.038</td>
</tr>
</tbody>
</table>


Table 1. Keyword effectiveness (average number of tweets per 10 minutes)

The second list was chosen as the best list, not only because of the higher percentage of used messages, but also because the first list matched more flaming words, and the first list on average was very negative. The second list includes more irrelevant tweets about public transport, but the keywords are much broader. Third list was too specific. The percentages do not say anything about the filters. The numbers are heavily dependable on the moment of the day they were recorded.

Unfortunately, the Twitter Streaming API limits the rate[22] of messages returned and IP’s connected. Therefore, if Y messages are tracked by keywords and Y exceeds the 1% messages per second Twitter offers, Y - 1% messages are not returned. During the data gathering, this became an issue. Partner companies of Twitter offer the complete stream (called the ‘firehose’) for enterprises in return for a payment. As an alternative, the Twitter Search API could be used. However, this API is limited to 150 request per hour, is not real-time and more processing is required to filter duplicate tweets.

In total, 3629 filtered messages during a period of six days. In total, 31526 messages were received. An total of 20782 messages were posted from a BlackBerry, Android or iPhone smartphone, identified by the Twitter client used.

³Dutch ‘internetstoring’ for ‘internet failure’ Dutch words were used for the example because in the Dutch language, composite words are concatenated to each other instead of separated by spaces.
3.1.3 Filtering

Each message retrieved from the Twitter streaming API was processed via a filter chain as described by Lethbridge et al.[13]. A simplified version of this chain is drawn in figure 2. At each stage, a filter can decide to throw a message away. The messages will then not be processed any further. Two stages record the messages to the database. The first saves all Dutch messages for simple playback and the last one records all messages that meet the requirements.

In the first stage, basic processing is executed. Since there are a lot of auto generated, non-human Twitter messages about delays, problems e.g., they must be removed to get accurate results. These type of messages are posted as a result of events, and this research tries to predict the events. Removal can be easily fixed by comparing the sender to a blacklist of non-human user accounts. Another thing that is an important aspect of Twitter, is retweeted messages. This are messages that are simply re-broadcasted by other users. While the fact that a user has retweeted another message is likely to classify to the same class and generate incorrect statistics. Retweets appear in two ways: classic retweets and modern retweets. The first appear in the format of ‘RT [@originaluser] [message]’, while the second one contains a property indicating it is an retweet. The filter decides to remove retweets by looking at the format and the property.

The second stage is more important, but also simple. Each tweet contains information about the user who posted the message. By comparing the preferred language of an user to Dutch, we know the language of the message. Any other tweets are disposed.

The fourth filter tries to add more information to a tweet via three methods. Each method is less accurate, and if one method succeeds, the other methods are skipped and the message is passed down. Also, if none of the three methods succeed, the message is disposed.

Twitter messages can contain geographical information, from where a message was posted. A list of all Dutch railroad stations was fetched from the NS API[15], including the GPS location of a station. The Vincenty Distance formula was chosen as to calculate the distance between the tweet GPS location and each station’ GPS location. This formula has an accuracy of 0.5mm[23] and takes into account that the earth is not perfectly round. This formula may seem over-accurate for this application, but it has great implementations available. The station closest to the tweet is marked as a candidate, and if the candidate station was within a range of 3.0 kilometers, it is more certain that the tweet was posted near a station. The same process is repeated for the Dutch railroad tracks, which can be downloaded from OpenStreetMap. However, since a track consists of multiple lines per track, each segment for each track was matched to the tweet’s location, and the closest track within a range of 1.5 kilometers is chosen. The implementation is naive and not optimized, but on average, it takes seven seconds on a 2.4 Ghz Intel Core2Duo notebook to match a GPS location to all stations and tracks available. Even at this speed, it is still a good candidate for real time usage. The range of 3.0 kilometers and 1.5 kilometers was chosen to compensate mobile phone GPS accuracy in buildings and in moving trains or busses.

Users mention their current location or their destination in a message very often. For example, see the three example tweets below.

1. ‘Rotterdam centraal wachten op trein’
2. ‘In #veld nu wachten op overstap.’
3. ‘In de trein naar enschede! #vrij’

The place name method (see figure 2) scans messages for place names or station names. Then, the name of the station and its location is appended to a message. However, this approach has some major drawbacks. Only in the first two examples, we know (almost) for sure that the sender is at the station he or she mentions (Rotterdam Central or Velp). However, in the third example, we only know that the person will arrive at station Enschede, but we cannot tell where he or she is at this moment. It can be seen, the context of the message tells how accurate the location is. If this fails too, the last method is to use Foursquare. Some users link their Foursquare check-in service to their Twitter. When they check-in, a message like below will be posted (note the mix of English and Dutch).

1. ‘I am at Weer voor de spoorweg wachten op nen vertraging bij een trein pffff http://t.co/SkGgNvVR’

All check-in messages are of the same format: ‘I’m at [user message] [URL to Foursquare]’. When we follow the URL, the location of a check-in can be fetched. Unfortunately at this moment of time, of all retrieved messages, only a few messages were posted via Foursquare, making it not worth the time needed to implement this method.

![Figure 2. Simplified filter chain used to process tweets. Messages start at the top and may be rejected at each stage (not shown).]
**Attribute filter**

The most important step of the filter chain is the attribute filter. In this filter, data is extracted from the gathered data as attributes for the classifier. The list of attributes extracted from a message can be found in appendix B. For each problem, mentioned in section 3.1.4, a list of possible keywords related to that class is constructed. This list can be found in the appendix. The same counts for sentiment analyses. As mentioned before, keyword matching is used to match tokens. In general, if a specific keyword was found, its attribute flag is set to true. For sentiment analyses, each unique match of a positive word increases the sentiment by one, each negative word decreases the sentiment by one. To take into account spelling mistakes, the Levenshtein Distance is used for matching words. This function calculates the amount of differences between two words. At most, one different letter (or one mistake) was allowed.

One might ask what the difference is between the tracking keywords and the keywords used in the attribute filter. There is little difference in the words, except that we want as many as possible messages via a broad list of words. Then, to extract context of a message, we match them by category. Together with features like sentiment, during peak hours, day of the week etc., they can be classified.

**3.1.4 Possible classes**

By trial-and-error, the following list of classes was created. The Twitter search was used with the keywords from appendix A. For optimal performance, it is required that a class is common.

1. Type of transport:
   - Bus
   - Train
   - Unknown

2. Type of problems (or situations):
   - Delays
   - Waiting for transport
   - (Technical) problems
   - Overcrowded busses or trains
   - Temperature (hot or cold)
   - Other situations
   - In bus or train
   - Irrelevant

3. Message’ sentiment
   - Positive
   - Neutral
   - Negative

**Creating the classifiers**

The WEKA Toolkit[7] is used to analyze data and create a classifiers. For this tool, the J48 is chosen. This classifier is the open source implementation in WEKA of the commercial C4.5 classifier with pretty good performance [6][24][14]. This classifier generates a decision tree, which means that at each node, there are a few possibilities to choose from. A decision tree is easy to implement into software by the use of if-then-else statements.

The J48 classifier allows several parameters to be configured. However, changing the parameters did not influence the accuracy much. Therefore, the default parameters were used to construct the tree. WEKA has several modes to train the classifier. The default is cross validation, where the data set is split into N parts, and N rounds are executed with N - 1 parts are used for training, and 1 part for validation. The default value for N is 10.

With the above settings, the three classifiers in figure 3, 5 and ?? are generated.

![Classifier for the first class.](image)

When a message is classified as ’Unknown’ with the first classifier, it is automatically classified as ’Irrelevant’ with the seconds classifier. All three classifiers are independent of each other, which means that given a class, this class is not used to be classified with another classifier.

**Training the classifiers**

For creating, training and testing the classifier and the attribute filter, 726 randomly chosen instances (exactly 20% of all gathered instances) were labeled manually for all three classes. Initially, only 10% of the instances were selected, but the number of manually labeled classes was not evenly divided.

Below are the confusion matrices for the three different categories above. The column refers to classified as, the rows refer to the actual class. An accuracy of x % means that x % of all instances are labeled correctly.

**3.2 Discussion**
The three constructed classifiers tell us something about the kind of tweets. For example, figure ?? show that hashtags in combination with company names are mostly associated with negative sentiment. This is also confirmed in figure 5 at the bottom of the diagram.

However, there are still some improvements possible and challenges ahead. For example, consider the tweets below.

1. 'En de volgende trein gemist. Vreselijk laat op het station en toen vertrok de trein ook nog van een ander perron. Vast de warmte:)'

2. 'Erg veel zin om met de trein te gaan. Er zal vast geen storing zijn.'

If we look at the first tweet, we notice a few things about it. First, it is about missing a train. Secondly, it mentions the problem and then, it ends with a suggestion. The type of problem classifier is based on keyword matching, and in this case, it will say it is about the temperature and missing the train. But the classifier can only select one class. Also, for the sentiment analyses, the smiley ':)' is matched. This incorrectly matches this message as more positive the sender actually meant. The second tweet contains sarcasm, which might be easy to detect for a human, but not for the computer. Therefore, the second message may be incorrectly classified as positive.

Building and training a classifier normally requires a data set with equal number of instances per class. However, with three different classes this is hard to accomplish. Also if we gather data, it is not likely that (for example with sentiment) exactly 50% of the messages are positive and the other 50% is exactly negative. Sentiment, type of problems and type of transport change over time. Bifet et al. mention this problem, and use the Kappa statistic with a

![Figure 5. Classifier for the third class.](image)
sliding window over the data, to normalize the accuracy of the classifier[3], because of the potential changes in the class distribution over time. In table 5 are the Kappa statistics of the classifiers with a fixed window of the 726 instances, as calculated by WEKA.

| Table 5. Kappa statistic of the three classifiers |
|-----------------|-----------------|-----------------|
| Transport type  | Problem type    | Sentiment       |
| 0.8662          | 0.5919          | 0.6748          |

Landis et al. interpret a Kappa value of 1 as 'the classifier is always right', and 0 as 'the classifier does not agree'[12]. A statistic between 0.61 - 0.80 is considered very reasonable and between. 0.81 - 1 as good. This means that most classifiers are reasonable. Looking at the Kappa statistic or not, this method is used for classification of real time data or huge data sets. This means that if one message might be classified incorrectly, it is most likely that (a part of) the other messages that are similar are correctly classified.

From the current tree for classifying type of problems it can be seen that not all attributes are used. The algorithm to construct the tree looks at a minimum number of instances before an attribute is chosen. The number of instances used to build the tree could be increased (e.g. from 20% to 33% of the data set). This allows the classifier algorithm to incorporate more attributes if it seems that more instances with a specific set of values classify to the same class. Unfortunately, this operation is labour intensive.

4. CONCLUSIONS

4.1 Research questions

In section 1.2 four questions were mentioned. This section will answer the questions. Any references to class letters (for example class a-e) refer to the classes in table 3.

Can a social media message be used to determine a person’s position?

From the data set that has been gathered, it is possible to determine the position of a person.

| Table 6. Sources of localization (number of tweets) |
|-----------------|-----------------|-----------------|
| Geo             | By class a or b | Foursquare      | Others          |
| 291             | 42              | 3               | 3293            |

It can be seen, only 9.25 % of the tweets that are gathered can be directly pinpointed on a map accurately. The other tweets do not contain any accurate reference to a location on a map. This does not mean the other tweets are directly useless. If we compare the number to the approximately geotagged tweets of one percent[10], 9.25% is much better.

Is it possible to use social media to give an insight in the quality of public transport?

For this question to be answered, an heuristic must be designed to think of how quality can be measured. Twitter has proven to be an excellent source for companies to get insight in their service. The following two heuristics are proposed as an answer (the numbers are based on the confusion matrices and the number of instances recorded): By monitoring the above suggested heuristics, one can grade public transport. For example, consider the numbers in table 7 and 8 as a baseline for the six days of recording. If the numbers for the next six days show an explosive rise for some class(es), one could see this as a change in quality and could react to that, if required.

| Table 7. By third classifier with 78.78 % accuracy (number of tweets) |
|-----------------|-----------------|-----------------|
| Positive        | Neutral         | Negative        |
| 650             | 1269            | 1710            |

What kind of categories can be found in tweets about public transport?

Messages from Twitter can be clustered in the following categories. This list was created by a trial-and-error approach and the Twitter Search. An important requirement for a class is that it is common in the search results.

- Delays
- Waiting for transport
- (Technical) problems
- Overcrowded busses or trains
- Temperature (hot or cold)
- Other situations
- In bus or train
- Irrelevant

The class 'Irrelevant' is used for messages not about public transport, while the class ‘Other situations’ is used for any other bus or train problems/events etc.

Can an automated tool be created to monitor social media about public transport?

As part of this paper, a tool was created to monitor Twitter messages and process them. Basically, this answers the question. But the big problem is the number of relevant messages right now. With only 3629 messages passing the filter chain, and 1424 messages marked as irrelevant, 2205 messages are left (see table 8). The number of days in which these messages were recorded was six, which means that 368 messages per day are really relevant. This results in only 15 messages per hour.

One part of the problem is because not all messages are retrieved from Twitter, but in general, the conclusion would be that the Dutch people do not tweet too often about public transport, even though the Dutch people send 5 million tweets per day[10].

4.2 Conclusion

In general, the method in this paper shows it is possible to create a tool to monitor social media for messages about public transport, with pretty good accuracy for the sentiment classifier and the type of transport classifier. The type of problem classifier can perform better, but with almost 70% of accuracy, this means that two out of three messages is still correctly classified. One could argue about messages being missed, but if something big and relevant happens, and people post messages about it, then it is
likely that some messages from other users will be correctly noticed by the classifier.

Apart from the classifiers performance, this method is very useful for companies. The method applied to the data set used during this research, reduced the number of messages from 31526 messages to 2205 relevant messages. This is a reduction of almost 86 %. For workers at companies who monitor social media messages, a lot of boring filtering is already completed.

4.3 Future work

4.3.1 Improved language detection

In this proof of concept, the language of the user is detected by simple keyword matching and the user profile language. However, users are not enforced to change their profile language, resulting in the default profile language of English. There are online services available to detect the language of a message. With this feature implemented, more Twitter messages could be retrieved for processing.

4.3.2 Use the full fire hose

As mentioned in section 3.1.2, the Twitter Streaming API will at most return 1% of the current available messages. This limits us the number of keywords and reduces the accuracy. By using commercial services, more keywords can be used, or the complete fire hose can be used to gather the tweets.

4.3.3 GPS location matching

In this concept, the only focus was at railroad stations and tracks. A map of bus stops was available, but it was simply too large to be handled efficiently. The current (very naive) matching algorithm could be optimized a lot further. There are special database implementations available (like PostGIS) that implement special GPS formulas for easy searching. Then, matching by bus stops could be implemented too.

5. REFERENCES


APPENDIX

A. TRACKING KEYWORDS
The following Dutch keywords were used for tracking Twitter messages. First, a very broad range of words were used. However, these words resulted in a lot of messages not about public transport. The second try removed all bad words from the first list, and changed short words to hashtags (like #ns and #ov). Last, the list was further minimized to only contains words that are directly related to public transport.

A.1 First try
arriva, bus, chauffeur, damn, fuck, genist, godver, godverdomme, in de bus, in de trein, jesus, kak, kut, machinist, ns, op de bus, op de trein, op tijd, opdagen, openbaar vervoer, opgeheven, ov, overvol, overvol, overvolle, pro-rail, seinstoring, shit, storing, stroomstoring, synthus, te laat, tering, tram, tram, trein, veolia, vertraging, wacht, wachten

A.2 Second try
#ns, #ov, arriva, bus gemist, chauffeur, in de bus, in de trein, machinist, niet opdagen, op de bus, op de trein, op tijd, openbaar vervoer, opgeheven, ov, overvol, overvol, overvolle bus, overvolle trein, prorail, seinstoring, storing, stroomstoring, synthus, te laat, veolia, vertraging, wachten op

A.3 Third try
#ns, arriva, bus gemist, in de bus, in de trein, op de bus, op de trein, overvolle bus, overvolle trein, prorail, seinstoring, storing, stroomstoring, synthus, veolia, vertraging

B. MESSAGE ATTRIBUTES
Each message that passes the filter chain, will be attributed with several attributes for the classifier.

- hour of the day
- day of the week
- during peak hours (on working days between 6 - 8 am and 4 - 6 pm
- message starts with hashtag
- message is a reply to another person
- refers to a transport company (by hashtag, or not), like #ns or Veolia
- sentiment of a message (negative or positive integer)
- number of station(s) mentioned
- station closest matched from GPS
- distance to closest station
- track closest matched from GPS
- distance to closest track

C. ATTRIBUTE KEYWORDS
A list of keywords below is used to attribute messages. In some cases, spelling mistakes are intended to compensate user mistakes.

C.1 Referals to transport companies
arriva, connexion, gvb, htm, ns, q-buz, qbuz, ret, synthus, veolia

C.2 Positive sentiment keywords
:), :-), :-d, :d, ;), ;-), (;, ;(-, S=S, aanrijding, bah, bang, bleek, commentaar, dann, dik, dom, dood, doodstraf, draaien, duur, eindeelijk, erger, faal, fail, file, fout, gek, gestrand, gevolg, haat, halelijk, hel, jammer, kapot, koud, kritiek, kut, kwijt, laag, langzaam, last, lelijk, log, lui, mis, moeilijk, moeite, moest, moesten, nauwelijks, nee, nog steeds, onbegrijpelijk, onzeker, oude, pdf, pijn, probleem, risico, shit, slaan, slaat, span, stoppen, storing, strijd, stuk, te veel, te weinig, teveel, velden, veel, vergeet, vergeten, verkeerd, verkeerd, verloren, verplicht, vertraging, wachten, warm, warmte, zeiken

C.3 Negative sentiment keywords
:(, :-(, :-@, ;(, ;-(, :S, =S, aanrijding, bah, bang, bleek, commentaar, dann, dik, dom, dood, doodstraf, draaien, duur, eindeelijk, erger, faal, fail, file, fout, gek, gestrand, gevolg, haat, halelijk, hel, jammer, kapot, koud, kritiek, kut, kwijt, laag, langzaam, last, lelijk, log, lui, mis, moeilijk, moeite, moest, moesten, nauwelijks, nee, nog steeds, onbegrijpelijk, onzeker, oude, pdf, pijn, probleem, risico, shit, slaan, slaat, span, stoppen, storing, strijd, stuk, te veel, te weinig, teveel, velden, veel, vergeet, vergeten, verkeerd, verkeerd, verloren, verplicht, vertraging, wachten, warm, warmte, zeiken, ziek

C.4 Other keywords
The kind of keywords refer to the classes mentioned in section 3.1.4.

To match problems: aanrijding, met de trein, met persoon, seinstoring, verstoring, voor de trein, wisselstoring
To match temperature: airco, gloeiend heet, hitte, te heet, uit te houden, warm
To match overcrowded busses or trains: geen plek, drukte, vol
To match waiting: wachten op, wacht op
To match in: in de bus, in de trein

D. USED SOFTWARE AND SERVICES
For completeness and reproducibility, the list below contains the software used to process the data.

- Python 2.7.1 - main programming language interpreter
- Django 1.4 - web framework and database ORM
- PyCURL 7.21.4 - cURL wrapper for Python to allow asynchronous streaming connections
- GeoPy 0.94.2 - GPS data objects, distance calculations and geocoding
- Shapely 1.2.14 - geometry objects and calculations
- BeautifulSoup 3.2.1 - XML manipulation for NS API
- WEKA Toolkit 3.3.7 - creating and testing classifier

The following services has been used.

- Twitter Streaming API - realtime but limited access to tweets
- NS API - retrieve Dutch railroad stations including geo information
- OpenStreetMap - provided Dutch railroad tracks and bus stops

4From the smiley xD, shattering eyes, big laugh