Title page

Thesis title

Yaser Kaddoura

Note: When reviewer/examiner decides that the report is close to be finished, contact coordinator for a report number and instructions to produce a title and abstract page.

This page intentionally left blank.

Abstract

This page intentionally left blank.

Contents

List of Figures

List of Tables

List of Acronyms

(Might be needed to make introduction starts at odd page number)

This page intentionally left blank.

Chapter 1

Introduction

Earlier in this century, floods around Lake Vänern and Arvika have costed Sweden an estimate of 11.1 billions Swedish Krona for damages and repairs [36]. Counties of Dalarna and Gävleborg suffered from flash floods in 2021 disturbing the daily life of their citizens and damaging public and private properties [5]. Flooding is a devastating natural disaster that threatens the lively hood of people and the infrastructure of communities around the world [13].

To facilitate the process of emergency management during these hazardous events, early warning systems analyse their risk, monitor and warn the public while ensuring their readiness [4]. Traditionally, meteorologists forecast the weather by relying on tools such as gauges, satellites, and radars for data extraction. The emergence of social media platforms such as Twitter provide individuals with a public space to share their experience, effectively creating another potential source of data.

Researchers started harnessing this new wealth of information to aid the disaster management procedure. Twitter streaming Application Programming Interfaces (API) makes it possible to create a monitoring system for early event detection on a global [7] and local [3] scales. Another use for it would be identifying victims in real-time, locating their physical location, and communicating the information to rescue teams [40]. After the threat subsides, emergency managers can use relevant tweets to assess the impact and plan the recovery phase [3]. To prepare for future floods, authoritative entities can make informed actions by analysing historical data and determining the locations suffering from recurrent calamities. This newly acquired knowledge can augment weather warning systems' pipelines improving their accuracies such as Swedish Meteorological and Hydrological Institute (SMHI), a Swedish expert authority with a global perspective and a vital task in predicting changes in weather, water, and climate [41].

This thesis project uses Swedish tweets to extract relevant knowledge about flood events in Sweden with the focus on answering the following questions:

1. What methods can be used to classify Swedish tweets related to flood events?
2. How to extract the locations mentioned in the tweets?
3. What visualizations can be used to represent the results?

4. What insights can be extracted by using text analysis on the tweets?

This thesis project is a part of a research project, AI for Climate Adaptation [27] at Linköping University (LiU) in collaboration with the SMHI. It implements a pipeline that provides a visual representation of tweets related to flood events in Sweden. First, relevant tweets are pulled, processed, and classified from the Twitter API using data mining techniques. Second, physical locations are extracted from tweets mentioning flood events employing Named-entity recognition (NER) and gazetteers. Finally, the results are presented on a spatio-temporal visualization, and text analysis techniques are done on the tweets. For verification purposes, the pipeline is applied on a week worth of tweets after past flood events.

Chapter 2

Literature Review

The massive and accessible volume of data that social media produces has attracted the attention of many researchers as a valuable data source for their research topic; however, collecting and processing data of this nature pose many challenges to extracting useful information. This section mentions what other researchers focusing on disaster management topics did to address these challenges while using Twitter; it also discusses the different approaches used for identifying relevant tweets, extracting geographical location from them, making text analysis on the text, and visualizing the results.

2.1 Data Collection

Twitter's API enables developers to retrieve historical tweets using queries that are made of operators to match a variety of tweet attributes, such as a specific keyword, having a geotag provided by the user who created the tweet, and the language classified by Twitter. Users generate around 500-700 million tweets a day [22], making it necessary to limit the number of tweets to fetch using the API to reduce computational power and downtime. Feng and Sester [12] only fetches geotagged tweets and then filters necessary them using 45 keywords in 7 languages; this approach filters out a big chunk of relevant tweets since 1% of tweets are geotagged [24]. A better approach is to fetch tweets using keywords related to the topic of interest in different languages. de Bruijn et al. [7] uses over 40 keywords associated with floods in 11 major languages in the query to fetch tweets.

In addition to using textual data, some researchers use other types of data to enhance their pipelines. Some tweets contain media attachments, such as images and videos that are potential visual information for the concerned research topic [1][37][28]; search engines are another resourceful source for images as well [12]. When it comes to flooding events, hydrological information can be a valuable source of information that can be extracted from a global precipitation dataset based on tweets' time stamps and location in the text [8]. Barker and Macleod [3] use Environment Agency flood-monitoring API[1] to get river gauge levels and flood warnings to identify at-risk flooding areas.

[1] https://environment.data.gov.uk/flood-monitoring/doc/reference#flood-warnings

Processing text is a crucial part of any Natural Language Processing (NLP) pipeline to train an effective classifier. Research requires that the corpus is in multiple languages, so translating the text to one language (most likely English) is needed if the classifier can not handle multilingual data [40]. One of the most common text-processing tasks is removing unnecessary terms such as stopwords, Uniform Resource Locator URLs, numbers, and punctuation marks. User mentions in tweets don't provide useful information, so pipelines often remove or replace them with a generic term such as "@user"[8]. The location of the flood event is an important piece of information that is extracted from a term in the tweet, making it a potential target that includes biases in the dataset by overusing it; de Bruijn et al. [8] replaces these terms by the country name that the location is located in; on the other hand, Petersen and Styve [33] replace the terms by the word "place" if they get mentioned more than 0.5% of the size of the data set. Another way to improve the performance of the classifier is to group the terms by converting them to lower-case and transforming them to their lexeme or word stem by lemmatisation or stemming respectively.

Some tweets are noisy or redundant, making them a target for filtering out. Retweets are identical to other tweets without additional context making them unneeded. Spam bots generate similar tweets for malicious reasons, such as spreading false content to manipulate the public; other reasons could be for utility reasons, such as creating a feed for users to check updates. These tweets introduce noise to the dataset that gets reduced by removing duplicate tweets. de Bruijn et al. [7] only considers one tweet from each user in the last 14 days mentioning a specific region; they also remove tweets containing more than five consecutive words that match with those in another tweet among the previous 100 talking about a location. Singh et al. [40] approaches this problem by only extracting tweets created from mobile phones and only considers tweets from users who have followers/following < 1.

2.2 Text Classification

Identifying disaster events using social media requires a classifier to determine the relevant data. Textual data containing terms related to a disaster doesn't mean that it discusses a disastrous event since words such as "flood" can be used figuratively in sentences (e.g., a flood of joy). A binary classifier labelling the data with "on-topic" and "off-topic" labels is needed to filter out irrelevant content.

Most classifiers use supervised Machine learning (ML) algorithms requiring labelled data for training. A straightforward approach is to manually label a sample of the tweets [7][3]. Petersen and Styve [33] use Crisilext6 [29], a crowdsource labelled tweets, for training their classifiers that get evaluated on 88 million unlabelled tweets containing flood-related terms [6]. Feng and Sester [12] automatically label the tweets by checking if there is rainfall during the provided time and city location by using a weather API[2]; if there's a rainfall, the tweet is labelled positive, negative otherwise.

[2]https://www.wunderground.com/weather/api/d/docs

A classifier needs a numerical representation of the textual data for training. Text is often represented in a real-valued vector by encoding words and their context. There are different word embedding techniques, such as Term Frequency–Inverse Document Frequency (TF-IDF) [47] that reflect how important a word is to a document in a corpus. Word2vec [26] and its extension doc2Vec [20] are other word embedding techniques that capture the semantic and syntactic qualities of words via a vector space with several hundred dimensions, where each unique word in the corpus gets assigned to a vector in the space.

There are different groups of ML algorithms to classify data for varying data types. Supervised algorithms are employed if the training data set is labelled; otherwise, a probabilistic approach can be used by training a naive Bayes classifier on labelled and unlabelled data [21]. Feng and Sester [12] use naive Bayes, random forest, logistic regression, Support Machine Vector (SVM) (RBF Kernel), and SVM (Linear Kernel) on labelled data transformed using TF-IDF with accuracies of 0.7109, 0.7582, 0.7705, 0.7712, and 0.7739, respectively. Petersen and Styve [33] results are more promising, where they train a logistic regression and random forest classifiers with 0.939 and 0.9253 accuracies, respectively. Deep learning approaches generally outperform classical algorithms; one example is Convolutional Neural Networks (CNN) trained on word embeddings for sentence classification. Feng and Sester [12] and Petersen and Styve [33] train a CNN model on word2vec embeddings with 0.7868 and 0.94611 accuracies, respectively.

Recently, transfer learning has been gaining popularity; it's the idea of transferring knowledge acquired by solving one problem to other related problems. In the case of text classification, RNN is a class of artificial neural networks that process sequences of data using Long Short-Term Memory (LSTM) making it a suitable algorithm for NLP tasks; Petersen and Styve [33] fine-tunes the pre-trained Universal Language Model Fine-tuning (ULMFit) [17] with an accuracy of 0.9499. Models using transforms architecture outperform RNN in many NLP tasks. One popular example is Bidirectional Encoder Representations from Transformers (BERT) [10], a deep learning-based NLP pre-training technique used by generalized models by training on a massive dataset; afterwards, they are fine-tuned on a smaller one for a specific task. Alam et al. [1] uses a pre-trained BERT model that works on one language with an accuracy of 0.853, and de Bruijn et al. [7] uses a multilingual model with 0.8 F1-score.

Visual data such as images are usually classified using CNN models by getting the results of the models after removing the output layer to get a feature vector to train classifiers. These models are pre-trained on massive datasets such as Imagenet [19] and places database [48]. Feng and Sester [12] use GoogLeNet (Inception-V3 model) [42] pre-trained on ImageNet to train multilayer perceptron, random Forest, gradient boosted trees, and xgboost with accuracies of 0.8907, 0.9133, 0.9252, and 0.9295, respectively. Ning et al. [28] uses VGGNet [39], Inception V3, ResNet [15], and DenseNet201 [18] with 0.91 accuracy.

2.3 Location Extraction

Identifying the locations of disasters is helpful for the disaster management cycle. Social media enables people to generate Volunteered Geographic Information (VGI), which is more advantageous over the more expensive accuracy testing by official agencies because contributors have unique local knowledge. Detecting a disastrous event and its location as soon as possible can reduce its impact on society [7] by informing the citizens and the authority to prepare for it. During the event, the rescue teams' task would be easier if they can locate the endangered people [40]. After the event wanes, assessing the most impacted spots can enable the authority to make informed decisions on a recovery plan.

Users can assign an accessible property to their tweets, called "geotag" , a geographical identification metadata. Adding "has:geo" to the query sent to the API will return geotagged tweets with metadata about the location, such as a display name, geo polygon, and a geo lat-log coordinate. The geotag is the most straightforward method to identify the locations [12], but unfortunately, only 1% of the tweets are geotagged [25], making it not include a massive amount of tweets mentioning locations.

Locations can be extracted using toponyms, a place's name, in tweets' text by using geoparsing. Geoparsing is a process of converting free-text descriptions of places (such as "twenty miles northeast of Jalalabad") into unambiguous geographic identifiers. A toponym can have more than one location candidate, such as "Boston", which is the name of several places, including "Boston, USA" and "Boston, UK"; this fact makes geoparsing tasks on a global scale harder than local ones. de Bruijn et al. [7] uses TAGGS de Bruijn et al. [9], a geoparsing algorithm, to extract countries, administrative areas, and settlements (i.e. cities, towns, and villages) mentioned within the tweets' text on a global scale. The process includes toponym recognition and toponym resolution. Toponym recognition extracts the toponyms that refer to one or more locations using a gazetteer, a geographical index, or a dictionary. Toponym resolution predicts the correct location for the toponyms in several steps.A score is assigned to each possible location using metadata related to the tweet, such as the user's timezone & hometown, the tweet's coordinates, and mentions of nearby locations. Then, calculate the average score of grouped tweets mentioning the same toponym within a 24-hour. Finally, assign the groups of tweets with the location that has the highest score. Petersen and Styve [33] uses geotag property, geoparsing using NER on text, and user's profile location to extract toponyms. If the text contains two toponyms, check if they are close with a distance threshold of 1500km, choose one of them randomly. They use GeoPy[3] to assign geographical locations to toponyms, a Python package that is a client for several popular geocoding web services (e.g., GoogleV3 and GeoNames). Singh et al. [40] uses the fact that people visit the same locations daily to generate a Markov chain model on historical tweets created by the same user to locate them.

[3]https://geopy.readthedocs.io/en/stable/

2.4 Text Analysis

Besides text classification and location extraction, other text analysis techniques extract valuable information from text data. In the case of disasters, disaster managers can use social media to get insights, such as how impactful an event is on society. They can visualize the results to understand the situation and act accordingly.

Gründer-Fahrer, Schlaf, and Wustmann [14] extract multiple relevant pieces of information from social media and present them to disaster managers via a searchable application. They extract the following: topics using HDP-CRF algorithm [43], locations using Openstreetmap[4] location markers, time using the social media meta data, and names of organizations using NER. They present the information using several interactive graphs such as pie charts, word clouds and line graphs.

Sentiment analysis is a popular text analysis technique that shows people's sentiments during an event. Lu et al. [23] extract sentiment analysis from Twitter about the Ebola virus using three different sentiment classifiers to measure the sentiment score of the tweet depending on the majority of the votes. Also, they calculate the inconsistency between the classifiers using an entropy measure [2]. The positive and negative sentiments are each presented in a density map using solid blue and red colours, respectively; the colour is blurred insread if the inconsistency score is above a certain threshold. Periñán-Pascual [32] tries to extract the sentiment by calculate three scores for the tweets: (1) the reliability of how much the tweet discusses a problem during a hazard, (2) the impact of the tweet by using the user's activity and popularity as well as how much influence the tweet is [31], (3) and the impact of the problem using the previous scores. They present the mean of the scores on a time frame basis on a line graph.

2.5 Visualization

Visualization of the results of an NLP pipeline is common practice for several reasons. The massive and complex data can communicate the needed knowledge for different audiences to understand the underlying situation and take action accordingly. The developers can use the visualization to validate that the pipeline is working as intended. The authority can check the Spatio-temporal data to identify places that have recurring floods and reinforce their infrastructure to prepare for future flooding. Also, the plots make event detection and monitoring much faster and more straightforward.

de Bruijn et al. [7] uses historical and real-time data to show flooding events on different levels (countries, administrative areas, and settlements). It is powered using a JavaScript library, leaflet[5]. The application, seen in figure 2.1, contains a map showing the flooding events with an adjustable timeline and a list of tweets for the selected location.

[4]https://www.openstreetmap.org/
[5]https://leafletjs.com/

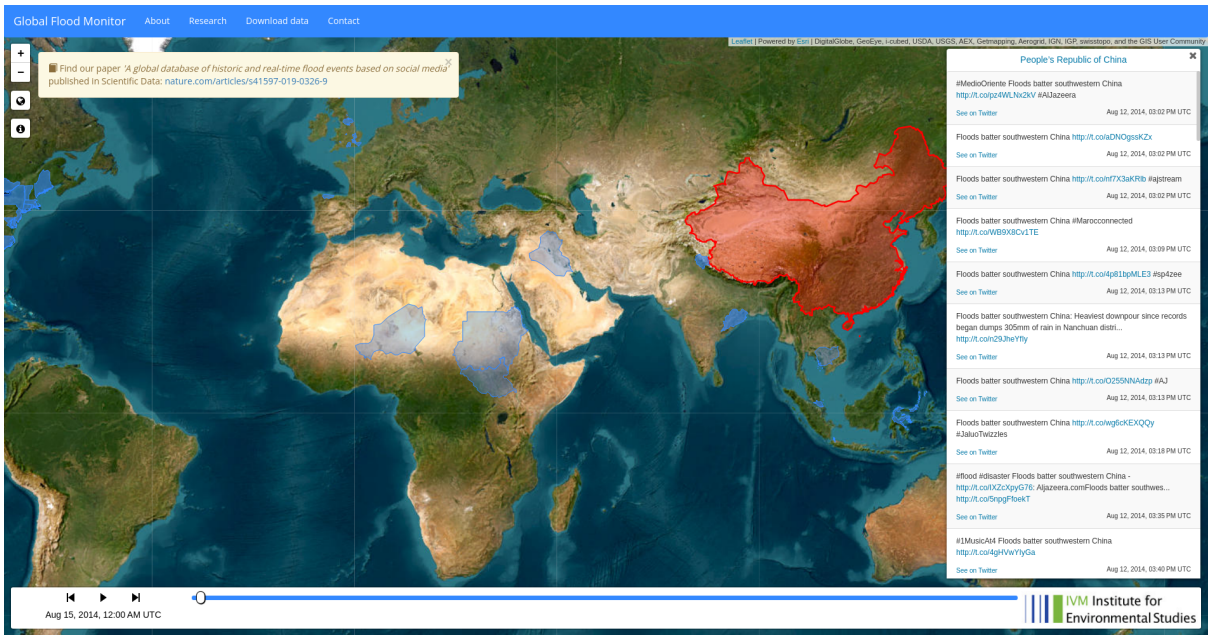

Figure 2.1: Global Flood Monitor application showing flood events

Petersen and Styve [33] provide multiple plots with sophisticated methods to configure the interface and filter the tweets. Their visualization is powered using the python libraries, Plotly[6] and Dash[7]. The app, shown in figure 2.2, provides an interface to showcase the different aspects of the data: spatial via a map, temporal via a histogram, and textual via a list of tweets and word cloud. They use a scatter map to show the locations extracted from the tweets, where the colour of each point represents the method used to identify the location. To resolve the problem of tweets overlapping each other due to the discussion of the same location, the identical points are spread by adding Gaussian noise to their coordinates points. As for representing the timestamps, they use a histogram aggregated by each day with a time slider. Researchers can pinpoint repetitive or interesting topics by navigating the word cloud to see the most frequent keywords or manually navigating the list of tweets. The plots are interactive, where actions in one of them would influence others. The data can be filtered in different ways: keywords in the text, the method used to extract the location, tweet type (a retweet or not), a map, and a histogram. In addition, there is a drop-down to change the map graph type and the algorithm used to classify the tweets.

[6]https://plotly.com/python/
[7]https://dash.plotly.com/

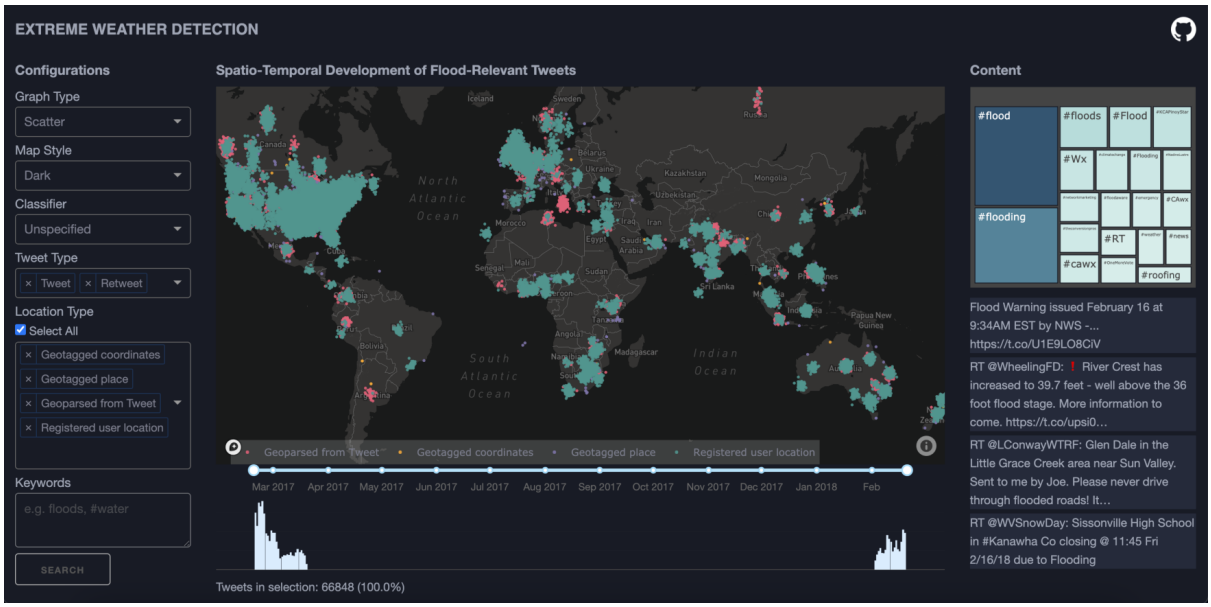

Figure 2.2: Petersen and Styve [33] application

Feng and Sester [12] use leaflet to plot a map showing flooding events as observed in figure 2.3. They use Getis-Ord Gi* [30] to detect statistical hot spots and present them as a choropleth map. The light blue circles represent the Spatio-temporal clusters of events, and the circles with numbers at the centre indicate clusters of tweets in that area with their total. The markers indicate individual tweets with a pop-up containing information about it.

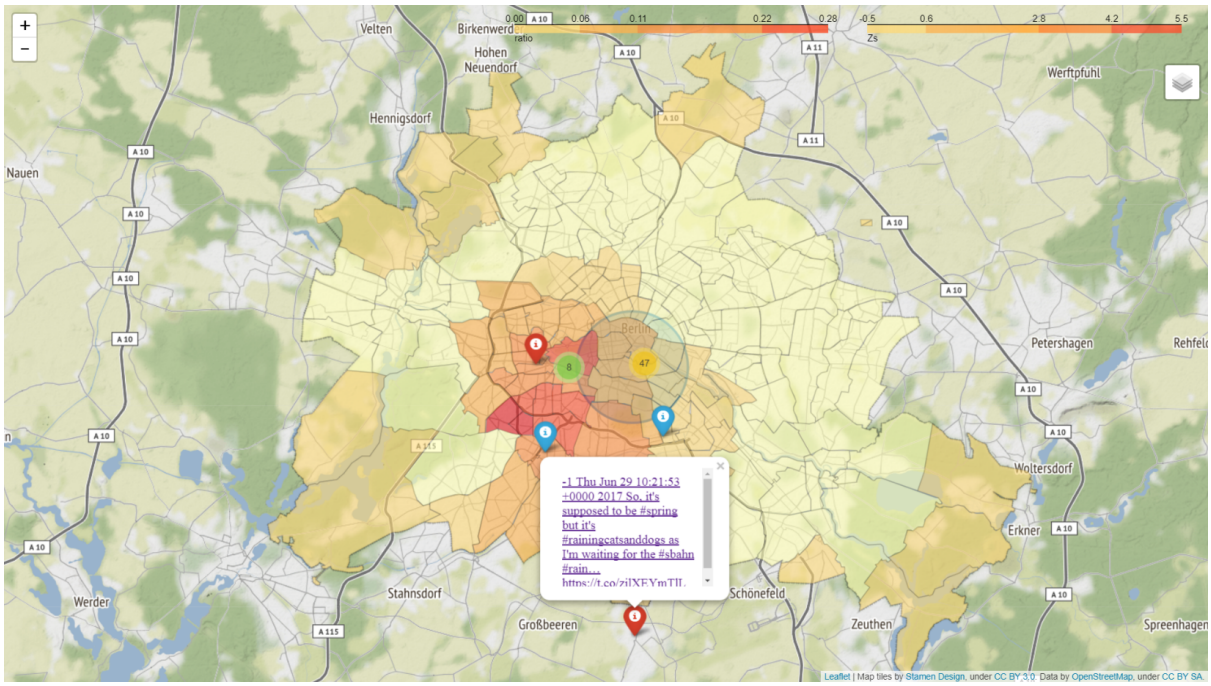

Figure 2.3: Web map application with pluvial flood in Berlin by Feng and Sester [12]

Barker and Macleod [3] visualizes the tweets using different map plots created by leaflet. The map plot in figure 2.4 consists of clickable pointers for pop-up boxes of the tweets.

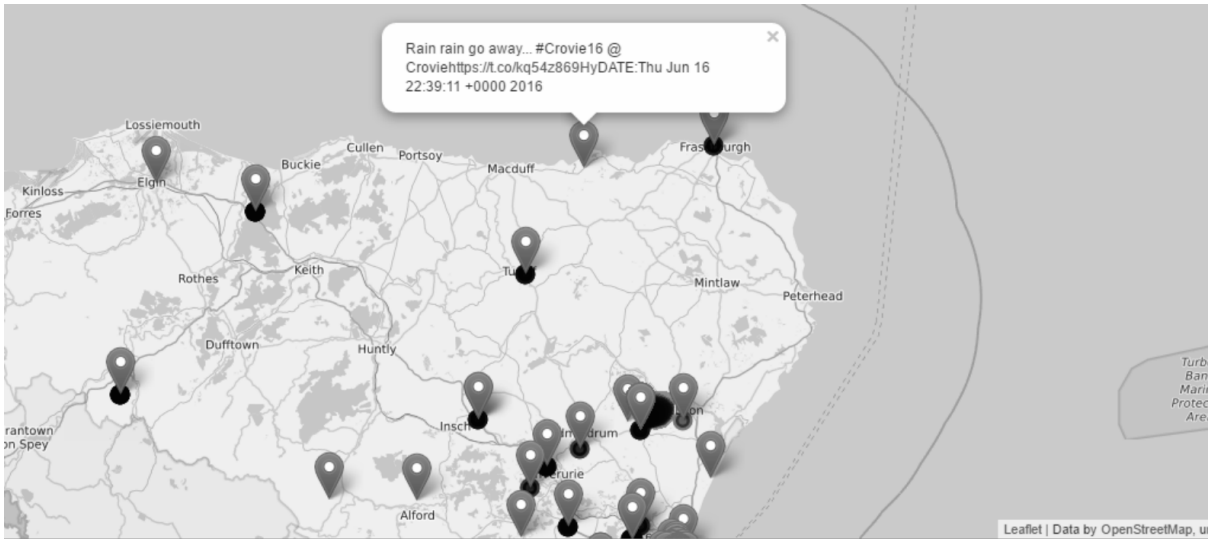

Figure 2.4: Map with tweet markers in by Barker and Macleod [3]

The bubble map in figure 2.5 displays the tweets with the size of the circles represent-

ing the area of the place and colour indicating the number of tweets talking about the location.

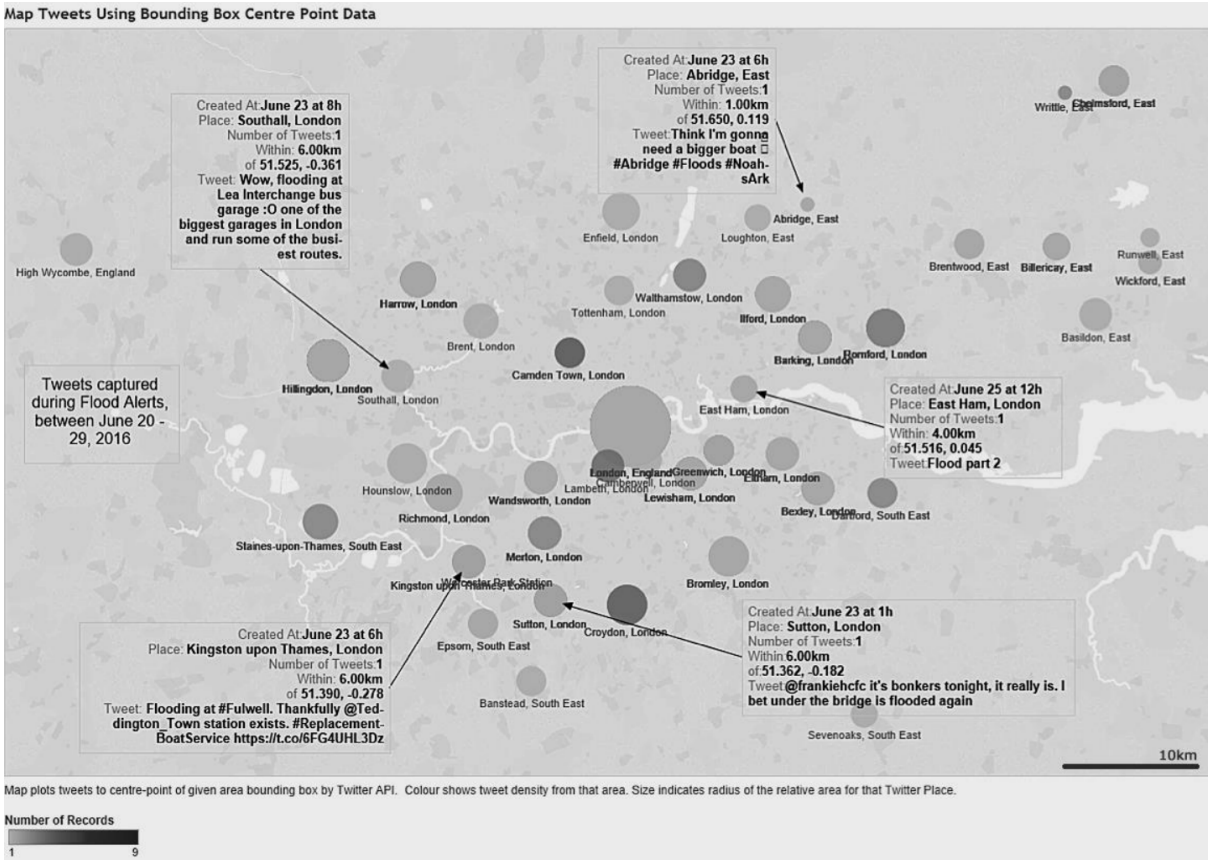

Figure 2.5: Bubble map of tweets by Barker and Macleod [3]

Chapter 3

Methods

This section discusses and motivates the methods used in the project. Figure 3.1 shows a flow chart of the pipeline (an enlarged copy is available in Figure A.1 of Appendix A). The pipeline consists of the following steps: Data collection, text classification, location extraction, and visualization. Python is the primary programming language used for the project because of the rich ecosystem surrounding it, especially when it comes to data science-related tasks. The code base is available at a GitHub repository[1] accompanied with a `README.md` containing instructions to set up the environment and run the project.

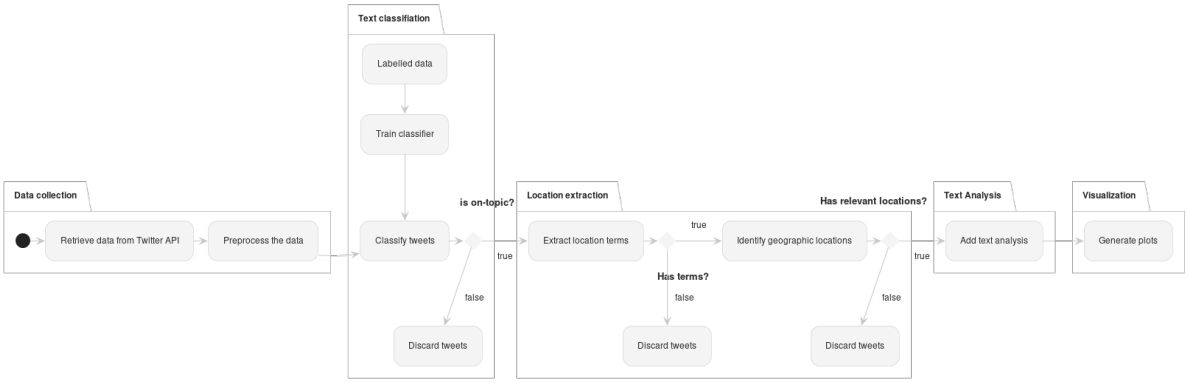

Figure 3.1: Flow chart for the pipeline

3.1 Data Collection

Finding a good quality data source is the first step to having a lean start for most research questions. The pipeline trains an ML classifier using a manually labelled dataset containing 4899 tweets with attributes presented in Table 3.1. The text and metadata of the tweets are extracted from Twitter's API using the IDs. The trained model performance is verified using the tweets from the API using Tweepy[2], a python library for accessing

[1] https://github.com/YasserKa/Classification-and-visualization-of-natural-disasters-using-Twitter
[2] https://docs.tweepy.org/en/latest/index.html

Field	Type	Description
ID	Int	ID of the tweet
On Topic	Bool	Text discusses an event
Informative sarcastic	Bool	Text contains relevant information about the event
Contains IMPACT info	Bool	Text discusses the impact of the event
Explicit location	Bool	Text mentions the location of the event

Table 3.1: Dataset attributes

Attribute	Type	Description
id	Int	The unique identifier of the requested Tweet
text	Str	The actual UTF-8 text of the Tweet
created at	Date	Creation time of the Tweet
author id	Str	The unique identifier of the tweet creator

Table 3.2: Tweet attributes used

Twitter API.

Textual data requires pre-processing for several tasks, such as training an ML algorithm, text analysis, and visualization. Parts of text that don't contribute to the context are removed: Uniform Resource Locator (URL)s, emojis, mentions, hashtag signs, numbers, new lines, punctuation, and stopwords (provided by spaCy[3], an NLP python library). Afterwards, duplicate tweets, tweets containing no text, and retweets are discarded from the dataset.

Data needs to be stored and managed to accommodate policies and regulations. Twitter's developer policy[4] has a content redistribution section stating that only the IDs of the tweets can be shared online. Thus, the tweets can't be available publicly on such as GitHub, the service that hosts the publicly available code base. To this end, the data is stored and cached after each step on google drive using Data Version Control (DVC)'s[5] data management capabilities.

Twitter's API provides an extensive list of information about the tweets[6]. It shares the engagement metrics of the tweet, including like count, reply count, and retweet count; as well as, an NLP analysis of its own, such as the language used, and entities parsed from the text. Table 3.1 shows the tweet's attributes used in this project for the following reasons: the id to generate the URL of the tweet, the text for NLP tasks, the created date for temporal analysis, and the author id to reduce spam.

[3]https://spacy.io/
[4]https://developer.twitter.com/en/developer-terms/policy
[5]https://dvc.org/doc
[6]https://developer.twitter.com/en/docs/twitter-api/data-dictionary/object-model/tweet

3.2 Text Classification

There are three groups of approaches for NLP tasks: heuristics, ML, and deep learning. The heuristics approach is the oldest one which builds rules manually for a specific task by using dictionaries and thesauruses. ML techniques, including probabilistic modelling and likelihood maximization, are used on a numerical representation of the textual data to learn a model. Neural networks are a popular choice for handling complex, and unstructured data, making them a suitable candidate for language.

RNN [16] is a common artificial neural network for NLP tasks, such as text classification, NER, and machine translation. Its memory enables it to take information from previous input to update the current input and output vector (called hidden state) as shown in Figure 3.2, making it appropriate for sequential. For common tasks such as translation an encoder-decoder architecture is needed, where the encoder encodes the input sequence into a numerical representation (called the last hidden state) that gets passed to the decoder for output sequence generation. Figure 3.3 shows an example of translating the English statement "Transformers are great!" to the German language. RNN has shortcomings when it tries to capture the context for long sequences of information, where the encoder might lose the information at the start of the sequence while forming the representation. RNN's weak memory can be addressed by using the attention mechanism that allows the decoder to access all the hidden states of the encoder. The main goal of attention is to enable the decoder to prioritize the states using weights it assigns at every decoding timestamp. Figure 3.4 shows an example for predicting the third token in the output sequence. Even though attention improves the accuracy of the translations, the computations are sequential and cannot be parallelized.

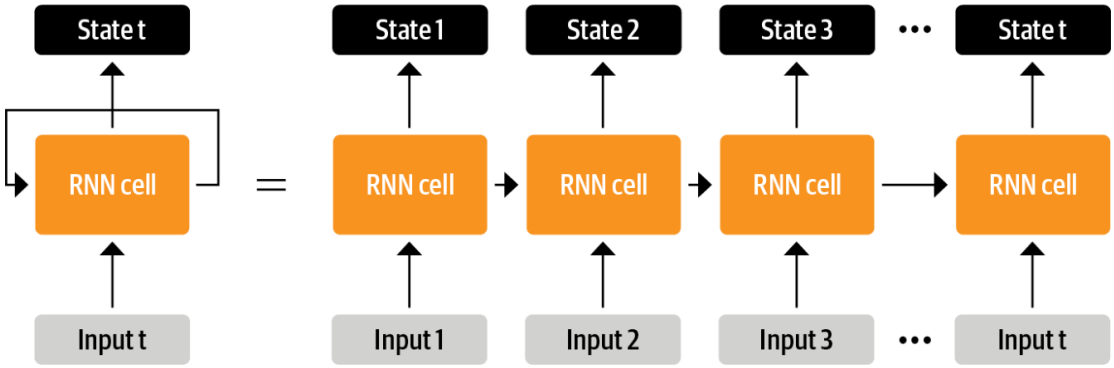

Figure 3.2: RNN example [44]

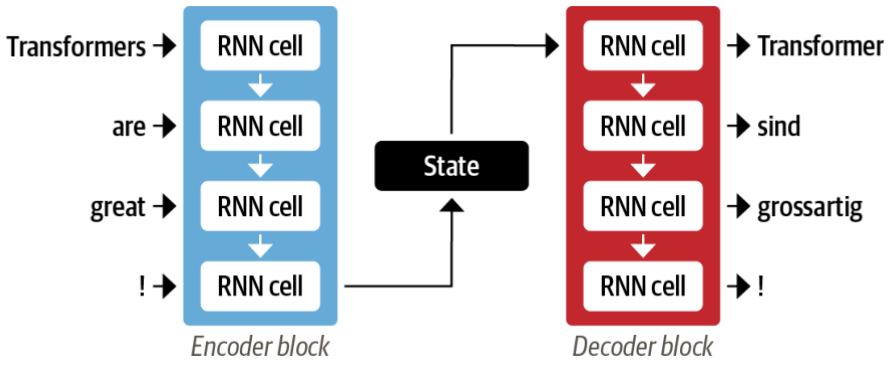

Figure 3.3: Two RNNs making an encoder-decoder architecture [44]

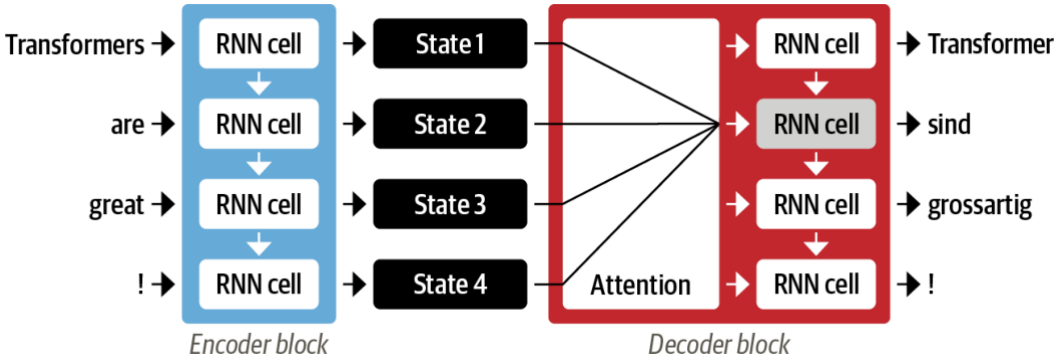

Figure 3.4: Two RNNs making an encoder-decoder architecture with attention mechanism [44]

Transformers with its self-attention architecture, proposed by google researchers [46], made the training process much faster. The idea is to use attention on all states in the same layer of the neural network. Figure 3.5 shows the self-attention mechanism on both encoder and decoder with their outputs fed to feed-forward neural networks. Transformers still require a large amount of labelled text data to train the models,which gets circumvent using transfer learning by pre-training a model on a huge dataset that gets fine-tuned afterwards on a much smaller dataset for a more specific task.

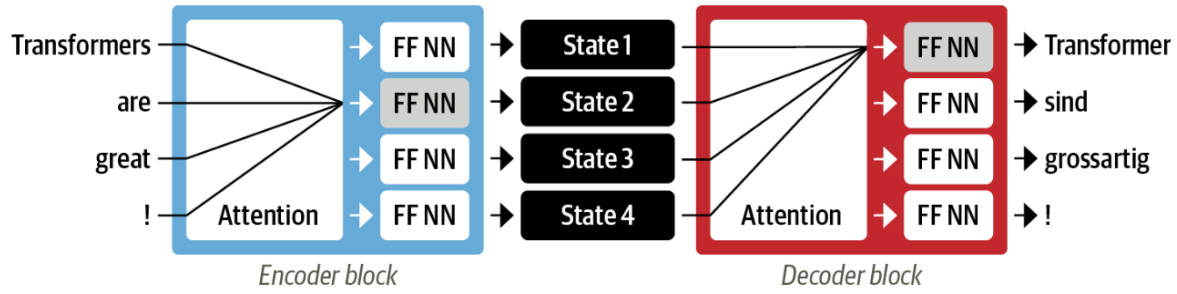

Figure 3.5: Transformer's encoder-decoder architecture [44]

This project uses the DistilBERT transformer[38], a variant of BERT, for text classification. The main advantage of this model is that it achieves comparable performance to BERT while being significantly smaller than BERT while being significantly smaller and more efficient. A DistilBERT pre-trained model is provided by Hugging Face[7], a framework that provides a unified API for over more than 50 architectures, making it easier for users to integrate NLP models into their applications.

After pre-processing the data, there are several steps to obtain the trained model. DistilBERT works on the English language, and since Sweden is the focus of the research, most of the text is in Swedish; thus, the text is translated to English using google translate[8] by a python library wrapper deep-translate[9]. The text classification purpose is to identify the tweets that discuss flood events, so the "On Topic" attribute of the dataset is used as a label during training. The learning rate for the neural network is $5 \times e^{-5}$ with 100 warmup steps over three epochs using 90% of the labelled tweets as training data, 5% as test data, and 5% for validation. Training the model locally takes a long time with the available resources, so the training is done using Amazon SageMaker[10], a service that covers tools to build, train, and deploy ML models. The data is uploaded to Amazon Simple Storage Service (Amazon S3) to make it accessible for the Hugging Face training script that is executed in an instance available in the cloud. After the training is complete, the fine-tuned model and the evaluation metric are downloaded. The evaluation metric consists of the following:

- Accuracy: the fraction of the number of correctly classified instances (i.e., true positives and true negatives) among all instances (i.e., whole dataset) (equation 3.1).

$$\text{Accuracy} = \frac{TN + TP}{TN + FN + TP + FP} \tag{3.1}$$

- Precision: the fraction of the number of correctly classified relevant instances (i.e., true positives) among the total number of instances classified as relevant (i.e., true

[7]https://huggingface.co/
[8]https://translate.google.com/
[9]https://deep-translator.readthedocs.io/en/latest/
[10]https://aws.amazon.com/sagemaker/

positives and false positives) (equation 3.2).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.2}$$

- Recall: the fraction of the correctly classified relevant instances (i.e., true positives) among all relevant instances (i.e. true positives and false negatives) (equation 3.3).

$$\text{Precision} = \frac{TP}{TP + FN} \tag{3.3}$$

- F_1 score: the harmonic mean of precision and recall (equation 3.4).

$$F_1 = 2.\frac{\text{Precision.Recall}}{\text{Precision} + \text{Recall}} \tag{3.4}$$

3.3 Location Extraction

The project uses a hybrid approach for geoparsing to extract locations. For toponym recognition, the tokens representing locations are extracted using the KBLab/bert-base-swedish-cased-ner model[11]. The model is based on BERT and fine-tuned for NER using The Stockholm-Umeå Corpus, a collection of Swedish texts from the 1990s that consists of one million words. As for toponym resolution, the location tokens are disambiguated using Nominatim and GeoNames geocoders through Geopy[12], a Python client for several popular geocoding web services. Nominatim retrieves data about the location from OpenStreetMap. The following shows the possible fields retrieved:


```
{
  "place_id": "100149",
  "licence": "Data © OpenStreetMap contributors,
             ODbL 1.0. https://osm.org/copyright",
  "osm_type": "node",
  "osm_id": "107775",
  "boundingbox": ["51.3473219", "51.6673219",
                  "-0.2876474", "0.0323526"],
  "lat": "51.5073219",
  "lon": "-0.1276474",
  "display_name": "London, Greater London, England,
                   SW1A 2DU, United Kingdom",
  "class": "place",
  "type": "city",
  "importance": 0.9654895765402,
  "icon": "https://nominatim.openstreetmap.org/
```


[11] https://huggingface.co/KBLab/bert-base-swedish-cased-ner
[12] https://geopy.readthedocs.io/en/latest

```
            images/mapicons/poi_place_city.p.20.png",
    "address": {
      "city": "London",
      "state_district": "Greater London",
      "state": "England",
      "ISO3166-2-lvl4": "GB-ENG",
      "postcode": "SW1A 2DU",
      "country": "United Kingdom",
      "country_code": "gb"
    },
    "extratags": {
      "capital": "yes",
      "website": "http://www.london.gov.uk",
      "wikidata": "Q84",
      "wikipedia": "en:London",
      "population": "8416535"
    }
}
```


The descriptions for the fields are available in the documentation[13]. The project uses the lat, lon, and display_name to represent the location on a map. In some cases, the text might contain two locations, the one with the smaller bounding box (area of corner coordinates) is used, which is, in most cases, a more specific place located in the bigger one (e.g. a street within a municipality). The geocoder services provide the ability to limit the search of the locations within a specific country. Since the project is limited to Sweden, the output can be limited using this option , reducing the false positives that happens when different countries have places with the same name.

3.4 Text analysis

Further pre-processing is done on the dataset to prepare for text analysis tasks. Lemmatisation is done on the text, using Natural Language Toolkit (NLTK)[14], to reduce words to their lemmas. Afterwards, Tokenisation is done on the corpus. Terms occurring in less than 20 documents or 5% of the documents are removed, as well as the terms mentioned in more than 75% of the documents. Bigrams that occur more than 20 times in the corpus are included, such as traffic jams, and climate change. To reduce spam, tweets created by the same user who tweeted about the same location the past week are discarded.

The text analysis used in the project are Latent Dirichlet Allocation (LDA) [34] [11], TF-IDF, and T-distributed Stochastic Neighbor Embedding (t-SNE)[45]. LDA is a topic

[13]https://nominatim.org/release-docs/develop/api/Output/
[14]https://www.nltk.org/

modelling method that generates topics (a set of terms) in a corpus and assigns the relevancy of each topic in each document. The LDA model is initialized and trained using Gensim [35], where the number of discovered topics is adjustable in the visualization. The second text analysis technique used is TF-IDF, using scikit-learn[15], to extract interesting terms by checking their average weight and frequency in the corpus. Lastly, t-SNE is a visualization method for high-dimensional data by reducing their dimensions to two or three-dimensional maps. In this project, t-SNE reduces the dimensions of a TF-IDF matrix generated from the corpus to 2-dimensional space and then presented on a scatter plot; the points are clustered before applying t-SNE using Density-Based Spatial Clustering of Applications with Noise (DBSCAN) with adjustable eps (maximum distance between neighbours), and min_samples (number of samples in a neighbourhood for the point to be considered as a core point). t-SNE, the generation of the TF-IDF matrix, and clustering are done using scikit-learn.

3.5 Visualization

Visualization is often placed at the end of the pipeline and might be the most important since it brings meaning to the results, which can be interrupted by most audiences. Also, it's a direct way to verify that the pipeline is working. The web application is made by Dash[16] to create an interface for Plotly[17]'s visualizations. Dash Bootstrap Components[18] is used as well for an easier way to use Bootstrap components for Plotly Dash, such as buttons, input, and tables.

There are several interactive graphs to enable spatial, temporal, and textual exploration of tweets. The text, created time, location extracted, and other features of the selected tweets are explored using a table. The identified locations are presented using clustered pointers on a map; tweets selection can be done by clicking on the clusters or on a different level of regions (counties or municipalities) that can be changed using radio items. The creation dates of tweets are aggregated and plotted using a histogram; they are aggregated by day if the tweets span a month or less; otherwise, by month. The results of LDA and TF-IDF are displayed in two tables showing the frequency of the terms and their mean weights. The number of topics generated by LDA can be changed using a text input, and the tables can be regenerated after changing the selected tweets by clicking a button. A scatter plot shows the t-SNE's 2-dimensional space with DBSCAN clustering, where the clustering properties (eps and min samples) can be adjusted by text inputs. Lastly, metadata about the interface is available: the total number of tweets, the number of selected tweets, the oldest and newest tweet creation dates, the total number of locations, the number of selected locations, and a list of locations' names. All elements of the interface influence each other to some extent. Selecting tweets using the map, histogram, or scatter plot will alter the view for the rest of the presentations by highlighting

[15]https://scikit-learn.org/stable/
[16]https://dash.plotly.com/
[17]https://plotly.com/python/
[18]https://dash-bootstrap-components.opensource.faculty.ai/

only the selected tweets.

3.6 Verification

The pipeline is applied on one week worth of tweets starting from a date when a flood event happened in Sweden and verified using the visualizations presented at the find step. The query used to extract the tweets contain flood relevant terms in Swedish:

```
"atmosfärisk flod" OR "hög vatten" OR åskskur
OR regnskur OR dagvattensystem OR dränering OR "höga vågor"
OR "höga flöden" OR dämmor
OR snösmältning OR blött OR oväder OR stormflod OR vattenstånd
OR vattennivå OR åskväder OR regnstorm"
OR "mycket regn" OR "kraftig regn" OR översvämningsskador
OR översvämningar OR översvämning
```

Appendix A

Diagrams

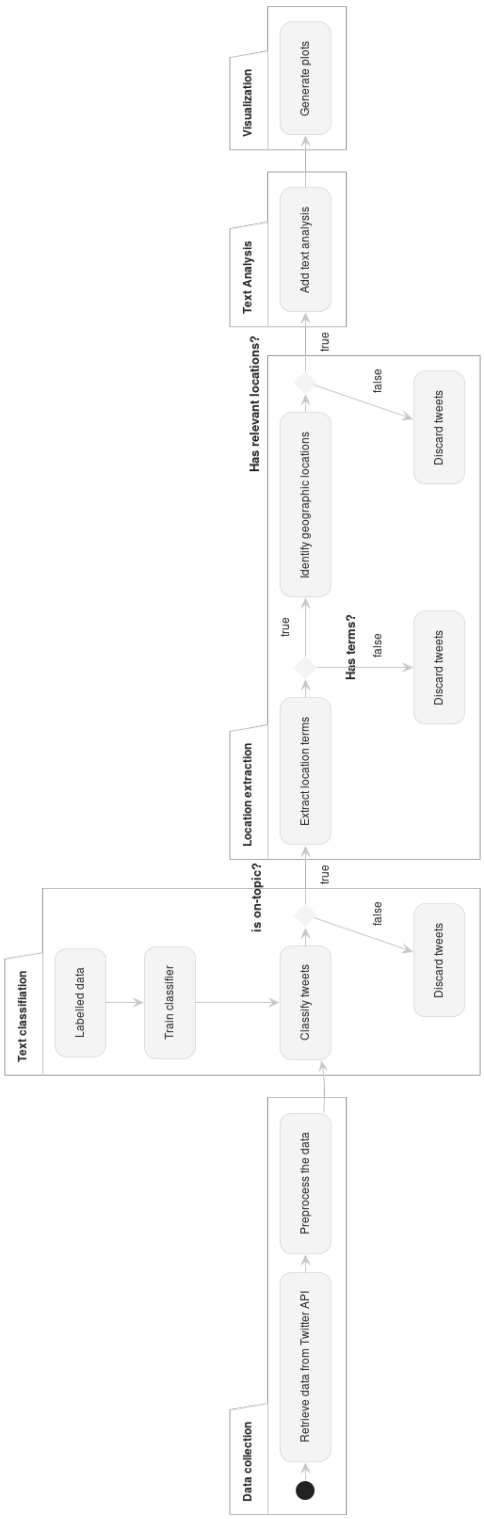

Figure A.1: Flow chart for the pipeline

References

[1] Firoj Alam et al. *Flood Detection via Twitter Streams Using Textual and Visual Features*. Version 1. Nov. 30, 2020. DOI: 10.48550/arXiv.2011.14944. arXiv: 2011.14944 [cs]. URL: http://arxiv.org/abs/2011.14944 (visited on 10/18/2022).

[2] S. Argamon-Engelson and I. Dagan. "Committee-Based Sample Selection for Probabilistic Classifiers". In: *Journal of Artificial Intelligence Research* 11 (Nov. 15, 1999), pp. 335–360. ISSN: 1076-9757. DOI: 10.1613/jair.612. arXiv: 1106.0220 [cs]. URL: http://arxiv.org/abs/1106.0220 (visited on 01/19/2023).

[3] J.L.P. Barker and C.J.A. Macleod. "Development of a National-Scale Real-Time Twitter Data Mining Pipeline for Social Geodata on the Potential Impacts of Flooding on Communities". In: *Environmental Modelling & Software* 115 (May 2019), pp. 213–227. ISSN: 13648152. DOI: 10.1016/j.envsoft.2018.11.013. URL: https://linkinghub.elsevier.com/retrieve/pii/S136481521830094X (visited on 09/07/2022).

[4] Wikipedia contributors. *Early Warning System*. In: *Wikipedia*. 1119015319th ed. Wikipedia, The Free Encyclopedia, 10/30/2022, 06:41:00 AM. URL: https://en.wikipedia.org/w/index.php?title=Early_warning_system&oldid=1119015319 (visited on 11/17/2022).

[5] Richard Davies. *Sweden – Flash Floods in Dalarna and Gävleborg After Record Rainfall*. FloodList. Aug. 19, 2021. URL: https://floodlist.com/europe/central-sweden-floods-august-2021 (visited on 11/17/2022).

[6] Jens de. *Flood Tweet IDs (Multilingual)*. Version V2. 2019. DOI: 10.7910/DVN/T3ZFMR. URL: https://doi.org/10.7910/DVN/T3ZFMR.

[7] Jens A. de Bruijn et al. "A Global Database of Historic and Real-Time Flood Events Based on Social Media". In: *Scientific Data* 6.1 (1 Dec. 9, 2019), p. 311. ISSN: 2052-4463. DOI: 10.1038/s41597-019-0326-9. URL: https://www.nature.com/articles/s41597-019-0326-9 (visited on 10/04/2022).

[8] Jens A. de Bruijn et al. "Improving the Classification of Flood Tweets with Contextual Hydrological Information in a Multimodal Neural Network". In: *Computers & Geosciences* 140 (July 2020), p. 104485. ISSN: 00983004. DOI: 10.1016/j.cageo.2020.104485. URL: https://linkinghub.elsevier.com/retrieve/pii/S0098300419308106 (visited on 11/28/2022).

[9] Jens A. de Bruijn et al. "TAGGS: Grouping Tweets to Improve Global Geoparsing for Disaster Response". In: *Journal of Geovisualization and Spatial Analysis* 2.1 (Dec. 26, 2017), p. 2. ISSN: 2509-8829. DOI: 10.1007/s41651-017-0010-6. URL: https://doi.org/10.1007/s41651-017-0010-6 (visited on 10/04/2022).

[10] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. May 24, 2019. DOI: 10.48550/arXiv.1810.04805. arXiv: 1810.04805 [cs]. URL: http://arxiv.org/abs/1810.04805 (visited on 11/26/2022).

[11] Daniel Falush, Matthew Stephens, and Jonathan K Pritchard. "Inference of Population Structure Using Multilocus Genotype Data: Linked Loci and Correlated Allele Frequencies". In: *Genetics* 164.4 (Aug. 1, 2003), pp. 1567–1587. ISSN: 1943-2631. DOI: 10.1093/genetics/164.4.1567. URL: https://academic.oup.com/genetics/article/164/4/1567/6050225 (visited on 01/26/2023).

[12] Yu Feng and Monika Sester. "Extraction of Pluvial Flood Relevant Volunteered Geographic Information (VGI) by Deep Learning from User Generated Texts and Photos". In: *ISPRS International Journal of Geo-Information* 7.2 (2 Feb. 2018), p. 39. ISSN: 2220-9964. DOI: 10.3390/ijgi7020039. URL: https://www.mdpi.com/2220-9964/7/2/39 (visited on 09/07/2022).

[13] *Floodlist*. FloodList. Aug. 19, 2021. URL: https://floodlist.com/europe/central-sweden-floods-august-2021 (visited on 11/17/2022).

[14] Sabine Gründer-Fahrer, Antje Schlaf, and Sebastian Wustmann. "How Social Media Text Analysis Can Inform Disaster Management". In: *Language Technologies for the Challenges of the Digital Age*. Ed. by Georg Rehm and Thierry Declerck. Vol. 10713. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 199–207. ISBN: 978-3-319-73705-8 978-3-319-73706-5. DOI: 10.1007/978-3-319-73706-5_17. URL: http://link.springer.com/10.1007/978-3-319-73706-5_17 (visited on 01/17/2023).

[15] Kaiming He et al. *Deep Residual Learning for Image Recognition*. Dec. 10, 2015. DOI: 10.48550/arXiv.1512.03385. arXiv: 1512.03385 [cs]. URL: http://arxiv.org/abs/1512.03385 (visited on 01/04/2023).

[16] J J Hopfield. "Neural Networks and Physical Systems with Emergent Collective Computational Abilities." In: *Proceedings of the National Academy of Sciences* 79.8 (Apr. 1982), pp. 2554–2558. DOI: 10.1073/pnas.79.8.2554. URL: https://www.pnas.org/doi/10.1073/pnas.79.8.2554 (visited on 01/24/2023).

[17] Jeremy Howard and Sebastian Ruder. *Universal Language Model Fine-tuning for Text Classification*. May 23, 2018. DOI: 10.48550/arXiv.1801.06146. arXiv: 1801.06146 [cs, stat]. URL: http://arxiv.org/abs/1801.06146 (visited on 01/04/2023).

[18] Gao Huang et al. *Densely Connected Convolutional Networks*. Jan. 28, 2018. DOI: 10.48550/arXiv.1608.06993. arXiv: 1608.06993 [cs]. URL: http://arxiv.org/abs/1608.06993 (visited on 01/04/2023).

[19] Alex Krizhevsky, Ilya Sutskever, and zz Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Communications of the ACM* 60.6 (May 24, 2017), pp. 84–90. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/3065386. URL: https://dl.acm.org/doi/10.1145/3065386 (visited on 12/15/2022).

[20] Quoc V. Le and Tomas Mikolov. *Distributed Representations of Sentences and Documents*. May 22, 2014. DOI: 10.48550/arXiv.1405.4053. arXiv: 1405.4053 [cs]. URL: http://arxiv.org/abs/1405.4053 (visited on 12/30/2022).

[21] Hongmin Li et al. "Disaster Response Aided by Tweet Classification with a Domain Adaptation Approach". In: *Journal of Contingencies and Crisis Management* 26.1 (Mar. 2018), pp. 16–27. ISSN: 0966-0879, 1468-5973. DOI: 10.1111/1468-5973.12194. URL: https://onlinelibrary.wiley.com/doi/10.1111/1468-5973.12194 (visited on 09/11/2022).

[22] Quanzhi Li et al. "How Much Data Do You Need? Twitter Decahose Data Analysis". In: July 2016.

[23] Yafeng Lu et al. "Visualizing Social Media Sentiment in Disaster Scenarios". In: *Proceedings of the 24th International Conference on World Wide Web*. WWW '15: 24th International World Wide Web Conference. Florence Italy: ACM, May 18, 2015, pp. 1211–1215. ISBN: 978-1-4503-3473-0. DOI: 10.1145/2740908.2741720. URL: https://dl.acm.org/doi/10.1145/2740908.2741720 (visited on 12/16/2022).

[24] Stuart E. Middleton, Lee Middleton, and Stefano Modafferi. "Real-Time Crisis Mapping of Natural Disasters Using Social Media". In: *IEEE Intelligent Systems* 29.2 (Mar. 2014), pp. 9–17. ISSN: 1541-1672. DOI: 10.1109/MIS.2013.126. URL: http://ieeexplore.ieee.org/document/6692841/ (visited on 10/19/2022).

[25] Stuart E. Middleton et al. "Location Extraction from Social Media: Geoparsing, Location Disambiguation, and Geotagging". In: *ACM Transactions on Information Systems* 36.4 (June 13, 2018), 40:1–40:27. ISSN: 1046-8188. DOI: 10.1145/3202662. URL: https://doi.org/10.1145/3202662 (visited on 10/19/2022).

[26] Tomas Mikolov et al. *Efficient Estimation of Word Representations in Vector Space*. Sept. 6, 2013. DOI: 10.48550/arXiv.1301.3781. arXiv: 1301.3781 [cs]. URL: http://arxiv.org/abs/1301.3781 (visited on 12/30/2022).

[27] Tina Neset. *AI4ClimateAdaptation*. Linköping University. URL: https://liu.se/en/research/ai4climateadaptation (visited on 11/18/2022).

[28] Huan Ning et al. "Prototyping a Social Media Flooding Photo Screening System Based on Deep Learning". In: *ISPRS International Journal of Geo-Information* 9.2 (2 Feb. 2020), p. 104. ISSN: 2220-9964. DOI: 10.3390/ijgi9020104. URL: https://www.mdpi.com/2220-9964/9/2/104 (visited on 09/11/2022).

[29] Alexandra Olteanu et al. "CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises". In: *Proceedings of the International AAAI Conference on Web and Social Media* 8.1 (May 16, 2014), pp. 376–385. ISSN: 2334-0770, 2162-3449. DOI: 10.1609/icwsm.v8i1.14538. URL: https://ojs.aaai.org/index.php/ICWSM/article/view/14538 (visited on 11/28/2022).

[30] J. K. Ord and Arthur Getis. "Local Spatial Autocorrelation Statistics: Distributional Issues and an Application". In: *Geographical Analysis* 27.4 (Sept. 3, 2010), pp. 286–306. ISSN: 00167363. DOI: 10.1111/j.1538-4632.1995.tb00912.x. URL: https://onlinelibrary.wiley.com/doi/10.1111/j.1538-4632.1995.tb00912.x (visited on 01/08/2023).

[31] Aditya Pal and Scott Counts. "Identifying Topical Authorities in Microblogs". In: *Proceedings of the Fourth ACM International Conference on Web Search and Data Mining*. WSDM'11: Fourth ACM International Conference on Web Search and Data Mining. Hong Kong China: ACM, Feb. 9, 2011, pp. 45–54. ISBN: 978-1-4503-0493-1. DOI: 10.1145/1935826.1935843. URL: https://dl.acm.org/doi/10.1145/1935826.1935843 (visited on 01/19/2023).

[32] Carlos Periñán-Pascual. "Assessing the Impact of Tweets in Flood Events". In: *1st International Workshop on Social Media Analysis for Intelligent Environment* (Jan. 1, 2020). URL: https://www.academia.edu/44757497/Assessing_the_Impact_of_Tweets_in_Flood_Events (visited on 12/16/2022).

[33] Julie Maria Petersen and Lise Styve. "Identification and Exploration of Extreme Weather Events From Twitter Data". Linköping University, 2021.

[34] Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. "Inference of Population Structure Using Multilocus Genotype Data". In: *Genetics* 155.2 (June 1, 2000), pp. 945–959. ISSN: 1943-2631. DOI: 10.1093/genetics/155.2.945. URL: https://academic.oup.com/genetics/article/155/2/945/6048111 (visited on 01/26/2023).

[35] Radim Řehůřek and Petr Sojka. "Software Framework for Topic Modelling with Large Corpora". In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 22, 2010, pp. 45–50.

[36] *River Floods Sweden*. ClimateChangePost. Nov. 6, 2022. URL: https://www.climatechangepost.com/sweden/river-floods/ (visited on 11/17/2022).

[37] Naina Said et al. *Floods Detection in Twitter Text and Images*. Nov. 30, 2020. DOI: 10.48550/arXiv.2011.14943. arXiv: 2011.14943 [cs]. URL: http://arxiv.org/abs/2011.14943 (visited on 11/26/2022).

[38] Victor Sanh et al. "DistilBERT, a Distilled Version of BERT: Smaller, Faster, Cheaper and Lighter". In: *ArXiv* abs/1910.01108 (2019).

[39] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Apr. 10, 2015. DOI: 10.48550/arXiv.1409.1556. arXiv: 1409.1556 [cs]. URL: http://arxiv.org/abs/1409.1556 (visited on 12/15/2022).

[40] Jyoti Prakash Singh et al. "Event Classification and Location Prediction from Tweets during Disasters". In: *Annals of Operations Research* 283.1 (Dec. 1, 2019), pp. 737–757. ISSN: 1572-9338. DOI: 10.1007/s10479-017-2522-3. URL: https://doi.org/10.1007/s10479-017-2522-3 (visited on 09/07/2022).

[41] *SMHI*. SMHI - Who we are. Apr. 30, 2021.

[42] Christian Szegedy et al. "Rethinking the Inception Architecture for Computer Vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2818–2826. DOI: 10.1109/CVPR.2016.308.

[43] Yee Whye Teh and Michael I. Jordan. "Hierarchical Bayesian Nonparametric Models with Applications". In: *Bayesian Nonparametrics*. Ed. by Chris Holmes et al. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge: Cambridge University Press, 2010, pp. 158–207. ISBN: 978-0-521-51346-3. DOI: 10.1017/CBO9780511802478.006. URL: https://www.cambridge.org/core/books/bayesian-nonparametrics/hierarchical-bayesian-nonparametric-models-with-applications/0051298A8C5D57586096CDDF02AB1B0F (visited on 01/19/2023).

[44] Lewis Tunstall et al. *Natural Language Processing with Transformers: Building Language Applications with Hugging Face*. Revised edition. Sebastopol: O'Reilly, 2022. 383 pp. ISBN: 978-1-09-813679-6.

[45] L.J.P. van der Maaten and G.E. Hinton. "Visualizing High-Dimensional Data Using t-SNE". In: *Journal of Machine Learning Research* 9 (nov 2008), pp. 2579–2605. ISSN: 1532-4435.

[46] Ashish Vaswani et al. *Attention Is All You Need*. Dec. 5, 2017. DOI: 10.48550/arXiv.1706.03762. arXiv: 1706.03762 [cs]. URL: http://arxiv.org/abs/1706.03762 (visited on 01/24/2023).

[47] Wikipedia contributors. *Tf–Idf — Wikipedia, the Free Encyclopedia*. 2022. URL: https://en.wikipedia.org/w/index.php?title=Tf%E2%80%93idf&oldid=1123031029.

[48] Bolei Zhou et al. "Learning Deep Features for Scene Recognition Using Places Database". In: *Advances in Neural Information Processing Systems*. Vol. 27. Curran Associates, Inc., 2014. URL: https://papers.nips.cc/paper/2014/hash/3fe94a002317b5f9259f82690aeea4cd-Abstract.html (visited on 12/15/2022).