Technical Report

IDMC: SOCIAL MEDIA MONITORING

Gokberk Ozsoy, Katharina Boersig, Michaela Wenner, Tabea Donauer

November 27, 2020

Hack4Good, Fall 2020 Edition

1. Introduction

In the last decade, increased availability of computational power has allowed for fast advances in data science. Research as well as industry highly benefits from gained analytic capabilities and resulting knowledge. However, a large data volume requires automated techniques to process the data and interpret its results. Especially for non-profit organisations (NPO), financial and time resources often hinder the application of data science techniques. As a result, the ETH Data Analytics Club started a pro-bono program, Hack4Good (H4G), targeting to tackle data science problems of NPOs. During 8 weeks, groups of ETH Students collaborate closely with NPOs to find data-driven solutions to make an impact on the NPOs cause. In this years H4G fall 2020 Edition, our team collaborated with the Internal Displacement Monitoring Center (IDMC). IDMC globally provides data on internal displacement due to natural disasters and conflict to inform policy makers and reduce risk of future displacement. For this task, IDMC largely relies on information from official reports and news paper articles. In recent years, social media platforms have gained importance in our society, not only to entertain but also to inform. This is especially the case for Twitter. Many individuals as well as organisations and governmental institutions make use of Twitter to efficiently and timely inform followers of happenings around the world. The goal of this project is, to make use of this information transfer and explore Twitter as a data source for Internal Displacement Monitoring. During the course of this project, we developed a pipeline to download, process and classify tweets as relevant or irrelevant for IDMC. Additionally, we use an entity recognition algorithm to extract important information from tweets, such as trigger and displacement term. Here, we describe the file structure, pipeline architecture, implemented algorithms provided to IDMC in the form of a GitLab repository (H4G-IDMC-GitLab). Furthermore we discuss our results and assumptions made for this project.

2. Data and Assumptions

IDMC provided 631 tweets in english language labelled as relevant or irrelevant as well as lists of displacement terms, units and triggers. The tweets were manually labelled by IDMC employees. Decision rules were based on the presence of displacements terms and triggers as well as on the expert knowledge, e.g. based on background information about certain disaster events. Out of these 631 tweets 236 are labelled as relevant and 397 as irrelevant.
The labelled tweets have been split into training and test data sets to train a machine learning classifier (on the training data) and show how it performs (on the test data). A few assumptions that shaped our solution have been taken in advance:

1. Acquisition of tweets via the Twitter API is based on a hard-coded rule. Only tweets containing a combination of displacement term and trigger are considered for the machine learning classifier.

2. Given the limited number of labelled tweets, we exclude the use of deep-learning models. Such models require a huge amount of training data and we therefore did not consider them feasible.

3. Methods

This Section explains the single methods and algorithms which have been implemented. An overview of the pipeline and how the steps are connected is given in Section 4.

3.1. Tweet Acquisition

In a first step, we acquire tweets using the Tweepy python library which allows easy access to the Twitter API (Roesslein, 2009). To gain access authorisation to the API, a Twitter Developer Account needs to be set up. The Twitter API allows to access tweets (including text body and some meta-information) from the last two weeks. From these tweets, we extract the ones containing a combination of displacement trigger and a displacement term. For a list of triggers and terms used see Appendix 2.

3.2. Preprocessing

Preprocessing aims at converting the natural language in the tweets to a more comprehensible format for machine learning (ML) algorithms to understand. Natural language preprocessing typically includes tasks like tokenization, stopwords removal, stemming and lemmatization, briefly described below. To perform these tasks, we use the 'Natural Language Toolkit (NLTK)', which is an open source python library (Bird et al., 2009).

- **Tokenization:** Tokenization separates the tweet into words, referred to as tokens.
- **Tweet cleaning:** Repeated letters, urls and user mentions often occur in tweets and have to be removed during preprocessing.
- **Stopwords removal:** Stopwords are words with little meaning in a text, such as 'and' or 'so'. They are filtered out of the tweet body because they are considered useless for the classification algorithm.
- **Stemming:** Stemming reduces a word to its root stem by removing its suffixes *(e.g., studies $\rightarrow$ studi).* It is applied to the list of keywords, triggers, terms and units.
- **Lemmatization:** Lemmatization is used to reduce a word to its 'lemma'. In contrast to stemming, the output of lemmatization is a valid word *(e.g., studies $\rightarrow$ study)*

3.3. Word Embedding

The goal of this step is to convert the words contained in the preprocessed tweets to numerical vectors, which can be handled by the subsequently applied classification algorithms. Words which have a similar meaning should have a similar mathematical representation. For the word embedding, we use an unsupervised algorithm called word2vec. Word2vec assumes that words which have a similar meaning are used in a similar context and requires vast amount of training data to find the words which are used in a similar context. For this project, we apply a word2vec embedding, which has been trained on 400 million tweets (Godin, 2019).

3.4. Classification Algorithms

The labelled preprocessed and as numerical vectors represented tweets are then used to train a binary classifier. The goal is to predict whether a tweet is relevant for IDMC or not. ML classifiers are trained on features of the data set. In this case, the features of a tweet are the numerical vectors of its words. To obtain the best classification results and analyse the robustness, we test three different classifiers: linear, Support Vector Machine (SVM), and Random Forest (RF). A linear classifier makes a decision based on a linear combination of all features. Hence, in an N-dimensional space, linear classifiers look for a hyperplane that best separates the data. One type of such a linear classifiers are SVMs. SVMs is thereby looking for an hyperplane with maximum distance to the closest samples on either side. Contrary to SVMs, RF is a non-linear classifier. RF is based on an aggregation of weak decision trees. Each tree is trained on a subset of training data and features. After each tree predicts a class, a majority vote is performed. The most often occurring class is then the final prediction.
To assess the performance of the ML algorithms we us two types of metrics: the confusion matrix and a balanced score as overall score, to counteract misleading results due to the imbalanced number of relevant and irrelevant tweets in the test data. A confusion matrix consists of the true label of the samples of each class as rows and the classifier predicted label as columns. For a perfect classifier all samples are located on the diagonal of the matrix. Using the confusion matrix, the classifier can be evaluated for each class separately.

3.5. Named Entity Recognition

To extract information from as-relevant-classified tweets, we use an algorithm called Named Entity Recognition (NER). A named entity can be for example: french language, or IDMC organisation. To recognise such entities we use the SpaCy library for python (Honnibal and Montani, 2017).

SpaCy provides core models in 16 languages, trained on up to 1.2 million vectors. It is able to extract a long list of entities, such as companies, geopolitical locations and nationalities. Additionally, SpaCy allows to train new models on custom entities. For this project, we make use of this feature by training a model on 170 labelled tweets containing following entity labels: term, unit, time, trigger (trig) and figure (fig). The training data for a SpaCy model needs to be brought into a specific format, a list of tuples containing i) the raw tweet and ii) a dictionary of entities containing the entity name and the position of the word (start and end) within the tweet. The training data is automatically formatted in a correct way, if the tweets, important words in the tweets and associated tags are given.

3.6. Extension

When using Twitter as an information source for global monitoring of displaced people, spatial coverage of the tweets should be taken into account. Two limitations of the platform, which can have a major impact on the number and type of displacement tweets monitored with the developed tool, have been identified:

1. **Spatial bias:** Twitter is most popular in western, industrialised countries. Much fewer tweets are posted in the developing world and in countries with restrictions in place such as China. The spatial limitation of Twitter is an inherent feature of the social media platform which could not be addressed in the scope of the project.
2. **Language bias:** Twitter users around the world post in different languages and extracting tweets in English severely restricts the area which can be monitored. We therefore implemented the necessary steps to use the tool for other languages. For tweet extraction in another language than English, a translated list of displacement terms and triggers has to be provided. Afterwards, the tweets are translated via GoogleTranslate and automatically the identical preprocessing steps and classification algorithm are applied.

4. Pipeline

To automate the above-described workflow, we utilise a machine learning pipeline containing several python scripts to input and transform data, train a ML model and output results. A simplified visualisation of the developed machine learning pipeline is shown in Fig. 1. Python scripts used in the pipeline are stored in the *src* directory of our GitLab repository.The data and results produced by the pipeline are stored in the *data*,*config* and *results* directories. In the following, we list corresponding directories and scripts with a short description of their task.

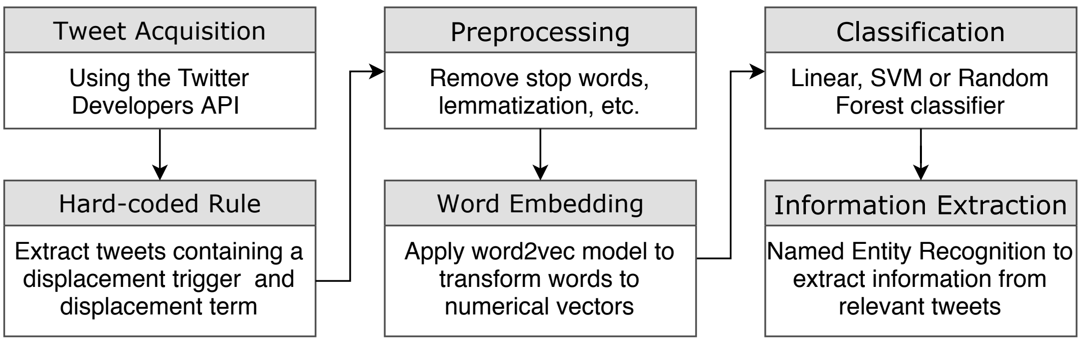
Tweet Acquisition
Using the Twitter Developers API
Hard-coded Rule
Extract tweets containing a displacement trigger and displacement term
Preprocessing
Remove stop words, lemmatization, etc.
Word Embedding
Apply word2vec model to transform words to numerical vectors
Classification
Linear, SVM or Random Forest classifier
Information Extraction
Named Entity Recognition to extract information from relevant tweets

Figure 1: Simplified visualization of the pipeline developed for acquisition, processing, classification and information extraction of displacement related tweets.

Python Scripts

data_acquisition

- **TweetExtractor.py** extracts relevant tweets from twitter.

data_preprocessing

- **Translator.py** translates extracted tweets to english (if needed).
- **TweetPreprocessor.py** applies preprocessing to extraced tweets.

data_classifcation

- **TweetClassifier.py** trains and tests binary tweet classifier on labelled tweets, predicts labels for unlabelled tweets .
- **InformationExtractor.py** extracts relevant information from tweets including metadata and relevant words using NERs.
- **CustomNER.py** trains a customized NER for detecting internal displacement words in tweets.

Data Files

models

- contains trained models for later usage.

preprocessed_data

- **labelled_tweets** contains preprocessed labelled tweets in different languages.
- **predict_tweets** contains preprocessed unlabelled tweets in different languages and different extraction dates.

raw_data

- **labelled_tweets** contains directories with csv files of labelled tweets in different languages.
- **raw_tweets** contains directories with csv files of unlabelled tweets in different languages and extraction dates.

utils_data

- **idmc_terms_units.csv** csv files with provided displacement terms and units.
- **idmc_triggers.csv** csv files with provided displacements triggers.

Config Files

- **data_acqusition** contains config files for tweet extraction.
- **data_classification** contains config file for setting model parameters.
- **keywords.json** contains all keywords (can be created/updated automatically from csv files in utils_data).
- **structure.json** contains all needed path definitions.

Result Files

- **analysis** contains results from initial data analysis.
- **modelResults** contains confusion matrices from classifier testing.
- **predictions** contains predictions of predicted and tested tweets.

5. Results

Different ML classifiers, including Linear, SVM and Random Forest have been trained on the training data set of handlabelled tweets. By a preliminary hard rule, tweets not containing any trigger word are excluded leaving 306 labelled tweets for training the classifier. In order to validate the classifier, it is trained on 80% of the training data and tested on the remaining 20% of the data. Testing the classifiers against test data (20% of labelled tweets containing trigger words), we find that SVM performs best, with a classification accuracy of approximately 80 %. The classification accuracy describes that 80 % of the tweets in the validation dataset were classified correctly, which is also displayed in the confusion matrix, see Table 1.
Due to the small data set of manually labelled tweets cross-validation (repeating the training and validation steps multiple times) is also an important step and has confirmed the use of SVM classifier. Other classifiers may have performed better for one single test and very poorly in another test (meaning that they predict the labels for tweets in the validation data sets very accurately once and then fail to do so when repeated). The reason for this is that some classifiers are more prone to overfitting, which means they adjust very well to one particular data set, but fail to predict the labels for a different data set.

Table 1: Predicted labels from the SVM classifier compared with the labels from the validation data set. The overall accuracy calculated for the SVM classifier is approximately 80 %.

	SVM Classifier	
	Predicted relevant	Predicted irrelevant
Relevant tweet	19	8
Irrelevant tweet	3	32

6. Discussion and Conclusion

During this project, we developed a pipeline for tweet acquisition, classification into relevant and irrelevant tweets by machine learning algorithms and extraction of related informations (such as displacement figure and location). The SVM classifier successfully predicts for approximately 80% of the tweets in the validation dataset whether they are relevant or not. Moreover, we have been able to extend the pipeline to handle tweets in other languages than English. Despite the promising results, there are also some limitations to this project, including:

- Due to the limited availability of training and validation data it is not clear yet, whether the classifier will perform as well on any set of displacement related tweets.
- The fact that identical tweets, which have been retweeted or posted multiple times, will be extracted multiple times, has not been handled in the scope of this project.
- Moreover, tweets which are classified as relevant and from which information is extracted, should be examined by an expert, as we do not check the quality of the source (or the reliability of the twitter user respectively).

Improving the performance of the machine learning classifier can be achieved by providing a bigger training dataset, which requires the input of IDMC experts.

References

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python.* O'Reilly Media, Inc., 1st edition.

Godin, F. (2019). *Improving and Interpreting Neural Networks for Word-Level Prediction Tasks in Natural Language Processing.* PhD thesis, Ghent University, Belgium.

Honnibal, M. and Montani, I. (2017). spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Roesslein, J. (2009). tweepy documentation. *Online] http://tweepy. readthedocs. io/en/v3*, 5.

A. List of Displacement Triggers, Terms and Units

Table 2: List of displacement trigger, terms and units provided by IDMC. Presence of triggers and terms used as filter for tweets scraped from the Twitter API.

Trigger	conflict, attack, battle, clashes, cyclone, destroyed housing, earthquake, flood, hurricane, landslide, storm, tsunami, typhoon, wildfire, fire, violence, fight
Terms	destroyed, homeless, sheltered, displaced, evacuate, evacuated, arrived, collapsed, crossed, damaged, entered, evicted, eviction, flee, forced, relief camp, relocated, IDP, evacuees
Units	families, households, people, person, asylum seeker, building, citizens, dwelling,home, house, hut, individuals, inhabitants, migrants, refugee, residents, villagers, civilians