M.Sc. Thesis

Master of Science in Engineering

Identification and Exploration of Extreme Weather Events From Twitter Data

Authors:
Julie Maria Petersen, s164510
Lise Styve, s153748

Supervisors:
Sune Lehmann
Katerina Vrotsou

Kongens Lyngby 2021

DTU Compute
Department of Applied Mathematics and Computer Science
Technical University of Denmark

Matematiktorvet, Building 303B
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk

Abstract

Extreme weather events are becoming more frequent and intense due climate change. When disaster events are emerging, up-to-date information from eye-witnesses is crucial for early detection and resilience. Recently, Twitter has become an important source of volunteered geographic information of key value for global monitoring systems and for increasing situational awareness. Hence, in this project, a pipeline of several steps was developed for identification of flood events from relevant textual data retrieved from Twitter. Initially, artificial intelligence based approaches were used for the task of classifying Tweets as relevant or not to a flood event. Specifically, four text classifiers were build using the classic algorithms, logistic regression and random forest, deep learning in the form of a convolutional neural network and lastly the transfer learning technique called universal language model fine-tuning. The highest accuracy was achieved using transfer learning, though promising accuracies above 90% were obtained with all approaches. Inspection of the classification of Tweets from a use case however revealed that further training and testing is needed to increase the reliability of the classifiers. Secondly, to detect areas with risk of flooding, a location extraction algorithm was constructed with use of geotags and geoparsing to relate the Tweets to a geographic location. Lastly, an interactive visual interface was developed using multiple coordinated views to enable exploration across the spatial, temporal and textual aspects of the Tweets. A demonstration of the interface on a use case enabled comparison to historical records showing the relevance of the pipeline. In general, the pipeline establishes a baseline of how to use Twitter data for flood detection as a supplement to other traditional monitoring systems. This can potentially be expanded to include more information sources and identify different kinds of extreme weather events.

Resumé

Ekstreme vejrbegivenheder bliver hyppigere og mere intense på grund af klimaændringer. Når katastrofehændelser opstår, er tidssvarende information fra øjenvidner afgørende for at kunne opdage dem tidligere og reagere hurtigere. For nylig er Twitter blevet et vigtigt supplement til globale overvågningssystemer og kan hjælpe til at øge situationskendskabet under katastrofer. I dette projekt blev der derfor udviklet en pipeline med flere trin til identifikation af oversvømmelser fra relevant tekstdata hentet fra Twitter. Indledningsvist blev tilgange baseret på kunstig intelligens brugt til at klassificere Tweets som relevante eller ikke for oversvømmelser. Specifikt blev fire modeller brugt til klassifikation af tekst hvilket indebar de klassiske algoritmer, logistisk regression og Random Forest, Deep Learning i form af et Convolutional Neural Network samt Transfer Learning metoden Universal Language Model Fine-Tuning. Den højeste accuracy blev opnået ved Transfer Learning, dog opnåede alle tilgange lovende accuracies over 90%. Undersøgelser af klassifikationer for Tweets fra en use case viste dog imidlertid, at der er behov for yderligere træning og test af modellerne for at øge deres pålidelighed. Det næste trin indebar konstruktion af en algoritme, der kunne relatere Tweetsene til en geografisk placering med den hensigt at kunne opdage områder med risiko for oversvømmelse. Som sidste trin blev en interaktiv visuel brugerflade udviklet for at muliggøre udforskning på tværs af de geografiske, tidsmæssige og tekstmæssige aspekter af Tweetsene. En demonstration af brugerfladen på en use case muliggjorde sammenligning med historisk data, der underbyggede pipelinens relevans. Generelt etablerer pipelinen et grundlag for, hvordan man kan bruge data fra Twitter til opdagelse af oversvømmelser som et supplement til andre traditionelle overvågningssystemer. Dette kan potentielt udvides til at omfatte flere informationskilder og identificere forskellige typer af ekstreme vejrhændelser.

Preface

This M.Sc. thesis was prepared at the Department of Applied Mathematics and Computer Science at the Technical University of Denmark (DTU) in fulfilment of the requirements for acquiring a *Master of Science in Engineering* degree, in *Business Analytics* for Julie Maria Petersen and *Mathematical Modelling and Computation* for Lise Styve. The thesis was completed as a part of the research project AI for Climate Adaptation at Linköping University (LiU) in collaboration with the Swedish Meteorological and Hydrological Institute. The project corresponded to 60 ECTS, lasted from 25th of February to 25th of June 2021 under supervision by Sune Lehmann from DTU and Katerina Vrotsou from LiU.

Lyngby, 25-June-2021

Julie Maria Petersen, s164510

Lise Styve, s153748

Acknowledgements

We would first like to thank our thesis advisor Katerina Vrotsou from the Department of Science and Technology at LiU that helped shape the project. She consistently supported us throughout the process and her excellent knowledge within the field of visual analytics was very useful. A special thanks also goes to our second advisor Sune Lehmann from DTU Compute. We were lucky to have him as a professor in the course Social Data and Visualisation from which we learned several methods used in this project. His teaching style and enthusiasm made a strong impression on us. We also want to thank Carlo Navarra from LiU for answering all our questions with regards to web application development. At last, we would like to thank the research partners for the AI for Climate Adaptation project that wanted to discuss our work during a webinar in May, 2021.

Contents

Abstract i

Resumé iii

Preface v

Acknowledgements vii

List of Figures xiv

List of Tables xv

List of Acronyms xvii

1 Introduction 1

2 Related Work 5
2.1 Data Collection and Location Extraction 6
2.2 Text Classification 7
2.3 Data Visualisation 10

3 Methods 15
3.1 Data Collection 16
3.1.1 Labelled Data 19
3.1.2 Unlabelled Data 21
3.2 Text Classification 21
3.2.1 Classic Algorithms 24
3.2.2 Deep Learning 26
3.2.3 Transfer Learning 33
3.2.4 Performance Measures 36

3.3 Location Extraction 38
3.4 Data Visualisation 40
3.4.1 Purpose 40
3.4.2 Design Requirements 42
3.4.3 Interface Design and Implementation 43

4 Results **47**
4.1 Performance of Text Classifiers 47
4.1.1 Classic Algorithms 48
4.1.2 Deep Learning 52
4.1.3 Transfer Learning 53
4.1.4 Comparison of Classifiers 54
4.2 Visual Interface 55
4.2.1 Interface Description 55
4.2.2 Usage and Interaction 56
4.3 Evaluation of Project Pipeline 61
4.3.1 Text Classification 61
4.3.2 Location Extraction 65
4.3.3 Interface Demonstration 68

5 Discussion **75**
5.1 Results and Methods 75
5.2 Future Research 79

6 Conclusion **81**

A Appendix **83**

Bibliography **87**

List of Figures

2.1 Dashboard prototype by Barker et al. with flood alerts in June 2016 created using the libraries Leaflet and Folium 10
2.2 Alternative dashboard prototype by Barker et al. with flood alerts in June 2016 created using Tableau 11
2.3 Web map application by Feng et al. with pluvial flood in Berlin on June 29, 2017 12
2.4 The Global Flood Monitor by de Bruijn et al. with floods in the UK on June 14, 2019 created using the Leaflet library 12
2.5 OmniSci Tweetmap by Dong et al. using the search word 'flood' 13

3.1 Project pipeline with the steps 1) data collection, 2) text classification, 3) location extraction and 4) data visualisation 15
3.2 Examples of Tweets from the CrisisLexT6 collection labelled as relevant 20
3.3 Examples of Tweets from the CrisisLexT6 collection labelled as non-relevant 20
3.4 Example of the structure for a random forest classifier for a binary classification problem with n decision trees 26
3.5 The Sigmoid, Softmax and ReLU activation functions 28
3.6 Word embedding visualisations using t-SNE on the 10 most similar words to the four selected words, weather, awesome, thunderstorm and week, that are marked with a blue diamond 30
3.7 The relations of the four selected words, weather, awesome, thunderstorm and week, in the full word embedding space 30
3.8 Example of convolution over word sequences in the CNN with a kernel of size 2×3. 31

3.9 Example of 1D convolution applied to word embeddings producing a feature vector. The vector is transformed using a ReLU function, wherafter a maxpooling layer then takes only the maximum value of the vector. 32
3.10 Final architecture of the CNN with an embedding layer, one convolutional layer using the ReLU function, one global max-pooling layer, a dense layer and an output layer using the sigmoid function. 33
3.11 The ULMFiT structure as presented in the paper by Howard and Ruder [29] with the three steps 1) language model pre-training, 2) language model fine-tuning and 3) classifier fine-tuning 34
3.12 Confusion matrix 36
3.13 Flow chart for the location extraction algorithm used to locate the Tweets in the four levels 1) geotagged coordinates, 2) geotagged place, 3) geoparsed from Tweet and 4) registered user location . . 38
3.14 The sense-making loop for visual analytics 41
3.15 The visual interface organised in multiple coordinated views: A. Selection pane, B. Spatial data, C. Temporal data and D. Textual data 43
3.16 The spread of 44 Tweets all located in Queensland by geoparsing with $\mu = (-22, 144)$ marked with pink 44

4.1 Word clouds that show the words classified as relevant and non-relevant by the logistic regression model variant 'Original Tweets' 49
4.2 Word clouds that show the words classified as relevant and non-relevant by the logistic regression model variant 'Remove keywords' 49
4.3 Randomly chosen decision tree from the random forest of 100 decision trees for model variant 'Remove keywords'. Two subtrees are marked for further inspection in Figure 4.4 51
4.4 Two subtrees from the randomly chosen decision tree in Figure 4.3 for model variant 'Remove keywords' 51
4.5 Validation and training accuracy as well as validation and training loss over 10 epochs for the CNN model variant 'Remove keywords' 52
4.6 Accuracy and F1 score for the four text classifiers using the three model variants. The average for the variants is marked with a black line. 54
4.7 The visual interface with multiple coordinated views: A. Configuration pane, B. Spatial data, C. Temporal data and D. Textual data 55
4.8 Timeline slider demonstration of hovering and selection options . 56
4.9 One-level dropdown menu for selection of classification and multi-level drowpdown menus for tweet Type and location type 57
4.10 Keyword or hashtag filtering option 57
4.11 Treemap with most frequent hashtags used in the Tweets in selection 58

4.12 Dropdown menus for selection of the graph type and map style . 58
4.13 Scatter map showing the spatial distribution of the flood-relevant Tweets . . . 59
4.14 Density heatmap visualising spatial differences of the density of flood-relevant Tweets . . . 60
4.15 Hexbin map visualising the spatial differences of the number of flood-relevant Tweets . . . 60
4.16 Total percentages of Tweets classified as relevant by each of the classifiers, by all classifiers and by at least one classifier for the variants 'Original Tweets' and 'Remove keywords'. The average for the variants is marked with a black line. . . . 62
4.17 Examples of Tweets classified as relevant by the variant 'Remove keywords' but non-relevant by the variant 'Original Tweets' . . . 63
4.18 Heatmap showing the similarity between the predictions made by the four classifiers for the variant 'Remove keywords' . . . 63
4.19 Examples of Tweets classified differently by the individual classifiers for the variant 'Remove keywords' . . . 64
4.20 Tweets that were classified as relevant by all classifiers . . . 65
4.21 Tweets that were classified as non-relevant by all classifiers . . . 65
4.22 Percentages of Tweets located by each of the location levels over the years 2016-2018 and in total . . . 66
4.23 Locations geoparsed from Tweets using NER as well as registered user locations . . . 67
4.24 Initial overview of the flood-relevant English Tweets in 2017-18 with no selections made . . . 68
4.25 Scatter map for spatial exploration of the Tweets excluding the location type 'Registered user location' . . . 69
4.26 Hexbin map for spatial exploration of the Tweets excluding the location type 'Registered user location" . . . 70
4.27 Histogram for temporal exploration of the number of Tweets over time showing two peaks . . . 70
4.28 Selection of peak in Feb-Mar 2017 on histogram and zoom to the UK area on scatter map . . . 71
4.29 Details-on-demand for Tweet in the UK . . . 71
4.30 Treemap showing the 20 most frequent hashtags used in the selected Tweets in UK . . . 72
4.31 Keyword search in the configuration pane using the word 'doris' to select Tweets related to this storm . . . 72
4.32 Inspection of dates in the histogram showing Tweet count for highest peak on February 23, 2017 . . . 73
4.33 Investigation of Tweets related to Storm Doris in scattermap, treemap and scroll-able table . . . 73
4.34 Headline and lead paragraph from the Guardian article on Storm Doris and flooding in the UK from February 23, 2017 . . . 74

4.35 Global Flood Monitor showing flood related Tweets in the UK on February 23, 2017 74

A.1 Enlarged project pipeline 84
A.2 Enlarged visualisation of the word embedding visualisations using t-SNE on the 10 most similar words to four selected words 85

List of Tables

3.1 Selected variables from the Tweet Data Dictionary 16
3.2 Variables added or updated in the data pre-processing 18
3.3 Data visualisation variables 45

4.1 Logistic regression performance measures 48
4.2 Random forest performance measures 50
4.3 CNN performance measures after 5 epochs 52
4.4 ULMFiT performance measures 53
4.5 Number of Tweets and the percentages of Tweets classified as relevant by each classifier, by all classifiers and by at least one classifier from 2016-2018 and in total for the variants 'Original Tweets' and 'Remove keywords' separated with a forward slash . 61
4.6 The number of Tweets classified as relevant by at least one classifier and percentages of these Tweets located by each of the location levels over the years 2016-2018 and in total 66
4.7 Questionable locations found by NER 67

List of Acronyms

AI	Artificial Intelligence
BERT	Bidirectional Encoder Representations from Transformers
CNN	Convolutional Neural Network
FN	False Negatives
FP	False Positives
LR	Logistic Regression
LSTM	Long Short-Term Memory
NER	Named Entity Recognition
NLP	Natural Language Processing
NT-ASGD	Non-Monotonically Triggered Averaged SGD
ReLU	Rectified Linear Unit
RF	Random Forest
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
TF-IDF	Frequency Inverse Document Frequency
TN	True Negatives
TP	True Positives

t-SNE t-distributed Stochastic Neighbor Embedding

ULMFiT Universal Language Model Fine-tuning

VGI Volunteered Geographic Information

Chapter 1

Introduction

The increase in intensity and frequency of extreme weather events in the recent years is one of the most visible consequences of human-induced climate changes [70]. As the rainfall patterns are changing and the sea levels are rising, flooding is recognized as one of the major concerns by the World Health Organization [71]. Floods, or inundation of land that is normally dry, are among the most common and destructive natural disasters, causing more than $40 billion annually in damage worldwide [46]. Only in 2020, it lead to 200 deaths in Kenya and Uganda in May, 550 deaths in Bangladesh after the most prolonged monsoon flooding of the century, 219 deaths in southern China from June to August and another 145 deaths after record flooding in Nigeria and Sudan in September [70].

Understanding the nuances of the different kinds of floods is central in the monitoring and development of flood detection systems. Fluvial floods are caused when consistent rain or snow melt causes a river to exceed its capacity, whereas coastal floods are caused by storm surges related to tropical cyclones and tsunamis [71]. Such flood types can often be monitored in real time using tide and river gauges [24]. Pluvial or flash floods are on the contrary more challenging to monitor as they are caused by excessive rainfall that rapidly raises the water heights. It is therefore desirable to develop a system to efficiently detect extreme pluvial flood events.

To follow and report global events, millions of people turn to online social media platforms such as Twitter[1]. With over 500 million Twitter posts issued every day, it has become a worldwide platform for public expression and interaction. Attempts at citizen journalism can be observed as social media users tend to provide relevant information about what is happening in the world around them. Social media enables individuals to act as sensors and even data sources themselves. This has drawn attention to research of social sensing, which examines how human-sourced data can be collected and used to improve situational awareness and early event detection [5].

The direct, real-time nature of social media sites enables extraction of up-to-date information from people in affected areas during extreme weather events. A study on social media communications across crises has shown that during natural disasters, 12% of Twitter posts come directly from eyewitnesses [47]. As the collection and communication of relevant first hand information is a key challenge during disaster events [51], social media has become an increasingly important source of information for disaster event detection and monitoring [30, 62].

With the rise of the GPS-enabled platforms, social media entries are in a larger extent not only temporally annotated but also geographically referenced. Thus, data can be collected from local sources experiencing a real-world phenomena. When pinpointed on a map, the information on an event becomes more actionable [69]. The huge amount of spatial information that Twitter provides makes it a central resource of free and open volunteered geographic information (VGI) [17]. The potential and value of the continuous flow of user-generated data that can be extracted from social media are not to be underestimated. Hence, the need for new approaches to collect, analyse and monitor the data streams arises. The continuous development within artificial intelligence (AI), in combination with VGI, provides new opportunities for methods that can facilitate effective monitoring and adaptation services.

Twitter-based monitoring systems enable rapid detection of extreme events, which is needed especially for regions that lack meteorological or geological infrastructure. A study has shown that about 75% of earthquakes detected by Twitter occurred within two minutes, thereby outpacing traditional geological survey detections [22]. This prompt identification enables effective decision making by the emergency managers and communication of essential information to rescue teams as well as victims, which can reduce losses. In addition to providing a two-way information channel, the monitoring can be used to better understand the related risks and their impacts [38].

[1] www.twitter.com

This thesis project is defined as a part of an ongoing research project, AI for Climate Adaptation[2], at Linköping University (LiU) in collaboration with the Swedish Meteorological and Hydrological Institute (SMHI). The research is concerned with developing AI-based methods as part of an early warning system for extreme weather. Project manager and associate professor, Tina-Simone Neset, expects extreme weather events to become more frequent as a consequence of climate change and hopes to contribute to further development of warning systems to increase the ability of society to act [2]. The aim is to assess to what extent machine learning algorithms for image processing and text mining can be employed to evaluate the accuracy of impact-based weather warnings, and further evaluate the added value of integrating AI-based information into the existing weather warning systems.

The objective of this thesis project was to implement methods to identify and visually explore extreme weather events, particularly floods, from Twitter data. To this end, the main tasks for the project were to implement text data mining methods for identification of relevant events from Twitter data and to create an accompanying visual interface for exploring the events. Hence, the project concerned a subfield of AI called Natural Language Processing (NLP) that seeks to program a computer to process and analyse natural language data. Further, it was focused within the multidisciplinary field of visual analytics referring to the science of analytical reasoning facilitated by interactive visual interfaces [64].

[2]https://liu.se/en/research/ai4climateadaptation

Chapter 2

Related Work

Studies have shown that social media can be used to develop detection and monitoring systems for disaster events. Present methodologies use different text data mining approaches for identification of relevant events from the data. There are several examples of systems using Twitter data for detecting high-impact disaster events. This includes Tweet4act that seeks to detect anything crisis-related [18] and Twicident that instead focuses solely on fires [1]. Many systems are also focused on earthquakes such as the Twitter Earthquake Detector [22] and the Ushahidi platform that was used specifically during the Haiti earthquake in 2010 to collect text-based geo-tagged reports from victims. The platform was used to organise rapid responses across agencies and points at locations where relief actions were needed [68]. The Toretter system developed by Sakaki et al. was able to detect earthquakes and announce it faster than the Japan Meteorological Agency [54]. The use of VGI and crowdsourcing has not only been observed for earthquake and fire events, but also for floods. The first system aimed specifically at helping victims during flood related disasters based on their Tweets was developed by Singh et al. [61]. Another example is the Global Flood Monitor developed by de Bruijn et al. [11] that detects flood events by continuously scraping and analysing Tweets in 11 languages based on a set of pre-defined key words. Other work include case studies that focus on mapping potential floods in specific areas of the world such as Jakarta [23], Great Britain [7] and Western Europe [24]. However, all with the goal of developing an approach that could potentially be expanded to other regions. The above mentioned systems all provide evidence for the relevance and importance of continuous research in using social media for detection of extreme weather events.

The systems were build using a variety of methods. Most of them were based on data retrieved from Twitter, as it is a platform for users to express immediate emotions on short and precise messages. Further, the public Twitter Streaming API makes it easier to access real-time data streams of the users' Tweets. As in most research based on data, the initial step has been data collection and pre-processing. The detection of potential events from the Twitter data has mainly been done through implementing one or more text classifiers using supervised machine learning. This calls for labelled data, wherefore labelling the Tweets as either relevant or non-relevant to the events in question has also been a necessary part of previous research. With the objective of improving the situational awareness during extreme weather events, a just as essential part has been to relate the Tweets to a geographic location for which several approaches have been used. The last step for most research has then included creating different visualisations of the relevant Tweets that could be used to explore at-risk areas.

2.1 Data Collection and Location Extraction

The current research mainly uses two different methods for retrieving data via the Twitter API. One method is keyword filtering that includes defining a set of keywords related to the content of interest so that only Tweets containing these words are extracted. Another approach is location filtering that instead extracts Tweets in pre-defined areas. Barker et al. [7] used location filtering with the focus of creating a national-scale social geodata pipeline for detecting flood events across Great Britain. This was done by obtaining a list of at-risk locations so that Tweets referenced with coordinates or places in that area could be extracted. The same approach was used by Feng et al. [24] that predefined a study area covering most of Western Europe for the Tweets to be referenced within. On the contrary, de Bruijn et al. [11] used keyword filtering by defining a list of flood-related words in 11 different languages to scrape Tweets at a global scale. However, only around one-eighth of the Tweets were found to be of relevance to an actual flood event.

With the use of keyword filtering, many Tweets risk being discarded as they cannot be related to a location. If the purpose is to improve situational awareness, the Tweets are then less relevant to use for event detection. This is oppositely not an issue when using location filtering, though this approach can instead result in many potentially relevant Tweets not being collected. An analysis by Middleton et. al. [42] has shown that only about 1% of Tweets actually contain a geotag during disaster events. Hence, to include more data, it is worth noting that the Tweet text potentially contains geographic references that can only be fully exploited if keyword filtering is used.

The process of extracting locations from text is referred to as geoparsing and has in general been shown to dramatically increase the number of localised Tweets [12]. However, using keyword filtering increases the demand of computer memory by collecting many irrelevant Tweets while geoparsing demands for more computational power. What filtering method to use therefore highly depends on the focus and scope of the research as well as the allocated resources. Part of the research by de Bruijn et al [12] merely focused on how to best extract the locations of Tweets and presented a geoparsing algorithm which relied on grouped geoparsing. This algorithm was applied to locate the keyword filtered Tweets used in the Global Flood Monitor by the same research team [11]. In similar efforts, Singh et al. [61] used a probabilistic approach to predict the user location of Tweets based on historical geotagged locations of the users through building a Markov chain model. The system achieved a promising location prediction accuracy of 87%.

In a paper by Middleton et al. [42], different methods for extracting locations from social media data were evaluated, which typically included combinations of geocoding, geoparsing and geotagging. As for the methods introduced, the process mainly starts with 1) toponym recognition followed by 2) toponym resolution. An approach to the first step has often involved Named Entity Recognition (NER), whereas the second step can be performed in many different ways and is often what poses the biggest challenge. In this project, it was considered important not to limit the amount of data just as it was not the focus to detect events in any specific area. Hence, this project was based on data mainly collected using keyword filtering, making it relevant to construct a location extraction algorithm inspired by the methods introduced above. This algorithm was aimed at enforcing a higher level of transparency than what to our knowledge has previously been the case. Specifically by enabling an easier differentiation of the way in which the location of a Tweet was extracted.

2.2 Text Classification

In related research, the detection of disaster events from textual data has typically been performed using supervised machine learning algorithms. These algorithms aim to find patterns in the Tweets such that those that are non-relevant for the event can be excluded. In the case of flood detection, this could be Tweets that contain key words such as 'flood' figuratively e.g. 'I will flood your feed with information' or Tweets that are simply geotagged in the pre-defined at-risk areas despite no mention of flooding.

For the algorithms to learn from the data, the collected Tweets must be labelled.

The labeling of training data is one of the most typical problems for supervised machine learning as large amounts of training data are required for the models to perform satisfactorily [24]. In the research by Barker [7], de Bruijn [11] and Lin [40], the collected Tweets were labelled manually resulting in a very time-consuming process and limited training data, all in amounts below 5000 samples. For Feng et al. [24], this was considered quite the disadvantage, hence it was avoided by instead introducing an automated labelling process. This included an initial keyword filtering of the individual geotagged Tweets followed by looking up the Tweets in historical weather records via a weather API. If the Tweet was posted in a period where the weather API also reported rainfalls, it was automatically labelled as relevant. This resulted in quite the increase in training samples with 72,938 Tweets. Automation of the labelling process was also one of the main purposes for Chowdhury et al. [18] using the Tweet4act system. In this research, an outlier detection system with the unsupervised clustering algorithm K-medoid was employed to verify whether Tweets were related to a crisis or not. This system performed with a precision above 90% for three separate use cases. Differently, in the research by Caragea et al. [16], a data set was downloaded from the CrisisLexT26[1] collection which contained Tweets that were already manually labelled by crowdsourced workers according to informativeness for six flood events [47]. Hence, this research leveraged other's work to get labelled Tweets and did not include considerations regarding this.

Besides the need for labelled data to train the algorithms, a necessary first step for NLP tasks is to obtain a numerical representation of the words in the textual data. For this purpose, different word embedding techniques have been used depending on the chosen classification algorithm. In the research by Feng et al. [24], several algorithms were deployed which included a convolutional neural network (CNN) as well as four classic approaches, specifically naive Bayes, logistic regression, random forest and support vector machine (SVM). All classic approaches were trained on data that was transformed using Term Frequency Inverse Document Frequency (TF-IDF) encoding, which mapped each word to a real number to reflect its importance in a collection of documents [24]. It was observed that expanding the TF-IDF encoding to use bigrams or trigrams could provide phrase information, though at the cost of a sparser TD-IDF matrix. However, the model still does not take into account sequential information and neither the semantic meaning [40].

[1] https://crisislex.org/data-collections.html#CrisisLexT26

For the CNN in the research by Feng et al. [24], the input was transformed to word embeddings using the Word2Vec model from the Gensim library. This model maps the words to vectors in a higher-dimensional feature space to capture semantic and syntactic relationships. The CNN was implemented using Tensorflow and had a rather simple architecture with one convolutional layer, one max-pooling layer and one output layer with two nodes representing the relevant and non-relevant class, respectively. The final predictions were generated by the softmax function. All algorithms performed with an accuracy between 71%-79% with naive Bayes performing the worst and the CNN performing the best. In the research by Caragea et al. [16], two approaches for identifying informative Tweets about flood events were implemented. Conclusively, a CNN trained on word embeddings was shown to outperform an SVM trained on TF-IDF features with an accuracy of 77.6% compared to 73.8%.

Barker et al. [7] also used the Gensim library, though instead for deploying a Doc2Vec model to represent each Tweet as a numerical vector. This is an extension of the Word2Vec model that generates vectors for entire documents instead of for single words. Next, for identifying the flood-relevant Tweets, a logistic regression model was trained using the scikit-learn library. The data set used for this research had an imbalance issue, wherefore the accuracy of 95% was misleading. This could also be seen by the lower performance scores in terms of 79% precision and 60% recall. For further work, it was suggested by Barker et al. [7] to use deep learning specifically a CNN based model.

Recent research has in general shown that using deep learning models instead of the classic algorithms can improve the accuracy of a text classifier by important percentages [43]. Especially, the CNN has recently achieved state-of-the-art results for sentence categorisation [36] despite the fact that language modelling in the field of deep learning is more typically related to recurrent neural networks (RNN) in the form of long short-term memory (LSTM). However, these type of networks are often used when it is relevant to learn long-term dependencies, that is to retain information from the beginning of a text sequence for longer periods of time. The CNN's are on the other hand designed to detect local patterns and key phrases, wherefore it has shown to perform better for tasks as sentiment classification [43] that is rather similar to the classification of a Tweet to be relevant or not to an event. The above research findings are reasons for why it was chosen to implement a CNN in this project rather than any other neural network for the text classification.

Another method gaining attention in the field of AI is transfer learning, which refers to a multi-step process with an initial pre-training step followed by fine-tuning on a new task using labelled data. A popular technique for transfer learning in NLP is bidirectional encoder representations from transformers (BERT) which in the research by de Bruijn et al. [11] was used to build the classifier for

detecting flood-relevant Tweets. Another technique specifically developed for text classification is universal language model fine-tuning (ULMFiT) proposed by Ruder and Howard [29]. The technique has been tested mainly for sentiment classification. To our knowledge the implementation of it for event detection is however limited, wherefore the technique becomes relevant to apply in this project.

2.3 Data Visualisation

An essential part of research within social media event detection has been the visual presentation to enable identification and monitoring of the events. Large amounts of complex data concludes nothing unless the users are able to make sense of it [27]. Visualisation has shown to have the potential of providing rapid insight and understandable assessments which can be communicated effectively for action [64]. Several visual analytics techniques have been provided by researchers to handle dynamic and ambiguous volumes of data by integrating human judgement, specifically with visual representations and interaction techniques in the analysis process [59, 34].

Barker et al. [7] proposed a geo-spatial dashboard for the Tweets detected as flood-relevant as a last step in their national-scale real-time Twitter data mining pipeline. Each Tweet was plotted with the interactive web mapping libraries Leaflet and Folium based on its geographic coordinates. This enabled navigation to examine both high-level patterns and low-level details in the data. Clickable markers were used to provide pop-up boxes of the Tweet when scrolled across, to enable details-on-demand as demonstrated in Figure 2.1.

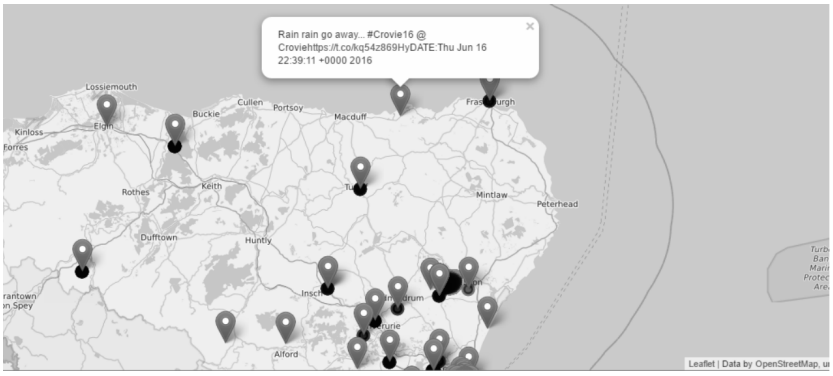


Figure 2.1: Dashboard prototype by Barker et al. with flood alerts in June 2016 created using the libraries Leaflet and Folium

An alternative interactive map visualisation was created using the business intelligence tool Tableau presented in Figure 2.2. The Tweets were here plotted as circles based on the area bounding box retrieved from the Twitter API. The hue indicates the Tweet density from that area, whereas the size indicates the radius of the relative area for that Tweet place. These variations of visual encoding facilitate the visual analysis for the user.

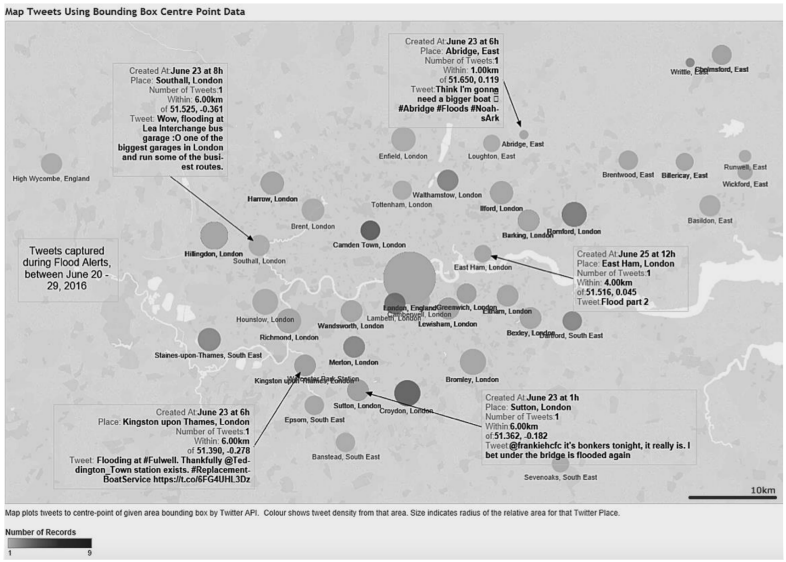


Figure 2.2: Alternative dashboard prototype by Barker et al. with flood alerts in June 2016 created using Tableau

In the research by Feng et al. [24], pluvial flood relevant information was visualised on a web map application as shown in Figure 2.3. Flood events were detected through spatio-temporal clustering and visualised as light blue circles. The circle radius is varied according to the data points of the clusters, the larger circles are thereby highlighting important areas to be inspected. Eyewitnesses of rainfall and flood events were retrieved from texts and photos and visualised as clustered point markers. Interactivity was integrated by pop-up windows at user locations and clicking events for the point markers providing detailed information of each Tweet. Hot spots were also detected using the geostatistics method Getis-Ord Gi* and visualised with a choropleth map. The coloring highlights places in which only few Tweets are normally sent but in which a large number of Tweets suddenly appears. This can be of great importance for the user to quickly determine potential flood events.

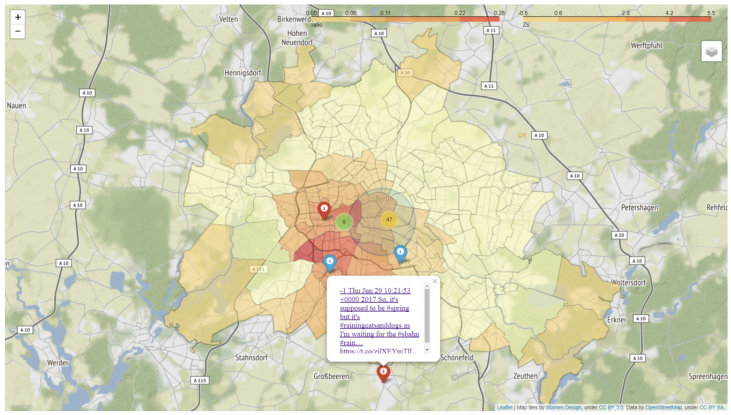


Figure 2.3: Web map application by Feng et al. with pluvial flood in Berlin on June 29, 2017

Various techniques have been used in the above mentioned visual applications to facilitate visual analysis and exploration. However, in terms of the visual design, a filtering option is missing in order to fulfill Shneiderman's basic principle of the visual information-seeking mantra; overview first, zoom and filter, then details-on-demand [59]. Instead of visualising the entire data set at once, different time slices are examined using a timeline bar in the Global Flood Monitor[2] by de Bruijn et al. [11] seen in Figure 2.4.

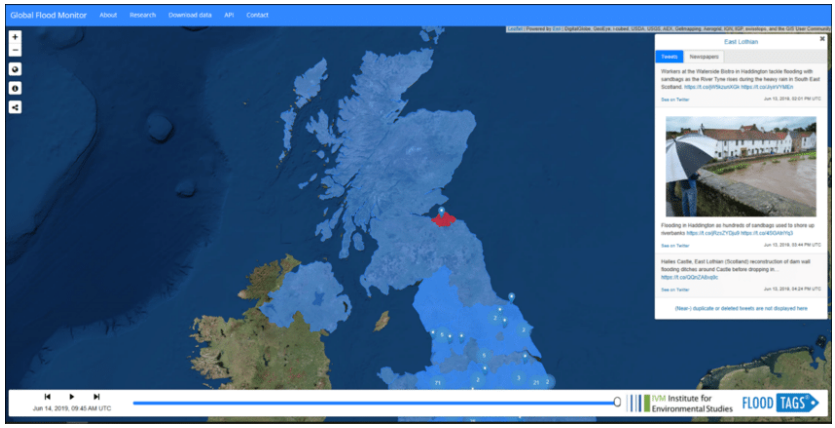


Figure 2.4: The Global Flood Monitor by de Bruijn et al. with floods in the UK on June 14, 2019 created using the Leaflet library

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

The map in Figure 2.4 was build using the Leaflet library and shows the temporal distribution of flood events using a timeline slider. The coloring of regions with enhanced flood-related Twitter activity allows the user to explore through spatially observing the most critical areas that might be worth zooming into. The Tweet locations are in this case also mapped with markers and the relevant Tweets are displayed when clicked on allowing for details-on-demand.

Another visual interface focused on exploration of large amounts of geo-located Tweets, not necessarily related to floods, is the OmniSci Tweetmap[3] build by Dong et al. [19] seen in Figure 2.5.

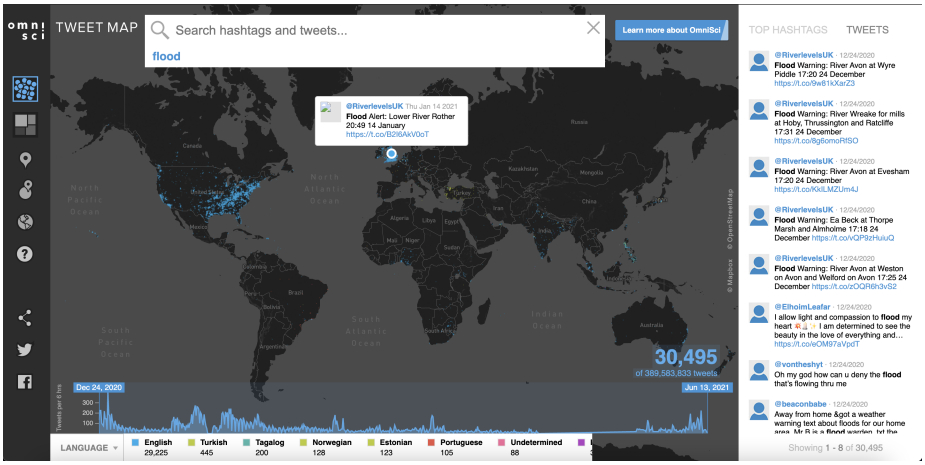


Figure 2.5: OmniSci Tweetmap by Dong et al. using the search word 'flood'

The interface in Figure 2.5 has incorporated various interactive dynamics for visual analysis. In terms of data and view specifications, it has different visual encodings that can be chosen; a point map with each Tweet represented and a choropleth map showing the number of Tweets within different countries. Additionally, it filters out the data both with the map and timeline slider to focus on Tweets within a selected area and period. The Tweets can also be filtered based on relevant data dimensions such as language, source, and specific hashtags or words used in the Tweets. It sorts items to expose patterns by showing the top hashtags used and derive values from the data e.g. by presenting the number of Tweets per six hours over a time period and per language. In terms of view manipulation, the interface enables navigation and selection of items to highlight, filter and manipulate for analysis. The multiple views are as opposed to in earlier mentioned visual applications organised in different

[3]https://www.omnisci.com/demos/tweetmap

workspaces and linked to enable multidimensional exploration that facilitates comparison [27].

As the OmniSci Tweetmap manages to incorporate many different and important interactive dynamics for visual analysis, it becomes relevant to use as inspiration to build the visual interface for this project. Instead of including all kinds of Tweets as the OmniSci Tweetmap currently does, the interface will be specifically designed to gain insight from Tweets related to flooding.

Chapter 3

Methods

In this section, the methods used for this project are described and motivated. A flow chart showing the project pipeline is presented in Figure 3.1. An enlarged version can be found in Appendix A.1. The different steps are elaborated in the subsequent sections including data collection and pre-processing, text classification, location extraction and data visualisation. All computations and analyses were performed using the programming language Python 3 with related libraries in Jupyter notebooks.

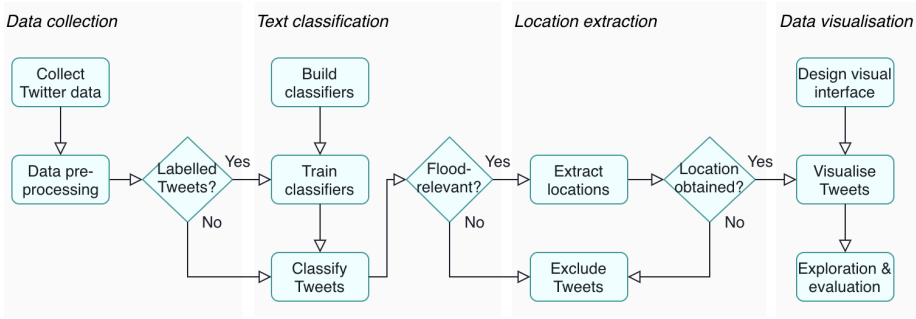


Figure 3.1: Project pipeline with the steps 1) data collection, 2) text classification, 3) location extraction and 4) data visualisation

3.1 Data Collection

Initially, two data sets were downloaded from the CrisisLexT6[1] collection and Harvard Dataverse[2], respectively. The first data set contained labelled Tweets and was used for training different text classification models. The other data set contained unlabelled Tweets and was subsequently employed as a use case for evaluation of the project pipeline. The data sets had limited information as they had been dehydrated such that each Tweet was only represented by its unique Tweet ID. This was done as the Twitter API's terms of service does not allow making large amounts of raw Twitter data available on the internet.

To enrich the data sets, the command line tool Twarc[3] was used to rehydrate the data sets based on the Tweet IDs. This tool uses the Twitter API to search through the current content on Twitter. It is thereby limited to collect the Tweets that still exist on the day of rehydration. As a result, two comprehensive data sets were obtained as JSON-objects with some root-level variables as well as child objects that in themselves consisted of variables. A set of the most relevant variables are shown in Table 3.1. The full list can be found at the Twitter Developer Platform[4].

Variable	Type	Description
created_at	Datetime	UTC time when the Tweet was created
id	Int64	Unique identifier for the Tweet
full_text	String	The actual text of the Tweet
source	String	Utility used to post the Tweet
user	User object	JSON-object with user information
geo	Coordinates object (latitude, longitude)	Geographic location of the Tweet as reported by user or client application
place	Place object	The place the Tweet is associated with
retweeted	Boolean	Indicates whether the Tweet is Retweeted
retweet_count	Int64	Number of times the Tweet is Retweeted
lang	String	Indicates machine-detected language of the Tweet text, e.g. 'en'

Table 3.1: Selected variables from the Tweet Data Dictionary

[1] https://crisislex.org/data-collections.html#CrisisLexT6
[2] https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/T3ZFMR
[3] https://scholarslab.github.io/learn-twarc/
[4] https://developer.twitter.com/en/docs/twitter-api/v1/data-dictionary/object-model/tweet

Some of the variables in the user and place objects are exemplified in the following.

```
"user":
{
    "id": 6253282,
    "name": "Dave Davidson",
    "screen_name": "TheDave",
    "location": "San Francisco, CA",
    "url": "https://developer.twitter.com",
    "description": "This is an example.",
    "verified": true,
    "followers_count": 6129794,
    "friends_count": 12,
    "listed_count": 12899,
    "favourites_count": 31,
    "statuses_count": 3658,
    "created_at": "Wed May 23 06:01:13 +0000 2007",
    "geo_enabled": false,
    "lang": "en"
    ...
 }
"place":
{
    "attributes":{},
    "bounding_box":
    {
        "coordinates":
        [[
           [-77.119759,38.791645],
           [-76.909393,38.791645],
           [-76.909393,38.995548],
           [-77.119759,38.995548]
        ]],
        "type":"Polygon"
    },
    "country":"United States",
    "country_code":"US",
    "full_name":"Washington, DC",
    "id":"01fbe706f872cb32",
    "name":"Washington",
    "place_type":"city"
}
```

From the user object, the 'user_name' and 'user_ location' were extracted and instead added as root level variables. The same applied for the place object where 'place_name', 'place_type' and 'place_bounding_box' were extracted. Together with the variable 'geo', these were used in the process of obtaining a geographic location for the Tweets, which is described in detail in Section 3.3. Further, a variable with the hashtags present in the Tweets was defined. This enabled a presentation of the most popular hashtags in the final visual interface.

Upon inspection of the data, it was found that the variables 'retweeted' and 'retweet_count' were not always reliable. This corresponds to the statement on the Twitter Data Platform saying that the 'retweet_count' may not reflect the

exact count[5]. Hence, it was chosen to redefine these variables such that they depended on the collected data. The 'retweet_count' was calculated by finding the number of duplicated Tweet texts if ignoring the initial retweet indicator 'RT'. The 'retweeted' variable was set to True if it was the first Tweet posted in a series of duplicate Tweets else it was set to False. To keep track of the related Tweets, the variable 'original_tweet_id' was added which contained the Tweet ID of the first Tweet in the series of duplicate Tweets. Lastly, the variable 'tweet_link' was added by knowing that the hyperlink to a Tweet always has the form of

https://twitter.com/{user_name}/status/{tweet_id}

The new and updated variables are presented in Table 3.2.

Variable	Type	Description
user_id	Int64	Unique identifier for user
user_name	String	Name chosen by user
user_location	String	Registered location by user
place_name	String	Name of place the Tweet is associated with
place_bounding_box	Coordinates in Polygon	Place the Tweet is associated with
retweeted	Boolean	Indicates whether the Tweet is a Retweet
retweet_count	Int64	Duplicate count of Tweet text
original_tweet_id	Int64	Unique identifier for first Tweet in series of Retweets
hashtags	List of strings	Hashtags parsed from Tweet
tweet_link	Hyperlink	Link to Tweet

Table 3.2: Variables added or updated in the data pre-processing

[5]https://developer.twitter.com/en/docs/twitter-api/v1/tweets/post-and-engage/api-reference/post-statuses-retweet-id

3.1.1 Labelled Data

The labelled data set was downloaded from the CrisisLexT6 collection and contained English Tweets across two large flood events: The Queensland floods in January and February in 2013 as well as the Alberta floods in June and July in 2013. The Tweets were collected using keyword filtering with keywords related to the events as well as location filtering by being geotagged with geographic coordinates inside the affected areas [47].

The keywords used in the collection of the Tweets in the labelled data set were:

1. Alberta floods: alberta flood, #abflood, alberta floods, #yycflood, #yycfloods, #yycflooding, canada flood, alberta flooding, canada flooding
2. Queensland floods: #qldflood, #bigwet, queensland flood, australia flood

The Tweets were labelled through the crowdsourcing platform Crowdflower[6] according to relatedness being either on-topic or off-topic. Spending unnecessary time on either manually labelling the Tweets or building an algorithm to automatically do so was thereby avoided. This was done similarly to the research by Caragea et al. [16] that however instead used the CrisisLexT26 collection. The T6 collection was chosen for this project as it contained Tweets labelled by relatedness rather than informativeness that was the case for the T26 collection.

The relevant class included Tweets found to be directly and indirectly related. The data set included one Tweet per row with following variables: 'tweet_id', 'tweet_text' and 'tweet_label'. By using Twarc for rehydration, the final data set contained 12,770 Tweets of which 57% were labelled as relevant for the flood event in question. This amount of data was chosen due recommendations from a survey of similar tasks [31] in which it was stated that training data in the thousands seemed fair.

A few examples of Tweets labelled as relevant to the flood events are shown in Figure 3.2 and include words such as warnings and help.

[6] http://www.crowdflower.com/

Figure 3.2: Examples of Tweets from the CrisisLexT6 collection labelled as relevant

Some examples of Tweets labelled as non-relevant are shown in Figure 3.3. It can be seen that the Tweets collected based on keywords contain sequences that can be harder to evaluate the relevance of compared to the location filtered Tweets as they contain no flood-related words.

(a) Keyword filtered Tweets labelled as non-relevant

(b) Location filtered Tweets labelled as non-relevant

Figure 3.3: Examples of Tweets from the CrisisLexT6 collection labelled as non-relevant

3.1.2 Unlabelled Data

The unlabelled data set downloaded from the Harvard Dataverse was originally used and collected by de Bruijn et al. [11] to build the Global Flood Monitor. The purpose of downloading this data set was to enable an evaluation of the steps in the project pipeline in Figure 3.1. The data set contained the Tweet IDs of 87,641,357 Tweets in 11 languages posted between July 29, 2014 and November 20, 2018. These were collected using keyword filtering based on a list of keywords related to floods in each of the languages. In English, these words were:

Keywords: flood, floods, flooding, flooded, inundation, inundations, inundated

The list of Tweet IDs was split in subsets to make the rehydration process using Twarc easier and more efficient. As the labelled data set used for training the classifiers only contained English Tweets, the non-English Tweets were excluded. This choice was was made as English is still the most used language on Twitter [67]. Further, it was chosen to only use a subset of the English Tweets in the years 2016, 2017 and 2018 for evaluation. The final data set then consisted of 251,018 Tweets. More data would not provide grounds for better evaluation but just unnecessarily increase the time spend on classifying and locating the Tweets.

3.2 Text Classification

An important step for performing any natural language task is the pre-processing of the textual input. Text is in itself rather unstructured and hard for a computer to handle as computers are designed to understand numbers rather than words. Hence, the main purpose of the text pre-processing is to obtain a numerical representation of the words in order for the computer to comprehend it. This can be done using different methods depending on the classification algorithm. The most basic methods represent the words as single numbers or vectors with no semantic context. Examples are one-hot-encoding and term frequency inverse document frequency (TF-IDF). The newer methods instead seek to capture the semantics to enable the computer to better understand the connection between different words in a higher dimensional space. This is typically done either with the Word2Vec model or through an embedding layer in a neural network. If this step is not performed optimally, it can have a significant impact on the final performance of a model [32].

The task for this project was a binary classification problem as the Tweets should be classified as relevant or not to flood events. The suitable approach was therefore supervised machine learning as the models should predict the class of the Tweets using the labelled Tweets as the correct outputs to learn from. The basic supervised algorithm can be defined as

$$Y = f(x) \tag{3.1}$$

where Y is the predicted outputs determined by a mapping function that assigns a class to an input value x. During training, the algorithm will search for patterns in the data that correlate with the desired outputs and make adjustments in the model weights to adapt. The weights are adjusted in the training in order to minimise the loss of the model. The binary cross-entropy is used, as it is the most common loss function used for binary classification problems. It is defined as:

$$J = -\frac{1}{N}\sum_{i}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{3.2}$$

where N is the size of the training set. It compares each of the predicted probabilities $\hat{y_i}$ to the actual class output y of either 0 or 1. Then it calculates the score that penalises the probabilities based on the distance from the actual value [55].

Once the model is trained, it is used to make predictions about unseen or future data. A precondition for good performance is to generalise well from the training data to the unseen data in the same domain, which means that it is important to focus on avoiding overfitting during training as most models are prone to this. If avoided appropriately, the models should be able to make accurate predictions and identify the Tweets that are flood-relevant amongst future unseen Tweets.

The first step towards avoiding overfitting and evaluate the generalisation power of the models was to separate the data into a training and test set. The test set was put aside during training such that it could be used to assess the performance of the models on unseen data. Prior to this, the data was filtered to only contain the unique Tweet texts. The duplicates would not improve the model performance, but rather extend the training time. Therefore, the Retweets which constituted 16.5% of the data were excluded. This balanced out the data set with 51.5% of the Tweets in the relevant class reduced from the initial 57.0%. The remaining data was then split in a 80/20 training/test split according to the Pareto Principle [14]. The total number of Tweets used for building the classifiers were 10,569, of which 8,455 were used for training and 2,114 were used for testing.

A simple model is typically implemented as a baseline for comparison to more advanced models. This can show whether adding additional complexity also adds more value. In general, one should follow the principle of parsimony saying that the simplest of competing explanations is the most likely to be correct [56], hence if the performance of a more complex model does not improve significantly, the simpler model could beneficially be preferred.

For this project, a risk of overfitting the models is posed by the fact that the labelled data contained Tweets related to and collected for two specific events. To overcome potential bias from the specific keywords used for the data collection, three different model variants were created. The first variant referred to as 'Original Tweets' included no transformation of the Tweet text. In the two other variants, the Tweet text was transformed with the aim of increasing the classifiers' ability to generalise to unseen Tweets. In a variant referred to as 'Remove keywords', all occurrences of keywords specifically related to the flood events in Alberta and Queensland were removed from the text to avoid overfitting. Specifically, the following keywords were replaced with an empty string.

> Replaced keywords: yyc, ab, edmonton, calgary, alberta, canada, nsw, qld, brisbane, queensland, australia, bigwet

In the last variant referred to as 'Replace places', all places that were mentioned more than 0.5% of the size of the data set, for this case 55 times, were replaced with the word 'place'. This threshold was chosen arbitrarily by inspecting the number of places found in the Tweet texts as well as the intuition that if a place was mentioned less than this, it would not pose a significant risk for creating bias in the models. The idea behind this approach was that the mentioning of a place could be of high importance in detecting relevant events, though having the specific place names could risk creating such bias towards certain locations [13]. A NER model from spaCy[7] with the English pipeline 'en_core_web_lg' was used to extract all entities within the Tweet with geographic references. The same approach was used to extract locations from the Tweets for mapping purposes which is elaborated in Section 3.3. However, it is worth noticing that this approach has some disadvantages as NER is prone to errors and requires quite some computation time.

[7] https://spacy.io/usage/spacy-101

3.2.1 Classic Algorithms

The classic supervised machine learning algorithms employed as baseline models were implemented using the open source machine learning library scikit-learn[8]. This library enables pre-processing of data and features many different machine learning algorithms, making it easy to build advanced models in very few steps. The Tweet text was first transformed to a numerical representation using TF-IDF. Then, logistic regression (LR) and random forest (RF) were used to train text classifiers as they are some of the most frequently used algorithms for binary classification [24]. This gave an indication of the classification performance using a traditional machine learning approach.

3.2.1.1 Term Frequency Inverse Document Frequency

In-build scikit-learn functions were used to do parts of the text pre-processing. This involved converting the collection of Tweets to a matrix representation using TF-IDF encoding. TF-IDF is a numerical statistic that reflects how important a word is to a document in a collection of documents. Hence, the statistic gives higher scores to words which are frequent in a document but rare overall in the collection of documents. This was specifically calculated using the following equation for word i in Tweet j.

$$\mathbf{X} = tfidf_{i,j} = tf_{i,j} \cdot log(\frac{1+N}{1+df(i)}) \tag{3.3}$$

Here N is the total number of Tweets, $tf_{i,j}$ is the occurrences of word i in Tweet j and lastly $df(i)$ is the number of Tweets containing word i. The input can then be represented as an M-by-N matrix $\mathbf{X}$ where each row represents a word i in the vocabulary of size M and each column represents a Tweet j.

Prior to this conversion, some initial text pre-processing steps made were tokenisation, normalisation and removal of noise. The first step involved splitting the Tweet text into single words by whitespace as well as removing all special characters and punctuation. This further included removing numbers, URLs, @mentions and stopwords to keep the Tweets simple. Stopwords are the most common words in a language, such as article, pronouns or prepositions. Lastly, the pre-processing included lemmatisation, which meant grouping together the different forms of a word so they could be analysed as a single word. It was preferred over stemming as it relies on a lexical knowledge base to link words with similar meaning to one word, instead of just cutting off the end or the beginning of the word. For example, both the words 'meanness' and 'meaning'

[8] https://scikit-learn.org/

are stemmed to 'mean', creating a false equivalence, which is avoided with lemmatisation where these words are instead kept as they are. The pre-processing resulted in defining a new variable called 'tokens' that included a list of strings for each Tweet. The scikit-learn class 'feature_extraction.text.TfidfVectorizer' was used to create the TF-IDF matrix based on the vocabulary of the final tokens in this new variable.

3.2.1.2 Logistic Regression

Logistic regression is a simple but powerful classification algorithm that resembles a linear function. In this case, weights are given each word in the vocabulary of the Tweets indicating how the word relates to the Tweet being either relevant or not to a flood event. The weighted combination of the input features are then passed through a logistic function:

$$\sigma(z) = \frac{1}{1 + \exp(-z)} \tag{3.4}$$

In the specific case of logistic regression $z = \beta_0 + \beta_1 \mathbf{X}$ that corresponds to a linear function where β_0 is the intercept, β_1 is a vector of weights and $\mathbf{X}$ is the input features represented as a TF-IDF matrix (3.3). The logistic function transforms an input to a number between 0 and 1, which is then the probability of the input belonging to class 1. If the probability is above the threshold of 0.5, the input, in this case a Tweet, is classified as relevant to a flood event, else it is classified as non-relevant. The scikit-learn class 'linear_model.LogisticRegression' was used to build the classifier. The logistic function is also referred to as the Sigmoid function, that is further introduced in Section 3.2.2 on deep learning where it is used as an activation function.

3.2.1.3 Random Forest

Random forest is an ensemble learning method that is suitable when having high-dimensional noisy data as is the case in text classification [26]. Random forest fits a set of decision tree classifiers each trained with random subsets of the features, which in this case are the words in the vocabulary. A decision tree consists of a series of true/false questions regarding the data that will eventually end in a predicted class. The internal nodes represent the questions based on which the tree splits into branches. The end of a branch that does not split further is called a leaf node and represents the final decision for the inputs. The random forest uses averaging over the individual trees to improve the its

accuracy and control overfitting. This works well as the ensemble predictions are more accurate than any of the individual predictions.

Different parameters can be chosen for random forest, overall how many individual decision trees the forest should consist of. For the individual trees, a maximum depth can be specified which refers to the the length of the longest path from a root to a leaf node. The split of each branch is chosen such that it minimises the loss and is evaluated using the Gini criterion. This criterion varies between 0 and 1, where 0 indicates a completely pure classification. This means that all inputs in a node are linked with the same class, hence this becomes a leaf node. An example of the structure is seen in Figure 3.4. The scikit-learn class 'ensemble.RandomForestClassifier' was used to build the classifier with the baseline model consisting of $n = 100$ decision trees each with a maximum depth of 11.

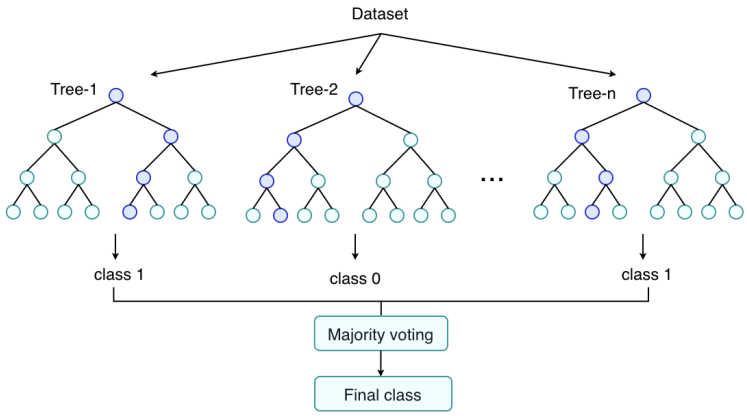


Figure 3.4: Example of the structure for a random forest classifier for a binary classification problem with n decision trees

3.2.2 Deep Learning

Deep learning has increasingly been used to improve the accuracy of text classifiers and has in general been growing in popularity within NLP [43]. It is a subfield of machine learning concerning multi-layered architectures of artificial neural networks that attempt to model the workings of the human brain. A network consists of input layers that take the raw data as input, then hidden layers that take the input from one layer and passes it to the next, and lastly an output layer that yields the prediction of the input. This can result in a network being very complex with many layers and hyperparameters that are not always

easy to define. However, the best performance is generally found when keeping the networks rather simple and thus creating a better understanding of it [43]. The training of a neural network involves using an optimisation algorithm to find a set of weights w to best map the inputs to the correct outputs. This can be posed as a non-convex optimization problem

$$\min_{w} \frac{1}{N} \sum_{i=1}^{N} J_i(w) \tag{3.5}$$

where N is the size of the training set. To minimise the loss function J_i, the weights w are updated during training by using backpropagation which employs a gradient-based optimisation algorithm [41]. Especially for NLP tasks, there is a tendency to prefer adaptive learning rate methods such as Adam, which is an extension to the classical Stochastic Gradient Descent (SGD) [37]. For the network trained in this project, the Adam optimiser was used as it has been shown to achieve good results fast compared to other optimisation methods [9]. The loss function J_i minimized is the binary cross-entropy (3.2).

The output from one hidden layer of a neural network to the next is often transformed using an activation function. The hidden layers typically use the same activation function, whereas the output layer will use a different function that depends on the specific task. The sigmoid function is typically used for binary classification to obtain a probability of the input belonging to class 1 as for the logistic regression model. It is defined as:

$$\text{Sigmoid}(z_i) = \frac{1}{1 + \exp(z_i)} \tag{3.6}$$

Another choice for the output layer could be the softmax function that instead of only outputting one probability as the sigmoid function, returns the probability of an input belonging to each class in the data. This is preferred in multi-class classification as it makes the outputs sum to 1. It is defined as:

$$\text{Softmax}(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \tag{3.7}$$

For the hidden layers, one of the most common activation functions is the rectified linear unit (ReLU) activation. This function transforms any negative input value to 0, and otherwise it returns the input value. It is defined as:

$$\text{ReLU}(z_i) = \max(0, z_i) \tag{3.8}$$

It applies for all above functions that z_i represents the values from the neurons of the input layer to the function. The sigmoid, softmax and ReLU activation functions are plotted in Figure 3.5 to better see their behaviour.

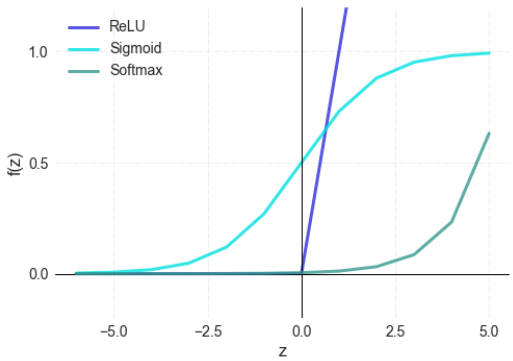


Figure 3.5: The Sigmoid, Softmax and ReLU activation functions

Besides choosing the correct activation functions, it is also important to choose the right hyperparameters for the network. One parameter is the number of epochs used for training the neural network, which defines how many times the learning algorithm will work through the training data. This corresponds to one forward pass and one backward pass.

Another parameter is the batch size, which is the number of samples to work through before updating the model parameters. A typical batch consists of either 32, 64 or 128 training samples. Hence, the training data is split in batches of the given size and when all batches have been passed through the model, one epoch is completed. After each epoch the performance of the model is evaluated by holding out a part of the training data as a validation data set, for this case 20%, and then comparing the training and validation losses each epoch. It is optimal to have the training loss equal the validation loss, whereas a training loss that becomes too low compared to the validation loss indicates that the model is overfitting the data. Our model was initially trained over 10 epochs and with a batch size of 64 samples. Early stopping was used during the training, meaning that it was stopped after a suitable number of epochs such that the training loss best equaled the validation loss.

Different regularisation techniques can be used to increase the generalisation power of the network with the purpose of reducing interdependent learning amongst its neurons. In the present network, L2 regularisation was introduced by adding a penalty term to the binary cross-entropy that the network minimises. Further, a drop-out layer was added before the output layer which works by reducing the number of nodes in the network. Specifically, individual nodes are either dropped out of the network with probability $1 - p$ or kept with probability p, where $p = 0.1$ was used for the initial network.

3.2.2.1 Convolutional Neural Network

There exist various kinds of neural networks that perform differently depending on the task. A CNN was used in this project for detecting relevant Tweets despite the fact that this is typically used for image tasks. However, the intuition is that the appearance of certain bi-grams, tri-grams and n-grams within the Tweet texts will be a good indication of whether the Tweet is relevant for the flooding event. Using a CNN for text classification includes the following main operations; 1) Embedding, 2) Convolution, 3) Non-linearity using a ReLU function, 4) Pooling, 5) Classification through fully connected layers.

In general, a CNN is suitable when the input data is high-dimensional as its layers naturally decrease the data dimension. This is therefore suitable for large amount of text data that is represented in a matrix output from the initial embedding layer. The embedding layer is used in NLP tasks to learn word embeddings that are tailored directly to the training data set. This requires an initial transformation of the text e.g. with TF-IDF as introduced in Section 3.2.1.1. However, it has become quite typical to leverage word embeddings learned elsewhere in this step with either the Word2vec or GloVe technique as they are trained on large amounts of data [36]. These methods transform the text input to a numerical representation by creating a vector representation of each word in a high-dimensional space, where words with similar meanings have similar representations and opposite. This aims at improving the computer's ability to learn the semantics of sentences and words [32]. All words present in the Tweets make up the vocabulary of the model, hence the input is the tokenised Tweets that are converted to matrices of which the rows are word vector representations of each token. For the embeddings, the Gensim module was used to load a Word2Vec model pre-trained on Google News. This includes word vectors for a vocabulary of 3 million words and phrases trained on roughly 100 billion words. Each word vector in this model has 300 features [36]. This means that the learned word weights in the embedding layer were not updated during training but remained static.

To present these embeddings visually, a nonlinear dimensionality reduction technique was used. This is called t-distributed stochastic neighbor embedding (t-SNE) and is well suited for embedding high dimension data into lower dimensional data for data visualisation. In Figure 3.6, the 2D word embeddings of the 10 most similar words to the randomly selected words, weather, awesome, thunderstorm and week, are visualised. An enlarged version can be seen in Appendix A.2.

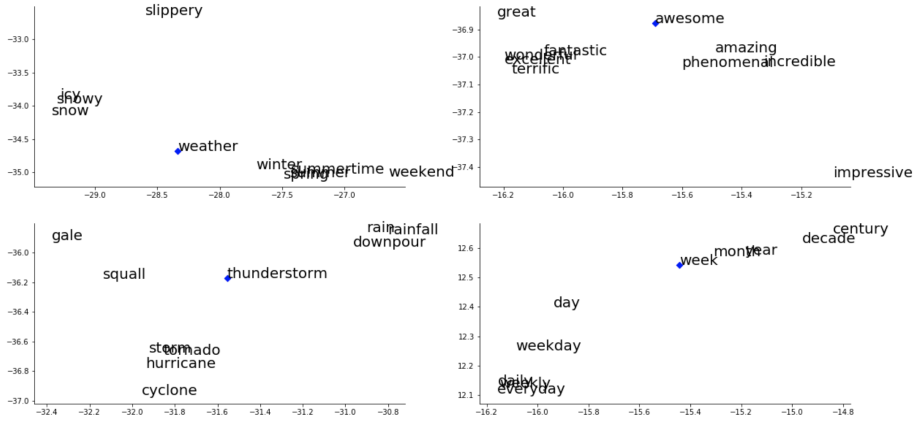


Figure 3.6: Word embedding visualisations using t-SNE on the 10 most similar words to the four selected words, weather, awesome, thunderstorm and week, that are marked with a blue diamond

In Figure 3.7, the selected words are shown in the full embedding space to better see their mutual relation. As expected, the words, weather and thunderstorm, are placed closer to each other than the other words as the meaning of these words are closely related.

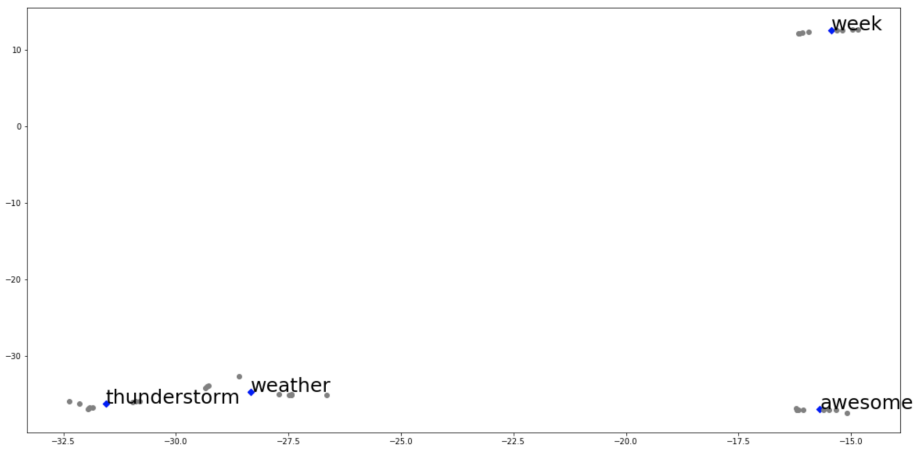


Figure 3.7: The relations of the four selected words, weather, awesome, thunderstorm and week, in the full word embedding space

By performing convolution and pooling during the training of the network, the neurons in the hidden layers seek to learn possible abstract representations over the input. Hence, these layers decrease the dimension of the data. In traditional image tasks, the convolutional layer finds spatial patterns in an image by sliding a small kernel window over the image, pixel-by-pixel. When a kernel slides over an image, the kernel weights are multiplied by the pixel value in the image underneath. Then all the multiplied values are summed up to an output, filtered pixel value. In the case of text classification, the convolutional kernel will slide over embeddings for multiple words (1D), rather than pixel areas in an image (2D). The width of the kernel corresponds to the length of the embedding and the height corresponds to the number of words that is considered at once. For the initial CNN in this project, an embedding with length of 300 and 5 words, or 5-grams, are considered at once.

A small example of a convolution with a kernel that is 2×3 is seen in Figure 3.8. A pair of filtered word embeddings are created where each word is encoded as an embedding with 3 values. The kernel weights and embedding values are multiplied in pairs and summed to obtain a single output value.

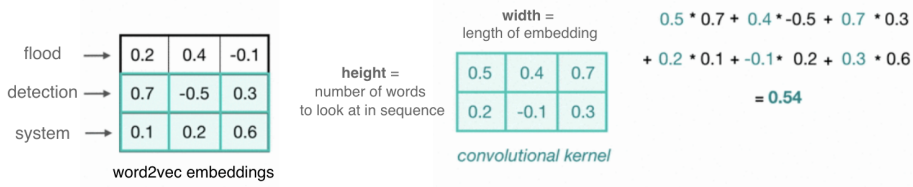


Figure 3.8: Example of convolution over word sequences in the CNN with a kernel of size 2×3.

In this way, each kernel will look at a word and its surrounding word(s) and output a value that captures something about that phrase. This can be referred to as window-based feature extraction, where the features are patterns in the word sequences, such as sentiments or grammatical functions. Just as similar words produced similar embeddings, the convolutional kernel will produce similar output values for different sets of similar words. Figure 3.9 shows an example of how a 1D convolution is applied to a sequence of word embeddings, producing a feature vector as output. Each feature vector is transformed using the ReLU activation function to discard negative values. Next, the pooling layer identifies the most important features by using a maxpooling operation that keeps only the maximum value of each feature vector.

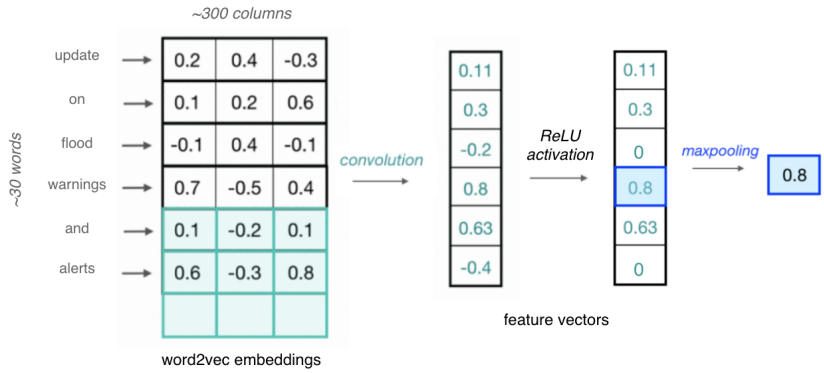


Figure 3.9: Example of 1D convolution applied to word embeddings producing a feature vector. The vector is transformed using a ReLU function, wherafter a maxpooling layer then takes only the maximum value of the vector.

In a classification task the output layer will simply use either the sigmoid or softmax function to transform the inputs to probabilities. For the binary classification problem a good choice is to only have one neuron in the output layer transformed by the sigmoid function as this will update faster than if using the softmax function for two output neurons. The same threshold of 0.5 is used as in the logistic regression model to decide whether a Tweet is relevant to a flood event or not. The CNN was developed and trained using Keras, the high-level API of the open source library TensorFlow[9]. The initial architecture used for the CNN is presented in Figure 3.10.

[9]https://www.tensorflow.org/guide/keras/sequential_model

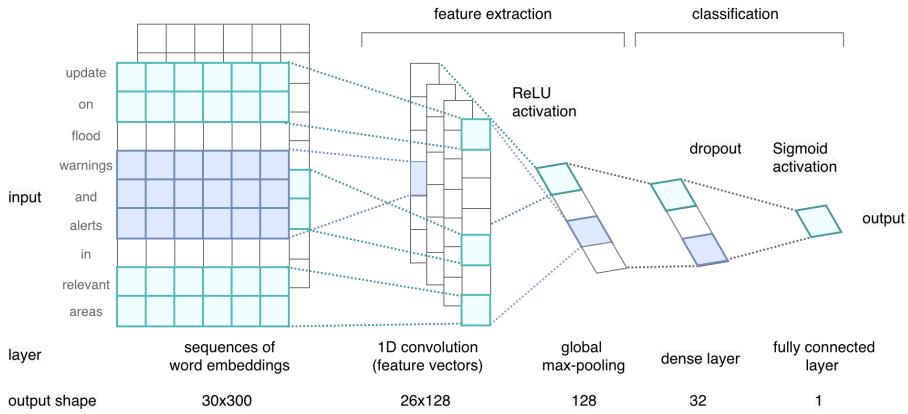


Figure 3.10: Final architecture of the CNN with an embedding layer, one convolutional layer using the ReLU function, one global max-pooling layer, a dense layer and an output layer using the sigmoid function.

3.2.3 Transfer Learning

A growing area within machine learning is transfer learning, which has recently improved the state-of-the-art for a variety of NLP tasks [49]. This method involves using a pre-trained model on a new problem in order to take full advantage of the work and knowledge gained from a previous task to improve performance for a new task. For the CNN introduced in Section 3.2.2.1, this concept was also utilised by using pre-trained word embeddings. However, it was only in the embedding layer that the knowledge from these were present, wherefore the entire network would still need to be defined and trained from scratch [53]. Instead, the benefits of transfer learning is that whole models made by experts can be applied. One thereby avoids concentrating too much on the choice of layers and parameters. Most importantly, since the models are so carefully designed, a better performance can be obtained. One should however keep in mind that a consequence of using transfer learning can be the lack of understanding and control of the model choice and structure.

3.2.3.1 Universal Language Model Fine-tuning

An example of transfer learning is the Universal Language Model Fine-tuning (ULMFiT) technique developed by Howard and Ruder in 2019 [29]. ULMFiT is claimed to be a good choice for smaller data sets as in our case [29]. The ULMFiT model was trained with modules from the fastai[10] library that is built on top of PyTorch[11]. The library was developed by the Fast.ai research group of which the author of the paper, Jeremy Howard, is a founder. The library makes this transfer learning technique affordable to deploy even for those with less deep learning experience.

ULMFiT involves a 3-layer ASGD Weight-Dropped LSTM. This is a popular type of RNN that uses LSTM cells to learn long-term dependencies, which is difficult for standard RNN's due to the vanishing gradient issue related to back-propagation and gradient descent over multiple time steps. The network employs various regularisation techniques including DropConnect which randomly drops the weights rather than the activations with probability $1 - p$. Moreover, it uses the optimisation algorithm Non-Monotonically Triggered Averaged SGD (NT-ASGD), which returns an average of last iterations of weights [41].

ULMFiT has three major steps 1) pre-training of a general language model, 2) fine-tuning the language model on a target task, and 3) fine-tuning the classifier on the target task. The model architecture and steps are shown in Figure 3.11 as presented by Ruder and Howard in their paper [29].

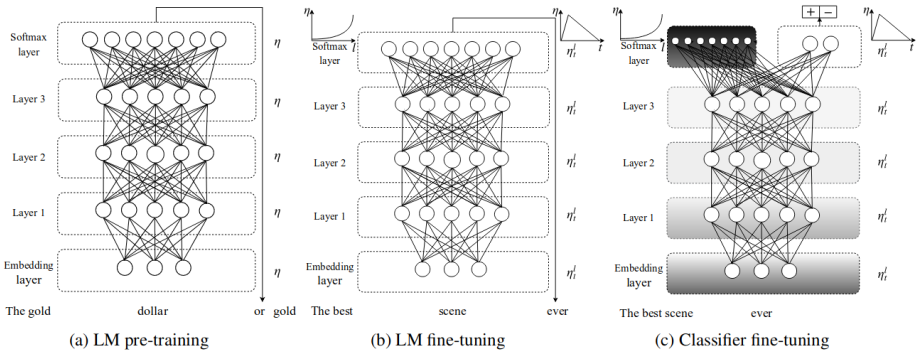


Figure 3.11: The ULMFiT structure as presented in the paper by Howard and Ruder [29] with the three steps 1) language model pre-training, 2) language model fine-tuning and 3) classifier fine-tuning

[10] https://docs.fast.ai/
[11] https://pytorch.org/

The first step includes pre-training of a language model which is a model that learns to predict the most probable next word in a sentence. For such model to truly learn semantic and syntactic relationships, this training must be done on large amounts of data. The ULMFiT model is trained on the WikiText-103 data set[12] that consists of a pre-processed subset of 103 million tokens extracted from Wikipedia. This data is unlabelled which means that this training is unsupervised. The ULMFiT technique is therefore more precisely termed as *semi*-supervised learning [49]. The WikiText-103 data set retains the original case, punctuation and numbers as it is wanted for the language model to capture some longer-term dependencies from relatively complex sentences. This also meant that no initial pre-processing of the Tweets in our data set was needed except removing URLs and @mentions. The 'TextLMDataBunch' class from the fastai library was used to do the final transformation of the Tweets. The initial pre-training step is the most computationally expensive, though it only needs to be performed once and will make the next steps converge faster.

In the next step, the language model is fine-tuned for the target task meaning that it needs to adapt to the words in the target data. Since the three hidden layers capture different kinds of information, it was shown that using discriminative fine-tuning resulted in a good performance. This means that a different learning rate is used for each layer. Additionally, the rates are not kept constant throughout the fine-tuning process and are increased linearly with a steep slope for the first iterations, whereas they are decreased linearly with a gradual slope for the remaining iterations. This is referred to as slanted triangular learning rates.

For the last step, where the target classifier is fine-tuned, gradual unfreezing is used. This means that not all layers are fine-tuned simultaneously as this could result in catastrophic forgetting, such that the model looses all prior knowledge. The layers are gradually unfrozen starting from the layer closest to the output. One by one, the layers are unfrozen and fine-tuned for one epoch. This is repeated till convergence. For the output layer the Softmax activation function is used such that the model is more flexible in terms of the number of output classes.

[12]https://blog.einstein.ai/the-wikitext-long-term-dependency-language-modeling-dataset/

3.2.4 Performance Measures

The machine learning algorithms are evaluated using different types of performance metrics. The performance of classification models are typically described through a confusion matrix as presented in Figure 3.12, with the associated terms next to it.

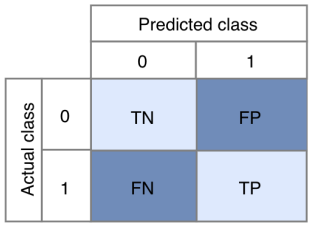


Figure 3.12: Confusion matrix

- True Negatives (TN): actual class 1 predicted as class 0
- False Positives (FP): actual class 0 predicted as class 1
- False Negatives (FN): actual class 0 predicted as class 0
- True Positives (TP): actual class 1 predicted as class 1

For the binary classification problem in this project, the unseen test data of Tweets can either be classified as relevant (class 1) or non-relevant (class 0) to flood events. Ideally the classifiers should have only true positives and true negatives to be 100% accurate, though that is rarely the case. Many performance measures are based on the numbers in the confusion matrix [58], some of which are introduced in the following.

Accuracy is a frequently used measure which answers the question of how often the classifier predicts correctly. It is important to note that accuracy is only a good measure when the target classes in the data are nearly balanced. This is thereby a reliable measure in this case as the data set is fairly balanced with 51% of the Tweets belonging to the relevant class. The metric is given by:

$$\text{Accuracy} = \frac{TN + TP}{TN + FN + TP + FP} \tag{3.9}$$

Another performance metric is precision, which represents how often the classifier is correct when predicting a positive result. In other words how many of the Tweets classified as relevant are actually relevant. In general, precision is mostly used if seeking to minimise the number of false positives.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{3.10}$$

Next, recall is a measure defined as the ratio of accurately predicted positive values to all predicted positive values in the data. Oppositely to the precision, this metric is important when the goal is to limit the number of false negatives.

This is especially important if classifying whether a person has an illness or not as a model with low recall would risk not detecting this.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3.11}$$

A single metric that represents both precision and recall is the F1 score as it is defined as the harmonic mean of the two, given by:

$$\text{F1 score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.12}$$

3.3 Location Extraction

An important step in the pipeline for enabling the exploration of potential flood events was to obtain a location for the Tweets. As the data sets were mainly collected using keyword filtering, only a limited number of the Tweets were geotagged. In order to include as many Tweets as possible in the visual interface, it was therefore desirable to construct an algorithm to extract anything related to a location from the Tweets. The flow chart in Figure 3.13 illustrates the steps in the location extraction algorithm defined specifically for this project. The choices made in the different steps are described subsequently.

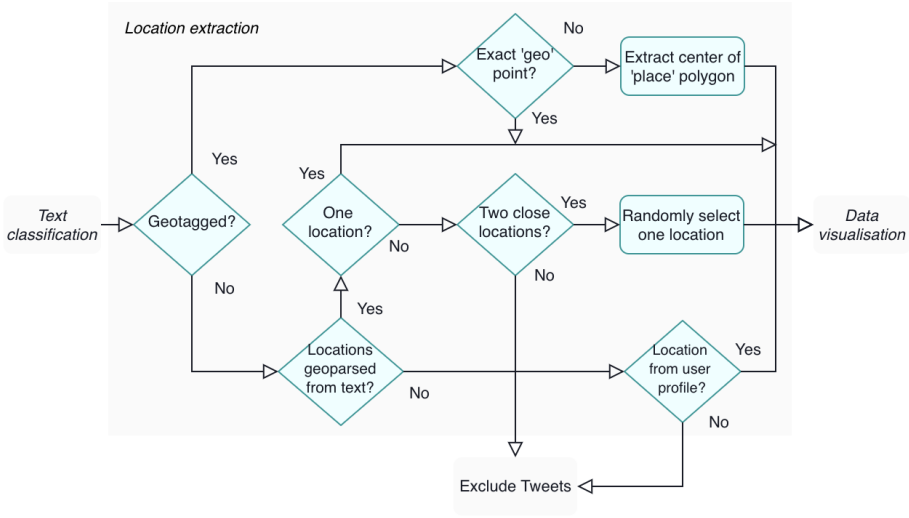


Figure 3.13: Flow chart for the location extraction algorithm used to locate the Tweets in the four levels 1) geotagged coordinates, 2) geotagged place, 3) geoparsed from Tweet and 4) registered user location

Some Tweets were geotagged with the exact coordinates in the variable 'geo' from Table 3.1. This approach was referred to as 'Geotagged coordinates' and was considered the most trustworthy way of obtaining a location. If the variable 'geo' was not available, the area from the variable 'place_bounding_box' was used instead. The exact coordinates were obtained as the center of the area and referred to as the 'Geotagged place'. The area could be of different sizes, either as a 'country', 'city', 'admin', 'neighborhood' or 'poi', making this location type less precise. This is also due to the very definition of this variable from

the Twitter Developer Platform stating that the Tweet is associated but not necessarily originating from this place. Still, 'Geotagged place' was considered the second most reliable approach to obtain a location.

If the Tweets did not contain geotagged information, textual data could instead be used to relate the Tweets to a location. This resulted in a third approach using geoparsing of the Tweets' text. This was done in two steps 1) through toponym recognition and 2) through toponym resolution. For the first step, the potential location candidates were extracted using spaCy for NER to find all entities within the Tweet with geographic references. The NER relied partly on the fact that proper nouns are always capitalised, wherefore the Tweets were kept in their original case and only URLs and @mention were removed, just as for ULMFiT described in Section 3.2.3.1.

The second step of the geoparsing was more challenging and demanded for a few decisions to be made. Prior to this step, the potential locations were geolocated with coordinates with the approach elaborated later in this section. If only one location was recognised in the text, no resolution was needed, hence the Tweet was kept and related to this location. Though, if two or more locations were found by NER, the spatial distance amongst them were found. Hereafter only the two locations with the shortest distance between them were kept for each Tweet. The last and most challenging part of the resolution, consisted of choosing between these two locations. Per intuition, if the distance between two potential locations was longer than d km, the location for the Tweet was too ambiguous, hence no location could be related to the Tweet. If the distance was however shorter than d km, one of the two locations was randomly chosen to be related to the Tweet. A threshold of $d = 1500$ km was chosen as it seemed an appropriate distance for two locations to both describe the same event. It was initially decided based on the fact that the distance from east to west Queensland is approx. 1500 km. The locations extracted from the Tweet text was referred to as 'Geoparsed from Tweet'.

If the geoparsing was not successful, the last approach included utilising that many Twitter users register their origin in their profiles given in the variable 'user_location'. Since this location might not be related to the specific Tweet, it should be used with caution for detecting events in the relevant areas. Besides, this variable can often contain noisy and redundant data. Some users had for example registered 'worldwide' as their location. Hence, this approach referred to as 'Registered user location' was considered the least reliable way to obtain a location. If none of the above location extraction approaches were possible, the Tweets were excluded and could not be shown in the final visual interface.

As mentioned, the last two approaches were text-based, wherefore the locations obtained in this way could not be mapped directly but needed geographic coordinates. These were obtained prior to the toponym resolution step in the geoparsing using GeoPy[13]. This is a Python client locating the coordinates of addresses, cities, countries, and landmarks using third-party geocoders and other data sources. As this step proved to be computationally heavy, it was not optimal to let the geolocator go through each individual Tweet. Instead, a look-up table was created by finding the coordinates of the unique user locations and geoparsed locations that were mentioned more than T number of times. The threshold T was determined for the user locations and geoparsed locations separately by the distribution of the number of unique mentions of the locations. Most locations were mentioned quite few times and therefore less likely to be relevant to a flood event. Hence all locations only mentioned once were excluded. The threshold T was then found as the median of the distribution of the remaining locations. The look-up table was thereafter used to obtain coordinates for the user locations and geoparsed locations related to the individual Tweets.

3.4 Data Visualisation

In this section, the methods related to the final step of visually exploring the relevant and geolocated Tweets are described. Firstly, the purpose of the visualisation is introduced in relation to the visual analytics approach. Secondly, the analysis tasks and design requirements for the interactive visual interface are presented and justified. Lastly, the specifics of the implementation are outlined.

3.4.1 Purpose

The overall purpose of this project was to enable identification and visual exploration of extreme flood events from VGI in particular from Twitter data. The goal of the visualisation part was specifically to facilitate spatial, temporal and textual exploration of the flood events through the collected Tweets. To achieve this, the approach of visual analytics was used to develop an interactive visual interface to provide multifaceted representations of the data which could help users carry out analysis tasks more effectively [44]. Visual analytics has become an emerging class of visualisations concerned with linking interactive visual representations with underlying analytical processes, to reinforce human cognition such that sense making and reasoning can be effectively performed [63].

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

Essentially, the visual analytics techniques enable a knowledge discovery process structured as the sense-making loop [66] shown in Figure 3.14. This concept emphasises where visual analytics differs from information visualisation by having a higher focus on data analysis through all iterations of the loop [33].

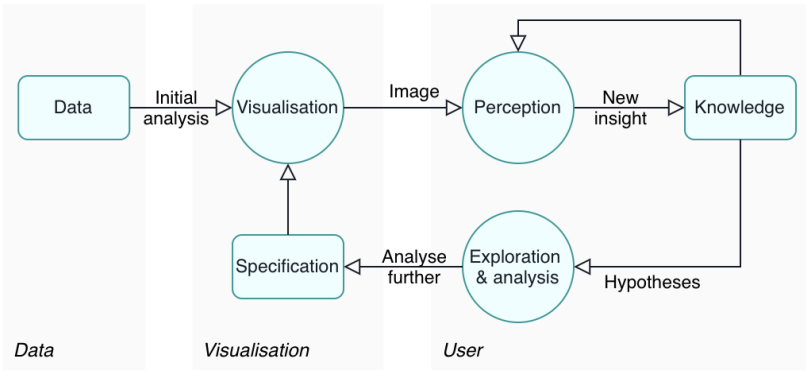


Figure 3.14: The sense-making loop for visual analytics

The analysis needs for the interface were established in collaboration with the project stakeholders based on what they want to be able to do with use of the interface.

Analysis needs:

1. Identification of ongoing/emerging flood events
2. Assessment/exploration of the situation's context and severity
3. Monitoring of the ongoing flooding situation over time

Following this, a number of concrete analysis tasks were defined to match the established analysis needs.

Analysis tasks:

AT1 Identification of areas of interest in the geospatial distribution of Tweets that indicate ongoing/emerging flood events

AT2 Identification of times or temporal ranges of interest that indicate ongoing/emerging flood events

AT3 Selection of an area of interest, containing a subset of Tweets, for further exploration

AT4 Selection of a temporal range of interest for further exploration

AT5 Meaningful summarising of the selected Tweets to give an overview of the content

AT6 Inspection of the textual content of the Tweets in order to reveal the context and relevance of a selected subset of interest

AT7 Exploration of the temporal distribution of Tweets and its relation to their geographic spread and variation over time as well as their content

Here, AT1-AT4 refer to the first analysis need of being able to detect ongoing events. AT5-AT6 aim to give insight into the relevance and context of the ongoing event to answer to the second analysis need. AT7 is concerned with the monitoring and exploration of the ongoing event as per third analysis need.

3.4.2 Design Requirements

Overall, the design of the visual interface was inspired by Keim's adjusted version of the visual information-seeking mantra. This included the steps 1) analyse first, 2) show the important, 3) zoom, filter and analyse further, and 4) details on demand [33]. Accordingly and based on the desired analysis tasks for the interface defined in Section 3.4.1, a set of design requirements were outlined in order to enable the user to perform these tasks.

Design requirements:

1. Create configuration pane through which the user can make selections and filter the Tweets
2. Display geospatial aspects of Tweets in different aggregation levels and enable the user to zoom in to areas and select Tweets of interest (answering to AT1, AT3, AT7)
3. Represent the temporal distribution of Tweets appropriately and enable the user to select peaks and define time ranges (answering to AT2, AT4, AT7)
4. Display the raw content of the Tweets in an individual as well as an aggregated view (answering to AT5, AT6, AT7)
5. Link visualisations such that any selection made applies to all views of the Tweets

3.4.3 Interface Design and Implementation

Based on the outlined analysis tasks and design requirements, a prototype interface was composed. The exploratory visualisation technique multiple coordinated views was used for the interface to allow exploration of the different data aspects of interest; space, time and textual context. This allows the user to get a better understanding of the data by interacting with it through different representations. Linking the views together allows the user to conduct their own investigations through reflecting selections made in one view in the other views [50].

The coordinated views for the interface were organised in different workspaces as shown in Figure 3.15.

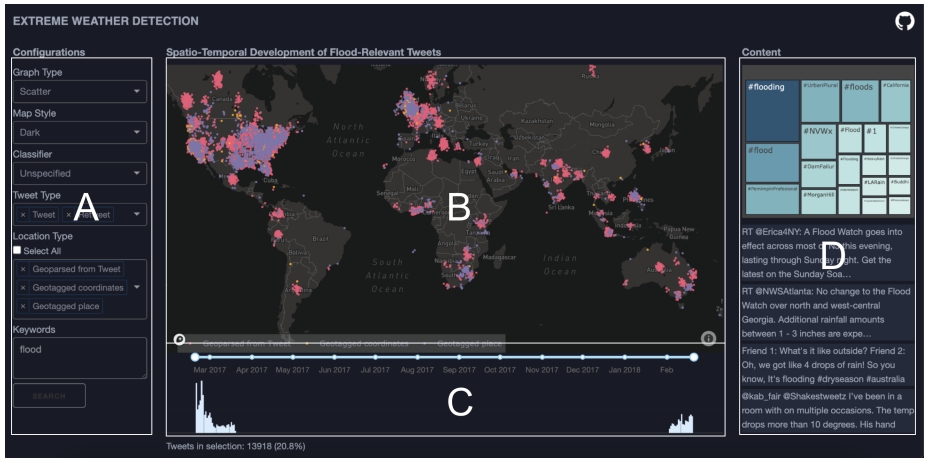


Figure 3.15: The visual interface organised in multiple coordinated views: A. Selection pane, B. Spatial data, C. Temporal data and D. Textual data

In line with Tufte's principles of graphical integrity, data variation was in focus rather than design variation. This included keeping the interface simple and clean using e.g. 2D instead of 3D visualisations and a minimal number of different colors for the major components. Further, the data was sought revealed at several levels of detail without distortions in addition to maximising the data-ink ratio to make the data stand out clearly [65].

In order for a user to successfully explore the data in the interface, the data needed to be appropriately prepared for inclusion [50]. Hence, the Tweets were

pre-processed as introduced in Section 3.1, then classified as flood-relevant or not using the classification models and lastly the location extraction algorithm was applied to the flood-relevant Tweets. When each Tweet was located, it was found that some Tweets had identical coordinates, mainly for the locations obtained through geoparsing and user locations. This was not optimal with the intention of showing the density of Tweets on a map. The Tweets would then appear on top of each other and skew the picture of how many Tweets really existed in an area. A solution to this was to spread the identical points by adding Gaussian noise to their coordinates. For the n identical points, noise was added by drawing n sample points from a multivariate normal distribution

$$\mathbf{X} \sim \mathcal{N}_n(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \boldsymbol{\mu} = (x_0, y_0), \boldsymbol{\Sigma} = \begin{bmatrix} 0.7 & 0 \\ 0 & 0.7 \end{bmatrix} \tag{3.13}$$

where the random variable $\mathbf{X} = ((x_1, y_1), \ldots, (x_n, y_n))^T$ contains the new, adjusted points. The mean of the noise is the original point itself $\boldsymbol{\mu} = (x_0, y_0)$, where x_0 is the latitude and y_0 is the longitude. The variance $\boldsymbol{\Sigma}$ was chosen based on visual inspection of the resulting noise added to the points. An example of the spread of 44 Tweets all located in Queensland by geoparsing with $\boldsymbol{\mu} = (-22, 144)$ can be seen in Figure 3.16.

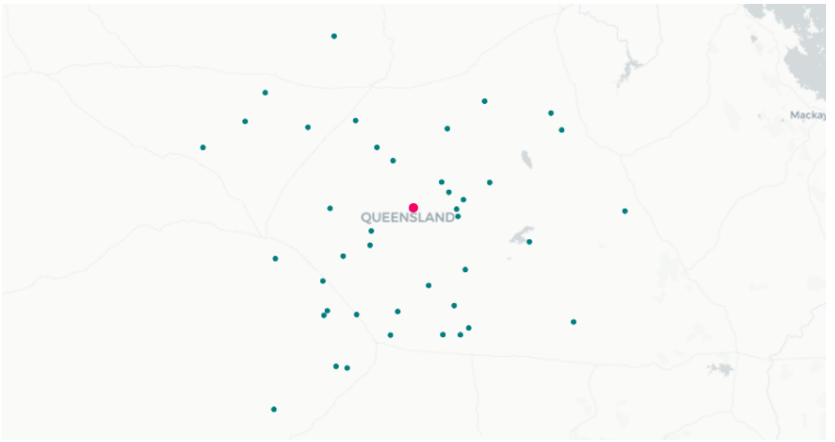


Figure 3.16: The spread of 44 Tweets all located in Queensland by geoparsing with $\mu = (-22, 144)$ marked with pink

Besides discarding some Tweets during the pre-processing steps, new variables were also added. The final data included in the interface thereby contained the variables presented in Table 3.3.

Variable	Type	Description
full_text	String	The actual UTF-8 text of the Tweet
hashtags	List of strings	Hashtags parsed from Tweet
created_at	datetime64	UTC time when the Tweet was created
date	datetime64	Date when the Tweet was created
user_name	String	Name chosen by user
user_location	String	Registered location by user
source	String	Utility used to post the Tweet
type	String	Indicates whether it is a Tweet or Retweet
localisation	String	Indicates location type
retweet_count	Int64	Duplicate count of Tweet text
lat	float64	Latitude coordinate for Tweet
lon	float64	Longitude coordinate for Tweet
logistic regression	int64	1 if classified relevant by LR, else 0
random forest	int64	1 if classified relevant by RF, else 0
cnn	int64	1 if classified relevant by CNN, else 0
ulmfit	int64	1 if classified relevant by ULMFiT, else 0

Table 3.3: Data visualisation variables

According to the defined analysis tasks and requirements, it seemed suitable to build an application with a custom user interface using the Python libraries, Plotly[14] and Dash[15]. The Dash Core Component library contains higher-level components such as graphs, sliders and drop-down menus that can be linked though callback functions. The graph components have specific variables utilised for user interaction; hoverData, clickData, selectedData and relayoutData, that are updated when the user hovers over a point, clicks on a point, selects regions of points, or zooms or pans in a graph. These can be used to enable multidimensional exploration and linking of views. In terms of layout, the Dash Bootstrap Components and separate CSS files were used to obtain a consistent styling.

The Python application was hosted using the Github platform for code sharing and development. The data used in the application at once was thereby restricted by Github's file limit of 100MB. In order to use the application, the repository extreme-weather-detection can be cloned and ran locally by the user. It is also deployed to the cloud application platform Heroku[16] and thereby made available as a public website: extremeweatherdetection.herokuapp.com. It should be noted that the application is under development and could need further improvements.

[14]https://plotly.com/python/
[15]https://dash.plotly.com/
[16]https://www.heroku.com

Chapter 4

Results

This section presents the results of the project. Firstly, the performance of the different classifiers is assessed based on the measures presented in Section 3.2.4. Secondly, the classifiers are evaluated by inspecting the predictions they made for the unlabelled data set along with an evaluation of the locations found by the location extraction algorithm. This finally leads to a description of the visual interface and a demonstration of it on a use case.

4.1 Performance of Text Classifiers

The performance of the text classifiers for each of the three model variants described in Section 3.2 is evaluated in this section. These variants are referred to as 'Original Tweets', 'Remove keywords' and 'Replace places'. Chosen visualisations for the classifiers are presented, which are used as an additional assessment for evaluating whether the classifiers perform satisfactorily.

4.1.1 Classic Algorithms

The result of the classification using the classic machine learning models is presented and evaluated in the following.

4.1.1.1 Logistic Regression

The performance of the logistic regression models is presented in Table 4.1. The model for the variant with keywords removed from the Tweets performs the best on all performance measures. This includes an accuracy of 93.90% compared to 93.61% for the model using original Tweets, though this a only a slight difference. The model for the variant with replacement of places is performing below the other two on all performance measures. This variant also results in the most false negatives and false positives seen in the confusion matrix.

	Accuracy	Precision	Recall	F1 score	Confusion matrix
Original Tweets	0.9361	0.9516	0.9204	0.9357	$\begin{bmatrix} 996 & 50 \\ 85 & 983 \end{bmatrix}$
Remove keywords	0.9390	0.9563	0.9213	0.9385	$\begin{bmatrix} 1001 & 45 \\ 84 & 984 \end{bmatrix}$
Replace places	0.9201	0.9377	0.9017	0.9193	$\begin{bmatrix} 982 & 64 \\ 105 & 963 \end{bmatrix}$

Table 4.1: Logistic regression performance measures

To further evaluate the two best logistic regression models, the weights given to the words in the vocabulary of the Tweets are investigated. The relevant words are those that the models gave positive weights, whereas the non-relevant words were those given negative weights. These words are presented with word clouds where the size of the single word is relative to its absolute weight in Figure 4.2 for 'Original Tweets' and 4.1 for 'Remove keywords'.

(a) Relevant words **(b)** Non-relevant words

Figure 4.1: Word clouds that show the words classified as relevant and non-relevant by the logistic regression model variant 'Original Tweets'

(a) Relevant words **(b)** Non-relevant words

Figure 4.2: Word clouds that show the words classified as relevant and non-relevant by the logistic regression model variant 'Remove keywords'

The word clouds for relevant words show that the models manage to give positive weights to words that are more likely to be relevant to flood events such as flood, water, crisis and help. Oppositely, the non-relevant words appear to concern a broader range of words such as happy, someone and beach. By comparing the words presented for the two variants, it appears that the relevant words for 'Original Tweets' are specific to the Alberta and Queensland floods. This indicates that the model for this variant might struggle to generalise to other flood events, which inhibits its performance.

4.1.1.2 Random Forest

The performance metrics of the random forest classifiers with the initial parameters introduced in Section 3.2.1.3 are presented in Table 4.2. The variant with the keywords removed performs the best with 92.53% accuracy compared to 89.82% for the variant with replacement of places. The model using original Tweets has the lowest performance on accuracy, recall and F1 score, but outperforms the variant with replacement of places on precision. As the random forest algorithm was primarily used for establishing a performance baseline, no focus was dedicated to tuning its parameters.

	Accuracy	Precision	Recall	F1 score	Confusion matrix
Original Tweets	0.8841	0.9468	0.8165	0.8768	$\begin{bmatrix} 997 & 49 \\ 196 & 872 \end{bmatrix}$
Remove keywords	0.9253	0.9470	0.9026	0.9243	$\begin{bmatrix} 992 & 54 \\ 104 & 964 \end{bmatrix}$
Replace places	0.8982	0.9392	0.8539	0.8946	$\begin{bmatrix} 987 & 59 \\ 156 & 912 \end{bmatrix}$

Table 4.2: Random forest performance measures

As the variant with keywords removed performed the best, only the classifier for this variant is explored further. One of the decision trees in the random forest of 100 trees was chosen randomly for inspection. The tree does not include all words in the vocabulary of the Tweets as the individual trees were trained with different random subsets of the words. The chosen tree is shown in its full depth of 11 in Figure 4.3. As the full tree is hard to inspect in detail, two sub-trees are marked and chosen for further inspection in Figure 4.4.

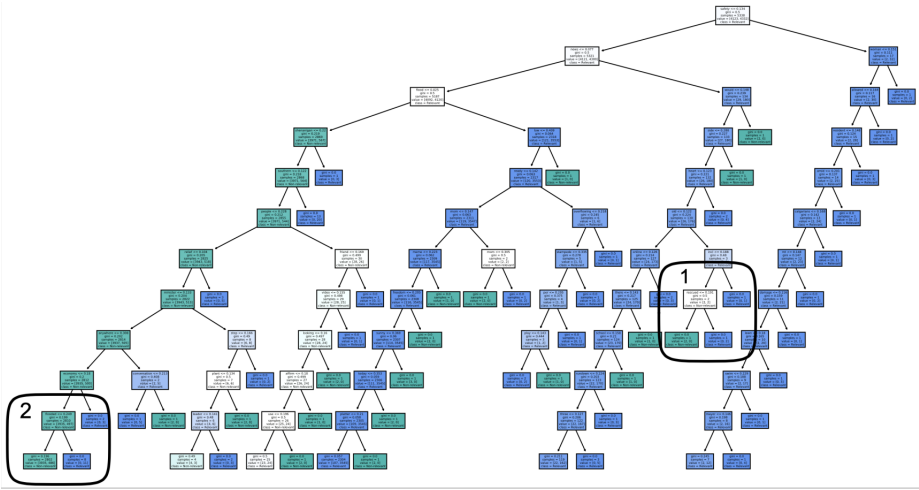
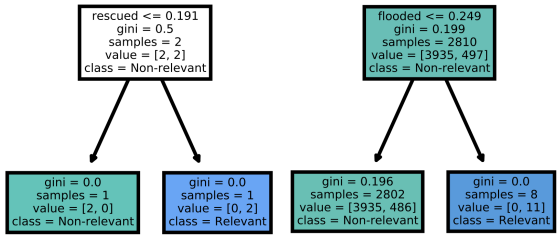


Figure 4.3: Randomly chosen decision tree from the random forest of 100 decision trees for model variant 'Remove keywords'. Two subtrees are marked for further inspection in Figure 4.4



(a) Subtree 1 **(b)** Subtree 2

Figure 4.4: Two subtrees from the randomly chosen decision tree in Figure 4.3 for model variant 'Remove keywords'

The decision node for subtree 1 is split in two leaf nodes based on whether the word 'rescued' is included in the Tweet. Specifically, the decision is based on the TF-IDF score of the word. If the score is above 0.191, which is only the case if the word is included in the Tweet, then it is classified as flood-relevant. If the word is not included, the TF-IDF score will be 0 and below 0.191, wherefore the input will be classified as non-relevant. Likewise, the word 'flooded' determines the split for subtree 2. These examples show that the tree has learned that the words 'rescued' and 'flooded' are typically included in flood-relevant Tweets.

4.1.2 Deep Learning

The performance for the CNN with the architecture and parameters described in Section 3.2.2.1 is evaluated in two steps. First, to spot potential overfitting, the accuracy and loss for the validation and training sets were compared for the three model variants. These are shown for the variant with removed keywords over the 10 epochs in Figure 4.5. The other variants had similar results.

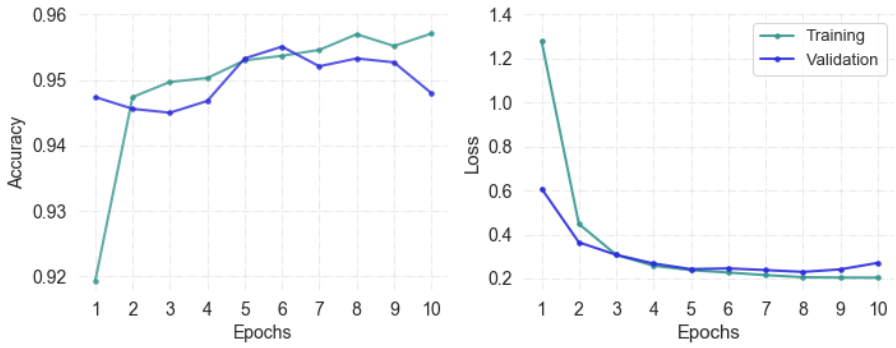


Figure 4.5: Validation and training accuracy as well as validation and training loss over 10 epochs for the CNN model variant 'Remove keywords'

The loss functions on the right-hand side illustrate that the optimal performance already occurred after 4-5 epochs with the validation error being equal to the training error. After this, the model starts overfitting the training data as the validation loss exceeds the training loss. Hence, the CNN's were trained again, though this time the training was stopped after 5 epochs. The resulting performance measures after this second training are reported in Table 4.3 for the three model variants.

	Accuracy	Precision	Recall	F1 Score	Confusion matrix
Original Tweets	0.9432	0.9489	0.9363	0.9425	$\begin{bmatrix} 1010 & 53 \\ 67 & 984 \end{bmatrix}$
Remove keywords	0.9461	0.9399	0.9524	0.9430	$\begin{bmatrix} 999 & 64 \\ 50 & 1001 \end{bmatrix}$
Replace places	0.9357	0.9378	0.9324	0.9351	$\begin{bmatrix} 998 & 65 \\ 71 & 980 \end{bmatrix}$

Table 4.3: CNN performance measures after 5 epochs

The results were similar for the three variants, but it is observed that the model with keywords removed performed slightly better with an accuracy of 94.61%, compared to 94.32% for the second best variant using original Tweets.

Since the models made very accurate classifications of the Tweets, there was only room for small performance improvements. Therefore, it was not chosen to tune the hyperparameters or change the architecture of the CNN to increase the performance.

4.1.3 Transfer Learning

The performance of the ULMFiT classifier is presented in Table 4.4. Both the variant with original Tweets and with keywords removed obtained a promising accuracy of 94.99%. In general, these two variants had almost identical performances that exceeded the performance of the last variant with replacement of places.

	Accuracy	Precision	Recall	F1 score	Confusion matrix
Original Tweets	0.9499	0.9487	0.9522	0.9505	$\begin{bmatrix} 991 & 55 \\ 51 & 1017 \end{bmatrix}$
Remove keywords	0.9499	0.9512	0.9494	0.9503	$\begin{bmatrix} 994 & 52 \\ 54 & 1014 \end{bmatrix}$
Replace places	0.9338	0.9462	0.9213	0.9336	$\begin{bmatrix} 990 & 56 \\ 84 & 984 \end{bmatrix}$

Table 4.4: ULMFiT performance measures

Due the very nature of ULMFiT as a transfer learning technique with limited possibilities of making changes, no further inspection of the models was carried out. Hence, the models were only evaluated based on the above performance measures.

4.1.4 Comparison of Classifiers

To obtain an overall impression of the results, the performances of the different text classifiers are compared by visualising the important measures of accuracy and F1 score in Figure 4.6.

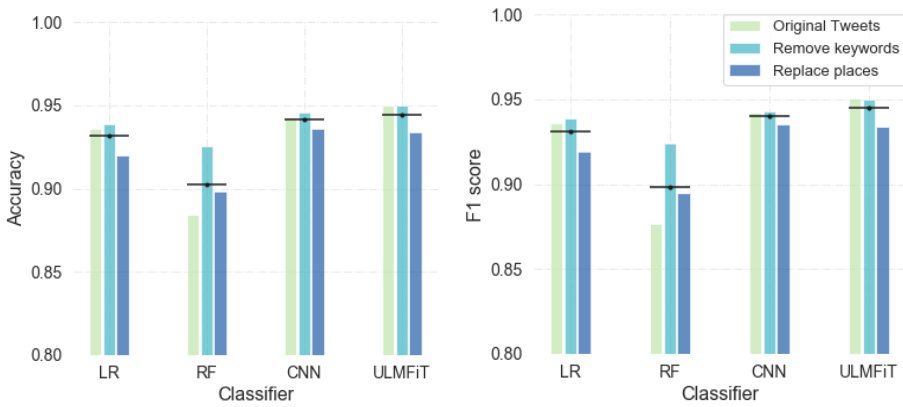


Figure 4.6: Accuracy and F1 score for the four text classifiers using the three model variants. The average for the variants is marked with a black line.

The result is almost identical for the two measures. All models performed surprisingly well with the majority achieving accuracies and F1 scores above 90%. In general, the best performance was obtained for the variant with removed keywords, though the variant with original Tweets performed almost as good for three of the four models. However, one should be aware that this might be due bias in this variant, which was particularly shown for the logistic regression model by inspecting the words with highest weights. It is only for the random forest classifier that the variant with replacement of places performed second best.

The models trained using transfer learning through ULMFiT achieved the best results, closely followed by the deep learning model, CNN. Out of the two classic algorithms, the logistic regression performed the best. Overall, this indicates that using word embeddings to represent the textual input improves the models' ability to differentiate between Tweets that are relevant and non-relevant to flood events.

4.2 Visual Interface

This section first includes a description of the views in the final visual interface. Then, more details in terms of the usage of the interface are introduced alongside specific design choices.

4.2.1 Interface Description

The individual views that were implemented in the visual interface are described in relation to the design requirements presented in Section 3.4.2. The coordinated views of the interface are shown in Figure 4.7.

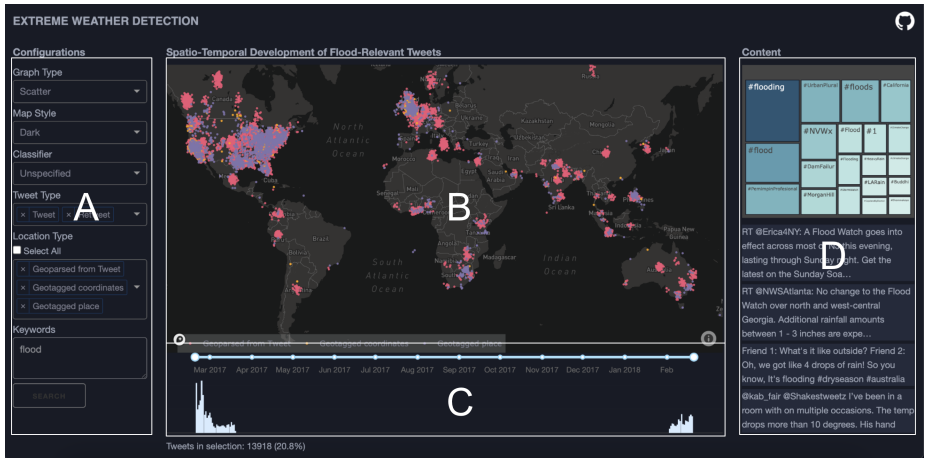


Figure 4.7: The visual interface with multiple coordinated views: A. Configuration pane, B. Spatial data, C. Temporal data and D. Textual data

The left part of the visual interface holds the configuration panel (A) allowing the user to make selections regarding the representations and applying filters. This panel fulfills design requirement 1. The central view of the interface is the map view (B) which is overlaid with geospatial information regarding the Tweets. This view facilitates identification of spatial relationships and patterns, thereby fulfilling design requirement 2. The bottom view holds a histogram (C) showing the development of the number of flood-relevant Tweets per day. This also includes a timeline slider over the different months, enabling filtering on time slices and periods. This part of the interface fulfills design requirement

3. The right part (D) of the interface includes a treemap presenting the most frequent hashtags in the Tweets and a scroll-able table displaying the original Tweets to enable inspection of the content. Hence, this fulfills design requirement 5.

4.2.2 Usage and Interaction

The different ways in which to use and interact with the visual interface are introduced in the following. Design requirement 5 in Section 3.4.2 is fulfilled by ensuring that the views are coordinated and interactive. This facilitates and eases the comparison of the visualisations to arrive at better analysis results [27].

The locations of the Tweets are plotted on a 2D interactive map that is initially zoomed out to create an overview. The user can navigate on the map using pan and zoom options to explore patterns within specific areas and locations. Brushing and linking is used by changing the color saturation to highlight the points that are clicked or selected on the map and opposite to hide the non-selected points. Only the data corresponding to the points selected in view B will be included in the other views.

The map in view B is also combined with the timeline slider that enables filtering of time periods. The temporal histogram allows for the user to more easily detect periods of time with larger amounts of Tweets, i.e. peaks. Hence, the user is explicitly encouraged to explore such time periods by interacting with the slider or selecting the peaks in the histogram. The slider is updated according to the horizontal range selected on the histogram by the user. Brushing and linking is used as for the map, but here based on the selected bars or time periods on the slider. The number of Tweets at different dates are shown upon hovering as seen in Figure 4.8.

Figure 4.8: Timeline slider demonstration of hovering and selection options

Interactivity is further encouraged through the configuration panel. This enables the user to iterate back and forth in the sense-making loop (in Figure 3.14) to view the data from different perspectives and obtain new insight. Again as the views are coordinated, any selection made in the panel will apply to all views.

Dropdown menus allow for the user to filter the underlying Twitter data on different variables. The third one-level dropdown menu admits the user to specify a classification model. The default option is 'Unspecified', hence the initial view includes the Tweets that were classified as relevant by at least one of the four implemented models introduced in Section 3.2. If preferred, the user can instead choose to include the Tweets that were classified as relevant by only one of the model options shown in Figure 4.9. Multi-dropdown menus instead enable the user to choose more than one option. This applies to the Tweet type and location type, where a checkbox was used to provide the option of selecting all location types at once. The options are shown in Figure 4.9.

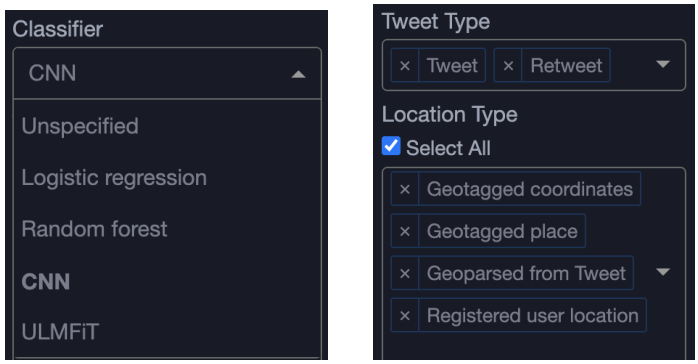


Figure 4.9: One-level dropdown menu for selection of classification and multi-level drowpdown menus for tweet Type and location type

A text area with a search button as shown in Figure 4.10 can be used to filter the Tweets on any keyword or hashtag. Only the Tweets containing the words written in this field will then be included in the visualisations.

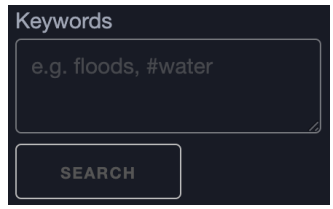


Figure 4.10: Keyword or hashtag filtering option

This can be confirmed by inspecting the content of the individual Tweets or in an aggregated manner by checking the top-20 most frequent hashtags as in Figure 4.11. The count of each hashtag is provided by hovering over the hashtags.

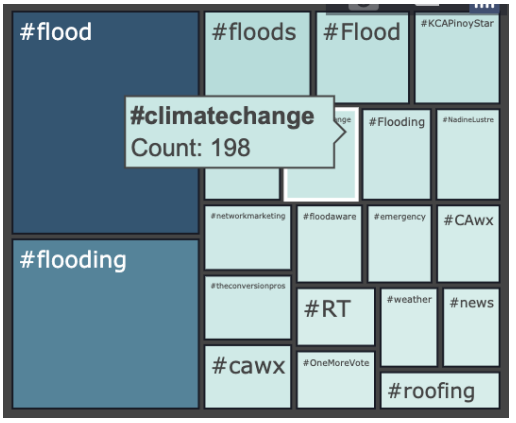


Figure 4.11: Treemap with most frequent hashtags used in the Tweets in selection

The first two drop-down menus in the configuration panel applies to the map view. The encoding for the graph type was selected from the first one-level drop-down menu as seen in Figure 4.12. This was then represented on a world map with a selected Mapbox[1] style from the second one-level drop-down menu. The chosen styles, light, dark, streets and satellite, were available using a Mapbox access token.

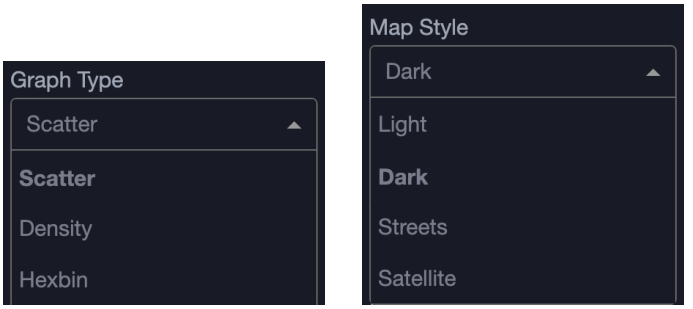


Figure 4.12: Dropdown menus for selection of the graph type and map style

[1]https://www.mapbox.com/

The three visual encoding options for the graph type all provide different insights about the data. Firstly, to focus on the individual Tweets and their specific locations, an instance of the scattermapbox class from Plotly's graph objects was used to create a scatter map. Hence, the longitude and latitude pairs of the Tweets are represented as scatter points as seen in 4.13. This enables separability by position of the points when zooming in as well as separability by hue (color) of the points depending on the location type of the Tweets. As a part of the knowledge discovery process, mouse hovering is enabled to provide details-on-demand for each Tweet. A challenge of the scatter map is that large amounts of points can be overlapping in the same regions, making the impression chaotic unless the zoom factor is very high.

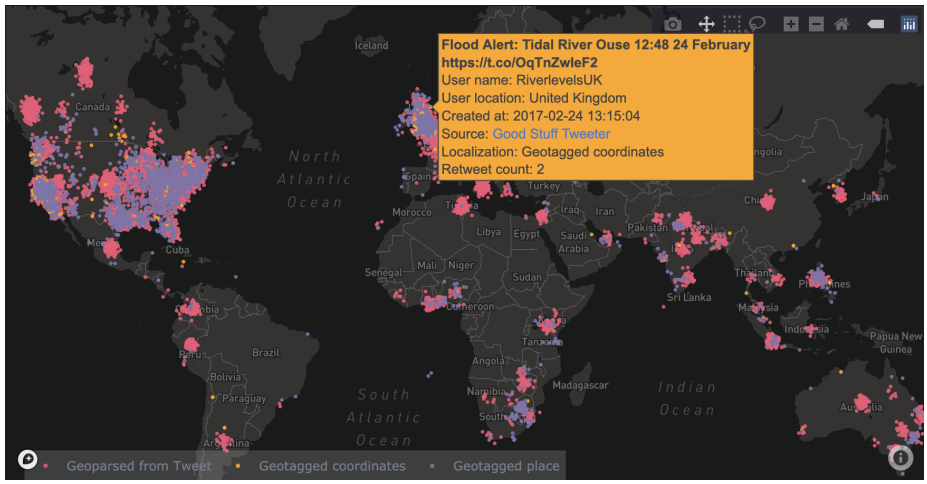


Figure 4.13: Scatter map showing the spatial distribution of the flood-relevant Tweets

Secondly, for the purpose of perceiving larger amounts of Tweets at once, a density heatmap was created using the densitymapbox class as seen in 4.14. It aggregates the scatter points and color-codes them with different color saturation based on the number of points in each group. It is then easier to perceive density of points independently of the zoom factor.

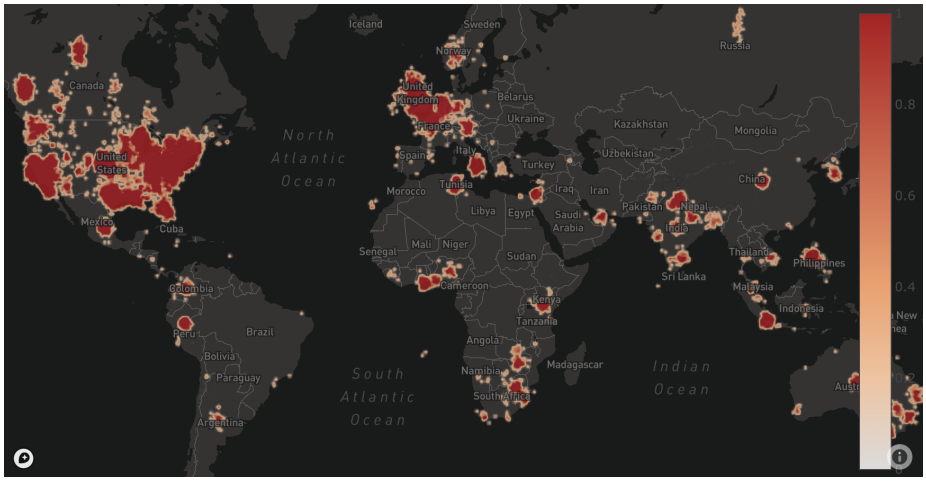


Figure 4.14: Density heatmap visualising spatial differences of the density of flood-relevant Tweets

Thirdly, since only the density and not the different counts of Tweets were presented in the density heatmap, an annotated hexbin map was created with Plotly's figure factory module as seen in Figure 4.15. This map type aggregates the scattered points into connected hexagons and the number of points in each are represented with color gradients and annotations upon hovering. The colorbar presents the range of counts from minimum to maximum.

Figure 4.15: Hexbin map visualising the spatial differences of the number of flood-relevant Tweets

4.3 Evaluation of Project Pipeline

This section presents the results from going through the steps of the project pipeline in Figure 3.1 using the unlabelled English Tweets described in Section 3.1. First, the English Tweets were classified as relevant to flood events or not. The classifications were evaluated through presenting spot checks of specific Tweets. Next, the location extraction algorithm was applied to Tweets that were classified as relevant by at least one classifier. Lastly, it is demonstrated how the relevant and geolocated Tweets can be explored in the visual interface. This identifies some historical flood events that happened in the period of 2016-2018. These can possibly be confirmed by comparing to real historical records of flood events making it possible to assess the overall contribution of the pipeline.

4.3.1 Text Classification

The classifiers for the two best performing variants 'Original Tweets' and 'Remove keywords' were tested on the subsets of unlabelled English Tweets from 2016-18. The number of Tweets and the percentage of Tweets classified as relevant by each classifier, by all classifiers and by at least one classifier are presented in Table 4.5 for the years 2016-18 and in total. The numbers are separated with a forward slash, where the first relates to the variant 'Original Tweets' and the second relates to the variant 'Remove keywords'.

	2016	2017	2018	Total
Number of Tweets	80,933	74,386	95,699	251,018
LR (%)	51.1 / 73.6	44.6 / 68.0	40.7 / 65.8	45.5 / 69.1
RF (%)	64.3 / 67.8	58.9 / 65.1	57.0 / 63.3	60.1 / 65.4
CNN (%)	58.0 / 61.3	58.3 / 61.7	59.4 / 63.9	58.6 / 62.3
ULMFiT (%)	65.0 / 69.7	56.8 / 64.2	53.9 / 62.0	58.6 / 65.3
Rel. by all (%)	27.2 / 37.5	20.4 / 34.1	21.4 / 36.3	23.0 / 36.0
Rel. by at least one (%)	88.9 / 91.4	87.6 / 91.1	84.2 / 87.6	86.9 / 90.0

Table 4.5: Number of Tweets and the percentages of Tweets classified as relevant by each classifier, by all classifiers and by at least one classifier from 2016-2018 and in total for the variants 'Original Tweets' and 'Remove keywords' separated with a forward slash

Overall, the percentages of Tweets classified as relevant for the classifiers and

model variants seem quite similar across the different years. Hence, the total percentages are accompanied by Figure 4.16 to ease the comparison.

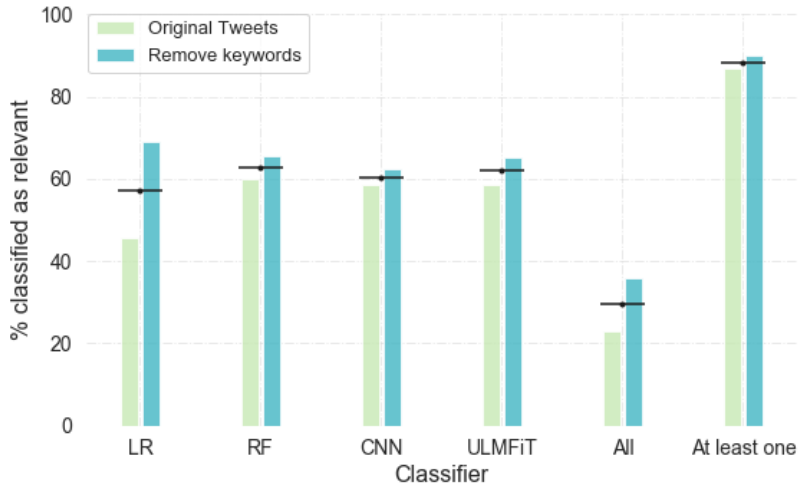


Figure 4.16: Total percentages of Tweets classified as relevant by each of the classifiers, by all classifiers and by at least one classifier for the variants 'Original Tweets' and 'Remove keywords'. The average for the variants is marked with a black line.

From Figure 4.16, it can be seen that around 90% of the Tweets were classified as relevant by at least one of the classifiers, whereas only 20-40% were classified as relevant by all classifiers. This shows that there must be some differences when it comes to how the individual Tweets are classified.

In general, the models for the variant 'Original Tweets' classify fewer Tweets as relevant than the models for the variant 'Remove keywords'. The individual classifiers for both variants predict 55-65% of the Tweets as relevant, except for the logistic regression models that differ the most by classifying between 20-25% fewer Tweets as relevant with the variant 'Original Tweets'. This demonstrates that the models using the original Tweets likely have bias towards the specific flood events in Alberta and Queensland, thereby being less able to generalise to other events. Two examples of Tweets that were classified as non-relevant by the variant 'Original Tweets' but relevant by the variant 'Remove keywords' are shown in Figure 4.17.

Figure 4.17: Examples of Tweets classified as relevant by the variant 'Remove keywords' but non-relevant by the variant 'Original Tweets'

These examples are both potentially related to an emerging flood event. By using the models for the variant with original Tweets, this information would risk being excluded. Hence, the models for this variant is not considered for further inspections.

As mentioned, it seems that the classifiers make different predictions for individual Tweets. That is the same Tweets are not necessarily classified as relevant by the individual classifiers, which naturally decreases the reliability of their predictions. For comparison, the prediction similarity between the four classifiers for the variant with removed keywords is presented in Figure 4.18.

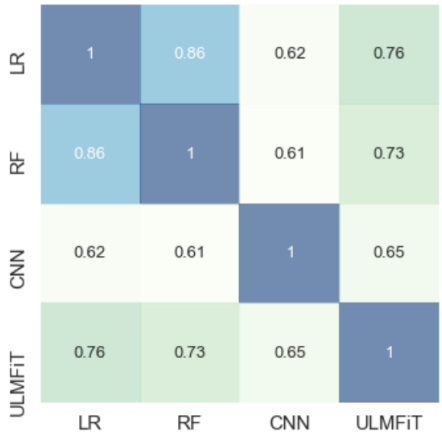


Figure 4.18: Heatmap showing the similarity between the predictions made by the four classifiers for the variant 'Remove keywords'

From Figure 4.18, it can be seen that the classic algorithms have the highest similarity in terms of the Tweets that they predict as relevant and non-relevant to flooding. In general, the lowest similarities occur between the CNN and the

other models by only around 60% of the Tweets being classified similarly. The dissimilarities can be inspected through the specific content of the Tweets, hence some examples of Tweets that are classified differently are shown in Figure 4.19.

(a) Non-relevant by ULMFiT but relevant by others

(b) Non-relevant by RF but relevant by others

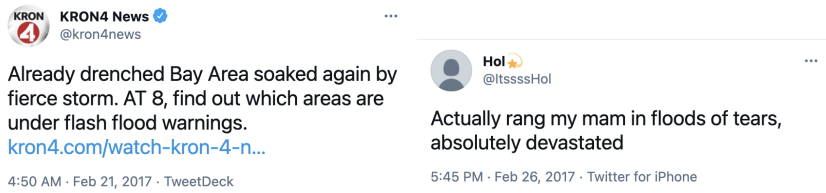
(c) Relevant by RF and LR, non-relevant by ULMFiT and CNN

(d) Relevant by CNN and ULMFiT, non-relevant by RF and LR

Figure 4.19: Examples of Tweets classified differently by the individual classifiers for the variant 'Remove keywords'

The example in Subfigure 4.19 (a) demonstrates the challenge in the text classification task for this project. It contains flood-related terms such as alert, stay safe and even flooding, though it does not concern an actual flood event. The fact that ULMFiT is able to classify it as non-relevant shows that it might be better at capturing the semantics than the other classifiers. The example in Subfigure 4.19 (b) seems to be flood-relevant, though this is not found by the random forest classifier which substantiates why it performs lower than the other classifiers. The examples in Subfigure 4.19 (c) and 4.19 (d) further demonstrate that the classic algorithms fall short when it comes to understanding the correct meaning of the Tweets, that however seems to be possible for the complex models.

For an overall assessment, examples of Tweets that were classified as relevant to flooding by all classifiers are seen in Figure 4.20. One example seems to be a true positive, whereas the other is considered a false positive. Oppositely, Figure 4.21 shows examples of Tweets that all classifiers predicted as non-relevant and that are considered a true negative and false negative, respectively.



(a) "True positive" **(b)** "False positive"

Figure 4.20: Tweets that were classified as relevant by all classifiers

(a) "True negative" **(b)** "False negative"

Figure 4.21: Tweets that were classified as non-relevant by all classifiers

The false positive example shows how all models struggle to entirely exclude Tweets that use flood-related words figuratively such as 'floods of tears'. Though, the true negative example demonstrates that the models are actually capable of this for some cases. In general, no consistent patterns can be immediately inferred in terms of the predictions, which again adds to a decrease in the reliability of them. Important to note is however that Figure 4.19 and 4.21 only show a few examples of the predictions, hence cannot be used to draw overall conclusions regarding the classifiers.

4.3.2 Location Extraction

The next step included applying the location extraction algorithm on the Tweets that were classified as relevant to flooding by at least one of the classifiers. This resulted in finding locations for 59.17% of the English Tweets in total. The remaining Tweets that could not be related to a location were not included in the visual interface. The number of Tweets found relevant by at least one classifier as well as the their distribution between the location levels; geotagged coordinates, geotagged place, geoparsed from Tweet and registered user location, are shown in Table 4.6 over the years 2016-18 and in total. The table is accompanied by Figure 4.22.

	2016	2017	2018	Total
Number of relevant Tweets	45,282	45,219	58,022	148,527
Geotagged coordinates (%)	1.78	1.05	0.75	1.19
Geotagged place (%)	4.37	3.92	3.63	3.97
Geoparsed from Tweet (%)	20.27	16.02	17.27	17.96
Registered user location (%)	73.58	79.01	78.35	76.88

Table 4.6: The number of Tweets classified as relevant by at least one classifier and percentages of these Tweets located by each of the location levels over the years 2016-2018 and in total

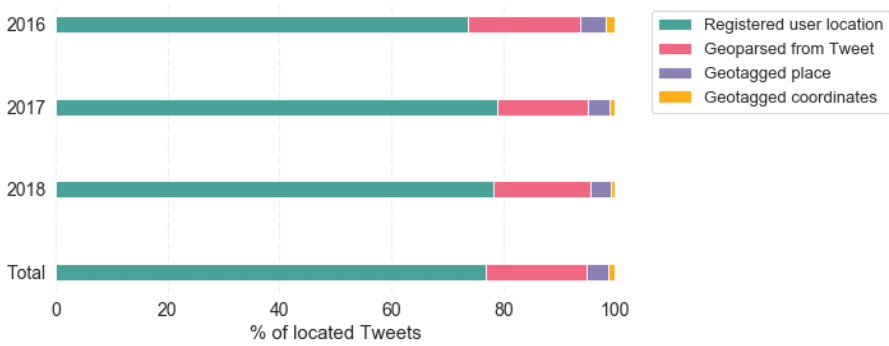


Figure 4.22: Percentages of Tweets located by each of the location levels over the years 2016-2018 and in total

It can be seen that most Tweets were related to a location based on the registered user location, which was the least trustworthy level. Oppositely and as expected, only around 5% of the Tweets were located based on the most trustworthy level by being geotagged. Despite the different levels of trust in the location types, this confirms the reason for using geoparsing as it means for less flood-relevant Tweets to be discarded.

The success of the algorithm in terms of the geoparsing relies heavily on the models and libraries chosen for the initial typonym recognition step. Hence this is evaluated by inspecting the places with geographic reference that were found by NER. By inspection it was found to only have issues for very few Tweets, though a few examples of the problems that were encountered are presented in Table 4.7.

Tweet	Geoparsed locations
Staying inside in flood ravaged *Calgary* trying to avoid the massive amounts of *Mosquitos* that have now hatched.	*Calgary*, *Mosquitos*
"Beautiful *Bohol*" is turning out to be my longest series in *Instagram*. There's just too many highlights, sorry for flooding your feed.	*Bohol*, *Instagram*

Table 4.7: Questionable locations found by NER

The examples show how the NER relied partly on the fact that proper nouns are always capitalised, wherefore it often related capitalised words to a place despite the fact that they were not phrased as such in the Tweet. Here this applies to the words 'Mosquitos' and 'Instagram'. Oppositely, the NER would struggle to identify places that were not capitalised and likely missed extracting potential candidates in this way.

To further inspect the locations, the unique locations geoparsed from the Tweets and registered user locations are shown on a world map in Figure 4.23.

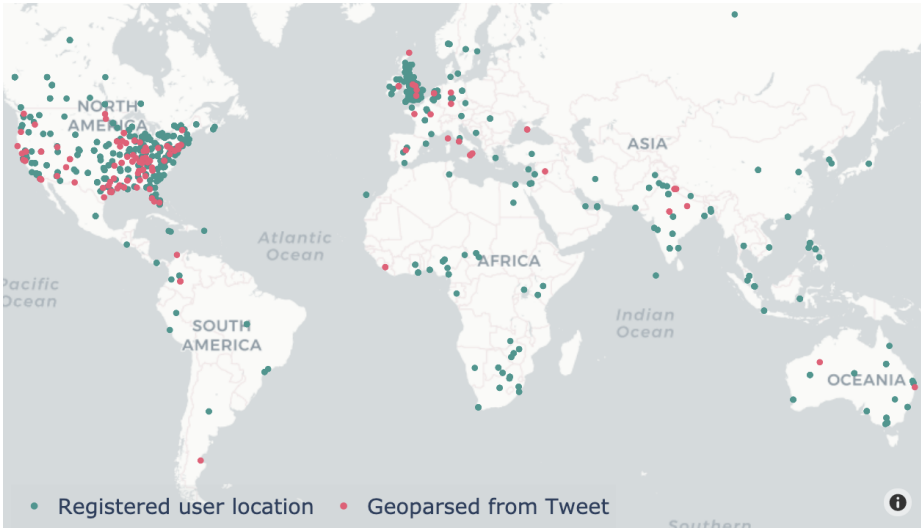


Figure 4.23: Locations geoparsed from Tweets using NER as well as registered user locations

No unusual patterns can be seen in Figure 4.23 demonstrating that the few examples of doubtful locations presented in Table 4.7 did not skew the overall picture particularly. Besides, there are steps in the location extraction algorithm that might exclude Tweets in which unusual place names were found due to them being too far apart or being discarded prior to geocoding. It can be seen that most of the English Tweets were related to various locations across the US and the UK.

4.3.3 Interface Demonstration

This section gives a demonstration of how the Twitter data can be explored in the visual interface. The interface includes the Tweets from the unlabelled data set that could be related to a geographic location and were classified as relevant to a flood event by at least one classifier.

The visual interface is exemplified by going through an exploration of a subset of the English Tweets related to flood events in 2017-18. An evaluation of the interface could thereby be done by comparing identified events to real historical records of flood events in the same period. The initial overview is given in Figure 4.24, where it can be seen that there are 66,848 Tweets in total when no filter is applied.

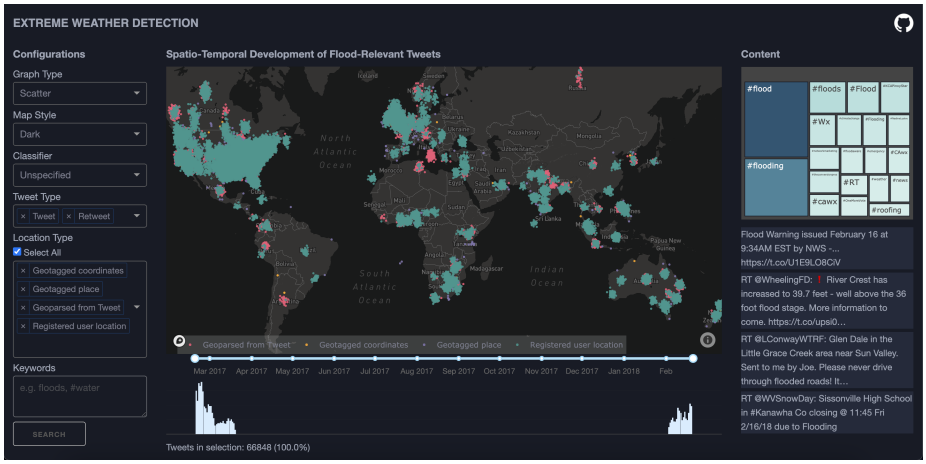


Figure 4.24: Initial overview of the flood-relevant English Tweets in 2017-18 with no selections made

As each of the legends of the map is connected to a distinct color of the scatter points, it is seen that the Tweets dominating the scatter map are the teal colored points that are the registered user locations. If the user considers these as insufficiently reliable locations, they can be unselected in the configuration panel. This then yields 14,282, i.e. 21.4%, relevant Tweets as shown in Figure 4.25. At the first look at the scatter map in this figure, the places in the world with a large number of Tweets can be detected.

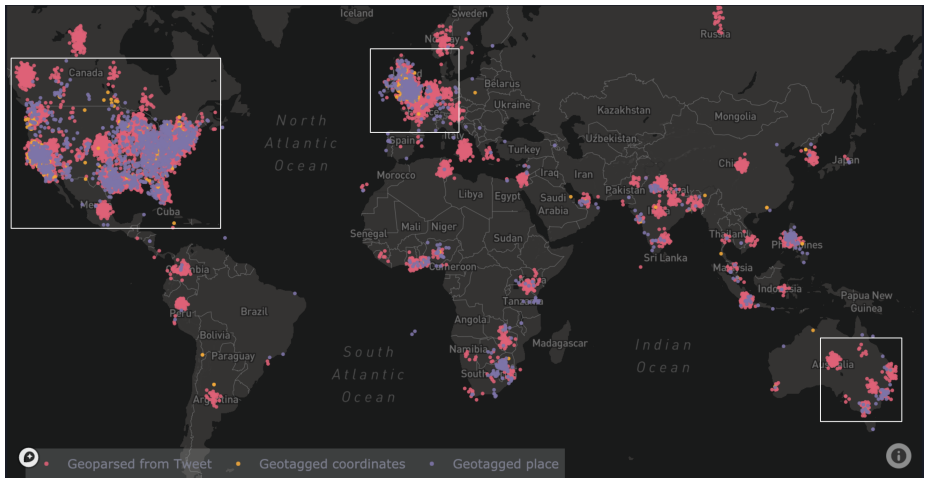


Figure 4.25: Scatter map for spatial exploration of the Tweets excluding the location type 'Registered user location'

The spatial observations are confirmed if the graph type 'Hexbin' is selected as seen in Figure 4.26, as the gradual coloring based on number of Tweets makes it easy to spot the 'red' areas with the highest density of Tweets. As expected, the majority of the Tweets are located where people tend to use Twitter more frequently and Tweet in English, that is in the US, Canada, the UK and Australia. Several Tweets are also observed in areas that are typically affected by flooding, such as the city Jakarta.

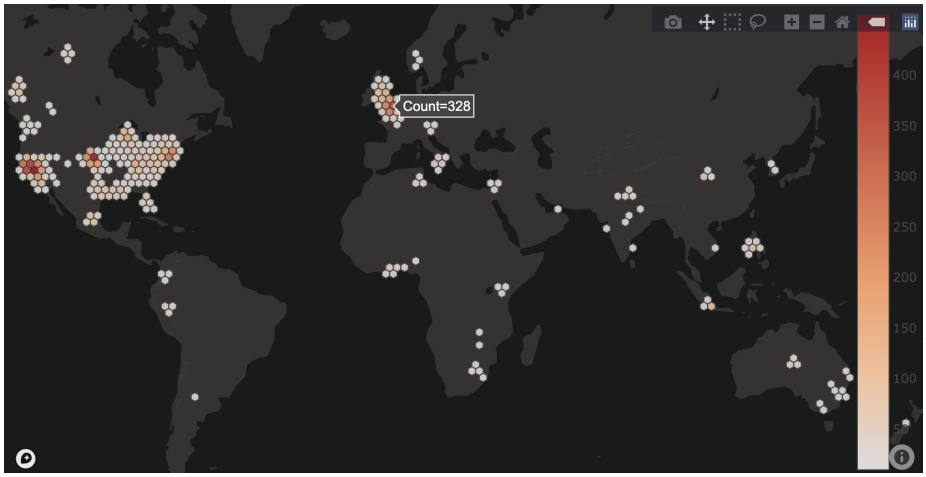


Figure 4.26: Hexbin map for spatial exploration of the Tweets excluding the location type 'Registered user location"

To monitor specific areas with the purpose of detecting emerging flood events, the most recent time period would be selected using the time slider or in the histogram. However, since the interface is not adapted to a real-time continuous stream of Twitter data, the focus is instead on exploring the historical data. Two periods with peaks on the histogram in Figure 4.27 stand out from the rest.

Figure 4.27: Histogram for temporal exploration of the number of Tweets over time showing two peaks

By selecting the first peak in February and March 2017, the number of Tweets is narrowed down to 9,742, i.e. 14.6%. The exploration can continue by for example zooming into a specific area of interest on the scatter map. The Tweets related to floods in the UK are seen in Figure 4.28.

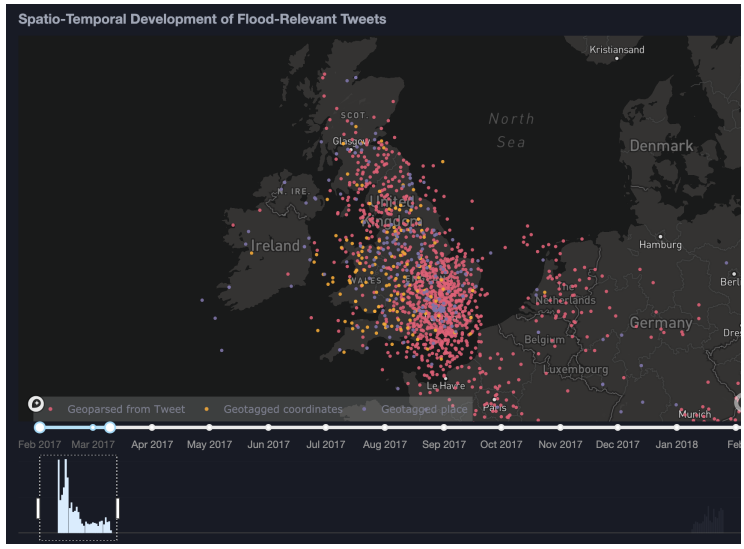


Figure 4.28: Selection of peak in Feb-Mar 2017 on histogram and zoom to the UK area on scatter map

After selecting a time period and zooming in to an area of interest, it is possible to explore further by hovering over the individual Tweets to get details-on-demand. These details include the username, user location, creation time, source, location time and Retweet count. As seen in Figure 4.29, the Tweets can for example be a flood alert update.

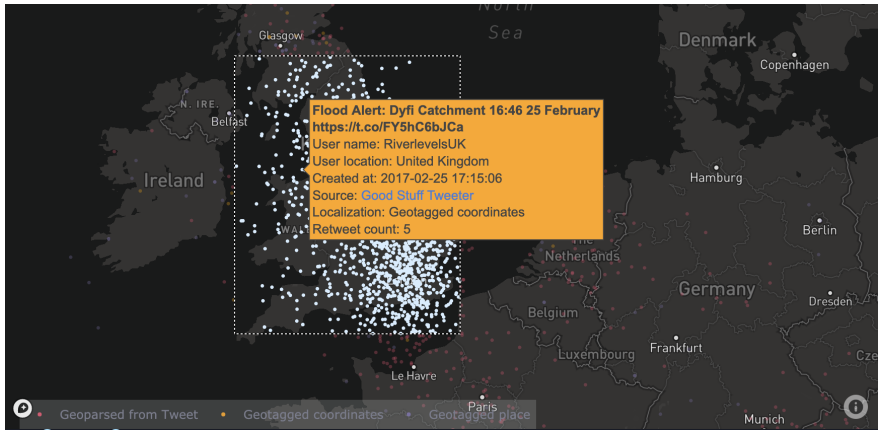


Figure 4.29: Details-on-demand for Tweet in the UK

By selecting a group of Tweets on the scatter map, the content of the treemap with hashtags and the table with Tweets are updated accordingly. It is for example seen from the treemap in Figure 4.30, that a frequent hashtag used in this area and time period is '#stormdoris'.

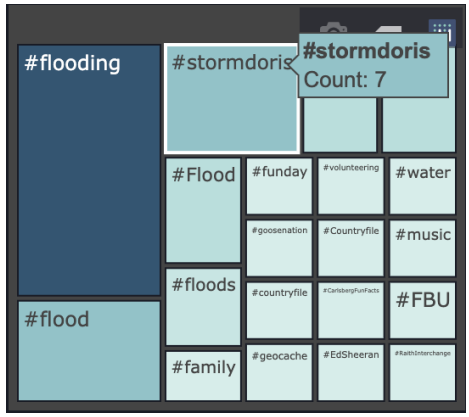


Figure 4.30: Treemap showing the 20 most frequent hashtags used in the selected Tweets in UK

To investigate this storm event and its consequences further, 'doris' is used as a search word in the search box in the configuration pane as shown in Figure 4.31.

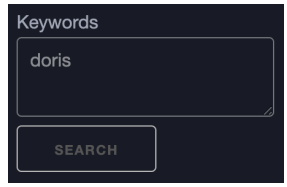


Figure 4.31: Keyword search in the configuration pane using the word 'doris' to select Tweets related to this storm

If all location types are selected in the 'select all' checklist in the configuration pane, the search results in 141 Tweets appearing on the scatter map, where 109 of them are located in the UK. By inspecting the histogram in Figure 4.32, a date that stands out is February 23, indicating that the storm was very intense on this day causing heavier flooding.

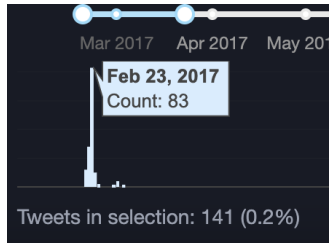


Figure 4.32: Inspection of dates in the histogram showing Tweet count for highest peak on February 23, 2017

The table of Tweets in Figure 4.33 reveals that eyewitnesses seemed highly affected by the event stating 'no wifi, no electric, no heating'. Several of the Tweets are also Retweets from the Guardian asking "Are you affected by Storm Doris and flooding in the UK?" that is shown by hovering in Figure 4.33.

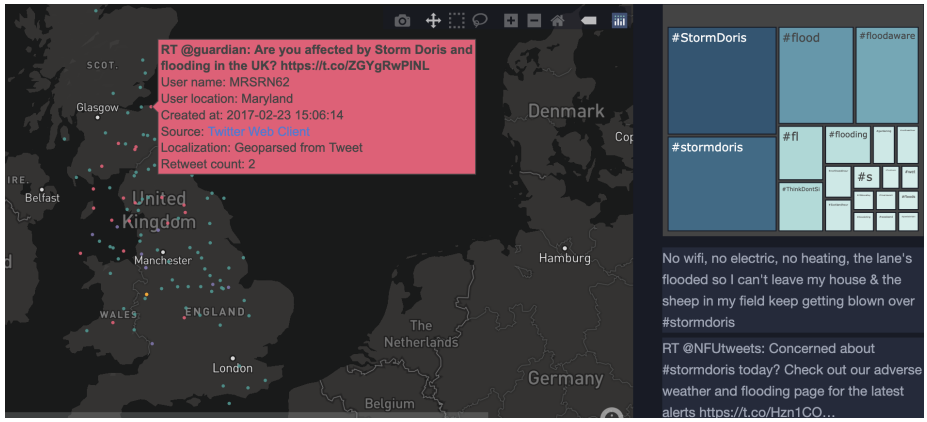


Figure 4.33: Investigation of Tweets related to Storm Doris in scattermap, treemap and scroll-able table

A quick Google search on floods in the UK during 2017 brings forward the very same article referring to the flood event caused by Storm Doris [2]. As seen in Figure 4.34, it was published on February 23, which matches the findings from the temporal exploration.

[2]https://www.theguardian.com/uk-news/2017/feb/23/are-you-affected-by-storm-doris-and-flooding-in-the-uk


Community

Are you affected by Storm Doris and flooding in the UK?

With strong winds and heavy rain brought by Storm Doris we would like to hear from you if you are affected by extreme weather across the UK

Guardian readers

Thu 23 Feb 2017 10.30 GMT

Figure 4.34: Headline and lead paragraph from the Guardian article on Storm Doris and flooding in the UK from February 23, 2017

To further evaluate the findings using the visual interface, a quick exploration was performed using the Global Flood Monitor[3] developed by de Bruijn et al. [11]. The Figure 4.35 shows signs of a large flood event in the UK during the same day, confirming the potential of the project pipeline.

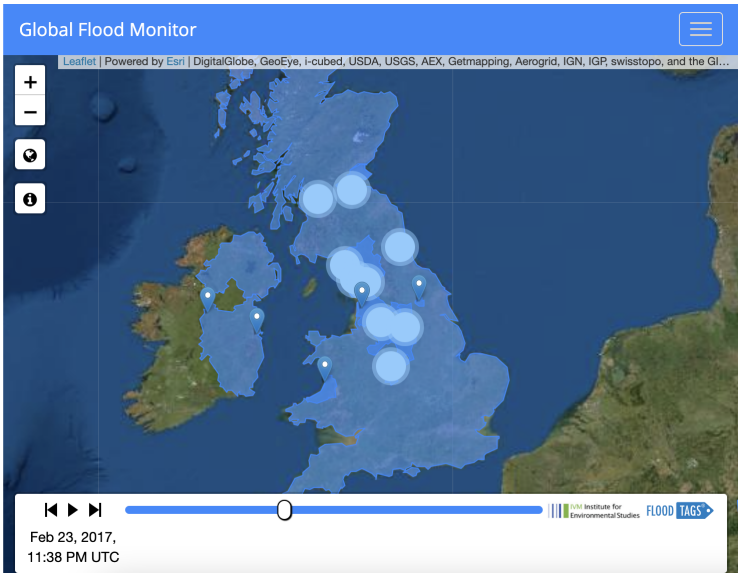


Figure 4.35: Global Flood Monitor showing flood related Tweets in the UK on February 23, 2017

[3]https://www.globalfloodmonitor.org/

Chapter 5

Discussion

This section includes a discussion of the most important results and method choices. Moreover, further research possibilities are considered.

5.1 Results and Methods

The results showed that all classification models performed rather similarly in terms of predicting Tweets to be relevant or not to flood events. The best performance was obtained for the model variant where specific keywords had been removed from the Tweets, though the variant with the original Tweets performed almost as well. This was likely due bias in the model as it learned that keywords such as 'yyc' and 'Queensland' indicated that a Tweet was flood-relevant. Hence, since these words were also included in the test data, a deceptively high performance was obtained. The model variant simply became too specific for the flood events covered by the Tweets in the training data and would not generalise well to other cases. The models trained with this variant are therefore not recommended to use despite their convincing performance results. This problem was also apparent when comparing the predictions made for the unlabelled Tweets by the models of this variant to the models trained on the Tweets where the specific keywords had been removed. The logistic regression model for the variant with original Tweets only found 51% of the unlabelled English Tweets in 2016 to be relevant compared to 73% for the variant with keywords removed.

Thus, more than 20% of the Tweets potentially related to a flood event were excluded.

In general, all models performed with an impressive accuracy between 92-95% which is better than what has been achieved in some of the research papers introduced [7, 11, 24]. The accuracy is a metric that should be carefully considered in evaluating the performance of a machine learning model, especially when the objective is to predict the relevant class. However, this is mainly a problem for imbalanced data sets. Since the training data set used in this project was almost perfectly balanced with a 51/49 split this was therefore not likely to be causing the high accuracy. Nevertheless, the F1 score was also reported to take into account both the precision and recall of the models. The F1 score was also between 92-95% which did not indicate any need for improvements either. If this was not the case, the performance could for example have been improved by including more training data, by tuning the model parameters or by using an ensemble technique to combine multiple weaker models to obtain better results.

There can be several reasons for why the text classifiers achieved such high scores on the performance measures. One reason might concern the nature of this specific task, as it may come across as a less complex task to identify words and phrases related to flood events, compared to capturing the sentiment or stance due to the use of irony and sarcasm. On the contrary, the language from social media messages can in general be concise, informal, and ill-formed, and thereby challenging to process [28]. Besides, the related research performing the same task of detecting flood events from Tweets did not achieve the same convincing results [7, 24, 11]. Another more likely reason for achieving such high performance scores could be related to the choice of data for training and evaluating the classifiers. The labelled training data set was collected using a combination of keyword and location filtering. It might have been better to use training data that was collected using only one filtering method as this could have lead to a more focused classification task. The task can in fact be rather different depending on the initial choice of filtering method.

The benefit of location filtering is that it decreases the number of confusing sequences in the Tweets where flood-related words are used out of context. Some of the Tweets collected with this method might even be completely out of scope by no mentioning of flooding. It then becomes easier for the classifier to separate the relevant Tweets from the non-relevant based on single words, wherefore there is less need for more complex methods to incorporate semantics. The training data for this project included such Tweets that were completely out of context, which can then be a reason why using a CNN or ULMFiT did not outperform the classic models by significant measures.

If the data is instead collected based on keyword filtering, it becomes more relevant to add complexity. The task of excluding the non-relevant Tweets becomes more challenging as more Tweets will contain flood-related words, though not necessarily to describe an on-going flood event. This task therefore demands for the models to understand different meanings of the flood related terms. Hence, the more complex models that use word embeddings to represent the relations between words to better capture semantics are more applicable. When the text classification models in this project were used to classify the unlabelled Tweets, which were only collected using keywords, some false positive examples showed that the models were not able to exclude all Tweets that used flood-related words more figuratively. As this was the case for all models, it can be argued that adding the extra complexity by using deep learning and transfer learning in this project did not improve the classification significantly. This was also clear from the performance of the models that only differed with small measures. Hence, it can be argued in line with the law of parsimony that the simple machine learning models could be preferred in this case in order to ease the understanding and implementation.

Both filtering approaches used for data collection clearly have their pitfalls. Location filtering is limited in its nature by only including the few Tweets that are geotagged, whereas keyword filtering often results in many redundant Tweets and false positive results. The tradeoff therefore concerns whether the aim is to obtain as much data as possible to identify all potential events or to focus on ensuring that the collected Tweets are actually describing an event to act on. In an attempt to have both of these points in mind, the location extraction algorithm was constructed to incorporate different levels of location certainty. The dropdown menu in the visual interface for selection of location type therefore enables the user to show only geotagged Tweets. However, it was seen that the Tweets located using geoparsing or through the user profile location were generally placed in the same areas as the geotagged Tweets (Figure 4.23). This indicates that the Tweets that were not geotagged can also provide important information. Hence, only using location filtering could prevent collection of many potentially relevant and informative Tweets.

In general, more reliability could have been placed in the classifiers for real-world application if they had been trained using a larger and more covering data set. It was chosen to collect a data set that was labelled through a crowdsourcing platform. With crowdsourced data it is important to be aware of the implications on the data quality as this can contain a lot of uncertainty. This could include users uploading false information, intentionally or unintentionally. An alternative for better control of the quality could be to go through the Tweets manually as done by Barker et. al [7] or to construct a method for automatic labelling. The first approach was avoided in this project due being too costly and demanding, whereas the second approach was not part of the desired scope.

This could however be a part of further work to increase the amount of training data, as shown by Feng et al. [24].

The functionality of the interactive visual interface was tested and exemplified through exploration of a subset of English Tweets related to flood events in the UK during 2017-18. From this, it seemed as the interface successfully facilitated a knowledge discovery process with its interactive dynamics and coordinated views. Initially, it provided overview for analysis of the most important spatio-temporal patterns in terms of places and peaking time periods with flood-related Twitter activity. The interface then encouraged further user-driven exploration with its options of zooming and selection, hovering and annotations to obtain details-on-demand, and filtering of the data. The test exploration lead to the identification of a storm that clearly had an important impact in the UK at that time. However, it is not completely known whether it missed to capture other large flood events. Generally, the potential of the visual interface should be demonstrated using more data to evaluate on a broader range of flood events. Optimally, different stakeholders should also be included to evaluate the interface quantitatively and empirically. In this way the interface could be refined after building up evidence for how usable and effective it was. This type of evaluation was though not in focus in this project, but could be included in further work.

The choices of mapping were limited to using scatter, density and hexbin maps including spatial data in terms of coordinates. One could also have utilised spatial data representing a partition of the geographic space to create choropleth maps, where pre-defined areas are colored in proportion to a particular variable to present aggregates for different regions. In this case the spatial data would have been at a very large-scale, but when monitoring a smaller area as a country, it could be beneficial to include a choropleth map to compare relevant geographic regions. It is generally easier to recognise a geographic area by the shape and orientation compared to other areas. On the other hand, one should be aware of the bias introduced by large polygons' data looking more emphasised just because of their size, which is why choropleths should only be used for rates of data and not actual magnitudes. A hexbin map would dismiss this bias by letting each region be represented more equally. Still, the analysis can for both approaches be affected by the way the data is aggregated to areal units, referred to as the modifiable area unit problem [73].

5.2 Future Research

This project had a defined and limited scope but provides occasion for further research in different directions. For the visual interface to be deployed as part of a real weather warning system, it is needed to be adjusted to and tested on even more data. It should also be extended to enable real-time monitoring as is the case for the Global Flood Monitor and OmniSciTweetMap. This would however require more resources in terms of memory, computational power and API's. Further, this relates to one of the main challenges of exploratory visualisations being that if the amount of data is increased then so is the processing time. This can easily disturb the user experience as the views can take longer time to render, which will hinder the flow of exploration [50].

Only textual data was used for flood detection in this project, though there are a lot of potential in including and combining more information sources. The research by Feng et al. [24] did not only include user-generated text in their pluvial flood detection system, but also images from Twitter and Instagram [24]. Some powerful deep learning approaches for image classification exist including the CNN implemented in this project. The same applies for transfer learning that was initially used for image tasks and is part of the reason for the rapid developments in the area of computer vision [10].

A natural further step would therefore be to incorporate the use of images to potentially improve the detection of flood events. Since the VGI can also be extracted from images, it would be possible to include image data as well, though with precautions to ensure GDPR compliance. Presenting real images as a part of the visual interface could improve the user exploration and give impressions from eye-witnesses potentially worth a thousand words. Including even more data sources such as meteorological data in terms of e.g. rainfall records or areas prone to flood events, could also improve the system [7]. Particularly, de Bruijn et al. [13] proposed a multilingual multimodal neural network that could effectively use both textual and hydrological information for flood detection. In addition to incorporating multiple information sources, it could also be beneficial to combine two or more classification models to a hybrid model as this would leverage the benefits from individual models as well as increase the efficiency and accuracy of the results. This idea is specifically relevant by the fact that the classifiers in this project did not give similar predictions for the individual Tweets, which indicates that the individual classifiers capture different aspects.

For the classification models and visual interface to be applied, it is required that people post Tweets during flood events. This consideration was a part of the reason for choosing English Tweets as Twitter is still dominated by the English language with most users from the United States. As the project is conducted in

collaboration with SMHI, further research would however include developing a similar method specifically for the Swedish language and/or pre-defined at-risk areas in Sweden. This would then demand for the Swedish people to start using Twitter to post information during extreme weather events. In this regard, the approach could also be expanded to include data from other social media such as Facebook, Instagram or any other media of potential relevance. Most research in NLP has included English data due to more availability, though more focus has recently been on expanding to other languages. The research by Lin et al. [40] included building a CNN to detect disaster events based on Chinese microblog messages on the platform Weibo by using semantic word vectors trained on Chinese Wikipedia. The same applies for a recent study by Ali et al. [4] that proposed models for event classification trained on data in the Urdu language. Though this research still stressed a need for more data resources. Besides using a different language, the main idea in the research by Ali et al. [4] was to create multi-class event classification of twelve events using convolutional, recurrent and deep neural networks. A promising accuracy of 84% was achieved in extracting and classifying the events in the Urdu language script [4], which substantiates the reason for investigating this multi-class approach as well as the use of other languages.

An idea for further work could be to expand the scope from only concerning floods to other natural disasters such as earthquakes, tsunamis and fires as well as man-made disasters such as riots and terrorist attacks. Instead of being limited to a binary problem of detecting whether or not the Tweets were related to a flood event, a multi-classifier could be developed to also determine the type of event based on the text, images or any other relevant input data. The same principles could be used, but with changes in the specific keywords used for data collection for example. However, it could be a challenge to make the classifier generalise well to all kinds of events and the data collection process would also demand more resources. One could imagine that images of fires from eye-witnesses would be possible for the image classifiers to detect quickly, whereas detecting earthquakes might be more challenging. In cases where the already existing warning systems are sufficient, with well functioning tide and river gauges, fire alarms or seismometers, the need of a supplementary social media detection system is smaller. The purpose could then instead be to create a historic database of such events used for visual exploration and pattern detection.

In general, the use of social media for event detection is not meant to replace other traditional monitoring systems but to be supplementary to improve resilience during extreme weather events. Further research could therefore concern how to integrate existing and new methods to create a more extensive weather warning system that leverage different information sources.

Chapter 6

Conclusion

This thesis project examined the use of Twitter data and AI-based methods in detection and exploration of flood events. For this purpose a pipeline was defined consisting of data collection, text classification, location extraction and data visualisation. For the training and testing of the text classification models, a data set of Tweets labelled for two specific flood events was used. For subsequent evaluation and demonstration of the pipeline, another data set was used which consisted of unlabelled English Tweets collected using flood-related keywords.

Different approaches were taken to classify the Tweets as flood-relevant or not. This included building four text classifiers using the classic algorithms, logistic regression and random forest, deep learning in the form of a CNN and lastly the transfer learning technique ULMFiT. In efforts to prevent bias in the models, keywords specific to the flood events in the training data were removed from the Tweets, which resulted in better performing classification models compared to the models trained on the original Tweets. The results demonstrated accuracies and F1 scores in the range of 92.5% for the random forest classifier to 95.0% for the classifier using ULMFiT. Despite the high performance, it was however seen by the predictions for the unlabelled Tweets, that the models still struggled to exclude all Tweets that used flood-related terms figuratively. As the simple models were not outperformed by significant measures, it could therefore be considered whether the increase in performance obtained by the CNN and ULMFiT make up for their additional complexity.

In order to map the Tweets for visual exploration, an algorithm for extracting locations from the Tweets was constructed. This included relating the Tweets to a location based on their geotagged coordinates if available, else the content of the Tweets through geoparsing or as a last option the registered user location. By applying this algorithm on the unlabelled Tweets, it was found that only 5% of the Tweets were geotagged, meaning that the remaining Tweets could only be located through geoparsing or by the registered user location. However, it was seen that these Tweets were typically located in the exact same areas as the geotagged Tweets, hence excluding non-geotagged Tweets could result in loss of important information.

An interactive visual interface was developed to provide a spatial, temporal and textual exploration and analysis of the Tweets detected as flood-relevant by the classifiers. Exploration of the temporal aspect was enabled through a histogram and time slider, whereas the spatial aspect was enabled through 2D mapping of the Tweets on a world map. Lastly the textual content of the Tweets could be inspected on a tree map for an aggregated view or in a table with the individual Tweets. The views were coordinated such that any selections made in the individual views or configuration pane were reflected in all views. This enabled a potential user to draw conclusions across the three different aspects of the data more efficiently.

A use case was constructed using the unlabelled Tweets to demonstrate the different steps of the pipeline. This proved its potential to detect a flood event, however it would need further improvement to function as a real-time flood monitoring system. This partly includes increasing the reliability of the classifiers e.g. by using more or different data for their training. Thus could also include incorporating additional data sources or combining different classifiers to leverage their individual strengths. Additionally, the visual interface could benefit from being evaluated quantitatively by involving relevant stakeholders, and then be refined accordingly to optimise its usability and effectiveness.

In conclusion, the use of Twitter data for flood event detection is attainable but should only be considered supplementary to other traditional monitoring systems. Hence, further research could concern how to integrate existing and new methods to create a more extensive weather warning system that is potentially able to incorporate additional information sources and manage to identify different kinds of events.

Appendix A

Appendix

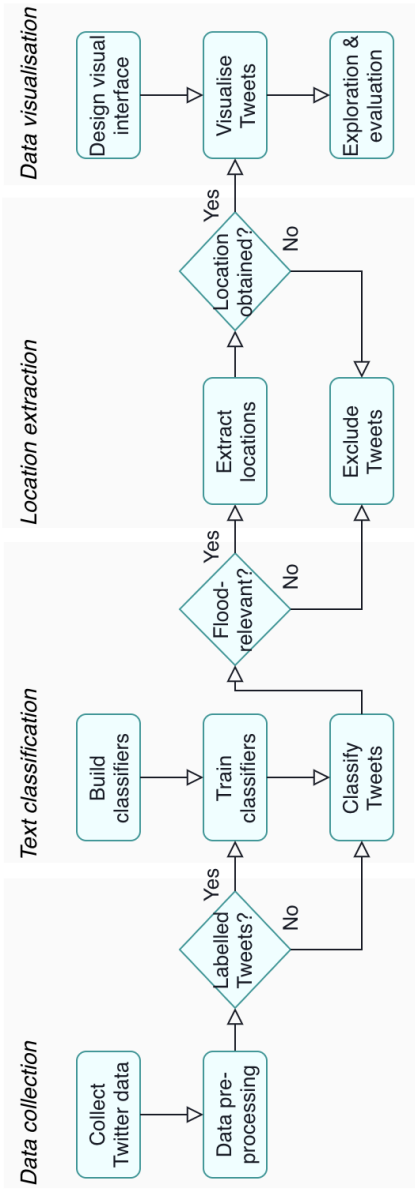


Figure A.1: Enlarged project pipeline

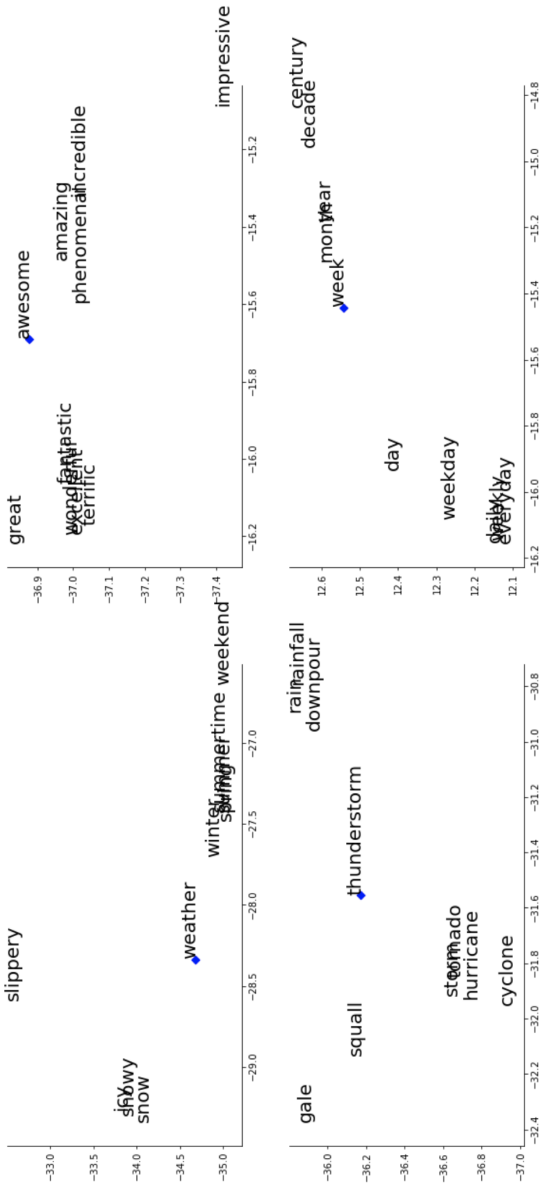


Figure A.2: Enlarged visualisation of the word embedding visualisations using t-SNE on the 10 most similar words to four selected words

Bibliography

[1] Abel, F., Hauff, C., Houben, G.-J., Stronkman, R. & TAO, K. (2012).Twitcident: fighting fire with information from social web streams. Proceedings of the 21st International Conference on World Wide Web. ACM, 305-308.

[2] Research into AI and climate change awarded SEK 6 million. (2021). Retrieved 23 May 2021, from https://liu.se/en/news-item/forskning-om-ai-i-klimatets-tjanst-far-sex-miljoner-kronor

[3] Aigner, W., Miksch, S., Schumann, H., & Tominski, C. (2011). Visualization of time-oriented data. Springer Science & Business Media.

[4] Ali, D., Muhammad, M., Missen, S., Husnain, M. (2021). Multiclass Event Classification from Text, Scientific Programming, vol. 2021, Article ID 6660651, 15 pages, 2021. https://doi.org/10.1155/2021/6660651

[5] Avvenuti M, Cresci S, La Polla MN, et al. (2014). Earthquake emergency management by social sensing. In: The second IEEE international workshop on social and community intelligence, Budapest, 24–28 March 2014. New York: IEEE.

[6] Avvenuti, M., Cimino, M. G. C. A., Cresci, S., Marchetti, A. & Tesconi, M. (2016). A framework for detecting unfolding emergencies using humans as sensors. Springerplus 5.

[7] Barker, J. L. P., & Macleod, C. J. A. (2019). Development of a national-scale real-time Twitter data mining pipeline for social geodata on the potential impacts of flooding on communities. Environmental Modelling and Software, 115(January 2018), 213–227. https://doi.org/10.1016/j.envsoft.2018.11.013

[8] Bishop, M. C. (2006). Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg.

[9] Brownlee, J. (2021). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. Retrieved 14 June 2021, from https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[10] Code, T. (2021). Top 4 Pre-Trained Models for Image Classification | With Python Code. Retrieved 15 June 2021, from https://www.analyticsvidhya.com/blog/2020/08/top-4-pre-trained-models-for-image-classification-with-python-code/

[11] de Bruijn, J.A., de Moel, H., Jongman, B. et al. (2019). A global database of historic and real-time flood events based on social media. Sci Data 6, 311.

[12] de Bruijn, J.A., de Moel, H., Jongman, B. et al. (2017). TAGGS: Grouping Tweets to Improve Global Geoparsing for Disaster Response.

[13] de Bruijn, J.A., de Moel, Weerts, A. H., (2020). Improving the classification of flood tweets with contextual hydrological information in a multimodal neural network.

[14] Bunkley, N. (2008, March). Joseph Juran, 103, Pioneer in Quality Control, Dies. The New York Times.

[15] Cameron, M. A., Power, R., Robinson, B. & Yin, J. (2012). Emergency situation awareness from twitter for crisis management. Proceedings of the 21st International Conference on World Wide Web. ACM, 695-698.

[16] Caragea, C., Silvescu, A. and Tapia, A.H., (2016). Identifying informative messages in disaster events using convolutional neural networks. In Proceedings of the ISCRAM 2016 Conference–Rio de Janeiro, Brazil.

[17] Cheng, T., Wicks, T. (2014). Event Detection using Twitter: A Spatio-Temporal Approach. PLoS ONE 9(6): e97807. https://doi.org/10.1371/journal.pone.0097807

[18] Chowdhury, S. R., Imran, M., Asghar, M. R., Amer-Yahia, S. & Castillo, C. (2013). Tweet4act: Using incident-specific profiles for classifying crisis-related messages. 10th International ISCRAM Conference.

[19] Dong, L., Feng, B., Liu, W. (2020). Density-aware Stratified Sampling for Visualizing Large Volume Geo-Spatial Data. https://dl.acm.org/doi/10.1145/3406971.3406991

[20] Du, L., Buntine, W. & Jin, H. (2010). A segmented topic model based on the two-parameter Poisson-Dirichlet process. Mach Learn 81, 5–19. https://doi.org/10.1007/s10994-010-5197-4

[21] Du, L., Buntine, W. & Jin, H. (2010). A segmented topic model based on the two-parameter Poisson-Dirichlet process. Machine learning, 81, 5-19

[22] Earle, P. S., Bowden, D. C. & Guy, M. (2012). Twitter earthquake detection: earthquake monitoring in a social world. Annals of Geophysics, 54-55.

[23] Eilandera, D., Trambauera P., Wagemakerb J., van Loenena A. (2016). Harvesting social media for generation of near real-time flood maps.

[24] Feng, Y., & Sester, M. (2018). Extraction of pluvial flood relevant volunteered geographic information (VGI) by deep learning from user generated texts and photos. ISPRS International Journal of Geo-Information, 7(2). https://doi.org/10.3390/ijgi7020039

[25] Halterman, A. (2017). Mordecai: Full Text Geoparsing and Event Geocoding. The Journal of Open Source Software. 2. 10.21105/joss.00091.

[26] Hastie, T., Tibshriani, R. & Friedman, J. (2001). The elements of statistical learning: data mining, inference, and prediction. New York: Springer.

[27] Heer, J., & Shneiderman, B. (2012). Interactive Dynamics for Visual Analysis. Communications of the ACM, 55(4), 45–54.

[28] Hernández D.I., Rosso, P. (2017). Chapter 7 - Irony, Sarcasm, and Sentiment Analysis, Editor(s): Federico Alberto Pozzi, Elisabetta Fersini, Enza Messina, Bing Liu, Sentiment Analysis in Social Networks, Morgan Kaufmann, Pages 113-128, ISBN 9780128044124, https://doi.org/10.1016/B978-0-12-804412-4.00007-3.

[29] Howard, J. & Ruder, S. (2018). Universal Language Model Fine-tuning for Text Classification.

[30] Huges, A. L., Peterson, S. & Palen, L. (2014). Social media in emergency management. Issues in Disaster Science and Management: A Critical Dialogue Between Scientists and Emergency Managers. FEMA in Higher Education Program.

[31] Imran, M., Castillo, C., Diaz, D., & Vieweg, S. (2015). Processing social media messages in mass emergency: A survey. ACM Comput. Surv. 47, 4, Article 67 (June 2015), 38 pages. http://dx.doi.org/10.1145/2771588

[32] Kana, M. (2019). Representing text in natural language processing, towardsdatascience.com https://towardsdatascience.com/representing-text-in-natural-language-processing-1eead30e57d8

[33] Keim D. A., Mansmann F., Schneidewind J., Thomas J., Ziegler H. (2008). Visual Analytics: Scope and Challenges. In: Simoff S.J., Böhlen M.H., Mazeika A. (eds) Visual Data Mining. Lecture Notes in Computer Science, vol 4404. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-71080-6_6

[34] Keim, D., Andrienko, G., Fekete, J. D., Görg, C., Kohlhammer, J., & Melançon, G. (2008). Visual analytics: Definition, process, and challenges. In Information visualization (pp. 154-175). Springer, Berlin, Heidelberg.

[35] Keim, D. a, Mansmann, F., Thomas, J., & Keim, D. (2010). Visual Analytics: How Much Visualization and How Much Analytics? ACM SIGKDD Explorations Newsletter, 11(2), 5–8. https://doi.org/10.1145/1809400.1809403

[36] Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[37] Kingma, D. P. & Ba, J. L., (2014). Adam: A method for stochastic optimization.

[38] Kryvasheyeu, Y., Chen, H., Obradovich, N., Moro, E., Van Hentenryck, P., Fowler, J. and Cebrian, M., (2016). Rapid assessment of disaster damage using social media activity. Science Advances, 2(3), p.e1500779.

[39] Kucher, K., & Kerren, A. (2015). Text visualization techniques: Taxonomy, visual survey, and community insights. IEEE Pacific Visualization Symposium, 2015-July, 117–121. https://doi.org/10.1109/PACIFICVIS.2015.7156366

[40] Lin, Z., Jin, H., Robinson, B., & Lin, X. (2016). Towards an accurate social media disaster event detection system based on deep learning and semantic representation. 14th Australasian Data Mining Conference, AusDM, 2, 15–23.

[41] Merity, S., Keskar, N.S., & Socher, R. (2017). Regularizing and optimizing lstm language models. ArXiv.org.

[42] Middleton, S. E., Kordopatis-Zilos, G.,Papadopoulos, S., Kompatsiaris, Y. (2018). Location Extraction from Social Media: Geoparsing, Location Disambiguation, and Geotagging. https://doi.org/10.1145/3202662

[43] Minaee, S.Kalchbrenner, N, Cambria, E. et al. (2021) Deep Learning Based Text Classification: A Comprehensive Review

[44] Munzner, T. (2014). Visualization analysis and design. CRC press.

[45] Nguyen VQ, Anh TN, Yang H-J. (2019). Real-time event detection using recurrent neural network in social sensors. International Journal of Distributed Sensor Networks. https://doi.org/10.1177/1550147719856492

[46] OECD. (2016). Financial Management of Flood Risk, OECD Publishing, Paris, https://doi.org/10.1787/9789264257689-en.

[47] Olteanu, A, Castillo, C, Diaz, F., Vieweg, S. (2014). CrisisLex: A Lexicon for Collecting and Filtering Microblogged Communications in Crises. In Proceedings of the AAAI Conference on Weblogs and Social Media (ICWSM'14). AAAI Press, Ann Arbor, MI, USA.

[48] Olteanu, A., Vieweg, S. & Castillo, C. (2015). What to expect when the unexpected happens: Social media communications across crises. Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing. ACM, 994-1009.

[49] Rao, P. (2019). Transfer Learning in NLP for Tweet Stance Classification, towardsdatascience.com https://towardsdatascience.com/transfer-learning-in-nlp-for-tweet-stance-classification-8ab014da8dde

[50] Roberts, J.C., (2007). State of the Art: Coordinated & Multiple Views in Exploratory Visualization. Fifth international conference on coordinated and multiple views in exploratory visualization (CMV 2007). IEEE.

[51] Robinson, B., Bai, H., Power, R. & Lin, X. (2014). Developing a Sina Weibo incident monitor for disasters. Australasian Language Technology Association Workshop 2014. 59-68

[52] Rodrigues, E., Assunçao, R., Pappa, G. L., Renno, D., & Meira, W, Jr. (2016). Exploring multiple evidence to infer users location in twitter. Neurocomputing, 171, 30–38.

[53] Ruder, S. (2018), NLP's ImageNet moment has arrived, Blogpost Sebastian Ruder, https://ruder.io/nlp-imagenet/

[54] Sakaki, T., Okazaki, M. & Matuso, Y. (2013). Tweet analysis for real-time event detection and earthquake reporting system development. IEEE Transactions on Knowledge and Data Engineering, 25, 919-931.

[55] Saxena, S. (2021). Binary Cross Entropy/Log Loss for Binary Classification. Retrieved 18 June 2021, from https://www.analyticsvidhya.com/blog/2021/03/binary-cross-entropy-log-loss-for-binary-classification/

[56] Schaffer, Jonathan (2015). What Not to Multiply Without Necessity. Australasian Journal of Philosophy. 93 (4): 644–664. https://doi.org/10.1080/00048402.2014.992447

[57] Segel, Edward & Heer, Jeffrey. (2011). Narrative Visualization: Telling Stories with Data. IEEE transactions on visualization and computer graphics. 16. 1139-48. 10.1109/TVCG.2010.179.

[58] Sunasra, M. Performance Metrics for Classification problems in Machine Learning. [online] (2021) (Medium). Available from: https://medium.com/@MohammedS/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b. [Accessed 2021 -06 -08].

[59] Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In IEEE Symposium on Visual Languages, pages 336–343.

[60] Shneiderman, B., & Plaisant, C. (2010). Designing the user interface: Strategies for effective human-computer interaction. Pearson Education India.

[61] Singh, J.P., Dwivedi, Y.K., Rana, N.P. et al. (2019). Event classification and location prediction from tweets during disasters. Ann Oper Res 283, 737–757. https://doi.org/10.1007/s10479-017-2522-3

[62] Thelwall, M. & Stuart, D. (2007). RUOK? Blogging communication technologies during crises. Journal of Computer-Mediated Communication, 12, 523-548.

[63] Thomas, J.J. & Cook, K.A.. (2006). A visual analytics agenda. Computer Graphics and Applications, IEEE. 26. 10- 13. 10.1109/MCG.2006.5.

[64] Thomas, J.J. & Cook, K.A.., (2005). eds., Illuminating the Path: The Research and Development Agenda for Visual Analytics, IEEE CS Press, 2005.

[65] Tufte, Edward R., 1942-. (2001). The visual display of quantitative information. Cheshire, Conn. :Graphics Press.

[66] Van Wijk, J. J. (2005, October). The value of visualization. In VIS 05. IEEE Visualization. (pp. 79-86). IEEE.

[67] Vicinitas. (2021). Social Media Strategy for Twitter - 2018 Research on 100 Million Tweets. Retrieved 17 June 2021, from https://www.vicinitas.io/blog/twitter-social-media-strategy-2018-research-100-million-tweets

[68] Zook, M., Graham, M., Shelton, T. & Gorman, S. (2010). Volunteered geographic information and crowdsourcing disaster relief: A case study of the Haitian earthquake. World Med. Health Policy, 2, 7–33.

[69] Walther M., Kaisser M. (2013). Geo-spatial Event Detection in the Twitter Stream. In: Serdyukov P. et al. (eds) Advances in Information Retrieval. ECIR 2013. Lecture Notes in Computer Science, vol 7814. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-36973-5_30

[70] Watts, J., 2020. Floods, storms and searing heat: 2020 in extreme weather. [online] The Guardian. Available at: <https://www.theguardian.com/environment/2020/dec/30/floods-storms-and-searing-heat-2020-in-extreme-weather> [Accessed 11 March 2021].

[71] WHO Regional Office for Europe, (2017). Protecting health in Europe from climate change: 2017 update. [online] Available at: <https://www.euro.who.int/_data/assets/pdf_file/0004/355792/ProtectingHealthEuropeFromClimateChange.pdf> [Accessed 11 March 2021].

[72] Wilson, A. (2021). A Brief Introduction to Supervised Learning. Retrieved 10 June 2021, from https://towardsdatascience.com/a-brief-introduction-to-supervised-learning-54a3e3932590

[73] Wong, D.W.S. (2004). The Modifiable Areal Unit Problem (MAUP). In: Janelle D.G., Warf B., Hansen K. (eds) WorldMinds: Geographical Perspectives on 100 Problems. Springer, Dordrecht. https://doi.org/10.1007/978-1-4020-2352-1_93