CIRRELT-2023-09

Multilingual Text Classification on Social Media Data for Incident Alert in Subway Transportation Network

Banafsheh Mehri
Martin Trépanier
Yves Goussard

January 2023

Bureau de Montréal
Université de Montréal
C.P. 6128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél : 1-514-343-7575
Télécopie : 1-514-343-7121

Bureau de Québec
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél : 1-418-656-2073
Télécopie : 1-418-656-2624

Multilingual Text Classification on Social Media Data for Incident Alert in Subway Transportation Network

Banafsheh Mehri[1,2,*], Martin Trépanier[1,3] Yves Goussard[2]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Electrical Engineering, Polytechnique Montréal

[3] Department of Mathematics and Industrial Enginnering, Polytechnique Montréal

Abstract. The traditional way of gathering data to alert subway incidents, such as being stuck on a train, electrocutions due to train malfunctions …, is through dispersed sensors or cameras embedded in the infrastructure of transportation networks. However, fixed detectors as the main source of incident detection in the transportation network may not bring all the relevant information. In the past decade, Twitter data have been used as a source of information for many event detection tasks empowering organizations to acquire actionable knowledge. However, the challenge of filtering out the relevant information from large amounts of data is a hindrance to the efficient usage of Twitter data for transportation incident detection. Therefore, a proper content filtering methodology is crucial to filter out noisy information from this massive data. To improve the basic keyword search (as the traditional way of information filtering), many researchers used word embeddings as a type of distributed representation for text that allows words with similar meanings to have similar representations. Nevertheless, with such representations for tweets, multilingual and cross-lingual use of the model is not an option. In other words, scaling to new languages requires new embeddings and the initial model does not allow for parameter sharing. To address the above-mentioned issue, we present a methodology to detect subway-related incidents based on Bidirectional Encoder Representations from Transformers (BERT) applied to a set of tweets extracted from the Twitter API. Our methodology is based on a pre-trained deep bidirectional representation from the unlabeled text by jointly conditioning on both left and right side of each word's context during the training phase which makes these embedings context-dependent. Then we fine-tune the model to the subway incident detection application. Our experimental results show that the performance of the proposed method is competitive with traditional machine learning models whilst carrying out multilingual & cross-lingual tweet modeling.

Keywords: Transportation incident, cross-lingual, multilingual, text embedding, text classification, BERT, Twitter data.

Acknowledgements. The authors gratefully acknowledge the financial support provided by the Natural Sciences and Engineering Council of Canada (NSERC), and the Société de transport de Montréal (STM) for data provisioning.

Results and views expressed in this publication are the sole responsibility of the authors and do not necessarily reflect those of CIRRELT.

Les résultats et opinions contenus dans cette publication ne reflètent pas nécessairement la position du CIRRELT et n'engagent pas sa responsabilité.

[*] Corresponding author: banafsheh.mehri@polymtl.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2023

I. Introduction

SINCE the 2000s the usage of social media platforms has been growing at exponential rates. Social media channels such as Facebook, Twitter, and Instagram can be used as real-time and cost-effective sources of information. Twitter is one of the fast growing social media tools that enables users to post and read short messages. By using the Twitter application on smartphones, users are able to immediately report events happening around them on a real-time basis. The information generated by active users everyday provides a new type of dynamic data source that contains information about various topics.

Mining this open source real-time information offered by eyewitnesses during an incident event can be utilized in the field of intelligent transportation systems (ITSs) to harvest semantic and spatial information for transportation incident detection in order to track and improve situation awareness during the course of an incident. In other words, analyzing the Twitter data can provide us information to characterize occurring incidents as a substitute of an ensemble of sensors and cameras installed in the transportation network. However, there are numerous challenges when considering the usage of Twitter data for transportation incident detection. Some of the main issues include: collecting reliable data with all necessary associated information, translating reported observations into a numeric form suitable to be used with classification algorithms, and filtering out significant amount of irrelevant data.

The conventional method for incident detection using Twitter data is based on keyword filtering. The problem with this approach is that different groups of people might use different words to describe an incident. Hence, the vocabulary set may not include all keywords and the the list of keywords can also be altered over time. Since basic keyword filtering approaches do not provide satisfactory results, many researchers have resorted to statistic and automatic text classification algorithms. However, the text data which is a sequence of symbols cannot be fed directly to the algorithms as most of the algorithms expect numerical representation vectors with a fixed size rather than raw text data with variable length. The most common text representation techniques are simple count of word frequency vectors and Term Frequency Inverse Document Frequency (TF-IDF) vectors, in which, after transforming the tweets into a set of tokens, a type of occurrence measurement is assigned to each token as the feature value.

Although these techniques are commonly used in the literature, they are subject to the curse of dimensionality. Since the tweet corpus is very large and only a small subset of words is used in each tweet, the representation matrix suffers from sparsity with lots of feature values set to zero. This problem can be solved with the usage of statistical feature selection techniques, but it risks eliminating fundamental information, hence lowering the accuracy of the model.

Recently, as a more advanced alternative, many studies have been focused on developing classifiers on top of word embedding vectors such as Word2Vec [1]. Nevertheless, the most concerning issue with these kind of word embeddings is its lack of ability to deal with unseen or out-of-vocabulary words. In the cases where the model do not encounter a word during the training phase, it would not be able to cipher such word into a vector. This would be an issue, in cases such as processing Twitter data where there is a large amount of noisy data, with words that may only occurs once or twice in a massive corpus. Another shortcoming is that this method is a monolingual distributional word vector representation. Using this method, scaling to a new language calls for completely new embedding vectors. This is an issue particularly when the dataset is a mixture of more than one language such as Twitter data from bilingual/multilingual regions around the world.

In order to address the above-mentioned drawbacks, we propose to utilize the bidirectional encoder representations from transformers [2] sequence classification method to enhance the performance of transportation incident detection. BERT is recognized as an open source tool that has achieved the state-of-the-art performance in many Natural Language Processing (NLP) tasks. It carries out multilingual vector representation, therefore it can be used to test our methodology in order to detect incidents using both English and French tweets.

In order to implement this approach, we first extract, transform and aggregate datasets from the Twitter API including historical tweets within a specific region. Next, we manually annotate tweets and prepare labelled data to train a binary transportation incident classification. Then, we perform experiments with monolingual, cross-lingual and multilingual BERT sequence classification. We train the BERT sequence classification model on the dataset containing both languages. We also train BERT on one language and evaluate its performance on another language to put to the test the ability of this model for zero-shot cross-lingual transfer. The model then is trained in a way that shuffles the training data while fine-tuning it for both target languages. We also setup and implement a vast verity of text representation and text classification combinations of conventional techniques applied to our labelled dataset and we compare the performance of new method with them.

The main contributions of this study are:

- The introduction of the Bidirectional Encoder Representations from Transformers (BERT) method in subway incident

detection from social media.
- The study of the concept of multilingual and cross-lingual text classification in subway incident detection by implementing and examining 26 combinations of methods for feature representation and classification.
- The annotation of subway-related incident dataset containing over 32000 records to examine the performance of our approach.

We demonstrate that our approach can help not only improve subway incident detection from social media data compared with the competing models, but also resolve the incapability of conventional models in multi cross-lingual settings.

II. Related Works

By means of user-generated data and the accessibility of this data on social media there are now many prospects for multifaceted studies. In the field of intelligent transportation systems (ITSs) several studies have aimed to extract useful information from social media streams to study and improve various aspects of transportation systems, including user opinions of mobility networks [3], [4], travel behavior modeling [5], trajectory/traffic estimation [6] & [7] and traffic event detection [8]. In this context, the approaches to derive information from informal and unstructured textual data plays a key role in the effectiveness of the decision support systems built using social media data.

Regarding the previous studies related to traffic monitoring and incident detection with Twitter data, most of them are focused on traditional approaches of harvesting information from social media data using a predefined set of keywords.

In this setting, researchers first prepared vocabulary sets related to their research subjects, such as traffic categories [9], traffic conditions [10], traffic incidents [11], [12]. Then they search real-time tweets and select the tweets that contain the keywords in the corresponding vocabulary set and they typically rely on identifying specific keywords to classify tweets or discovering topics of tweets.

However, the quality of the text classification and topic discovery using keyword matching based on predefined keyword sets was unsatisfactory as it did not account for semantic and for the context. In addition maintaining the rules to classify texts by keywords would be a never ending task as the vocabulary evolves with time and different group of people may use different set of words or jargon to describe a situation. Moreover, as this approach is not language agnostic, for each additional language the extra work of determining keywords will add to the tediousness of the process. In addition, misspelling usually occurs in user generated text such as tweets. Nevertheless, as keyword search returns the exact match for each keyword, the problem of misspelled words is unsolved with this approach.

To address some of the above-mentioned shortcomings of simple keyword matching, many researchers used numerical text representations to map tweets to numerical feature vectors that can be processed by more advanced algorithms.

Among studies on extracting travel-related content using Twitter data, the work of [4] in 2010 was one of the earliest that used numerical features for tweet representations. In their research, tweets were tokenized in uni-grams and a SVM algorithm [13] were used as the travel-related classifier. The training set however, was heavily unbalanced with a low percentage of tweets belonging to the travel-related class. The evaluation assessment showed F-measure of approximately 23% which was not significantly better than traditional approaches. D'Andrea et al. [14] also developed a system for the real-time monitoring of several areas of the Italian road networks by analysing the Twitter stream coming from selected areas. They transformed the collected tweets to bag-of-words representations [15]. Then they used the information gain [16] to transform the sparse bag-of-words matrix into a dense matrix. They then transferred the dense matrix to an SVM classifier to conduct tweet classification. Withal, they only fetched and used tweets in Italian language in their study.

Following the same direction, other researchers such as [17], [18], [19], etc, used traditional numerical text representations combined with traditional classification algorithms such as SVM to use the Twitter API as a source of data in transportation studies. However, traditional classification algorithms such as SVM are not suitable for large data sets. Moreover, they do not perform well when the dataset has more noise i.e. target classes are overlapping.

Then Zhang et al. [20] published a study that focused on detecting traffic accidents in Northern Virginia and New York City using social media data. For the classification part of their research they used more advanced methods such as Deep Belief Network (DBN) [21] and Long Short-Term Memory (LSTM) [22]. They reported that DBN performed better than LSTM.

Chen et al. [23] also focused on traffic event detection using Twitter data using more advanced algorithms. They used convolutional neural networks (CNN), LSTM model, and LSTM-CNN with pre-trained word embedding of Chinese microblogs. They reported LSTM-CNN on top of Word2Vec as the most efficient model amongst the model that they used in their comparison. Dabiri and Heaslip [24] also took advantage of word embedding for tweet representations. They then deployed supervised deep-learning algorithms including CNN and LSTM model as the recurrent neural network (RNN) and a combination

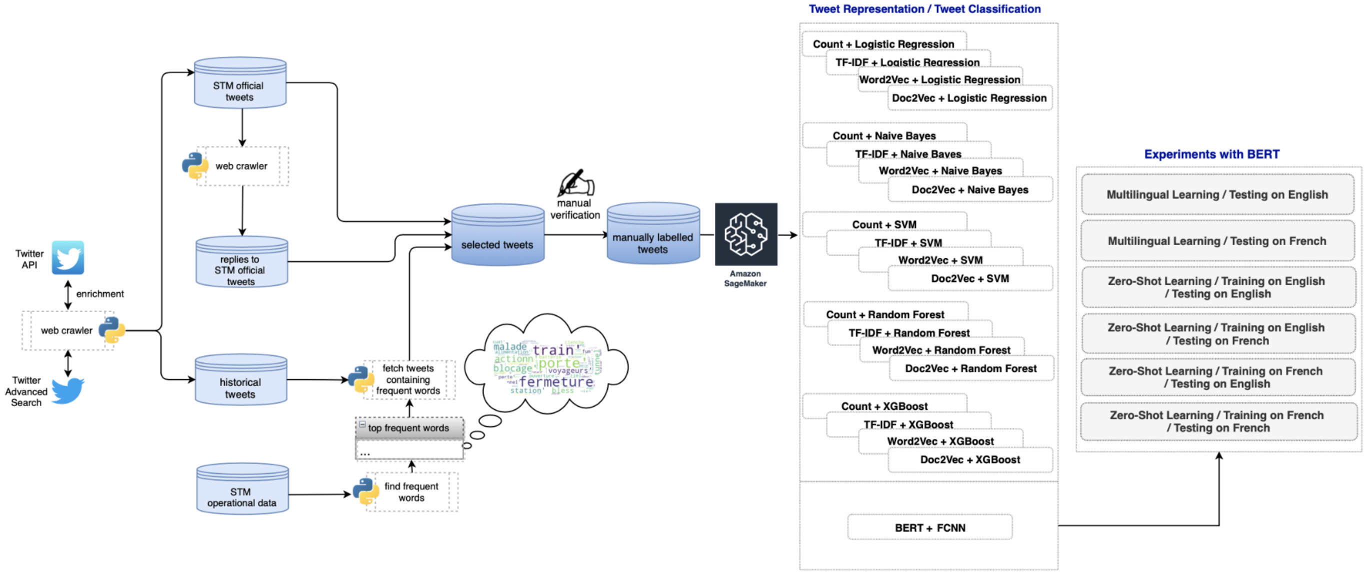


Fig. 1. Methodology Steps

of both models such as LSTM-CNN on top of word embedding models for detecting traffic events among tweets. Their results showed that CNN on top of Word2Vec provided the best accuracy.

Overall, there are a few works in the transportation studies using numerical text representation for tweets. However, these numerical text representations ignore the location information of the word that is an important piece of information in the text. Also, these models do not account for the semantics of the words. For example, words that are often used in the same context would have totally different corresponding vectors that are far from each other in the vector space. Nonetheless, such representations cannot generate vectors for words encountered outside the vocabulary space, hence, does not support out-of-vocabulary words. In addition, non of the studies amongst the few researches using numerical text representations in the literature examines and uses cross-lingual and multilingual text representation for traffic incident detection tasks. Hence, the application of recently developed NLP techniques is not yet largely explored in transportation incident detection.

III. Methodology

Figure 1 shows the steps of applying our methodology to our particular case of transportation incident detection. An automatic approach to collect tweets is followed by a careful selection of labelled data while performing manual verification to ensure data quality. The data collection approach is explained in details in the following section. The incident detection from the text of tweets consists of two core parts of text representation and text classification on Twitter data.
We also perform several experiments in order to investigate the ability of Bidirectional Encoder Representation from Transformers sequence classification in the context of multilingual and cross-lingual transportation incident detection and comparing the performance of this new method against popular traditional natural language processing methods.

We implement several methods and algorithms that we use as baseline models along with the implementation and training of Bidirectional Encoder Representation from Transformers sequence classification for text representation and text classification which are two necessary parts of our process of assigning tags or categories to tweets according to their contents.

The aim of text representation part of this research is to numerically represent the unstructured tweets in order to make them mathematically computable. For a given set of tweets $D = d_i$, i = 1, 2, ..., n where each d_i stands for a tweet, the problem of text representation for tweets is to represent each d_i of D as a point s_i, in a numerical space S, where the distance/similarity between each pair of points in space S is well defined. After converting all the tweets into representation vectors we perform tweet classification to automatically classify the tweet representations and assign relevant tags of incident/non-incident to them. We use Bidirectional Encoder Representations from Transformers (BERT) for text representation. BERT [2] is a contextual word representation model. Its architecture has been built on top of *Transformers* [25] which are bidirectional encoders that scan the whole sequence of textual data to solve sequence-to-sequence tasks while handling long-range dependencies with ease. It is basically pre-trained transformers using a combination of masked language modeling objective and next sentence prediction. Unlike unidirectional embeddings that are commonly generated by scanning forward or backward from the word of interest, bidirectional representations are generated by scanning both sides of the word together which leads the representation to consider the meaning of a word in its context that depends on both the words before and after it.

BERT is pre-trained on 104 languages with a large corpus of unlabelled text including the entire Wikipedia which contains 2,500 million words and Book Corpus with 800 million words. This way, the model learns an inner representation of the languages in the training set that can then be used to extract features useful for downstream tasks and we can train a standard classifier using the features produced by the BERT model as inputs. The model can be fine-tuned to perform text classification with the help of Transformers library.

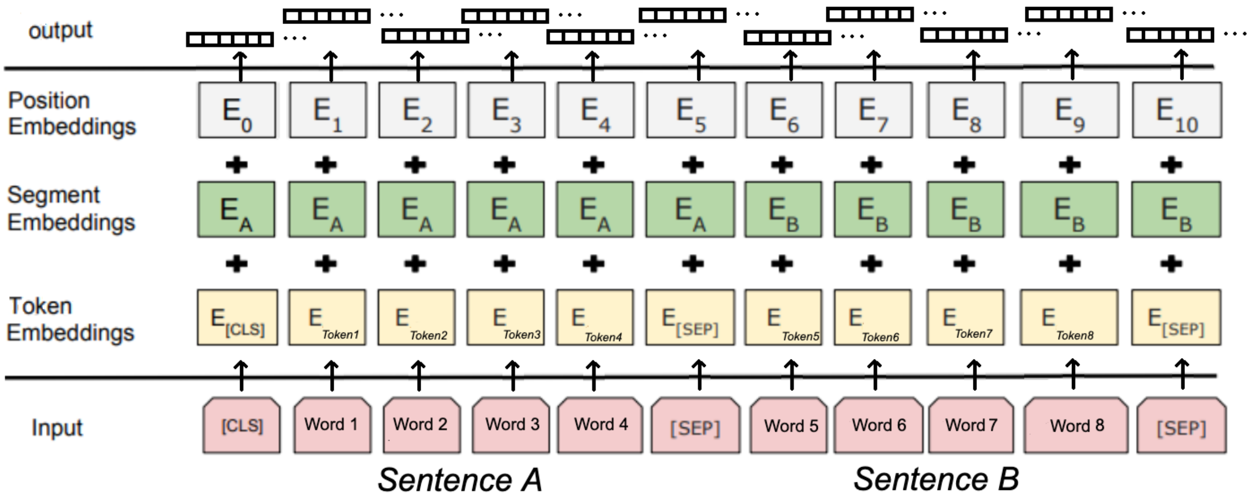

output
Position Embeddings
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10}
Segment Embeddings
E_A E_A E_A E_A E_A E_A E_B E_B E_B E_B E_B
Token Embeddings
$E_{[CLS]}$ E_{Token1} E_{Token2} E_{Token3} E_{Token4} $E_{[SEP]}$ E_{Token5} E_{Token6} E_{Token7} E_{Token8} $E_{[SEP]}$
Input
[CLS] Word 1 Word 2 Word 3 Word 4 [SEP] Word 5 Word 6 Word 7 Word 8 [SEP]
Sentence A
Sentence B

Fig. 2. An illustration of BERT's initial vectors

As shown in figure 2, the initial vector, for every input embeddings is a combination of three embeddings:

Token Embeddings: It is pre-train embeddings for each token based on the **WordPiece**[1] token vocabulary.

Segment Embeddings: It is basically the sentence number that is encoded to a vector, to help the model distinguish between sentences.

Position Embeddings: It is the position of a word within that sentence that is encoded to a vector, to enable Transformer to capture “sequence” or “order” information.

The input representation for each and every token is a summation of the corresponding token embeddings, segment embeddings, and position embeddings. Segment and position Embeddings are both required for temporal ordering since all these vectors are fed in simultaneously into Bidirectional Encoders representations from Transformers; and language modelling needs these orderings preserved. Then the vector representation is passed to Bidirectional Encoders Representations from Transformers which under the hood is a stack of Transformer encoders that scan the entire sequence of words at once and it outputs language model vectors. We then build a model by combining BERT with a single-layer fully connected neural network (FCNN) as the classifier. We train our model and fine-tune it on the transportation incident data in monolingual, multilingual and cross-lingual fashion which we will explain in the next section.

IV. Experiments

In this section, we first explain our data collection, data labelling strategy and we introduce the dataset that we use in our experiments. Then, we introduce the benchmark models, we share the evaluation and the results for different representation and classification methods. Lastly, we compare our method with benchmark models.

[1] WordPiece is a commonly used technique to segment words into subword-level in NLP tasks. The vocabulary is initialized with all the individual characters in the language, and then the most frequent combinations of the symbols in the vocabulary are iteratively added to the vocabulary.

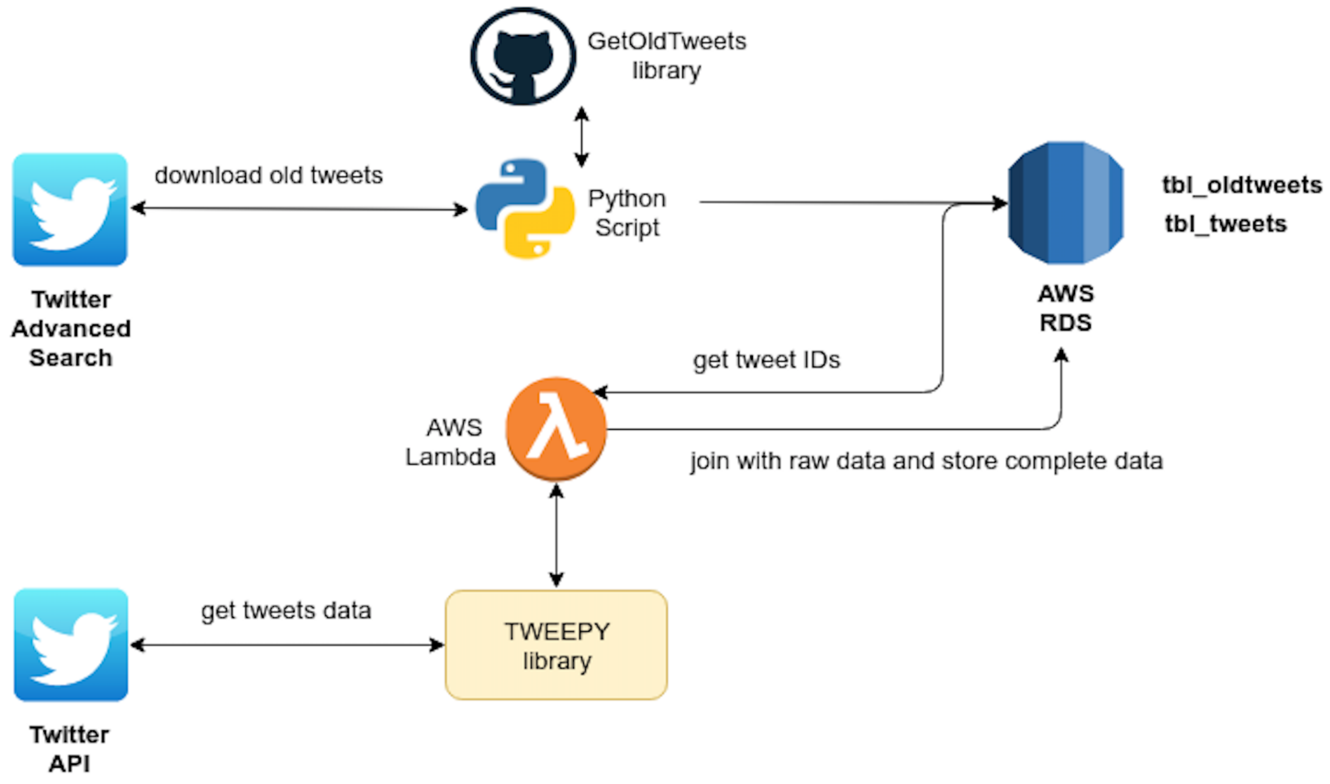


Fig. 3. Data Collection Model

A. Twitter Data Acquisition

A systematic data collection strategy has been developed and put in place to capture, transform and store tweets in order to prepare the dataset for this analysis. Although tweets are in form of semi-structured data, in particular in JSON format, we decided to transform tweets into a schema-based relational database in Cloud that would enable us to store separate versions of historical data with few or less columns, also enables us to perform analysis in a more easier structured fashion. Nevertheless, it was inevitable to join raw data which is scraped from web with data collected from the Twitter API in order to enrich the collection by having more fields for each and every tweet.

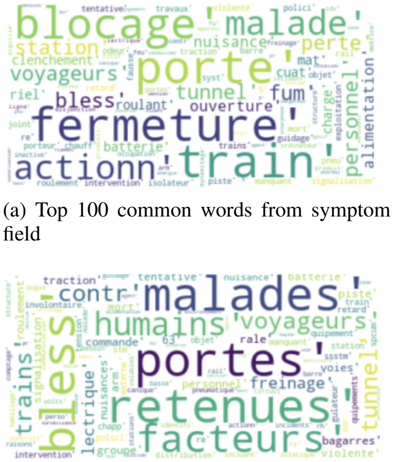
As shown in figure 3, our approach to data collection is planned to be carried out in several phases:

First we have scraped 583,412 historical tweets over four years from 2016 to 2019; from the web, within the geolocation boundaries of Montreal metropolitan area in Quebec, Canada. Twitter Official API has the limitation of time constraints. We cannot get tweets back in the timeline more than one week back. However, Twitter provides advanced online search that can be used to search through tweets in specific areas since March 21st 2006 until today without any limit. Results are provided in browser, therefore, in order to store historical tweets, we have used web scraping methods. A Python script has been written that takes advantage of a Python library called GetOldTweets3 [26] which can be used to download tweets filtered by location.

When scraping data from Twitter advanced search results, there are few fields available for tweets such as ID, Date (posted on Twitter), Truncated Text (Truncated body of the tweet message). To carry our study we needed more information about tweets i.e. complete body of tweet message and complete list of available fields.

To store retrieved tweets, we set up an Amazon Relational Database Service (RDS). One table contains all the historical tweets in raw format (fewer fields), and the other table is used to store all the fields available for tweets that we are going to explain the procedure of collecting them in few latter lines.

Since we have already collected tweet IDs corresponding to our search criteria in previous step, we were able to use Twitter API and retrieve all the available fields with regard to each and every tweet. Therefore, we developed another Python script



(a) Top 100 common words from symptom field

(b) Top 100 common words from code fields

Fig. 4. Common words inspection extracted from operational data sources

based on a Python library[2] that uses Twitter API to obtain information about the collected tweets in previous step using tweet IDs. For this purpose, an AWS[3] Lambda function[4] has been deployed which uses the Tweepy Python library [27] to interact with Twitter API, and passes a list of tweets' IDs and retrieves the data (enriched and extended information) in order to store in our database. The procedure can be summarized as follow:

- Get a list of tweets' IDs from RDS as the reference for tweets
- Call Twitter API and fetch the complete list of fields (columns) available for tweets
- Transform (date conversions, string manipulations, etc.), clean and flatten JSON objects
- Respect the Twitter API limitation by calling it in chunks (450 IDs per hour)
- Store collected data in CSV format, possible to be inserted into the relational database

The Twitter API provides all the available fields for tweets (retweets, responses, location, language, etc.), however, because of its limitations, it is extremely difficult to retrieve large amounts of data such as historical tweets needed for this research. That is the reason why we identified tweets related to the scope of our study using the web scraper then we used Twitter API to collect all the required data for tweets. This approach helped us to overcome the complexity of search back in time by location in Twitter API and avoid the limitations of it.

B. Data Labelling

We have access to the operational data source of STM (Société de transport de Montréal). It can facilitate the process of labelling the data by using the STM operational data to get some insights about French frequent vocabulary used in the fields of the data. These fields are about incident description, specification about different types of incidents, the symptoms of different types of incidents and the causes of different types of incidents. We mainly use these fields of the STM operational data to excerpt subway incident related tweets. Therefore, we developed a script to extract the common words used in the above-mentioned fields of the STM operational data source for variety of data types. We also manually added the common English subway incident terms to the vocabulary list to be used as a primary filter for manual labelling of data.

Figure 4 shows some examples of common words extracted from STM operational data sources.

Next, we manually verified the relevance of the vocabulary words and the distribution of their occurrence in years from 2016 to 2019 in the incident data. Then we filtered the dataset that we already collected from Twitter API and Web scraping based on retrieved common words to extract the tweets containing those words. To help the process of labelling even more, we also identified the official accounts that frequently post tweets about subway incident events and we extracted those tweets and the replies to official tweets about incident as well. By consolidating filtered tweets, official tweets about incidents and replies from users to official tweets, we started the process of manually labelling the data into two classes which are defined as follows:

[2] https://www.tweepy.org

[3] Amazon Web Services

[4] A serverless, event-driven computing service that executes code in response to events, and manages the computing resources provided by that code automatically.

- Incident: This type of tweets report any event that causes any amount of delay or somehow is related to abnormality in subway functionality. The examples of these events include lack of train staff, non-closing door, any nuisances for the driver, violent death attempt, suicide attempt, etc.

- Non-incident: Any tweet not falling into the other category shall be labelled as Non-incident. This class also includes the tweets that include the above-mentioned terminology but do not disclose any events relevant to subway incidents, and tweets that report incidents but are not relevant to subway.

We manually labelled a subset of our tweets in the geolocation of Montreal metropolitan boundaries in above mentioned classes. The process of annotating the data for incident-related records has been done by extracting a subset of tweets based on the frequently occurring words used in STM incident reports and manually sifting through the subset of data to filter out irrelevant records. The prepared training dataset has been then evaluated with STM experts. For the baseline models we utilize a manually labelled dataset consists of 21370 records of only English tweets since they do not exhibit the ability of multilingual and cross-lingual classification. The labelled dataset that we used for models based on BERT contains 21389 labelled tweets consists of 10381 records in English and 11008 records in French. We were able to preserve the balance in terms of the amount of data for each class in both languages.

C. Benchmark Methods

Followings are the algorithms that we implemented as baseline for representation techniques.

Count: Term Frequency (Count) model is the simplest text representation. In this representation, every document is considered as a vector where its components are represented by the terms in the document collection. It regards each word as a separate individual. It represents a document d as $d = (w1, w2, ..., wl)$, where wi represents the ith word appearing in document d, and l represents the number of words in the document d.

TFIDF: Term Frequency-Inverse Document Frequency (TFIDF) [28] is another most popular text representation amongst term weighting methods. Term frequency alone may not have the discriminating power to pick up all relevant documents from other irrelevant documents. Hence, idf (inverse document frequency) factor that takes the collection distribution into account has been added to the basic count factor. The idf factor varies inversely with the number of documents n_i which contain the term t_i in a collection of N documents. It is typically computed as $log(\frac{N}{n_i})$.

Word2Vec: Word2Vec [1] is one of the most popular techniques to learn word embeddings with neural networks. Word2Vec proposes two embedding algorithms, Continuous Bag-of-Words (CBoW) model and Skip-Gram model. CBoW learns word vectors by trying to predict a word given its context. While, Skip-Gram predicts the context words given the center word.

Doc2Vec: Doc2vec [29] is a generalizing of the word2vec for representing documents as a vector. It is based on Word2Vec, with only adding another vector (paragraph ID) to the input. The Doc2Vec based on CBoW is called distributed Memory version of Paragraph Vector (PV-DM). The other Doc2Vec based on Skip-Gram is called Distributed Bag of Words version of Paragraph Vector (PV-DBOW).

We also implemented the following models as the baseline for classification:

NB: Naive Bayes classifier is based on the Bayes probability theorem. It computes the posterior probability of a document belonging to different classes, then assigns the document to the class with the highest posterior probability. In text classification with naıve bayes, word frequency is usually used as the text features. Nevertheless, the frequency-based probability may introduce zeros when multiplying the probabilities, resulting in a failure in preserving the information contributed by the non-zero probabilities. Therefore, a smoothing approach, such as Laplace smoothing, must be adopted to tackle this problem.

LR: Logistic regression (LR) [30] measures the relationship between the dependent variable and one or more independent variables by estimating probabilities using a sigmoid function.

SVM: Support Vector Machines (SVM) [31] learn the boundaries for classification between samples of two or more classes by mapping sample points into a higher dimensional space. The best separation is achieved by the hyperplane that has the largest distance to the nearest training data point (functional margin) of any class. In addition to performing linear classification, by using the kernel trick, SVM can efficiently perform non-linear classification.

RandomForest: Random forests [32] are meta estimators that fit a number of decision tree classifiers on various sub-samples of the dataset and use averaging or voting to improve the predictive accuracy and control over-fitting.

***XGBOOST**:* Extreme Gradient Boosting (XGBOOST) [33] is an implementation of gradient boosted decision trees designed for speed and performance which produces a prediction model in the form of an ensemble of weak decision trees. It builds the model in different stages, and it generalizes them by allowing optimization of an arbitrary differentiable loss function.

Random forests are a large number of trees, combined using averages or ”majority rules” at the end of the process. While, Gradient boosting machines combine decision trees with starting the combining process at the beginning, instead of at the end.

D. Evaluation method

TABLE I
ACCURACY, PRECISION, RECALL AND F1-SCORE RESULTS FOR BASELINE MODELS

	Accuracy	Precision	Recall	F1-Score
NB + Count	0.742	0.741	0.744	0.741
NB + TF-IDF	0.775	0.771	0.772	0.772
NB + Word2Vec	0.663	0.726	0.692	0.657
NB + Doc2Vec	0.699	0.705	0.683	0.683
LR + Count	0.805	0.807	0.797	0.800
LR + TF-IDF	0.802	0.801	0.796	0.798
LR + Word2Vec	0.775	0.771	0.773	0.772
LR + Doc2Vec	0.801	0.810	0.782	0.788
SVM + Count	**0.828**	0.824	**0.824**	**0.824**
SVM + TF-IDF	**0.835**	**0.835**	**0.825**	**0.829**
SVM + Word2Vec	0.800	0.798	0.803	0.799
SVM + Doc2Vec	0.794	0.797	0.780	0.784
RandomForests + Count	0.822	0.822	0.814	0.817
RandomForests + TF-IDF	0.824	**0.829**	0.811	0.816
RandomForests + Word2Vec	0.800	0.801	0.790	0.793
RandomForests + Doc2Vec	0.788	0.789	0.788	0.783
XGBoost + Count	0.810	0.809	0.806	0.807
XGBoost + TF-IDF	0.813	0.814	0.807	0.809
XGBoost + Word2Vec	0.813	0.810	0.810	0.810
XGBoost + Doc2Vec	0.783	0.783	0.779	0.780

Since the dataset that we we prepared for this study is well balanced in terms of almost same amount of data for each class, we use accuracy as the main evaluation metric in order to evaluate the performance of the classifiers which is calculated from below equation:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

Where, TP represents the True Positive cases, FP represents the False Positive cases, TN represents True Negative cases and FN represents False Negative cases.

Moreover, the macro Precision, macro Recall, macro F1-Score, are considered, measured and reported when evaluating each model which are calculated as below:

$$Precision_M = \frac{\sum_{i=1}^{l} \frac{TP_i}{TP_i + FP_i}}{l} \tag{2}$$

$$Recall_M = \frac{\sum_{i=1}^{l} \frac{TP_i}{TP_i + FN_i}}{l} \tag{3}$$

TABLE II
ACCURACY, PRECISION, RECALL AND F1-SCORE RESULTS FOR BERT MODELS

	Accuracy	Precision	Recall	F1-Score
	BERT Multilingual Classification			
Testing on English	**0.891**	**0.889**	**0.887**	**0.888**
Testing on French	0.870	0.870	0.871	0.870
	BERT Zero-Shot Cross-Lingual / Training on English			
Testing on English	0.866	0.864	0.860	0.862
Testing on French	0.837	0.842	0.834	0.835
	BERT Zero-Shot Cross-Lingual / Training on French			
Testing on English	0.858	0.866	0.844	0.850
Testing on French	0.864	0.867	0.866	0.864

$$Fscore_M = \frac{(\beta^2 + 1) Precision_M Recall_M}{\beta^2 Precision_M + Recall_M} \tag{4}$$

Where, l is the number of classes and β is a parameter that can be used to tune the relative importance of precision and recall.

E. Experimental details

To be able to deploy all our implemented models and to conduct our experiments at scale, we used Amazon SageMaker[5] which provides the ability to use high computational power such as GPU clusters.

In the data pre-processing phase we took following steps:

- Removing extra spaces
- Striping ending and leading spaces
- Removing special characters such as ”&”, ”<”, ”>” and ”@” symbols
- Removing the URL links
- Removing non-ascii words
- Converting all words to lower-case

In addition, only for baseline models, we also remove *stop words* from tweets in both languages.

For all baseline algorithms we used 80% of the data for training and 20% for testing. We also performed 10-fold cross validation for all baseline models to help us in gauging the effectiveness of the model's performance.

As for the Word2Vec model, the embedding of all tweet words are averaged, such that each tweet is represented by a 300-dimensional vector.

For the Bidirectional Encoders Representations from Transformers models, we conduct experiments with Zero-Shot in cross-lingual and monolingual transfer learning in which we fine-tune BERT and train a classifier on one language and test on another language as well as training on each language separately and test on each language separately as well. We also performed multilingual learning in which we shuffle the training data for both languages to evaluate the performance of the model by fine-tuning it on both languages in the data set together. In order to do so, we use a model of type *BERT For Sequence Classification*. This consists of BERT's pre-trained core with 12 transformer layers, 12 self-attention heads, and 768 hidden dimensions and a one-layer classifier on top that maps the output for the token to the required number of classes. We use BertAdam as the optimizer and Cross Entropy loss as the loss function.

[5] https://aws.amazon.com/sagemaker/

In all experiments with Bidirectional Encoders Representations from Transformers, we used 80% of the data for training and we kept 10% of the data as the development data and 10% as the test data. At each epoch; we train the model on the training data and evaluate it on the development data while keeping a history of the loss function which is Binary Cross Entropy. We stop the training process when the loss on the development set does not improve for a certain number of steps.

F. Results

In order to understand the performance of our trained models, we compare their results with our chosen baseline models that we implemented and trained on our labelled data. Table I shows the Accuracy, macro Precision, macro Recall and macro F1-Score for baseline models. Our results indicate that the accuracy of baseline models ranges between 0.663 to 0.835 where the worst outcome belongs to Naive Bayes classifier with Word2Vec tweet representation and the best one belongs to Support Vector Machines classifier with TF-IDF tweet representation followed by Support Vector Machines classifier with Count vectorizer. In terms of macro precision of classification procedure, Support Vector Machines classifier with with TF-IDF tweet representation obtains again the best score, followed by Random Forests classifier with TF-IDF vectorizer and Support Vector Machines classifier with Count vectorizer, respectively. For the macro recall of classification procedure, the results represent Naive Bayes classifier with Doc2Vec tweet representation as the algorithm with 0.683 score as the worst macro recall, whilst Support Vector Machines classifier with TF-IDF tweet representation outperforms the rest, followed by Support Vector Machines classifier with Count vectorizer. In the results of the table I we can also see that the Support Vector Machines classification with TF-IDF vectorizer and Support Vector Machines classification with Count vectorizer are the most effective methods in terms of F1-score.

Table II shows the results of our experiments with Bidirectional Encoders Representations from Transformers in multilingual training and also zero-shot cross-lingual and monolingual with training on English and French separately. Our experimental results in zero-shot training fashion depict that training BERT in one language provides the accuracy around 0.837 and 0.866 for training on English tweets and testing on English tweets and French tweets respectively and the accuracy around 0.858 and 0.864 for training on French tweets and testing on English tweets and French tweets respectively. Such observations reveals that all the outcomes of BERT in zero-shot experiments outperforms the results of all of the baseline models. The results of our experiments also depict the accuracy of 0.891 for tweets in English language in our dataset and the accuracy of 0.870 for tweets in French language in our dataset for multilingual classification which both are even higher than the accuracy of all zero-shot classification experiments in both cross-lingual and monolingual settings.

V. Conclusion

In this research, we introduced a structured approach to collect, transform and use tweets posted by users on Twitter to identify metro incidents. First, we conducted our experiments on widely used natural language processing classification methods combined with multiple text representation algorithms. Then we provided comprehensive study comparing the performance of those methods with Bidirectional Encoder Representations from Transformers (BERT) technique, which allows understanding not previously tested ability of the BERT algorithm capability to generalize cross-lingual and multilingual representation for subway incident related tweet classification. Such study became necessary in cases when tweet dataset contains more than one language, hence making it impossible for traditional techniques to overcome the classification problem for those languages at once. We have implemented a method that leverages Bidirectional Encoders Representations from Transformers with a single layer fully connected neural network for sequence classification and we carried out several experiments with zero-shot cross-lingual, monolingual and multilingual training procedure. We have shown that this model handles transfer across both languages in our dataset with satisfactory results when training on both English and French languages together. Our experiments show that fine-tuning the model in both languages in the dataset together during the training process would improve the model's efficiency and obtains even higher accuracy, concluding that training BERT, in the multilingual way, can yield best results obtained from all the models implemented in this analysis. It also delivers better results than baseline models when training on one language, tested on the other unseen language during training process as well as in monolingual setups. Our method demonstrates that it is possible to train a more accurate text classifier that is capable of performing subway transportation incident detection using tweets in different languages.

References

[1] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[3] Z. Kokkinogenis, J. Filguieras, S. Carvalho, L. Sarmento, and R. J. Rossetti, "Mobility network evaluation in the user perspective: Real-time sensing of traffic information in twitter messages," in *Advances in Artificial Transportation Systems and Simulation*. Elsevier, 2015, pp. 219–234.

[4] S. F. L. d. Carvalho *et al.*, "Real-time sensing of traffic information in twitter messages," 2010.

[5] T. H. Rashidi, A. Abbasi, M. Maghrebi, S. Hasan, and T. S. Waller, "Exploring the capacity of social media data for modelling travel behaviour: Opportunities and challenges," *Transportation Research Part C: Emerging Technologies*, vol. 75, pp. 197–211, 2017.

[6] M. Parsafard, G. Chi, X. Qu, X. Li, and H. Wang, "Error measures for trajectory estimations with geo-tagged mobility sample data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 7, pp. 2566–2583, 2018.

[7] S. Wang, X. Zhang, F. Li, S. Y. Philip, and Z. Huang, "Efficient traffic estimation with multi-sourced data by parallel coupled hidden markov model," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 3010–3023, 2018.

[8] R. D. Das and R. S. Purves, "Exploring the potential of twitter to understand traffic events and their locations in greater mumbai, india," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[9] N. Wanichayapong, W. Pruthipunyaskul, W. Pattara-Atikom, and P. Chaovalit, "Social-based traffic information extraction and classification," in *2011 11th International Conference on ITS Telecommunications*. IEEE, 2011, pp. 107–112.

[10] S. S. Ribeiro Jr, C. A. Davis Jr, D. R. R. Oliveira, W. Meira Jr, T. S. Gonçalves, and G. L. Pappa, "Traffic observatory: a system to detect and locate traffic events and conditions using twitter," in *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Location-Based Social Networks*. ACM, 2012, pp. 5–11.

[11] K. Fu, R. Nune, and J. X. Tao, "Social media data analysis for traffic incident detection and management," Tech. Rep., 2015.

[12] F. Rebelo, C. Soares, and R. J. Rossetti, "Twitterjam: Identification of mobility patterns in urban centers based on tweets," in *2015 IEEE First International Smart Cities Conference (ISC2)*. IEEE, 2015, pp. 1–6.

[13] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," in *Proceedings of the fifth annual workshop on Computational learning theory*, 1992, pp. 144–152.

[14] E. D'Andrea, P. Ducange, B. Lazzerini, and F. Marcelloni, "Real-time detection of traffic from twitter stream analysis," *IEEE transactions on intelligent transportation systems*, vol. 16, no. 4, pp. 2269–2283, 2015.

[15] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *European conference on machine learning*. Springer, 1998, pp. 137–142.

[16] J. T. Kent, "Information gain and a general measure of correlation," *Biometrika*, vol. 70, no. 1, pp. 163–173, 1983.

[17] Y. Gu, Z. S. Qian, and F. Chen, "From twitter to detector: Real-time traffic incident detection using social media data," *Transportation research part C: emerging technologies*, vol. 67, pp. 321–342, 2016.

[18] T. Kuflik, E. Minkov, S. Nocera, S. Grant-Muller, A. Gal-Tzur, and I. Shoor, "Automating a framework to extract and analyse transport related social media content: The potential and the challenges," *Transportation Research Part C: Emerging Technologies*, vol. 77, pp. 275–291, 2017.

[19] J. Pereira, A. Pasquali, P. Saleiro, and R. Rossetti, "Transportation in social media: an automatic classifier for travel-related tweets," in *EPIA Conference on Artificial Intelligence*. Springer, 2017, pp. 355–366.

[20] Z. Zhang, Q. He, J. Gao, and M. Ni, "A deep learning approach for detecting traffic accidents from social media data," *Transportation research part C: emerging technologies*, vol. 86, pp. 580–596, 2018.

[21] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[22] A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[23] Y. Chen, Y. Lv, X. Wang, L. Li, and F.-Y. Wang, "Detecting traffic information from social media texts with deep learning approaches," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 3049–3058, 2018.

[24] S. Dabiri and K. Heaslip, "Developing a twitter-based traffic event detection model using deep learning architectures," *Expert Systems with Applications*, vol. 118, pp. 425–439, 2019.

[25] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[26] August 2020. [Online]. Available: https://pypi.org/project/GetOldTweets3/

[27] August 2020. [Online]. Available: https://www.tweepy.org/

[28] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information processing & management*, vol. 24, no. 5, pp. 513–523, 1988.

[29] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *International conference on machine learning*, 2014, pp. 1188–1196.

[30] R. E. Wright, "Logistic regression." 1995.

[31] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[32] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[33] 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*, 2016, pp. 785–794.