Natural Language Processing methods for short informal text

Jabir Alshehabi Al-Ani
School of Computer Science and Electronic Engineering
University of Essex

A thesis submitted for the degree of
Doctor of Philosophy in Computer Science
April 2020

Abstract

The change in the English language is faster than any time before. Social media is playing a great role in this change as it has become an essential part of peoples social life. Thoughts, ideas, feelings, or even special moments are the main contents of the posts on Twitter and Facebook which are the most popular social media platforms. In this work, we addressed the change in language problem and how it affects the traditional techniques of Natural Language Processing (NLP) for this specific domain. Such a domain is considered to be a challenge for many NLP methods like topic modelling, named entity recognition, and sentiment analysis. We produced novel methods in NLP that target the short text informality.

Our first novel model is in topic modelling for short messy text. The proposed model was inspired by the relation between the word's frequency and the context words frequencies (words surrounding the selected word) over time. This relation had been translated to co-occurrence patterns and stored as word embeddings after being transformed into feature space. The features had been generated from the frequencies of words and context words by our novel Term Frequency-Inverse Context Term Frequency (TF-ICTF) algorithm. TF-ICTF had been derived from the traditional standard algorithm Term Frequency-Inverse Document Frequency (TF-IDF) which did not perform well on short messy text. The proposed model is based on the words probabilities and co-occurrences between words within the short text. Therefore, we named our proposed approach the Probabilistic Relational Supervised Topic Modelling.

The second approach addresses the non-standard entities in short text. We proposed a new model using word patterns embeddings that is generated from the Twitter streamed data. These patterns should include entities that are identified by the state-of-the-art of the named entity recognition (NER) algorithms. We named our approach the Probabilistic Named Entity Recognition (PNER). PNER was trained on the identified entities in the pattern embeddings to identify the non-standard entities format.

Lastly, our Probabilistic co-occurrence Relational Sentiment (PR_ Sentiment) approach proposed to sentimentally classify tweets. We used sentiment patterns detected from the short text tweets. These patterns are structured by an n-gram technique. These n-grams will be detected from sentimentally annotated tweets and labelled accordingly. The dataset that was used is a standard dataset with more than one million annotated tweets. Moreover, the PR_ Sentiment model performs within near real time. The aim of our project is to address the informality and non-standardization in social media short text and produce novel NLP methods. These methods were designed as a novel approach towards generalising the short messy text processing. Therefore, our methods have been tested and compared against several state-of-the-art approaches to show novelty.

Acknowledgements

I would like to show my gratitude to my supervisor Professor Maria Fasli as without her this work will not be completed. Professor Maria, you guided me through my Ph.D. journey with patience and support all the way. You were always understanding and supportive of hard times and your precious advice helped me and still all the way during my research journey.

In addition, I want to thank the Supervisory panel, Dr. Martin Reed and Dr. Ian Daly for their valuable remarks. Likewise, I want to thank my examiners, Professor Sophia Ananiadou and Dr. Spyros Samothrakis for their professional and remarkable feedback. I am also grateful for my country, Iraq who gave me this opportunity and supported me to grant my Ph.D. degree from the University of Essex. In addition, I want to show my appreciation to my family: father, mother, brother, sister, father and mother in law, and brother and sisters in law. Your encouraging and supporting words always helped me.

Finally, to the only person who stood beside me and still, and always telling me that I can do it. She carried most of the burden and without her, this work will not exist. The love of my life and my wife, Samar, you always supported me through hard times and shared good times with me. The words cannot show my appreciation to you and I hope to continue our journey with happiness as we always do.

To my daughters, Jana and Dana, this work is for you and hope you will read this one day and see how much I love you. The look to your faces always makes me forget all of the hard times and fill me with joy and happiness.

Contents

List of Figures

List of Tables

List of algorithms

List of Publications

Al-Ani, J. A., & Fasli, M. (2019). Probabilistic Relational Supervised Topic Modelling using Word Embeddings. In Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018. https://doi.org/10.1109/BigData.2018.8622326.

Chapter 1

Introduction

Human activity produces a huge amount of data uploaded every day on the web especially on social networks, as well as news which starts getting involve in social media trends. The collected data from social networks (media) tempted researchers to invest their knowledge in this huge amount of data trying to gather as much information as they could. These information could be valuable for business people and they look for customer review, for example, or any automated way to reduce time and cost. Thus, studies were made and are on going in social media. Moreover, social media users are sharing their interests or some personal life events, their feelings, and incidents that might attract them making some researchers like Wang et al. [3] consider them as "human sensors".

The amount of published data on social media is not informality free, on the contrary, it is written in the informal text format. We mean by the informal text format text that contains words which do not exist in English dictionaries or exist on non-standard format. These words existance became fast according to Oxford English Dictionary (OED) [4]. Our main objective is to target this informality in text by producing new methods in natural language processing (NLP). Also, producing new NLP models that consider most of the text evenly without giving initial weights or depending on the text itself without considering any auxiliary information like Wikipedia database. The text we are targeting is a messy noisy short text that could be streamed from social media by developers channel which are provided for free in most social media corporation like Twitter and Facebook.

In fact, social media users are growing very fast and has become one of the people's favourite ways to communicate as Baruah discussed in [5] showing the effectiveness of social media in communication. There were more than a billion and a half people with a social media profile until March 2017 [6]. Thus, it became the most interactive way to track any change or trend on any usual event or news item. Most of the famous newspapers and news networks have their channels and official accounts on social media which shows how people are starting to get their news from social media. Text specifically is the easiest and popular short way to express any idea without the need of pictures or videos which might need some preparation before going public. Therefore, our research will target this part of social media data which is classified very messy and noisy.

Twitter has been classified as a noisy short text by many researchers like Derczynski et al. in their researches [7], and [8]. The messiness in short text which is considered as

noise in short text is a result of the informality in social media text which is written by the bloggers (people who are posting on social media) who do not pay attention to formality of text. Formal English language has its restrictions and rules that determine the meaning of each word that implies some changes in words meanings according to context. However, people tend to break these restrictions and use some trending words or abbreviations as an alternative. Thus, language evolves as does everything in the world based on peoples' needs. Because of this, the Oxford English Dictionary (OED)[4] announced that more than 1,000 words, tenses, and sub-entries will be added to the dictionary and published next year. This assessment is made every three months and the new updates will be available in the next quarter. As a result, some old words might disappear and some new ones will take their place and this process is accelerating.

Developing methods for processing short text especially in a noisy environment, is a challenge but it is important for many applications like costumer satisfaction and reviews which falls under the natural language processing umbrella. One of these methods is topic modelling which is classifying documents (long or short) within a specific topic. Thus, finding an autonomous way to classify noisy short text like Twitter text within specific topics (or classes) is a very challenging task, especially when these topics are related to real life subjects like weather, traffic, shopping, etc. The challenge comes from the sentences that are not directly describing the purpose like using names of known places or organizations instead of referring to their adjective or nouns. For example, one person wrote a tweet saying "Thanks m25 I reached to work early today". The "M25" is one of the famous highways in London which is always crowded. Therefore, classifying these kind of tweets under the "Traffic" topic will be an achievement.

Similarly, Named Entity Recognition is considered a difficult task when implemented on Twitter data. Finding the names of the places or organizations needs training on labelled data. Thus, entities in formal language text could be easily spotted even out of context, in contrast to the informal place token which be much more difficult to detect without the context. Therefore, social media could be considered as another dimension for text.

Furthermore, Sentiment Text Analysis for social media is another challenge in text analysis and Natural Language Processing (NLP). Finding text polarity (negative or positive) is another challenge that attracts many researchers. Some of the informal words might be used as negative words on their own but means another thing depending on the context. One of the possible proposed solutions is to track the changes of words by continuously updating how these words occur within context and find a suitable co-occurrence representation between words.

Famous topic modelling algorithms like LDA (Latent Drichlet Alocation) [9] do not perform well in some topic modelling cases when applied to short text for example. Pedrosa et al. [10] stated that the short text topic modelling is a challenging task. Therefore, they focused on enhancing the quality of topics from short text with the use of word co-occurrence. Likewise, Quan et al. [11] discussed the quality of topics that resulted from their proposed self aggregation topic modelling (SATM) proposed model and compare it to other state-of-the-art algorithms. They discussed many difficulties in short text analysis like sparsity in

short text which results in a negative impact with any topic modelling algorithm.

In our research we produce a novel probabilistic model for short text topic modelling which is one of the challenging tasks in Natural Language Processing (NLP). We targeted social media short text and Twitter specifically as it provides the noisy and messy textual environment that we want to challenge our proposed work on. Noisiness and informality in social media short text (such as in Twitter) make it hard to implement modules that are already trained on formal text.

Furthermore, we produced a new method for Named Entity Recognition (NER) that detects entities within short text data. Likewise, the problem discussed in topic modelling, entities like names, places, or organisations for example might not be written in social media in its formal format. Thus, we targeted the informality of these entities representation to enhance the performance of any named entity recogniser algorithm.

Moreover, the sentiment analysis that applied to Twitter data does not show perfect results in some cases due to the same previously mentioned reasons. Thus, our probabilistic sentiment model produces a solution in this domain that makes it possible to find the short text sentiment polarity using probabilistic and statistical methods.

Our overall target is to produce new methods in natural language processing and enhance others to produce better performance, specifically on short informal noisy text. This short text can obviously be seen in enormous growth in social media. Moreover, people tend to use trend terms on social media regardless of the formal words format as social media became a fast and easy way to communicate which creates the challenges that we addressed earlier. Accordingly, our research produced several generalised models on short text regardless of informality and noisiness.

1.1 Research Contribution

Short text analysis is a very challenging task especially for social media as it is short and written in a non-standard format. In addition, the noise in such a text is computationally intensive on the analysis procedure. Therefore, several standard and state-of-the-art algorithms show weak performance when applied on Twitter data.

In our research, a new model was proposed, the Probabilistic Relational Supervised Topic Modelling (PRSTM). The challenging task for the PRSTM is the topics diversity as a wide range of new topics might produced every single day, although, similar topic modelling approaches rely on the standard text to train their model and produce a narrow range of topics. The main concept in PRSTM using machine learning for short text topic modelling to train the model on tweets. The main idea is to derive a mathematical representation for tweets based on words probability with respect to the word context within the whole corpus. This mathematical representation with the a machine learning pipeline are the main elements of the PRSTM to achieve the main goal which is classifying tweets to several topics.

The mathematical representation is a result of applying TF-ICTF (Term Frequency-Inverse Context Text Frequency), which is a new algorithm proposed within the PRSTM, on annotated tweets. It is derived from the standard algorithm TF-IDF(Term Frequency – Inverse Document Frequency). The purpose of the TF-ICTF is to convert terms to a features'

domain depending on the term and the context term frequencies among the whole corpus.

The analysis for each class was produced individually after the tweets had been classified under each distinct topic. A new improvement on Named Entity Recognition was produced on the classifiers results. The places detection within the text is the motivation for this level. Training tweets contain places like streets and facilities which will produce an enhancement on places detection within the text. Following the tweets textual patterns the model is trained to detect non-standard entities within tweets.

Moreover, the sentiment analysis probabilistic model was produced on a each class. As our analysis demonstrates, that each topic has its own measure for negative and positive biasing. Our proposed model was applied on a standard annotated tweets dataset. The proposed method uses words co-occurrence patterns to detect the tweets sentiment polarity of positive, negative or neutral. This approach will not depend on specific words to classify tweets comparing to the standard sentiment approaches.

Finally, the produced three methods are connected by using the same word co-occurrence pattern approach. These patterns were generated using the words embeddings which are a representation of the words distribution of millions of tweets.

1.2 Problem Description

Many researchers targeted topic modelling on long documents as it is easy to extract more information from and to get accurate results together with the standard and formal text [12]. Thus, finding the correct topic for a document became very common especially when the proposed models were trained on standard and reliable data sets such as Wikipedia, and Gazetteer corpus. On the other hand, there is obvious difficulty when text loses formality and standardization as well as being short.

Twitter is a perfect example for short noisy environment with a length of 140 letters per blog post (The tweets' text size had been extended from 140 characters to 280 for the last extension on September 2017), although, it is still very short and challenging. Moreover, the tweets are still considered as part of natural language regardless to the formal language perspective that is changing as we described previously. People use alternative words in language as they are influenced by the audience around them or the groups they tend to talk with so these changes happen fast and disappear fast as new words are always replace the less frequent or less popular words.

Named Entity Recognition is another subtask of Information Extraction (IE) and NLP which suffers from formality problem when applied on social media text. Capitalization of places and names, and lexical errors, are the last concerns when posting on social media. Thus, the context will play a role in the entity recognition for some specific words.

Likewise, Sentiment Analysis which suffers from some words that do not fit with the context and might change the polarity of the meaning from positive to negative and vice versa. Again, the context will decide or contribute in this decision.

The PRSTM proposed novel solutions for the previously mentioned problems, with good performance accuracy together with the near real time processing. The probability and words occurrences are the main basis of PRSTM. In addition, machine learning trained labelled data

for testing and validation purposes as well as some NLP techniques which help to improve the quality of the results.

1.3 Aims and Objectives

The main aim of the research is producing new models of natural language processing to be applied on short noisy text (Twitter) from real time data streams. The proposed models will classify them under related topics on a generalised approach regardless to the topic. Also, generalizing the words representation in feature vector will make an equivalent chance for both formal and informal (non-standard) words format. Also, another approach to recognise non standard format entities. These entities are a result of the changing in English language that we have described in the introduction section. The other challenge that we addressed in our research is sentiment analysis regardless to the non standard and informality in text. Our approaches applied on several dataset : standard and collected using standard methods. The challenges that had been previously stated are the main discussion of this research. Also, devoting the results to propose solutions for real life problems or investing in them.

The research questions for our study were:

1. **What is the main problem that our research tend to address?** The main challenge that we faced in the informality in short text and the new words in the English language. These challenges impact reflected on the performance of the standard methods and tools in NLP. This change in the English language became more frequent which added another challenge over time.

2. **What is the novelty of our research compared to related approaches?** We produced three methods in NLP targeting the informality and the continuous change in the English language. These three methods were produced in Topic Modelling, Named Entity Recognition (NER), and Sentiment Analysis which are three subtasks in NLP. The novelty of our work in the three methods was by using non-traditional approaches on a machine learning pipeline to produce better performance compared to standard methods.

   We used the co-occurrence words patterns together with a new word distribution to predict the topic of each short text (tweet). Word embeddings were generated from millions of tweets to find the co-occurrences between words regardless to these words' informality.

   Likewise, we did not adopt the standard approach of using the parsing tree with the Part of Speech Tagging (POS) to detect the entity. We use the words' co-occurrence patterns to predict the entity as it helped to detect the informal entity. The POS and parsing tree are less performing with informal entities and noisy text. Our proposed approach increase the performance of any standard NER when applied on a noisy environment like Twitter.

   Our proposed sentiment novelty is in using co-occurrence patterns to detect tweets polarity. The informality is also a challenge for the standard approaches as specific

words or emojis will bias the tweet polarity to negative, positive, or neutral. The existence of these words and emojies will make the sentiment analysis approach more selective when compared to our proposed approach.

3. **What are the main steps that will be considered to address this problem?** The main steps of the three methods started by using the pre-processing subtasks from NLP like removing stop words, website links, and special characters (not all of the special characters depending on the task as it will be described in some chapters). The next step will be creating the words distribution and words embeddings from the streamed tweets. Data annotation is another part of the process using either standard techniques or innovative techniques as will be described in the NER chapter.

   Finally, we proposed machine learning supervised pipeline as part of out training, testing, and prediction subtasks. Several evaluation techniques and metrics were used to test and evaluate our proposed methods.

All of the extracted information or the resulted from classification could take part in specific areas if we could put them on the right form and deliver it to the perfect consumer. Therefore, we compared our proposed models with standard and state-of-the-art approaches who had been already used to solve natural language related real life problems. These approaches and many other researches related to our research will be presented in the following chapter.

Chapter 2

Literature Review

2.1 Overview

Natural language processing (NLP) includes many wide range of approaches as many as the language involved in real life problems. Thus, the NLP techniques and methods had been applied on many fields of science helping to facilitate complicated or time consuming problems. For example, costumer service which might need the automatic answering machine, or a costumer service that depends on costumers feedback. These problems needs many hours to be done manually by human. Therefore, NLP could provide several solutions to make it easy and less time consuming.

Several methods and models had been developed for this purpose. Most of these solutions are designed to solve a specific problems. Other approaches designed for generalised used for short or long text (documents) like topic modelling, named entity, chat bots, sentiment analysis, and many other goals. The main challenge that most of the NLP approaches is sparsity and non-standard text format (messy and noisy text). Moreover, applying traditional methods on short text like newspapers titles or Yahoo answers is another challenge. Therefore, several researchers targeted these challenge for the last few years.

In this chapter, we will present several approaches which are related to our three main proposed models: topic modelling, named entity recognition (NER), and sentiment analysis. Our three proposed approaches had been designed as generalised approaches for short messy noisy text. Thus, we will apply these approaches on Twitter data which is more challenging to see what are the limitations and how much we could achieve.

2.2 Topic Modelling

Finding the topic for a document is an easy task for humans due to the knowledge they have on most topics. On the other hand, some humans will not be able to know a text topic even if they read it because it is an article which is not a speciality of theirs. Therefore, topic modelling will depend mainly on the knowledge that is previously known to specify a certain text for a specific topic.

Documents could represent several latent topics which make it challenging task to classify the document under particular topic. Latent topics are the most common way to represent

For each topic k in 1..K, draw a multinomial distribution β_k from symmetric Dirichlet prior η.
For each tweet d in 1..D:

1. Build a label set Λ_d describing the tweet from the deterministic prior Φ
2. Select a multinomial distribution θ_d over the labels Λ_d from symmetric Dirichlet prior α.
3. For each word position i 1..N in tweet d
   a. Draw a label $z_{d,i}$ from label multinomial θ_d
   b. Draw a word $w_{d,i}$ from word multinomial β_z

Figure 2.2: Baysian graphical model of labelled LDA

these kind of documents as several topics could be represented within the same document. There being Several topics in one document is very common in long documents but it is rare in the short ones like newspapers titles and short blogs like in social media similar to Twitter and Facebook.

On the other hand, a short text (document) is messy and noisy due to the lack of informality in writing because the users do not need to worry about text formality. Speed and short ideas are the main factors that create this noisiness in text, but studies target short text as it is very messy and a huge medium which makes it a raw material and hard to be handled manually.

Classifying tweets into certain topics is not an easy task. Using different ways of classification like auxiliary information without relying on the tweets text might help to some extent but not generally. Topic modelling Twitter posts using the LDA (Latent Drichlet Allocation model) was the main contribution of [9] research. LDA was presented as a partially supervised learning model. This algorithm was implemented in a scalable way that maps Twitter feeds into several dimensions. Generalisation of LDA labelled was shown as an example in this research which increased its performance compared to the standard LDA. The labelled version of LDA proposed the existence of a set of labels. These labels were identified by a multinomial distribution. An assumption regarding the documents was that it should use a subset of these labels. As a result, the labels within each will be selected according to the latter assumption. This will make each document work as a magnet for some certain words rather than other documents. Labels in this algorithm are considered as "Topics" that will be discovered by applying the generalized version of LDA. Figure 2.1 shows the graphical model of LDA.

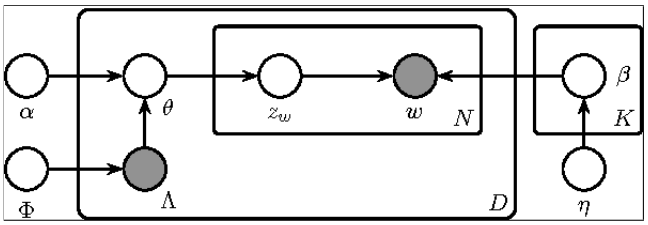


Figure 2.1: Baysian graphical model of Labelled LDA

The following figure 2.2 shows the general process will describes the mechanism of LDA:

Two other proposed methods by Mehrotra et al. [13] aimed to improve LDA topic models: Tweets Pooling and Automatic labelling. This proposition made in a condition of not changing of the LDA basic machinery. Preprocessing steps was made through various pooling schemes this leads to the establishments of a novel method of tweet pooling by hashtags. Again it leads to wide improvements on topic coherence measures across three Twitter datasets (Generic, Specific, and Event) compared to the original LDA and some of the pooling schemes (Basic, Author-Wise, Burst-Score Wise, Temporal, and Hashtag-Based). The results show that the two schemes made an improvement on LDA topic modelling.

Research on short text collected from social media especially Twitter is very popular nowadays. The rich information presented in the social media postings make it a popular topic to study, but challenging at the same time because of the messy and noisy nature of social media posts such as tweets. One study presented a novel approach including a self-aggregation process which is included within topic modelling [11]. The good results shown were for Yahoo answers which are less likely to have noise or be messy as text. Therefore, the results are more favourable when compared against other similar topic models. Likewise, an empirical study [12] addressed topic modelling on social media by using standard topic modelling techniques. The proposed approach was demonstrated on two real world classification problems and obtained good results when the tweets were aggregated depending on the user profile. This might have the drawback of directing most of the process according to the user's interests or finding the users who could fit in the topic. Therefore, it cannot be considered as a generalized approach for topic modelling.

On the other hand, an unsupervised topic modelling scheme was produced by Sridhar [14] to cluster similar tweets. The approach shows better results in topic modelling than the LDA [9]. Chenliang Li et al. [15] included auxiliary embeddings together with the Generalized Polya Urn-Dirichlet Multinomial Mixture(GPU-DMM). GPU-DMM is a mix between a neural network and a GPU sampling process. The proposed model was applied to two standard datasets of Q and A and web snippets which are clean short text data. Compared to the Biterm model [16], the GPU-DMM shows better results but the Biterm model is more likely to be applied to noisy data like Twitter.

Furthermore, many studies used another neural networks approach for tweets clustering or classification. Vakulenko et al. [17] used character-based neural networks to show how it can leverage the performance of tweets clustering. They used SNOW 2014 [18] test dataset for evaluation and another tweets were downloaded for testing. They used annotators to classify the tweets to certain topics. The comparison was between two methods: TweetTerm [18] and Word2Vec [19] as their approach was applied on both methods. They concluded that neural networks performance increased simultaneously with the data size. Although, it does show good performance to some level. On the other hand, the proposed approach capabilities is limited in term of distinguishing between the semantic and syntactic patterns in strings. Additionally, neural networks in general lacks for interpretability which makes it hard to modify and adjust. This feature reflects on the limitation of neural networks in general compared to the standard statistical approach.

Hayashi et al. [20] produced new near real-time Twitter topic detection with the use

of Topic Hijack Filtering technique. They used Japanese twitter stream to feed and collect tweets to the proposed approach. Non-negative Matrix factorization (NMF) [21] was used together with replacing part of the tweets with machine-generated-phrases to reduce the influence of unwanted topics. They used a correlation between Yahoo newspaper headlines with the Twitter feeds that have been published on the same time window. They showed 127 topics predicted correctly using this technique. The clustering of these tweets was based on these specific topics that have been previously determined as news papers headlines which happened to be considered as topics. The results show that 10 topics had been hijacked out of the 127 topics.

Several variations shown in the previously shown topic modelling techniques as some used supervised and others used unsupervised techniques. The supervised shows more obvious and clear topics which are much related to real life like weather, food, or even specific event as described by Bai et al. [22]. On the other side, the unsupervised techniques produces more latent topic or topic which are related to specific keywords. Unsupervised techniques could work on specific topic but after defining the topic after clustering similar documents then linking each cluster to specific topic with a relativity accuracy. This accuracy will depend on how much this topic is related to that topic. Therefore, the supervised techniques are sharper in making decision and much reliable as the validation process could be included within the training process according to Alegre et al. [23]. Supervised LDA is a developed technique of the LDA is one of the examples of the supervised is outperforms the unsupervised techniques on labelled data.

Finally, the previously described works show very good results on topic modelling but we have noticed that the topics were uncorrelated which is one of the drawbacks of the well known algorithms like LDA.

2.3 Semantic Analysis

A Semantic similarity method was used to find whether two tweets are semantically the same or not. The skip-gram proposed algorithm relies on the LSA word similarity model [24]. They use a measurement based on semantic similarity values. Two tweets will be examined and compared against their similarity values and a computation made of the semantic similarity for the possible pairs of trigrams. These trigrams were chosen in two sets using the UMBC Semantic Textual Similarity system. The evaluation methods (F1 score, Precision, and Recall) show acceptable results and address other challenges such as out-of-vocabulary words. In our research we will address this problem that is discussed in several studies.

The researchers contribution [25] is to predict the traffic status for more than an hour before any changes happened. They performed a correlation between the data traffic sensors and the tweets counts of San Francisco Bay, then propose a general optimization framework to extract traffic indicators based on tweet semantics. The correlation seems to be negative for all four time resolutions tested, which indicates that when the current activity is less painful than the average level, the traffic activity on the road network is typically more painful than the average level will be in the near future. In terms of correlation levels, longer

time resolutions such as 12 hours and 24 hours are inclined to have higher absolute values of correlation than shorter ones such as 1 hour and 2 hours. The results shows an improvement in traffic prediction using tweets semantics based on auto regression.

By means of a faceted search, different strategies were proposed to enrich the semantics of individual tweets [26] . Semantic enrichment, adaptive faceted search framework, and evaluation framework summarize the main contribution of this research. The framework allows the enrichment of the extracted entities by external resources. This supports users through their faceted search. Work evaluation shows positive effectiveness of the used strategies that results in significant improvements over keyword search and shows the benefits of those strategies on semantic tweets enrichment using exploited posted links and enhancing the user faceted search.

Moreover, based on the users microblogging activities, the semantics of Twitter posts were enriched. The idea of this work [27] is to find the correlation between Twitter posts and external news resources. This was done through two strategies: URL based and content based strategies. An evaluation was made of the strategies performance with respect to coverage and precision. This evaluation was based on big data fed from three famous media sites: the BBC, CNN and the New York Times. The result clearly show a strong impact on the construction of semantic user profile using the semantics of tweets.

A proposed research by Lecue et al. [28] is the Star City, system scenario designed for the Dublin City, Ireland and Bologna City Italy transportation systems. Humans and machines are considered as sensors that feed the system by various real time traffic status with different data formats. The proposed system serves road traffic congestion supported by semantic web technologies to make analysis and exploration for data sensors easier. It also provides spatio-temporal analysis and diagnosis for traffic status and exploration of contextual information. Traffic status prediction conditions are provided as well by the system. Heterogeneous city data could be described and integrated using a semantic web technology stack that is deeply used in the system.

This work [29] addresses the transportation problems especially road traffic congestion which can be predicted. Demonstrating the challenges, why a semantic web is used, and exposing scalability for those problems would make it easier to design a real time prototype. An adaptive and systematized prototypes had been applied to the Dublin city traffic system as an experiment and it was supported by semantic web technologies. It shows efficiency by working with real and live streams of data. The experiments show accuracy and consistency in road traffic conditions prediction. Unusual traffic patterns were identified using the prediction model after understanding and analyzing the whole traffic patterns of Dublin City. The model was designed based on data mining techniques and regression algorithms correlated with virtually generated real time tweets. A high performance for the system was shown in the results by using it as a web application fed by data streamed from 500 billion traffic observations.

2.4 Twitter Sentiment Analysis

Text sentiment analysis determines its orientation by being classified traditionally into three main types negative, positive or neutral. Several researchers developed methods in the same area and the tweets were a perfect environment for their implementation.

A lexicon adaptation approach was proposed by [30] using the contextual semantics of words. By capturing the words' context, they were able to recognize the changes of the words' orientation and strength. SentiCircle [31] was used (which is a previous work for the same researchers) to determine the contextual semantics of a word from its co-occurrence patterns with other words. These words were in a given collection of tweets. The name of SentiCircle refers to a geometrical circle to represent how words are categorized to four patterns (positive, negative, very positive, and very negative).

The unsupervised sentiment analysis approach [32] was implemented on Twitter feeds. This novel approach classifies Twitter posts according to their sentiment polarity by extracting a vector of weighted nodes. This vector was extracted from the graph of Wordnet [33] [32] then these weights were used in SentiWordnet to estimate the polarity. SentiWordnet is a lexical resource based on Wordnet. The proposed method uses real values $[-1, 1]$ to describe the weights. The neutrality is measured by the nearest value to zero. As a result, a new method was proposed but it has many concerns regarding polarity that seems not to be so clear in describing the words.

Another novel approach of adding semantics was presented in this research [31]. The added semantics were considered as additional features combined with the training set for a sentiment analysis. Sentiments were discriminated as positive and negative in tweets and this classification was enhanced using the value of the used semantic features. Correlation of the representative concept measurement was made every time after each entity extraction from tweets with the two kinds of sentiments. Three different Twitter datasets were chosen to predict sentiment from. The results show an increase in the F harmonic accuracy score for identifying both negative and positive sentiment. Semantic features produce better results with negative and positive sentiment against a comparison made by an approach based on sentiment-bearing topic analysis.

A study of fine grained sentiment analysis done by Chen et al. [34] on financial and news microblogs datasets which had been collected from Twitter streams and news headlines from different resources. These data went through several pre-processing stages to create the the data corpus. One of these important stages was data annotation which was undertaken by financial experts using a web platform. The annotation process produced a value (score) to each microblog between 1 and -1. Two classification baselines were used, the first one was Random selection and the second was the Senti-Circle lexicon classifier [30]that we have described in the second paragraph in this section. This study is the task five from SemEval-2017[35] as several participants used several machine learning techniques and word embeddings. Regression and deep learning as the machine learning techniques and Glove [36] and Word2vec [19] as the word embeddings. Using these tools several lexical and sentiment features were extracted from the corpus. Cosine similarity and another modification was used for evaluation reaching to the result that the best technique to be used

was hybrid consists of combining deep learning with lexical features. There are several disadvantages or drawbacks on this model starting from the evaluation technique that they suggested to improve in the future and the other one is the static concept that they rely on with no suggestion for a real-time process.

Another study was carried out by Nakov et al. [37] on development and evaluation on SemVal2013 [38]. This study proposed a new approach on sentiment analysis by creating a large contextual corpus consists of tweets, short messages, and other resources. It also includes a special test on sarcastic tweets and live journal messages. Two tasks were included in SemVal2013, the first one is to disclose the ambiguity of simple word polarity by using contextual perspective. The proposed model to solve this task should produce one of three categories to each word which is either positive, negative, or neutral. The second task is to classify any message either positive, negative, or neutral and deciding between positive or negative if the message includes both polarities. Data was collected from the previously mentioned resources and was annotated using the Amazon Mechanical Turk [39]. Several features were extracted like n-grams and part-of-speech (POS) tags and passed to the Support Vector Machine classifier. The base line was decided to be on three categories. Majority class, Target's majority class, and lexicon based are the three baselines that was suggested by either the most frequent class, frequent class with additional weight of frequent terms, or using lexicon words. On the other hand, SVM was applied on either unigrams or both unigrams and bigrams features. The results of the authors were better than the contributing teams in the SemVal2013 on both task with a good results on the sarcastic special test. The overall was good with some areas that they never covered like real-time, using other metrics not just f-score, and other classifiers to show what are the differences between the classification performance. Also, the reason of using SVM only was not shown in the proposed research.

Finally, many of the previously mentioned studies are focusing on twitter and small text sentiment analysis. The challenges that they discussed are related to the perfect measurement, annotation, real-time, unsupervised, and ill-formed terms in noisy text. We have addressed most of these challenges in our sentiment proposed model.

2.5 Event Detection and Prediction

Event detection from social media text is one of the application that are related to Natural Language Processing (NLP) methods. Two of the models that we have developed in our thesis are related directly to this field which are the Named Entity Recognition (NER) and Topic Modelling. One of the classes produced by topic modelling is "Traffic" which is linked to the events that are reported by people on social media. Location, organization, street, or any place detected within text is the job of NER. Therefore, we made a statistical study of the related terms to traffic as an example of the importance of that field of study in Appendix A. Several studies in this field show how NLP methods and models could be used for event prediction and detection.

A prediction model for a previously planned event was produced by [40]. A solution was integrated by this model to predict and visualize non-recurring traffic congestion in urban environments. Planned special events like soccer games or concert were taken into

consideration in this research. Two machine learning approaches (Category-Based modelling Approach (CBMA) and Inbound-Based Prediction Approach (IBPA)) were applied to predict the outbound traffic. The data provided in this model was the delay time of traffic in the congestion near the event during the whole day. The prediction shows good results that were confirmed by a visualization tool. The research area was the inner city of Cologne in Germany and for a period of seven months of data time frames.

On the other hand, the proposed work [41] is another prediction and traffic monitoring approach that had been applied on the traffic in Jakarta. This algorithm mechanism is based on extracting traffic information from multi Twitter channels and then presenting these data on the map for public use. The analysis shows that ungrammatical analysis was used on Twitter so they used some language techniques to solve it. However, a comparison was made between the CCTV cameras in the streets with the information harvested from Twitter and shows accuracy in determining the incidents. Several improvements to the system were suggested. The first one is adding an analysis component so that the context of the message can be understood, even if the tweets are ungrammatical. The second possible improvement is to develop interface usability that might improve the system optimization.

Extracting information from social media to enhance traffic prediction [41] is the idea of another model of traffic flow that was presented but under special event conditions. a short-term traffic flow prediction model was developed which integrated with features extracted from tweets to predict the traffic flow before a sport event. Features and semantic features were included then the whole system performance was tested and compared using four different regression methods which were (Auto-regressive, Neural networks, Support vector, and k-nearest neighbour). These tests show improvements that resulted from the inclusion of Twitter information on the prediction model and its computational ability.

Another estimation approach [42] was used to infer value parameters from complex data. The applied approach uses the “Kalman Filter”. This filter takes data from different resources such as the observations on previous journeys and measurements as well as dynamic real time data and waiting traffic time, vehicles speeds produced from the GPS. All of this data was combined with information extracted from social media relating to the weather, waiting for buses, traffic congestion, and special events that people publish to enhance the prediction of the arrival time for public transportation vehicles. A general framework was presented with the social media combined with the Kalman filter models for the estimation of bus arrival time. The ability of this filter depends on noisy measurements. All of this work should be done in a condition of using trusted messages on social media from trusted sources. Integration of robustness data from social media with the Kalman model would be considered as a future work for this model.

2.6 Named Entity Recognition and Word Matching

Named Entity Recognition (NER) [43] is one of the natural language processing and information retrieval methods. The input for any NER method is text and the output will be the identified entities from this text such as place, name, organisation, or even a street name. The traditional (classical) way of determining the named entity is by using the sentences parsing

tree [44]. The parsing tree of any sentence will determine the grammatical structure of this sentence, for example, by determining the adjectives, pronouns, nouns, verbs, etc. All of these sentence contents can be identified using Part Of Speech (POS) tagging [45] (Refer to Appendix C for more information about the list of tags).

Ritter et al. [46] produced their novel approach in NER. The proposed approach named T_ NER as the "T" refers to "Twitter". Their proposed work on Twitter shows enhancement on the F1-score result compared to Stanford NER [47], although, they have used Labelled LDA [48] as part of their model to get training data as well as part of speech tagging.

Likewise, Derczynski et al. [8] applied their proposed model to tweets which had been described as a challenging task in their research. They made an empirical study of some of the important NER models like Stanford NER [47], and ANNIE (A Nearly-New Information Extraction system) NER by GATE [1]. Also, they produced a new entity disambiguation dataset. This dataset had been created using Twitter text that passed through a standard natural language processing pipeline. This pipeline started from pre-processing, which is the cleaning process for Twitter text, attempting to identifying entities using the state-of-the-art NERs. As a result, this study shows the differences in performance between the state-of-the-art and how the pipeline processes affects the accuracy.

On the other hand, there are many studies on using auxiliary information on NER. This information was either taken from a corpus like Wikipedia as in the research that had been produced by Hachey et al. [49], or by using DBpedia like in Named Entity rEcognition and Linking (NEEL) [50] which is a competition to find the most entities in a Twitter using the previously mentioned dataset.

Nothman et al. [51] produced a new approach of learning named entities from different languages using silver-standard annotation training. They used manually labelled Wikipedia articles for training then projecting their classification approach on other articles text. The proposed approach used silver-standard and it outperforms other models that used golden-standard. Automatic annotations of entities in this approach shows better performance than the golden-standard. Case sensitivity was taken into consideration as some of these entities were mentioned in non-capitalised format. The non-standard format entities was not discussed in this study as all of the language is in standard form in all Wikipedia corpus. The results shows many entities were detected in nine languages against two other approaches uses the Wikipedia corpus.

On the contrary, Han and Baldwin [52] targeted the ill-formed (non-standard) entities in Twitter using Lexical Normalisation. They used morphophonemic similarity then using word and context similarity to exploiter the correct replacement word. There proposed model does not require data annotation and was applied on SMS corpus and a Twitter dataset.

Han et al. [53] produced an automatically constructed dictionary for the lexical variants of known words (ill-formed) in social media especially twitter. They used 10 million tweets to generate this dictionary. The main concepts of their approach is to use context and string similarity to exploit similar words in form of pairs. Many words distributions were used to evaluate their approach as well as many string similarity approaches. The experimental results shows good performance compared to state-of-the-art studies. Their proposed model

did not discuss if it is applicable words related to specific entities. Therefore it is hard to predict if their work could detect named entities compared to our approach which uses NER within the pipeline to produce these entities.

Finally, several methods had been developed as mentioned before for short text, especially Twitter just to show how good the performance could be when applied to this messy noisy environment. This shows how challenging this subtask (NER) is when applied to tweets.

2.7 Other Short Text and Twitter Analysis Researches

A novel approach [54] was interested in finding small scale incidents such as fires or car crashes within tweets by analyzing the information of the tweet's message itself. They did not take into account the metadata of tweets (e.g. location, longitude and latitude). This idea was presented as a rapid prototype framework to extract these incidents. Multi-label classification had been used in this framework and their approach shows the ability to detect multiple labels in the tweets feeds. They just focus on four different types of labels (fire, shooting, crash, and injury) which orient their work to a specific kind of information in tweets. The selection of the four classes was based on their research made on Seattle real time calls datasets then applying the framework on 7.5 million tweets collected through Twitter API.

The system [55] targets earthquake events (large scale events) prediction and determining location as well depending on different data feeds. Earthquake sensors are not enough to discover these events before they happen. A Twitter classification tool had been used depending on keywords it tweets to detect the target event. An automated classification tweets model was presented by preparing positive and negative examples as training sets. Also a probabilistic spatio-temporal model was produced to discover not just the event but the event location as well. Each Twitter user was considered as a sensor and a particular filter was used for location estimation. An earthquake reporting system was developed to use in Japan. The Japan Meteorological Agency (JMA) confirmed that more than 3 earthquakes were detected through tweets monitoring. The proposed system detects earthquakes promptly and much faster than JMA broadcast announcements.

Automated Gauge feedback was presented by [56] to align tweets with related events. Two different machine learning techniques were used: Naive Bayes for classification and Proximity Based for clustering. The tweets dataset was collected during the Extended Semantic Web Conference. They used the Zemanta API to extract relevant topics from tweets. Extracted features were used in both of these techniques and their effect on the alignment process was explored. For each event, precision and recall measures were used to evaluate the performance of the proposed approach which showed good results.

Another study looked at large-scale events in the east of Japan. This research [57] addresses the problem of decision making of people returning to their home after the collected Twitter data during the Great East of Japan Earthquake. Behavioural data were extracted first from tweets then an identification for the user's returning-home behaviour was determined by using support vector machines. The second step of behavioural Analysis is creating verbal and non-verbal exploratory factors, these factors uses tweet data and geo-tag

data respectively. Finally they stated the advantages from Twitter data. First, rare events and incidents can be found within tweets. In addition, it is open data and available to everyone, as well as the availability of GPS information within. The disadvantages vary from the lack of the availability of all the searched data to the ambiguity of users' posts.

Twitter analysis could go further to be more beneficial in saving human lives. One study benefits from the Twitter geo-location System and invests it in Public Health Applications [58] . They demonstrates the system utility for improving influenza surveillance. First, they deduce the geo-location from the structured object provided by the user's message. The location might be identified through place of object of tweet, coordinates from tweets, location from user profile, or content based geo-location. The Carmen model identifies the locations using the latter's stated methods then locates the word "influenza" within text and determines the location for the trends related to this subject. Finally, the results were compared against government statistics for evaluation.

Theme tracking in Twitter is another idea that proposes a Dynamic Query Expansion (DQE) model [59]. This model characterizes the uniformity of a theme among various entities. The tracked theme in this research is "civil unrest" and this theme was chosen across 8 Latin American countries. Finally, the new optimization algorithm estimates the weight for minimizing relationships by the Kullback-Leibler divergence.

Another method of analysing tweets has been presented to provide geo-located places from non-geo tagged tweets [60]. These tweets indicate the venue's name and location simultaneously. The Venue Inference for Tweets (VIT) method is the proposed method. A heterogeneous social network was constructed in order to analyse social relations and leverage limited geographic data to help in geo-location estimation from tweets. The system showed that performance was improved over a simple text-only model. They used a single classifier that was trained for the probability of a tweet and a geo-located venue estimation had been linked. The performance of several types inferred social relation features and geographic features that are embedded in a social network shows high accuracy for identifying venue geo-locations from tweets by checking the linkage between tweets and venues.

2.8 Accuracy Metrics for Machine Learning

Measuring the performance of any proposed work depends on standard metrics and measurements. These standards shows many characteristics of the tested approach depending on the matching output to the target. There are several types of accuracies depending on the machine learning (ML) algorithms.

The simplest type of metrics is the "Accuracy". The accuracy for any machine learning output is to measure the ratio of the correct output observations to the total number of input observations as shown in the following equation 2.1 as described by [61]

$$Accuracy = \frac{Number\ of\ correct\ predicted\ observations}{Number\ of\ total\ observations} \tag{2.1}$$

This accuracy works when there are an even number of elements in each class (i.e, balanced classes). On the other hand, when the accuracy is to evaluate a classifier result

depending a multiple classes accuracy, a confusion matrix will be used, as described briefly by Fawcett [62]. The confusion matrix defined the results with a more detailed perspective. The output of each classifier can be divided to four types:

1. **True Positive (TP)** where any predicted output is one and the actual output is one as well.
2. **True Negative (TN)** where the predicted output is zero and the actual output is zero as well.
3. **False Positive (FP)** where the predicted output is one and the actual output is zero.
4. **False Negative (FN)** where the predicted output is zero and the actual output is one.

Therefore, the accuracy on this basis will be as in the following equation 2.2:

$$Accuracy = \frac{TP+FN}{P+N} \tag{2.2}$$

where P refers to positive and N refers to negative and their addition refers to the total number of samples.

Also, there are a couple of metrics that show there could be defined as the ratio for the correctly relevant observations, which is called Precision [63][61]. The following equation 2.3 shows how the precision is calculated:

$$Precision = \frac{TP}{TP+FP} \tag{2.3}$$

The other metric that measures the ratio of the correctly retrieved observations is called Recall [63][61]. The recall is calculated using the following equation 2.4:

$$Recall = \frac{TP}{TP+FN} \tag{2.4}$$

The other metric that we we will describe in this section is the F1-score [64] that could describe the performance of the ML algorithm output. This metric is defined as the harmonic between precision and recall. The F-score can be calculated using the following equation 2.5:

$$F1_Score = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}} \tag{2.5}$$

Cosine similarity [65] for text classification is one of the metrics that is used to produce the distance weight between two input texts. The big the cosine value is, the angle between the two documents will be smaller resulting to classify both documents or texts in the same class and visa versa. The two documents should be first converted to feature vectors using one of the traditional techniques, for example, Term Frequency-Inverse Document Frequency (TF-IDF). These two vectors will be passed to the cosine similarity equation to produce the similarity value. Equation 2.6 shows the calculation of this in equation provided by Li and Han [65]:

$$Cosine(X,Y) = \frac{\sum_{i=1}^{n} X_i \times Y_i}{\sqrt{\sum_{i=1}^{n} X_i^2} \times \sqrt{\sum_{i=1}^{n} Y_i^2}} \tag{2.6}$$

As X and Y are the two documents with a features vector of n features and i as the component. The result will show the score of similarity as described in the previous paragraph.

Jaccard similarity [66] is another measure that is also used between two documents. Jaccard formula is very simple and easy to implement as shown in equation 2.7 produced by Hamers et al. [66].

$$Jaccard(X,Y) = \frac{\mid X \cap Y \mid}{\mid X \cup Y \mid} \tag{2.7}$$

As the numerator of the equation is the magnitude of the intersection of the two documents X and Y and the denominator is the magnitude of the union of the two documents.

Likewise, Perplexity measurement produced by Brown et al. [67] to find the similarity between two documents based on words. The value of perplexity depends mainly on the probability distribution of words within the sentence. It is calculated by equation 2.8 produced by Brown et al. [67]

$$Perp(W_{i=1}^{n}) = \sqrt[n]{\frac{1}{P(w_1, w_2, ..., w_n)}} \tag{2.8}$$

As n is the number of words in the compared documents and w is the word with index i in documents that are represented by the set W. While P is the probability of all of the words in documents.

There are several metrics that could be described in this section but we have mentioned the most popular and most used ones in ML. Most of other ML metrics are either derived from the previously described metrics or work on the same concept.

Chapter 3

Probabilistic Relational Topic Modelling

3.1 Introduction

This chapter introduces the proposed two models for short text topic modelling which consist of two main parts: the first is tweets streaming, text cleaning, and creating embeddings. Then, the second part that produces tweets classes, which consists of word matching, a feature vector generator, and then classification or clustering. The two models have been evaluated using standard metrics and against state-of-the-art models. However, some standard methods have been used as part of these components, like word matching and classification algorithms to produce better results.

Cleaning the tweets' text is the first stage which aggregated the streamed and saved tweets for over a year, then created the word embeddings afterwards. Our version of word embeddings were generated with the n-gram technique with a three-words window size. These embeddings will help in extracting words co-occurrence patterns that will then be passed to the word matching stage. The main purpose of word matching is to reduce the similar words representation at the feature level. This reduction increased the accuracy level and reduced sparsity, which most of the topic modelling models suffer from, for example Biterm Topic modelling [68] and word2vec [19]. Additionally, a keyword (Anchor) detection stage depending on word-context frequency change over time is included in the word matching stage to reduce the number of words to be processed which leads to another reduction in the sparsity problem. The resulting words co-occurrence patterns are then passed to the classification stage as feature vectors through the Term Frequency - Inverse Context Term Frequency (TF-ICTF) stage which is a developed version of the famous Term Frequency - Inverse Document Frequency (TF-IDF) algorithm.

The classification stage produced tweets classes related to real life topics such as traffic, and weather. Tweets that are related to a specific topic do not necessarily contain the word weather, for example, to be considered within the weather topic. Therefore, this kind of co-occurrences was one of the motivations to investigate more about words co-occurrences within the same topic. Thus, we used annotated tweets to train and validate our proposed model, which is named the Probabilistic Relational Supervised Topic modelling

(PRSTM) which depends mainly on the co-occurrences between the words within the same context (words co-occurrence) and words co-occurrence probability over time. Short text, as described in the introduction chapter, became a trend in social media, and the text should be related to a specific topic in general if the text is in a formal format. The real challenge that we faced was text informality, which is considered as noise if not defined or classified to a topic, and how changes in language could affect topic modelling accuracy.

3.2 Data Sets Collection and Streaming

In order to undertake the research and analysis reported in this thesis, a number of datasets were collected and used. These datasets were collected by targeting a number of topics as described in section 3.1. The datasets that will be presented in this section will be all about the mentioned topics. Three datasets have been collected and streamed using the Twitter API (Application Program Interface) using a double authentication key. This API called "Listener" from a "Tweepy" package in the Python programming language [69] stores the streamed tweets in CSV (comma-separated value) files. This type of file has no limitation in size and saves the data in a form of spreadsheet. Refer to figure 3.1 that shows a sample of stored tweets in a CSV file.

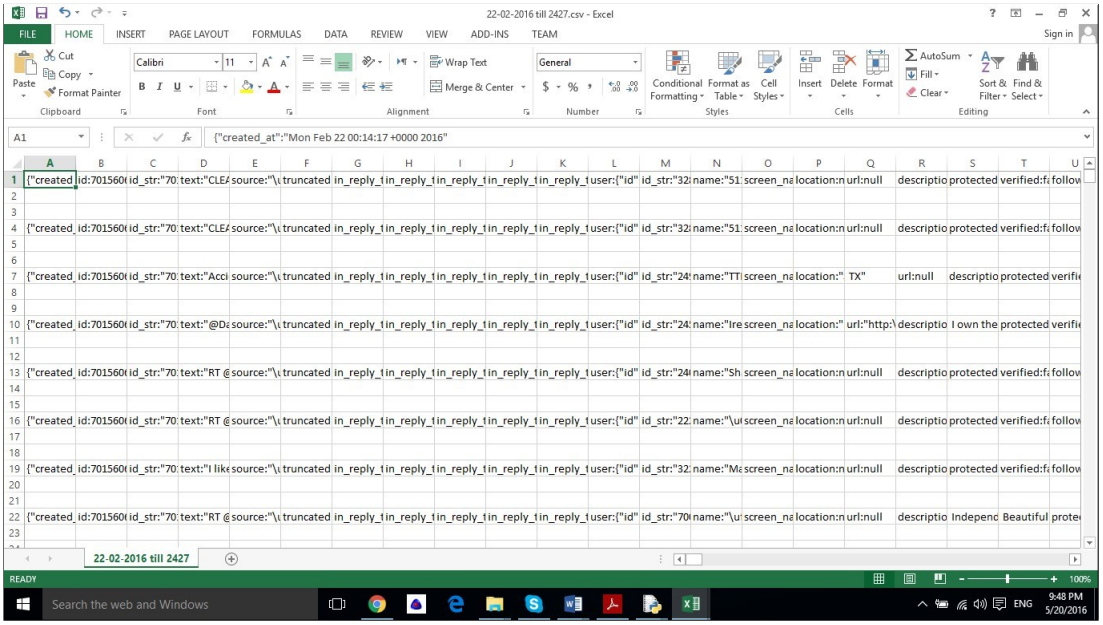


Figure 3.1: Screen shot of streamed tweets

The tweet data format consists of several parts (as shown in figure 3.1) like (Creation date, User ID, Date of account creation, geo-location, URL for any Contents, text ,....., etc). The data that has been used are the creation date of tweet, Text (which is the main body of the tweet) and, geo-location (the geographical location of the tweet if available). Furthermore, text is selected and passed to the pre-processing step where the data is cleaned. The other mentioned selected parts of the tweet are processed as well, but with little effort as it is structured data, and a very simple single step in the pre-processing step puts the data in the right format. This API cannot be used until one registers as a Twitter developer. This process

allows streaming of 1% of the actual Twitter data for free. Morstatter et al. [70] discuss the idea of sufficiency of the free tweets against the paid ones that called Firehose for research purposes and they stated that it is reasonable to work on the first type of streamed tweets.

Thus, two types of Twitter datasets (either using geo-location, or using keywords) have been collected for this research for nearly two years as well as two other datasets from a reliable government body. Accordingly, London was chosen as the area of the research case study from where most of the following datasets were collected. The datasets are as follows:

1. The first Twitter dataset was collected in general depending on some keywords related to traffic (e.g. accident, crash, traffic, congestion). This type of tweet was collected from all over the world without any specification apart from the words being related to traffic, and the time that it was created. It was collected from the 17th. February 2016 until 19th. April 2016, with a total size of 40.9 GB (gigabytes) and around 20 million tweets in total. Most of these tweets were written in the English language. This dataset had been used to show the relation between the social media text and real events. Appendix A shows the relation between traffic detected events from the Twitter dataset and the TFL (Transport for London) accidents dataset that will be described as the fifth dataset in this section.

2. The second Twitter dataset was collected within the geographic location of the city of London from (51.263117 latitude / -0.659189 longitude) in the south west, to (51.700991 latitude / 0.302114 longitude) in the north east. These data collected from the 20th of April 2016 and until the 3rd of December 2018, produced a total number of 146.2 million tweets and a size of 278.6 GB. Most of these tweets were posted in English but not all of them were related to traffic events.

3. The third Twitter dataset comprises some randomly selected tweets from the second dataset. Thirty thousand such tweets were annotated using Amazon Mechanical Turk [39] into several topics as shown in figure 3.2. The annotation was made by dozens of annotators with the condition that they had a linguistic background, and they were given specific instructions that they should follow for annotation. This dataset was then used on the supervised topic modelling that will be described in detail in the results section.

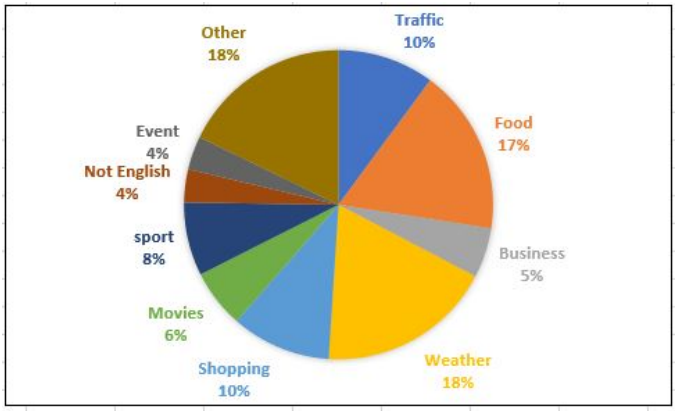


Figure 3.2: Annotated tweets classes.

4. The fourth dataset consisted of street names for the London area collected from OSM (Open Street Map) [71]. The data contains streets and place names in London. Some of these names (entities) might be similar to a person entity (person name) which makes it a case for study in natural language processing, especially named entity recognition. These data are used later in the PNER (Probabilistic Named Entity Recognition) chapter.

5. The final dataset was the TFL (Transport for London) accidents dataset which provides data for all the accidents that happened in London, including location, description, and time of the accident.

In the following sections two proposed models will be presented. The first one uses the supervised machine learning in topic classification with the annotated dataset. This model is named Probabilistic Relational Supervised Topic modelling (PRSTM). The results were evaluated against some state-of-the-art models as well as using cross validation and accuracy measures. The second model is an unsupervised model that clusters similar tweets using unsupervised machine learning. A purity metric [72] was used to evaluate the accuracy of the proposed model. This unsupervised model is called Probabilistic Relational Unsupervised Topic modelling (PRUTM). The supervised approach shows better performance than the PRUTM when applied to the same dataset. The following sections will describe both models in detail with the evaluation, experimental, and results sections.

3.3 Word Co-occurrence Pattern

Words might have several meanings when they occur within different contexts, especially in short text. Thus, aggregating short text based on topics will not be reasonable if it depends solely on words on their own. Therefore, there is a likelihood of multiple words that occur within the same context might define the topic that a short text belongs to, and this likelihood is called a word co-occurrence pattern.

We have been inspired by the relation between the context and words as well as the vast changes in the English language that have been noticed and tracked by the Oxford English Dictionary [4]. Therefore, the first goal was to undertake an analysis of a stream of textual data to detect these words' co-occurrence patterns, and then track the changes over time if any appeared.

The main purpose of considering these patterns as part of the proposed model is using these patterns as the landmarks for each single topic, where these patterns will differ from one topic to another. The analysis for the word-context co-occurrence took place at the word level depending on the frequencies of words and the context-word. The context-words are the words surrounding another word within the same sentence, or according to some researchers [73] [68] [74], those occurring in the same document. The co-occurrence pattern works according to the same concept, considering the likelihood of similar words occurring several times in different documents.

Recent approaches like the Self-Aggregation Topic Model (SATM) [11], pseudo-document-based Topic Model (PTM)[75], [73], and Sparsity-enhanced PTM model (SPTM)[76] use

words embeddings or pseudo-documents for topic modelling with respect to words' co-occurrence patterns. However, we could not find any obvious study that could track any word-context co-occurrence changes over time for social media short text. Before we describe our model, we have studied these changes over time by calculating the frequency change with respect to time for the words as well as the context pattern co-occurrence over the same time. A time window of three months from Twitter data was selected, as can be seen in figures 3.3, 3.4, and 3.5.

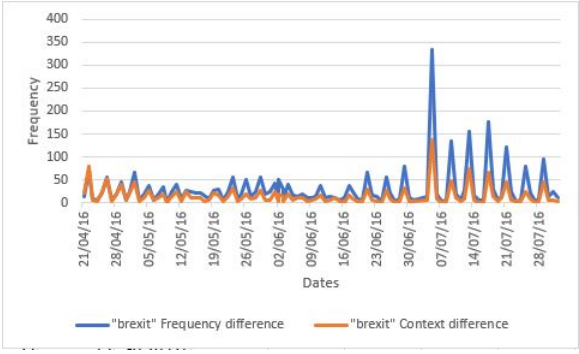


Figure 3.3: "brexit" word frequency difference vs context frequency difference

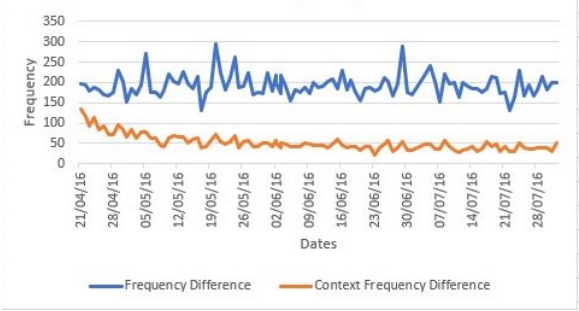


Figure 3.4: "car" word frequency difference vs context frequency difference

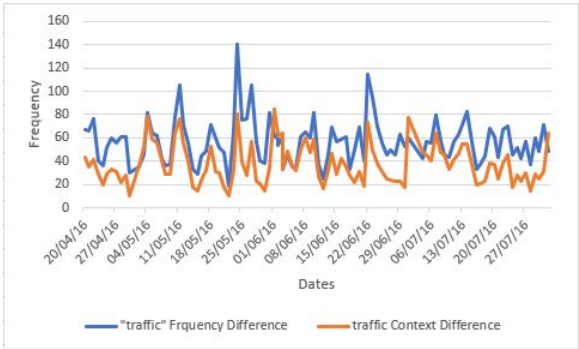


Figure 3.5: "traffic" word frequency difference vs context frequency difference

In figures 3.3 and 3.5, the difference between any two days' frequencies for the word "brexit", for example, for the following chosen three months is very close to the difference of the context words frequency. Additionally, the ratio between both context and word frequencies are more likely to be consistent. This gives an indication of having similar context words for "brexit" in any tweet. These kinds of words are more likely to be considered in a

tweet when compared to other word such as “car”, as shown in figure 3.4. Hence, the word “car” will be removed by the word-to-features process for any tweet before converting it to features because it shows less consistency between its context and words frequencies. Figure 3.4 shows the orange curve that represents the context frequency difference for the word “car” that is described as dying. On the other hand, the blue curve represents the word’s frequency difference which shows a consistency of over 200. Thus, fewer words will produce a lower cost for an additional criterion for the proposed model.

We refer to the word frequency as t and for the context frequency as c. Also, w represents the time window which is one day. Hence the equation for calculating the differences for both of the last two columns will be :

$$f(w,t) = (t_{w-1} - t_w) \ \textit{where} \ \{w \in W | w\text{=}2,...,n\} \tag{3.1}$$

$$f(w,c) = (c_{w-1} - c_w) \ \textit{where} \ \{w \in W | w\text{=}2,...,n\} \tag{3.2}$$

where W represents the dates for the whole corpus of the streamed tweets. Also, n represents the number of the selected dates for this example, so $n\text{=}14$ as shown table 3.1. The two equations 3.1 and 3.2 show how each cell in the last column is calculated producing the conclusion of having a ratio between the word “brexit” frequency and its context words frequency over time. This ratio will detect the anchor words from the whole corpus. Thus we calculated the average of the difference over time then took the ratio for the two results as shown in the following equations:

$$F(w,t) = \sum_{w=1}^{n} f(w,t) \text{ where } \{w \in W | w\text{=}2,...,n\} \tag{3.3}$$

$$F(w,c) = \sum_{w=1}^{n} f(w,c) \text{ where } \{w \in W | w\text{=}2,...,n\} \tag{3.4}$$

The following table 3.1 shows the implementation of these equations on the words frequencies over time.

Dates	“brexit" frequency	Context word Frequency	Word frequency difference	Context-word frequency difference
01/05/16	285	287	-	-
02/05/16	350	332	65	45
03/05/16	355	335	5	3
04/05/16	373	347	18	12
05/05/16	410	373	37	26
06/05/16	416	379	6	6
07/05/16	436	390	20	11
08/05/16	471	409	35	19
09/05/16	472	409	1	1
10/05/16	495	423	23	14
11/05/16	535	448	40	25
12/05/16	543	451	8	3
13/05/16	569	476	26	25
14/05/16	592	488	-	-

Table 3.1: Two weeks of the word "brexit" frequencies and context-words frequencies

Table 3.1 shows 14 consecutive days of the word “brexit” frequencies in the word frequency column. The next column shows the words of context frequencies co-occurring with “brexit” in the same tweets per day. For example, we will consider the first “brexit” word in the tweet “I m so sick of hearing Brexit means Brexit”. The words “hearing” and “means” are occurring on both sides of “brexit” and are the context words. These words will not be counted in the context frequency if they occurred again with “brexit” as it will be counted only once. So, the differences between each successive cell in the same column are shown in the columns of context and words frequencies differences. We took these differences into account as they show the relation between both context and word frequencies as described previously in figure 3.3. Moreover, the words’ context frequency is calculated once by the occurrence of the context word and it will not be counted if it appears again as described in the previous example. Therefore, the differences in context words frequency mean the frequency of new words added every time window which is one day.

Moreover, a ratio was calculated using the following equation 3.5 to determine the word importance within tweet.

$$R = \frac{F(w,c)}{F(w,t)} \tag{3.5}$$

The value of R will decide which word would be considered as an anchor and which will be removed from the feature vector calculation that will be described in the TF-ICTF section. According to our experiments, we found that the threshold that improved the proposed model classification accuracy was less than 5. So, R should be less than 5 for any word to be considered as an anchor word. As a result, the words that are similar to “brexit” and “traffic”

as in figure 3.5 will be considered anchor words when appearing in each tweet. This will make the relations between tweets that have similar words easier to classify. Examples of anchor words and removed words can be seen in table 3.2. The anchor words are the ones that will be considered when calculating the features vector for each single tweet while the rest of the words will be removed.

Anchor Words	Removed Words
traffic brexit weather rain stuck shopping stadium sunny football	the more london car new tfl

Table 3.2: Anchor words and removed words

Anchor words, and the ones that will be removed, will depend on how they appear in the tweets. Thus, they may not fall under any English language grammar rules. Accordingly, there might be some changes to the words to be either anchor or not over time, depending on how people are using these terms (words). However, our proposed approach produces the facility to track these changes. As a result, differences between words and context will provide a clear view about the selected words from each tweet.

3.4 PRSTM framework

Topic modelling of short noisy text, like tweets, is much more difficult than modelling long documents due to the informality of short text. The difficulty is mainly because there is less information in short text making it hard to aggregate similar ones under the same topic. Moreover, the formality in long documents makes it easy to predict which words are related to a specific topic, unlike in social media text that does not fall under most of the formal writing conditions. Thus, training any model on a corpus like Wikipedia might produce good results on short text but not in the long term, due to the changes in language, but it can be used as an auxiliary information for training. For this reason, we have not considered training on an external dataset like the Wikipedia corpus because of not having most of the informal words taken into account. Traditional classification techniques like Latent Dirichlet allocation (LDA) [9] , supervised latent Dirichlet allocation (sLDA) [75], and probabilistic latent semantic analysis (PLSA) [77] will face similar challenges because of words' sparsity in the short text environment [78]. Additionally, much information will be considered as noise and ignored in most text analysis which makes it a loss of valuable information.

Semantic co-occurrences between words like "m8" which refers to "mate" or "ez" which means "easy" , for example, are hard to find unless they have been used by many people and become a trending word. Our proposed approach tracks the changes of most of the new words regardless of the informality and catches these words co-occurrence patterns. These patterns were part of multiple stages that led us to several proposed contributions and enhancements to NLP of short and noisy text. The proposed models in topic modelling are our first contribution and will be demonstrated in this section.

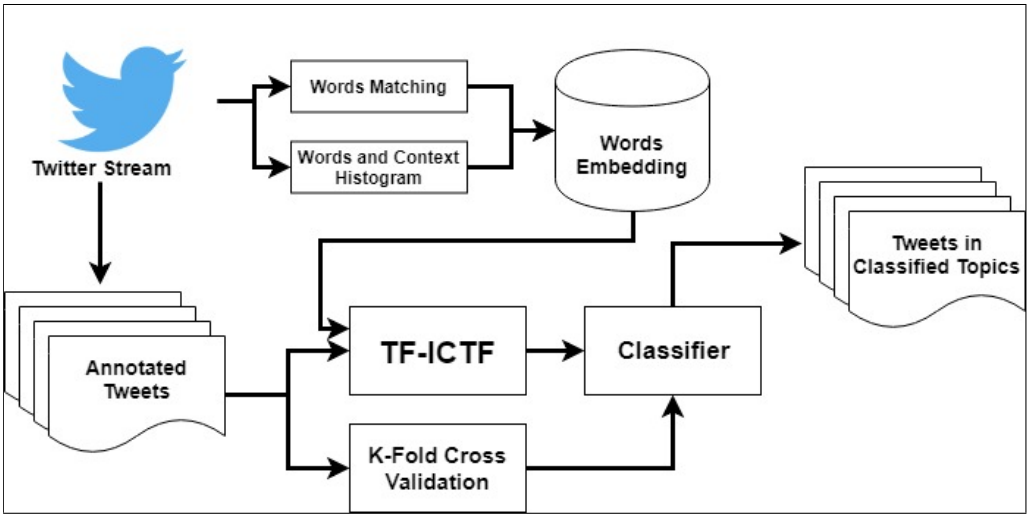


Figure 3.6: PRSTM with supervised machine learning

PRSTM, as shown in the figure 3.6 is a statistical model which consists of multiple stages that have Twitter data as input to be classified into several topics. Pre-processing is the first stage of this model which extracts the text from the tweets, cleans it, then produces cleaned tweets' text to the next stage. In addition, a word matching technique for short text was developed as a second step in the pre-processing level and it will be described in the pre-processing section.

Annotated tweets classified into topics by a trusted party provides excellent training data for PRSTM. This data was also used in the testing and validation process for both supervised and unsupervised machine learning. Amazon Mechanical Turk [39] tools were used for tweets annotation to multiple topics. Refer to Appendix B for more details about the annotation process. These tweets were taken from our dataset that we collected using the Twitter API, as described in the dataset collection section 3.2.

The other main stage in the proposed model is supervised machine learning of the annotated dataset that is fully described in the dataset section 3.2. In addition, the Features Extraction level converts the tweets to the vector space. The Features Extraction stage is the main core for the machine learning part together with the classifiers that will be discussed later. The following sections will describe the stages in figure 3.6 in detail.

3.4.1 Pre-processing

The streamed tweets were stored in CSV files, as described in the data collection section 3.2. These tweets' text had been cleaned, although the cleaning process varied in terms of which part of the text should be taken into consideration. For example, the special characters like hashtags in some studies [15] [79] are considered as either auxiliary data or main data in helping to find the sentiment polarity of the tweet. Thus, the noisy Twitter text posted by someone might have some contextual, or lexical error, in addition to some informal words which are known as "slang" or "urban" words. Accordingly, it does not mean that these words are not part of the natural language or do not have a meaning. Thus, the PRSTM takes the informality case into consideration and treats these informal words in the same way

as the formal words. As part of this model, the pre-processing step does not check in the dictionary for the words and meaning, but focuses on the words themselves regardless of the formality, with the convention that each word in the tweet which is posted by a human should mean something with respect to the context.

The non-standard word formats in text are increasing according to what is used among people on social media. Furthermore, the words which are in non-standard format individually might not have meaning on their own but might have been within the context. This kind of noise could be linked to a topic or similar word. The following figure 3.7 shows the tweets cleaning process.

Figure 3.7: Tweet sleaning

To reduce the noise from tweets, several measures and techniques had been taken. As shown in figure 3.7, the words in red in the tweet were links, special characters, and sometimes numbers. The first step in noise reduction is removing the links and special characters that affects text and replacing them with spaces together with the "NULL" which had been added as the margins of the tweet. These margins will help in the creation of the words embeddings which will be described in detail in the word-embeddings section

Secondly, the stop words that ware shown in yellow were removed as well. The main reason is to keep just the words so we were able to establish co-occurrences between the words directly without any noise that might affect the result. Additionally, we found that this extra step does not have any effect on the meaning of the tweet. This is shown clearly in the data annotation that will be described on the next section 3.4.2. Additionally, the re-tweeted tweets had been removed from the dataset as the *"RT"* in any tweet indicates that it is re-tweeted.

Additional step of the noise reduction in our proposed model had been made by removing the words with high frequency. These words have similar effect of the stop words and the technique that used to remove these words will be discussed furthermore in the following sections.

Removing the English standard stop words was done by the provided NLTK (Natural Language Processing Tool Kit) in Python [80]. NLTK also provides the regular expression package which had been used to clean the previously mentioned noise like links and special characters. Part of these regular expression functions had been reused from one of the Github repositories [81] that provides a much better cleaning process of tweets than the provided tools in NLTK and other language packages. Some of these regular expressions can be seen in the figure 3.8.

```
emoticons_str = r"""
    (?:
        [:=;] # Eyes
        [oO\-]? # Nose (optional)
        [D\)\]\(\]/\\OpP] # Mouth
    )"""

regex_str = [
    emoticons_str,
    r'<[^>]+>', # HTML tags
    r'(?:@[\w_]+)', # @-mentions
    r"(?:\#+[\w_]+[\w\'_\-]*[\w_]+)", # hash-tags
    r'http[s]?://(?:[a-z]|[0-9]|[$-_@.&amp;+]|[!*\(\),]|(?:%[0-9a-f][0-9a-f]))+', # URLs
    r'http[s]?:\\(?:[a-z]|[0-9]|[$-_@.&amp;+]|[!*\(\),]|(?:%[0-9a-f][0-9a-f]))+',   # URLs2
    r'\\(?:[a-z]|[0-9]|[$-_@.&amp;+]|[!*\(\),]|(?:%[0-9a-f][0-9a-f]))+',   # URLs3
    r'(?:(?:\d+,?)+(?:\.?\d+)?)', # numbers
    r"(?:[a-z][a-z'\-_]+[a-z])", # words with - and '
    r'(?:[\w_]+)', # other words
    r'(?:\S)' # anything else
```

Figure 3.8: Regular expression written in the Python language for tweets text cleaning

Algorithm 1 shows how the input text of the tweet is cleaned. The regular expression is part of this cleaning process as shown in figure 3.8 where it removes the URLs (Unified Resource Locator) or "Links" as well as hashtags and emojis as described in the previous paragraph. The rest of the algorithm will convert the resulting text into a list of words to be forwarded to the next stage to create the words embeddings.

algorithm 1 Cleaning tweets of noise

Require: X as input text tweet

Ensure: $Outlist$ as a list of clean text

Regular Expression Clean(X)

{Refer to figure 3.8 for more information about the Regular Expressions used}

$Wl \leftarrow$ tokenize(X)

$L \leftarrow$ length(Wl)

Initialize $Outlist \leftarrow$ ["$NULL$"]

for $i = 0$ to L Step 1 **do**

  if (Wl [i] not in $StopWords$) Or (Wl [i] not $Special\ Characters$) **then**

    $Outlist \leftarrow\square\ Wl$ [i]

  end if

end for

$Outlist \leftarrow$ ["$NULL$"]

Return ($Outlist$)

Finally, the resulting tweet is shown in the middle of the two "NULL" words which replace the beginning and the end of the tweet. Next, the cleaned tweets go in one of two directions. The first direction is the data annotation where it will be classified using Amazon Mechanical Turk, the second is the n-gram generator or creator as the n-gram is the infrastructure of the PRSTM.

3.4.2 Words Embeddings

In our proposed model we have generated words distributed representations. These representations are called word embeddings that is structured in a form of trigrams. The words co-occurrences were calculated based on the embeddings distributions of words that could

be continuously updated based on the tweets stream feed. These tweets will be converted to words co-occurrences and then to features using our proposed modified version of TF-IDF that will be described further in the next section. Compared to neural network character based embeddings produced by Vakulenko et al. [17], our embedding outperforms these embeddings by the correlation technique between word and the continues lexical representation of words in real-time. Although, some of these characteristics were not discussed properly in the character-based approach. Moreover, language variation was limited to the misspelled words rather than extending this approach to cover the alternative words. These alternatives are words with abbreviations (words style) or new variation that were not connected directly to the real formal word like hash tags writing and mentioning which we have covered within the variation concept.

Likewise, word2vec [19] considered as a standard approach for creating word embeddings using neural networks. This approach takes the vocabulary into consideration compared to our approach which adopts the words co-occurrence patterns into consideration. The aim of word2vec is to convert the words to vector (numerical) representation as we share the same perspective. On the other hand, our approach relying more on how the word is represented within the context in addition to the vectorisation technique that we derived from TF-IDF. Our approach outperforms word2vec in updating the co-occurrence patterns of words based on new input.

An enormous number of tweets were streamed to build word embeddings and then the topic modelling process. Thus, the main task is to represent these tweets in a dataframe that could preserve words and context co-occurrences. The n-grams technique [82] was proposed to accomplish this task but with additional adjustments that make it work with the PRSTM. The n-grams form, which is the result of applying the n-gram technique, is defined as the contiguous sequence of n number of terms, where it is used in statistical linguistics.

A trigram is the proposed version of an n-gram with a structure of three words for each tuple, the centre word and one on each side. Figure 3.9 shows how the selected window moves across the tweet between the two "NULL" words that had been added, as has been described. Accordingly, the "NULL" word will not go through the same process as other words in the Twitter text. Although, the trigram will make it very easy to spot these kind of words by simply finding the percentage of the "NULL" words before or after the word over the word's frequency. Moreover, the sequence of the words within the tweet will be kept as well for further analysis.

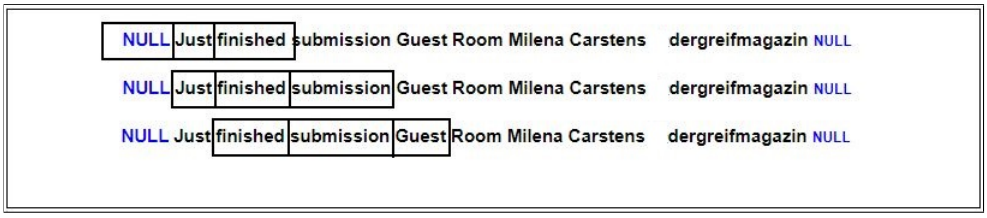


Figure 3.9: Trigram generating process

Additionally, adjacent words co-occurrence frequency is calculated as each two words had their co-occurrence frequency analyzed within the whole corpus. This frequency is

stored as a tuple together with the related word in a vector of the same kind of tuple related to a centre word. As a result, the trigram takes the form of three tuple vectors: one for the centre words and two for the words on each side related to each single word as shown in figure 3.10.

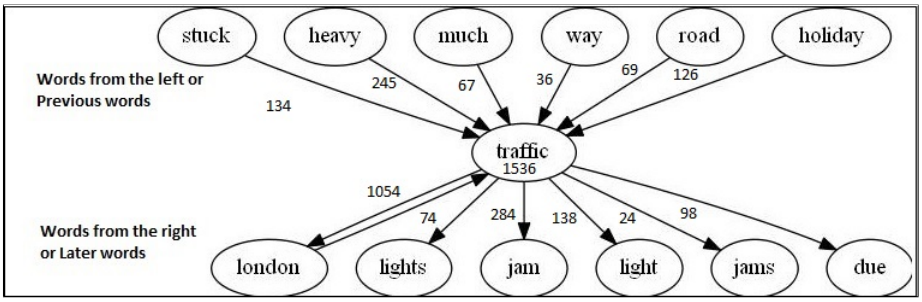


Figure 3.10: centre Word example from trigram

As shown in figure 3.10, the word "heavy" occurred 245 times with the centre word "traffic" with its frequency of 1536. This does not mean that the word “heavy” frequency is 245 as it has its own frequency and record that is similar to “traffic”. The word “london” also has a special case as it came before and after the centre word with a total frequency of 1054, thus, there are two arrows to and from the centre word. The arrows denote the precedence of words in a tweet and how many times this word appeared in this position before or after the centre word. This gave us another predictive estimation to find the next possible word that could be mentioned depending on the frequencies in the embeddings. Also, this could be applicable using the co-occurrence patterns which provides more information about the possible word sequence in a written text. We will consider that as a future work.

Similar words being in many topics is the main reason for using co-occurrence patterns rather than single words for a specific topic. Some words appeared in several topics with high frequencies making it hard to use the bag of words approach for the topics classification. Words such as “london”, “like”, “time” were frequently used in several topics as shown in bold in table 3.3. The words in table 3.3 shows the most frequent words in the third dataset mentioned in the dataset section 3.2 according to the classes. The “Non English” class could be the most unique class as the words in this class are totally different from the others unless there are similar words between English and the non-English one, which is not that frequent.

Traffic	Food	Bussiness	Weather	Shopping
delay	**like**	job	mph	get
traffic	food	**london**	today	one
severe	one	**like**	wind	shop
roadworks	**london**	uk	rain	**new**
london	get	**new**	humidity	good
road	day	work	fine	please
pending	going	good	slowly	need
closure	good	get	temperature	today
m25	lunch	people	rising	day
lanes	much	one	weather	got
time	**time**	**time**	cloud	always
station	eat	great	**london**	money
train	chicken	business	ukweather	shoes
j11	tea	need	tmp	**london**

Movies	Sport	Not English	Event	Other
like	game	de	**like**	**like**
film	football	que	get	get
new	**like**	la	people	one
get	team	en	see	love
love	get	le	**london**	know
still	good	un	day	**time**
see	great	el	last	good
watch	england	je	today	**london**
time	one	es	love	people
show	well	si	great	see
got	go	se	**time**	day
last	season	eu	**new**	got
great	players	pas	good	**new**
tv	**time**	**london**	uk	back

Table 3.3: Word frequencies for the topics of the third dataset in dataset section 3.2

Discovering co-occurrence patterns as described previously is one of the main steps in the topic modelling process. The following table 3.4 shows some frequent patterns detected from the annotated tweets as described in the third dataset of the dataset collection section 3.2. The first row represents the topics and each topic of the column contains ten of the frequent co-occurrence patterns.

Traffic	Food	Bussiness	Weather	Shopping
[null | roadworks | severe]	[join | us | drink]	[jobs | hiring | careerarc]	[rain | today | mm]	[free | wholesale | orders]
[roadworks | severe | delay]	[null | delicious | paid]	[job | jobs | careerarc]	[null | wind | mph]	[purchase | day | amp]
[status | pending | closures]	[delicious | paid | visit]	[job | near | london]	[today | mm | humidity]	[check | uk | size]
[severe | delay | m25]	[fest | tonight | cakes]	[researching | websites | business]	[temperature | rain | today]	[uk | size | womens]
[pending | closures | lanes]	[back | healthy | eatclean]	[mind | answering | questions]	[mph | press | mb]	[spent | pretty | much]
[closures | lanes | beepbeep]	[live | pizza | express]	[end | years | unemployment]	[null | tmp | wind]	[louis | vuitton | pair]
[access | status | pending]	[im | make | food]	[contract | place | sort]	[press | mb | cloud]	[store | extra | free]
[delay | m25 | j11]	[place | dont | like]	[friendship | marketing | director]	[ft | rain | mm]	[order | soon | already]
[14 | may | 2016]	[aldi | marlborough | pinot]	[london | sales | job]	[rain | mm | humidity]	[online | collect | parcel]
[a285 | near | chichester]	[india | chefs | opened]	[round | interviews | taking]	[wind | mph | press]	[ive | checked | sold]

Movies	Sport	Not English	Event	Other
[star | wars | day]	[mitcham | utd | fc]	[haha | kalau | aku]	[null | looking | forward]	[love | love | love]
[cinema | guildford | vote]	[utd | fc | vs]	[bon | jovi | van]	[please | help | end]	[null | really | need]
[guildford | vote | films]	[fc | london | fa]	[haha | babi | la]	[team | would | love]	[mph | press | mb]
[films | youd | like]	[london | fa | senior]	[aku | buat | ni]	[help | like | information]	[really | need | votes]
[prod | filmed | via]	[fa | senior | cup]	[eu | xingo | aslaug]	[and | huge | thanks]	[nobody | cares | nobody]
[casting | season | follow]	[jamie | vardy | jamie]	[adalarini | aldjklari | gaspettikleri]	[information | wed | happy]	[pledges | vote | share]
[review | film | son]	[null | euro | 2016]	[puedo | pas | de]	[null | happy | birthday]	[london | greater | london]
[another | marvel | movie]	[foals | night | tonight]	[mais | cest | du]	[null | hello | announcement]	[breathing | happiness | back]
[lights | camera | action]	[finishing | fantastic | goal]	[et | ils | sont]	[please | rt | thanks]	[happiness | back | long]
[tickets | still | available]	[football | player | imtrying]	[sinister | dictatorship | key]	[null | cant | believe]	[purchased | tickets | rtn]

Table 3.4: Word co-occurrence patterns for the topics of the third dataset in dataset section 3.2

The co-occurrence patterns (that have been described in section 3.3) for a specific topic do not necessarily include words related directly to the topic which might be semantically related, although, there still might be some degree of similarity between words that could describe other patterns in the same topic. We tracked the patterns in the first level for the "traffic" topic, as shown in the example in figure 3.11, then we fetched the similar patterns that have partially similar words. Figure 3.11 shows the topic co-occurrence patterns in the first node on the top. The first level connected to this represents the co-occurrence patterns found in this topic. Reaching to the bottom level in figure 3.11, the similar patterns are results from the similarity process mentioned at the beginning of this paragraph.

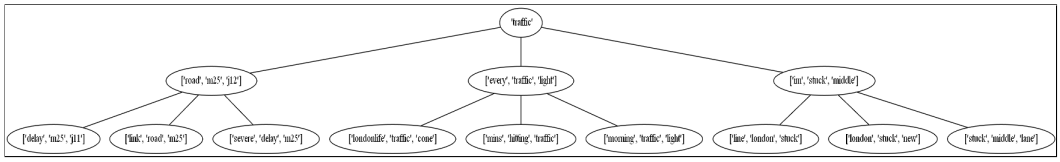


Figure 3.11: "traffic" topic two level co-occurrence pattern with a three words window size

The process of generating or updating the word embeddings is described in detail in the following algorithm 2.

algorithm 2 Building Word Embeddings

Require: X as input list of clean tweets, T as the list of word embeddings

Ensure: Updated T $\{T$ is structured as list of vocabulary with a frequency $T[i] = [term, frequency]$. Also, two lists for each record of T with similar structure of T[i] but with the co-occurent words on both sides of the word , Right list as *Rlist* and the left list as *Llist*. $\}$

$L \leftarrow length(X)$

for $i = 1$ to $L-1$ Step 1 **do**

if X $[i]$ in T **then**

$Ind \leftarrow X$ $[i]$ {F}inding the index of the word in T

Increment(T $[Ind]$ $frequency$)

else

$T \leftarrow\square X$ $[i-1]$

T $[i]$ $frequency = 1$

end if

{I}ncrement the frequency of the Word in left

if X $[i-1]$ in *Llist* **then**

$Ind \leftarrow X$ $[i-1]$

Increment($Llist$ $[Ind]$ $frequency$)

else

$Llist \leftarrow\square X$ $[i-1]$

$Llist \leftarrow\square X$ $[i-1]$ $frequency = 1$

end if

{I}ncrement the frequency of the Word in right

if X $[i+1]$ in *Rlist* **then**

$Ind \leftarrow X$ $[i+1]$

Increment($Rlist$ $[Ind]$ $frequency$)

else

$Rlist \leftarrow\square X$ $[i+1]$

$Rlist \leftarrow\square X$ $[i+1]$ $frequency = 1$

end if

end for

$Outlist \leftarrow\square$ $[T, Rlist, Llist]$

Return ($Outlist$)

The algorithm takes a list of list as an input. The first list is the tweets lists as each tweet is a list of words which had been generated by the pre-processing algorithm 1.

Finally, the last outcome from the trigram is the automatically generated bag of words which needs just the input for the topic and it will generate this input's bag of words depending on the stored co-occurrences. In addition, other words will be added to the bag of words using the relational factor that will be described in detail in the next section.

3.4.3 TF-ICTF

Topic modelling for short text is challenging especially when the text is as noisy as Twitter if compared to standard text. The shortness and informality, which resulted from the users' various posts, were highlighted among these challenges. Further, the lack of one general representation for all the tweets' text, for example, minimizes the state-of-the-art algorithms' ability to produce good results. In this research, we produce a unified tweets text platform that links all terms in one single structure in the form of a connected graph. This structure is generated automatically and is dynamically grown as several tweets keep feeding into the system. Moreover, all the co-occurrences for any term were registered with a weight that represents it.

Many traditional algorithms do not perform well on short text [83] [84] . The TF-IDF algorithm is one of these algorithms that evaluates the importance of a word within a document by calculating its weight. TF-IDF works successfully on long documents to extract any word's weight depending on its frequency regardless of the context, but not on short text [85].

Topic modelling for a short text environment like Twitter is challenging, where 140 to 280 characters is the maximum length for a tweet. The word's context in these tweets is very important in the topic modelling process. Moreover, the topics for the collected tweets are not known in advance making it a challenging task to gather everything in one corpus. Thus, we proposed a new algorithm, based on TF-IDF, that will take into account the informality and noisiness in short text; it is named TF-ICTF (Term Frequency-Inverse Context Term Frequency).

TF-TDF stands for Term Frequency-Inverse Document Frequency. This algorithm is applied to long text to show the topic for long text depending on words' frequency. Let t refer to the term's frequency in a document and d refers to the number of documents that this term appeared in. The equation below shows the standard equation of the algorithm:

$$tf(t,d) = 0.5 + 0.5 \cdot f_{t,d}/max\{f_{t',d} : t' \in d\} \tag{3.6}$$

where $tf(t,d)$ is the term frequency weight which means how many times the term appears in the document. The 0.5 is double normalization and $f_{t,d}$ is the term frequency derived by $|maxft', d : t \in d|$ the maximum number of this term appears in all of the documents.
and the IDF equation is:

$$idf(t,D) = log(N/|\{d \in D : t \in T\}|) \tag{3.7}$$

where $idf(t,D)$ is the result of the inverse document log. N is the total number of documents and the base $|d \in D : t \in T|$ is the frequency of a term in all of the documents.

In order to deal with the problems that short informal text presents, the TF-IDF algorithm has been extended to TF-ICTF. Many factors were included to make it applicable to short text. We found that there is a relation between the context words frequency and the word frequency within the context and this relation had been described briefly in section

3.3. Therefore, we produced a developed version of tf-idf equations that could utilize this relation to produce a form of feature with respect to any word frequency and its context words frequency. Our equations are as follows:

$$tf(t,d) = 0.5 + 0.5 \cdot f_{t,d}/C \tag{3.8}$$

where C is the number of words that came with the selected word (the word context frequency). For example, refer to figure 3.10 where $C = 12$.

The other equation for the IDF is:

$$ictf(t,D) = log(N/C) + RF \tag{3.9}$$

where C is as defined before and RF is the relational factor. This RF is calculated as in the following equation:

$$RF(t) = \sum_{i=1}^{n} f(t_i) \tag{3.10}$$

$$f(t_i) = \text{the frequency of any term related to the topic } / \text{its frequency} \tag{3.11}$$

The topic is an input that should be entered to direct the search in a specific direction rather than calculating the weight for all the given words. Accordingly, replacing the number of documents by the number of related words is justified by the vast number of tweets that reached millions. Therefore, as long as the tweets frequency is increasing, the tf-idf value will get smaller which will lead to the feature of any word becoming inconsistent over time. If we consider each single tweet as a document we will not get an accurate weight for each word. The alternative is to calculate the term weight within its topic (or related words).

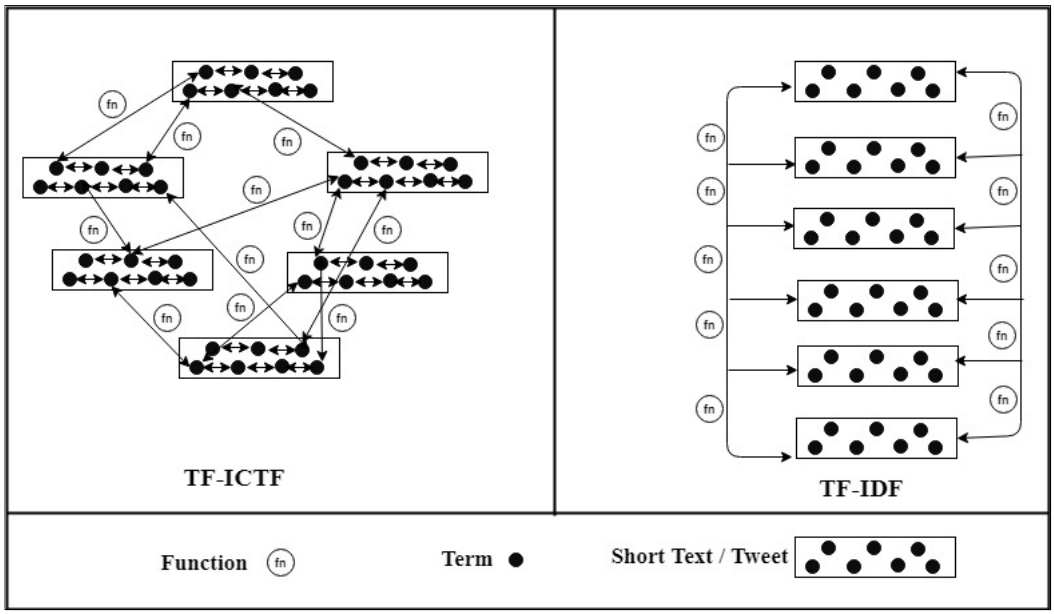


Figure 3.12: TF-ICTF vs TF-IDF

The difference between the two algorithms is shown in figure 3.12. Thus, the TF-IDF algorithm considers the single tweet as a document where the term frequency (tf) will be calculated regarding all classified and unclassified tweets. The TF-IDF value for each term will be very small because of the number of documents which could be miniscule. On the other hand, TF-ICTF will localize the problem to the term with respect to the adjacent ones within the tweet as we described in the trigram section. Accordingly, the value of the term is unique and calculated with respect to other terms related to it. These related terms appear more frequently in similar tweets. Nevertheless, there is a possibility of the related words occurring in other tweets. The TF-ICTF value for each tweet will be biased by the main term value.

The small arrows between the terms in the TF-ICTF part as shown in figure 3.12 are the RF function which finds the relational factor between the words with respect to context. Furthermore, the "fn" sign which refers to function is defined as the TF-ICTF value. Likewise, the same symbol is a function as well but it is the TF-IDf value in its part of the figure.

Moreover, the TF-ICTF value for each entity is more stable than the TF-IDF value which is affected by the frequency of the documents (tweets). The ICTF value will remain stable because it depends on the term context rather than the term frequency. The following figures show the huge differences between the two algorithms for different terms on randomly selected days where the term frequency might change.

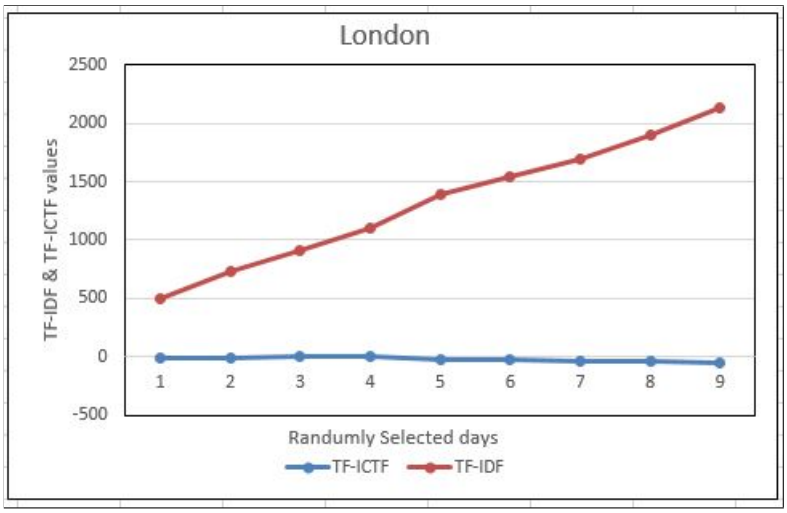


Figure 3.13: “London” word TF-ICTF vs TF-IDF values

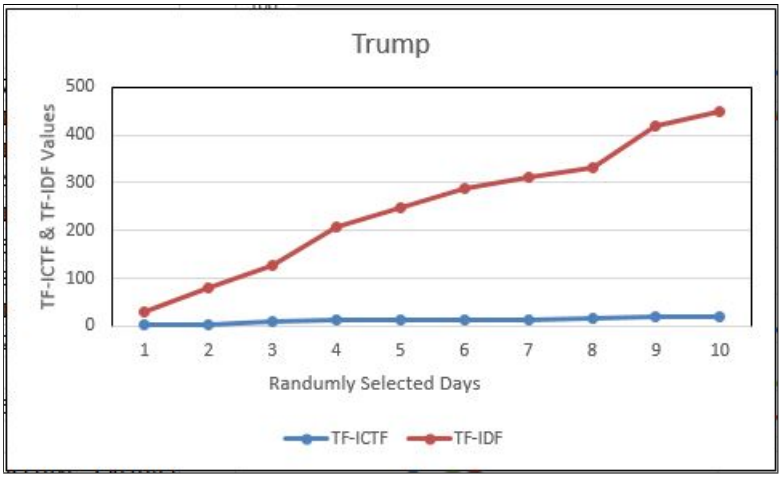


Figure 3.14: "Trump" word TF-ICTF vs TF-IDF values

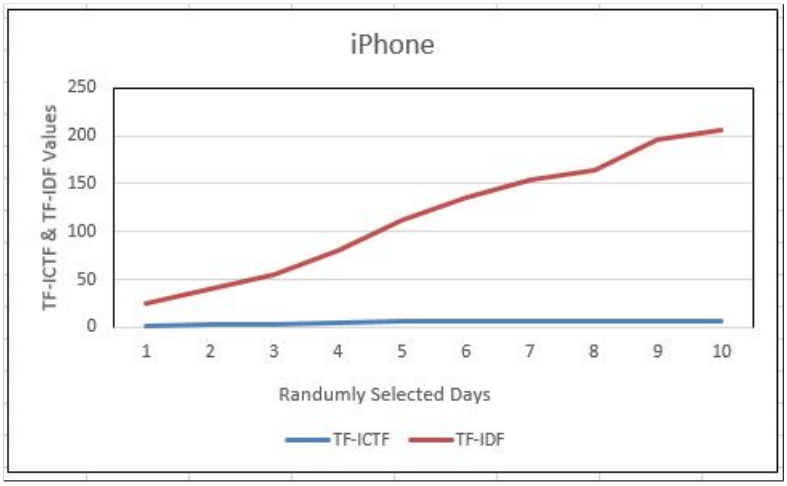


Figure 3.15: "iPhone" word TF-ICTF vs TF-IDF values

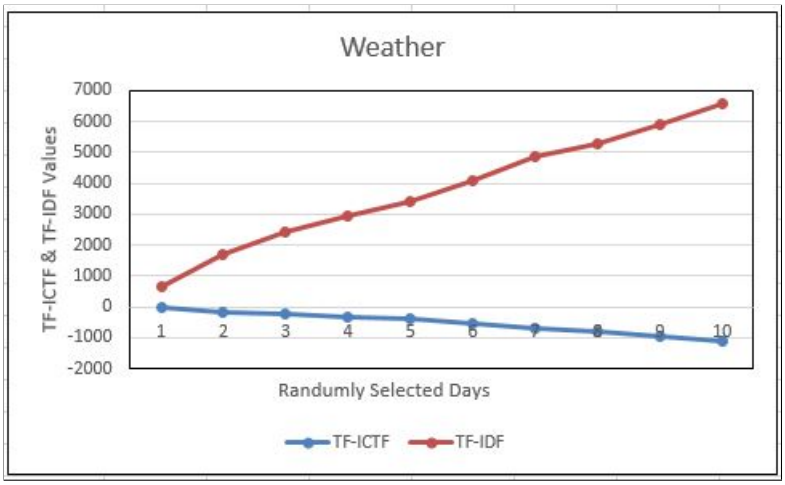


Figure 3.16: "Weather" word TF-ICTF vs TF-IDF values

The TF-ICTF properties can be summarized in the following points:

1. As previously mentioned, the stable TF-ICTF value for each term provides a unique

representation for this term. Consequently, the patterns generation for each tweet will be unique.

2. The ICTF value could be either positive or negative. A positive value means that the words of the context that are surrounding the term are changing frequently, in a manner similar to stopwords that change frequently depending on the context requirements.
3. The words with a stable average frequency are more likely to be keywords.
4. TF-ICTF ensures the dynamic update of words where any change in the context or frequency will be counted regularly.
5. It is easy to detect the words' life cycle as many new words will appear and be used in social media. Also, some other words might not be used anymore. For example, "brexit" was not trending a few years ago compared to "weather" which is a word that is used frequently. As a results, some words occur for a short time and disappear or new words might replace some old used ones.

3.4.4 Word Matching

Words matching is the second stage before words embeddings and follows cleaning of the tweets. It aims to reduce the informality by finding similar words. The state-of-the-art Approximate String Matching Algorithm (fuzzy similarity) [86] is used to find the similarity percentage between words. Noise reduction is the main goal of this stage in general as it will produce less text ambiguity, i.e. the classifier will not give similar terms different values. Furthermore, this step will lead to sparsity reduction as the feature vector will be more consistent.

The syntactic approach of finding similar words implies that these words should have the same spelling or at least for the most part. Those, using many metrics like fuzzy [86] and cosine similarity [87] are examples of this kind of similarity: one of the reasons that affects the accuracy is the noise in text. This noise as we stated earlier was described because the machine did not understand it or could not classify it according to some criteria. In short text case, many types of noise could be words written with syntax errors or written in an irregular form. Hence, one of the PRSTM parts is the Word Matching process. The main objective of this step is to gather similar words together as one entity so that any occurring word in that entity will be processed similarly.

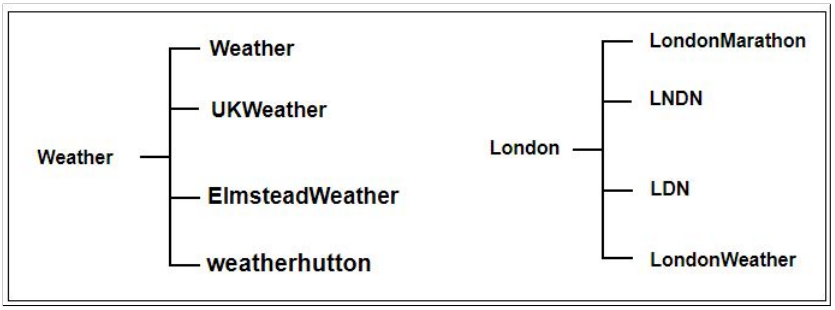


Figure 3.17: Word matching examples

Language variation could be one of the approaches that is included within the word matching technique. Yang and Eisenstein [88] discussed language variation and its growth in social media. They proposed a neural network pipeline to overcome this challenge and show the effect on classification performance. As an alternative, we have used our word co-occurrence matching technique to overcome the same problem. Examples of word matching are shown in figure 3.17 which shows how words that might have similar letters, or some similar words, could have the same meaning. Therefore, each word in the trigram should be checked after it had been positioned within the dataset. The check will operate as a letter check from the first letter of the input word up to the last letter to be matched with output that should be either "NULL", or shows some other possibilities of similarity. Additionally, the process will continue to check if the candidate word is similar to the input by checking the related words (i.e. the context of the two words). If there are many similarities then they should be combined, otherwise, the process will move to the next word. The process is shown in the following algorithm 3. More examples are shown in the following table depending on word lengths and how they are similar depending on the context.

Word	**Word Length >**	**Word Length <**	**Word Length =**
traffic	@london_traffic #slowtraffic #traffic #atxtraffic	trafic trafik	trafficccc
weather	#weirdweather #amazingsunnyweathertoday @skynewsweather	wether	weatheeeeer
brexit	#brexiters #thebrexitbill #thisisbrexit #brexitdefeat	-----	brexittt
m25	#m25 @m25news #worsethanm25 #m25traffic	------	------
congestion	congestione #congestion #beatcongestion #trafficcongestion	congesti	-------
goal	goals #goals2017 goalkeepers @101greatgoals	--------	goooooooooaaal

Table 3.5: Word matching examples depending on the words length and context

algorithm 3 Finding Matched Words

Require: T as the list of word embeddings

Ensure: Wrd List of updated bag of informal similar words {Wrd which is structured as a list of lists where each list consists of similar words }

$Temp = T$

$L \leftarrow length(T)$

for $i = 0$ to $L-1$ Step 1 **do**

if T [i] not in Wrd **then**

$Wrd \leftarrow\square\ T$ [i]

Delete T [i] from $Temp$

$L1 \leftarrow length(Temp)$

for $j = 0$ to $Ll-1$ Step 1 **do**

if $(Threshold1 < Similarity(T[i], Temp[j]))$ and $(Threshold2 < ContextWordsSimilarity(T[i], Temp[j]))$ **then**

$Wrd \leftarrow\square\ Temp[j].indexof(T[i])$ {T}hreshold1 is the level of similarity between the two words. Also, similarity is the Fuzzy similarity algorithm. {T}hreshold2 is the percentage of similar context words of the two word intersection divided by the context words of $T[i]$

else

if T [i] in $Temp$ [j] **then**

{i}f the word is part of a longer one

if $Threshold2 < ContextWordsSimilarityT[i], Temp[j]$ **then**

$Wrd \leftarrow\square\ Temp$ [j].$indexof\ T$ [i] {Threshold2 is the percentage of similar context words of the two word intersection divided by the context words of $T[i]$}

end if

end if

end if

end for

end if

end for

where the *Threshold* is not constant and it affects the results. After several experiments, the value of the *Threshold* that gave the best accuracy was greater than 0.7, where 0.7 represents the percentage of the two matched words context words' frequencies as described in algorithm 3. One more case that we have included in our matching algorithm is of some words which have repeated letters within the same word. Extra characters within certain words are used to place emphasis on the words or emotional states described by the words. For example the word "Goooooooal" where "Goal" is the root of the first one. Thus, a letters padding process has been implemented to solve this problem by keeping reading the repeated letter until reaching the next in the target (base word). Then, the resulting word will be combined but only after checking the context as stated in algorithm 3.

3.4.5 Bag of Informal Words

Non-standard words are more frequently occurring in social media. The easiest proposed way to describe these words meaning is by linking them to some formal or known words. Several words have been shown as linked to the keywords (anchor) as described in table 3.2. These words were classified under this anchor's umbrella and labelled with a numerical representation as described in the word co-occurrence section 3.3. TF-ICTF produces this unique representation for each bag according to the description in the TF-ICTF section. On the other hand, the word matching technique decides on the divergence of each word in a specific bag. Hence, two factors will decide this divergence decision : the first one is the word formation from a lexical perspective, and secondly, a word contextual perspective with respect to the keyword.

Multiple bags of words are classified under one topic depending on the annotated tweets, although many words might be common among several topics but the contextual word representation will decide to which topic they belong more closely. Table 3.5 shows the structure of the bags of words within each topic, using a matching process that will decide accordingly.

3.4.6 Experimental Results

The supervised learning shows very good results compared to the unsupervised learning. Several classifiers have been implemented on the features vector. Additionally, each topic was considered as a class and 30 % from the whole annotated dataset was selected for testing that had been chosen chosen randomly. This implies that the rest of the 70 % were the training data. The k-fold cross validation was implemented on the data before classification. Figure 3.18 shows how the data moves from the streaming stage where it was in the form of messy short text as a tweet. The resulting co-occurrence patterns are used later for the online phase where the proposed model predicts the class for the tweet using the word co-occurrence patterns and word embeddings.

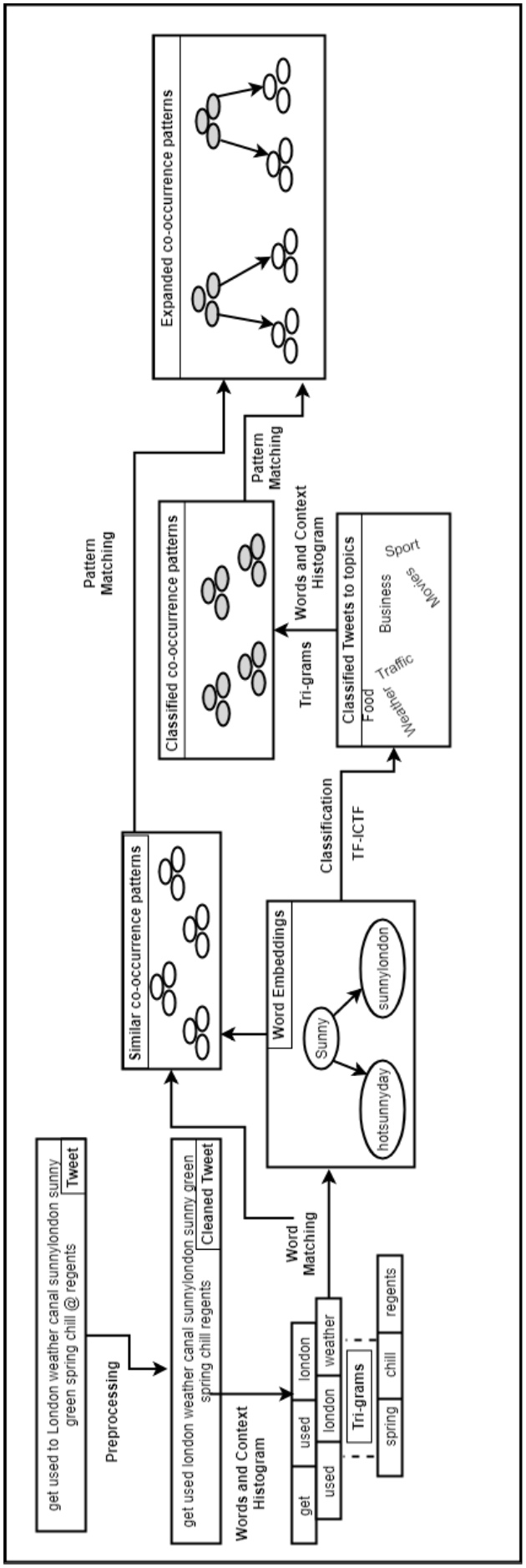


Figure 3.18: Data flow diagram within each stage to produce topic co-occurrence patterns.

After the tweet has been passed to the model, the cleaning process removes special characters and stop words as has been described in detail in the pre-processing section. Moving forward, the n-gram techniques process the tweet's text producing trigrams. These trigrams provide both information of the context and words' frequencies to build the words embeddings and the co-occurrence patterns at the same time. The co-occurrence patterns are not classified as these patterns will be linked to the classified ones afterwards. The word matching process helps in reducing the sparsity problem by minimizing the gap between the new words which have no record in the words embeddings. Additionally, the matching technique was applied on the co-occurrence patterns to provide strong semantic co-occurrences between words. These strong co-occurrences could be considered as probable patterns for the prediction process after they were linked to the classified patterns. The classification process takes place to produce classified tweets. The patterns again were extracted from the classified tweets as classified patterns. These patterns had been linked to the unclassified patterns to provide a broad space of prediction and reduce sparsity as mentioned before using the matching technique.

Table 3.6 shows three examples of tweets for each topic selected randomly from the annotated dataset that has 30,000 observations (tweets). The chosen anchor words can be shown against each tweet on the same row. These words were detected according to words' co-occurrence patterns as described previously. The PRSTM was trained according to the TF-ICTF features vector and the R value in sections 3.5 in 3.3 of these anchor words.

Topics	Tweets	Co-occurrence patterns
Traffic	poferriesfr great news makes worse im stuck middle lane surrounded trucks cant pull	stuck worse lane truck
	southernrailuk daily dose stress due train delays cancellations referendum anxiety enough	rail daily stress train delays
	Another weekend slow traffic leisure centre paid garden recycling harrow_ouncil	weekend slow traffic
Food	Mmmmm dinner time Our new Chicken Livers crispy pancetta almonds toasted brioche	dinner chicken crispy toast
	nice lunch friends yesterday corona lime lunch pub summer sun red lion	nice lunch lime pub
	oyster shucking masterclass amp course meal w paired drinks yes pls clubdandd	course meal drink
Bussiness	Stock Market Investment Like game Chess Focus present planning next strategy gt gt NTSLequity	stock market investment planning
	Euro depression deliberate EU choice says former Bank England chief via telebusiness Should know	bank bussiness
	Remarkable Email Marketing Tips You Need Implement Right Away via HuffPostBiz	market implement
Weather	Wind mph NE Barometer mb Rising Temperature Rain today mm Humidity	wind ne rising temprature rain today humidity
	creativemadhaus Karenanne Good morning ladies I hope good weekend weather horrible cold grey	morning weekend weather horrible
	cositohoracio really good thanks beautiful blue skies suffolk today amp getting warmer	blue sky today warm
Shopping	HeathrowExpress would amazing its light brown cloth hanging suit bag Inside suit jacket pair jeans	cloth hanging suit bag jacket jeans
	adidas techfall available online instore ever seen boxing wrestling boots sugarrays	online store boot
	check uk size womens azzurra steampunk brown suede ankle boots gold detail ebay	size brown seude boot ebay
Movies	KerryInTheCity Hiya lovely luck films LMK cos get out X	lovely film
	StarWars Rogue One A Star Wars Story Underground armiesamp super troopers Released later year NewMovies	story release movie
	Help shape future Cinema Guildford Vote films youd like see CGi Autumn here nhttps	cinema film autumn
Sport	Why shouldnt Gino amp Scott want aim more turgid football pts last games Thank Quique I back Gino watfordfc	football game
	bmsleight We big game tomorrow prot Ranieri Huth Youll get one day	big game
	Our grass looking fantastic ready junior football tomorrow se selkent football	grass looking fantastic
Not English	drscratch donc les pi sont aux normes cest juste la paperasse qui mal faite je r raison de plus pour regarder de pr	donc les pi sont cest mal qui faite je mal
	La situa oggi non sono andato palestra sono grasso e povero con voglia di pizza la mangio	oggi sono andato palestra grasso
	Vampida Jajaja te entiendo Pero ir todo bien ya ver	vampida te entiendo todo bien
Event	My memorable Eurovision yrs back mates Bottle vodka amp trying But English subs	memorable Eurovision bottle vodka
	Great night Yellow_omedy Thanks chrisogle And huge thanks And huge thanks AndrewCarberry croxley	great night comedy
	Last night I went see Twelth Night grassrootsLON I thought entertaining I thought cast excellent	last night entertain cast
Other	So much camera zoom its like trying video call parents	camera zoom video call parent
	Pink white classic colour combination Spiral Design makes arrangement contemporary stylish	pink white classic design stylish
	One revealing days life travel growth parkour journey pilgrimage Big	revealing travel growth image

Table 3.6: Chosen anchor words from tweets according to TF-ICTF and co-occurrence patterns

Furthermore, several words were removed from the process because no co-occurrence patterns could be detected. On the other hand, anchor words in table 3.6 show how the selection of these words could represent the whole tweet. The selected anchor words are not exactly as mentioned in the tweets, for example, “drink” which is mentioned as "drinks" in the tweet and “market” that appears as “Marketing”. This is because of the word matching that links any chosen word to the anchor word as previously described in the embeddings section.

3.4.6.1 Results with TF-ICTF

The experimental process went through several testing stages to challenge our proposed model. The first stage was by using the traditional algorithm (TF-IDF) to represent tweets' words as features vectors to be compared against TF-ICTF and then forwarding the features to a classifier. The results where not as good as expected and as described by several researchers like Hong and Davison [12] who made their empirical study of topic modelling over Twitter and came up with the same conclusion. The reason for that is that the TF-IDF algorithm considers each tweet to be a document. The huge increment in the tweets count produces *tf* values peaking with no effect from *idf* and its value cannot balance the resulting product value (refer to equations 3.6 and 3.7).

Keeping the word's value nearly consistent over time was the main reason why we developed the TF-IDF to our new version and called it TF-ICTF. The consistency in words' features (values) needs to be updated for the words frequencies which will be reflected in the co-occurrence patterns that might be changed over time as well. The sparsity problem will be discussed in every single testing step. The changes might affect words with no information where the words matching technique will take place to find the nearest word depending on context and similarity metrics as described in the matching section.

The testing process for this stage came with an accuracy using equation 2.1 of 24% overall with the use of TF-IDF as shown in table 3.7. The accuracy increased by using the TF-ICTF to reach 44.8% . This accuracy did not satisfy our ambition to reach the best result. The representation of each tweet was traditionally taken as the average of the words within each tweet then passing the results to the classification stage. The comparison in table 3.7 of using TF-IDF and TF-ICTF on the annotated dataset shows the difference described earlier.

Classes	**TF-IDF**	**TF-ICTF**
Traffic	35.34	53.971
Food	16.567	34.67
Business	24.231	32.561
Weather	39.289	58.324
Shopping	15.654	36.541
Movies	25.672	35.56
Sport	20.873	47.879
Not English	16.345	38.783
Event	10.64	38.56
Other	19.763	48.398
Accuracy	**24.04626**	**44.84075**

Table 3.7: Comparison between the accuracy of using TF-IDF and TF-ICTF on the annotated dataset.

The classifiers that were applied on the dataset using the previously mentioned features TF-IDF and TF-ICTF show different results. We took the highest result and show it in table 3.7 and the results of the whole classifiers on both feature types shown in figure 3.20. The SVM polynomial classifier shows better accuracy with TF-ICTF compared to the LDA classifier with TF-IDF which shows the best accuracy.

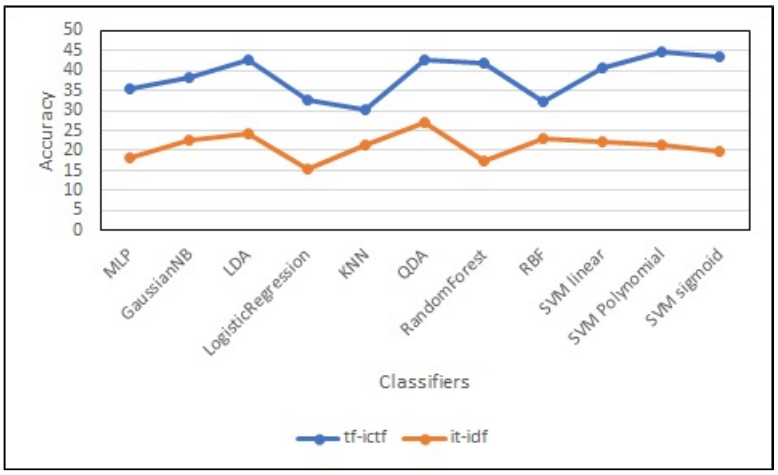


Figure 3.19: Accuracy of classifiers applied on both features TF-IDF and TF-ICTF

3.4.6.2 Results with word matching

Data sparsity is one of the biggest problems in natural language processing. Words with no information in the embeddings might cause bad results regardless of what classifier is used. Word2vec [19], Biterm [68] and many other topic modelling techniques tried to reduce the sparsity impact on prediction accuracy. We have addressed this problem by linking words to similar ones by a modified matching technique which depends on the context. Word embeddings provide this information that shows the degree of divergence (similarity) between any given word and words in the embedding. The matching technique may be time consuming for a huge amount of text but not for a short single sentence which took less than millisecond to find its place within the embedding. We have mapped these words to the nearest similar ones using the bag of informal words as has been described in the informal bag of words section 3.4.5.

Our modified matching approach depends on two factors: the first one using the state-of-the-art Approximate String Matching Algorithm [86] which we found more computationally expensive but more accurate than cosine similarity [87] as Goyal et al. demonstrated on their research [89]. The second factor is the similarity between the words of context for the two words in the similarity process. Therefore, we took these factors values changes to check if there would be an effect on accuracy. We tried each factor individually on the results from the TF-ICTF algorithm. The accuracy decreased at the beginning when we set the word's matching threshold to 10 % as shown in figure 3.20, then started increasing until it peaked at 61.671 % on a 60 % matching threshold value. It then starts to decrease afterwards. The reason for the very poor accuracy is due to aggregating unrelated words all together as the threshold is so small. For example, the word "broad" is similar to "broccoli" by more than

10 % as three letters from both of words are similar. The unreasonable threshold value will mix words from many classes forcing the classifier to produce inaccurate results. On the other hand the accuracy starts decreasing when the threshold value increases from 70 % as we found that the words returned to the initial state when each word gets its own feature value. The classification accuracy returned to a similar one to that described in the previous section .

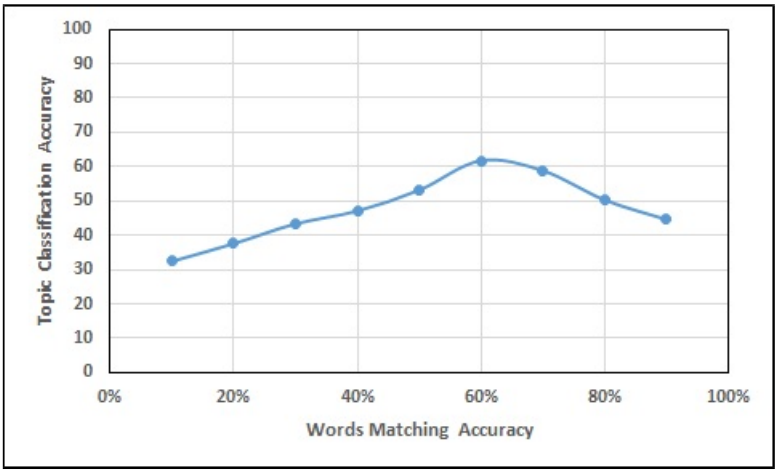


Figure 3.20: Accuracy of classifiers applied on the annotated dataset with TF-ICTF features and different word's matching threshold similarity value

Context words matching is the second proposed factor that shows greater impact on topic classification accuracy results compared to the first factor. Increasing the context words matching threshold gradually shows how the topic classification accuracy is affected. Figure 3.21 shows how the threshold of 10 % for context matching negatively affects the accuracy, reducing from the initial 44.8 % as shown in the previous section 3.4.6.1 to 28.91 %, then increasing gradually until it flattened between 60 % and 90 % matching threshold with an accuracy around 66 %. We investigated this behaviour of accuracy along the whole increment process to find there are some similar context words when the threshold is low. The resulting aggregated words reduce the boundaries between classes, resulting in lower classification accuracy. Thus, we increased the threshold reaching the best accuracy, although, this accuracy was still not good compared to the final accuracy that we got after applying both of the factors, which was 78.53 %, where the first factor (word matching threshold) was 90 % and the second one (context words matching threshold) was 70 %. This was additional to adding the condition of finding some words in long trending ones within tweets, as described in the informal bag of words section 3.4.5.

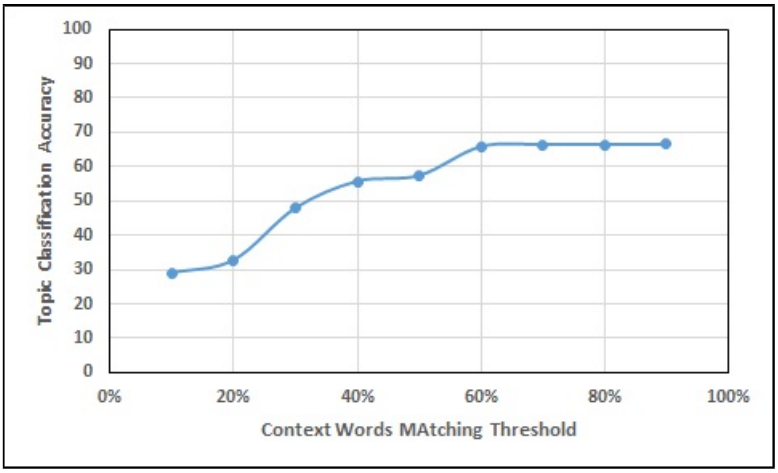


Figure 3.21: Accuracy of classifiers applied on the annotated dataset with TF-ICTF features and different context words' matching threshold similarity value

Moreover, several words did not produce a co-occurrence pattern as the word's frequency and the context words frequency were not co-related as described previously in the word co-occurrence section 3.3. These words had been classified within the word embeddings and the results of finding anchor words with the patterns within each tweet is shown in table 3.6. This step increased the classification accuracy, reaching 89.05 % as will be shown in the final results section. This accuracy increased after setting the R value to 5 using equation 3.5 which was described in detail in section 3.3. One more change can affect the accuracy, which is the change in the word embeddings over time, which will be described in detail in the following section.

3.4.6.3 Results with the Embeddings change over time

Changes in language affect most natural language processing tools. New words appear every single day and some of these words may be related to a specific event like a football game, or a new word that people like to use, for example, "field" and "stadium" are different words but the first one is more commonly used nowadays compared to the second one which is used to refer to a the place of the football game. We investigated our dataset to tackle these kind of changes by creating word embeddings during the collection of tweets at different times. We took three months for tweets collected from year 2017 and a similar time window from 2018. The comparison was made according to the similar words in both embeddings. The difference was huge as we found that 32 % of the words had been changed. New words came up and other words stopped occurring. Moving forward in this investigation, a word matching technique was used to find similar words from both embeddings to minimize this gap which is reflected on results in terms of sparsity. The difference had been reduced to 18.7 %, then we applied our model to both embeddings to check the accuracy. The difference between the two results was about 10 % in total for the ten classes of the annotated dataset described in the third point in the dataset section. Figure 3.22 shows the difference in accuracy because of the words change over time.

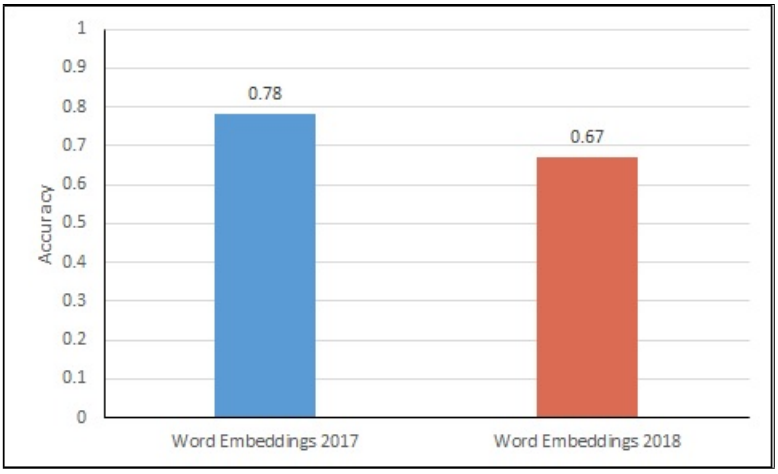


Figure 3.22: The change of accuracy when different embeddings from same dataset used in training

Therefore, keeping track of the changes over word embeddings will help to fill this sparsity gap with the new words that keep appearing. The final results were found to be reasonable after this intensive analysis. The results have been evaluated using several techniques and against state-of-the-art algorithms, as will be shown in the final results section.

3.4.6.4 Final Results and evaluation

Several methods used character-based methods and neural networks such Vakulenko et al. [17] but they faced one of the hardest challenges in NLP which is the annotated data size. Vakulenko discussed one of the major drawbackes in using neural networks were tha traditional supervised techniques outperforms the neural networks techniques in this challenge. Additionally, neural networks suffers from black box issue as the area of modification on an algorithm is very narrow unlike the traditional supervised techniques. We have used the supervised technique for these reasons as we have only 30 k annotated tweets.

As a result, the several stages of the PRSTM classify tweets as per the related topics using the detected patterns from words embeddings, built from around 20 million tweets, with respect to time and words' frequency changes. Thus, we have to evaluate our approach. The next sections will show how much the PRSTM is significant and what the limitations are that could be addressed in the future. k-fold cross validation was used with k =10 to produce a better evaluation of our model's performance. Furthermore, a comparison to other models will be given, describing the differences against our approach.

The k-fold cross validation was classified into two types : exhaustive and non-exhaustive cross validation. The first one roles are committed to divide the observations into even folds (sets) of training and validation. The observations in each validation set should appear only once during the whole process. Thus, the number these sets will depend on the K number that defines the number of folds.

The challenge that was considered in this process is how to make even number of observations in each class. Accordingly, we will have unbalanced datasets. The proposed solution in our case leaves us with two choices:

The first proposed solution is to divide each class observation on the number of folds

leaving a remainder of a maximum $K-1$ observations for each class. As a result, the total number of uncounted observations within the whole process will be $K*(k-1)$.

The second proposed solution is a repetition of the remainder samples for a padding purpose to complete K number of observations, either by choosing random samples from the remainder or depending on the observation index. Both of the suggestions will produce a number of repetitions between 1 to $K-1$ with a maximum total number of observation being $K*(k-1)$.

According to the above solutions, The number of added or remaining observations will be a maximum of $K*(K-1)$. If we proposed K to be 10 for example, this will leave us with 90 observations maximum that will never selected and could be considered as an error. As we have 30,000 observations, the remainder percentage will be 0.003 which is a percentage of the error in case these observations do not work in training as they are supposed to do. As a result, it is a very small error ratio that could be accepted, especially given the fact that it is proportionally divergent with a number of observations. The following figure shows how the K-fold works with the above described hypothesis.

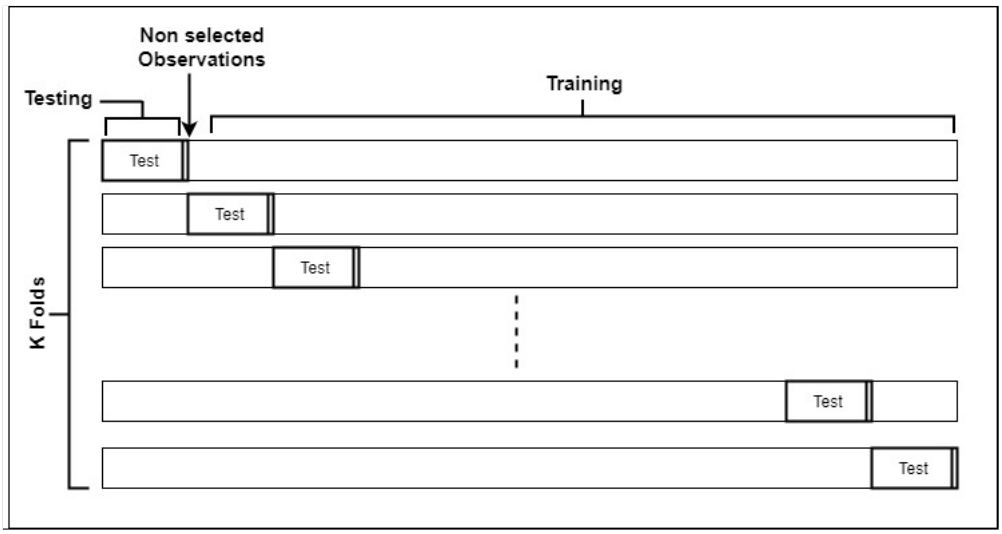


Figure 3.23: K-folds Cross Validation with the error

The results of the 10 folds cross validation for 11 classifiers is shown in the following table:

Classifiers	Fold1	Fold2	Fold3	Fold4	Fold5	Fold6	Fold7	Fold8	Fold9	Fold10
GaussianNB	0.784501	0.794933	0.796423	0.740238	0.78152	0.796423	0.78301	0.799404	0.796423	0.659314
LDA	0.82772	0.821759	0.796423	0.844769	0.830462	0.818778	0.837705	0.796423	0.818778	0.829553
RBF	0.796423	0.757914	0.734423	0.746557	0.787481	0.766423	0.78152	0.799404	0.724123	0.702534
SVM Polynomial	0.830462	0.800894	0.796423	0.822504	0.794501	0.840894	0.790656	0.797914	0.803875	0.751669
SVM Sigmoid	0.705365	0.730894	0.696423	0.672355	0.691952	0.629404	0.666617	0.703875	0.650894	0.669747
SVM Linear	0.731133	0.726662	0.796423	0.784918	0.790462	0.816662	0.796423	0.743875	0.706662	0.689121
KNN	0.812817	0.821759	0.856423	0.842832	0.850462	0.817288	0.835991	0.797914	0.817288	0.869747
MLP	0.833681	0.8307	0.796423	0.843428	0.790462	0.8307	0.834501	0.78152	0.8307	0.781043
Logistic Regression	0.700894	0.760656	0.697914	0.700894	0.721759	0.796423	0.644769	0.721759	0.700894	0.696423
Random Forest	0.699404	0.666617	0.703875	0.699404	0.657914	0.636423	0.746557	0.797914	0.676662	0.696423
QDA	0.850894	0.860656	0.837914	0.820894	0.840894	0.796423	0.862504	0.830894	0.809642	0.856423

Table 3.8: The accuracy of 10-folds Cross Validation for 11 kernels

The F1 Score had been chosen as the score measure for the classification accuracy prediction. Nine kernels had been applied separately to the feature vectors. The kernels are as follow: Radial Basis Function (RBF)[90], three Support Vector Machine (SVM) kernels (Polynomial, Sigmoid, amd Linear) [91], K-Nearest Neighbor(KNN)[92], Multi-Layer Perception (MLP), Logistic Regression, Random Forest, Gaussian Naive Bayes (GNB), Linear Discriminant Analysis (LDA), and Quadratic Discriminant Analysis (QDA).

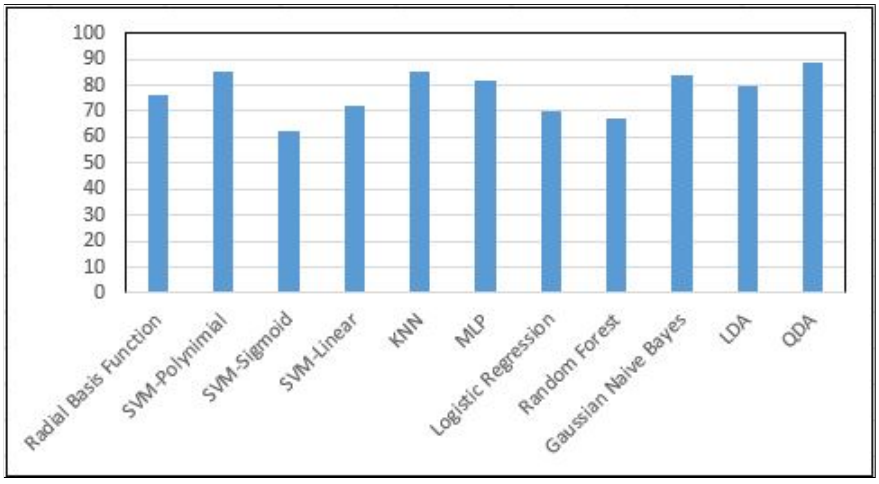


Figure 3.24: F1 Score Classification Accuracy Measure

The results in figure 3.24 show that QDA was the most successful kernel with 89% accuracy. Then SVM-Polynomial and KNN with 85% as well as the GNB kernel. The accuracy results are reasonable for noisy text without any training of the external database. We have done further analysis using confusion matrix to show the data results among all of the classes. The following matrix 3.25.

		Predicted										
Actual	Classes	Traffic	Food	Business	Weather	Shopping	Movies	Sport	Not English	Event	Other	F1-Score Accuracy
	Traffic	2869	20	43	15	17	0	11	7	0	18	0.9564
	Food	23	4832	50	5	4	12	30	17	25	102	0.9476
	Business	32	110	1086	21	121	24	10	23	35	38	0.7245
	Weather	95	21	16	5070	28	9	102	26	19	14	0.9389
	Shopping	56	27	28	61	2586	36	0	40	48	118	0.8623
	Movies	0	24	9	16	3	1706	5	10	2	25	0.9478
	Sport	28	16	9	40	0	1	2237	2	4	63	0.9324
	Not English	75	31	20	28	47	12	37	745	63	142	0.6215
	Event	51	38	37	19	4	27	59	21	908	36	0.7567
	Other	117	98	156	91	78	71	42	33	42	4672	0.8653

Figure 3.25: Confusion matrix with f1-score accuracy for the ten classes of the annotated dataset

The confusion matrix shows the TP values in pink color as the degree of Pink will be lighter and darker depending on the number of tweets classified. The Blue color shows the other misclassified tweets and where each tweet classified exactly. We have calculated two more metrics which are Precision and Recall using in addition to F1-score using equation 2.3, 2.4, and 2.5 accordingly for the ten classes based on the confusion matrix as shown in table 3.9

Classes	Precision	Recall	F1-score Accuracy
Traffic	0.857442	0.956333	0.9564
Food	0.926203	0.947451	0.9476
Business	0.746905	0.724	0.7245
Weather	0.944838	0.938889	0.9389
Shopping	0.895429	0.862	0.8623
Movies	0.898841	0.947778	0.9478
Sport	0.883143	0.932083	0.9324
Not English	0.806277	0.620833	0.6215
Event	0.792321	0.756667	0.7567
Other	0.89365	0.865185	0.8653

Table 3.9: Precision, recall and f1-score for the ten classes average accuracy

The accuracy of the three metrics was the result of average of the eleven classifiers applied on the annotated dataset. Moreover, the results will be evaluated against other models on the same dataset to show the differences.

Compared to Glove [36] and word2vec [19] after using the same annotated dataset in the dataset section, TF-ICTF shows better performance overall. On the other hand, Glove and word2vec took the lead in some topics as shown in Table 3.10.

Topics	PRSTM	Word2vec	Glove	TF-IDF
Traffic	0.9564	0.8097	0.7953	0.4567
Food	0.9476	0.7034	0.7812	0.3487
Business	0.7245	0.8078	0.6	0.5489
Weather	0.9389	0.9194	0.9426	0.6578
Shopping	0.8623	0.8657	0.5792	0.4378
Movies	0.9478	0.9536	0.8945	0.3576
Sport	0.9324	0.8543	0.8734	0.5634
Not English	0.6215	0.7813	0.7023	0.5367
Event	0.7567	0.8436	0.7378	0.2657
Other	0.8653	0.7392	0.8189	0.4367
Accuracy	0.8905	0.8166	0.7985	0.4172

Table 3.10: Accuracy comparison between word2vec, Glove, TF-IDF, and PRSTM

Compared to TF-IDF, the TF-ICTF algorithm shows better performance in all classes. The representation of the words using TF-IDF did not take into account the evaluation of words with respect to context. Additionally, TF-IDF is affected by the frequency, unlike TF-ICTF which works on a more coherent relation between context and words frequency when calculated over time. In other words, the TF-ICTF value for each word is less affected by the change of frequency for both frequencies of the word and its context words. Working within the PRSTM also provides less noise by using the proposed word matching stage with respect to context.

On the other hand, word2vec shows better performance on topics with lower representation in the dataset. For example, Business, Not English, and Event are 5% , 4%, and 4% respectively which is reasonable as we rely on words co-occurrence patterns. Thus, more patterns will provide better performance. Despite that, the average accuracy of our proposed model shows better performance. Likewise, compared to the Glove approach, PRSTM shows better performance in all of the topics.

We produced an unsupervised approach for topic modelling using similar PRSTM pipeline. Thsi had been just to show the performance will be using clustering algorithms.

3.5 Probabilistic Relational Unsupervised Topic modelling

The difficulty of machine learning from messy short text comes from the sparsity of information and the proposed model generality. When both of these elements are available, any learning task will be easier to implement. Therefore, the proposed supervised model (PRSTM) model satisfies parts of these difficulties to get better results. This section shows the unsupervised version of our proposed model. The other challenge for unsupervised approaches is how to link these clusters to a meaningful topic name not like the latent topic which every document could be reloated to a mixture of documents. This issue was addressed by several studies such as the study of Mu et al. [93] as they used several layers of documents aggregation. Deciding the topic in this study will be determined based on

the resulted phrases or terms that will be linked to the right topic using ranking scheme extracted using co-embedding information. Despite the cost of this long process, there still some limitations regarding the existence of such a topic information that could be linked to the extracted ones. The other limitation is the multi-topic nature of some documents that require further investigation. For these reasons we prefer the supervised approach.

After the data had been cleaned, the data was tested using unsupervised machine learning techniques that consist of multiple steps. The main goal of this stage is to cluster similar tweets based on feature vectors. The resulting clusters consist of tweets which should belong to the same topic. Figure 3.26 shows these steps from the data collection to the clustering stage where the resulting tweets are produced.

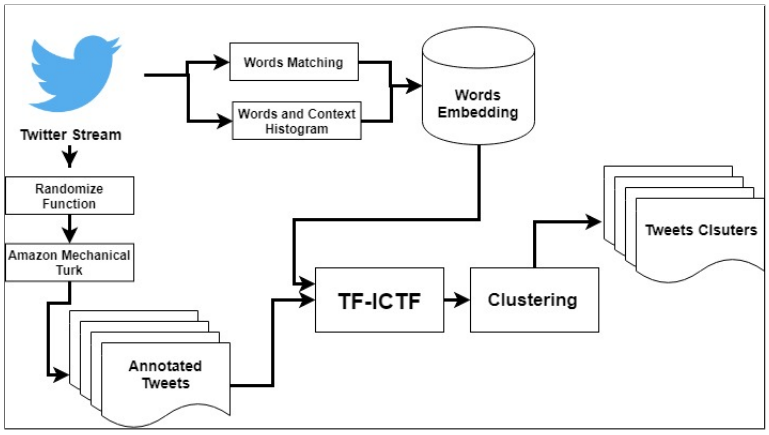


Figure 3.26: PRUTM with Unsupervised Machine Learning

Feature vectors are generated for each tweet separately then passed to the clustering algorithms. Two feature vectors are generated for each tweet. The two vectors are generated according to the *RF* from equation 3.10 and TF-ICTF from equations 3.9 and 3.8 that were described in the TF-ICTF section 3.4.3 earlier. The two features were created using the word embeddings after the word matching and cleaning processes had been completed. The clustering result using several kernels is shown in figure 3.27.

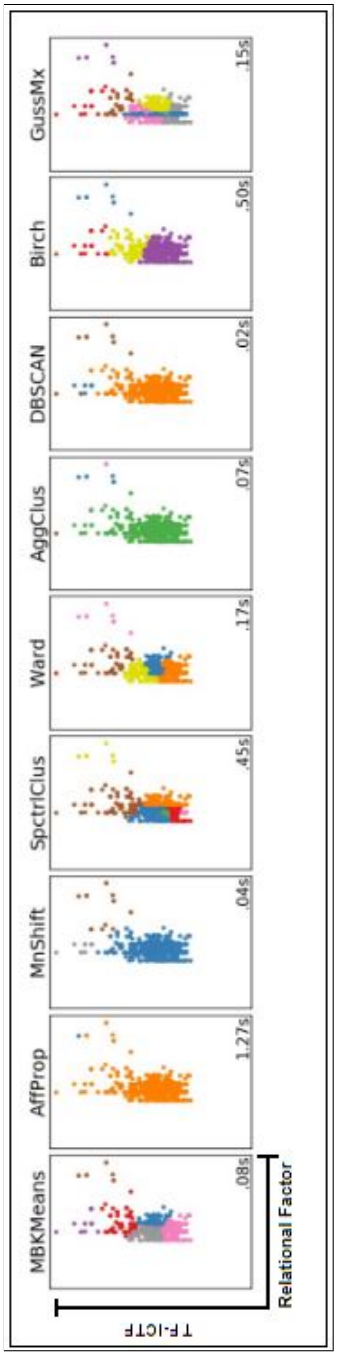


Figure 3.27: Clustering Result

The clustering algorithms had been applied to the annotated data from the Mechanical Turk that was described in the dataset section 3.2. The results are shown in figure 3.28 where several clustering algorithms were applied to the features vectors and they were: Mini Batch k-means, Affinity Propagation, Mean Shift, Spectral Clustering, Ward's minimum variance, Agglomerative Hierarchical Clustering, DBSCAN (Density-based spatial clustering of applications with noise), Birch (Balanced Iterative Reducing and Clustering using Hierarchies), and Gaussian Mixture. Accordingly, the accuracy of these clustering algorithms was not as googd as the supervised model's results. The highest accuracy came from the Gaussian algorithm as shown in figure 3.28.

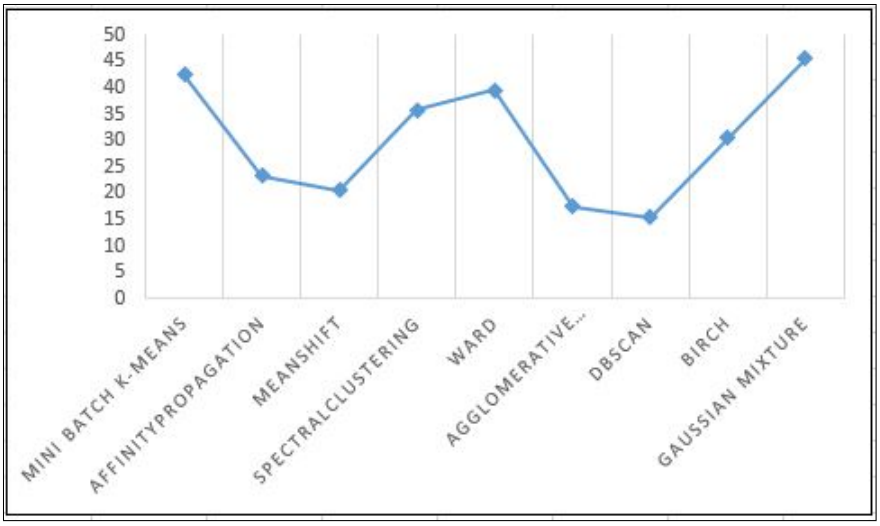


Figure 3.28: Clustering Accuracy Results

A purity accuracy measure was used for the clustering algorithms results, where N is the number of documents (tweets) and $M = \{m_1, m_2, m_3, ..., m_k\}$ are the clusters and $D = \{d_1, d_2, d_3, ..., d_j\}$ where k is the number of clusters resulting and j is the number of classes as d_j is the classification of m_k which holds the maximum value. Equation 3.12 shows how the purity measure is calculated.

$$Purity(m,d) = \frac{1}{N} \sum_{m \in M} \max_{d \in D} |m \cap d| \tag{3.12}$$

Unfortunately, the purity accuracy of the PRUTM model was only 0.41 which was as expected for noisy and messy data like Twitter. Therefore, we produced our PRSTM model which is a generalized and more accurate model compared to this unsupervised model. As a result, this model needs more development to handle clustering the noisy short text.

3.6 Summary

The world is changing so fast and people need more vocabulary to describe the new changes as the old words are not enough. These changes will affect all natural language processing aspects from the simplest to the most complicated approach. Thus, tracking these changes will make it easier to adapt to some point and produce better performance. From our analysis we found that people's language is influenced by the language in social media and the environment they live in. Therefore, words' co-occurrences are the only constant thing in the whole equation with a context that might change accordingly.

We built our proposed approach on the previously mentioned hypothesis after finding a simple co-relation between the changes of words with the changes in context frequencies. These changes helped us to track many changes and find relation between similar ones to reduce word sparsity as much as we could. Thus we used a modified words matching algorithm to find the similarity between words according to the context words. This matching process was created after building the main core that holds millions of relations between words streamed from Twitter and which had been classified as one of the most messy noisy

text environments. Therefore, we treated all words evenly without any predefined parameters or any auxiliary information.

Finally, the training process makes it possible for us to test our approach by the annotated data and classes that had been determind by people as shown in the annotation process in appendix B. The annotated tweets enable us to test our approach's classification capability with a challenging cross validation. The results were good compared to other state-of-the-art approaches as we come up with a 89% total accuracy, especially the words' features as we developed the TF-ICTF from the well known TD-IDF algorithm. Our approach based on word co-occurrences and word frequency probabilities as it defines the core of our work on text.

For future work, we plan to build our own classifier that could work unsupervised, with the ability to predict the topic depending on the words co-occurrences only. This might be a difficult task with high cost but the first step has been made by this approach moving towards a more generalized approach.

Chapter 4

Probabilistic Named Entity Recognition

4.1 Introduction

Extracting information from text, such as locations or organisations, is not easy without some prior knowledge [94], although humans can guess what this information refers to just by reading it or maybe by taking context into account [95]. When considering this from the point of view of a software system trying to identify such entities, the problem is more challenging. One important element is any prior knowledge before locating the information within text. For example, "London" could be known as a city without any context as it is well-known and recognised, but an area called "Al-mansour" which is in Baghdad could not be classified as a location without prior knowledge. Therefore, the training data is needed to show the contextual information which leads to entity identification. As we are interested in identifying entities such locations from tweets, typos and other problems related to informal text also need to be addressed by a named entity recognition (NER) method. As in the previous example, "lndn" is an informal new word for London which would not be identified as place within text directly; it is not similar to the formal abbreviation "LDN" used for London.

NER for social media data has been gaining importance as useful information can be extracted for a range of applications, and many studies have targeted this area like TwitIE [96], Named Entity rEcognition and Linking (NEEL) [50], and Stanford NER nested and baseline [47] [97]. The main concept of NER is to detect entities within the text such as places, streets, cities, and attractions. Given the issues with short and noisy text that social media postings tend to be, we have developed a new named entity recognition method in which the main concept is probabilistic and depends mainly on word embeddings and word co-occurrence patterns. The proposed Probabilistic Named Entity Recognition (PNER) method is designed to be a generalised approach to finding entities within the tweets' informal and often noisy text. The prediction phase (online phase) can predict the designated entity without the need for getting through several stages, like part of speech tagging, which is time consuming and also an essential stage in most of the state-of-the-art and standard named entity recognition algorithms. We targeted places and streets as part of this work, but PNER could be applied to

other entities as part of future work. The proposed model was trained on the results of named entity recognition standard algorithms which are well-known and widely used: Stanford NER [98], and ANNIE (A Nearly-New Information Extraction system) NER by GATE [1] from the TwitIE open source API developed by a team from the University of Sheffield, UK. These algorithms annotate the streamed tweets by finding the entities then forwarding them to the entity pattern creator stage.

Parts of this chapter is part of our published paper [99]. The rest of the chapter is organised as follows. The next section describes our proposed model framework (PNER) and shows the functionality of each part of the model. This section includes a discussion on the datasets that have been collected for both training and testing purposes. Next we describe the training and testing stages in respective sections. An analysis of the experimental results and evaluation follow. Finally, a summary of the work presented.

4.2 PNER Framework

Our proposed PNER framework is illustrated in figure 4.1. The framework includes two phases: a training (offline) phase and a real time (online) phase.

The training phase starts by passing tweets to the NER standard algorithms to find the entities within tweets. Then the entities are taken in the form of trigrams including the context words (one word before and the one after) as it they have appeared in the tweet. These tweets will be referred to as candidate tweets and the trigrams will be called co-occurrence patterns. Similar patterns for the same entity will be gathered from the entity pattern embeddings in the form of trigrams along with the frequency of these patterns with respect to each entity.

The entities patterns will be expanded by using the word embeddings. These word embeddings have been created based on prior work on the Probabilistic Relational Supervised Topic Modeling (PRSTM) method as described in [100]. The main concept of PRSTM is to build co-occurrence patterns for each single word from a huge number of short text samples (tweets). These patterns were formed as trigrams as each word will get its own list of context words with its frequency in the whole corpus. In addition, any context word which occurs with the main one will have a frequency which defines the times that the main word and the context word occur together. Please refer to [100] for further details on the PRSTM method and creation of embeddings.

This word embedding will expand the patterns for each entity by just simply locating it within the corpus. Moreover, there is no need for the same entity to be looked up again which minimizes the search time. Finally, non-standard formats of the identified entity will be located based on the context matching that will be described in the matching process.

All the candidate tweets which include the standard and non-standard (informal) entities were saved for testing purposes. The informal entities were saved in a form of an informal bag of words for each standard format entity identified by NER. The main purpose of using this bag of words is to convert the non-standard identified entity to its standard format as well as shorten the process time. If no entity is identified in the tweet, the word embeddings generated by PRSTM [100] will be used to find if this tweet is a candidate for an entity to occur in or not. PRSTM is a supervised topic modeling method which can classify tweets to

certain classes based on co-occurrence patterns and word embeddings. Therefore, we used this method to indicate whether this tweet is a candidate tweet or not. If it is and the NER has not detected an entity, the tweet will be converted to trigrams and passed to the matching technique in the PNER which might then find the informal entity. The process just described from the input of the tweet until the matching procedure are the steps of the online phase. Figure 4.1 illustrates for the proposed framework.

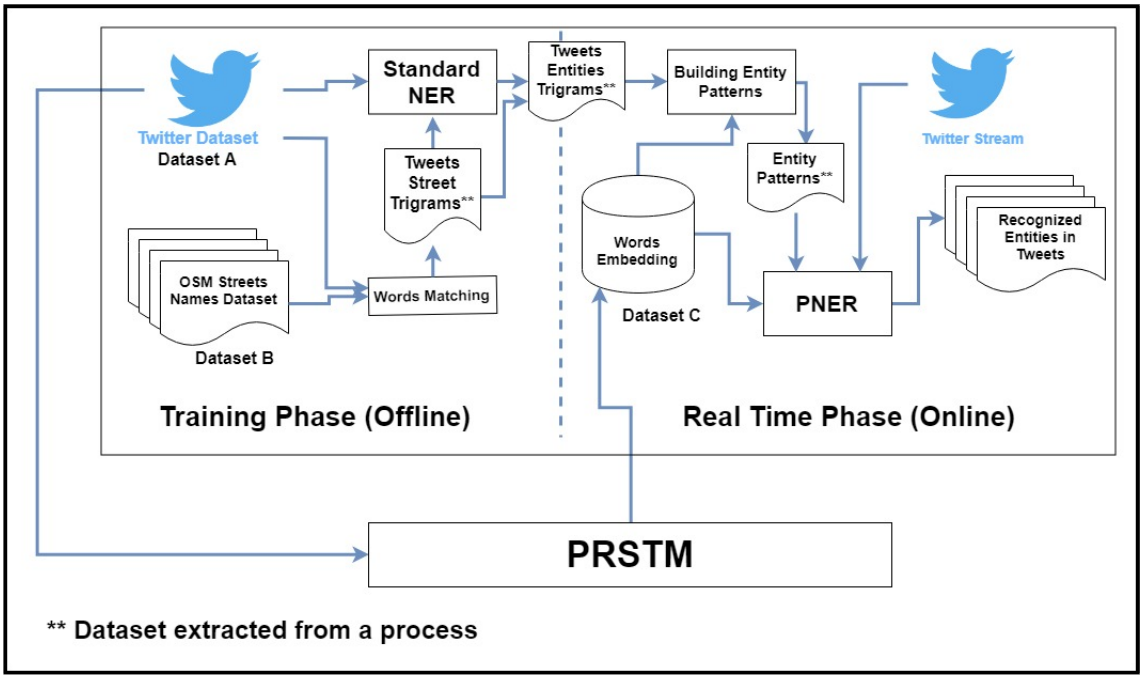


Figure 4.1: PNER Framework

Different types of entities have been tested with our approach such as people, places, street names, and organisations. The entity detection process, for example, in the PNER model basically starts with word matching, where each tweet that includes any street name will be passed to the PRSTM (the street names were taken from the dataset B provided by the OSM as shown in figure 4.1). The patterns of each detected tweet are generated by the PRSTM and passed to the PNER step, with the tweets being provided by the Twitter Dataset A. Subsequently, the patterns of the new tweets are extracted and compared to the trained patterns.

One of the major differences between our proposed model and standard NER is the absence of a POS (Part Of Speech tagger). PNER does not rely on a POS, either in the training or in the recognition. As a result, the processing time will be much faster and less complex as will be shown in the experiments. Moreover, capitalization is not a matter for concern in PNER where the pattern of the tweet will play the major role in detecting entities within the tweet's context. This address text informality which is one of the main motivations that led us to create PNER as described in the introduction. The results will show the potential of finding informal (non-standard) entities by training on formal ones.

4.2.1 Dataset

Several studies discussed the use word tag in general to disclose the named entity within text. Although, it is a very successful and widely used traditional approach. Sang and Meulder [101] published the NER task of CoNLL2003 which describes this previously mentioned approach. The challenges are all related to language variation and Part Of Speech (POS) commitment to detect named entities. Therefore, we produce our non-traditional techniques that relies on co-occurrence patterns extracted from our generated word embeddings.

Detecting entities within text can be easy if the text is formal, as traditional NLP tools typically work well on such text. Thus, our challenge was extracting existing information and co-occurrences from text which is defined as informal and noisy. We have not used any auxiliary information, like Wikipedia, we just used raw Twitter text streamed using the streaming API in Python programming language [69] with the help of OSM information to detect the entities within the Twitter text for training. The streamed Tweets have been stored in CSV (comma-separated value) files. The datasets that we have used are described below:

1. **Dataset A**: We collected a dataset of tweets depending on a set of keywords related to traffic (e.g. Accident, Crash, Traffic, Congestion). These tweets were collected from all over the world without specifying location, but just using words typically related to traffic while we also collected the time that a tweet was created. This dataset was collated from 17th February 2016 until 19th April 2016 with a total size of 40.9 GB (Gigabyte) and around 20 million tweets in total. Most of these tweets were written in the English language. Another Twitter dataset comprised tweets whose sender geo-location was identified as being London and had been streamed using Python language API [69]. This dataset was collated over nearly two years from 20th of April 2016 and until the 3rd of December 2018 and includes 146.2 million tweets, with a total size of 278.6 GB. These embeddings were reused in the co-occurrence patterns and tweets features vector, as will be explained in detail in the subsequent sections in this chapter. Both twitter datasets combined and used to create the word embeddings using the prospective of the PRSTM model [100].

2. **Dataset B**: We collated a dataset containing the street names for the London area through using the OSM (Open Street Map) [71]. Some of these names (entities) could also represent a person entity (person name) which is of particular interest in natural language processing, especially named entity recognition.

3. **Dataset C**: This is the word embeddings that have been created from the PRSTM method using dataset A which is then used in the online-phase as shown in figure 4.1.

4. **Tweets Entities Trigrams**: A dataset of entities which are annotated to several categories as place, street, organization, and person. The annotation process used both the NER TwitIE and Stanford algorithms with matching to the OSM dataset. Both NERs received the same tweets in parallel and the results of the annotation were combined to create this dataset; this is the output of the Training phase. The total number of entities detected within tweets was 5,641 in 3,532 tweets that was randomly chosen

from Dataset A. Figure 4.3 in the word matching section (Training phase) shows how many entities, by percentage, were annotated per class from the whole dataset.

By looking at the Twitter text, useful insights can be gained. We found for instance, that one of the special characters could play a key role in locating entities such as a location or person. This special character is "@" and it is typically used in Twitter as the identifier for the user's id. Moreover, it is used as an abbreviation of the word "at", as in the sentence "i am at my brother's house", but we found that in informal text it is used as, "at a place", as in "@London". Hence this could be a character that could be detected within the patterns that are related to either location or place which could be used by our NER approach, especially given that the main concept in our proposed model is probabilistic. Therefore, the entities patterns will include one special character that will be considered as a word in our method.

4.2.2 Offline Phase (Training)

The purpose of the Training Phase (Offline is to annotate the tweets from dataset A by finding entities like places, organisations, people, or streets and then forward the resulting tweets to the next phase, Figure 4.1. The training process depends on two standard methods: Stanford Named Entity Recognizer [98] and ANNIE [1]. Both methods applied on the tweets in parallel to identify entities. The detected entities by any of these methods will be considered, therefore, the results of both NER methods will be combined. NER also receives another dataset feed resulting from the matching process that includes entities in the form of trigrams as shown in the training phase in figure 4.1. The word matching receives the names of streets and places provided by OSM to look for them within the text and then forwards them to NER to be checked again. The resulting tweets are then structured in the form of (i) the text of the tweet as a list of words; (ii) the entity location in the list; and (iii) the entity category. This process facilitates classifying entities, and then finding the co-occurrence patterns as will be described in detail in the online phase section. The following table 4.1 shows a sample of the structure for the file that is forwarded to the online phase.

Index	Tweets	Location Pattern	The location position within tweet
1	theres nowhere else like London nothing anywhere viviene westwood happy star	[[u'like', u'london', u'nothing']]	[4]
2	Watching EurovisionSongContest good see Sweden using normal aka mixed size models	[[u'see', u'Sweden', u'using']]	[4]
3	Ive driven 80 mins hitting traffic Hampton Village Green	[[u'traffic', u'Hampton', u'Village'], [u'Hampton', u'Village', u'Green'], [u'Village', u'Green', 'null']]	[6, 7, 8]
4	For everyone thought Azerbaijan long lost Kardashian aint seen nothing yet Eurovision ARM	[[u'thought', u'Azerbaijan', u'long']]	[3]
5	occupy wall st goldman sachs finally united protect america donald trump	[[u'protect', u'america', u'donald']]	[8]

Table 4.1: The file structure which results from the training phase

The NER stage is now described in more detail to show how it works.

4.2.2.1 Named Entity Recognition

The named entity recognition task is not simple as the entities vary in type, structure, and length. For example some of them are just abbreviations like "LSE" as an organisation which

refers to “London School of Economics”, or could be a place such as “Iraq Republic”. The entity length in the previous examples could be one of the challenges that the state-of-the-art algorithms address as in “Iraq” or “Iraq Republic” as they are both the same entity but not all NER models could detect this. Ambiguity may be another challenge for any NER algorithm. For instance, the name of a person and street could be the same as in “James” as a person and “St. James Park” as a place; this is why context is so important. Lastly, capitalisation is another challenge: the first letter of a name should be in capital (upper) case to indicate that it is a name or a place. These challenges could pose difficulties for any NER model. For this reason we are addressing these problems in a more generalised model by making the chances of any word being an entity equal for most words. This might require more training and several stages before reaching this goal. We are therefore using well-known standard algorithms in the training phase, and the probabilistic approach based on the PRSTM model [100] which includes word matching and word embeddings as will be described further in this section.

The baseline model of Stanford NER [98] addressed some of the previously mentioned challenges by training on several corpora to obtain good results, and then by extending this model to produce the nested version which provides the ability to catch nested entities that are longer than one word. This was not easy without using the context, which is the main concept for most NER models. These models look for landmarks after converting the text to tags using part of speech (POS) taggers. The POS tagger will identify if a word is a proper noun, when the tag of the word will be *NNP*, or *NNS* for a plural common noun. There are several tags for nouns. The main problem in this kind of NER is capitalisation which makes it hard to decide if a word is a noun or something else. Following the tagger, the Stanford model uses the Conditional Random Field (CRF) features that represent these entities. The model was tested on several corpora in genetics and other Spanish ones, showing good results when compared to standard models with several entities like person, organisation, location, date, and number.

The second NER model that we use was developed by a team from Sheffield University and is called TwitIE [1], and is an open source information extraction pipeline. It is mainly dedicated to microblog text such as Twitter. Figure 4.2 shows the stages of this pipeline and the steps in each stage.

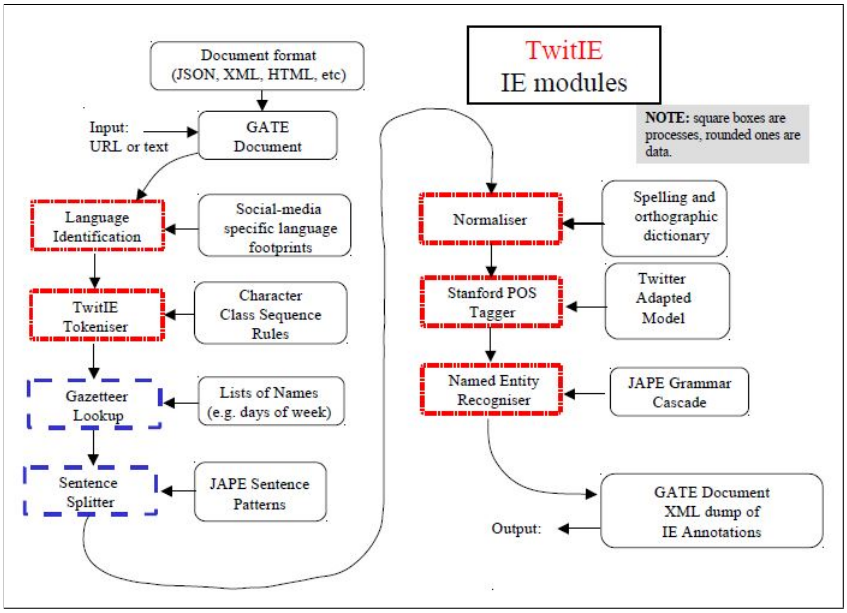


Figure 4.2: TwitIE information extraction pipeline [1]

The first step was identifying the language that the blog was written in as this is a piece of information that is part of each micro-blog structure. For example, in each tweet there is a section called *"lang"* which specifies the language. Thus, in our collected Twitter dataset A we have the English language in most of the tweets so this section will be set to *"en"*. Tokenisation is the second main step to produce a list of tokens to be processed as words, with an extra cleaning process to filter the words with no special characters or links. The Gazetteer Lookup is a matching tool that consists of standard lists collected from several corpora such as the names of days of week, names of cities, or places. Next is the sentence splitter which splits the text into multiple sentences to be ready for tagging using the standard Stanford POS tagger, after checking spelling and capitalization in the normalisation stage. Finally, both NER standard methods will provide the extracted information annotated according to several entities like names, places, organisations, etc.

The extracted dataset from the matching process in a form of lists of annotations will be taken with their locations within the tweet's text to be used for training purposes, as shown previously in table 4.1. Moreover, we included more annotated tweets using OSM street names and places [71]. The word matching stage in the offline phase in figure 4.1 will produce the tweets from matching the street names similar to the Gazetteer Lookup described earlier in the TwitIE NER.

4.2.2.2 Annotation process and Word Matching

the training, testing, and evaluation process requires annotated data to evaluate our proposed approach. This annotation process was made using three methods:

1. The first method was word matching as we have used the OSM dataset to match any similar entity in the tweets.

2. Using both standard NER to detect entities within the same selected tweets.

3. The rest of non-standard entity format was made by hand as these entities considered as part of the noise or the undetected entities by the standard methods.

Word matching is chosen for two tasks. The first task is detecting tweets that include specific entities to be forwarded to the NER stage. The entities are street names, locations, attractions, and other entity names that have been taken from the OSM [71] database which is available for public use. The entities are processed and produce additional word co-occurrence patterns for each entity that will be forwarded with the detected entities to the online phase. The resulting co-occurrence patterns with the tweets text will be in a similar form to the ones described in table 4.1. The second task of the matching algorithm is to find similar co-occurrence patterns by matching the context words of the entities and aggregate similar ones which will be described in the online section.

The dataset extracted from a process in a form entities trigrams was annotated to several categories as place, street, organization, and person. The annotation process was produced using both the TwitIE NER and Stanford NER with matching to the OSM dataset as shown in figure 4.1. The two methods applied in parallel to extract this annotated dataset. Figure 4.3 shows how many entities, by percentage, were annotated per class from the whole extracted dataset.

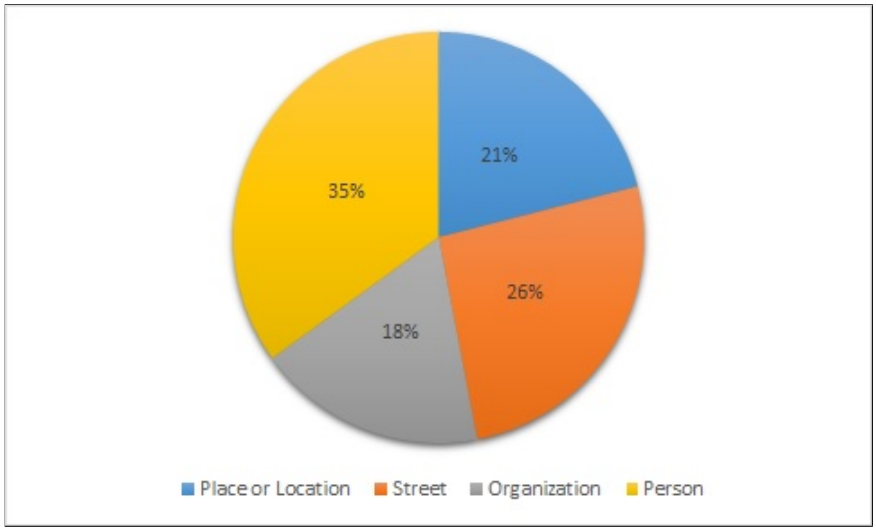


Figure 4.3: Annotated entities dataset pie-chart showing the classes percentage

Table 4.2 shows a breakdown of the annotated entities by percentages and the number of entities per class. Person was the highest recognised entity with 35% of the whole dataset with 1,974 entities recognized by the two standard combined NERs described earlier.

Entity Class	**Percentage**	**Number of Entities**
Place or Location	0.21	1185
Street	0.26	1467
Organization	0.18	1015
Person	0.35	1974

Table 4.2: Annotated Entities dataset classes using state-of-the-art NERs and word matching

Most of the "street" class resulted from the word matching process with the OSM dataset. These entities are not fully detected by the standard NER algorithms, but we include them to show the effect of the co-occurrence patterns on NER and how we could make use of this model to enhance the results. Finally, these entities will be directed to the online phase for testing and detection purposes.

4.2.3 Online Phase(Detection)

Detecting the non-standard entities in real time is the main purpose of this online phase. This will be accomplished by using the extracted (annotated dataset), word embeddings, and the stream of real-time tweets.

The core of our work could simply be explained by finding the standard format of the non-standard ones through co-occurrence patterns matching. The detection of the entities and the core of PNER is described in detail in the following sections.

4.2.3.1 Words Embeddings for PNER

The word embeddings section was described in the PRSTM chapter but this section will describe how to utilize it within the PNER model. As a start, Tweets from dataset A were streamed for to create word embeddings. The main task is to represent these tweets in a data-frame that could preserve words and context co-occurrences. We used the n-grams technique [82] where n-grams are defined as the contiguous sequence of N number of terms as it is used in statistical linguistics.

First, the tweets have to be pre-processed by removing the special characters (except the "@" as described in the dataset section), website links, and English language stop words. A list of words is the result from the pre-processing step with two special characters on both ends of the list to designate the margins of the tweet. We have used "&" as the special character on both ends of the list.

A trigram is the proposed version of an n-gram with a structure of three words for each tuple, the centre word and one on each side. Therefore, the "&" character will not go through the same process as other words in the tweet text. The use of trigrams enables to identify these kind of words by simply finding the percentage of the "&" characters before or after the word over the word's frequency. Moreover, the sequence of the words within the tweet will be kept as well for further analysis. Table 4.3 shows the result of the pre-processing step using trigrams. In addition, the co-occurrence frequency of adjacent words will be calculated within the whole dataset. This frequency is stored in the form of list which is linked to the word in the centre. As a result, each word will be represented in the embedding in the form of three vectors: one for the centre word and two for the words on both sides related to each single word.

For example, the trigram "[london | beautiful | day]" in table 4.3 will be stored in the embeddings as a centre word "beautiful" with frequency "13" occurred with "london" word "5" times, and with "day" word "2" times. The frequencies with the words will be stored as lists in the word embedding. The process of generating or updating the word embeddings is described in detail in Algorithm 2.

Entities Co-occurrence patterns
[&| good| morning]
[visit| scotland| friendly]
[getting| ready| &]
[london| beautiful| day]
[britains| got| talent]
[normality| america| &]
[people| florida| news]
[goodnight| everyone| null]
[meeting| London| last]
[&| austria| considers]

Table 4.3: trigrams generated from the pre-processing step

The algorithm takes a list as an input. The first list is the tweets lists as each tweet is a list of words which had been generated by the pre-processing step that has been described before.

The embeddings were utilized in PNER by extracting the word co-occurrence of any informal entity for use of context matching. Therefore, it had been created in general (i.e. not for particular words or entities) so that any entity might occur in any non-standard format, but the co-occurrence pattern will lead to identify this entity.

4.2.3.2 Word Patterns Matching

The word matching step relies mainly on the word embeddings and the words co-occurrences. The matching process depends on the context of any given word with other words. For example, if the word “London” occurs in the context of the tweet “London, here I come”, and the non-standard format of it “londoooon” appeared in a tweet “londoooon here I come”, the matching will depend on the context not on the word itself. These two, standard and non-standard formats occurred with several words in the streamed tweets. These words had been stored with a relation to each format as described in the embeddings section. Therefore, a simple matching step will find how these words could be related.

In addition, the informal bag of words from word embeddings consists of similar words that could improve the process of finding undetected entities by just finding the entity’s bag. For example, the word “London” bag of words shown in figure 3.17 has many words similar to it such as “LNDN” which is not detected by the state-of-the-art NER. This mechanism in our proposed approach will enhance any named entity model with less cost just by linking these entities to the right informal bag of words. An example of informal bag of words is shown in figure 3.17 in the PRSTM chapter.

The next section will describe how the informal bag of words utilised in the Detecting Entities patterns.

4.2.3.3 Detected Entities Patterns

The classes of entities vary in type and in length as well. Some of these entities are extended to three words making the NER process a bit difficult. The Stanford extended version [47] which is described as nested named entity recognition method together with the TwetIE NER was able to detect these entities as nested entities. PNER detected these recognized entities and fitted them into three patterns as shown in table 4.1 in the third example. Nevertheless, there are still some non-standard entities within the text that the standard NER algorithms could not detect. The informal bag of words will help to achieve this goal by creating a bag of informal words for the entities. Similar to the described informal bag in section 3.4.5 the resulting entities will be converted to features and looked for within the text and produce them as suggested entities to the NER. This step will be used just in the case of not finding an entity in a tweet, so the word embeddings will be used to find if the input tweet belongs to the annotated tweets class or not. Table 4.4 shows examples of entities taken from the annotated entities with the informal words related to the main entity.

Index	Entity	Similar Informal Entities
1	London	londddddoooon londonnnnn londonnnnnn nlondon londoooon londonnn
2	Paris	pariiiiiiiis parisss parsis parisa
3	bbc	cbc bbbc cbbc bbbbbcccc

Table 4.4: Informal Entities bag of words

As shown in Table 4.4, there are several entities that cannot be recognised by the state of the art algorithms, like “pariiiiiiiis” and “londddddoooon”. These entities had been aggregated in informal bags with the formal entities “Paris” and “London” consecutively. Moreover, the “BBC” entity informal bag includes several entities like “cbbc” and “cbc” which are organizations but not the same one. This demonstrates how context words are the main factor and how effective they are in terms of supporting Natural Language Processing methods in general.

4.2.3.4 PNER

Several named entity algorithms have common attributes to count on when recognizing entities, such as capitalization and part of speech tags. These entities need to be in a standard format to be recognized otherwise they will be either wrongly tagged as “NN” which stands for “Normal Noun”, for example, or even as un-tagged words. This informality has been tracked by our proposed model and has been named as an informal bag of words but for the detected entities. This problem of informality was addressed by using the word embeddings, and words matching patterns as shown in figure 4.4.

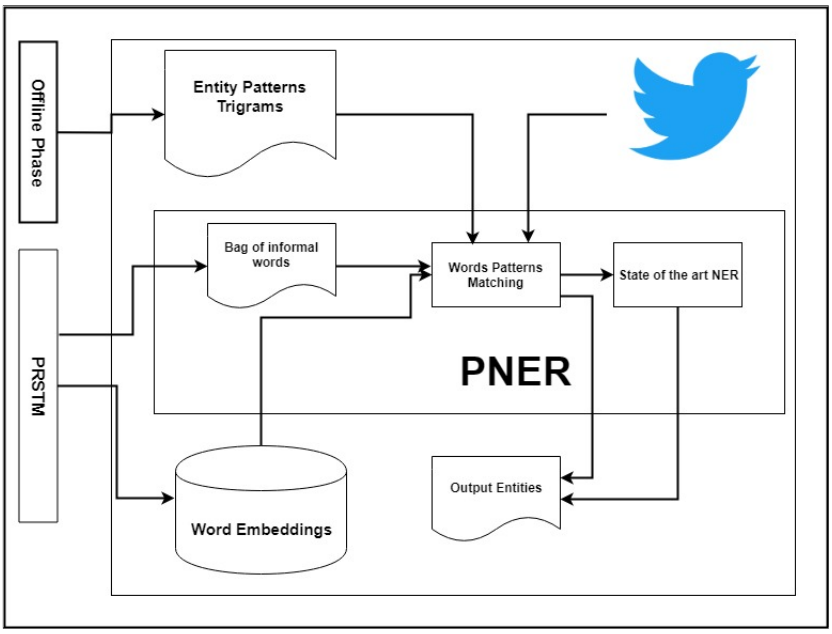


Figure 4.4: PNER detailed core with components

The streamed real_time tweets are directed to the word matching as well as to the informal bag of words and the word embeddings. The entity patterns will be the list of the destination matching as the comparison will be between the entities that have been detected by the NER standard algorithms. This step is more about finding all the similar entities and creating a new bag of words if not found in the informal bag of words that have been already collected. Therefore, these bags of words might change according to the new entities by checking the word embeddings if not found in the existing ones. After the informal word has been found and linked to the formal ones, the resulting list will be directed to the standard NER algorithm to be validated. Algorithm 4 describes the step of finding the standard format of the informal entity starting from the input of the word embeddings, entity patterns, informal bag of words, and the tweet and ending with the result of the formal entities related to the informal input entities.

algorithm 4 Looking up informal entity

Require: T as the list of word embeddings, Bow as the informal bag of words, EP as the entity patterns,X as the input tweet.

Ensure: Inf_Ent The list of formal entities for the input unidentified (informal) entities.

{After checking the words that are not detected by NER and removing the stop words, the rest of the text will be considered as the input X}

$L \leftarrow length(X)$

for $i = 0$ to $L-1$ Step 1 **do**

if X [i] not in Bow **then**

$L \leftarrow\square$ Bow [Index of the found word X [i]]

$wrd \leftarrow$ formal word in L {The formal word could be found using a special package in Python. Also checked by the words in EP}

$Inf\_Ent \leftarrow\square$ wrd

else

if X [i] not in T **then**

Compare patterns in EP with T [Index of the found word X [i]] co-occurrence patterns

if Compared pattern found match **then**

$Inf\_Ent \leftarrow\square$ wrd

end if

end if

end if

end for

return Inf_Ent

Algorithm 4 will get the word embeddings, informal bag of words , entity patterns, and the input tweet. The tweet will be pre-processed by removing the English stop words and converting it to a list of words. If any of these words are found in the informal bag of words, it will be added to the output list. Otherwise, the word will be looked up in the word embeddings to find its context words then look it up in the entity patterns. If it is not found, it will not be an entity.

In the next section, we demonstrate the application of our method with standard NER algorithms and the results obtained.

4.3 Experimental Results and Evaluation

Most formal entities are taken from Wikipedia [102] for example. This explains how some well-known entities can be recognised even if they are not capitalised as shown in figure 4.5 which is a snapshot from the TwitIE NER. The two locations “london” and “paris” were non-capitalised as locations but had been recognised by the NER algorithm because of previous training.

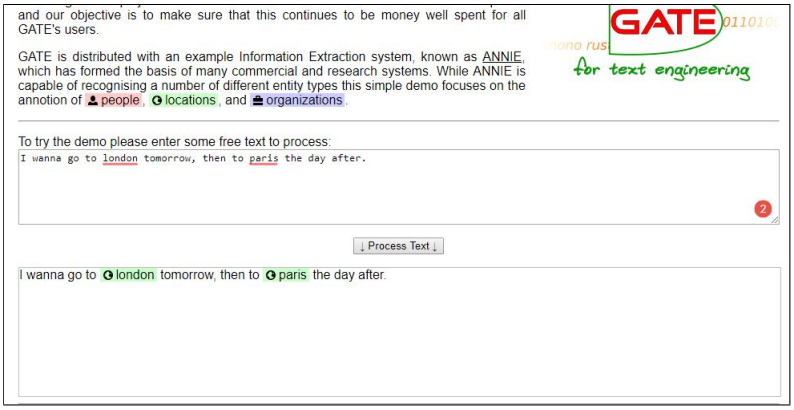


Figure 4.5: TwitIE non-capitalized location entities

Likewise, Stanford NER was able to recognize these entities having been previously trained on a corpus or by semantically searching for the word. On the other hand, our PNER model had been trained on the annotated dataset that was described in the dataset section. This training process makes use of the word embeddings that were explained in the word patterns matching section. A simple comparison of the resulting entities for both our approach and the TwitIE NER (that was designed and implemented by GATE) is shown in figure 4.5. Some of these streets names or places are not written in the standard form but PNER was able to detect them. The problem of nested entities was solved by our approach of saving the same entity in multiple entity patterns as shown previously in the file structure resulting from the training phase in table 4.1 in the third row.

	Tweet	Gate	PNER
1	\ud83c\uddec\ud83c\udde7 @ Big Ben London - Palace Of Westminster https:\/\/t.co\/xY8C9j03qN	Ben London-People , Westminster-Location	Big Ben - Location, London - Location, Palace of Westminister -Location
2	\ud83c\uddec\ud83c\udde7 @ Tower Bridge https:\/\/t.co\/dMnGEHPsqm	Bridge - location	Tower Bridge - Location
3	The park towards North is getting less nicer - the middle and the south of park is nice	-----	The park - Location
4	Early Commute\u203c\ufe0f @ Notting Hill Gate tube station https:\/\/t.co\/SBHshJD7YY	Notting Hill Gate-Location	Notting Hill Gate-Location, tube station-Location
5	Whitechapel Road 12\/3\/2016 #London #jacktheripper #urban #streetphotography #culture #eastend\u2026 https:\/\/t.co\/O9Kg48ZHr5	Whitechapel-Location, London-Location	Whitechapel Road Location, London-Location
6	Sol #itscoldoutside @ Grove Terrace London https:\/\/t.co\/y6OekrBdNb	Grove Terrace London - People, Sol - People	Grove Terrace-Location, London - Location
7	Breakfast is ready @ Moorgate station https:\/\/t.co\/praR3wRhhV	Moorgate - Location	Moorgate station - Location
8	#Traffic update: Congestion on the #A127 into Harold Wood at the Ardleigh Green Road junction	Harold Wood - Location, Ardleigh - Location	Harold Wood - Location, Green Road - Location, A127 - Location
9	Just seen a bloke jogging down the Kings Road in a Leicester shirt\n\nFuck off mate	Leicester - Location	Leicester - Location, Kings Road -Loaction

Table 4.5: PNER results compared against Gate

PNER is more generalised approach compared to standard in working with entities by removing the capitalization barrier and by finding the informal entity pattern and linking it to the informal bag. Figure 4.6 shows some examples of the detected informal entities by our proposed approach through trained patterns. TwitIE and Stanford NER are not able to recognize most of these entities, such as “liverpoolStreet” or “brighton” which are both in informal case. On the other hand, our approach has some limitations, for example depending on other named entity approaches for annotating leading to entity patterns generation.

Figure 4.6: PNER Results Samples

To demonstrate how our proposed method could work and add value to other methods, we compared our approach to both NER algorithms after applying the three approaches to the same annotated datasets that have been mentioned before. The four entity classes were evaluated according to the results in accuracy. The accuracy for both named entities shows better performance than our proposed approach as shown in table 4.11. The results were checked according to the true positive "TP" results to see how much the three approaches could achieve from the whole dataset. PNER shows the lowest accuracy in the "Person" class which was an expected result. The problem with names is that there are no specific entities that any person name could co-occur with, with high frequency. Thus, low frequency patterns cause low performance as they will confuse the approach in finding the targeted class.

The NER evaluation approaches are either Strict or Lenient. Lenient means it is not the exact entity which implies several interpretations such as synonyms or non-standard (ill-formed) entity [103] [95]. On the other hand, the strict (or exact) evaluation implies getting the exact prediction of the entity. Our approach is more on the lenient side rather than the exact as it all depends on the nature of the Twitter data which we have explained previously how noisy and non-standard it is.

In the Message Understanding Conference (MUC) [103] a proposed approach of evaluation was produced depending on the error that that is resulted by comparing the prediction results by the annotated data. This error could be calculated using five metrics. these metrics are correct (COR) which means that the prediction is totally match, Incorrect (INC) as the output does not match, Partial (PAR) when part of the annotation is similar, Missing (MIS) when the results are not recognized, and Spurius (SPU) when the output is not from the annotation data as might be new class.

By reflecting on these metrics the precision and recall [63][61] can be calculated. The following equations 4.1 and 4.2 were used to calculate the accuracy of our proposed model as proved by [104]:

$$Precision = \frac{COR}{COR + INC + PAR + SPU} = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{COR}{COR + INC + PAR + MIS} = \frac{TP}{TP + FN} \tag{4.2}$$

The TP is the actual number of correctly predicted entities for the same class, TN is the number of correctly predicted entities which not belongs to the designated class, FP is the number of entities that are correctly predicted that belongs to other classes, and finally FN is

the number of wrongly predicted entities even for the other classes. therefore, the f1-score accuracy could be measured using the following equation 4.3 as proved by [104]:

$$F1-score\ Accuracy = 2 \times \frac{((Precision \times Recall))}{Precision + Recall} \tag{4.3}$$

The previously mentioned equations were applied on the following confusion metrics to calculate precision, recall, and accuracy.

		Predicted				
	Classes	**Places or location**	**Person**	**Organization**	**Street**	**Accuracy**
Actual	**Places or location**	462	312	123	288	0.39
	Person	235	176	513	543	0.12
	Organization	309	97	375	234	0.37
	Street	256	367	305	1046	0.53

Table 4.6: PNER confusion matrix and accuracy of the annotated dataset

The average accuracy of the PNER for the four classes is 0.34. The average precision is 0.33, and the average recall is 0.35. The following table 4.7 shows the confusion matrix and the accuracy of prediction for each class.

		Predicted				
	Classes	**Places or location**	**Person**	**Organization**	**Street**	**Accuracy**
Actual	**Places or location**	663	140	207	175	0.56
	Person	206	924	151	186	0.63
	Organization	170	123	588	134	0.58
	Street	215	302	234	1223	0.62

Table 4.7: Stanford confusion matrix and accuracy of the annotated dataset

The average accuracy of the Stanford NER for the four classes is 0.59. The average precision is 0.59, and the average recall is 0.60. Table 4.7 shows the confusion matrix and the accuracy of prediction for each class.

		Predicted				
	Classes	**Places or location**	**Person**	**Organization**	**Street**	**Accuracy**
Actual	**Places or location**	865	131	99	90	0.73
	Person	110	1202	76	79	0.82
	Organization	81	120	730	84	0.72
	Street	321	251	396	1006	0.51

Table 4.8: TwitIE confusion matrix and accuracy of the annotated dataset

The average accuracy of the Stanford NER for the four classes is 0.59. The average precision is 0.59, and the average recall is 0.60. Table 4.8 shows the confusion matrix and the accuracy of prediction for each class.

After investigating the results, we found that most of the entities that we have detected were informal. The rest of the detected entities fall within the generated patterns, so the entities had been recognized by PNER. Thus, we found that our proposed approach could work as an additional layer to enhance the performance of any state-of-the-art or standard NER by maximizing the ability of retrieving informal entities within text. As previously shown in the core of PNER section, the linking between the informal and formal entities has been made through the informal bag of words then forwarding the resulting formal related entity to the standard NER to validate this new result. Thus, we tested the two enhanced approaches on the annotated dataset. The following two tables 4.9 and 4.10 show the confusion matrix for both the Stanford and TwitIE combined individually with the PNER.

		Predicted				
	Classes	**Places or location**	**Person**	**Organization**	**Street**	**Accuracy**
Actual	**Places or location**	865	102	127	91	0.73
	Person	202	953	140	172	0.65
	Organization	152	120	619	124	0.61
	Street	173	246	164	1381	0.7

Table 4.9: Fused Stanford with PNER confusion matrix and accuracy of the annotated dataset

		Predicted				
	Classes	**Places or location**	**Person**	**Organization**	**Street**	**Accuracy**
Actual	**Places or location**	959	95	75	56	0.81
	Person	110	1202	76	79	0.82
	Organization	70	95	791	59	0.78
	Street	217	180	334	1243	0.63

Table 4.10: Fused TwitIE with PNER confusion matrix and accuracy of the annotated dataset

The following table 4.11 shows the full results after applying the three equations of precision, recall, and f1-score accuracy on the confusion matrices.

Entity Classes	**PNER**	**Stanford**	**TwitIE**
Place or Location	0.39	0.56	0.73
Person	0.12	0.63	0.82
Organisation	0.37	0.58	0.72
Street	0.53	0.62	0.51
Total Accuracy	0.3652	0.6028	0.6746

Table 4.11: Impact of PNER on state-of-the-art NER accuracy

Entity Classes	**PNER**	**Stanford**	**TwitIE**	**Stanford +PNER**	**TwitIE +PNER**
Place or Location	0.39	0.56	0.73	0.73	0.81
Person	0.12	0.63	0.82	0.65	0.82
Organisation	0.37	0.58	0.72	0.61	0.78
Street	0.53	0.62	0.51	0.7	0.63
Total Accuracy	0.3652	0.6028	0.6746	0.6771	0.7442

Table 4.12: Impact of PNER on standard NER accuracy

Table 4.12 shows the impact of PNER on the standard NER algorithms. The accuracy of both approaches has been increased by 7 % approximately. This shows the impact of the co-occurrence patterns on both approaches. Figure 4.7 shows a clearer comparison between all of the NERs including the enhanced ones.

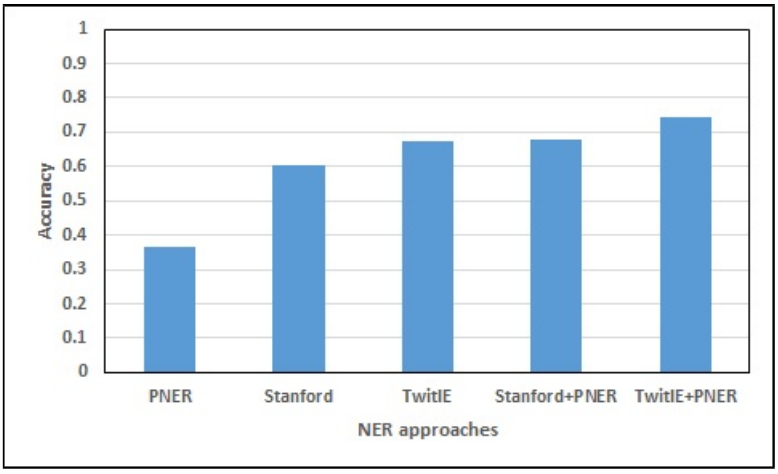


Figure 4.7: Accuracy of the three NER approaches and the tow enhanced approaches

Finally, it is important to clarify that PNER works better on informal entities than on the formal ones unless the co-occurrence patterns of these entities can be detected. The low frequency entity patterns can also affect the PNER performance. Nevertheless, PNER could be applied to different NER methods as it works from a black box perspective with the state-of-the-art and standard NER algorithms. Therefore, it will be easy to work with any system and it is re-usable.

4.4 Summary

The co-occurrence patterns for informal entities were mainly targeted by our proposed approach. The motivation was to increase the capability of finding entities in the informal case. We had been influenced by social media text which does not conform to any regulations and is classified as a messy noisy environment. There are still many limitations to our approach as it depends more on the state-of-the-art algorithms. Thus, any bad performance from the NER algorithm will be reflected in the total PNER performance, and therefore, it could be classified as a dependent approach. On the other hand, the best feature that could make it good for production is re-usability.

PNER uses word patterns embeddings created from Twitter text. The co-occurrences among words was based on frequency and co-occurrence patterns. Therefore, patterns for specific entities like locations or a person's name had been created by people. The tweets are the natural language of people as they write normal text in a similar way to their everyday talk. The normal text gave more confidence for the annotation process and on the results from having it from a natural resource. Additionally, the noisiness in tweets make it more challenging task in natural language processing.

The total accuracy of both Stanford NER and the TweitIE increased by approximately 7% after the state-of-the-art algorithms combined with PNER. This accuracy increased due to the new non-standard format entities that had been identified using our approach using the patterns embeddings. These patterns provides the capability of finding new entities similar to the candidate patterns of entities already used for training in the training stage. Moreover, PNER overcomes the problem of capitalization in some entities by finding similar words

depending on the context. Finally, PNER will not need the parsing tree of the part of speech tagging because it is converting sentences to patterns directly.

Combining both our approach and the state-of-the-art ones shows better performance with the resulting combined model. The better performance implies that there were some limitations in the state-of-the-art approaches and we found a simple way to address these limitations and enhance the performance in a more generalized perspective. The challenge will be more difficult in future as the language is changing, but our approach could be easily updated and tested on a regular basis to see the impact of language changes on performance.

Chapter 5

PR_Sentiment Model

5.1 Introduction

Sentiment analysis for short text is another challenge in NLP. The main concept is to find any information that could lead to specify if this text is either positive or negative (happy or sad) and sometimes the degree of happiness that could be predicted at some point. Traditional approaches produced several solutions, like using auxiliary information such as emoticons or a specific kind of information depending on the supporting resource as Cheng et al. explained in their approach [105]. Also, Wagh and Punde made a survey [106] on sentiment analysis especially for Twitter data as they described how keywords and emoticons are used on Twitter sentiment analysis. Our proposed approach uses a supervised machine learning technique which is trained on a standard dataset classified by sentiment. Several parts of the proposed supervised approach will be derived from the PRSTM 3.4 approachas described in word co-occurrence patterns, word embeddings, and word matching are some of these parts. Therefore, we applied the same concept with the proposed model but for the Twitter sentiment analysis (TSA). We have named our proposed approach in this chapter as PR_Sentiment which stands for probabilistic relational sentiment.

On the other hand, several studies using neural networks for sentiment analysis. SemVal [38] is one of the most popular semantic analysis development systems. Many publications in several SemVal versions to solve tasks related to sentiment analysis and other NLP subjects. Novak et al. [37] published a paper in 2015 which is main goal is to develop a SemVal successful task in sentiment analysis for social media and short text. We described the whole published study in the literature review chapter and we found that the overall approach was good with some areas that they never covered like real-time Twitter sentiment detection, using other metrics not just f-score, and other classifiers to show what are the differences between the classification performance. Also, the reason of using SVM only was not shown in the proposed research. Task 5 of Semval2013 [38] was addressed in Chen et al. [34] proposed approach that uses neural networks to produce a score for the input tweets or short text between 1 and -1. There are several disadvantages or drawbacks on this model starting from the evaluation technique that they suggested to improve in the future to the other one is the static concept that they rely on with no suggestion for a real-time process. Additionally, they did not discuss the word variation impact on sentiment analysis in general

as we discussed it in the word embeddings section 3.4.2.

Sentiment Analysis is a text polarity classification concept. This polarity could be either positive, negative, or neutral or it could be multiple levels of positive and negative. The PR_Sentiment Model works on each topic individually and depends upon trained data, which is why we trained the model with samples from annotated positive and negative data. The evaluation of our model will be based on a standard sentiment Twitter dataset.

The general concept of polarity (from a traditional point of view) depends mainly on specific words that give either a positive or a negative meaning [106]. Thus, sentiment analysis based on trained tweets will solve the lack of these words within the text context. Further, the topic plays a role in deciding the sentiment orientation.

5.2 Dataset

The Sentiment 140 dataset used in our proposed model is a standard dataset that was collected by Alec Go et al. [2] at Stanford University. The tweets were collected in 2009 in two separate sets: training, and testing. The training dataset consists of 1.6 million tweets which were labelled depending on emoticons, and divided into 800,000 positive and a similar number of negative tweets. The emoticons were removed after labelling so it will not affect the classification process as it is considered as noise within text. The testing dataset consists of 359 tweets which were marked manually. Table 5.1 shows the number of labelled tweets in both the training and testing datasets.

	Training Dataset	Testing Dataset
Positive Labels	800,000	182
Negative labels	800,000	177

Table 5.1: Sentiment 140 dataset number of labels for training and testing data.[2]

The dataset consists of six columns: tweet polarity, tweet id, date of the tweet, query used to create the tweet (NO_ QUERY for the training dataset), the user id, and the text of the tweet. The testing data was created using several queries depending either on a specific product or a company. The table 5.2 shows how these testing tweets were divided into categories. The queries represent the product types, companies, or locations as described in the previous table. Moreover, the tweet polarity, which is the most important column in the dataset consists of one of two values, either “0” or “4” for negative and positive polarity respectively.

Category	**Total**	**Percent**
Company	119	33.15 %
Event	8	2.23 %
Location	18	5.01 %
Misc.	67	18.66 %
Movie	19	5.29 %
Person	65	18.11 %
Product	63	17.55 %
Grand Total	**359**	

Table 5.2: Sentiment 140 test data categories [2]

The dataset is retweet free as all of the retweeted tweets had been removed based on the existence of “RT” in any tweet. Also, some post processing had been applied, such as removing some repeated letters in some words, as in the example the authors mentioned in their work (e.g. huuuungry, huuuuuuungry, huuuuuuuuuungry) [2]. They removed the repeated “u” and replaced it with just one to produce a word “huungry” with two “u” letters.

We used the Sentiment 140 dataset to train our model then tested it on the testing data for evaluation. Our proposed model will be compared to the accuracy that has been shown

by other studies which have used the Sentiment 140 dataset. The next section will show the proposed model framework and the described dataset.

5.3 PR_Sentiment Framework

Unlike in several studies [107] [108] [109], the proposed Probabilistic Relational Sentiment model (PR_Sentiment) does not include auxiliary information to bias the tweets polarity. The emoticons, some words, or even the word "not" as a positive or negative word could be considered as auxiliary information. The occurrence of this information in tweets is not a major factor so it will not be considered as part of the main concept. The use of similar techniques which targets special domains implies the existence of such information. Therefore, we produced our approach towards generalising this concept for short text by finding word co-occurrence patterns on cases with similar polarity. The proposed model was then trained on the datatset of tweets that was described in the dataset section 5.2 where the emoticons helped to classify tweets and were then deleted from the text.

The annotated tweets were classified as positive or negative, or "0 or 4" as labels, then passed to the proposed model. The first part of the proposed model is the offline process which produces word embeddings from the 1.6 million annotated training tweets. In addition, we tried our word embeddings that had been generated from the streamed tweets as described in chapters 3 and 4. The annotated tweets were passed to the PRSTM to extract the patterns for both polarities and deliver them to the PR_Sentiment model. The other feed is the Real-Stream of tweets to be classified, which is for evaluation purposes, as the annotated testing data is already provided for from the testing stage. Eventually, the results shows the tweet polarity as either positive or negative. The framework is shown in figure 5.1.

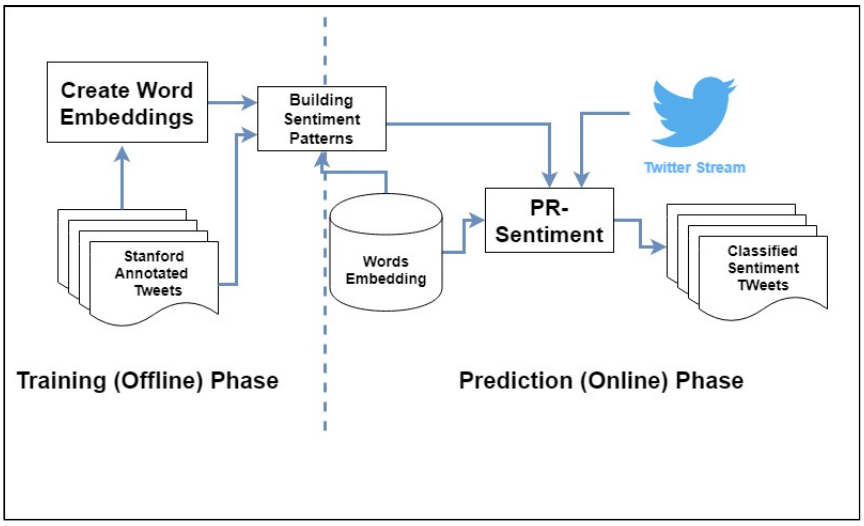


Figure 5.1: PR_Sentiment Framework

The resulting classified tweets show polarity as either positive or negative based on how likely the negative or positive patterns occur in the tweet. The main concept of our proposed model works on a huge dataset which produces more word co-occurrence patterns for both classes. The following section describes in detail the functionality of each stage of

the proposed model.

5.3.1 Training (offline) Phase

The training phase is the most costly part of the proposed model as it consists of stages related to the pre-processing of text as well as building or creating the word embeddings. The proposed probabilistic relational sentiment approach works on an infrastructure which is the word embeddings. All of the words co-occurrences, including the frequencies and co-occurrence frequencies, were mapped to the embeddings which are simply structured, but the creation process is time consuming. On the other hand, it makes the feature extraction process of converting words to the feature space much easier and faster on the prediction level. Words are simply easy to search for and represented. The main embeddings structure was described in earlier chapters, but we now apply this to the sentiment classification concept. The following sections will describe the stages of the offline phase which consists of two main stages, which are word embeddings, and the sentiment patterns.

5.3.1.1 Creating word Embeddings and Sentiment Patterns

As described in the previous section, our proposed model depends on words co-occurrences within text; these co-occurrences are not artificially created but exist naturally. The cultural and environmental impact on language is a reflection of any person who wrote this piece of text (tweet). The most amazing thing in short text is how you can fit an abstraction of an idea that you think of in a very short text and expect that there will be an audience to understand it, unless this text is targeting a special person or people who are already aware of what has been written. On the other hand, there exists a general type of tweet that can be understood by the public audience or one to a particular company relating to feedback or a complaint. The majority of the sentiment work is about customer reviews, which most companies need as feedback for their services or products. For this reason, generalising the sentiment perspective is not an easy task.

We proposed PR_Sentiment as a generic approach to short text sentiment analysis. Our proposed approach is to deal with words co-occurrences, especially in social media short text which is challenging because of sparsity and noisiness. These challenges make it difficult to implement; the change in language is another challenge that makes it much harder for implementation on a real time basis. Unless updating these word co-occurrences, Keeping track of changes needs a data stream and a very regular check on statistics similar to the Oxford English Dictionary [4] check but on several levels, such as the word and word co-occurrence patterns levels.

The Sentiment 140 dataset includes 1.6 million tweets that passed through the word embeddings creation stage. This stage is based on the skip-gram method to create n-grams, which happened to be trigrams in our proposed model. The most balanced n-gram form for text data is the trigram compared to a greater than or less than window (i,e. $n \leqslant 3$).N-grams where n is less than 3 aggregate the words on a very basic level making co-occurrence patterns ineffective as they will be similar to words histograms. On the other hand, finding frequent n-grams where n is higher than 3 in short text is rare. Therefore, the higher “n” is similar to

uni-grams but with more complexity. Table 5.3 shows the top 20 highest frequencies of the trigrams within 50,000 randomly selected negative and positive tweets. The trigrams in red are the common co-occurence patterns (trigrams) between the two classes. These common patterns cause confusion in the classification process, thus, these patterns will be isolated to a new class at some point.

Positive Co-occurrence Patterns	**Positive Patterns Frequency**	**Negative Co-occurrence Patterns**	**Negative Patterns Frequency**
['null', 'good', 'morning']	353	['i', 'have', 'to']	587
['thanks', 'for', 'the']	339	['to', 'go', 'to']	572
['null', 'going', 'to']	323	['i', 'wish', 'i']	455
['looking', 'forward', 'to']	264	['i', 'want', 'to']	386
['cant', 'wait', 'to']	236	['want', 'to', 'go']	322
['i', 'love', 'you']	219	**['null', 'i', 'am']**	**309**
['null', 'just', 'got']	213	['i', 'have', 'a']	297
['im', 'going', 'to']	**200**	['i', 'need', 'to']	294
['have', 'a', 'great']	199	['null', 'i', 'have']	284
['null', 'i', 'am']	**190**	['clean', 'me', 'null']	284
['a', 'great', 'day']	187	['wish', 'i', 'could']	281
['is', 'going', 'to']	185	['dont', 'want', 'to']	269
['null', 'i', 'love']	183	['wish', 'i', 'was']	248
['thank', 'you', 'null']	183	['to', 'work', 'null']	248
['cant', 'wait', 'for']	175	**['going', 'to', 'be']**	**245**
['i', 'cant', 'wait']	168	['null', 'i', 'dont']	244
['how', 'are', 'you']	166	['null', 'i', 'want']	239
['thank', 'you', 'for']	164	['go', 'back', 'to']	236
['going', 'to', 'be']	162	**['im', 'going', 'to']**	**231**
['have', 'a', 'good']	161	['null', 'i', 'wish']	223

Table 5.3: Positive and negative co-occurrence patterns frequencies for 50,000 tweets from the Sentiment 140 dataset.

The 50,000 co-occurrence patterns in table 5.3 show how there are some identical patterns in both negative and positive classes. The similarity of patterns for the selected tweets were 51,728 patterns out of 493,705 in total. The summation of total co-occurrence patterns frequencies was 102,854 resulting in an average of nearly 2 frequencies to each pattern. The low frequencies to each pattern shows how irrelevant these common patterns are and could be removed from the whole process. On the other hand, the summation of the first 1,000 highest patterns frequencies is more than 50,000 which show that these are the most effective patterns. Also, most of these patterns are not common between the two classes. Figure 5.2 shows how the accumulative summation of the first 1,000 tweets reached to 50,000 in negative patterns and the first 1,500 accumulative summation for the patterns frequencies is more than 50,000, which is equal to the number of the randomly selected tweets. This means

that the existence of any of any of these patterns in any tweets could define its polarity even if we removed the low frequency patterns above the 1,500.

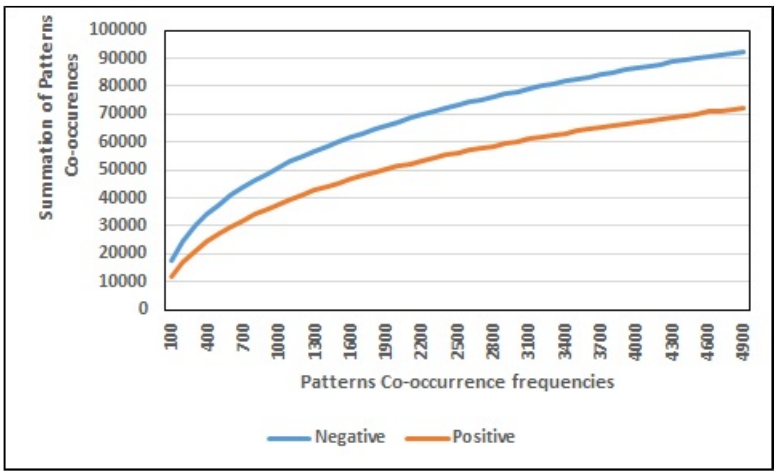


Figure 5.2: Positive and negative patterns accumulative frequencies for a 50,000 randomly selected tweets..

Similarly, we derived statistics on 100,000 randomly selected patterns from both negative and positive classes. The negative and positive patterns are 524,288 in each class and the similar patterns between the two classes equals 55,879 patterns which is not higher than the similar ones in the 50,000 tweets described in the previous section. The total summation of frequencies of the similar patterns was 268,409 resulting in the average of nearly 5 occurrences as a frequency to each of the similar patterns. This ratio is still small compared to the huge number of tweets selected. Table 5.4 shows the similar pattern in both negative and positive classes. These patterns are quite similar to the ones in the 50,000 table showing that there will not be that much change when data has been duplicated. This analysis shows the consistency of the similar patterns in these classes.

Positive Co-occurrence Patterns	Positive Patterns Frequency	Negative Co-occurrence Patterns	Negative Patterns Frequency
['thanks', 'for', 'the']	740	['i', 'have', 'to']	1194
['null', 'good', 'morning']	673	**['to', 'go', 'to']**	**1068**
['null', 'going', 'to']	649	['i', 'wish', 'i']	999
['cant', 'wait', 'to']	503	['i', 'want', 'to']	758
['looking', 'forward', 'to']	483	['wish', 'i', 'could']	624
['null', 'just', 'got']	452	['i', 'have', 'a']	615
['im', 'going', 'to']	**433**	**['null', 'i', 'am']**	**606**
['i', 'love', 'you']	426	['i', 'need', 'to']	604
['have', 'a', 'great']	419	['null', 'i', 'have']	574
['null', 'i', 'am']	**418**	['want', 'to', 'go']	558
['null', 'i', 'love']	416	['dont', 'want', 'to']	526
['to', 'go', 'to']	**353**	['null', 'i', 'dont']	516
['a', 'great', 'day']	345	['wish', 'i', 'was']	511
['thank', 'you', 'null']	336	**['im', 'going', 'to']**	**501**
['i', 'cant', 'wait']	332	['null', 'i', 'want']	484
['is', 'going', 'to']	330	['null', 'i', 'hate']	465
['have', 'a', 'good']	324	['null', 'i', 'wish']	463
['null', 'had', 'a']	323	['going', 'to', 'be']	462
['cant', 'wait', 'for']	321	['i', 'dont', 'know']	453
['how', 'are', 'you']	311	['i', 'dont', 'want']	451

Table 5.4: Positive and negative co-occurrence patterns frequencies for a 100,000 tweets from the sentiment140 dataset.

The patterns frequencies are sorted in descending order to find how many of patterns with high frequencies could cover most of the patterns in tweets. Similar to the previously shown 50,000 randomly selected patterns, figure 5.3 shows how the accumulative frequencies of patterns occurrence summation reached 100,000 total occurrences of 1,100 for the negative patterns and 1,700 patterns for the positive. The curve starts to increase at a lower rate and nearly flattens after these two values, indicating that the frequencies of the patterns are decreasing quickly compared to the first 2,000. The number of the patterns increased from the 50,000 tweets but this increase is relatively small. Therefore, the frequencies of patterns might increase when selecting more tweets, but this increment is still small when compared to this large amount of data. These statistics will help more to optimise the solution to get better performance. The way of dealing with this problem is to remove the low frequency co-occurrence patterns. Also, the processing time will be focused on a designated number of patterns in each class after ensuring that there will not be common patterns between the positive and negative classes.

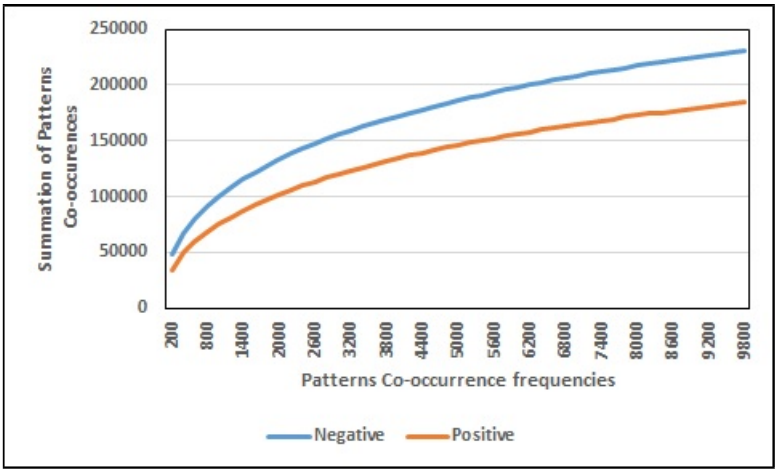


Figure 5.3: Positive and negative patterns accumulative frequencies for a 100,000 randomly selected tweets.

Creating the word embeddings will not just be about the words co-occurrences and frequencies, as explained in the PRSTM section 3.4.2. A table of patterns frequencies worked as filters to check which patterns should be considered in the process of features extraction. The process of building the word embeddings will be similar to that previously mentioned in the word embeddings section. Finally, we will not consider removing the stop words as some of these words, like "not", could convert the polarity of any positive or negative word.

5.3.1.2 Sentiment Patterns Features Representation

Converting words to features is not an easy task, especially for noisy and messy text. Additionally, single word representation causes more confusion in machine learning as many words could be common in several classes. Therefore, the word patterns will be the commonest entity that could represent any class for the reasons we discussed in the previous section.

In this section we will propose two different ways to convert these patterns to features. The first one is using our TF-ICTF equations 3.8 and 3.9 with the Relational Factor equation 3.10 which will take the centre word with the relation to both words on each side of the pattern. This kind of representation needs the type of word embeddings that was described in the word embeddings section 3.4.2 as there will be three arrays with the frequencies; the vocabulary array with frequencies, and two arrays of context words for each word in the vocabulary list with the co-occurrence frequency with this word. Therefore, the TF-ICTF equations could be applicable after creating the word embeddings from the sentiment training dataset.

The second proposed way of converting patterns to features is mainly using the co-occurrence patterns embeddings where the patterns frequencies play the main role as the main factor. All patterns and frequencies shown in tables 5.3 and 5.4 will be converted to features. We will refer to the vector of patterns frequencies as P, where each $p_i \in P$ and where $i = 1,...,k$ as p_i is each pattern in the list of patterns P and k is the total number of patterns. Fq represents the vector of frequencies of each pattern in P and the $fq_i \in Fq$.

Equation 5.1 shows how the probability of each feature is represented.

$$ft_i(fq_i,k) = \frac{(fq_i/k) - Fq_{min}}{Fq_{max} - Fq_{min}} \tag{5.1}$$

where ft_i is the normalized feature which represents the pattern probability within the total k patterns in the whole corpus. Thus, $ft_i \in Ft : i = 1,2,3,...,k$ as Ft is the normalized feature vector for all the patterns in the corpus. These patterns could be either positive, negative, or exist in both negative and positive tweets so we could call them neutral. The results for these features are all > 0 which makes no distinction between negative or positive patterns representation. Therefore, we have worked on each class separately to generate these patterns and then we found the common patterns and made them the neutral class. All the features of the negative extracted patterns were multiplied by -1 to make the feature value less than zero. The neutral patterns were set to zero whenever they were found within any tweet. The resulting feature for any tweet after finding the summation for the patterns will be either less than or greater than zero, or could be zero if neutral. Equation 5.2 shows how the summation of each feature for any pattern in a tweet is calculated.

$$ft_i(fq_i,k) = \begin{cases} -ft_i(fq_i,k), & \text{IF } p_i \in P_{Negative} \\ ft_i(fq_i,k), & \text{IF } p_i \in P_{Positive} \\ 0, & \text{IF } p_i \in P_{Neutral} \text{ or Otherwise} \end{cases} \tag{5.2}$$

ft_i feature will be used to determine pattern's polarity as it had been divided to three separate patterns that can be shown in the equation 5.2 and were described in the last paragraph. The resulting patterns will be negative, positive, and neutral and if the pattern is not found then it will be set to zero, similar to neutral. These features will be ready for the online phase to make it possible to predict polarity using a simple conditional algorithm.

The prediction for any input short text will depend mainly on the trained data. The existing patterns are the embeddings that will be used to calculate the features as explained earlier in this section.

5.3.2 Prediction (online) Phase

Finding a word that gives a positive sentiment indication in any short text does not mean that the sentiment polarity of the whole short text is positive. The context around any word's role, more than the word itself as any word within the context, could suggest a different meaning. Therefore, co-occurrence patterns are more accepted to define the polarity for a short text. Combining multiple patterns from the same class is another way to represent the tweet or even take the highest score; there are so many different ways to be directed to a classifier for prediction of text polarity.

The short text, i.e. the tweets, went through several stages in the prediction phase reaching to the stage of decision of polarity. The stages started from pre-processing and cleaning and progressed to the polarity check which gives one of three outputs: positive, negative , or neutral. This decision will be decided depending on the score of the feature vector as shown in detail in figure 5.4.

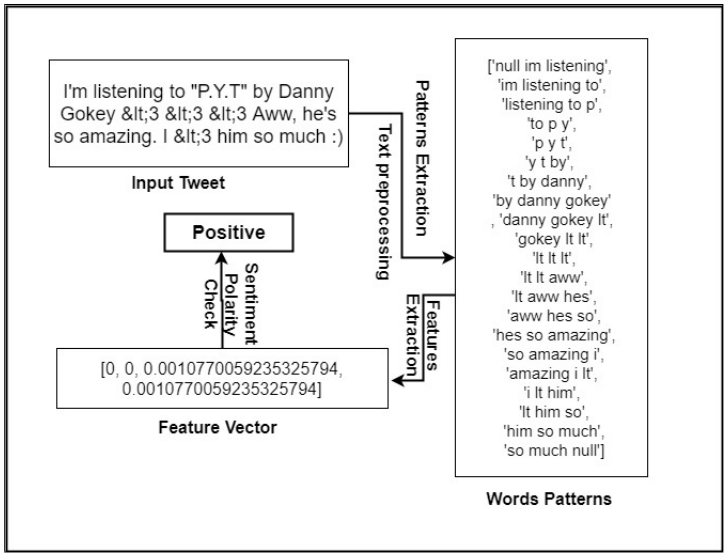


Figure 5.4: Sentiment polarity prediction process from input until decision.

The online phase requires the input of a tweet which will be directed to the pre-processing stage as shown in figure 5.4. The pre-processing includes removing special characters and links from text then converting the whole text to a list of words (tokens) using the tokenization process. We used the *tokenize* process from the *NLTK* package to get the words list as well as the regular expressions from the *re* package in the *Python* language. The resulting list of words will be passed to the patterns creation stage where patterns of trigrams will be produced. These patterns go through the features extraction process which includes the patterns (words) embeddings that had been created in the offline phase already. Any found existing patterns in the embeddings will be converted to features using the equations 5.1 and 5.2 and take the frequencies of patterns from the table of embeddings similar to table 5.4.

The resulting feature vector for the shown tweet in figure 5.4 is not a similar length to the list of the resulting patterns. This shows how much sparsity impacts any process of natural language processing, although, the sentiment process still produces good results. The patterns embedding will produce the feature value of the input patterns from a tweet after mapping it against the training patterns in the embeddings. Then, the features will be collected as positive feature values for positive polarity patterns, negative feature values for negative polarity patterns, and zero for the neutral patterns. Summation of the whole resulting feature vector will produce a value that decides the polarity of the input tweet.

We run the proposed model on a Lenovo Thinkpad machine with Core i7 processor and eight gigabyte memory. The result for a node process will take less than a second as it could be considered as a near real-time process. This is due to the simple calculations and the long preparation (offline) training process. The core of the proposed Probabilistic Relational Sentiment model will be described in more detail in the following section.

5.3.2.1 PR_Sentiment

The main purpose of sentiment analysis is to find the sentiment polarity for an input text, although the ultimate goal is to provide feedback of an answer, either positive or negative,

from a very large amount of text, especially from social media. It is considered as a quantitative measurement when it is applied to social media. Therefore, there is still work to be done in the part of textual analysis as the English language is changing fast, especially in social media.

Our probabilistic approach had been trained, validated, and tested on the standard dataset Sentiment 140 which was collected from Twitter. The input tweet, shown in figure 5.4 of the proposed model was positive after getting through several processes. The one thing that helped to get that result was the occurrence of the positive pattern, for example, *"hes so amazing"* which occur in the training patterns of our embeddings. The other zeros in the feature vectors were related to the neutral patterns. The rest of the text patterns were unknown so there are no values related to most of the patterns. Two points can be identified, which are the sparsity of patterns, and the informality and noisiness of text which leads to the sparsity in the first place. We used the informal bag of words which is ready to fill this gap and expand the ability of the proposed model to provide more accurate results.

The first step in the PR_ Sentiment model is to clean the tweet of the special characters and links. The cleaning process was done using regular expressions to remove the noise. Using the "re" package in Python and some simple steps, a list of tokens was produced. One of the powerful regular expression that was used is *r'[^a-zA-Z0-9]'* which produces text only. Other regular expressions which are much longer than this one was also used to isolate the links (URLs) from the text. The resulting text is just a list of words that will be directed to the step of generating the trigram patterns.

The process of converting the list of words to patterns started from adding two "NULL" words to both ends of the list to define the margins of the tweet. Moving forward, each three words will be grouped in small lists to be compared against the embeddings and retrieve the feature of this pattern, if it exists. The process will return nothing if the pattern does not exist and zero if the pattern is in the neutral zone. The third possible value that could be returned is a number that represents this pattern, which could be either positive if the pattern matches the positive patterns or negative if it matches the negative ones. The summation of the resulting feature vector will define the polarity of this input text. The features are calculated using the equations 5.1 and 5.2 in section 5.3.1.2. The following algorithm shows the prediction process starting from the input tweet reaching the resulted polarity.

algorithm 5 PR _ Sentiment Algorithm to convert tweet to features vector

Require: X as input text tweet, FT as the patterns embedding

{Patterns Embeddings consists of a list of tuples, Each tuples consists of a pattern with its feature where each tuple is structured as [pattern, feature] with indices of 0 and 1 consecutively}

Ensure: *OutVect* Features vector

Regular Expression Clean(X)

{The regular expression will remove the special characters and links}

$Wl \leftarrow$ tokenize(X)

$L \leftarrow$ length(Wl)

Initialize $Wl \leftarrow$ ["$NULL$" + Wl + "$NULL$"]

for $i = 1$ to $L-1$ Step 1 **do**

  $temp = Wl[i-1 : i+1]$

  if $temp$ in FT **then**

    $OutVect \leftarrow\square\ Wl\ [i][1]$

  end if

  $L = Length(OutVect)$

  $Pol_Val = \sum_1^L OutVect$

  if $Pol_Val > 0$ **then**

    $Polarity =$ "$Positive''$

    if $Pol_Val < 0$ **then**

      $Polarity =$ "$Negative''$

    else

      $Polarity =$ "$Neutral''$

    end if

  end if

end for

Return ($OutVect$, $Polarity$)

The PR_ Sentiment algorithm 5 converts the input tweet to a feature vector with a final result of the polarity after finding the summation of the feature vector. The condition that determines the polarity works based on the value of the summation. There are several situations that could be a mix of co-occurrences from both negative and positive patterns in the same tweet. Neutral patterns could join both polarities but it will not affect the polarity unless the whole tweet is a neutral pattern. In general, the bigger value will decide the polarity for both the negative or positive. The main idea behind this concept is that the pattern which most people agreed to be negative or positive text will be the most dominant and have a greater value. Normalization that had been applied to both features in the embeddings ensures that the values of both polarities get equal chances (probabilities). Therefore, the normalisation is an essential step (which is described in the offline stage in equation 5.1).

5.3.2.2 PR_Sentiment with Informal Bag of Words

Our proposed approach produced large number of patterns which reached more than one million. These patterns shows diversity of both polarities (positive and negative) of several patterns in different subjects in general. Also, there were more than 60,000 neutral patterns that had been derived from finding the common tweets patterns from both polarities. These neutral patterns increased the accuracy in the testing stage but there is still something more to add. We have analysed the dataset to find how we could improve the performance and we came back with the same conclusion that the non-standard have the main impact on performance. Most of the words in the testing stage might appear to be informal. For example, the word “love” appeared many times in different forms like “looove” or just “lov”. The informal bag of words could make an adjustment to the algorithm to derive the formal form of any undiscovered or not found pattern then search for it in the embeddings to find its value if it exists. This small addition had been made using the informal bag of words that had already been created in our first proposed model PRSTM 3.4.

algorithm 6 PR _ Sentiment Algorithm with Bag of Words to convert tweet to features vector

Require: X as input text tweet, FT as the patterns embedding, BoW as the informal bag of words

{Patterns Embeddings consists of a list of tuples, Each tuples consists of a pattern with its feature where each tuple is structured as [pattern, feature] with indexes of 0 and 1 consequently}

Ensure: $OutVect$ Features vector

Regular Expression Clean(X)

{The regular expression will remove the special characters and links}

$Wl \leftarrow$ tokenize(X)

$L \leftarrow$ length(Wl)

Initialize $Wl \leftarrow$ ["$NULL$" + Wl + "$NULL$"]

for $i = 1$ to $L-1$ Step 1 **do**

  $temp = Wl[i-1 : i+1]$

  if $temp$ in FT **then**

    $OutVect \leftarrow\square\ Wl\ [i][1]$

  else

    if Any($temp$) in Bow **then**

      Replace the informal word in $temp$

      if $temp$ in FT **then**

        $OutVect \leftarrow\square\ Wl\ [i][1]$

      end if

    end if

  end if

  $L = Length(OutVect)$

  $Pol_Val = \sum_1^L OutVect$

  if $Pol_Val > 0$ **then**

    $Polarity =$ "$Positive''$

    if $Pol_Val < 0$ **then**

      $Polarity =$ "$Negative''$

    else

      $Polarity =$ "$Neutral''$

    end if

  end if

end for

Return ($OutVect$, $Polarity$)

The added section after the IF statement within the loop checks "IF Any($temp$) in Bow", which means any word in the pattern which is "$temp$" that might be found in the bag of words "Bow" will be replaced by its formal form from the "Bow". Then the resulting modified pattern will be tested again to find its value in the embedding. If not found, no feature value will be added to the feature vector. The evaluation and experimental results of our approach

will be described in detail in the following section

5.4 Evaluation and Experimental Results

Our proposed approach generated performance will be compared to state of art algorithms and recent work. The challenges in the sentiment probabilistic approach was how to find the common patterns within each class. Therefore, the analysis for each class was isolated and then combined in the evaluation and testing. Additionally, common patterns between both classes (positive and negative) have to be isolated from both classes to avoid confusion. The testing set was already annotated, as described in the dataset section. The annotation for the testing was to three classes, positive, negative and neutral as the third one. The common patterns that had been isolated should satisfy this method of classification. The testing and evaluation process will be divided into multiple section to challenge our model using several techniques.

5.4.1 Evaluation with different training dataset size

As described in the previous paragraph, the process of evaluating and testing our algorithm went through several stages. The reason behind that is to show if our proposed model could be trained on a smaller dataset. We took 100,000 tweets randomly selected from positive ones. A similar number of tweets was taken from negative ones, and then a similar number from both negative and positive. The testing size was gradually increased to show how it affects the prediction accuracy. Testing data was selected from the training Sentiment 140 dataset randomly,i.e, from 1.6 million tweets. Table 5.5 shows how the accuracy was nearly 79.7 % when the testing tweets were 2,000 and the training dataset was 100,000. The testing tweets were selected randomly from the 800,000 tweets which were all positively annotated.

Testing Tweets	Accuracy	Precision	Recall	F1-Score
2000	0.797	1	0.797	0.887
10000	0.743	1	0.743	0.853
20000	0.737	1	0.737	0.849

Table 5.5: Positive classification tweets accuracy for 100,000 training tweets

The results show how PR_ Sentimentthe trained on 100,000 tweets could show high performance as the selected testing tweets might not be from the training tweets. The accuracy dropped to 73.7% when the testing tweets reached 20% of the training dataset. The results also show 100% of precision as the selected tweets were all from the same class. Also, the recall and accuracy have the same value of accuracy which is reasonable as well. The f1-score shows more consistency while the number of testing tweets is increasing. Similarly, accuracy started to flatten after the tweets testing amount reached 10,000 tweets. These results in the table 5.5 shows the classification performance of PR_ Sentiment for the positive annotated tweets. Figure 5.5 shows the f1-score consistency much clearer, as well as the accuracy, recall and precision as described earlier in this paragraph.

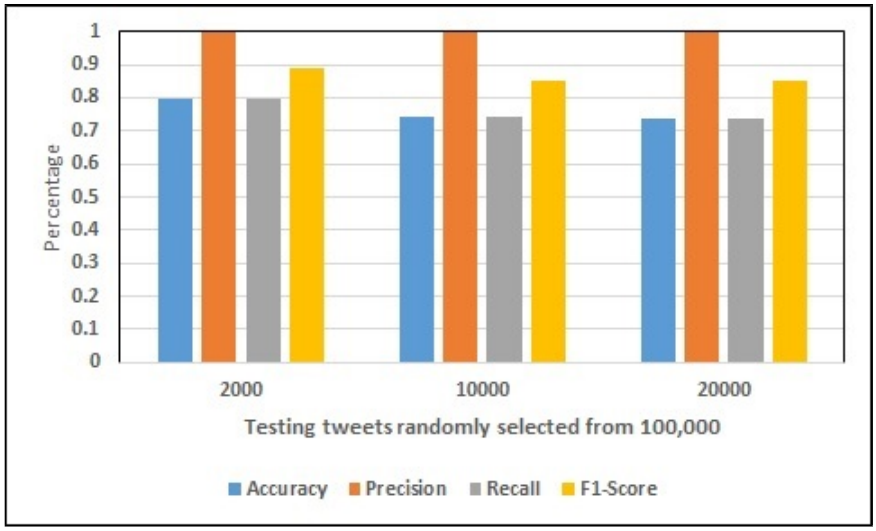


Figure 5.5: PR_ Sentiment accuracy, precision, recall, and f1-score for different sizes of positive testing data.

Similarly, the negative part of data had been tested individually to see how well our proposed model worked with the negative class. The testing of each class individually will show how our proposed model could perform without the impact of the other model as we found there are some similar patterns between the two classes. Then we mixed the testing samples of both classes to show the performance in general. Table 5.6 shows less accuracy if compared to the positive individually tested tweets which leads us to conclude that the positive patterns are more dominant than the negative patterns. Although, the accuracy is still fairly good, the model still needs more improvements. This is shown in Table 5.6 from 60.9% and 75.7% of the f1-score with a testing of 2,000 tweets reaching to 53.3 % accuracy and f1-score of 69.5% with 20,000 training tweets. The accuracy decreased by nearly 7% with the increase in the training tweets. Compared to the positive patterns accuracy in figure 5.5, the f1-score continued consistently as the precision equals one because of the one class of negative. Recall has similar values in the accuracy, of both the False Positive "*FP*" and False Negative "*FN*" as we have two annotated classes only, and the neutral is predicted from the similar patterns of both.

Testing Tweets	**Accuracy**	**Precision**	**Recall**	**F1-Score**
2000	0.609	1	0.609	0.757
10000	0.55	1	0.55	0.709
20000	0.533	1	0.533	0.695

Table 5.6: Negative tweets classification accuracy for 100,000 training tweets

Figure 5.6 summarizes table 5.6 in a bar chart which shows the consistency of the f1-score and the accuracy after the first 10,000 tweets. The final results for both classes will show a more balanced performance among the changes of the dataset.

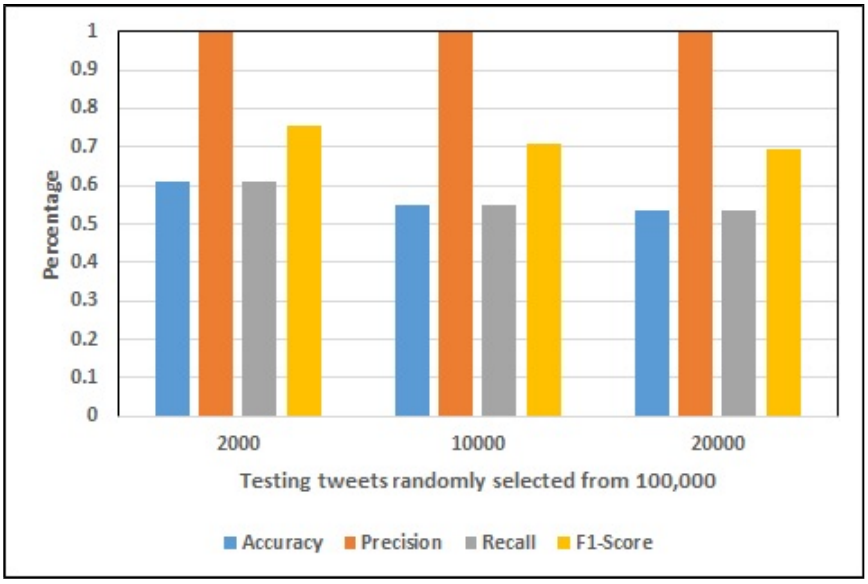


Figure 5.6: PR_ Sentiment accuracy, precision, recall, and f1-score for different sizes of negative testing data.

Further testing was done by choosing randomly from both positive and negative annotated tweets. This test went through the same process of the previous individual tests. The results show more consistency of accuracy and recall with fairly good results in precision as their values started to be nearly flattened earlier than the accuracy and recall. The True Positive results "*TP*" have more impact on precision, making it consistent even when the volume of testing data increased. Table 5.7 shows the precision started with 87.1% then its value declined to 82.9% and 80.9% gradually. After that, the decrement was very slight, achieving 78.3 percent with 20,000 testing tweets. While the accuracy decreased by nearly 12% from 75.3% as well as the recall.Also, the f1-score was impacted by the changes by 2% less compared to accuracy declining to 70% from 80%. The results show good performance compared to both the individual test results.

Testing Tweets	**Accuracy**	**Precision**	**Recall**	**F1-Score**
2000	0.753	0.871	0.753	0.808
10000	0.646	0.792	0.646	0.712
20000	0.635	0.783	0.635	0.701

Table 5.7: Positive and negative tweets classification metrics for 100,000 training tweets

Finally, figure 5.8 shows a clearer view about what was discussed in the previous table 5.7. As shown in this figure, the four values of accuracy started to flatten early with 6,000 tweets with a consistent decrement which was very small compared to the individual experiments that were described earlier.

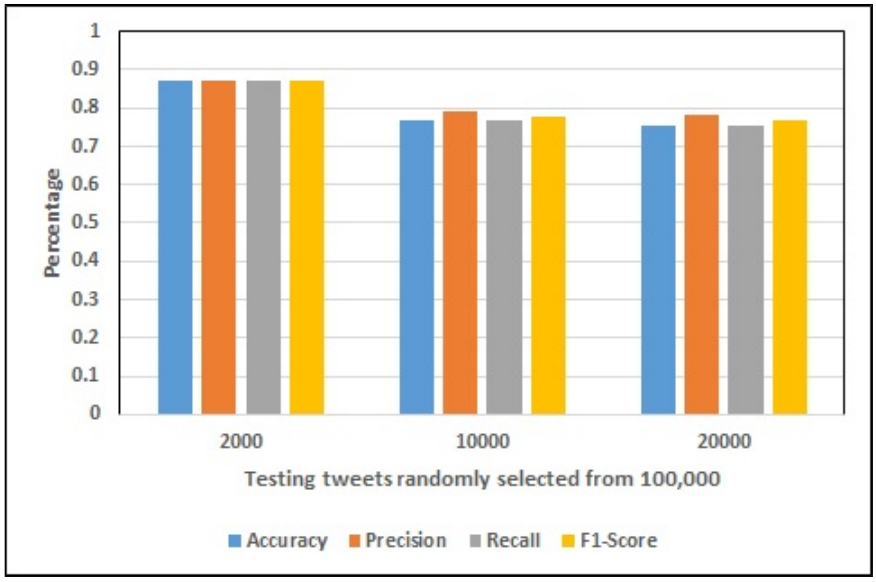


Figure 5.7: PR_ Sentiment accuracy, precision, recall, and f1-score for different sizes of positive and negative testing data.

5.4.2 Evaluation of PR _ Sentiment with Bag of Words

The use of the informal bag of words added another advantage to our proposed work to fill the gap of word's sparsity. The main reason for this step is in bridging the gap between the similar patterns that are unclassified with the ones that are already classified. This step was discussed in detail in section 5.3.2.2. The results in table 5.8 shows how the informal bag of words affects the results by increasing the accuracy which leads to harmonic consistent values (f1-score). Moreover, precision shows 87 % on average with a high consistency over the testing size range.

Testing Tweets	Accuracy	Precision	Recall	F1-Score
2000	0.865	0.874	0.865	0.869477
10000	0.859	0.874	0.859	0.866435
20000	0.852	0.868	0.852	0.859926

Table 5.8: Positive and negative tweets classification accuracy for 200,000 training tweets with bag of words.

The good results demonstrate high flexibility in working on a small dataset of 100,000 or 200,000 tweets, which is considered to be a small part of the whole 1.6 million tweets. This training process was discussed in the training stage of PR _ Sentiment where the number of patterns extracted from part of the dataset could perform well compared to the whole dataset, as shown in the previous figures 5.2 and 5.3 . Thus, we performed this test to show how well our model could perform even on a small dataset and with different, randomly selected sizes of testing data. Figure 5.8 shows that the complete results are consistent over the testing data range.

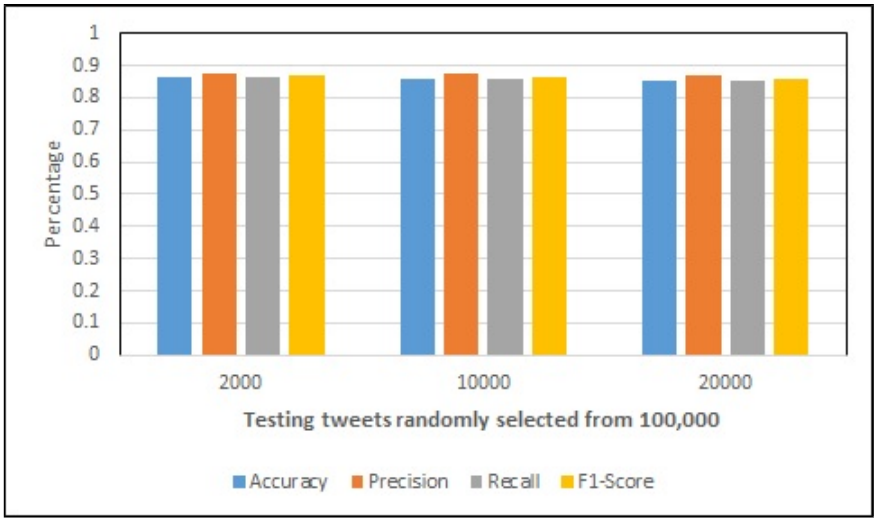


Figure 5.8: PR_ Sentiment with bag of words accuracy, precision, recall, and f1-score for different sizes of positive and negative testing data.

The final challenge for our model was the training over the whole 1.6 million tweets to be tested on the testing dataset that had been annotated by hand, as described in the dataset section. The following section provides an evaluation of our model with some of the state-of-the-art results over the same dataset.

5.4.3 Testing results with full dataset training

The PR_ Sentiment model is simply designed to perform regardless of the size of the training data. Its main objective is to identify the differences of word patterns between the sentimentally annotated classes, and then produce a third neutral class based on the common words patterns. The features will depend on the frequency of these patterns within the dataset, which make some patterns produce large values compared to other ones. The whole training process took 4 days and 3 hours before implementing in the online phase. Moreover, the proposed model is flexible to be updated, as it is just a simple step of updating the word embeddings (patterns embedding) preceded by a pre-processing step to clean the input tweets. Additionally, the informal bag of words increased the accuracy by finding similarly classified patterns to the unclassified patterns, depending on the context and the word similarity.

As a result, the PR_ Sentiment model uses the whole 1.6 million tweets to create the patterns embeddings and then extract the features using the equations 5.1 and 5.2. Compared to the proposed machine learning model by Alec et al. [2] that used Distant Supervision , our model is much simpler. Three algorithms were applied to the same dataset, Sentiment 140; Naive Bayes [110], Maximum Entropy [111], and Support Vector Machine [91] were applied with two different features and two combined features. The features are Unigram, and Bigram and the combined features were Unigram with Bigram, and Unigram with Part of Speech tagging (POS). The accuracy of the four features with the three algorithms shows accuracy of around 80%. Our proposed model shows accuracy higher than 85% when combined with the informal bag of words. Moreover, Unigram uses keywords in the classification but the classification accuracy equals only 65.2% which is considered low compared to other combinations. Table 5.9 shows the accuracy of Distant Supervision

compared to our proposed model. Most of the table was taken directly from their published paper as it had already been trained and tested on the same dataset that we used.

Features	**Naive Bayes**	**Max. Entropy**	**SVM**	**PR_ Sentiment**
Unigram	81.3	80.5	82.2	N/A
Bigram	81.6	79.1	78.8	N/A
Unigram + Bigram	82.7	83.0	81.6	N/A
Unigram + POS	79.9	79.9	81.9	N/A
Pattern Embeddings	N/A	N/A	N/A	87.5

Table 5.9: PR_ Sentiment accuracy compared to Distant Supervision accuracy [2]

The major difference between our proposed work and Distant Supervision on the testing level is that they show only testing on the testing dataset which consists of only 359 tweets. In contrast, we have discussed how our model could work on testing data taken from the training and how the PR_ Sentiment model shows good performance on different sizes of testing data with a small training dataset compared to the whole Sentiment 140 dataset.

Finally, the proposed model with its online version could make predictions depending on any input within less than a second as we have tested it using Lenovo Thinkpad machine with Core i7 processor and eight gigabyte memory. It could also be applied to a set of tweets to classify them.

5.5 Summary

Sentiment analysis could be applicable to several real life problems, like customer and movie reviews, and ratings depending on the reviews as well. The reviews could be collected from social media, for example by following hashtags, or for specific groups for movies over the internet. Either way, there should be generalized tools that could handle these data if they were huge or small depending on the stakeholder needs. Thus, we have provided a generalized model that could be applied to many problems with reasonable efficiency and time. Our supervised model shows high accuracy consistency over the range of testing data size. Therefore, it is flexible in working on both small and large dataset sizes.

The PR_ Sentiment is flexible to be updated, as it is just a simple step of updating the word embeddings (patterns embedding) preceded by a pre-processing step to clean the input tweets. Additionally, the informal bag of words increased the accuracy by finding similarly classified patterns to the unclassified patterns, depending on the context and the word similarity.

We had been inspired by the informality of text and how words are related within context. We chose the trigram to represent our patterns embeddings as each item is a trigram. Patterns frequencies over the whole dataset are the keys to the next stage, which is the features extraction. Sentimentally negative polarity training patterns transformed to negative values features after being normalized similar to the positive polarity patterns but the positive ones will be converted to positive values features. Therefore, the classification task will be very easy to implement, depending on the summation of patterns values. This step was

implemented after isolating the common patterns to produce a neutral class of patterns that will be set to zero wherever they exist within a tweet.

The whole process, which is very easy and simple to implement, could be applied either online to a single input tweet, or to a huge amount of short messy text. Both proposed implementations produce good results as shown in the results and evaluation section. As the code is written in Python, it could be applied to several platforms as Python is widely used and its applications are widely spread and can easily fit into any platform.

Compared to the state-of-the-art which are either using emoticons, or keywords. The state-of-the-art model that had been proposed by Alec et al. [2] used supervised machine learning classifiers and features extraction techniques. They trained their model as we have on 1.6 million annotated tweets. The testing dataset was about 350 tweets. The results shows good performance but with long process compared to our proposed model. Moreover, our proposed model could be updated with new annotated tweets when to avoid sparsity. Also, we added the informal bag of word as another layer that could reduce more from sparsity effect. PR_ Sentiment shows flexibility on the size of training dataset with a good performance as we could not see similar thing on the state-of-the-art algorithms.

Finally, we are planning to develop our proposed model to produce several levels of polarity, like neutral, positive, strongly positive, negative, and strongly negative. This suggested development needs more study of the language and how people are using words to express strongly positive or strongly negative feelings in text.

Chapter 6

Conclusion and Future Work

The language is changing very fast and these changes affecting the natural language processing techniques that have to adapt with these changes. Although, finding an generalised solution for this changing problem is difficult unless the training process includes updating step for the new words. On the other hand, proposing a generalized approach to keep track with these changes and keep updating with new words and the words-context co-occurrences.

The challenges like sparsity should be addressed using new ideas and by continuous monitoring on the social media. Therefore, providing a generalised approach to handle this task could be the ultimate contribution. We produced our proposed models towards addressing this goal. Also, linking the three proposed model will produce a complete approach for NLP that could produce several extracted information from any input text. The extracted information will be sentiment polarity, the topic that tweet most likely belongs to, and the included entities within the tweet as shown in figure 6.1.

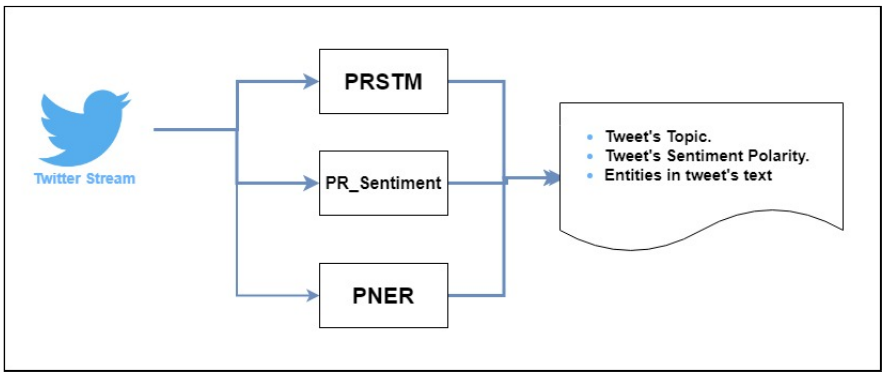


Figure 6.1: Combined approach to extract several information from an input tweet

The following sections will describe more about the research contributions and the proposed future work.

6.1 Contributions of this research

In this section, we will produce the contribution of our research individually as we produced several natural language processing methods. The main aim is to work towars a generalised approach that could be implemented without any auxiliary information and learn from the

environemt that it is working on. Twitter is our challenging environment which provides the noisy and messy short text with changing over time. The contributions are as follow:

1. **Term Frequency Inverse Context Term Frequency:** a modified version of the traditional Term Frequency Inverse Context Frequency (TF-IDF) algorithm which converts word within document to feature vector by using the relation between long document and term frequency. As we considered short text in our approach, we developed this traditional algorithm to work on short noisy messy text (Twitter). The developed TF-ICTF algorithm derivation had been stated in section 3.4.3. The results of comparison between both algorithms shown in section 3.4.6.1.

2. **Probabelistic Relational Supervised Topic modelling approach:** The proposed novel approach had been designed as a supervised topic modelling pipeline to be applied on short messy noisy text. Using the co-occurrence patterns that had been described in section3.3. Also, PRSTM uses word embeddings for short text that had been created from the streamed tweets dataset (the embeddings creation process is in section 3.4.2. The previously described TF-ICTF algorithm had been applied on the word embeddings and using the extracted co-occurrence patterns to convert input tweets to feature space which is described in details in sections 3.4, 3.3, and 3.4.2. PRSTM had been compared against state-of-the-art algorithms and show good results that were shown in section 3.4.6.1.

3. **Probabilistic Named Entity Recognition model:** we have developed a novel approach that targets the entities that are in non- standard format within short text. First , we annotated tweets by finding the entities within its text in section 4.2.1 using two of the state-of-the-art named entity algorithms. Then, we used our training process on these tweets and compare the detected entities with our word embeddings to find the probable non-standard entities in section 4.2.2. The training process completed at this point and the online process produces good results when combining our approach with the state-of-the-art in sections 4.2.3 and 4.3. The proposed model is adaptive to work with any named entity algorithms and enhance its results by finding non-standard entities.

4. **Probabilistic Relational Sentiment Model:** a novel approach that could provide near-real time accurate results. The main concept had been described in chapter 5. The main concept is to create pattern embedding from the training dataset based on trigram concept which is described in section 5.2 and section 5.3.1.2. These patterns will be seperated to positive, negative, and neutral (similar patterns) in section 5.3.1. Converting these patterns to features is depending on the patterns frequency in dataset with a a novel proposed equation 5.2 that converts the pattern to normalised positive or negative value depending on its polarity (from which class) described in section 5.3.1. The results of PR_ Sentiment shows good performance when compared to state-of-the-art algorithm in section 5.4

These contributions all have the same concept of using word embedding and co-occurrence

patterns. Also, the main addressed problem was the text informality whic is started growing on social media making a challenge for traditional natural language processing tools to be implemented on. The next section will provide a summary for the whole thesis chapters.

6.2 Summary

The aim of our project is to produce methods that works towards generalisation of short noisy text processing. Therefore, we choose the Twitter to get short text with the same characteristics as it had been defined as short noisy text by many researchers. This Twitter text is also updated by new words every single day or maybe hour as new hash-tags and words related to events.

We studied the relation between words and context and the new words increment over time as well as the non-standard words. These analysis had been described briefly in section 3.3 which shows a consistent relation between some words frequencies and context words frequency over time. This novel study lead us to produce a formula that identify the important words within text. The importance of these words came from using these words in topic modelling as landmarks in each individual topic within context. Now, the story continues by converting these words to feature space which had been achieved by developing the traditional TF-IDF algorithm as it is the most powerful word feature representation algorithm but for long documents. We produced the Term Frequency - Inverse Context Term Frequency (TF-ICTF) which works on context words frequencies together with the word's frequency in any short text. Word embeddings had been created using trigram technique to provide the required frequencies for TF-ICTF from more than 2500 million tweets. The rest of the proposed model that we name it Probabilistic Relational Supervised Topic Modelling (PRSTM) was applying nine classifiers on the features created from the tweets. We annotated 30,000 randomly selected tweets using the Amazon Mechanical Turk to test our ptoposed model which produce an accuracy result of 89%. The aacuuracy had been compared to two state-of-the-art algorithms that had been used to convert words to feature space first, then use the similar classification method. The PRSTM pipeline provides an update on the words embeddings which was costly at the beggining. The processing time was 15 days to produce word embeddings file of type comma separated value (CSV). The characteristic of PRSTM novelty is the non-standard words are treated all the same depending on how any word occur within text.

Another method had been developed for named entity recognition (NER). We addressed the same challenge which is non-standardised text but more targeted data within text which are the named entities in non standard format (informal). Our proposed approach Probabilistic Named Entity Recognition (PNER) aims to address this non-standardised challenge. Our approach trained on entities identified using state-of-the-art algorithms which had been applied on our streamed tweets. We extracted the co-occurrence patterns for these entities in shape of trigrams. Using the word embeddings created in PRSTM model, we find similar non-standard entities which are similar to the identified entities to be forwarded to the same state-of-the-art NERs. This challenge was just to find the ability of these NERs of detecting non-standard entities. The results shows that combining PNER with these NERs will enhance

the performance by 7% in total when applied to the annotated tweets. The annotation process was made using the state-of-the-art NERs then combine them with the similar non-standard format entities.

The proposed model could be considered as novel approach considering the pipeline from the input tweets reaching to identified entities. The performance of our approach will depend on the state-of-the-art NER and it is flexible to accept any NER model.

The third last approach that had been targeted in our thesis is the sentiment analysis for informal short noisy text. The dataset this time is a standard Twitter dataset called sentiment140. It consists of 1.6 million annotated tweets using emojies prospect to find the tweets polarity, then removed from tweet. These tweets are divided evenly to two classes consists of 800,000 negative and 800,000 positive tweets. Also, about 300 manually annotated tweets for testing. We produced our Probabilistic Relational Sentiment (PR_ Sentiment) approach to find the tweets polarity using words' pattern embeddings. We created these pattern embeddings by breaking down the tweets to trigrams and finding the frequency to each pattern. Then we isolated the neutral patterns which are the common patterns between the two annotated classes. A simple equation had been produced using the patterns frequencies and convert them to positive and negative features after normalizing the values for both classes as the total number of patterns for each class is not even. The summation of the input tested tweet patterns will decide if it is negative or positive depending on the summation result value.

Finally, these proposed methods were inspired by the change that is happening in English language. This change might stop in some point if there will be some tools that could predict the correct words in future but it is still a human nature to improvise and produce new ideas and suing new words to explain it.

6.3 Future Work

Natural language processing is a very wide field in computer science which is As we have produced three different methods for natural language processing, we will discuss the future work individually and as follow:

1. the proposed PRSTM is supervised model. Developing this proposed model to unsupervised will be a new proposed future work. This could be achieved by designing a clustering algorithm or developing one of the traditional approaches.

2. The development of PNER could be towards an independent approach that is linked to a gazetteer. Then extracting the non-standard entities to create a new dictionary that is automatically updated.

3. Our proposed sentiment approach could be developed by using more complicated co-occurrence relations. these relations will be invested in expanding the negative and positive polarity to multiple levels depending on how strong the meaning will be.

There are many ideas that could be developed from our proposed methods like the prediction of successive words using word embedding. These projects are still ideas and

could be implemented as individual published work.

Appendix A

Traffic Events Detection Using Auto-Generated bag-of-words

In chapter, our analytical model built on Machine learning to train Twitter data set to extract information related to traffic. This data set collected under two criteria: Geo-Tagged , and Non Geo-tagged Tweets using "Listener" which is a Python API. The difference between the two types is the geo-location that will simplify finding the tweet events location.

The 10,681,546 Geo-Tagged tweets were collected for the city of London in the duration between the 20th of April 2016 till the 1st of August 2016. The other 9,668,138 Non Geo-Tagged Tweets were collected from the 2nd August 2016 to the 22nd of November 2016 depending on a list of words related to traffic. The list of words related to traffic that had been chosen depending on a correlation between list of word from WordNet [112] and the TFL (Transport for London) [113] traffic accident data set. Several researches tried to detect events using statistical analysis but by relying on authentic data channels.

The weights of the Words selected will be given weights regarding several criteria that depends mainly on the words frequency, Time, and two stages of correlation between the words. Preliminary analysis was made on the previously mentioned data sets and they are shown in the following tables and figures:

Searched words / Date	Traffic	Accident	Road	Crash	Car	Closed	Broken	congestion	Total
18-02-16	76251	44620	10851	3964	14411	3117	312	2520	111833
19-02-16	70641	38305	9587	4356	15023	2904	224	2428	100340
20-02-16	61155	45503	6433	12057	23601	1824	212	934	102703
21-02-16	64070	54106	6873	11375	23981	2082	257	954	113351
22-02-16	94430	63823	12667	11255	24971	3040	452	2386	148155
23-02-16	70798	41012	9382	5651	16149	2685	267	1743	102801
24-02-16	116285	73297	14890	21424	39683	3754	460	2563	178702
25-02-16	110775	74971	14092	20274	38250	3922	483	2711	175503
26-02-16	103781	64554	13218	18278	32460	3053	316	2484	158348
27-02-16	79613	52319	9513	10573	24876	3010	283	1096	126100

Table A.1: Frequency of Some Selected Words from 1st Dataset

Table A.2 shows the frequency of some selected words from the 1st data set. These words were manually chosen after searching on dictionary for the most related words to traffic in general. It shows high frequency because the streaming was based upon these words with no certain location mentioned in text or geographical location. Thus, it has been collected from different places around the world on different times subject to the time zone of the tweet location.

Searched Words / Date	Traffic	Accident	Road	Crash	Car	Closed	Broken	congestion	total
21-04-16	191	110	1903	89	8522	88	344	29	106838
22-04-16	176	113	2030	100	8212	72	207	18	97262
23-04-16	98	95	1953	74	8310	64	161	6	102237
24-04-16	110	93	1926	72	8098	59	196	3	102831
25-04-16	165	119	1882	124	9500	87	232	7	104258
26-04-16	190	112	1952	64	8505	102	292	7	98516

Table A.2: The Frequency of some Selected Words from the 2nd Dataset

On the other hand, Table A.2 shows the frequencies of the same selected words from the 2nd data set. The frequency was less comparing to Table 1 due to the method of streaming that depends on the geographical location. London city was chosen in this research to collect tweets from. There is no common topic among the streamed tweets like in 1st data set but the geographical location.

Preliminary analysis was made on data by correlate words and if we could locate them in the same tweets. The frequency of these words shown in the following tables A.3 and A.4:

Searched words / Date	Traffic Accident	Car Crash	Road Closed	Car Broken	Total
18-02-16	7162	2051	419	66	111833
19-02-16	6592	2878	282	56	100340
20-02-16	3826	10953	197	60	102703
21-02-16	4494	10224	339	85	113351
22-02-16	8340	8853	434	93	148155
23-02-16	7147	3945	434	58	102801
24-02-16	8985	19157	556	134	178702
25-02-16	8499	18224	605	138	175503
26-02-16	8360	16097	543	91	158348
27-02-16	5739	9240	339	93	126100

Table A.3: The Frequency of Correlated Words from the 1st data set

Searched Words / Date	Traffic Accident	Car Crash	Road Closed	Car Broken	total
21-04-16	6	20	8	23	106838
22-04-16	10	27	8	32	97262
23-04-16	5	12	4	22	102237
24-04-16	3	14	7	31	102831
25-04-16	2	29	11	25	104258
26-04-16	2	18	15	17	98516

Table A.4: The Frequency of Correlated Words from the 2st data set

Tables A.3 and A.4 show less frequencies, this encourages locating more events by using ontological approach to make the most of the streamed data. These later tables demonstrate that such events exist within tweets and reporting them by ordinary people has turned them function as human sensors. For such extracted information to be valuable, the tweets need to be in real or near real time of the events occurrence.

In addition, the following charts show some patterns located by further preliminary analysis. The charts depend on data in tables A.3 and A.4:

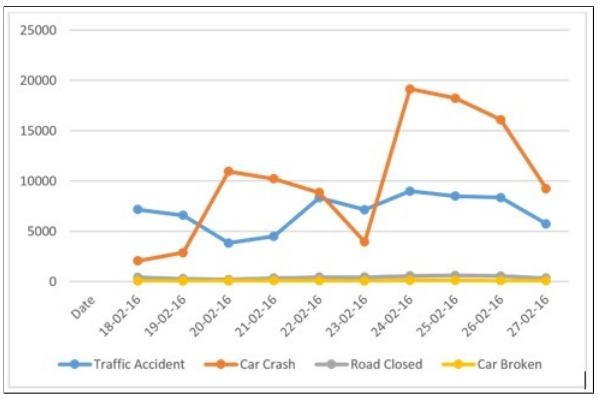


Figure A.1: Selected words frequencies for 10 days from the 1st dataset

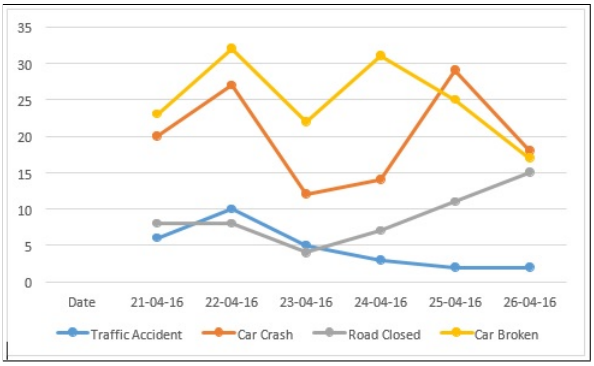


Figure A.2: Selected words frequencies for 10 days from the 1st dataset

Figure 4 shows no identified pattern. This is because it was collected from different places around the world with different time zones. Unlike the data in Figure 3 which shows peaks during weekend night (22th OF April 2016) for the combinations and decreasing in weekend (23rd and 24th of April) then it Increases back by the beginning of the next week. This demonstrates the activity of posting regarding traffic topic.

The aim of this chapter is to show the relation between the social media data and the actual data streamed from an authenticated parity. Figure A.3 shows the correlated tweets events with the event from traffic for London dataset on the same date.

Figure A.3: Shows the detected events in tweets which are related to real traffic events for London area.

Appendix B

Data Annotation

Validating the results on formal bases proven its weaknesses in short text analysis by training on external corpus like Wikipedia. Even if it worked to some extent, any change with the words or ill-formed words will bias the error ratio. Thus, keeping dynamic or real-time analysis and updating the data are the suggested solutions for the new informal words problem.

Using annotated data from the streamed tweet to train our proposed model produces more realistic results and overcome the previously mentioned challenges. Amazon Mechanical Turk was used i our approach for data annotation to label tweets to several topics. We designed the template using HTML language to ensure the easy and fast response for the annotator. The resulted tweets were labelled by to ten different classes by 110 annotators. The most common topics were the selected classes for these annotated tweets.

The first step to use the Amazon mechanical turk is creating a new account as “Requester”. Moving forward to use one of the provided templates that could provide the most relevant form as possible as most of time it has to be customized based on the task requirements. In our case. we have used the “Sentiment Project” template after customizing it to fit to our task.as shown in figure B.1.

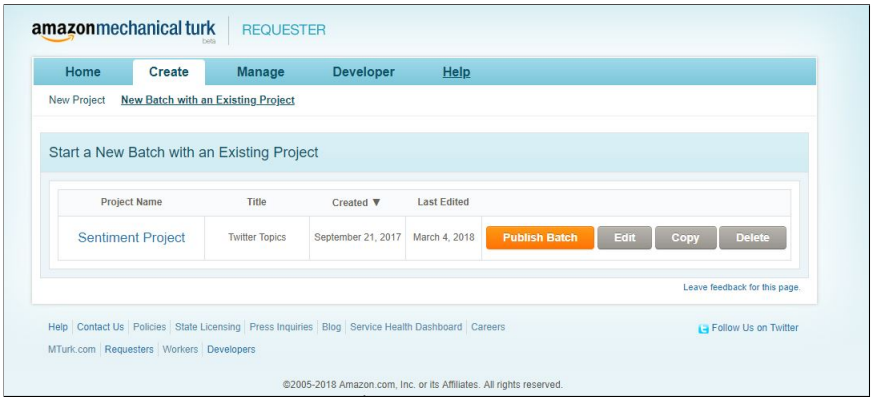
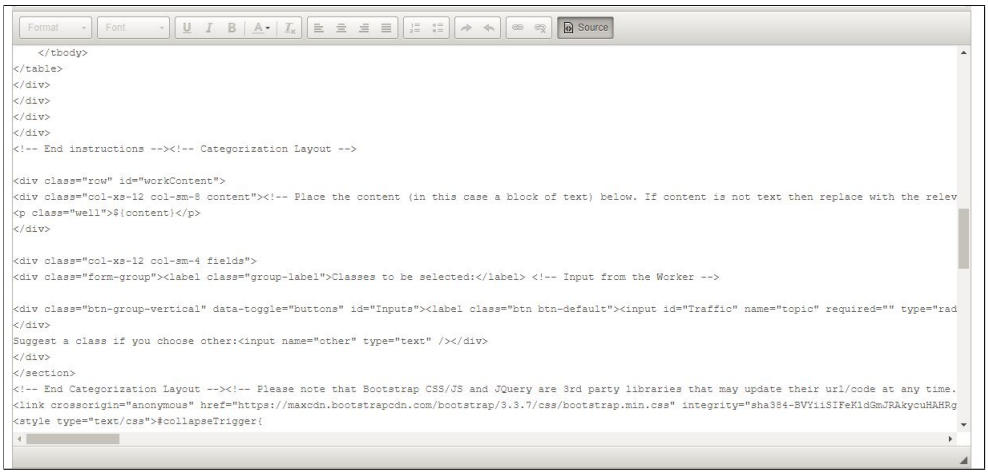


Figure B.1: Amazon Mechanical Turk

The customization process may require some hard coding to the template. We have done that by adding more suggested fields to the annotators. These fields are the most common classes and the last field will be the default class with a possibility of suggested class. A

sample of the HTML code is shown on figure B.2.



[

Figure B.2: Editing view for the template in HTML code

Amazon provided an extra detailed page for the annotator directions. We have provided full details for annotators as shown in figure B.3.

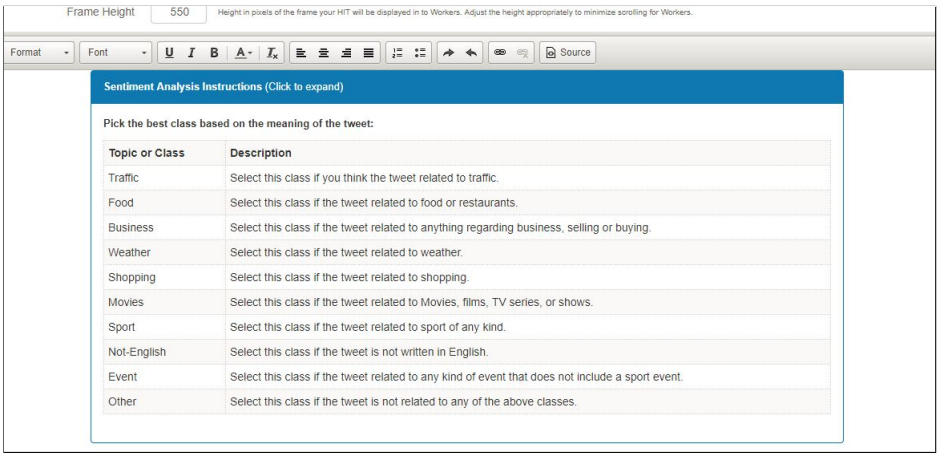


Figure B.3: Annotators directions to guide through the annotation process

After customizing the HTML code as required, the use interface for the annotators will be ready to work on. The "$(content)" field shown in figure B.4 will be filled with tweet from uploaded CSV file. This file contains the tweets we require to be annotated.

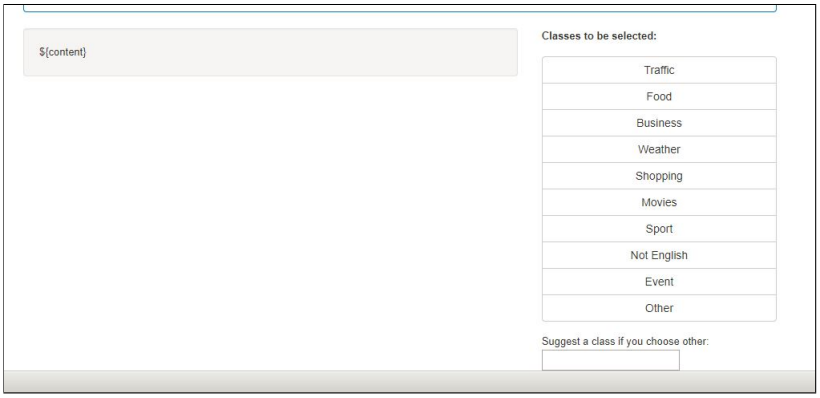


Figure B.4: The final annotation template user (annotator) interface

Another service provided by Amazon which is showing the annotation progress. Figure B.5 shows the progress of tweets annotation over time.

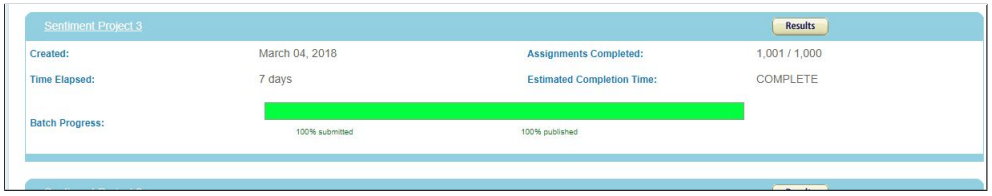


Figure B.5: The progress of annotation for the whole uploaded CSV tweets file

When the annotation process is done, the results will be stored of a CSV file as well with many information related to the ID of the annotator, time of annotation, how many hits per tweet, the class of tweet, and many other information as shown in table B.6.

Moreover, the annotated data will be passed to the PRSTM through the Machine Learning part that will decide the data to testing and training. This will be explained in details in the results chapter.

HITId	HITTypeId	Title	Description	Keywords	Reward	CreationT	Max Assignmer	Requester Annotation	Assignme	Auto Appl	Expiration	AssignmentId
3D0LPO3EABFSMWM6 2LGN	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3634BBTX0PAFD3S 1P2GWK0ODIYZIFT
3J9UN9O9J38HOQZOE 0D94	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	392SFO5CA9C0RW9 X29WVJ14G50JJU3
3CMV9YRYP3HULMO A3TZS	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3PEULRY6U9EEK3R EDQXRYX51EUXW7
3C8OQOM6JPHYQB44L BE5(	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3DYGAII7PMO4LE5E GDHA5WT3BE0PQ3
3IVEC1GSLPFHVFK6A JH9IF4	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3RWE2M8QWIQGU0O 2J0A3M1XQYV1N07
3X55NP42EOW9RTJJS00 C34	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	33ISQZVXPQ2HXOMR 6QU91R5F8GLCC8
3CRWSLD91KKSSFXSJ ISCA3	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3UXV6UZ1YZCGTX AGXGQ1P2WLM9RIW
335VBRURDJGR94190V LB05	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3HMIGG0U4MMSOOX JL6WTWW1FLDJY8X
3TZ0XG8CBU0A03RINS ZB8V	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3W8CV64QJ3F6GYQ4 GYYJAS1NFNP9HZ
3ZFRE2BDQ9UFLVA32Y 521I	3Q5NPJ2FFOG731 TMYTB7	Twitter Topics	Tweets topic classification	Tweet, Traffic, Sport, Busir	$.01	Sun Mar 0	1	BatchId:3139011;OriginalHit TemplateId:926	5400	604800	Sun Mar 11 2	3DH6GAKTYZ578KY6 UWW69GVJKMPYZ1

WorkerId	Assignment Status	Accept Time	Submit Time	WorkTime inSeconds	Lifetime A	Last7Days Appr	Inputcontent	Answerot	Answerto	Approve	Reject
A26866QOJMANB4	Submitted	Sun Mar 04 21:34:31 PST	Sun Mar 04 21:34:58 PST 2018	27	0% (0/0)	0% (0/0)	bunting going queens 90 th birthday buntin	{}	9	x	
A2UHF7UL7G0Y78	Submitted	Sun Mar 04 21:05:48 PST	Sun Mar 04 21:15:27 PST 2018	579	0% (0/0)	0% (0/0)	@marksfc50 never dull moment summer mi	{}	10	x	
A3GUJ6JD25FX7O	Submitted	Sun Mar 04 21:46:14 PST	Sun Mar 04 21:46:30 PST 2018	16	0% (0/0)	0% (0/0)	family fun day saturday 11 june 2016 see poi	{}	9	x	
A2UHF7UL7G0Y78	Submitted	Sun Mar 04 20:55:24 PST	Sun Mar 04 21:05:56 PST 2018	632	0% (0/0)	0% (0/0)	cmon hignfy time angus deaytons return sut	{}	10	x	
A3NND5DLB35WTX	Submitted	Sun Mar 04 20:55:53 PST	Sun Mar 04 20:56:09 PST 2018	16	0% (0/0)	0% (0/0)	@yesthatcherdead @lushrabbi much corruj	{}	8	x	
A2UHF7UL7G0Y78	Submitted	Sun Mar 04 21:06:29 PST	Sun Mar 04 21:16:49 PST 2018	620	0% (0/0)	0% (0/0)	@juliadsnow delivered tomorrow wednesd	{}	10	x	
A26866QOJMANB4	Submitted	Sun Mar 04 21:28:26 PST	Sun Mar 04 21:28:49 PST 2018	23	0% (0/0)	0% (0/0)	im actually really enjoying driving fiat 500 m	{}	9	x	
A2UHF7UL7G0Y78	Submitted	Sun Mar 04 21:25:42 PST	Sun Mar 04 21:26:43 PST 2018	61	0% (0/0)	0% (0/0)	headed swindon today research councils he	{}	10	x	
A2UHF7UL7G0Y78	Submitted	Sun Mar 04 21:31:11 PST	Sun Mar 04 21:33:43 PST 2018	152	0% (0/0)	0% (0/0)	@arron_hutt wish theyd admit screwed bad	{}	10	x	
A3GUJ6JD25FX7O	Submitted	Sun Mar 04 21:17:49 PST	Sun Mar 04 21:17:58 PST 2018	9	0% (0/0)	0% (0/0)	@e9_esident @nikewomen @nikeuk least	{}	7	x	

Figure B.6: Mechanical Turk output after annotation process

Appendix C

Part Of Speech tags

The part of speech tagging is a technique to represent any text in a form of sentence. Therefore, if the input was a normal text to the pos, the output will be a list of words and each word had been tagged. These tags represents the word's grammatical tag or its category, for example, the word "ship" is considered as a "noun" and its tag could be "NN" or "NNP" which is normal noun or normal noun in singular representation consequently. The following table C.1 shows the complete list of tags used in the NLTK package in Python language with some examples.

Tag	**Description and Example**
CC	coordinating conjunction
CD	cardinal digit
DT	determiner
EX	existential there (like: "there is" ... think of it like "there exists")
FW	foreign word
IN	preposition/subordinating conjunction
JJ	adjective 'big'
JJR	adjective, comparative 'bigger'
JJS	adjective, superlative 'biggest'
LS	list marker 1)
MD	modal could, will
NN	noun, singular 'desk'
NNS	noun plural 'desks'
NNP	proper noun, singular 'Harrison'
NNPS	proper noun, plural 'Americans'
PDT	predeterminer 'all the kids'
POS	possessive ending parent's
PRP	personal pronoun I, he, she
PRP$	possessive pronoun my, his, hers
RB	adverb very, silently,
RBR	adverb, comparative better
RBS	adverb, superlative best
RP	particle give up
TO	to go 'to' the store.
UH	interjection errrrrrrrm
VB	verb, base form take
VBD	verb, past tense took
VBG	verb, gerund/present participle taking
VBN	verb, past participle taken
VBP	verb, sing. present, non-3d take
VBZ	verb, 3rd person sing. present takes
WDT	wh-determiner which
WP	wh-pronoun who, what
WP$	possessive wh-pronoun whose
WRB	wh-abverb where, when

Table C.1: Tagging list of NLTK package in Python language

Bibliography

[1] Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A Greenwood, Diana Maynard, and Niraj Aswani. TwitIE: An Open-Source Information Extraction Pipeline for Microblog Text. *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 83–90, 2013.

[2] Alec Go, Richa Bhayani, and Lei Huang. Twitter Sentiment Classification using Distant Supervision. *Processing*, 150(12):1–6, 2009.

[3] Dong Wang, Boleslaw K. Szymanski, Tarek Abdelzaher, Heng Ji, and Lance Kaplan. The age of social sensing. *Computer*, 52(1):36–45, 2019.

[4] OED. New words list September 2017 | Oxford English Dictionary, 2017.

[5] Trisha Dowerah Baruah. Effectiveness of Social Media as a tool of communication and its potential for technology enabled connections: A micro-level study. *International Journal of Scientific and Research Publications*, 2(5):1–10, 2012.

[6] Edison Research; Triton Digital. Social Media Users Statistics, 2017.

[7] Leon Derczynski, Alan Ritter, Sam Clark, and Kalina Bontcheva. Twitter part-of-speech tagging for all: Overcoming sparse and noisy data. *Proceedings of the Recent Advances in Natural Language Processing*, (September):198–206, 2013.

[8] Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke Van Erp, Genevieve Gorrell, Raphaël Troncy, Johann Petrak, and Kalina Bontcheva. Analysis of named entity recognition and linking for tweets. *Information Processing and Management*, 51(2):32–49, 2015.

[9] David M Blei, Blei@cs Berkeley Edu, Andrew Y Ng, Ang@cs Stanford Edu, Michael I Jordan, and Jordan@cs Berkeley Edu. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[10] Gabriel Pedrosa, Marcelo Pita, Paulo Bicalho, Anisio Lacerda, and Gisele L. Pappa. Topic Modeling for Short Texts with Co-occurrence Frequency-Based Expansion. *Proceedings - 2016 5th Brazilian Conference on Intelligent Systems, BRACIS 2016*, pages 277–282, 2017.

[11] Xiaojun Quan, Chunyu Kit, Yong Ge, and Sinno Jialin Pan. Short and sparse text topic modeling via self-aggregation. *IJCAI International Joint Conference on Artificial Intelligence*, 2015-Janua(Ijcai):2270–2276, 2015.

[12] Liangjie Hong and Brian D. Davison. Empirical study of topic modeling in Twitter. *Proceedings of the First Workshop on Social Media Analytics - SOMA '10*, pages 80–88, 2010.

[13] Rishabh Mehrotra, Scott Sanner, Wray Buntine, and Lexing Xie. Improving LDA topic models for microblogs via tweet pooling and automatic labeling. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '13*, page 889, 2013.

[14] Vivek Kumar Rangarajan Sridhar. Unsupervised Topic Modeling for Short Texts Using Distributed Representations of Words. *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 192–200, 2015.

[15] Chenliang Li, Yu Duan, Haoran Wang, Zhiqian Zhang, Aixin Sun, and Zongyang Ma. Enhancing Topic Modeling for Short Texts with Auxiliary Word Embeddings. *ACM Transactions on Information Systems*, 36(2):1–30, 2017.

[16] X. Yan, J. Guo, Y. Lan, and X. Cheng. A biterm topic model for short texts. *WWW '13 Proceedings of the 22nd international conference on World Wide Web*, pages 1445–1456, 2013.

[17] Svitlana Vakulenko, Lyndon Nixon, and Mihai Lupu. Character-based Neural Embeddings for Tweet Clustering. pages 36–44, 2017.

[18] Georgiana Ifrim, Bichen Shi, and Igor Brigadir. Event detection in twitter using aggressive filtering and hierarchical tweet clustering. In Symeon Papadopoulos, David Corney, and Luca Maria Aiello, editors, *SNOW-DC@WWW*, volume 1150 of *CEUR Workshop Proceedings*, pages 33–40. CEUR-WS.org, 2014.

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA, 2013. Curran Associates Inc.

[20] Kohei Hayashi, Takanori Maehara, Masashi Toyoda, and Ken-ichi Kawarabayashi. Real-time top-r topic detection on twitter with topic hijack filtering. pages 417–426, 08 2015.

[21] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 556–562. MIT Press, 2001.

[22] S. Bai, C. Huang, B. Ma, and H. Li. Semi-supervised learning of language model using unsupervised topic model. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5386–5389, March 2010.

[23] E. Alegre, V. González-Castro, S. Suárez, and M. Castejón. Comparison of supervised and unsupervised methods to classify boar acrosomes using texture descriptors. In *2009 International Symposium ELMAR*, pages 65–70, Sep. 2009.

[24] Taneeya Satyapanich, Hang Gao, and Tim Finin. Ebiquity : Paraphrase and Semantic Similarity in Twitter using Skipgram. *Proceedings of SemEval*, (SemEval):51–55, 2015.

[25] Jingrui He, Wei Shen, Phani Divakaruni, Laura Wynter, and Rick Lawrence. Improving traffic prediction with tweet semantics. *IJCAI International Joint Conference on Artificial Intelligence*, pages 1387–1393, 2013.

[26] Fabian Abel, Qi Gao, Geert Jan Houben, and Ke Tao. Semantic enrichment of twitter posts for user profile construction on the social web. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6643 LNCS(PART 2):375–389, 2011.

[27] Fabian Abel, Ilknur Celik, Geert Jan Houben, and Patrick Siehndel. Leveraging the semantics of tweets for adaptive faceted search on twitter. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7031 LNCS(PART 1):1–17, 2011.

[28] F Lécué and S Tallevi-Diotallevi. STAR-CITY: semantic traffic analytics and reasoning for CITY. *Proceedings of the 19th . . .* , 318201:1–8, 2014.

[29] Declan Mchugh. Traffic Prediction and Analysis using a Big Data and Visualisation Approach. 2015.

[30] Hassan Saif, Yulan He, Miriam Fernandez, and Harith Alani. Adapting sentiment lexicons using contextual semantics for sentiment analysis of Twitter. *CEUR Workshop Proceedings*, 1329:5–12, 2014.

[31] Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7649 LNCS(PART 1):508–524, 2012.

[32] Arturo Montejo-Ráez, Eugenio Martínez-Cámara, M. Teresa Martín-Valdivia, and L. Alfonso Ureña-López. Ranked WordNet graph for Sentiment Polarity Classification in Twitter. *Computer Speech and Language*, 28(1):93–107, 2014.

[33] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. SentiWordNet 3 . 0 : An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining SentiWordNet. *Analysis*, 0:1–12, 2010.

[34] Chung-Chi Chen, Hen-Hsen Huang, and Hsin-Hsi Chen. NLG301 at SemEval-2017 Task 5: Fine-Grained Sentiment Analysis on Financial Microblogs and News. pages 847–851, 2018.

[35] Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel M. Cer, and David Jurgens, editors. *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*. Association for Computational Linguistics, 2017.

[36] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe : Global Vectors for Word Representation.

[37] Preslav Nakov, Sara Rosenthal, Svetlana Kiritchenko, Saif M. Mohammad, Zornitsa Kozareva, Alan Ritter, Veselin Stoyanov, and Xiaodan Zhu. Developing a successful semeval task in sentiment analysis of twitter and other social media texts. *Language Resources and Evaluation*, 50(1):35–65, Mar 2016.

[38] Mona T. Diab, Timothy Baldwin, and Marco Baroni, editors. *Proceedings of the 7th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2013, Atlanta, Georgia, USA, June 14-15, 2013*. The Association for Computer Linguistics, 2013.

[39] Amazon Mechanical Turk. No Title, 2005.

[40] Simon Kwoczek, Sergio Di Martino, and Wolfgang Nejdl. Predicting and visualizing traffic congestion in the presence of planned special events. *Journal of Visual Languages and Computing*, 25(6):973–980, 2014.

[41] Raymondus Kosala, Erwin Adi, and Steven. Harvesting real time traffic information from twitter. *Procedia Engineering*, 50:1–11, 2012.

[42] Ahmad Faisal Abidin, Mario Kolberg, and Amir Hussain. Improved Traffic Prediction Accuracy in Public Transport Using Trusted Information in Social Networks Using Kalman Filters for Traffic Arrival Prediction. 2011.

[43] Erik F Tjong, Kim Sang, and Fien De Meulder. Language-Independent Named Entity Recognition. 2003.

[44] D. Nadeau. A survey of named entity recognition and classification. *Linguisticae Investigationes*, (30):3–26., 2007.

[45] Steven J. DeRose. Grammatical category disambiguation by statistical optimization. *Comput. Linguist.*, 14(1):31–39, January 1988.

[46] Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. Named Entity Recognition in Tweets: An Experimental Study. *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, 2011.

[47] Jenny Rose Finkel and Christopher D. Manning. Nested named entity recognition. page 141, 2010.

[48] D Ramage, D Hall, R Nallapati, and C D Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*, pages 248–256, 2009.

[49] Ben Hachey, Will Radford, Joel Nothman, Matthew Honnibal, and James R. Curran. Evaluating entity linking with wikipedia. *Artificial Intelligence*, 194:130–150, 2013.

[50] Luis P. Prieto, María Jesús Rodríguez-Triana, Marge Kusmin, and Mart Laanpere. Smart school multimodal dataset and challenges. *CEUR Workshop Proceedings*, 1828:53–59, 2017.

[51] Joel Nothman, Nicky Ringland, Will Radford, Tara Murphy, and James R Curran. Learning multilingual named entity recognition from Wikipedia. 10 2017.

[52] Bo Han and Timothy Baldwin. Lexical normalisation of short text messages: Makn sens a #Twitter. *ACL-HLT 2011 - Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, 1:368–378, 2011.

[53] Bo Han, Paul Cook, and Timothy Baldwin. Automatically constructing a normalisation dictionary for microblogs. *EMNLP-CoNLL 2012 - 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Proceedings of the Conference*, (July):421–432, 2012.

[54] Axel Schulz, Eneldo Loza Mencia, and Benedikt Schmidt. A rapid-prototyping framework for extracting small-scale incident-related information in microblogs: Application of multi-label classification on tweets. *Information Systems*, 57:88–110, 2016.

[55] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Transactions on Knowledge and Data Engineering*, 25(4):919–931, 2013.

[56] Alexandre Passant, N U I Galway, Fabien Gandon, Inria Sophia Antipolis, Harith Alani, and The Open. Aligning Tweets with Events : Automation via Semantics. 1:1–5, 2009.

[57] Yusuke Hara. Behaviour analysis using tweet data and geo-tag data in a natural disaster. *Transportation Research Procedia*, 11:399–412, 2015.

[58] Mark Dredze, Mj Paul, Shane Bergsma, and Hieu Tran. Carmen: A twitter geolocation system with applications to public health. *Expanding the Boundaries of Health Informatics Using Artificial Intelligence: Papers from the AAAI 2013 Workshop*, pages 20–24, 2013.

[59] Liang Zhao, Feng Chen, Chang Tien Lu, and Naren Ramakrishnan. Dynamic theme tracking in Twitter. *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, pages 561–570, 2015.

[60] Bokai Cao, Francine Chen, Dhiraj Joshi, and Philip S. Yu. Inferring crowd-sourced venues for tweets. *Proceedings - 2015 IEEE International Conference on Big Data, IEEE Big Data 2015*, pages 639–648, 2015.

[61] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, USA, 1999.

[62] Tom Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[63] Vijay Raghavan, Peter Bollmann, and Gwang S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Trans. Inf. Syst.*, 7(3):205–229, July 1989.

[64] Yutaka Sasaki. The truth of the F-measure. *Teach Tutor mater*, pages 1–5, 2007.

[65] Baoli Li and Liping Han. Distance weighted cosine similarity measure for text classification. In Hujun Yin, Ke Tang, Yang Gao, Frank Klawonn, Minho Lee, Thomas Weise, Bin Li, and Xin Yao, editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2013*, pages 611–618, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[66] Lieve Hamers, Yves Hemeryck, Guido Herweyers, Marc Janssen, Hans Keters, Ronald Rousseau, and André Vanhoutte. Similarity measures in scientometric research: The jaccard index versus salton's cosine formula. *Information Processing & Management*, 25(3):315 – 318, 1989.

[67] Peter F. Brown, Vincent J. Della Pietra, Robert L. Mercer, Stephen A. Della Pietra, and Jennifer C. Lai. An estimate of an upper bound for the entropy of english. *Comput. Linguist.*, 18(1):31–40, March 1992.

[68] Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. BTM: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):2928–2941, 2014.

[69] Twitter. Streaming With Tweepy.

[70] Fred Morstatter, Jürgen Pfeffer, Huan Liu, and Kathleen M. Carley. Is the Sample Good Enough? Comparing Data from Twitter's Streaming API with Twitter's Firehose. pages 400–408, 2013.

[71] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[72] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schutze. *Introduction to Information Retrieval*. 2008.

[73] Yuan Zuo, Junjie Wu, Hui Zhang, Hao Lin, Fei Wang, Ke Xu, and Hui Xiong. Topic Modeling of Short Texts: A Pseudo-Document View Yuan. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16*, pages 2105–2114, 2016.

[74] Salvatore Gaglio, Giuseppe Lo Re, and Marco Morana. A framework for real-time Twitter data analysis. *Computer Communications*, 73:236–242, 2016.

[75] David M. Blei and Jon D. McAuliffe. Supervised Topic Models. pages 1–22, 2010.

[76] Yuan Zuo, Jichang Zhao, and Ke Xu. Word network topic model: a simple but general solution for short and imbalanced texts. *Knowledge and Information Systems*, 48(2):379–398, 2016.

[77] Thomas Hofmann. Probabilistic Latent Semantic Analysis. 2004.

[78] Qian Chen, Xin Guo, and Hexiang Bai. Semantic-based topic detection using Markov decision processes. *Neurocomputing*, 242:40–50, 2017.

[79] Kar Wai Lim, Changyou Chen, and Wray Buntine. Twitter-Network Topic Model: A Full Bayesian Treatment for Social Network and Text Modeling. pages 1–6, 2016.

[80] Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the COLING/ACL on Interactive Presentation Sessions*, COLING-ACL '06, pages 69–72, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

[81] Marco Bonzanini. Twitter most common words, 21-11-2017.

[82] William B. Cavnar and John M.Trenkle. N-gram based text categorization. pages 161–175, 1994.

[83] Asbjørn Ottesen, Steinskog Jonas, Foyn Therkelsen, and Björn Gambäck. Twitter Topic Modeling by Tweet Aggregation. *Proceedings of the 21st Nordic Conference of Computational Linguistics*, (May):77–86, 2017.

[84] Minglai Shao and Liangxi Qin. Text Similarity Computing Based on LDA Topic Model and Word Co-occurrence. (Sekeie):199–203, 2014.

[85] Cedric De Boom, Steven Van Canneyt, Thomas Demeester, and Bart Dhoedt. Representation learning for very short texts using weighted word embedding aggregation. pages 1–8, 2016.

[86] Esko Ukkonen. Algorithms for approximate string matching. *Information and Control*, 64(1-3):100–118, 1985.

[87] Pang-Ning Tan, 1956 Kumar, Vipin, and Michael Steinbach. *Introduction to data mining*. Boston : Pearson Addison Wesley, 1st ed edition, 2005. Includes bibliographical references and indexes.

[88] Yi Yang and Jacob Eisenstein. Overcoming language variation in sentiment analysis with social attention. *Transactions of the Association for Computational Linguistics*, 5:295–307, 2017.

[89] Manan Mohan Goyal, Neha Agrawal, Manoj Kumar Sarma, and Nayan Kalita. Comparison clustering using cosine and fuzzy set based similarity measures of text documents. 05 2015.

[90] Martin D. Buhmann and M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, New York, NY, USA, 2003.

[91] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf. Support vector machines. *IEEE Intelligent Systems and their Applications*, 13(4):18–28, July 1998.

[92] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, January 1967.

[93] Tingting Mu, John Y. Goulermas, Ioannis Korkontzelos, and Sophia Ananiadou. Descriptive document clustering via discriminant learning in a co-embedded space of multilevel similarities. *J. Assoc. Inf. Sci. Technol.*, 67(1):106–133, January 2016.

[94] Monica Marrero, Julian Urbano, Sonia Sanchez-Cuadrado, Jorge Morato, and Juan Miguel Gomez-Berbis. Named Entity Recognition: Fallacies, challenges and opportunities. *Computer Standards and Interfaces*, 35(5):482–489, 2013.

[95] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30(1):3–26, January 2007. Publisher: John Benjamins Publishing Company.

[96] Kalina Bontcheva, Leon Derczynski, Adam Funk, Mark A. Greenwood, Diana Maynard, and Niraj Aswani. TwitIE: An open-source information extraction pipeline for microblog text. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*. Association for Computational Linguistics, 2013.

[97] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Prismatic Inc, Steven J. Bethard, and David Mcclosky. The stanford corenlp natural language processing toolkit. 2014.

[98] John Bauer, Mihai Surdeanu, Steven Bethard, Christopher Manning, David McClosky, and Jenny Finkel. The Stanford CoreNLP Natural Language Processing Toolkit. pages 55–60, 2015.

[99] Jabir Alshehabi Al-ani and Maria Fasli. Probabilistic Named Entity Recognition for non-standard format entities using co-occurrence word embeddings. (December), 2019.

[100] Jabir Alshehabi Al-Ani and Maria Fasli. Probabilistic relational supervised topic modelling using word embeddings. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2035–2043, Dec 2018.

[101] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*, CONLL '03, page 142–147, USA, 2003. Association for Computational Linguistics.

[102] Jeff Pasternack and Dan Roth. The Wikipedia Corpus. 2008.

[103] Nancy Chinchor and Beth Sundheim. MUC-5 evaluation metrics. In *Fifth Message Understanding Conference (MUC-5): Proceedings of a Conference Held in Baltimore, Maryland, August 25-27, 1993*, 1993.

[104] Isabel Segura-Bedmar. Semeval-2013 task 9: Extraction of drug-drug interactions from biomedical texts (ddiextraction 2013). *Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, 01 2013.

[105] Y. Chen and Z. Zhang. Research on text sentiment analysis based on cnns and svm. pages 2731–2734, May 2018.

[106] R. Wagh and P. Punde. Survey on sentiment analysis using twitter dataset. pages 208–211, March 2018.

[107] T. LeCompte and J. Chen. Sentiment analysis of tweets including emoji data. pages 793–798, Dec 2017.

[108] T. Tran, D. Nguyen, A. Nguyen, and E. Golen. Sentiment analysis of marijuana content via facebook emoji-based reactions. pages 1–6, May 2018.

[109] J. Berengueres and D. Castro. Differences in emoji sentiment perception between readers and writers. pages 4321–4328, Dec 2017.

[110] Geoffrey I. Webb. *Naïve Bayes*, pages 713–714. Springer US, Boston, MA, 2010.

[111] Adam L. Berger, Vincent J. Della Pietra, and Stephen A. Della Pietra. A maximum entropy approach to natural language processing. *Comput. Linguist.*, 22(1):39–71, March 1996.

[112] Monica Monachini, Antonio Toral, and Rafael Mu. Named Entity WordNet. pages 741–747, 2007.

[113] Traffic for London. Open data for developers.