Final Report

Application of EventMapper Framework for Coronavirus Detection (LITMUS Project)

Name: Neha Pande
GT Id: 903519910

1. Motivation and Objective

Weak signal events cannot be detected and verified easily unlike signal events. This is because they have fewer reports for each event. Detecting them from social media streams is difficult when corroborative event detection is not available. Only a single post is present for such events in more than 50% of the events and 5 or less posts for 85% of the events [1]. Further, due to the presence of noise or surrounding large-signal events they are difficult to identify.

The EventMapper framework integrates corroborative and probabilistic sources for real-time weak-signal detection. Corroborative sources include news articles, historical databases and reporting agencies. Probabilistic data sources include social media sources including Twitter, Facebook, Youtube among others. Social media has been used in the past after-the-event detection for large events. It uses corroborative sources to continuously fine-tune event processing on probabilistic sources and reduce long term performance degradation. The corroborative events are used for ML classifier finetuning. Classifiers are subsequently used for prediction. The EventMapper Framework is robust to concept drift. It has been used for applications such as landslide, wildfire and flooding detection. In this work, the focus is on the extension of the EventMapper Framework for real-time, accurate and global detection of coronavirus.

2. Background

According to global statistics, there have been more than 1.8 million confirmed cases worldwide, 100 thousand deaths and more than 210 countries affected. There have been severe outbreaks in some regions of the world such as the United States, China, Spain, Italy and South Korea. It also has been declared as a pandemic by the World Health Organization. The main reason for concern is that it is spread by asymptotic people and there is no vaccine available yet. It has been observed to be more contagious and has a higher transmission rate when compared to SARS and MERS.

3. Data Sources

There are two types of data sources. These are the Probabilistic and Corroborative sources (high

confidence sources). The advantage of probabilistic sources is that they are a good source for real-time data and have global coverage. These are mainly social media sources. For example, Twitter, Facebook. Using only probabilistic sources is not sufficient. The advantage of using corroborative sources is that they have less noise and high accuracy. So, it is used along with probabilistic sources. Examples of corroborative sources include News Articles, Historical Databases and Reporting Agencies.

Data Characteristics

- Kaggle Dataset on Novel Coronavirus 2019 [Link]
    - Day-wise time series at global level
    - Has data on confirmed cases, recovered cases and deaths
    - Important fields include Id, Province, Country, Day-wise cases
    - Original Source: John Hopkins
    - Update Frequency: 2 days
- Twitter Datastream
    - Complete size is 5GB
    - Important attributes extracted: created_at, id, id_str, text, source, retweeted, retweet_count, timestamp_ms, land, coordinates, place etc

Tweet Example

{ "created_at": "Sat Feb 29 18:59:56 +0000 2020", "id": 1233829273691049984,"text": "Coronavirus will spread in California, health officials say: 'It's already out of the bag' https:\/\/t.co\/YHBt1myH5X #uncategorized #feedly … }

Kaggle Example

Province/State, Country/Region, Lat, Long, 1/22/20, 1/23/20, 1/24/20, 1/25/20 …

Beijing, Mainland China, 40.1824, 116.4142, 14, 22, 36, 41…

4. Technologies

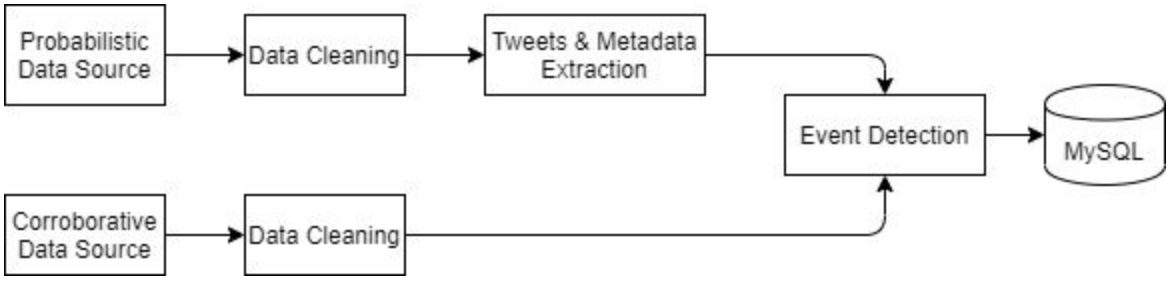
- Database: MySQL
- Python Libraries: pandas, json, pickle, keras, tensorflow, numpy, nltk
- Web Framework: Flask
- User Interface Design: HTML, CSS, JavaScript, Twitter Bootstrap

5. Methodology

5.1 Design

5.1.1 Data flow in EventMapper

Figure below shows the data flow in EventMapper. It shows the integration of corroborative and probabilistic sources for global and real-time physical event recognition.

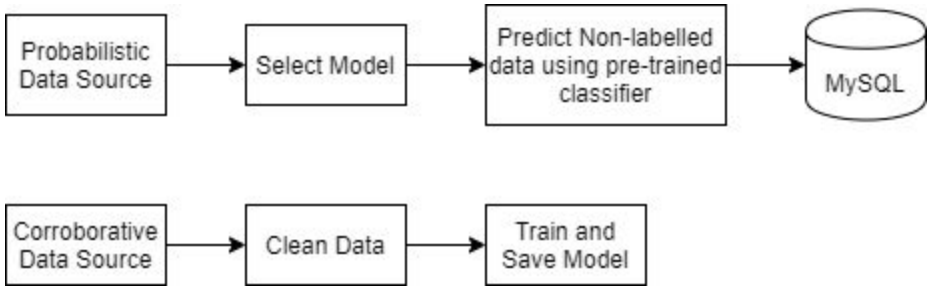


Data cleaning is done for probabilistic sources since they are noisy. Stopwords are used for performing data cleaning on probabilistic sources. It uses most frequent data from the prior irrelevant data to continuously update the stopwords list. The old terms in the list are pruned.

Metadata extraction involves location extraction. It is performed from the text content since only 0.5% of the social media posts have geotag. Named Entity Recognition fails to find locations present in a post's text content. This happens because of high noise in social data. Prior events are therefore used to improve the location extraction.

Corroborative sources are slower and have low coverage because they are obtained from multiple sources and human experts. Probabilistic sources are obtained from social media sources. They are obtained from a large variety of events. Relevant signals need to be identified for detecting the events for which data processing is done.

5.1.2 Event Detection



The figure above shows how the event detection is performed using ML classifier for probabilistic source posts that could not be labeled with corroborative integration.

ML classifier takes a text post with its metadata as input and provides a label of relevance or irrelevant as output. Since the performance of static ML classifiers deteriorates over time, ML classifier maps both corroborative and probabilistic sources to the same grid, then it automatically labels some data points to create training data. If both sources have the same event keywords, it indicates a high likelihood that social media posts are relevant to the event. For all

non-labeled posts with corroborative integration, its relevance with the ML classifier is classified.

5.2 Implementation

This project was implemented in three phases. These were the Heterogeneous Data Integration phase, User Interface and MySQL integration phase and the third phase was the ML Classifiers phase.

5.2.1 Heterogeneous Data Integration Phase

The first phase was the heterogeneous data integration phase. This phase involved three main tasks. The first task was to fetch the data from corroborative and probabilistic data sources. The kaggle data source was chosen as the corroborative source since it had day-wise data and included counts for confirmed cases, recovered patients and deaths occured. It was also updated every 2 days on an average. The data was populated based on statistics present on John Hopkins website. The Kaggle API was used to fetch this data. For obtaining data from Twitter (probabilistic source), keywords related to coronavirus were identified. These are given below.

Twitter Keywords

["2019-ncov", "2019ncov", "Chinesepneumonia", "corona", "coronavirus", "covid19", "covid2019", "epidemic", "hubei virus", "infected pneumonia", "mers-cov", "ncov", "novel coronavirus", "outbreak", "pheic", "quaratine", "sars", "sars-cov-2", "world health organization", "wuhan"]

Also, the important tags to be extracted from each of the tweets were also found. MySQL scripts were written to create tables and populate them in MySQL. Before populating, the data was parsed and cleaned. Location and time attributes were used for combining data from both data sources. The location and timestamp tags present in the tweet were extracted and converted into the standard format before matching it with the province and day attribute in the Kaggle data. If the tweet matched with the corroborative data, it was labelled and a weight was assigned to the tweet. The weight was calculated using values obtained from the tweet tags such as favorite_count, retweet_count, reply_count etc. Both the labelled and not labelled tweets were stored separately in MySQL.

5.2.2 User Interface & MySQL Integration Phase

In this phase, the user interface for the system was designed. The UI consists of a map and a side panel. The side panel implemented the search by province functionality. This lets the user give province names as input and retrieve event details. Markers are displayed on the map for each of these provinces. Information windows are used to display latest statistics for the province such as the number of confirmed cases, number of recovered patients and number of deaths occured.

This lets the user compare the data in one or more provinces effectively. A province and its neighbors can be analyzed to understand how the virus is spreading. A panel on the top of the web page consists of three options - Clear, Delete and Show Markers. Clear Markers hides some of the markers from the current view. The Show Markers displays the hidden markers once again. Delete Markers removes all the markers from the map. To see trends in different countries, the modal present on the side panel can be obtained. Opening the modal displays time series data of the most affected countries. The Flask app is integrated with MySQL. So, data is retrieved from MySQL database to display on Flask User interface.

5.2.3 ML Classifiers Phase

The model was trained using a 3 layer sequential neural net using the keras and tensorflow python modules. The tweets labelled in the heterogeneous data integration phase were used for training the model. The model was saved in hd5 format to be used later for prediction. A sentence to vector model was constructed using word2vec since loading the sentence2vec model was very expensive. The tweets were first cleaned and then the word2vec model was loaded. The average of the sum of all word embeddings was calculated by taking the average of each contiguous word for every bigram word embedding. For predicting the class using the pretrained ML model, some of the data was manually labelled. This was done because the number of labels were found to be insufficient. The pre-trained model created earlier was then loaded and used to predict the class.

6. Challenges

- Change in data format (Kaggle). Old data format has province names but the new data format uses zip codes
- Errors in location fields (province, country) in Kaggle source. For example, spelling mistakes, use of abbreviations, missing fields in data etc.
- Tried using fasttext vectors but found the model very expensive to load
- Used word2vec model and averaged continuous words since sentence2vec was also very expensive to load
- Building classifiers for event detection

Through this project, I learnt some of the concepts in Machine Learning and Natural Language Processing and how to apply them in a software system. I also learnt to work with several python libraries such as Tensorflow, keras, word2vec, nltk and pickle. Overall, working on the project has been a very good learning experience for me since I didn't have any prior knowledge of some of the concepts and skills used in the project.

7. Conclusion

The goal of the project was to successfully extend the EventMapper framework for coronavirus detection. Overall the system performance was functional and achieved the desired intent. This

project can be extended to incorporate data from multiple corroborative sources and using different ML classifiers. The related project files are included with this project report. The README.txt outlines the included files and directories.

Project Demo Video link : https://youtu.be/UivQ1fZoTR4
Project Presentation Video link : https://youtu.be/DZcj1QFaikU

8. References

- Suprem, Abhijit & Pu, Calton. (2020). EventMapper: Detecting Real-World Physical Events Using Corroborative and Probabilistic Sources
- Suprem, Abhijit & Pu, Calton. (2019). Event Detection in Noisy Streaming Data with Combination of Corroborative and Probabilistic Sources
- CDC Coronavirus https://www.cdc.gov/coronavirus/2019-ncov/index.html [Online]
- WHO Situation Reports https://www.who.int/emergencies/diseases/novel-coronavirus-2019/situation-reports/ [Online]
- Keras Model/Tensor flow https://www.dlology.com/blog/how-to-convert-trained-keras-model-to-tensorflow-and-make-prediction/ [Online]
- Keras https://keras.io/applications/ [Online]
- Tensorflow https://www.tensorflow.org/tutorials/text/word_embeddings [Online]
- Nltk https://medium.com/@Intellica.AI/comparison-of-different-word-embeddings-on-text-similarity-a-use-case-in-nlp-e83e08469c1c [Online]
- Pickle https://wiki.python.org/moin/UsingPickle [Online]
- Sentence2Vec https://medium.com/explorations-in-language-and-learning/how-to-obtain-sentence-vectors-2a6d88bd3c8b [Online]