BIRMINGHAM CITY UNIVERSITY

DOCTORAL THESIS

Text Embedding-based Event Detection for Social and News Media

---

Author:
Hansi Hiranya HETTIARACHCHI

Supervisors:
Dr Mariam ADEDOYIN-OLOWE
Dr Jagdev BHOGAL
Prof Mohamed Medhat GABER

A thesis submitted in fulfillment of the requirements

for the degree of Doctor of Philosophy

in the

Data Analytics and Artificial Intelligence
School of Computing and Digital Technology

May 5, 2023

Declaration of Authorship

I, Hansi Hiranya HETTIARACHCHI, declare that this thesis titled, "Text Embedding-based Event Detection for Social and News Media" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

Abstract

Today, social and news media are the leading platforms that distribute newsworthy content, and most internet users access them regularly to get information. However, due to the data's unstructured nature and vast volume, manual analyses to extract information require enormous effort. Thus, automated intelligent mechanisms have become crucial. The literature presents several emerging approaches for social and news media event detection, along with distinct evolutions, mainly due to the variations in the media. However, most available social media event detection approaches primarily rely on data statistics, ignoring linguistics, making them vulnerable to information loss. Also, the available news media event detection approaches mostly fail to capture long-range text dependencies and support predictions of low-resource languages (i.e. languages with relatively fewer data). The possibility of utilising interconnections between different data levels to improve final predictions also has not been adequately explored.

This research investigates how the characteristics of text embeddings built using prediction-based models that have proven capabilities to capture linguistics can be used in event detection while defeating available limitations. Initially, it redefines the problem of event detection based on two data granularities, coarse- and fine-grained levels, to allow systems to tackle different information requirements. Mainly, the coarse-grained level targets the notification of event occurrences and the fine-grained level targets the provision of event details. Following the new definition, this research proposes two novel approaches for coarse- and fine-grained level event detections on social media, *Embed2Detect* and *WhatsUp*, mainly utilising linguistics captured by self-learned word embeddings and their hierarchical relationships in dendrograms. For news media event detection, this proposes a *TRansformer-based Event Document classification architecture (TRED)* involving long-sequence and cross-lingual transformer encoders and a novel learning strategy, *Two-phase Transfer Learning (TTL)*, supporting the capturing of long-range dependencies and data level interconnections.

All the proposed approaches have been evaluated on recent real datasets, covering four aspects crucial for event detection: accuracy, efficiency, expandability and scalability. Social media data from two diverse domains and news media data from four high- and low-resource languages are mainly involved. The obtained results reveal that the proposed approaches outperform the state-of-the-art methods despite the data diversities, proving their accuracy and expandability. Additionally, the evaluations on efficiency and scalability adequately confirm the methods' appropriateness for (near) real-time processing and ability to handle large data volumes. In summary, the achievement of all crucial requirements evidences the potential and utility of proposed approaches for event detection in social and news media.

Acknowledgements

This thesis would not have been possible without the support of several wonderful people and organisations who helped me throughout my PhD studies. I keep the details brief, but I hope I have not missed anyone important.

Initially, I would like to express my sincere gratitude to my Director of Studies, Dr Mariam Adedoyin-Olowe, for her continuous guidance and support throughout my PhD journey. I am particularly thankful to her for responding to my very first email, where everything had begun, even without knowing me, and for trusting and encouraging me since that day. Without her generous actions, I will not be able to come this far in this PhD. Also, I am very grateful for our regular meetings, which encourage me to successfully complete my research and thesis write-up, adhering to the timeline.

I want to extend my sincere acknowledgement to my other supervisors, Dr Jagdev Bhogal and Prof Mohamed Gaber. I would like to thank Dr Jagdev Bhogal, particularly for her comments and compliments, which encouraged me throughout this PhD journey. It has indeed been an honour and pleasure to work with Prof Mohamed Gaber, who is one of the top scientists in the world. I am incredibly grateful for his scientific and technical knowledge, which intensely benefited my research, and his vision, which tremendously influenced the direction of my PhD journey. Also, I greatly appreciate his continued encouragement and feedback to improve the quality of my work despite his busy schedule. Above everything, I am very thankful for his understanding and invaluable life lessons.

Apart from my supervisory team, members of my PhD progression panel significantly helped me improve the quality of my work. I am very grateful to Dr Ryan Stables, Dr Shereen Fouad, Dr Quanbin Sun, Dr Edlira Vakaj and Dr Chris Creed for their time allocated to me and for constructive feedback. Also, I would like to thank all the members of Doctoral Research College for providing me with administrative support during my PhD. Mainly, I want to thank Sue Witton for her timely assistance.

My sincere appreciation also goes to the members who organised/taught the Postgraduate Certificate in Research Practice 2019/20 course of the faculty of Computing, Engineering and the Built Environment (CEBE) at Birmingham City University (BCU), which provided me with valuable insights to conduct my PhD research successfully. Especially, I would like to thank Prof Peter Larkham and Dr Chris Creed for their commitment to the success of this course. Furthermore, it is a pleasure to conduct research under the Data Analytics and Artificial Intelligence (DAAI) research group of BCU. I am grateful to all members for the quality time spent together. I would like to especially thank my colleagues Khadijah Hanga, Zakaria Senousy and

Lorraine Chambers for their friendship which helped me settle down in BCU and England quickly. Also, I am very grateful for every piece of advice and feedback I received from the members of DAAI and BCU staff.

Additionally, I thank BCU for offering me a fee waiver to conduct my PhD studies. Without this financial support, it would be infeasible for me to come this far with my studies. Also, I am grateful to Twitter Developer Platform, JetBrains and Google Colaboratory for their support for academic researchers.

I would like to offer a special thanks to Dr Upali Kohomban for giving me my first opportunity to work on a machine learning and natural language processing-based project during my undergraduate internship program. Undoubtedly, his support and encouragement paved the path of my research journey, which led to this PhD. Additionally, I would like to thank the Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE) 2021 organising committee, including Dr Ali Hürriyetoğlu for organising interesting shared tasks to tackle real-world problems and sharing resources, which I found very useful during my PhD journey.

Also, I would like to thank my examiners, Dr Ella Haig and Dr Shadi Basurra. I highly appreciate their valuable comments and the time they spent to read and evaluate the thesis. I am also grateful to the chair of my viva, Prof Peter Larkham, for smoothly handling the whole process.

None of these would be possible without my family. I am forever grateful to my mother, Chitra Senehelatha, for her unconditional love, support, understanding, encouragement and sacrifices to make me who I am. I do not have enough words to express my gratitude to her. She is the main reason for everything I possess today. Also, I am very thankful to my husband, Tharindu Ranasinghe, for being a great friend, companion and comforter to me throughout the ups and downs of my life. I highly appreciate his support throughout this course, including reading this thesis multiple times. Finally, I would like to thank my father, Ariyasiri Hettiarachchi, and brother, Indeera Hettiarachchi, for their support and love throughout my life and, particularly during my PhD to come this far.

Contents

List of Figures

List of Tables

List of Publications

The following conference and journal papers have been published along with this research.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (Aug. 2021a). "DAAI at CASE 2021 Task 1: Transformer-based Multilingual Socio-political and Crisis Event Detection". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 120–130. DOI: 10.18653/v1/2021.case-1.16. URL: https://aclanthology.org/2021.case-1.16.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2021b). "Embed2Detect: Temporally Clustered Embedded Words for Event Detection in Social Media: Extended Abstract". In: *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–2. DOI: 10.1109/DSAA53316.2021.9564110. URL: https://ieeexplore.ieee.org/document/9564110.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2022a). "Embed2Detect: Temporally clustered embedded words for event detection in social media". In: *Machine Learning* 111, pp. 49–87. DOI: 10.1007/s10994-021-05988-7.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2023a). "TTL: transformer-based two-phase transfer learning

for cross-lingual news event detection". In: *International Journal of Machine Learning and Cybernetics*. DOI: 10.1007/s13042-023-01795-9.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2023b). "WhatsUp: An event resolution approach for co-occurring events in social media". In: *Information Sciences* 625, pp. 553–577. ISSN: 0020-0255. DOI: 10.1016/j.ins.2023.01.001.

Hettiarachchi, Hansi, Doaa Al-Turkey, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2022b). "TED-S: Twitter Event Data in Sports and Politics with Aggregated Sentiments". In: *Data* 7.7. ISSN: 2306-5729. DOI: 10.3390/data7070090. URL: https://www.mdpi.com/2306-5729/7/7/90.

List of Abbreviations

ACE	**A**utomatic **C**ontent **E**xtraction
BERT	**B**idirectional **E**ncoder **R**epresentations from **T**ransformers
CASE	**C**hallenges and **A**pplications of Automated Extraction of **S**ocio-political **E**vents from Text
CoNLL	**Co**nference on Computational **N**atural **L**anguage **L**earning
DL Sim.	**D**endrogram **L**evel Similarity
GLOCON	**Glo**bal **Con**tentious Politics Dataset
GT	**G**round **T**ruth
HAC	**H**ierarchical **A**gglomerative **C**lustering
LDL Sim.	**L**ocal **D**endrogram **L**evel Similarity
LFC	**L**iverpool **F**ootball **C**lub
mBERT	**M**ultilingual **BERT**
MLM	**M**asked **L**anguage **M**odelling
MTL	**M**ulti-**t**ask **L**earning
MUFC	**M**anchester **U**nited **F**ootball **C**lub
RoBERTa	**R**obustly **o**ptimized **BERT** **a**pproach
TED	**T**witter **E**vent **D**ata
TED-S	**T**witter **E**vent **D**ata with **S**entiments
TL	**T**ransfer **L**earning
TRED	**TR**ansformer-based **E**vent **D**ocument classification architecture
t-SNE	**t**-distributed **S**tochastic **N**eighbor **E**mbedding
TTL	**T**wo-phase **T**ransfer **L**earning
TW	**T**ime **W**indow
XLM	Cross-lingual **L**anguage **M**odel

Dedicated to my mother,

Marasinghe Mudiyanselage Chitra Senehelatha

(1950-2020)

the greatest person I have ever met,
for her unconditional love, support and encouragement,
which defined who I am today.

Chapter 1

Introduction

With the technological advancements that happened over the past years, there has been remarkable worldwide digital growth. According to a survey conducted in January 2022, there were 4.95 billion internet users, approximately 62.5% of the total population (Kemp, 2022). Also, there exist many platforms which encourage people to be active via the internet and generate or access a vast amount of data. To mention a few figures, an analysis done in 2021 revealed that roughly 5.7 million Google searches happen and 575,000 tweets post in every minute (James, 2021). Overall, social media platforms and online news agencies were found to generate a large proportion of data available on the internet (Balali et al., 2020). Even though online news media gradually attracted the community's attention over the past few decades, social media indicated a drastic increase in usage, estimated as 10.1% annual change at the beginning of 2022; more than two times of internet user growth (Kemp, 2022).

Finding information and keeping up-to-date with news and events are the top reasons why people access the internet, with 61% and 53% votes according to a survey in early 2022 (Kemp, 2022). It is not only news media which produce newsworthy content today. Social media also play a crucial role, generating diverse information such as personal updates and opinions, including breaking news. Further confirming this fact, Shearer and Gottfried (2017) revealed that 67% of American adults get news from social media in 2017. In

addition to the diversity of information, the fast dispersal supported by large user bases spread worldwide and availability via different web-based services made social media more popular (Castillo et al., 2011). Also, in some cases, social media was found to broadcast news faster than traditional news media, emphasising its capabilities (Kwak et al., 2010). Following these trends, we can also witness a tendency in news services such as BBC and CNN to use social media actively to publish news to a huge user base instantly in the present.

It is not only the general public who seeks newsworthy or important information through the internet. For example, detecting breaking disaster events will be helpful for multiple parties, including rescue teams, security forces and policymakers, to take immediate actions (Nugent et al., 2017). Also, public opinion can be gathered from data on the internet, especially from social media, to understand the future adjustments that are necessary to make by organisations or governments to prevent crises or escalations (van der Meer and Verhoeven, 2013). Furthermore, the available data can be organised into knowledge bases to support decision-making processes and studies in various areas such as sociology and political science (Hürriyetoğlu et al., 2021b). However, it is impractical to manually analyse data to extract newsworthy content due to the huge volume and the unstructured nature of the vast majority (Nugent et al., 2017; Balali et al., 2020). Thus, to effectively utilise available data, the requirement of automated intelligent processes to extract information is crucial (Small and Medsker, 2014).

This research targets addressing this crucial requirement, developing automated intelligent processes to extract information from data. The rest of this chapter introduces this work in detail, initiating with the research focus, mainly considering internet users' requirements and recent trends in data generation and processing techniques in Section 1.1. Subsequently, Section 1.2 introduces the aim and objectives of this research and Section 1.3 summarises

the contributions made achieving the objectives. Finally, Section 1.4 provides a detailed overview of the rest of this thesis.

1.1 Research Focus

As we are aware, a vast majority of data available via the internet, which we access to obtain newsworthy details, is unstructured. Thus, we require accurate and efficient information extraction mechanisms to effectively utilise the huge data volume available. We discuss the idea behind information extraction, and its subarea focused on by this research in Section 1.1.1. The targeted data sources are described in Section 1.1.2. Section 1.1.3 provides an overview of the evolution of Natural Language Processing (NLP) techniques and details of the techniques focused on by this research. Finally, Section 1.1.4 summarises the requirements which need to be fulfilled by developed systems.

1.1.1 Information Extraction

Information Extraction (IE) is an area of NLP that deals with finding factual information in free text/unstructured data (Piskorski and Yangarber, 2013). Unstructured data does not have a clear, semantically overt and easy-for-a-computer structure, being the opposite of structured data (Manning et al., 2008). However, even though we generally consider the text as unstructured data, it actually has some implicit structure. Thus, IE can be more precisely mentioned as a process that targets making the text's semantic structure explicit to facilitate its effective usage (Grishman, 2015). There are a lot of techniques or subareas under IE, such as named entity tagging, relation extraction, coreference resolution and event detection, which support obtaining different levels and types of information from text.

Among different subareas under IE, we targeted event detection in this research. Event detection is the process of analysing data streams to automatically discover events described in data (Mellin and Berndtsson, 2009; Panagiotou et al., 2016). Events can be incidents such as a goal scored during a football match, a result announcement of a parliament vote or a fire at a public location. For further clarity, we defined the concept of *event* in Chapter 2. We especially selected this area considering the internet users' top requirements of accessing data (i.e. finding information and keeping up-to-date with news and events). Also, this area attracted researchers' attention over past years due to the widespread availability of huge data volumes which hold information on various events (Panagiotou et al., 2016; Hürriyetoğlu et al., 2021a). Furthermore, this area can be generalised across different domains and data sources being useful for multiple parties, including the general public, organisations, governments and rescue teams.

We further divided event detection into two categories: (1) coarse-grained level and (2) fine-grained level based on data granularity. This decision is mainly encouraged by the diversity in data available via the internet and user requirements. At the coarse-grained level, we targeted notifying users about event occurrences so that they can take necessary further actions to obtain the required event details. Such a mechanism is helpful in situations where sensitive events are targeted and cannot completely rely on automated processes. At the fine-grained level, we targeted extracting event-described text segments at event occurrences to provide detailed insight. Such a mechanism is helpful when full automation is preferred to get event details concisely and quickly without involving any manual process.

1.1.2 Data Sources

Several platforms or sources generate and disperse data via the internet. Among them, social and news media play key roles in distributing newsworthy content today (Balali et al., 2020). Thus, these platforms are very popular among internet users and widely accessed to obtain information. Also, these platforms indicate notable growth in users and data generation, emphasising the importance of automated mechanisms to extract information from data effectively. Considering these facts, we focused on developing event detection approaches for data from social and news media in this research. We especially targeted textual data, considering its wide availability across these media and informativeness.

TABLE 1.1: Main distinctions in social and news media data

Social Media	**News Media**
Short text posts	Long text documents
Less organised data, reporting only the aware details	More organised data, reporting comprehensive summaries
Written by the general public with diverse knowledge levels	Written by professionals in news agencies
Dynamic/rapidly changing data generation	(Near) Static/non-rapidly changing data generation
Contain various types of information: newsworthy details, personal updates, opinions, recommendations, etc.	Only contain newsworthy details

Additionally, social and news media have diverse characteristics, as summarised in Table 1.1. This diversity encourages the generation of various types of information useful for different scenarios. The dynamicity, along with the real-timeliness, allow social media to propagate important information instantly all over the world. Also, the involvement of various parties to generate data help expose different aspects of an event. Contrarily, news media data are more comprehensive and reliable as only professionals generate them. Thus,

they are more helpful in generating complete event summaries or knowledge bases to support future studies.

1.1.3 NLP Techniques

Natural Language Processing (NLP) is a subfield of Artificial Intelligence (AI) that enables computers to understand and analyse human language/naturally occurring texts (Liddy, 2001). To be processed by computers, the text needs to be converted into some machine-readable format or numerals. Since all the performances of later computational techniques depend on the successful conversion of text into numerals, a high focus was given to the conversion techniques. Figure 1.1 illustrates some key algorithms designed to make the text understandable to computers along the NLP journey.

FIGURE 1.1: Evolution of NLP models [1]

NLP was initiated with simple models such as Bag of Words (BoW). BoW counts each word's occurrence in a document and generates a vector with dimensions equivalent to the vocabulary size occupied by counts per document. However, the count gets dominated by common words (e.g. stopwords), which would not be much informative. To overcome this issue, Term Frequency-Inverse Document Frequency (TF-IDF) scoring mechanism was introduced. It reduces the weight of common words while highlighting the distinct words which may contain helpful information. However, it is not sufficient to rely only on statistical measures when processing a language built using syntax and semantics, mainly (Pollard and Sag, 1988). Syntax defines the

[1] This figure is adapted from (Rahane and Pawar, 2020)

arrangement of words in sequences, and semantics describes the connections between words and their meanings. Considering this fact, co-occurrence matrices were introduced. They are square matrices which store the co-occurrence of words in a context. However, a matrix requires a large amount of memory. Also, the larger the vocabulary, the matrix will be large and sparse similar to the BoW vectors.

These limitations led to more advanced techniques involving neural networks. Word2Vec (Mikolov et al., 2013a) is one of the first prediction-based modelling techniques, which attracted wide attention from the NLP community. This technique uses a shallow neural network to generate word embeddings, considering the syntax and semantics of the text. However, models such as Word2Vec are not capable of capturing homonyms or contextual impact on a word's meaning. Transformer models were introduced lately to overcome this limitation (Vaswani et al., 2017). They use deep neural networks and attention mechanisms to capture text's contextual details and long-range dependencies. Following transformers, BERT architecture was proposed based on the bidirectional transformer encoder to generate contextual text representations, providing state-of-the-art results in many NLP benchmarks (Devlin et al., 2019). It encouraged the development of various transformer-based models, including cross-lingual models such as XLM-R (Conneau and Lample, 2019), which recently gained wide attention.

When detecting events described in social and news media text, great attention should be given to underlying linguistics (syntax and semantics). They allow a language to express the same idea using different word sequences and vice versa. This effect can be noticed to a certain level in news media as several reporters write news articles. However, it is enormous in social media as different types of people post. Thus, a serious amount of important information can be lost without properly capturing linguistics. To fulfil this crucial

requirement, we decided to focus on prediction-based modelling/text embedding learning approaches in this research. We especially targeted Word2Vec and transformer-based models considering their proven abilities to capture linguistics, recency, popularity and outstanding performance. Our decision is also encouraged by the limitation we recognised in available event detection approaches, which are further described in Chapter 2. For simplicity, we will refer to the transformer-based language models or encoders (e.g. BERT) as *'transformers'* in the below content.

1.1.4 System Requirements

Depending on users and targeted tasks, systems have different requirements to fulfil to be productive. We recognised the following qualities are crucial when designing a system to detect events from social and news media and targeted their achievement in this research.

- ***Accuracy***: Achieving a high degree of accuracy in results/predictions is the main target of an automated system. This fact is also highly applicable to event detection because the extracted information will not be useful for any party if the results are not sufficiently accurate. Different metrics such as recall, precision, and F1 are commonly used to evaluate a method's accuracy.

- ***Efficiency***: Since data generation and dispersal happen so fast through the internet, event detection mechanisms need to be efficient to collect information in (near) real-time before they get outdated. Thus, the time takes to process data or make predictions should be sufficiently minimal compared to the data generation time.

- ***Expandability***: Social and news media provide access to a wide range of data which vary from different factors such as domain, language and platform. Thus, to support effective event detection from any data, the method should be designed with expandability: the ability to extend to new domains, languages or platforms easily. It is important to design the method without relying on any domain-, language- or platform-specific features and external knowledge bases that introduce restrictions to provide expandability.

- ***Scalability***: Recent surveys indicated a high volume of data generation via the internet with a high growth rate (Kemp, 2022; James, 2021). Thus, the event detection mechanisms should be scalable to handle increasing data volumes efficiently with affordable resources to the majority to be beneficial for many parties ranging from the general public to high-profit organisations.

1.2 Aim and Objectives

This research aims to investigate how different capabilities of prediction-based text embedding learning approaches can be utilised for effective event detection from textual data in social media data streams and news media articles while overcoming the limitations of traditional approaches. To achieve this aim, the following objectives have been considered.

1. Conduct a thorough literature review covering the areas of social media event detection and news media event detection to understand the strengths and weaknesses of available approaches.

2. Analyse the characteristics of social media data streams and limitations in the available approaches and, focusing on the findings, develop effective methods for event detection by utilising the competencies of embedding learning approaches.

3. Evaluate the approaches proposed for social media event detection using recent real datasets and compare the performance with state-of-the-art methods.

4. Analyse the characteristics of news media articles and limitations in the available approaches and, focusing on the findings, develop effective methods for event detection by utilising the competencies of embedding learning approaches.

5. Evaluate the approaches proposed for news media event detection using recent real datasets and compare the performance with state-of-the-art methods.

1.3 Contributions

By fulfilling the objectives of this research, we made the following contributions. Each part (social media and news media event detection) has its original contributions, as summarised below.

Social Media Event Detection

1. We proposed a novel method named *Embed2Detect* (Hettiarachchi et al., 2021b; Hettiarachchi et al., 2022a) for coarse-grained level event detection in social media or to identify event occurred time windows in (near) real-time to notify users about events, involving linguistical features in the underlying text, overcoming a major limitation in available approaches, the less semantic involvement. To capture linguistic variations over time,

we utilised self-learned word embeddings along with a text/cluster similarity measure, which we introduced as *Dendrogram Level (DL) Similarity*.

2. We proposed a novel method named *WhatsUp* (Hettiarachchi et al., 2023b) for fine-grained level event detection in social media or to identify text of co-occurred events at temporal event occurrences in (near) real-time to automate the complete flow of event detection, involving statistics and linguistics of underlying data in an unsupervised manner. We introduced a localised version of DL Similarity named *Local Dendrogram Level (LDL) Similarity*, a technique for text similarity change calculation named *Positive Similarity Change* and a novel textual change-based clustering approach along with WhatsUp.

3. We prepared and published social media datasets (*TED* and *TED-S*) representing two diverse domains (i.e. sports and politics) with ground truth event labels, addressing the lack of recent data availability in the area of social media event detection (Hettiarachchi et al., 2022a; Hettiarachchi et al., 2022b). We also designed a comprehensive set of metrics/framework that evaluates both temporal and textual details of detected events and released its implementation, aiming to support related evaluations in a unified way.

News Media Event Detection

4. We proposed a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformer models for coarse-grained level event detection in news media or event-described news article identification (Hettiarachchi et al., 2021a). The involvement of long-sequence models helped capture long-range dependencies in text effectively, outperforming the state-of-the-art methods.

5. We proposed a novel learning strategy named *Two-phase Transfer Learning (TTL)* (Hettiarachchi et al., 2023a), involving different levels of data granularity (i.e. sentence and token levels) and the capabilities of state-of-the-art transformer models. We applied TTL for fine-grained level event detection in news media or to extract event-described sentences and words/tokens from news articles and discussed its effectiveness and applicability, introducing a new research direction.

6. We empirically evaluated how the performance of news media event detection at the document, sentence and token levels can be improved for high- and low-resource languages involving different language-based learning strategies (i.e. multilingual, zero-shot and transfer learning) and the characteristics of state-of-the-art transformer models and proposed architectures.

Additionally, Figure 1.2 illustrates an overview of this research's contributions. This diagram represents the outline of this research, the involvement of text embeddings for this work and concepts proposed during this study, along with the main contributions made.

1.4 Thesis Organisation

The rest of this thesis is organised as follows.

Chapter 2: Event Detection from Text. This chapter provides an overview of the work done by previous research in event detection from text. It defines the concept of *event* based on several published definitions. It also reviews the methods used for event detection in social and news media,

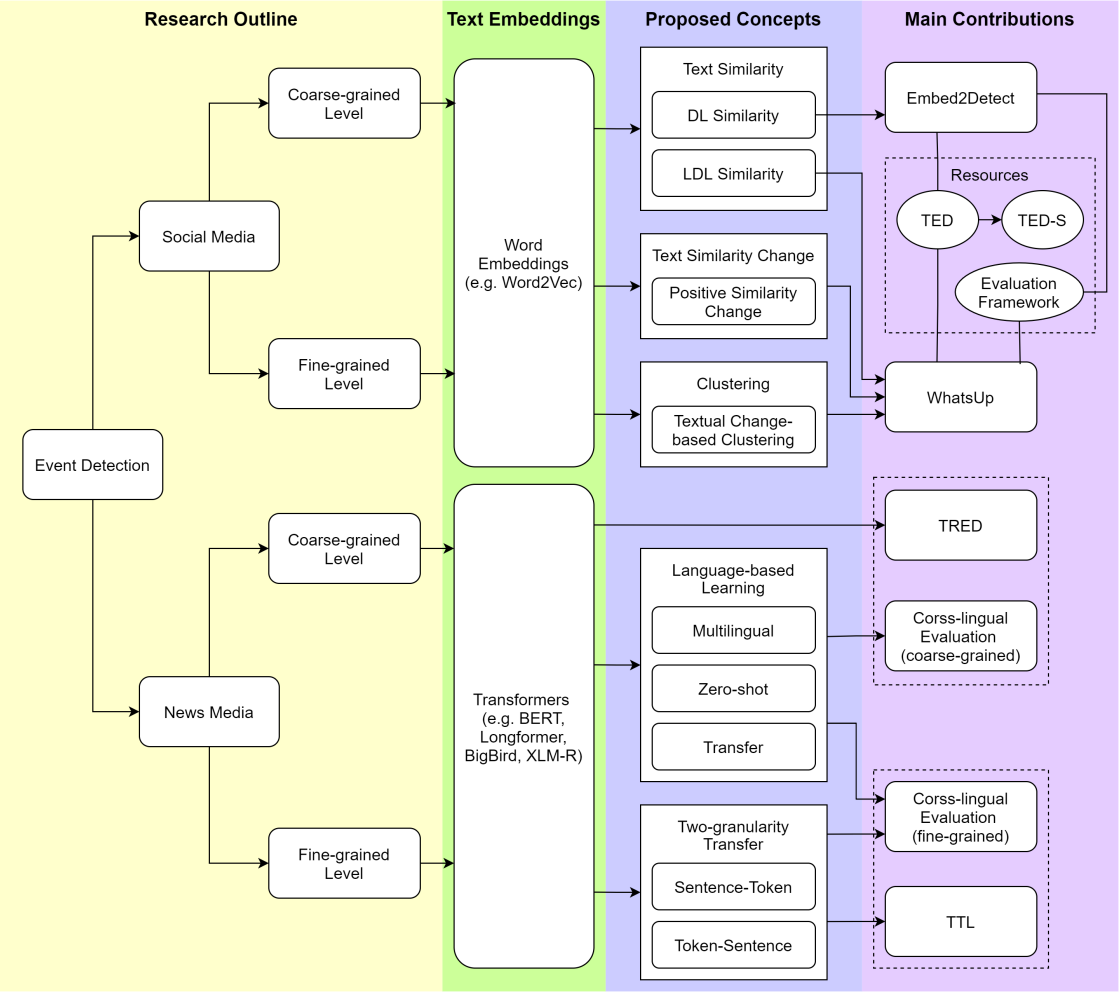


FIGURE 1.2: Overview of contributions

recognising different categories and their evolutions. Following the review, it discusses some of the gaps/open challenges recognised in the areas of social and news media event detection, which aim to be addressed by this research.

Part I: Social Media Event Detection

Chapter 3: Introduction to Social Media Event Detection. This chapter introduces the idea behind social media event detection, providing an overview of Part I of the thesis and defining the problem comprehensively. It also describes the details of social media datasets prepared for evaluations, covering two diverse domains (i.e. sports and politics),

including data collection, cleaning and ground truth preparation. Furthermore, it details the metrics designed to perform overall evaluations covering temporal and textual event details. Additionally, the background concepts (i.e. Word2Vec and dendrograms) used to design the approaches proposed in Chapters 4 and 5 are also described in this chapter.

Chapter 4: Embed2Detect: Coarse-grained Level – Event Window Identification. This chapter presents the first main contribution of this research. It introduces *Embed2Detect*, a novel approach proposed for coarse-grained event detection in social media – event window identification, to notify users about events in data streams, relying on the temporal variations of tokens and their hierarchical relationships captured using self-learned word embeddings and dendrograms. This chapter further details the conducted experiments along with the comparisons with several recently published methods. Additionally, it discusses the possibilities of involving other state-of-the-art text embeddings. It ends with conclusions and ideas for the future directions of this subarea.

Chapter 5: WhatsUp: Fine-grained Level – Co-occurring Event Identification. This chapter introduces the second main contribution of this research. It presents *WhatsUp*, a novel approach proposed for fine-grained event detection in social media – co-occurring event identification, to extract event-described texts at temporal event occurrences, involving self-learned word embeddings and unsupervised learning techniques. It also proposes improvements for event window identification, introducing techniques that improve temporal textual change calculation. Furthermore, it reports the conducted experiments and their results, along

with the comparisons with recently published methods. Finally, it summarises the conclusions and ideas for future directions, targeting a more comprehensive event output.

Part II: News Media Event Detection

Chapter 6: Introduction to News Media Event Detection. This chapter introduces the idea behind news media event detection, providing an overview of Part II of the thesis and defining the targeted problem in detail. It also describes the datasets and evaluation metrics involved in experiments on news media event detection. Furthermore, it describes the concepts behind transformer-based language model architecture, which are used to design the approaches proposed in Chapters 7 and 8.

Chapter 7: TRED: Coarse-grained Level – Event Article Identification. This chapter presents the third main contribution of this research. It proposes a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformer models for coarse-grained event detection in news media – event article identification, to notify users about event mentions in news articles. It also reports an empirical evaluation of how the performance of event article identification can be improved for high- and low-resource languages using transformer models and different learning techniques. Finally, it summarises the conclusions made following the experiments and ideas for future research in this subarea.

Chapter 8: TTL: Fine-grained Level – Event Sentence and Word Extraction. This chapter describes the fourth main contribution of this research. It introduces *Two-phase Transfer Learning (TTL)*, a novel learning strategy proposed utilising the capabilities of transformer models to learn from different levels of data granularity. TTL is applied for fine-grained

event detection in news media – event sentence and word extraction, to facilitate an effective event detail capturing using available labelled data. An empirical evaluation of how the performance of event sentence and word extraction can be improved for high- and low-resource languages utilising the transformer models and different learning techniques along with TTL is also reported in this chapter. It ends with conclusions and ideas for future directions on considered subareas.

Chapter 9: Final Remarks and Perspectives. This chapter summarises the thesis, providing an overview of the whole work and specifying the main achievements. It also discusses the potential future directions of event detection in social and news media.

Chapter 2

Event Detection from Text

The text has been a key source of data or information dispersal since the past. However, recent technological advancements boosted text generation, popularising various platforms, including social and news media, which currently hold the top positions. The extensive data growth that occurred afterwards, which we experience today, led to generating a high volume of important information useful for various parties all over the world. Thus, automatic event detection from the text in social and news media has gained wide research attention over the past few years.

In this chapter, we discuss the evolution of previous research in event detection from text, summarising their achievements and limitations. We initiate our discussion, defining what is an event in Section 2.1, as its idea can be vague due to language ambiguities. Then, we comprehensively describe the recent approaches proposed for social media event detection, covering diverse research areas in Section 2.2. Similarly, recent approaches proposed for news media event detection, covering popular research areas and data granularities, are described in Section 2.3. Finally, we summarise this chapter in Section 2.4, highlighting the gaps we recognised in the light of previous research, which we aim to address in this research.

2.1 Defining an Event

Previous research used different definitions for events. To mention a few, targeting social media data streams, Sayyadi et al. (2009) defined an event as some news-related thing happening at a specific place and time. Also, events were considered as occurrences which have the ability to create an observable change in a particular context (Aldhaheri and Lee, 2017). Giving more focus to event contents, Li et al. (2017a) described an event as a composition of answers to WH questions (i.e. who, what, when and where). Similarly, focusing on news media also, different definitions were introduced by previous research. For example, Allan et al. (1998) defined an event as something that happens at a particular time and place. Automatic Content Extraction (ACE) Program[1] considered an event as a specific occurrence involving participants or something that happens or a change of state.

Most of these definitions only cover the events that physically happened in some place. However, when processing social and news media data, there is a possibility to come across events which happened virtually, such as a virtual product launch and webinar. Also, by processing social and news media data, we can only recognise the events discussed or reported in that particular media. Considering these requirements and following the main ideas provided by available definitions, we generally define an event using the Definition 1 in this research.

Definition 1 *Event: An incident or activity which happened at a certain time and was discussed or reported in a data source/media.*

[1]Details of ACE are available on https://www.ldc.upenn.edu/collaborations/past-projects/ace

2.2 Event Detection in Social Media

Previous research has proposed various approaches for social media event detection, ranging from supervised to unsupervised techniques (Atefeh and Khreich, 2015; Weiler et al., 2016; Saeed et al., 2019). However, there was a more focus on real-time unsupervised techniques as they were found to be more effective for social media event detection, considering the dynamic nature of data streams and the high possibility of arising new events over time (Atefeh and Khreich, 2015). Online clustering is a commonly used unsupervised technique to capture events, including text and time. We further discuss the evolution of this area in recent research in Section 2.2.1. Also, the unsupervised approaches based on data bursts, temporal rule dynamics and community dynamics were commonly used for event detection targeting event text, time or both as described in Sections 2.2.2 - 2.2.4. A summary of reviewed approaches under these categories is available in Table 2.1. Finally, Section 2.2.5 discusses the gaps we recognised in the light of previous research, which we target to address in this research.

2.2.1 Online Clustering

Online/incremental clustering is an unsupervised process of assigning documents to clusters as they arrive (Yang et al., 1998). This process adds a newly arrived document to an existing cluster if the document is related/similar to that cluster. If there is no such relative cluster, that document will form a new cluster representing a novel event. Each cluster provides details of an event, and the document which initiated the cluster becomes the first story discussing that event, estimating its occurred time. Different vectors generated using frequency and prediction-based methods were used to represent documents during this process. Term Frequency-Inverse Document Frequency (TF-IDF)

vectors were the popularly used frequency-based representation, capturing the term occurrences in a document and their importance in the corpus (Yin et al., 2012; Sutanto and Nayak, 2018; Hasan et al., 2019; Nguyen et al., 2019). As prediction-based representations, vectors generated using word embedding models such as Word2Vec (Mikolov et al., 2013a) were commonly used (Comito et al., 2019b; Chen et al., 2019). Among these, prediction-based vectors were found to be more effective than frequency-based vectors, with their ability to capture semantics in text (Ertugrul et al., 2017). The similarity between vectors was mostly calculated using cosine similarity, considering its simplicity and appropriateness for text-based measures (Sutanto and Nayak, 2018; Hasan et al., 2019; Li et al., 2017a). Additionally, different aspects of the events and social media documents were also involved in representing documents and computing similarities to focus more on important event details. For instance, Li et al. (2017a) suggested using semantic classes (proper noun, hashtag, location, mention, common noun and verb) of incoming tweets to find the best cluster match considering their informativeness towards events. They aggregated the similarity between terms in different semantic classes to calculate similarity. Focusing more on the social aspect, Comito et al. (2019a) proposed to represent a tweet as a social object, including user, time and location details in addition to the textual content, and involve the object for similarity calculation covering different aspects.

A major issue encountered with online clustering is its increasing time and space complexity to compare documents with all historic event clusters with growing data volume in social media. As a simple solution, a lifespan was introduced to clusters so that the expired clusters could be removed or made inactive without involving in comparisons (Comito et al., 2019a; Comito et al., 2019b; Hasan et al., 2019). Also, candidate cluster generation methods were suggested to reduce the number of comparisons at a new document arrival

involving different inverted indices. Such an index helps conduct an initial comparison efficiently to filter out possible candidates to compare thoroughly. Following this idea, Li et al. (2017a) used semantic term indexing and Hasan et al. (2019) used term-tweets and term-eventIDs inverted indices. Nguyen et al. (2019) also used entity-cluster inverted indices to filter candidate clusters and additionally suggested representing clusters using a central centroid calculated only using the most similar L members of a cluster to faster the cluster update process. Overall, the involvement of semantics or entities filters the candidates effectively, focusing on event-related terms. However, the widely used rule-based approaches to extract such entities negatively impact the method's expandability.

Also, the returned clusters can hold non-events after assigning all incoming documents, which could contain personal updates, general discussions, advertisements, etc., except the newsworthy data to clusters. Targeting this issue, different statistical measures such as new tweet rate (Li et al., 2017a) and Z-score of distinct users, their location and tweet count (Comito et al., 2019a) were used to extract event clusters. More complex post-processing steps involving different aspects such as information amount, user diversity and sentence structure were also proposed to filter events more accurately (Hasan et al., 2019). Still, the requirement to process all incoming documents leaves online clustering a major limitation considering the growth rate of social media data generation. Thus, we found the following commonly used techniques: bursts in data streams, temporal rule dynamics and community dynamics, more useful for event detection as they process only the important documents or chunks of the whole data stream, being able to handle a vast data amount with low resources.

2.2.2 Data Bursts

In communication streams, bursts involve transmitting a larger amount of data than usual over a short time. We can expect bursts in data streams at events due to the high user involvement in communicating such information. Following this idea, there was a tendency to detect data bursts and only process the corresponding data to detect events. In this way, more focus could be provided on possible event data, reducing the processing and improving the performance. Van Oorschot et al. (2012) suggested using peaks in tweet volume to recognise event occurred times. Similarly, Li et al. (2014) proposed an incremental temporal topic model to capture events involving unusual tweet count changes. However, there is a possibility to occur events without notably increasing the whole data volume, and they would be missed by only considering the data at peak volumes. Overcoming this limitation, Corney et al. (2014) proposed focusing on bursts in word n-grams. They used temporal Document Frequency-Inverse Document Frequency (DF-IDF) to find bursty word n-grams at a time slot and applied hierarchical clustering on those n-grams to extract event text. However, frequency-based measures fail to differentiate events from general topics (e.g. car, food, music, etc.) because such topics are also discussed massively on social media. Also, frequency bursts appear when events become more popular or are trending. Targeting these issues, Xie et al. (2016) proposed a sketch-based topic model involving word acceleration and a tensor decomposition approach based on Singular Value Decomposition (SVD) (Anandkumar et al., 2014) to detect bursty events. According to their setting, word acceleration captures the change of arriving rate of documents, and they showed that it could identify events more accurately at their early stages.

Also, the social aspect was involved recently to recognise data bursts, considering its importance on social networks and impact on events. Guille and Favre (2015) proposed using mention anomalies in Twitter data to detect data bursts offline. They involved a word co-occurrence and temporal correlation-based approach to extract event text during bursts. Since mentions are added intentionally to connect users to discussions or dynamically during retweeting, they can capture the social aspect of data. Other research suggested an improved version of Twevent (Li et al., 2012), utilising different user diversity-based measures (i.e. user frequency, retweet count and follower count) to detect bursty text segments in a corpus (Morabia et al., 2019). They used Wikipedia as an external knowledge base to ensure the meaningfulness of segments and clustered the segments using the Jarvis-Patrick algorithm (Jarvis and Patrick, 1973) to recognise event text. Overall, incorporating the social aspect improved the performance rather than only involving text frequency-based measures (Guille and Favre, 2015; Morabia et al., 2019). However, this incorporation could limit the method's expandability because the measures which gauge social aspects are mostly specific to the social media platform. Similarly, external knowledge bases also introduce restrictions depending on their availability and language coverage.

2.2.3 Pattern Dynamics

Pattern and rule mining discover item sets in data and their relationships. Following this insight, Adedoyin-Olowe et al. (2016) proposed utilising temporal dynamics of Association Rules (ARs) in tweet hashtags to detect event occurrences/time. They suggested separating the data stream into

chunks/windows based on time and generating ARs per window using frequent patterns in hashtags to recognise different rule types based on AR variations over windows. Their study revealed that specific rule types (unexpected and emerging) greatly impact identifying event occurrences. However, Alkhamees and Fasli (2016) found it inappropriate to use a fixed support (metric for item retrieval) value for Frequent Pattern Mining (FPM) in dynamic data streams. They proposed a dynamic support calculation method based on window size and keyword occurrences which can automatically adapt to the dynamicity of social media data streams.

The recent research in this area focused more on High Utility Pattern Mining (HUPM) because it finds not only the frequent but also the high in utility item sets (Peng et al., 2018; Choi and Park, 2019). When only the frequency is considered, popularly discussed items can also be captured in addition to the event-related items, but the involvement of utility help overcome this issue. Different approaches have been proposed to calculate utility. For instance, Peng et al. (2018) used Local Weighted Linear Regression (LWLR) to calculate term novelty and combined it with term frequency to measure term utility. They used a graph-based approach to cluster detected patterns to extract event text. Choi and Park (2019) also followed a similar approach, involving the growth rate in word frequency to calculate utility. However, these HUPM-based approaches were designed only targeting the textual event details in a given corpus without focusing on temporal details, adhering to the original scope of pattern mining approaches.

2.2.4 Community Dynamics

A community in social media can be defined as a set of entities that are associated with a common element of interest (Papadopoulos et al., 2012). Thus, the

temporal dynamics of communities can be utilised to detect events. Following this idea, most previous research separated a data stream into time windows and recognised communities in each window. Then, the communities in windows were compared to capture their dynamics to detect events. Such communities provide event content, and the windows where they appear provide event times. Graph theory, clustering and topic modelling-based approaches were commonly used in previous research to identify communities.

Sayyadi et al. (2009) proposed generating keyword graphs, which represent keywords by nodes and their co-occurrence in documents by edges per time window. As keywords, noun phrases and named entities with high document frequency were considered. This approach used the betweenness centrality score to detect communities in each graph and compared their content similarities at consecutive windows to detect events. Similarly, Takaffoli et al. (2011) suggested detecting communities from document graphs in social networks and comparing communities' similarity over time windows to detect event evolution. Schinas et al. (2015) also involved documents in generating graphs per window but focused more on the community detection algorithm. They used Structural Clustering Algorithm for Networks (SCAN) (Xu et al., 2007) to extract graph communities, considering its ability to recognise bridges of clusters (hubs) to share across clusters and outliers to mark as noise. Even though graphs comprehensively capture all the intermediate relationships, they can be dense for high data volumes, increasing the computational complexity. Focusing on this issue, Edouard et al. (2017) suggested a named entity-based method to generate graphs with less density. They only used named entities in tweets extracted using NERD-ML (Erp et al., 2013) and their context terms for the graph generation. Even though focusing on specific terms speeds up the graph processing while maintaining informativeness, using tools specific to languages or domains negatively affects the approach's expandability.

Among different clustering algorithms, early research commonly used the K-means algorithm to detect events (McCreadie et al., 2013; Nur'aini et al., 2015). However, since it is not feasible to pre-define the cluster count for a dynamic data stream, algorithms that do not require specifying the count, such as hierarchical and affinity propagation, were focused more on later. Mu et al. (2018) hierarchically clustered the posts arrived during a period based on term and named entity similarity to identify communities. For community comparison across consecutive periods to recognise novel and evolved events, they used a spatio-temporal similarity of entities (WH elements–who, where and when). Other research suggested clustering critical domain-related patterns extracted from tweets using the affinity propagation algorithm to identify cybersecurity events from a data batch in a time window (Liu et al., 2020). This approach applied non-negative matrix factorisation to aggregate events over time windows. However, similar to the graph generation scenario, using language or domain-specific extractions to support clustering limits the approach's expandability.

Following the topic models' popularity, Lau et al. (2012) used Latent Dirichlet Allocation (LDA) (Blei et al., 2003) to generate communities per time window and analysed their temporal dynamics using Jensen-Shannon divergence (JSD). They also transferred parameters from the previous topic model to the next to maintain the online fashion. Chen et al. (2018) suggested a similar approach, but they used RL-LDA, a modified LDA, to capture event evolution incorporating retweeting behaviour. Deviating from temporal or platform-based modifications to topic models, some research involved advanced community comparison techniques to improve event detection. For instance, Huang et al. (2017) proposed comparing topics generated using LDA and Biterm Topic Model (BTM) in consecutive windows using novelty and fading calculated

by LWLR and Kullback-Leibler divergence (KLD). Other recent research suggested using shared topic words and temporal variations in topic word count to compare LDA topics, maintaining the simplicity (Unankard and Nadee, 2020). Overall, these modifications to the traditional topic models improved event detection performance and allowed capturing of events' temporal details. However, the requirement to pre-define topic count by topic models is a critical limitation when processing highly dynamic social media data streams.

We summarise reviewed approaches for event detection in social media under the above-mentioned categories: online clustering, data bursts, pattern dynamics and community dynamics in Table 2.1. The *detection technique* column represents the techniques used to identify events, which are further divisible from the main category, and the *general features* column represents the involved features which are not specific to any social media platform.

2.2.5 Discussion

Previous research in social media event detection targeted detecting event text, time or both, but a comparatively high focus is given to text extraction. Also, a clear majority focused on online processing-based approaches, while a few were based on offline approaches. Considering all the available combinations, online detection of event text and time provides a more comprehensive output helpful for real-time information extraction from social media, which benefits multiple parties.

To detect event time, available approaches mainly used various statistical measures (e.g. document/word frequency, retweet count, JSD, KLD, etc.) without involving the underlying text's linguistics (syntax and semantics). Syntax defines the word arrangement, and semantics describes the meanings, allowing a language to express the same idea using different word sequences

TABLE 2.1: Summary of reviewed methods for social media event detection

Category	**Detection Technique**	**General Features**	**Social Media-specific Features**	**Reference**
Online clustering		TF-IDF vectors	–	Yin et al. (2012), Hasan et al. (2019), Sutanto and Nayak (2018)
		TF-IDF vectors, named entities	–	Nguyen et al. (2019)
		N-grams, Word2Vec	Hashtags, mentions	Comito et al., 2019b
		Tokens, GloVe	–	Chen et al. (2019)
		Tokens, Word2Vec	–	Ertugrul et al. (2017)
		Named entities, nouns, verbs	Retweets, hashtags, mentions	Li et al. (2017a)
		N-grams	Hashtags, mentions, users, locations	Comito et al. (2019a)
Data bursts	Peak detection	Tweet count	–	Van Oorschot et al. (2012), Li et al. (2014)
		N-grams' DF-IDF	–	Corney et al. (2014)
	Tensor decomposition	Word acceleration	–	Xie et al. (2016)
	Anomaly detection	Word co-occurrence	Mentions	Guille and Favre (2015)
		Segments (using Wikipedia)	User frequency, retweet count, follower count	Morabia et al. (2019)
Pattern dynamics	ARM	–	Hashtags	Adedoyin-Olowe et al. (2016)
	FPM	Tokens	–	Alkhamees and Fasli (2016)
	HUPM	Term frequency, term novelty	–	Peng et al. (2018)

TABLE 2.1: Continued

Category	Detection Technique	General Features	Social Media-specific Features	Reference
Pattern dynamics	HUPM	Growth rate in word frequency	–	Choi and Park (2019)
Community dynamics	Graph community detection	Noun phrases, named entities	–	Sayyadi et al. (2009)
		–	Social network	Takaffoli et al. (2011)
		TF-IDF vectors, named entities	–	Schinas et al. (2015)
		Named entities, entity contexts	–	Edouard et al. (2017)
	Clustering	Document vectors	–	McCreadie et al. (2013)
		TF-IDF vectors	–	Nur'aini et al. (2015)
		BoW, named entities	–	Mu et al. (2018)
		Frequent text phrases, named entities	–	Liu et al. (2020)
	Topic modelling	BoW	–	Lau et al. (2012), Unankard and Nadee (2020)
		BoW	Retweets, hashtags, locations	Chen et al. (2018)
		BoW, term's novelty/fading	–	Huang et al. (2017)

and vice versa. Thus, a notable amount of important information can be lost, ignoring the linguistics, especially in the social media text, considering the diversity of users with different writing patterns. However, to extract event text, some approaches involved linguistics in addition to statistics, considering its

importance. These approaches mainly used external knowledge bases, rule-based methods and prediction-based methods to capture linguistics. Overall, using external knowledge bases introduces restrictions depending on their coverage and availability. Similarly, rules also negatively affect the approach's expandability due to their dependencies on languages and domains. As the prediction-based methods, pre-trained word embeddings are commonly used, but they also add constraints depending on the model availability and incapabilities to capture corpus-specific linguistics in social media.

Considering these limitations, this research targets proposing novel methods capable of online detection of event time and text from social media data streams, involving both statistics and linguistics, which are essential for effective information extraction from textual data (Chapters 4 and 5). We mainly focus on involving linguistics while covering corpus specificities and preserving the expandability of the approach.

2.3 Event Detection in News Media

Early research focused more on unsupervised approaches to detect events from news media, following the Topic Detection and Tracking (TDT) initiative (Allan et al., 1998; Dai et al., 2010). However, unlike the scenario with social media event detection, most recent research used supervised approaches for news media event detection (Basile and Caselli, 2020; Hürriyetoğlu et al., 2021a). The less dynamicity in news media compared to social media, which allows pre-defining event categories, can be mentioned as the main reason for this tendency. Also, supervised techniques can produce more accurate and detailed results than unsupervised techniques. The published labelled data corpora also further encouraged this trend (Doddington et al., 2004; Hürriyetoğlu et al., 2021b). Considering the length of news articles and their information

coverage, different data levels: document, sentence and token were targeted to extract events from news media. We further discuss the evolution of supervised techniques in recent research over these levels in Sections 2.3.1 - 2.3.3. Since we recognised some influences from sentence to other level approaches, we initiated our review with the sentence level below. A summary of reviewed approaches corresponding to each of these data levels is available in Table 2.2. Finally, we discuss the gaps we recognised following our review in Section 2.3.4, which we aim to address in this research.

2.3.1 Sentence Level

Previous research has proposed various classification approaches to identify event-described sentences, ranging from traditional machine learning (ML) to deep learning (DL). Recently, more focus has been given to DL-based methods, especially transformer-based models considering their effectiveness. We discuss more details about the previous approaches in Sections 2.3.1.1 - 2.3.1.3.

2.3.1.1 Traditional Machine Learning

Early research widely used feature-based approaches with traditional classification algorithms to identify event sentences. For example, Naughton et al. (2010) used a Support Vector Machine (SVM) model trained using a set of features, including stemmed terms, part of speech (POS) tags, noun chunks, sentence length, sentence position and presence/absence of negative terms. Another research also utilised the SVM model with Bag of Word (BoW) feature representations with token n-grams, character n-grams, lemma and POS tags (Lefever and Hoste, 2016). Additionally, this approach included named entities and special indicators such as numerals, symbols and time as model

features. Similarly, Basile and Caselli (2020) proposed using the Logistic Regression algorithm to classify event sentences using informative character n-gram and token unigram features. Among these commonly used features, BoW is vulnerable to a critical information loss, as it ignores the word semantics and order, which are crucial for text understanding (Hassan and Mahmood, 2017). This issue can be mitigated to a certain extent using n-grams as they capture local word orders, but n-grams suffer from data sparsity issues (Hassan and Mahmood, 2018). Overall, the involvement of language-based features (e.g. POS tags) helped capture important text units but negatively affected the method's expandability to different languages. Considering these limitations and the effectiveness of text embedding models (e.g. Word2Vec (Mikolov et al., 2013a)) along with the recent advances, DL-based approaches became more famous for sentence classification tasks in later research.

2.3.1.2 Deep Learning

Among different neural networks, previous research popularly used Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Convolutional Neural Network (CNN) (Lawrence et al., 1997) models for text classification tasks. LSTMs are more capable of learning long-term dependencies using their memory cells than vanilla Recurrent Neural Networks (RNN) (Hassan and Mahmood, 2018). CNNs are capable of capturing local text features such as syntax and semantics of words within a sentence using their multiple convolutional and pooling layers (Chen et al., 2015). Word embeddings were mostly used to input text into these networks.

Hassan and Mahmood (2017) used pre-trained Word2Vec (Mikolov et al., 2013a) embeddings with an LSTM model to classify sentences. In a later study,

they built a joint CNN and LSTM model, combining the characteristics (Hassan and Mahmood, 2018). More modified networks, such as Convolutional RNN (CRNN), which stacks a convolutional layer on top of an RNN and CNN with Attention (CNNA) which has an attention layer on top of a CNN, were also proposed by previous work (Huynh et al., 2016). Similar to CNNA, the LSTM model was also modified by adding an attention layer on top to allow the model to focus more on event-related words (Liu et al., 2019a). Overall, these architectural changes improved the event sentence classification by utilising the capabilities of different mechanisms. However, all these models require a sufficiently large amount of labelled data to effectively fine-tune model weights from scratch, making them impotent for low-resource scenarios. Also, the classic word embeddings fail to capture contextual details in the text, which are important to understanding sentences. Targeting these limitations, transformer-based architectures gained great attention in recent research.

2.3.1.3 Transformers

Transformers were originally designed with the ability to fine-tune for a downstream task by transferring the pre-trained knowledge (Devlin et al., 2019). This knowledge transfer allows the model to learn a downstream task effectively even with comparatively fewer training instances, overcoming a major limitation in deep neural networks. Also, the attention-based transformer encoder preserves contextual details in the text while generating representations, unlike the classic embedding models. These characteristics allowed transformers to improve the performance of many NLP applications with state-of-the-art results (Devlin et al., 2019).

Following this trend, transformers were also involved in event sentence

identification. The common approach to adapting a transformer for text classification is adding a simple linear layer on top and fine-tuning for the targeted task. Different pre-trained transformers were used with this architecture. For instance, Gürel and Emin (2021) used pre-trained monolingual and multilingual BERT (Devlin et al., 2019) models. Similarly, Hu and Stoehr (2021) used RoBERTa (Liu et al., 2019b) English model. Additionally, they suggested translating text from other languages to English to make predictions rather than using a multilingual model. XLM-RoBERTa (XLM-R) (Conneau et al., 2020) is also a commonly used transformer, especially for multilingual predictions (Awasthy et al., 2021; Re et al., 2021). Unlike other multilingual transformers, XLM-R generates cross-lingual embeddings, which attempt to ensure words with the same meaning in different languages map to almost the same vector. Thus, it showed improved results than translation-based approaches, which could suffer from language errors, and other transformers. Deviating from the common architecture, Kalyan et al. (2021) suggested adding an LSTM layer on top of the transformer and getting soft voting of models built using the pre-trained transformers: BERT, RoBERTa and DistilBERT (Sanh et al., 2019) as the final prediction. Also, another recent research experimented with the weighted ensemble of RoBERTa model and LexStem: a two-channel CNN with normal and stemmed text (Çelik et al., 2021). However, overall, these modifications did not outperform the simple architecture with a linear output layer on top of a large pre-trained transformer, which can be considered state-of-the-art for event sentence identification (Hürriyetoğlu et al., 2021a).

2.3.2 Document Level

Similar to sentence level extractions, diverse classification approaches, ranging from traditional ML to DL have proposed by previous research to identify

event-contained documents (Sections 2.3.2.1 and 2.3.2.2). Also, recent research mostly focused on DL, considering its effectiveness over traditional ML in text classification. However, transformer-based models were not popularly used for document level predictions, and we discuss the possible reasons recognised in Section 2.3.4.

2.3.2.1 Traditional Machine Learning

Traditional methods commonly represented documents using sparse vectors such as BoW and TF-IDF, and used SVM for classification (Kumar et al., 2012; Dadgar et al., 2016). The larger the document set, the BoW vector can reach thousands of features, negatively affecting the classifier's accuracy and increasing computational cost. Focusing on this issue, Le Nguyen and Ho Bao (2015) proposed a frequency and cluster-based approach to select important features from BoW. This approach marked a term important if it largely appears in a category and creates the separation of that category from others. Inability to capture synonymy and polysemy is another issue encountered with BoW. García et al. (2016) suggested incorporating Bag of Concepts (BoC) with BoW to capture these details. Concepts are units of meaning which can capture synonymy and polysemy. This research used Wikipedia knowledge to extract the concepts in documents effectively.

Also, BoW fails to capture semantics in the text. Targeting this limitation, Jing et al. (2013) proposed an updated version of Naïve Bayes classifier (NBC) with document level semantic information. They used Log-Bilinear Document Modelling (LBDM) (Maas et al., 2011) to extract semantics. Modifications to features were also made using word embeddings to capture semantics. For instance, Nugent et al. (2017) proposed averaging vectors of words in an article to obtain a document vector to use with classification algorithms such as

SVM and random forest. They experimented with different word embedding models such as Word2Vec (Mikolov et al., 2013a) and fastText (Bojanowski et al., 2017) to generate word vectors. Furthermore, algorithms particularly designed to generate document vectors, such as Doc2Vec (Le and Mikolov, 2014), were used to capture semantics more effectively (Lindén et al., 2018). Overall, traditional approaches evolved to capture lexical and semantical features in the text but fail to capture long-distant relationships in documents well. Thus, there was a high tendency to use DL approaches in recent research, considering their ability to mitigate this issue and overall effectiveness.

2.3.2.2 Deep Learning

Similar to sentence level approaches, previous research widely used LSTM (Hochreiter and Schmidhuber, 1997) and CNN (Lawrence et al., 1997) models for event document classification with various modifications appropriate for document processing. For instance, Lindén et al. (2018) suggested separating documents into sentences using LSTM's time parameter and vectorising them using Doc2Vec model (Le and Mikolov, 2014) to support the model to capture document structure. Irsan and Khodra (2019) proposed generating document representations by multiplying BoW vectors and averaged Word2Vec (Mikolov et al., 2013a) embeddings, combining lexical and semantical features, to use with a CNN model. In addition to changing input representations, some approaches targeted architectural modifications. Parida et al. (2021) designed a multichannel network, combining several CNN models which process various n-grams.

Hierarchical models were also commonly proposed targeting the structure in documents (words form sentences and sentences form the document). For

example, Hierarchical Attention Network (HAN), which captures the document's hierarchy by building representations from word level to document level, can be mentioned (Yang et al., 2016). It also captures the importance of words and sentences depending on the context using word and sentence level attention mechanisms. Furthermore, Mehta et al. (2019) incorporated a multi-aspect attention mechanism with HAN, allowing the model to capture multiple aspects in sentences related to an event. All these approaches used Word2Vec embeddings to vectorise text to use with their input layers. Additionally, graph-based approaches were also proposed to capture long-distance relationships in documents. For instance, Wang et al. (2020) proposed a hierarchical topic graph, integrating a probabilistic deep topic model to a Graph Convolutional Network (GCN), allowing to capture semantic relationships at different levels. Despite the various improvements that happened along with DL-based approaches, being deep networks, they require a vast amount of data to effectively build a model from scratch. Thus, these approaches fail to process languages or domains where labelled data is scarce, leaving a major limitation.

Transformers were not commonly involved in document level predictions, mainly considering their inability to process long sequences due to the complexities in full-attention operation used by these models. We further discuss the evolution of transformer-based architectures for long-sequence processing in Section 2.3.4.

2.3.3 Token Level

The main target of token level event detail extraction is identifying text spans representing event triggers and arguments. Event triggers are the words which express the event occurrences, and arguments are the entities which describe

the event attributes. Event trigger and argument extraction is commonly considered as a token classification or sequence labelling problem by previous research. Similar to sentence and document level tasks, various approaches based on traditional ML and DL have been used for this extraction (Sections 2.3.3.1 and 2.3.3.2). According to recent research, there is also a trend to involve transformers (Section 2.3.3.3).

2.3.3.1 Traditional Machine Learning

Most early works built classification models using linguistic features to extract event details at the token level. For example, Chen and Ng (2012) built separate classification models for trigger and argument extraction using the SVM algorithm. They used a wide range of linguistic features, including tokens, POS tags, dependency paths and synonyms from semantic dictionaries, and treated each word as a separate instance, to build the models. Hong et al. (2011) also used SVM classifiers to make final predictions but proposed using cross-entity inference to mitigate the possibility of missing events by only using the local features. They also used information from the Web to understand the background of entities, in addition to the knowledge in the training corpus, to improve the effectiveness of their approach. Rather than treating trigger and argument extraction as separate tasks, another research proposed a joint system based on structured perceptron with beam search, allowing to improve the predictions of each task mutually (Li et al., 2013). This approach is also highly based on linguistic features such as POS tags, lemmas, synonyms and dependencies. Overall, due to the complexities in token level event extraction, traditional approaches extensively relied on language-based features or knowledge bases, resulting in less generalisability across different languages. Thus, similar to the trends with sentence and document level detection, more

focus was given to DL-based approaches afterwards, mainly considering their ability to extract underlying features in text automatically.

2.3.3.2 Deep Learning

For event token extraction also, LSTM (Hochreiter and Schmidhuber, 1997) and CNN (Lawrence et al., 1997) are the commonly used neural network architectures by previous research. However, Bidirectional LSTM (Bi-LSTM) models were used over LSTM because both past and future states of the sequence are important for sequence labelling. Also, Conditional Random Fields (CRFs) were used for output generation since they take context into account, replacing simple linear layers. For example, Pandey et al. (2017) used a Bi-LSTM network with a CRF layer to extract event entities. They used Word2Vec (Mikolov et al., 2013a) and GloVe (Pennington et al., 2014) embeddings to feed text into the network. Another research used the same architecture with fastText (Bojanowski et al., 2017) and Multilingual Unsupervised and Supervised Embeddings (MUSE) (Lample et al., 2018) to extract event triggers (M'hamdi et al., 2019). Also, more advanced embeddings such as ELMo (Peters et al., 2018), character and POS were used with this architecture, targeting capturing contextual and syntactical details in the text to improve the accuracy (Basile and Caselli, 2020). To mention a few CNN-based approaches, Chen et al. (2015) involved separate Dynamic Multi-pooling CNNs (DMCNNs) with Word2Vec embeddings to extract triggers and arguments. Lu et al. (2022) proposed a path-aware GCN (PGCN) with BERT (Devlin et al., 2019) embeddings.

Joint neural network models were also proposed for trigger and argument extraction, considering the benefits of mutual learning and error propagation in pipelined methods. Nguyen et al. (2016) built a Bi-LSTM model with Word2Vec embeddings for joint trigger and argument extraction. Sha et al.

(2018) proposed adding dependency bridges over Bi-LSTM to utilise dependency relations with joint learning. A combination of Bi-LSTM and DMCNN was also suggested, combining their characteristics (Balali et al., 2020). This network used an advanced embedding layer formed by concatenating BERT, GloVe, entity type, POS and dependency relation embeddings, providing a wide range of features. Overall, these improved network architectures help capture hidden structures in text effectively, easing the requirement to hand pick an extensive amount of features from the text. Still, the involvement of language-based features such as POS by some approaches negatively affects their expandability. Also, these networks require a large amount of data for the from-scratch learning process limiting their usability only for high-resource scenarios. Thus, similar to the evolution of sentence level approaches, there is a recent trend to use transformers, considering their effectiveness along with language transferability for token level event extraction.

2.3.3.3 Transformers

Like with sentence level detection, transformers have been used for token level event extraction recently. Following the trends in DL-based approaches, M'hamdi et al. (2019) designed a network with a CRF layer on a transformer model to extract event triggers. They picked monolingual and multilingual BERT (Devlin et al., 2019) models as the transformer and analysed their performance in different languages. Following the simple approach, Yang et al. (2019a) added linear layers on the BERT model per token/word to extract triggers. They used a separate BERT-based model for argument extraction and occupied its input with the identified triggers following a pipelined approach. However, a comparatively high focus was given to building joint models for trigger and argument extraction using transformers, considering the resource

requirements, tasks' interconnections helpful for mutual learning and error-propagation in pipelined approaches. For instance, Çelik et al. (2021) proposed a joint model by adding a Bi-LSTM and CRF layer on a RoBERTa (Liu et al., 2019b) model. The XLM-R (Conneau et al., 2020) model was also used with linear output layers per token, targeting multilingual predictions, following the same trend noticed with sentence level predictions (Awasthy et al., 2021; Vivek Kalyan et al., 2021). Overall, this simple architecture with linear output layers outperformed other modifications, being the state-of-the-art for event trigger and argument extraction (Hürriyetoğlu et al., 2021a).

We summarise reviewed approaches for event detection in news media under the above-mentioned data levels: sentence, document and token in Table 2.2. The approaches are categorised into traditional machine learning, deep learning and transformers, as mentioned in the *category* column. The *model* and *features* columns represent the involved algorithms or architectures and features used with them, respectively.

2.3.4 Discussion

According to previous research, transformer-based models have state-of-the-art results for sentence and token level event detection from news media, outperforming traditional ML- and DL-based approaches. However, transformers were not commonly used for document level predictions, mainly due to the inability to process long sequences with above 512 sub-tokens (Devlin et al., 2019). Two major approaches: (1) hierarchical design and (2) sparse attention mechanisms, were considered to overcome this limitation recently. The hierarchical design splits the documents into chunks that fit the transformer's input sequence length (512), passes the chunks through a transformer, and generates document representations by processing the chunk representations through

TABLE 2.2: Summary of reviewed methods for news media event detection

Data Level	**Category**	**Model**	**Features**	**Reference**
Sentence	Traditional machine learning (ML)	SVM	Stemmed terms, lexical information, noun chunks, sentence features (length, position, etc.)	Naughton et al. (2010)
		SVM	Token/character n-grams, lemma, POS, numerals, symbols, named entities	Lefever and Hoste (2016)
		Logistic Regression	Character n-gram, tokens	Basile and Caselli (2020)
	Deep learning (DL)	LSTM	Tokens, Word2Vec	Hassan and Mahmood (2017)
		CNN-LSTM	Tokens, Word2Vec	Hassan and Mahmood (2018)
		CRNN, CNNA	Tokens	Huynh et al. (2016)
		LSTM-Attention	Tokens, Word2Vec	Liu et al. (2019a)
	Transformers	BERT	Tokens	Gürel and Emin (2021)
		RoBERTa	Tokens	Hu and Stoehr (2021)
		XLM-R	Tokens	Awasthy et al. (2021), Re et al. (2021)
		Ensembled BERT-, RoBERTa- and DistilBERT-LSTM	Tokens	Kalyan et al. (2021)
		Ensembled RoBERTa and LexStem	Tokens, stems	Çelik et al. (2021)
Document	Traditional ML	SVM	TF-IDF vectors	Kumar et al. (2012), Dadgar et al. (2016)

TABLE 2.2: Continued

Data Level	Category	Model	Features	Reference
Document	Traditional ML	SVM	Filtered BoW	Le Nguyen and Ho Bao (2015)
		Semantic NBC	Tokens, LBDM	Jing et al. (2013)
		SVM, Random Forest	BoC (using Wikipedia)	García et al. (2016)
		SVM, Random Forest	Tokens, Word2Vec, fastText	Nugent et al. (2017)
		Decision Tree, Random Forest	Documents, Doc2Vec	Lindén et al. (2018)
	DL	LSTM	Sentences, Doc2Vec	Lindén et al. (2018)
		CNN	BoW, tokens, Word2Vec	Irsan and Khodra (2019)
		Multichannel CNN	N-grams	Parida et al. (2021)
		HAN	Tokens, Word2Vec	Yang et al. (2016)
		Multi-aspect HAN	Tokens, Word2Vec	Mehta et al. (2019)
		GCN	Tokens	Wang et al. (2020)
Token	Traditional ML	SVM	Lexical features, syntactic features, semantic dictionaries, nearest entity information	Chen and Ng (2012)
		SVM	Cross-entity details	Hong et al. (2011)
		Structured perceptron with beam search	Lexical features, syntactic features, entity information	Li et al. (2013)
	DL	Bi-LSTM-CRF	Tokens, Word2Vec, GloVe	Pandey et al. (2017)
		Bi-LSTM-CRF	Tokens, fastText, MUSE	M'hamdi et al. (2019)

TABLE 2.2: Continued

Data Level	Category	Model	Features	Reference
Token	DL	Bi-LSTM-CRF	ELMo, POS, character embeddings	Basile and Caselli (2020)
		DMCNN	Tokens, Word2Vec	Chen et al. (2015)
		PGCN	Tokens, BERT	Lu et al. (2022)
		Bi-LSTM	Tokens, Word2Vec	Nguyen et al. (2016)
		Dependency-bridge Bi-LSTM	Tokens, Word2Vec	Sha et al. (2018)
		Bi-LSTM-DMCNN	Tokens, BERT, GloVe, entity types, POS, dependency relations	Balali et al. (2020)
	Transformers	BERT-CRF	Tokens	M'hamdi et al. (2019)
		BERT	Tokens, triggers	Yang et al. (2019a)
		RoBERTa-Bi-LSTM-CRF	Tokens	Çelik et al. (2021)
		XLM-R	Tokens	Awasthy et al. (2021), Vivek Kalyan et al. (2021)

an RNN layer to use with a classifier (Pappagari et al., 2019). However, this approach restricts the transformer's ability to capture long-range dependencies in documents by processing the entire sequence. With the sparse attention mechanisms, modified transformers which can process up to 4,096 sub-tokens, were proposed, opening a new avenue while mitigating the critical issues encountered with hierarchical design (Beltagy et al., 2020; Zaheer et al., 2020). However, as far as we know, these long-sequence transformers were not considered for event document identification by previous research. Thus, we target experimenting with their applicability and performance in this research

(Chapter 7).

Targeting sentence and token level event detection, most available transformer-based approaches treated these tasks separately without considering their interconnections. However, Basile and Caselli (2020) proposed a bottom-up approach from token to sentence level, being an exception. They initially extracted event triggers and arguments using a BERT sequence labelling model with linear output layers and then recognised a sentence as an event sentence if it contains a trigger. This approach mainly suffers from error propagation. Also, it only relies on token level labels which are scarce due to labelling complexities and does not account for the possibility of using sentence level knowledge for token level predictions. Targeting these gaps, this research aims to propose a novel learning strategy with transformers, which can learn from sentence to token level and vice versa, utilising knowledge from one level to support the predictions at the other level (Chapter 8).

Considering the language coverage of available methods, early research commonly proposed monolingual approaches, focusing on English mostly and a few other languages (e.g. Chinese (Chen and Ng, 2012), Dutch (Lefever and Hoste, 2016), German (Parida et al., 2021), etc.). With the involvement of transformers for sentence and token level detections, multilingual models have recently been used to support different languages. However, to the best of our knowledge, no comprehensive study that analyses the cross-lingual performance of different transformer models involving different learning strategies targeting sentence and token level event extraction is available in the literature. Also, no such study is available for document level, as transformers have not been popularly involved in previous research. Filling these gaps, we target conducting a thorough analysis in this research using the commonly used learning strategies and the ones we aim to propose (Chapters 7 and 8).

2.4 Summary

In previous research, diverse approaches have been proposed for automatic event detection from social and news media, considering its importance with increasing data volumes in both media with valuable information. Figure 2.1 illustrates an overview of the literature review conducted by this research. Due to data streams' dynamic and unpredictable nature, previous research recognised unsupervised approaches as more appropriate for social media event detection. However, supervised approaches were widely used for news media event detection following the less dynamicity and predictable nature of news documents than social media posts. Also, news documents require different levels of information extraction, as they are lengthier and more detailed than social media posts.

Analysing the previous work in social media event detection, we recognised that a clear majority of approaches only focused on statistical variations over data streams to identify events, ignoring the underlying linguistics (syntax and semantics). Since linguistics plays a crucial role in natural language, without its involvement, critical information loss can happen. We aim to propose novel approaches to overcome this limitation in this research while preserving the qualities: real-time efficiency, expandability and scalability which are important for social media event detection. The proposed approaches are described in detail in Part I of this thesis.

According to the previous news media event detection research, three data granularities: document, sentence and token, were commonly considered to extract event details comprehensively. Deep learning-based approaches were used in recent research for document level extractions. However, they fail to effectively fine-tune the deep networks from scratch when training data are scarce. We target involving transformer models for this task for the first time

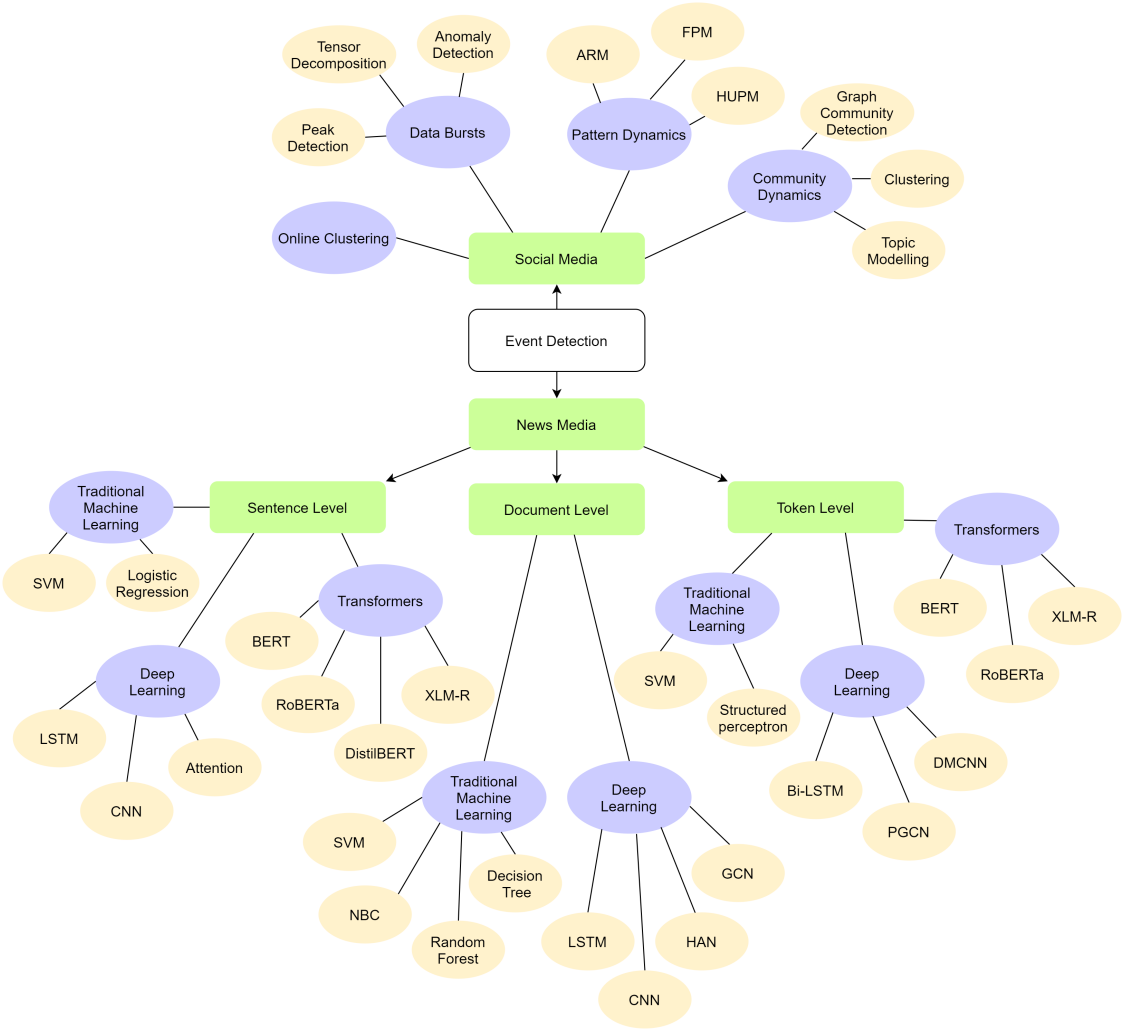


FIGURE 2.1: Overview of the literature review

to the best of our knowledge, considering their knowledge transferability helpful to handle fewer data situations and ability to capture contextual details and long-range dependencies in the text. Transformers have already been used for the sentence and token level tasks, setting the state-of-the-art results. However, the possibility of using interconnections between sentences and tokens for mutual improvements is not considered previously, and we aim to analyse it in this research. Additionally, we also target investigating the cross-lingual abilities of the proposed architectures. Our approaches are described in detail in Part II of this thesis.

Part I

Social Media Event Detection

Chapter 3

Introduction to Social Media Event Detection

Initiating Part I of the thesis, this chapter introduces social media event detection and provides an overview of this part. It mainly describes the important aspects to consider to effectively detect events from social media data streams, following the previous work discussed in Chapter 2 and the characteristics of social media data. Furthermore, this chapter defines the targeted problem, details the resources and concepts utilised for event detection by this part of the research and provides an overview of the remaining chapters (Chapters 4 and 5) of this part.

Social media services generate a vast amount of data which consists of diverse information as described in Chapter 1. However, due to the high volume and dynamicity of data, it is impractical to analyse them manually to extract important or newsworthy contents. Thus, the requirement of intelligent automated mechanisms for event detection from social media data becomes crucial (Small and Medsker, 2014). Addressing this requirement, various approaches have been proposed by previous research for event detection involving different techniques, which are discussed in Chapter 2 (Section 2.2). These approaches can be mainly divided into three types based on the targeted event output: text, text and time, and time. The text-targeted systems are designed

to identify all topics/events in a given corpus, similar to topic modelling (Li et al., 2012; Morabia et al., 2019). The text and time-targeted systems capture both event text and time by processing a data stream either offline (as a whole) (Guille and Favre, 2015) or online (as documents arrive) (Comito et al., 2019a). The time-targeted systems are designed as notification systems to notify event occurred times (Adedoyin-Olowe et al., 2016). Among these variants, we recognised that online text and time-targeted systems are more useful in event detection, considering their informative outputs.

Analysing the techniques used by available systems, we noticed that apart from a few notable exceptions, most rely only on data statistics without considering linguistics, specifically semantics. Since semantics describe the connections between words and their meanings, severe information loss can happen without considering them. For example, the following tweets:

> *'There are 13 million people living in poverty in the UK. 13M!!! Yet some MPs will vote for the deal with NO impact assessments. That 13M could become 20M?!#VoteTheDealDown #PeoplesVoteMarch #PeoplesVote #StopBrexit'*

> *'Luciana Berger - Steve Barclay confirmed that no economic analysis of the #BrexitDeal has been done... let that sink in. So how can we be expected to vote on a deal, that will affect this country for decades, today? #VoteDownTheDeal #PeoplesVote'*

which were posted during Brexit Super Saturday 2019, express the same idea but without common keywords except for a few hashtags. The main subject is also written using two different phases (*'impact assessments'* and *'economic analysis'*). In such cases common to social media, due to user diversities, we cannot understand the relationships between terms to extract

available information without semantics. Further analysing the few methods which involved semantics, we recognised that they also use semantics to extract only the event text. Commonly, these methods used rule-based and prediction-based approaches to capture semantics. Among them, rule-based approaches (e.g. semantic class extraction (Li et al., 2017a), named entity extraction (Nguyen et al., 2019)) mainly targeted filtering important terms without focusing on their relationships. Also, they are less expandable due to language dependencies. The prediction-based approaches mostly used word embeddings considering their ability to capture linguistical relationships between words (Ertugrul et al., 2017; Comito et al., 2019b). However, to the best of our knowledge, all available word embedding-based approaches use pre-trained models incapable of capturing linguistics specific to the underlying corpus, such as modified or misspelt words. Furthermore, using pre-trained models limits the expandability depending on the model availability.

Considering the crucial requirement to incorporate semantics and the limitations in available methods, we focus on developing effective methods for social media event detection in this research. Rather than focusing on complete online processing, we decided to choose an intermediate level using the concept of time windows, which processes a set of documents that arrived during a period at once. This concept supports handling large data volumes effectively while allowing users to customise the system depending on the targeted domain's characteristics, such as evolution rate and personal preferences, which include the intended update rate. We aim to extract event time and text, defining two data granularities useful for different user groups. At the coarse-grained level, we target identifying the event occurred time windows as described in Chapter 4. Such a system notifies users about event occurrences so that they can further analyse data manually to extract the necessary information, being helpful in situations where sensitive events are

targeted and cannot rely on fully automated processes. At the fine-grained level, we target identifying event occurred time windows together with the co-occurred events in those windows as described in Chapter 5. This system automates the whole process and will be helpful for users who intend to get event updates concisely and quickly without involving any manual process. Specifically, in our approaches, we involve linguistics in underlying text, captured using self-learned word embeddings and their hierarchical relationships in dendrograms to overcome the less semantic involvement in previous research (Section 2.2). Additionally, we annotate a social media dataset covering two diverse domains and design a comprehensive set of evaluation metrics to assess automated event detection mechanisms as a part of our research.

The main contributions of this part of the thesis are as follows.

1. We propose a novel method named *Embed2Detect* which identifies temporal event occurrences of social media data streams in (near) real-time to notify users about events, involving linguistical features in the underlying text to overcome a major limitation in available approaches, that lacked semantic involvement.

2. We propose a novel method named *WhatsUp* which detects both temporal and fine-grained textual event details from social media data streams in (near) real-time to automate the complete flow of event detection, considering statistics and linguistics of underlying data in an unsupervised manner.

3. We create and publish social media datasets representing two diverse domains (i.e. sports and politics) with ground truth event labels, addressing

the lack of recent data availability in the area of social media event detection[1].

4. We design a comprehensive set of metrics/framework that evaluates both temporal and textual details of detected events and release its implementation to support related evaluations in a unified way[2].

5. We release our method implementations as open-source projects to support applications and research in the area of social media event detection[3].

The rest of this chapter is organised as follows. Section 3.1 defines the problem targeted by this research. Section 3.2 describes the limitations in available datasets and details of datasets we prepared for evaluations, including data collection, data cleaning and ground truth labelling. Section 3.3 introduces the metrics we designed to evaluate temporal and textual event details. Section 3.4 summarises the background concepts we used, such as word embeddings and dendrograms. Section 3.5 introduces all the notations we used under social media event detection. Finally, Section 3.6 provides a summary of this chapter introducing the following chapters in Part I of the thesis.

3.1 Problem Definition

The problem targeted by this part of the research is automatically detecting events in (near) real-time from social media data streams. The concept behind a data stream is introduced with Definition 2.

[1]Social media event datasets are available on https://github.com/hhansi/twitter-event-data-2019

[2]Event evaluator implementation is available on https://github.com/HHansi/WhatsUp/tree/master/experiments/twitter_event_data_2019/evaluation

[3]Links to the GitHub repositories are provided in bellow Chapters 4 and 5

Definition 2 *Social Media Data Stream:* A continuous and chronological series of posts or documents D: $d_1, d_2, ...d_i, d_{i+1}, ...$ generated by social media users.

Available event detection approaches mainly consider two input data stream variants as general and filtered (or focused). In the general scenario, the whole data stream D is processed (McCreadie et al., 2013; Nguyen et al., 2019), and in the filtered scenario, a user-centred data stream D' extracted from the whole data stream D is processed. The filtering was commonly done based on keywords or locations. Keyword-based filtering extracts a domain-specific data stream using a set of keywords (Aiello et al., 2013; Comito et al., 2019b), and location-based filtering extracts a data stream composed of a set of documents posted by users in particular locations (Li et al., 2012; Guille and Favre, 2015). Among these two filtering techniques, the location-based method adds unnatural restrictions due to the unavailability of locations in all posts or user accounts and the possibility of reporting an event at a particular location by a user located elsewhere (e.g. report while travelling or watching television).

Following the above-mentioned facts, the attributes of a whole data stream and user requirements, we decided to focus on identifying events in keyword-based filtered data streams (Definition 3) in our research. Due to the popularity of social media, the whole data stream is massive and consists of lots of non-newsworthy contents in addition to the newsworthy contents. If we consider the real scenario, people or domain experts need the quick extraction of information in an interesting domain rather than extracting all the information available (Aiello et al., 2013). For example, football fans would like to know football updates, fire brigades would like to know fire updates, and BBC politics news crew would like to know political updates. Thus, processing a filtered data stream fulfils the real requirement while being resource-efficient. In this setup, we need to narrow down the whole data stream initially using

some keywords (seed terms) specific to the targeted domain. In the case of social media, we can consider the commonly used tags in the area of interest as keywords. Mostly, such tags are known to the domain experts or can be extracted using applications that identify the trending tags in social media[4].

Definition 3 *Filtered Data Stream:* A filtered or narrowed-down data stream consists of posts that contain at least one of the selected keywords.

We mainly divide social media event detection into two parts based on data granularity to address different user requirements. At the coarse-grained level, we aim to notify event occurrences in time, targeting the users who look for sensitive events and prefer to manually analyse the data stream to obtain more details. To facilitate event time extraction, we use the concept of time windows, which is widely used in previous research (Aiello et al., 2013; Adedoyin-Olowe et al., 2016; Morabia et al., 2019). Also, using time windows allows users to customise the system depending on the targeted domain and intended update rate. Briefly, upon the data arrival via a filtered stream D', the targeted system needs to separate data into time windows W: $W_1, W_2, ...W_{t-1}, W_t, ...$ (durations of time) of user-defined length l and assess each window W_t to identify event occurred time windows W^d (Definition 4).

Definition 4 *Event Occurred Time Window/Event Window:* Duration of time W_t, where at least one event has occurred.

At the fine-grained level, we aim to identify textual event details within event windows targeting the users who prefer to follow a fully automated process to get event details quickly. Since multiple events can happen during an event window W_t^d, we introduce the concept of co-occurred events with Definition 5. Per event e that happened during W_t^d, we focus on extracting a word/token cluster c, which expresses the event's textual details.

[4]Popular hashtags under different domains can be found at http://best-hashtags.com/

Definition 5 *Co-occurred Events:* Multiple events $E_{W_t^d}$ happened/reported within the same time window (event window W_t^d).

In summary, we aim to develop systems to identify coarse-grained (event windows W^d) and fine-grained (token clusters that represent the co-occurred events within event windows $E_{W_t^d} : W_t^d \in W^d$) event details by processing an incoming filtered social media data stream D' in (near) real-time in this part of the research.

3.2 Datasets

To conduct the experiments and evaluations, we used data collected from Twitter. Twitter is considered because it is widely used as an information network than social media (Adedoyin-Olowe et al., 2016; Kwak et al., 2010), and has limited restrictions to access data with enough coverage for this research.

To the best of our knowledge, the most recent Twitter event datasets were released based on data in 2012 (Aiello et al., 2013; McMinn et al., 2013). The dataset released by McMinn et al. (2013) (*Events2012*) used the Twitter stream from October 10, 2012 to November 7, 2012. Aiello et al. (2013) extracted tweets corresponding to three major events in 2012 from sports and politics to prepare the dataset. Also, these datasets have only released the tweet IDs, following Twitter restrictions. Therefore, we needed to download the actual tweets from Twitter Application Programming Interfaces (APIs) using the IDs. Downloading the tweets in the Events2012 corpus, we could only retrieve 65.8% of the tweets, as the rest have been deleted by the users or those user accounts have been deactivated. Similarly, a large proportion of data from other corpora was also unavailable to download. For example, only 63.4% of the sports dataset (Aiello et al., 2013) could be downloaded. In addition to the issue of missing

a large proportion, a few more issues were encountered with the Events2012 dataset. Since it was originally designed to identify event clusters in a corpus, only the event descriptions were provided as ground truth (GT) without temporal details. Thus, we needed to separate data into 1-day time windows to assign time details, following a commonly used strategy (Alkhamees and Fasli, 2016; Morabia et al., 2019), but a 1-day window is too lengthy for quick updates targeted by this research. Also, after the window separation, we found that all the time windows have events due to the usage of multi-domain data with diverse events. This nature could negatively affect the event time evaluations because there exist no non-event windows to check the method's ability to ignore such windows while detecting event windows. Furthermore, a major change to the tweet content was made in 2017 by increasing the character limit to 280 from 140. Thus, tweets in 2012 were comparatively short compared to recent tweets, and models trained on short tweets might not work effectively on recent long tweets.

Considering the issues mentioned above in available datasets, we decided to create a new dataset named *Twitter Event Data (TED)* to evaluate our approach. Also, we believe that releasing recent datasets would be helpful for the research community. In our dataset, we focus on two diverse domains, sports and politics, to support proving the universality/expandability of a method. This domain selection is also motivated by the coverage of datasets released by Aiello et al. (2013). Mainly, sports is known as a domain with rapid evolution and politics with slow evolution (Adedoyin-Olowe et al., 2016). Also, the word/emoji usage and audiences of these domains have clear distinctions. More details on data collection and cleaning are available in Section 3.2.1 and 3.2.2. The strategies we followed to prepare GT events are discussed in Section 3.2.3. Additionally, we annotated our data with sentiment labels to support event sentiment-based research and the strategy used for data annotation is

described in Section 3.2.4.

3.2.1 Data Collection

We collected tweets using Twitter APIs[5] corresponding to two major events in sports and political domains that happened in 2019. As described in Section 3.1, we filtered the Twitter stream using keywords from each event. Initially, a trending hashtag of each event is used to extract tweets. Then we ranked the hashtags found in the extracted data based on their popularity and used the most popular tags for further extractions[6].

As the sports event, we selected English Premier League 19/20 match between Manchester United and Liverpool Football Clubs (MUFC and LFC) on October 20, 2019. This match was held at Old Trafford, Manchester, and each team scored a single goal. Starting at 15:30, it was held for 115 minutes, including the halftime break. For simplicity, we will refer to this event as '*MUNLIV*'. As the political event, we chose Brexit Super Saturday in 2019, a UK parliament session that happened on Saturday, October 19, 2019, after 37 years. It was organised to vote on a new Brexit deal, but the vote was cancelled due to an amendment passed against the deal. This session started at 08:30 and was held until around 15:30. We will refer to this event as '*BrexitVote*' in the following content. Also, all times are expressed in UTC (Coordinated Universal Time).

For MUNLIV, we collected 118,700 tweets during the period 15:15-17:30. From this collection, we used 99,995 (84.2%) tweets posted during the match (15:28-17:24) for experiments because we could extract GT events only for this

[5]More details about Twitter developer service including its APIs are available at https://developer.twitter.com/

[6]For sports (MUNLIV) data collection, hashtags; #MUNLIV, #MUFC, #LFC, #Liverpool, #GGMU, #PL, #VAR and #YNWA and for political (BrexitVote) data collection, hashtags; #BrexitVote, #SuperSaturday, #Brexit, #BrexitDeal, #FinalSay, #PeoplesVote, #PeoplesVoteMarch were used.

period using published media. More details on GT extraction are described in Section 3.2.3. For BrexitVote, we collected 276,448 tweets during 07:30-17:30 but only used 174,498 (63.1%) tweets posted from the beginning of the parliament session until the vote on the amendment (08:00-14:00) for experiments. Similar to the scenario with MUNLIV, the focus by news media was found to be high until the vote to extract more accurate GT events. Considering the evolution rate of each domain, for MUNLIV, we selected a 2-minute window length and for BrexitVote, a 30-minute length. Since high evolution rates generate more information during short periods, short time windows are appropriate for domains like sports for real-time event extraction. Contrarily, the domains with low evolution rates, like politics, take longer to generate information and long time windows are appropriate for them. After separating the data into chunks, there were 58 time windows for MUNLIV and 12 for BrexitVote. On average, a MUNLIV time window contained 1,724 tweets, and a BrexitVote window contained 14,542 tweets. Data statistics are summarised in Table 3.1.

TABLE 3.1: Statistics of the tweets corresponding to MUNLIV and BrexitVote

Dataset	**Domain**	**Period (UTC)**	**Total Tweets**
MUNLIV	Sports	15:28-17:24	99995
BrexitVote	Politics	08:00-14:00	174498

3.2.2 Data Cleaning

We followed a few language/domain-independent procedures to clean the data. Since word embedding models learn on token data, we tokenised tweet text using the TweetTokenizer model available with Natural Language Toolkit

(NLTK)[7]. TweetTokenizer was designed to be flexible on new domains considering the specialities in the social media text. It tokenises emotions and words specific to social media context (e.g. 1-0, c'mon, #LFC, :-)) correctly. Also, it can remove highly repeated characters to generalise various word forms written by users (e.g. goalll, goallll → goalll). We did not preserve the case sensitivity of tokens because, in social media, people are free to use wordings with different cases without following the standard language rules. Additionally, we removed retweet notations, links and hash symbols in the text. Retweet notations and links are removed because they are uninformative and could also introduce spurious patterns in the data. By removing hash symbols, we can treat hashtags similar to other words during word embedding learning. All these removals were automated using text pattern matching based on regular expressions.

3.2.3 Event Ground Truth (GT) Preparation

We used a similar strategy to Aiello et al. (2013) to prepare GT events. Initially, we reviewed the published media reports related to the chosen topics during the targeted periods and recognised a set of sub-events happened[8]. Each sub-event was labelled with a set of keywords extracted from the media reports which described it. Then, an event is marked as a GT event if at least one of its keywords was found in social media data during the period when that event happened, according to other published media reports. Following this process, 23 time windows were labelled as event windows, with 27 events for the MUNLIV dataset and 8 event windows with 19 events for the BrexitVote

[7]NLTK documentation is available on `https://www.nltk.org/`

[8]For MUNLIV GT event extraction, we reviewed the reports on Premier League official website, BBC Sports and WhoScored. For BrexitVote, we reviewed the reports on The Guardian and TheyWorkForYou parliamentary monitoring website.

dataset. We made these data, including the GT labels (TED), publicly available to support other research in event detection[9].

This approach does not consider the possibility of variations between the event reporting times in two different media. However, after carefully analysing the data and events, we noticed that it is possible to have slight temporal variations in social media events than the events in published media reports. For example, during a football match, people report on social media that a substitution is going to happen, seeing a player get ready to come before it actually happens or other media reports it. Without considering such variations, some GT events can be missed, and some can be labelled with incorrect time details. Thus, to improve the correctness of GT labels, for each GT event recognised at time window t (according to published media reports), we manually analysed the temporal variations of keyword frequencies in social media data in previous and next time windows $t-1$ and $t+1$ to identify the actual event reported time in social media. Following this analysis, the MUNLIV dataset is recognised with 31 events in 25 windows and the BrexitVote dataset with 27 events in 11 windows. The updated version (V2) of GT is also published in our data repository.

3.2.4 Event Sentiment Annotation

Additionally, we decided to annotate our datasets with sentiment labels to make them suitable for event sentiment-based research, especially considering the following limitations. To the best of our knowledge, none of the available social media datasets contained ground truth of events and sentiments together. Also, most sentiment datasets were composed of random sets of social media posts without the postings during a continuous period (Rosenthal

[9]TED including the GT events are available on https://github.com/hhansi/twitter-event-data-2019

et al., 2017; Aloufi and Saddik, 2018). We named this dataset version as *Twitter Event Data with Sentiments (TED-S)*.

Considering the cost of a manual process in annotating large datasets, we propose an ensembled approach for sentiment annotation. For this approach, we utilised the data annotation strategy proposed with democratic co-learning (Zhou and Goldman, 2004) considering its successful applications in different areas such as time-series prediction (Mohamed et al., 2007) and offensive language identification (Rosenthal et al., 2021). We trained a set of classifiers on available labelled data and a manually annotated subset of MUNLIV and BrexitVote data (8,344 tweets from MUNLIV and 2,016 tweets from BrexitVote), using different learning algorithms for this strategy. When different algorithms with different inductive biases are involved, it helps resolve individual model biases and produces predictions with lower noise. Then, the trained models are used to make predictions on unlabeled data and aggregate the outputs to generate final labels. We constructed classifiers using Long Short-term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), Convolutional Neural Network (CNN) (Lawrence et al., 1997) and Transformers (Devlin et al., 2019) to use with this approach, following their recent wide applications.

We labelled all the tweets of MUNLIV during the period of 15:28-17:24 and BrexitVote during 08:00-14:00 (Table 3.1) using our approach. The corresponding sentiment distributions are illustrated in Figures 3.1 and 3.2, respectively. Since more than one tweet can be posted during a particular time, we aggregated the repeated values and showed the mean with a 95% confidence interval in these line graphs. More details about data statistics and the conducted experiments, along with sentiment labelling, are available in our paper (Hettiarachchi et al., 2022b). Also, this dataset is publicly available to use by anybody interested in event-sentiment analysis[10]. However, adhering to

[10]TED-S is available on `https://github.com/HHansi/TED-S`.

this PhD's scope, plan, and timeline, we did not conduct any event sentiment-based research as a part of the PhD research. But, we plan to explore this area further in our future research utilising this dataset.

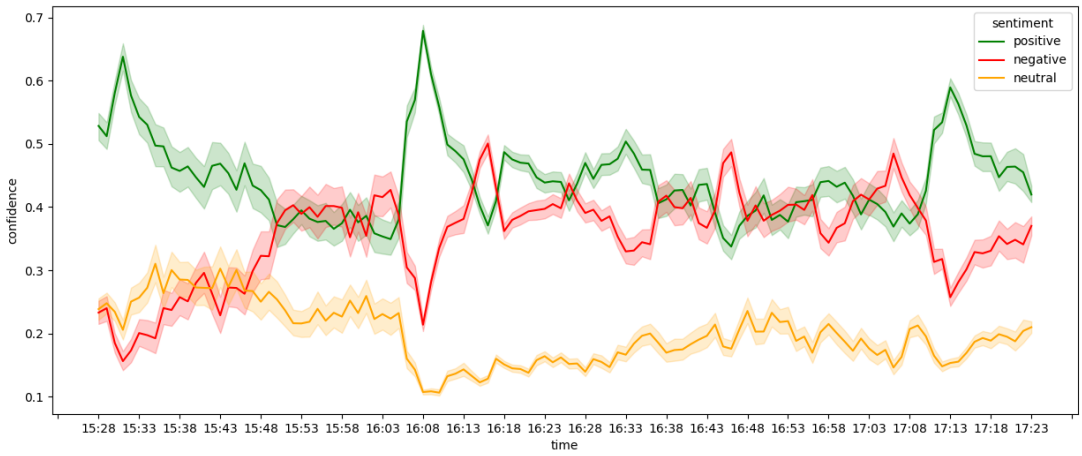


FIGURE 3.1: Sentiment distribution of MUNLIV tweets during 15:28-17:24

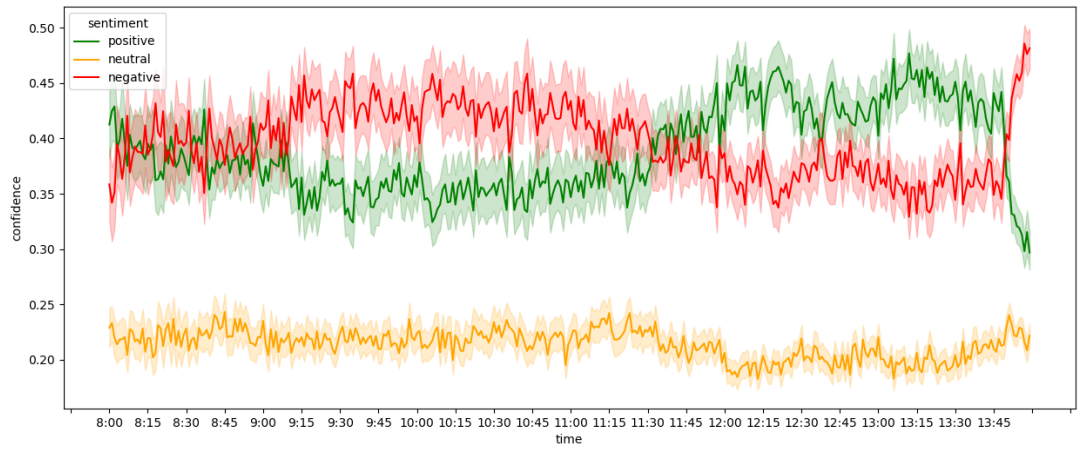


FIGURE 3.2: Sentiment distribution of BrexitVote tweets during 08:00-14:00

3.3 Evaluation Metrics

Identified events need to be compared with ground truth (GT) events considering both temporal and textual aspects to evaluate the performance of event

detection methods designed for social media data streams. Analysing previous research, we found methods that use either time-based metrics (Adedoyin-Olowe et al., 2016) or event text-based metrics (Aiello et al., 2013; Choi and Park, 2019; Morabia et al., 2019) for automated evaluations without a combination of both. Thus, utilising the available knowledge, we designed the following metrics (Section 3.3.1-3.3.2) to perform an overall evaluation.

3.3.1 Time-based Evaluation Metrics

We involved time windows in evaluating temporal event details using the following metrics, which are positively oriented. In the equations stated below, the set of all GT event windows in the dataset, detected event windows and relevant event windows found in detected windows are represented by W^{GT}, W^d and W^r, respectively. A detected window is marked as a relevant event window if all the GT events of that period are found in the events detected for that window.

- ***Time Window Recall:*** Fraction of the number of relevant event windows detected among the total number of GT event windows in the dataset.

$$Time\ Window\ Recall = \frac{|W^r|}{|W^{GT}|}$$

- ***Time Window Precision:*** Fraction of the number of relevant event windows detected among the total number of event windows detected.

$$Time\ Window\ Precision = \frac{|W^r|}{|W^d|}$$

- ***Time Window F1:*** Weighted harmonic mean of Time Window (TW) Recall and Precision.

$$Time\ Window\ F1 = 2 \times \frac{TW\ Precision \times TW\ Recall}{TW\ Precision + TW\ Recall}$$

3.3.2 Text-based Evaluation Metrics

Detected events and their keywords are compared with GT events using the following metrics, which are also positively oriented. In the equations stated below, $E_{W_t^{GT}}$ represents the GT events reported at time t (or W_t) and $E_{W_t^d}$ represents the events identified during the detected event window W_t^d. A match between a detected event and a GT event is established if at least one GT keyword is found from the identified event words.

- ***Event Recall:*** Fraction of the events successfully detected among the GT events happened during the time windows detected as event windows.

$$Event\ Recall = \frac{\sum_{t \in W^d} |e : e \in E_{W_t^{GT}}\ and\ e \in E_{W_t^d}|}{\sum_{t \in W^d} |e : e \in E_{W_t^{GT}}|}$$

- ***Event Relevance:*** Fraction of the detected events related to the GT events that happened during the time windows detected as event windows among all detected events.

$$Event\ Relevance = \frac{\sum_{t \in W^d} |e : e \in E_{W_t^d}\ and\ e \in E_{W_t^{GT}}|}{\sum_{t \in W^d} |e : e \in E_{W_t^d}|}$$

While matching detected events with GT events to calculate Event Recall, we did not allow the same detected event to match with multiple GT events. Otherwise, if a single event is detected, including all the keywords in GT events, it will be marked as a prediction of all events, even though having a single large event is not helpful in the real scenario. If the same detected event is matched with more than one GT event, only the best match (match with the minimum number of missed keywords) is kept. For the optimal event assignment, we used the Hungarian algorithm (Munkres, 1957) developed for the assignment problem.

Further, we used a keyword-based metric to measure the textual coverage of detected events. In the equation below, e_{W_t} is an event detected during the time window W_t, which matches with the GT event e_{GT} reported at the same

time window. Each detected and GT event contains a set of words/tokens which describes it. While comparing predicted event words with GT words, their exact matches are considered.

- ***Keyword Recall:*** Fraction of the correctly detected keywords among the total number of keywords of the GT events that have been matched to some candidate event in the time window under consideration. Only the optimally assigned events are considered.

$$Keyword\ Recall = \frac{\sum_{t \in W^d} |w : w \in e_{GT} \cap e_{W_t}, e_{GT} \in E_{W_t^{GT}}, e_{W_t} \in E_{W_t^d}|}{\sum_{t \in W^d} |w : w \in e_{GT}, e_{GT} \in E_{W_t^{GT}}|}$$

3.4 Theoretical Background

We adapt approaches based on word embeddings and hierarchical clustering for social media event detection in this research considering their potentials. Section 3.4.1 summarises the background details of word embeddings and their capabilities. The basic concepts of hierarchical clustering and dendrograms are explained in Section 3.4.2.

3.4.1 Word Embeddings

Word embeddings are numerical representations of text in vector space. Depending on the learning method, there are two main groups: frequency-based and prediction-based embeddings. Frequency-based embeddings consider different frequency measures of text to generate vectors preserving statistical features. Prediction-based embeddings learn representations using contextual predictions preserving both syntax and semantics in text. Considering these characteristics, we focus on prediction-based word embeddings in this research and will refer to them as *'word embeddings'*.

Different model architectures such as Neural Network Language Model (NNLM) (Bengio et al., 2003) and Recurrent Neural Network Language Model (RNNLM) (Mikolov et al., 2010) were proposed by previous research to generate prediction-based word embeddings. Considering the complexities of these architectures, log-linear models, which are also known as Word2Vec models (Mikolov et al., 2013a) were introduced later. The evaluations conducted by Mikolov et al. (2013a) showed that Word2Vec vectors have a high capability of preserving syntactic and semantic relationships between words. Thus, they are popularly used in the domain of NLP (Zhang et al., 2019; Yilmaz and Toklu, 2020; Škrlj et al., 2020). Two architectures named Continuous Bag-of-Words (CBOW) and Continuous Skip-gram were proposed as Word2Vec models. CBOW learns by predicting a word based on its context. Contrarily, Skip-gram learns by predicting the context of a given word. Among these two architectures, we focus on Skip-gram in this research because it resulted in high semantic accuracy compared to CBOW in previous analyses (Mikolov et al., 2013a; Mikolov et al., 2013b). Also, according to our initial experiments and analyses, Skip-gram outperformed the CBOW model. More details about Skip-gram architecture are available in Section 3.4.1.1. Furthermore, Section 3.4.1.2 discusses the qualities of Skip-gram embeddings learned on real datasets.

3.4.1.1 Skip-gram Model

Skip-gram model is a log-linear classifier composed by a 3-layer neural network with the objective of predicting the context or surrounding words of a centre word given a sequence of training words $w_1, w_2, ...w_n$ (Mikolov et al., 2013b). More formally, it focuses on maximizing the average log probability of context words $w_{k+j}| - m \leq j \leq m, j \neq 0$ of the centre word w_k by following the objective function in Equation 3.1. m represents the length of the training

context.

$$j = \frac{1}{n} \sum_{k=1}^{n} \sum_{-m \leq j \leq m, j \neq 0} \log p(w_{k+j}|w_k) \tag{3.1}$$

The probability of a context word given the centre word $p(w_{k+j}|w_k)$ is computed using the softmax function.

$$p(w_o|w_i) = \frac{\exp({v'_{w_o}}^T v_{w_i})}{\sum_{w=1}^{N} \exp({v'_w}^T v_{w_i})} \tag{3.2}$$

In Equation 3.2, w_o and w_i represent the output and input (i.e. context and centre words) and N represents the length of vocabulary. The input and output vectors of a word w is represented by v_w and v'_w. The input vectors are taken from the input-hidden layer weight matrix M which is sized $N \times D$ where D is the number of neurons in the hidden layer. Likewise, output vectors are taken from hidden-output layer weight matrix M' which is sized $D \times N$. The Skip-gram architecture including weight matrices is shown in Figure 3.3.

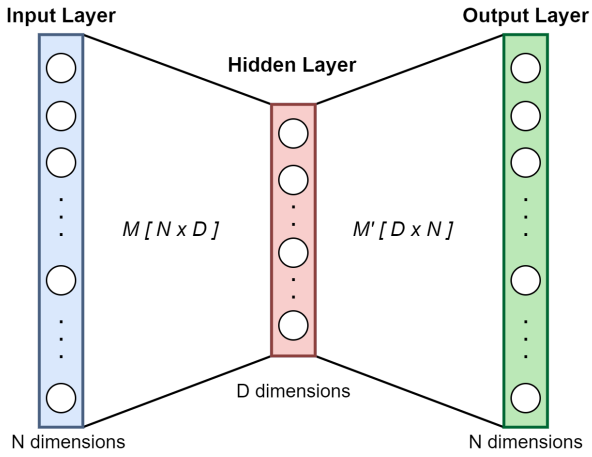


FIGURE 3.3: Skip-gram Architecture

Once the model converges, it gets the ability to predict the probability distributions of context words with good accuracy. At this point, instead of using the model for the trained task, adjusted weights between the input and hidden layers will be extracted as word vectors or embeddings. The vector

dimensionality equals the number of neurons in the hidden layer. During the training procedure, model weights get adjusted by learning the connections between nearby words. Thus, given a sufficient corpus, the model can learn connections between words capturing their underlying syntax and semantics, allowing to recognise similar/related words more effectively.

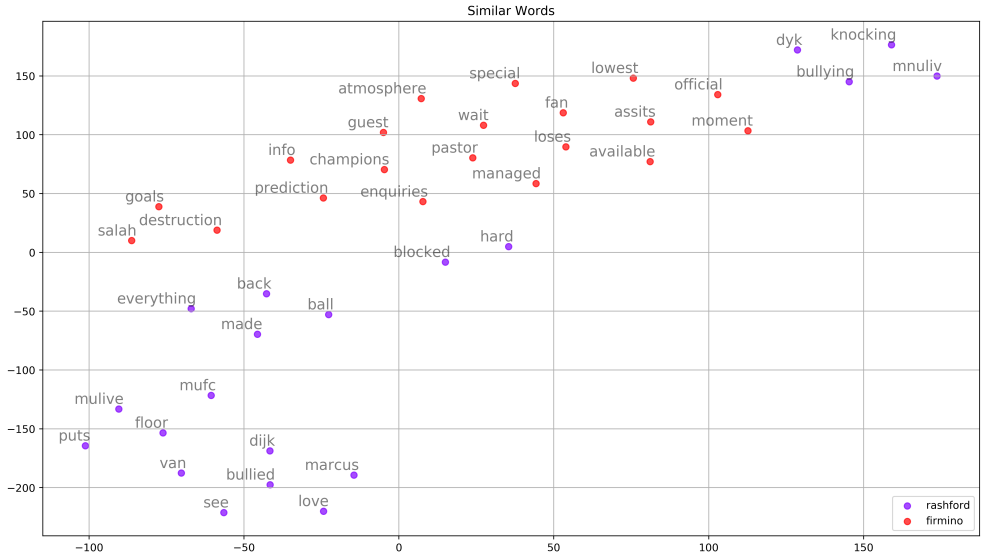
3.4.1.2 Skip-gram Vector Spaces

To analyse the characteristics and distribution of Skip-gram embeddings, we trained a few models using MUNLIV data. We used 2-minute windows as mentioned in Section 3.2.1. Using the learned embeddings, we analysed the words closer to the player names *'rashford'* and *'firmino'* during 15:52-15:54 and 16:06-16:08. The events reported within or closer to these time windows are summarised in Table 3.2. For word visualisations, we used T-distributed Stochastic Neighbor Embedding (t-SNE) algorithm (Maaten and Hinton, 2008), and the resulted graphs are shown in Figure 3.4.

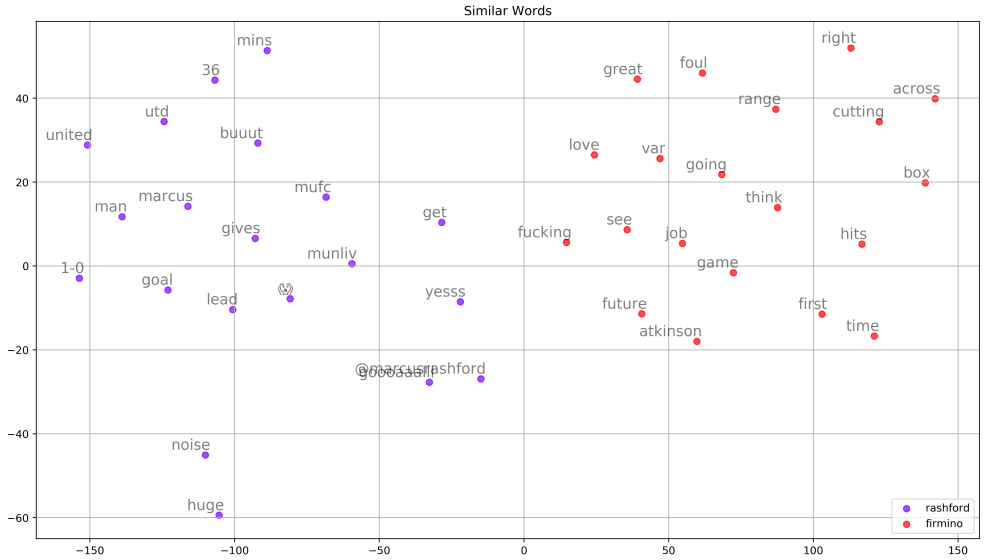
TABLE 3.2: Sample events from MUNLIV

Time	Event	Description
15:40	Missed Attempt	Missed attempt by Roberto Firmino(LFC)
15:52	Foul	Foul by Marcus Rashford(MUFC) on Virgil van Dijk(LFC)
16:04	Saved Attempt	Saved attempt by Roberto Firmino(LFC)
16:06	Goal	First goal by Marcus Rashford(MUFC)

The 15:52-15:54 visualisation (Figure 3.4a) shows that the foul-related words are located closer to *'rashford'* in the vector space representing the foul that happened during this period. Even after 12 minutes, a few words relating to the missed attempt at 15:40, such as *'loses'* and *'destruction'*, can be seen closer to *'firmino'*. Similarly, at 16:06-16:08, the goal can be clearly identified by the words nearby to *'rashford'* (Figure 3.4b). Also, more words relating to the saved attempt are located closer to *'firmino'*, as it happened 2 minutes before.



(A) time window 2019-10-20 15:52-15:54



(B) time window 2019-10-20 16:06-16:08

FIGURE 3.4: t-SNE visualisations of tokens closer to *'Rashford'* and *'Firmino'*

Furthermore, this diagram shows a clear separation in nearby word groups in the vector space when they represent actively discussing events.

These analyses prove that nearby word-groups recognised by Skip-gram word embeddings represent the events in the underlying corpus. Also, these groups capture directly as well as indirectly related words to events. For example, the top 20 words closer to *'rashford'* during 16:06-16:08 contains the words

such as *'goal'*, *'1-0'*, *'mufc'* and *'36'* which are directly related to and words such as *'huge'* and *'noise'* which are indirectly related to the goal event. These characteristics associated with Skip-gram embeddings can be effectively utilised for event detection in social media, capturing the linguistics in underlying data.

3.4.2 Hierarchical Clustering

Hierarchical clustering was popularly used by previous research for event detection considering the requirement to predefine the cluster count by flat clustering algorithms (e.g. K-means) (Corney et al., 2014; Ertugrul et al., 2017; Mu et al., 2018; Nguyen et al., 2019). Even though flat clustering is more efficient than hierarchical clustering, it is impractical to identify the number of clusters (events) in advance due to the unpredictability of social media data. Additionally, hierarchical clustering returns a hierarchy or structure of data points, known as a *dendrogram*, that can be used to identify the connections between data points.

There are two types of hierarchical clustering algorithms: (1) bottom-up or agglomerative and (2) top-down or divisive (Manning et al., 2008). In hierarchical agglomerative clustering (HAC), all data points are considered as separate clusters at the beginning and then merged based on cluster distance using a linkage method. The commonly used linkage criteria are single, complete and average. Single linkage considers the maximum similarity, and complete linkage considers the minimum similarity. Average of all similarities are considered in the average linkage. In contrast to HAC, hierarchical divisive clustering (HDC) considers all data points as one cluster at the beginning and then divide them until each point is in its own cluster. For data division, HDC requires a flat clustering algorithm. Due to this requirement, HDC is more complex than HAC, and it is recommended to use some stopping rules when

processing large datasets to avoid the generation of the complete dendrogram to mitigate the complexity (Roux, 2018). Thus, the HAC algorithm is more appropriate for this research because we need to process large datasets efficiently and obtain complete dendrograms to reveal textual relationships.

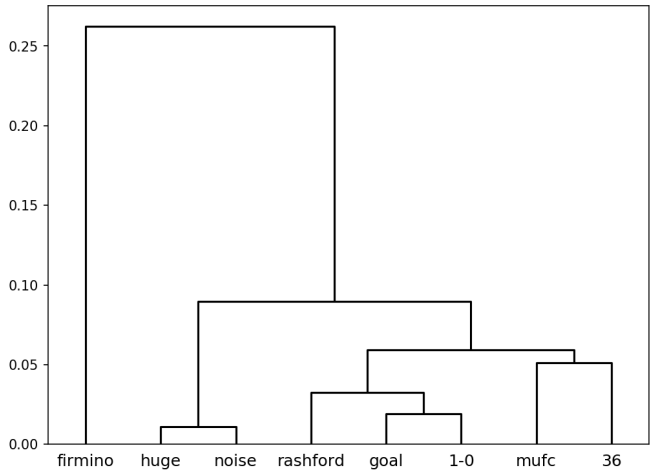


FIGURE 3.5: Sample dendrogram (y-coordinate denotes the distance and x-coordinate denotes the selected words)

Dendrogram: A dendrogram is a tree diagram that visualises hierarchical clusterings illustrating the relationships between objects. A sample dendrogram generated on a selected word set from tweets posted during the first goal of the MUNLIV dataset is shown in Figure 3.5. The horizontal lines represent the cluster merges that happened, considering the distance between clusters of words. Merges between the closer groups, such as the name of the player who scored the goal *'rashford'* and the cluster that holds *'goal'* happen at low distances ($\approx$ 0.025). Contrarily, merges between distant groups such as another player name *'firmino'* and the cluster of *'goal'* happen at high distance values ($\approx$ 0.25). Similarly, a dendrogram built on a corpus from any domain preserves informative relationships expressed in that corpus.

3.5 Notations

All the notations which are commonly used throughout this part of the thesis are summarised in Table 3.3.

TABLE 3.3: Summary of notations used in this part of the thesis

Notation	Description
d	document/post in a data stream D
D'	filtered data stream
W	time windows during a period
W^d	detected event windows during a period
W_t	window in W at time t
W_{t-1}	window in W at time $t-1$ (previous time window to W_t)
w	word/token
c	word/token cluster
e	event
E_{W_t}	set of events happened during W_t/co-occurred events during W_t
V_t	word embeddings/vector space learned from W_t
$vocab$	vocabulary
f_t	token frequencies of $vocab$ at time t ($vocab_t$)
dl	dendrogram level
$dl_{r \to x}$	number of dls from root: r to node: x
$dl_{(w_i,w_j)}$	number of shared dls between tokens: w_i and w_j, from root
L	set of leaf nodes in a dendrogram

3.6 Summary

Automated intelligent mechanisms are crucial to effectively extract newsworthy content from the vast volume of data generated in social media. It is impractical to manually analyse the social media data streams without any automation, considering their volume with high growth rates and dynamicity. Further emphasising the importance of event detection, previous research has also proposed different approaches for this task. However, most of them do

not involve underlying semantics, which describes the connections between words and their meanings for event detection from textual data, making those systems vulnerable to a critical information loss. Overcoming this major limitation, in this research, we focus on developing methods involving underlying semantics to effectively capture social media events with less information loss. We propose using the characteristics of self-learned word embeddings and their hierarchical relationships in dendrograms, paving a new direction for social media event detection.

In the next few chapters, we further describe our approaches to social media event detection. We mainly aimed at two levels of data granularities (i.e. coarse and fine) while designing our approaches considering different user requirements. Chapter 4 describes the coarse-grained approach and Chapter 5 describes the fine-grained approach. At the coarse-grained level, we target notifying users about event occurrences in time, allowing them to take further actions of their choice, such as manual data analysis to get event details. At the fine-grained level, we target enriching the output with textual event details in addition to temporal details automating the complete event detection process.

Chapter 4

Embed2Detect: Coarse-grained Level– Event Window Identification

As described in Chapter 3, this research focuses on detecting social media events based on two levels of data granularity, coarse and fine, considering different information requirements. Mainly, at the coarse-grained level, notifying users about event occurrences is targeted, and at the fine-grained level, extraction of event-described text segments at event occurrences is targeted. As the coarse-grained level of social media event detection, we target identifying event occurred time windows. Since social media generates a vast volume of data, it is impractical to manually filter all the available data to get interesting information, especially in real-time. However, if there is a mechanism to notify users about event occurrences in time, they can further analyse or filter upon such notification to get the necessary information. In this way, people can focus on important information in real-time without missing any and effectively utilise the vast amount of data generated in social media. Following this idea, fulfilling the coarse-grained level requirement of social media event detection, in this chapter, we propose a novel method named *Embed2Detect* to identify event occurred time windows (event windows) in (near) real-time. The findings reported in this chapter have been published in (Hettiarachchi et al., 2021b; Hettiarachchi et al., 2022a).

Considering the lack of semantic involvement in previous methods, which could lead to severe information loss, we targeted designing Embed2Detect involving all the important features in textual data: statistics and linguistics (syntax and semantics). Embed2Detect mainly utilises the characteristics of word embeddings and dendrograms (or hierarchical clusters). We propose using self-learned word embeddings over pre-trained embeddings to capture features specific to the targeted corpus. Our approach mainly structures tokens in a corpus to hierarchical clusters based on their linguistical relationships captured by word embeddings and analyses the temporal variations of clusters to identify event occurrences. Also, we use vocabulary changes over time to involve statistical details with our identifications. As tokens, we consider all the meaningful text units, including words, numerals and emojis, extending event coverage beyond the words. To measure cluster similarity between tokens to capture their temporal variations, we also introduce a new measure named *Dendrogram Level (DL) Similarity* along with Embed2Detect. We evaluated the performance of Embed2Detect in two diverse domains with several recently proposed methods and obtained promising results confirming its effectiveness for (near) real-time event window identification.

To the best of our knowledge, Embed2Detect is the first method that uses self-learned word embedding-based temporal cluster similarity for event window identification in social media. In summary, the main contributions of this chapter are as follows.

1. We propose a novel method named *Embed2Detect* for (near) real-time event window identification in social media by involving semantics captured by self-learned word embeddings in addition to the traditionally used statistical and syntactical features in the text.

2. We introduce a text similarity measure named *Dendrogram Level (DL)*

Similarity focusing on the characteristics associated with dendrograms to measure cluster similarity between tokens capturing their hierarchical relationships.

3. We leverage self-learned word embeddings in Embed2Detect to offer more effective and flexible event window identification, overcoming previous methods' limitations, specifically, the less semantic involvement, inability to capture linguistics specific to the underlying data and less expandability to different domains, languages and platforms.

4. We evaluate Embed2Detect on recent real datasets from two diverse domains, sports and politics, and show its effectiveness compared to several recently proposed methods from different competitive areas.

5. We release the implementation of Embed2Detect as an open-source project to support applications and research in the area of event detection[1].

The rest of this chapter is organised as follows. Section 4.1 introduces the proposed approach: Embed2Detect for event window identification in social media data streams. Section 4.2 comprehensively describes the conducted experiments and obtained results. Finally, Section 4.3 concludes the chapter with a discussion and future research directions.

4.1 Methodology - Embed2Detect

As *Embed2Detect*, we propose a novel event detection approach to recognise event occurred time windows based on the temporal variations of tokens and

[1]Embed2Detect implementation is publicly available on https://github.com/hhansi/embed2detect

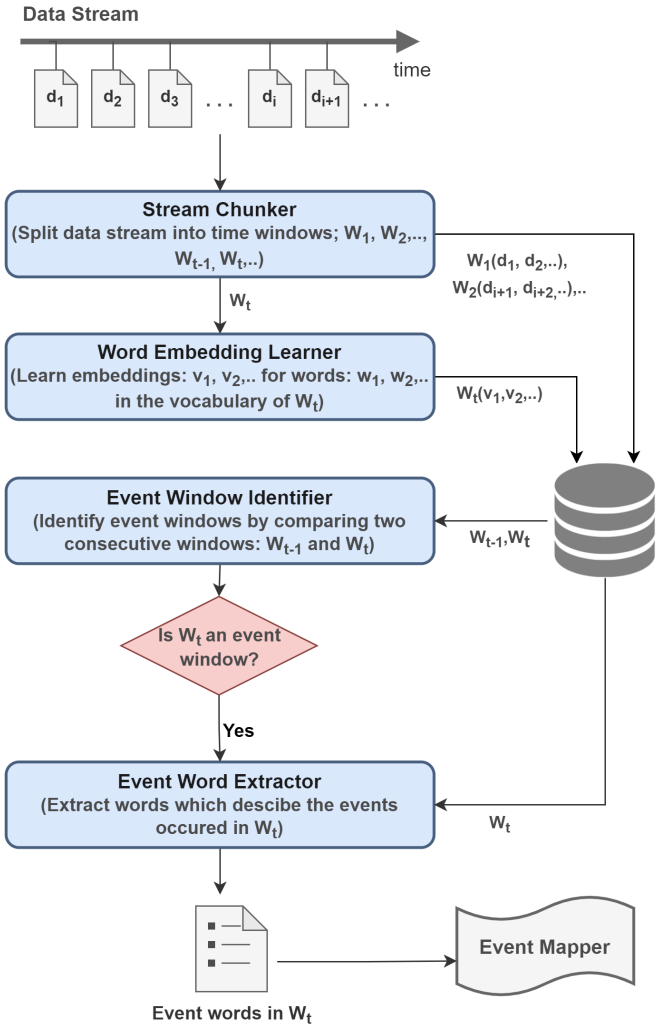


FIGURE 4.1: Overview of the proposed method – Embed2Detect

their hierarchical relationships. The main novelty of this approach is the involvement of corpus-oriented semantical features and their temporality for event detection using self-learned word embeddings. The Embed2Detect system has four main components: (1) stream chunker, (2) word embedding learner, (3) event window identifier and (4) event word extractor as shown in Figure 4.1. Self-learned word embeddings are used during event window identification and event word extraction phases. Additionally, we use the event mapper to map detected events with ground truth (GT) events during evaluations. Each of the main components is further described in Sections 4.1.1

- 4.1.4. Finally, the computational complexity of Embed2Detect is discussed in Section 4.1.5.

4.1.1 Stream Chunker

Data stream mining is mainly supported by three different time models, namely, landmark model, tilted-window model and sliding window model (Tsai, 2009). All the data from a specific time to the present is considered equally in the landmark model. The tilted-window model treats recent data with high importance compared to old data. The sliding window model splits the data stream into windows based on a fixed period or number of transactions and performs data mining tasks on the data that belong to each window.

Among these models, the time-based sliding window model was widely used by previous research in event detection (Sayyadi et al., 2009; Alkhamees and Fasli, 2016; Adedoyin-Olowe et al., 2016; Choi and Park, 2019). Following this tendency and considering the high data volume to process and requirements to capture temporal details, Embed2Detect also uses the sliding window model with a fixed time frame to facilitate the event detection in social media data streams. Also, the time window length can be adjusted depending on the evolution of events in the targeted domain. Stream chunker is the component that conducts data stream separation into windows.

4.1.2 Word Embedding Learner

We use word embeddings to incorporate linguistical features in text for event detection. Without using pre-trained word embeddings, we propose using self-learned embeddings or embeddings trained on the targeted corpus to capture its unique characteristics, such as modified or misspelt words and emoticons. The word embedding learner transfers the text of social media posts

in a time window to a vector space. For each time window, different vector spaces are learned so that they can be used to capture temporal variations over windows. The learned word embedding models are stored to facilitate event window identification and event word extraction.

Considering the high-quality vector representations by the Skip-gram algorithm (Section 3.4.1), we use it as the initial algorithm to learn embeddings in Embed2Detect. Also, due to the simplicity of Skip-gram architecture and usage of small training corpora (chunks of a data stream), learning time will not be considerably high to impact real-time event detection negatively. We further discuss the possible extensions to the embedding learning algorithm in Section 4.2.6. By designing the word embedding learner as a separate component, we also facilitate the easy integration of any appropriate embedding model.

4.1.3 Event Window Identifier

Given a chronological stream of time windows W: $W_0, W_1, ...W_{t-1}, W_t, ...$ (at least two consecutive windows W_{t-1}, W_t), event window identifier recognises the windows W^d where events have occurred. Since an event is an incident or activity that happened and was discussed, such an occurrence should significantly change data in the corresponding time window compared to its previous window. Based on this assumption, our method identifies windows with higher change than a predefined threshold (α) as event windows. From the perspective of use cases, this threshold mainly defines the significance of targeted incidents. A low α value would also identify less important events. Since we use normalised values to measure the change as described below, any value between 0 and 1 can be picked for α.

Before moving into the change calculations, the text in social media documents needs to be pre-processed for more accurate results and efficient processing. We do not conduct any pre-processing steps before learning the embeddings except tokenizing to preserve all valuable information, which would help the neural network to figure things out during word embedding learning. Under pre-processing, we mainly targeted removing the uninformative tokens using general procedures which are not strictly dependent on domains or languages to preserve the expandability of our approach. We remove redundant punctuation marks, stopwords and tokens with low frequency than a predefined threshold (β) as outlier tokens. The low-frequent words would be the misspelt words or words describing background topics that are not intensively discussed as events.

An event occurrence will change nearby tokens of a selected token or introduce new tokens to the vocabulary over time. For example, in a football match, if a goal is scored at W_{t-1}, *'goal'* will be highly mentioned in the textual context of the corresponding player's name. If that player receives a yellow card unexpectedly in W_t, new words *'yellow card'* will be added to the vocabulary, and they will appear in the context of the player's name instead of the word *'goal'*. Following this idea, in Embed2Detect, we consider two measures that capture the changes in nearby tokens and vocabularies to calculate textual data change between two consecutive time windows W_{t-1} and W_t. To compute the nearby token changes, we propose an approach based on clustered word embeddings under cluster change calculation (Section 4.1.3.1) using a novel similarity measure named *Dendrogram Level (DL) Similarity* (Section 4.1.3.2). Involvement of word embeddings allows the effective capturing of nearby word changes considering both syntactical and semantical aspects (Section 3.4.1). Vocabulary change is calculated using the approach described in Section 4.1.3.3, mainly targeting the statistical aspect. The final value for the overall textual change

between time windows is calculated by aggregating the two measures, cluster change and vocabulary change. To maintain the simplicity of the proposed approach, as the aggregation method, we experimented with maximum and average value calculations. Among these two methods, the best results could be obtained by maximum calculation for all the experimented diverse domains (Section 4.2.1). Algorithm 1 summarises the complete flow of event window identification.

Algorithm 1: Event Window Identification

input: W : time windows during a period ($[W_0, W_1, ...W_{t-1}, W_t, ...]$), α : change threshold
output: W^d : detected event windows

$W^d \leftarrow []$; `// empty array to keep event windows`
for *t=1 to length(W)* **do**
  $vocab_{t-1} \leftarrow$ vocabulary at W_{t-1};
  $vocab_t \leftarrow$ vocabulary at W_t;
  $V_{t-1} \leftarrow$ vector space at W_{t-1};
  $V_t \leftarrow$ vector space at W_t;
  `//` **Compute cluster change**
  $commonVocab \leftarrow$ common vocabulary for $vocab_{t-1}$ and $vocab_t$;
  $N \leftarrow length(commonVocab)$;
  `// Compute cluster similarities`
  $matrix_{t-1} \leftarrow$ similarity matrix of $commonVocab$ using V_{t-1};
  $matrix_t \leftarrow$ similarity matrix of $commonVocab$ using V_t;
  `// Compute similarity change`
  $diffMatrix \leftarrow matrix_t - matrix_{t-1}$;
  $clusterChange \leftarrow$ overall change calculated using $diffMatrix$;
  `//` **Compute vocabulary change**
  $vocabChange \leftarrow$ change calculated using $vocat_t$ and $vocab_{t-1}$;
  `//` **Compute overall change**
  $overallChange \leftarrow aggregate(clusterChange, vocabChange)$;
  if $overallChange \geq \alpha$ **then**
    $W^d.add(W_t)$;
  end
end
return W^d

4.1.3.1 Cluster Change Calculation

Cluster change calculation is proposed to measure nearby token or token group changes over time based on their hierarchical relationships. Mainly, token groups/clusters are targeted because ideas discussed in data streams form such groups, and their variations can be used to recognise newsworthy events. We generate separate hierarchical clusterings per time window for this calculation. Focusing on our requirement, we cluster tokens, and as tokens, we use words as well as other useful symbols, such as emojis, because they are widely used to express ideas in social media. As token representations, self-learned word embeddings are used, considering their ability to preserve linguistical features of the underlying corpus. For the clustering algorithm, we chose hierarchical agglomerative clustering (HAC), considering its advantages and the tendency of previous research (Section 3.4.2). As the linkage method, we use the average scheme to involve all the elements that belong to clusters during distance calculation. In average linkage, the distance between two clusters C_i and C_j is measured by following the Equation 4.1 (Müllner, 2011),

$$D(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{w_p \in C_i} \sum_{w_q \in C_j} d(w_p, w_q), \tag{4.1}$$

where $d(w_p, w_q)$ represents the distance between cluster elements (i.e. tokens) w_p and w_q which belong to the clusters C_i and C_j respectively. This distance is measured using cosine distance because it proved effective for measurements in textual data (Mikolov et al., 2013a; Mikolov et al., 2013b; Antoniak and Mimno, 2018). Since cosine distance calculation is independent of the magnitude/length of vectors (i.e. scalar that represents the size/length of a vector in a given vector space), it also does not get biased by the word frequency,

which is merely represented by the length of word vectors (Schakel and Wilson, 2015).

Since we generate hierarchical clusterings (or dendrograms) only to support the cluster change calculation, our method does not directly rely on the final clusters produced by the clustering algorithm as the majority of available methods (Li et al., 2017a; Ertugrul et al., 2017; Comito et al., 2019b). We only use the variations in hierarchical relationships to recognise nearby token changes over time. In this setting, the requirement for a cluster threshold can be eliminated, and it is advantageous in the context of social media due to the infeasibility to define a static threshold suitable for all event clusters considering their diversity.

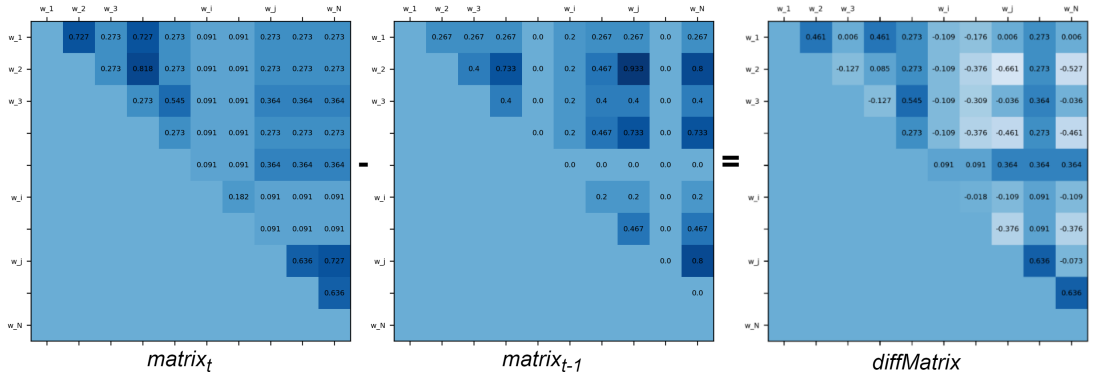


FIGURE 4.2: Matrix-based cluster change calculation (The colour scheme indicates high to low values using variants from dark blue to white.)

We propose a matrix-based approach to calculate cluster change efficiently. Since our target is to measure the change at a time window W_t compared to its previous window W_{t-1}, we generate matrices that hold the similarity values between all possible token pairs per window. Such a token similarity matrix is a square matrix of size $N \times N$ where N is the number of tokens in the vocabulary ($matrix_t$ and $matrix_{t-1}$ in Figure 4.2). Each cell in the matrix $matrix[i, j]$ represents the cluster similarity between tokens w_i and w_j calculated using *Dendrogram Level similarity* (Section 4.1.3.2). Due to the requirement to compare

similarity matrices in two consecutive time windows, we use one vocabulary to generate the matrices. As this common vocabulary, the pre-processed vocabulary at t $vocab_t$ is used for both windows W_{t-1} and W_t following our aim to capture the changes at W_t compared to W_{t-1}. After generating the similarity matrices at $t-1$ and t, we calculate their difference matrix *diffMatrix* (Figure 4.2) to use with the change calculation. Finally, as the overall cluster change, the average of absolute values (*Absolute Similarity Change*) is calculated.

Absolute (Abs.) Similarity Change: Abs. similarity change is calculated as the average of absolute values in *diffMatrix* following the Equation 4.2. Only the values at the upper triangular matrix except the diagonal are considered because the matrix is symmetric around the diagonal. This is a positively-oriented measure that gets a higher value for high changes.

$$Abs.\ Similarity\ Change = \frac{\sum_{i=1}^{N}\sum_{j=i+1}^{N}|diffMatrix[i,j]|}{(N\times(N-1))/2} \tag{4.2}$$

4.1.3.2 Dendrogram Level (DL) Similarity

Focusing on the characteristics associated with dendrograms (Section 3.4.2), we propose *Dendrogram Level (DL) Similairity* to calculate cluster similarity of tokens. As dendrogram levels, we consider horizontal lines or merges in the dendrogram. Given a dendrogram, the similarity between a token pair w_i and w_j is calculated as the normalised value (within the range of 0–1) of shared levels from root between those two words, as follows.

$$DL\ Similarity_{(w_i,w_j)} = \frac{dl_{(w_i,w_j)}}{max(dl_{r\to x} : x \in L)+1} \tag{4.3}$$

The numerator of Equation 4.3 represents the number of shared dendrogram

levels between w_i and w_j from the root. The denominator represents the maximum number of levels between root and leaf nodes. We added leaf node level also as an individual level during maximum level count calculation to make sure only the similarity between the same token is 1 ($DL\ Similarity_{(w_i,w_i)} = 1$).

For example, the maximum number of dendrogram levels in the dendrogram in Figure 3.5 without the leaf node level is 5 considering its longest path $r \rightarrow goal$ (or $r \rightarrow 1 - 0$). By adding the leaf level, the maximum level count becomes 6. For the word pair *'rashford'* and *'goal'*, the number of shared levels between the words is 4. Similarly, words; *'firmino'* and *'goal'* shares 1 level, because they appear in distant clusters. In measures, DL similarities between these words can be calculated as follows.

$$DL\ Similarity_{(rashford,goal)} = 4/6 = 0.667$$
$$DL\ Similarity_{(firmino,goal)} = 1/6 = 0.167$$

As can be seen in the above values, for hierarchically closer words, DL similarity returns a higher value than the distant words. The higher the DL similarity, the corresponding word pair has a close hierarchical relationship or high cluster similarity.

4.1.3.3 Vocabulary Change Calculation

A vocabulary is a set of distinct tokens that belong to a particular language, person, corpus, etc. In this research, we consider the tokens belonging to each time window's data corpus as separate vocabularies. Vocabulary change calculation is proposed to measure new word addition into time windows over time. Also, it incorporates the statistical details in the dataset. In order to have

a comparable value over all time windows, we calculated normalised vocabulary change value for W_t compared to W_{t-1} following the Equation 4.4.

$$Vocabulary\ Change(VC)_{(t-1,t)} = \frac{|w : w \in vocab_t \text{ and } w \notin vocab_{t-1}|}{|vocab_t|} \tag{4.4}$$

The numerator represents the cardinality of new tokens that appeared in the vocabulary of W_t compared to W_{t-1}, and the denominator represents the size of the vocabulary that belongs to W_t.

4.1.4 Event Word Extractor

After identifying a time window as an event window, the event word extractor facilitates the extraction of words (tokens) in that window that are related to the occurred events. Considering the ability of events to make changes to the textual corpus, this component marks all the tokens in an event window W_t, which showed cluster changes compared to its previous windows W_{t-1} as event words. We use the matrices generated during event window identification (Algorithm 1) for this task to maintain the efficiency of our approach. All token pairs in *diffMatrix* ($matrix_t - matrix_{t-1}$) with value (i.e. *DL similarity* change) above 0 indicate the pairs which had cluster changes over time. We consider the unique set of tokens belonging to the changed word pairs as the event words. Since we use the pre-processed vocabulary at t $vocab_t$ as the common vocabulary between the consecutive windows W_{t-1} and W_t to generate the similarity matrices $matrix_{t-1}$ and $matrix_t$ during the cluster change calculation, this approach also identifies the newly added tokens to W_t as tokens with cluster changes over W_{t-1} to W_t, returning them as event words.

4.1.5 Computational Complexity

Word embedding learner and event window identifier are the most computationally complex components available in Embed2Detect architecture. The complexity of stream chunker and event word extractor is negligible compared to them. Therefore, we only consider word embedding learner and event window identifier for time and space complexity calculations.

The training complexity of Skip-gram architecture is proportional to $C \times (D + D \times \log N)$, where C is the maximum distance of the words, D is the dimensionality of vectors and N is the size of vocabulary (Mikolov et al., 2013a). Under event window identifier, there are two complex subcomponents, clustering and similarity matrix generation. For N sized vocabulary, the time complexity for HAC algorithm is $O(N^2 \log N)$ (Manning et al., 2008) and for matrix generation is $O(N^2)$. By combining all these chained components, the time complexity of Embed2Detect can be calculated as $O(CD \log N + N^2 \log N)$. For the used application, both C and D values are comparatively smaller than N. Therefore, the time complexity can be further simplified to $O(N^2 \log N)$.

Following the 3-layer architecture, the space requirement of a Skip-gram model is equivalent to $N \times D + D \times N$. The HAC algorithm and similarity matrix generation have a space complexity of $O(N^2)$. Considering all cost values, the total space complexity of Embed2Detect can be summarised as $O(DN + N^2)$. Using the same assumption mentioned above, this can be simplified to $O(N^2)$.

Based on the complexities computed above, the vocabulary size N has a high impact on the computational complexity of this approach. According to the recent reports, approximately 511,200 tweets per minute were recorded in 2019 (James, 2019). Thus, for a 30-minute time window, which is sufficiently

long to highlight the worst-case scenario, the total tweet count would be approximately 15M. Looking at available twitter-based word embedding models, the Word2Vec_Twitter model trained on 400M tweets has a vocabulary of 3M tokens (Godin et al., 2015) and another popularly used model, GloVe Twitter trained on 2B tweets, has a vocabulary of 1.2M tokens[2]. Focusing on the worst-case scenario, if there were 3M vocabulary for 400M tweets, N could be approximated to 0.1M (100,000) for 15M tweets. Since our approach is targeted in processing a filtered data stream specific to a particular domain, N should be notably smaller than this value approximation (0.1M) in a real scenario. Further, the size of vocabulary can be controlled using the frequency threshold (β) mentioned in Section 4.1.3. Based on these findings, we can confirm the appropriateness of Embed2Detect for real-time event detection.

4.2 Experimental Study

This section summarises the experiments we conducted for event window identification using Embed2Detect on two recent social media datasets from two diverse domains, sports and politics, which are introduced in Section 3.2. To evaluate the results, we used time-based metrics described in Section 3.3. Since we do not identify event clusters at the coarse-grained level, all event words are considered as a single cluster and allowed to match with multiple GT events. We implemented Embed2Detect in Python and released the implementation on GitHub[3]. Initially, we analysed the impact of aggregation methods (Section 4.2.1) and text pre-processing techniques (Section 4.2.2) on Embed2Detect's performance. Then, an analysis of parameter sensitivity was

[2]GloVe pre-trained model details are available on https://nlp.stanford.edu/projects/glove/

[3]Embed2Detect implementation is publicly available on https://github.com/hhansi/embed2detect

conducted, and its findings were reported along with a discussion on heuristics behind parameter selection (Section 4.2.3). To compare the overall performance of Embed2Detect, we used three recently proposed event detection methods from different competitive areas, and the obtained results are summarised in Section 4.2.4. Furthermore, a comprehensive evaluation on the scalability of Embed2Detect is available in Section 4.2.5. Finally, we conducted some experiments to suggest possible extensions to other word embedding models and the obtained results and suggestions are summarised in Section 4.2.6. We used a Common KVM CPU @ 2.40GHz with 16GB RAM for all experiments.

4.2.1 Aggregation Methods

As described in Section 4.1.3, to measure the temporal data change between two consecutive time windows, Embed2Detect needs to aggregate the values computed using cluster change calculation (Section 4.1.3.1) and vocabulary change calculation (Section 4.1.3.3). For this aggregation, we experimented the techniques: average and maximum considering their simplicity and common usage. The obtained results are summarised in Table 4.1.

TABLE 4.1: Evaluation results: Time Window Recall (R), Precision (P) and F1 for different aggregation methods.

Aggregation Method	**MUNLIV**			**BrexitVote**		
	R	**P**	**F1**	**R**	**P**	**F1**
Average	0.6957	0.6154	0.6531	1.0000	0.7273	0.8421
Maximum	0.6522	0.6522	0.6522	1.0000	0.8000	0.8889

According to the results, for the MUNLIV dataset, there is a slight change in F1 between average and maximum calculations. However, we can see balanced values for both recall and precision with maximum aggregation. In

BrexitVote, there is a clear change in F1 with a higher value, using the maximum calculation. Based on these observations, we use maximum calculation as the default aggregation method of Embed2Detect.

4.2.2 Text Pre-processing

Even though pre-processing steps help to improve effectiveness, they can strongly restrict the method's generalisability across different domains and languages. For example, if any language-specific technique is used to pre-process the data, it will limit the method's scope only to that particular language. Therefore, we believe it is worth experimenting with the impact of text pre-processing on Embed2Detect. To maintain the simplicity of our method, we only suggest two pre-processing techniques, the removal of punctuation marks and stopwords. The evaluation results obtained with different configurations of these techniques are reported in Table 4.2. We only used cluster change calculation for these experiments because it has a high influence from token changes.

TABLE 4.2: Evaluation results: Time Window Recall (R), Precision (P) and F1 with different pre-processing techniques.

Pre-processing Steps	**MUNLIV**			**BrexitVote**		
	R	**P**	**F1**	**R**	**P**	**F1**
None	0.8261	0.4634	0.5938	1.0000	0.8000	0.8889
Punctuation removal	0.9130	0.4565	0.6087	1.0000	0.7273	0.8421
Punctuation and stopword removal	0.6957	0.5517	0.6154	1.0000	0.8000	0.8889

According to the obtained results, the tokens without punctuation and stopwords receive the highest F1 values for both sport and political datasets. Even though there is an improvement in the performance with pre-processing,

these results show that we can also obtain good measures without pre-processing. This ability will be helpful in situations where we cannot integrate direct pre-processing mechanisms such as identifying events in low-resource language or multilingual datasets. However, we used the tokens without punctuation and stopwords to conduct the following experiments because both datasets used in this research are mainly written in English.

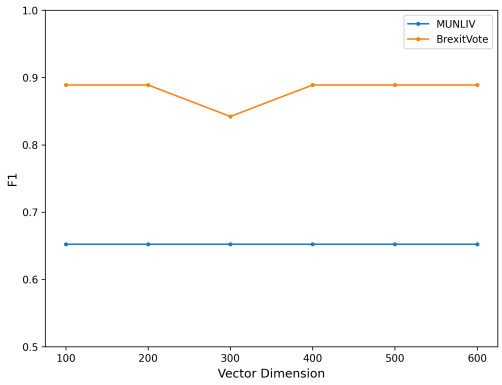
4.2.3 Parameter Sensitivity Analysis

Embed2Detect requires hyper-parameters during word embedding learning and event window identification. These parameters should mainly be picked considering the characteristics of the targeted domain and user preferences on resulting events. This section describes the impact of different hyper-parameter settings and heuristics behind their selections.

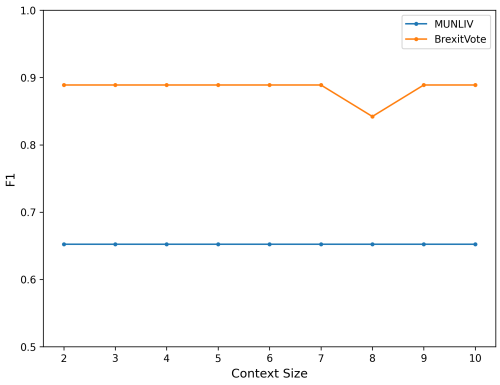
4.2.3.1 Word Embedding Learning

Word embedding learning requires three hyper-parameters: minimum word count, context size and vector dimension. Given a minimum word count, the learning phase ignores all tokens with less total frequency than the count. The context size defines the number of words around the word of interest to consider during the learning process. Vector dimension represents the number of neurons in the hidden layer, which also will be used as the dimensionality of word embeddings. Considering the limited amount of data available in a time window, we fixed the minimum word count to 1. Nevertheless, we analysed how the accuracy and efficiency of event detection vary with different vector dimensions and context sizes. To evaluate the accuracy, Time Window F1 was used, and results obtained for both datasets are visualised in Figure 4.3. According to the results, there is no significant change in F1 with varying vector

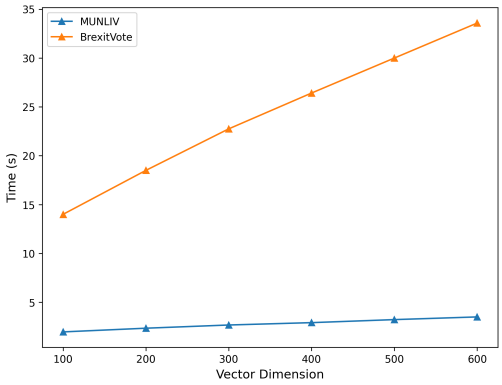
dimensions and context sizes. But, there is a gradual increase in execution time when both hyper-parameter values are increased.



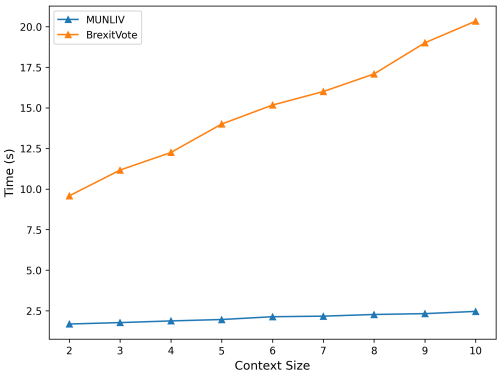
(A) F1 with different vector dimensions (with context size=5)



(B) F1 with different context sizes (with vector dimension=100)



(C) Time taken with different vector dimensions (with context size=5)



(D) Time taken with different context sizes (with vector dimension=100)

FIGURE 4.3: Analysis of Time Window F1 and execution time with different values for word embedding learning parameters; vector dimension and context size (Average time to learn word embeddings per window of both datasets is reported.)

Mikolov et al. (2013a) found that the accuracy of text-similarity predictions can be improved by training high-dimensional word vectors on a large amount of data. Similarly, a larger context size can result in higher accuracy in text-similarity predictions due to the provision of more training data (Li et al., 2017b). However, these effects are not notably captured with event detection (Figure 4.3). As a major reason for this, we can mention that event detection is

less sensitive to the syntactical and semantical structure of the text than text-similarity tasks. Particularly, it is less likely to have multiple meanings for a word (e.g. *'goal'*) within a time window even though it could appear in different contexts (e.g. context representing the winning team and the losing team). Thus, for event detection, it is sufficient to capture the general meaning of a word within a corpus rather than capturing its different contextual representations following fine granular text structures. However, it is crucial to capture such fine granular details in text for text-similarity tasks as they highly rely on underlying linguistics. Also, since we train separate models per time window, each model has comparatively small datasets to learn the embedding space. Therefore, even though the vector dimensions are increased, no sufficient data will be provided to optimise the model properly.

Following our analysis and previous research findings, we fix the vector dimension to 100 and the context size to 5. The decision on dimensionality is mainly influenced by the training data sizes and learning time to support real-time processing. The context size is chosen considering the requirement of providing sufficient knowledge for learning and the execution time.

4.2.3.2 Event Window Identification

Event window identification requires two hyper-parameters: change threshold (α) and frequency threshold (β) as described in Section 4.1.3. α is used to indicate the significance level of targeted events, and β is used as a frequency threshold to remove the outlier tokens. Both of these hyper-parameters need to be set by the user or domain expert, considering the characteristics associated with the selected domain or filtered data stream.

Embed2Detect identifies event windows using overall textual change between time windows. High overall changes indicate the occurrences of major

events, and low changes indicate minor events. α defines the threshold used for this event filtering. To provide a clearer insight, we plotted the variations of temporal overall change values of MUNLIV and BrexitVote datasets in Figure 4.4 and 4.5. Additionally, we plotted the total tweet count of each window in these graphs to highlight that the overall change-based measure is capable of identifying events that do not make a notable change to the total tweet count. As can be seen, the total tweet count changes can only capture major events which make bursts. For comparison purposes, tweet counts are scaled down using the min-max normalisation.

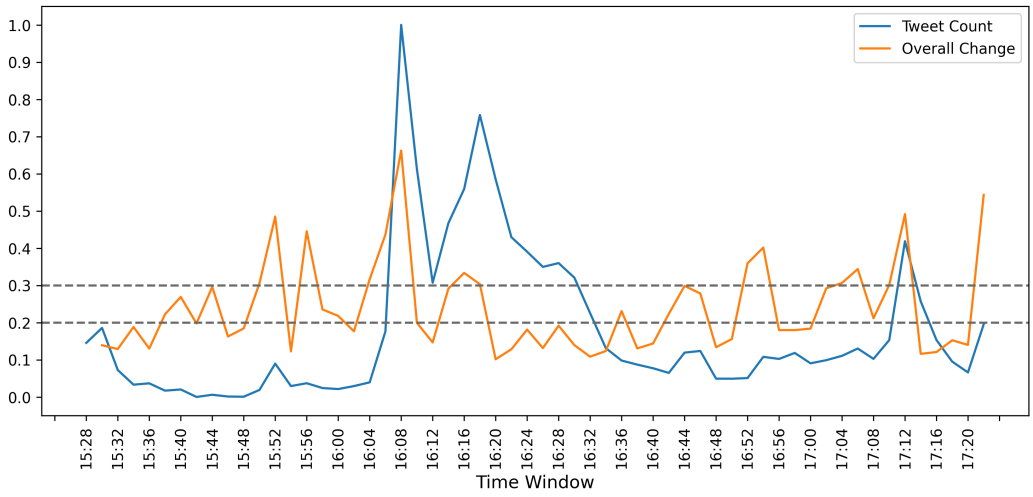


FIGURE 4.4: Overall change and tweet count variations over time windows - MUNLIV

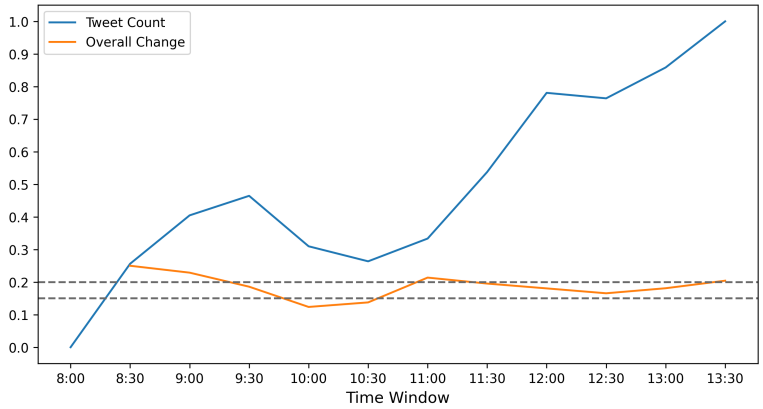


FIGURE 4.5: Overall change and tweet count variations over time windows - BrexitVote

As can be seen in Figure 4.4, MUNLIV data have more fluctuations on overall change due to the rapid evolution in the sports domain. To do a deep analysis, at time window 15:40, a change of 0.269 and at 16:06, a change of 0.436 is measured. Looking at other published media, at 15:40, a missed attempt, and at 16:06, a goal is reported. Compared to the goal, a missed attempt is a minor event in the sports domain, and overall change measure is capable of capturing that distinction successfully. Following this ability, α allows end-users to filter preferred events. For example, if α equals 0.2, both events missed attempt and goal will be captured by Embed2Detect. However, if the α value is increased to 0.3, only the goal will be captured among those two events. There was no high number of fluctuations for the BrexitVote dataset (Figure 4.5) because the political domain has a comparatively slow evolution than the sports domain. Unlike with the MUNLIV dataset, this effect limits the overall change values to a small range while increasing the sensitivity of α. Thus, a slight variation of α (e.g. from 0.15 to 0.2) changes the capturing events.

These analyses indicate that it is infeasible to define a standard α value for different domains as well as for a particular domain. This value needs to be picked for different domains, mainly considering the data evolution. Within a specific domain also, α may need to be varied according to personal preferences on event importance. However, it can be chosen simply by using the domain knowledge and analysing a few past time windows.

In addition to α, Embed2Detect requires β threshold to remove outlier tokens. All the tokens with less frequency than β, such as misspelt and uncommon words, will be removed by the β threshold. The impact of different β values on the overall change measure is illustrated in Figure 4.6. As can be seen in this plot, some event windows can be missed (e.g. time window 15:40 will not be identified with $\beta = 80$) with high β values. Also, high β could unnecessarily increase the overall change of some windows (e.g. overall change increase

at time windows 16:40 and 17:06). The main reason for these behaviours is the removal of tokens that are related to events or background discussions with a high β value. Thus, the effectiveness of event detection decreases with increasing β values, as shown in Figure 4.7.

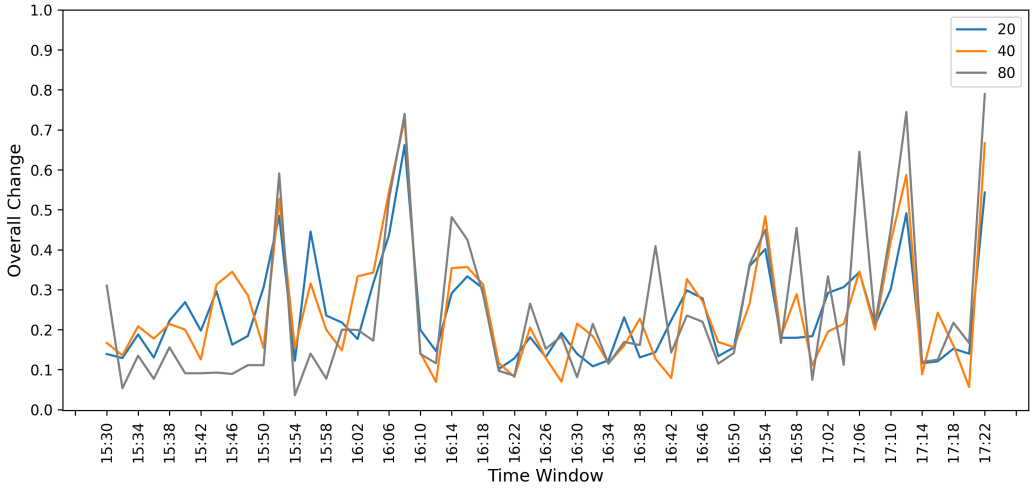


FIGURE 4.6: Analysis on impact by different β values on overall temporal change

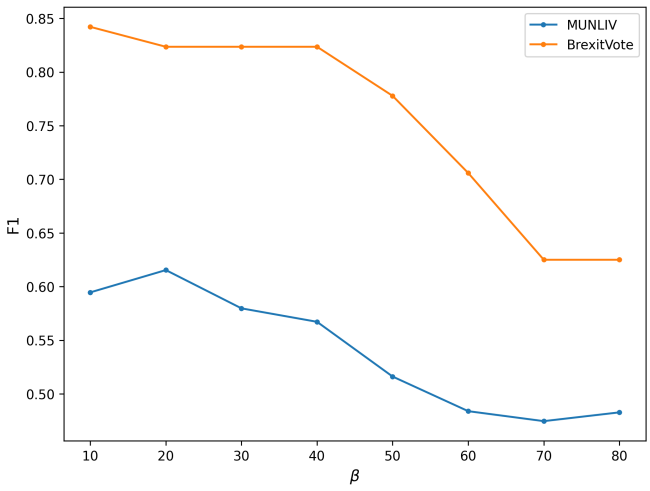


FIGURE 4.7: Analysis on F1 with different β values (with α=0.14)

Following the analyses of β, it is important to ensure that the selected β value is sufficiently large only to remove outliers. Analysing the sport and political datasets, values less than or equal to 20 are appropriate for β. However, similar to α, β also highly depends on domain-specific characteristics such as

word usage and audience. Thus, we believe that β is also a hyper-parameter that needs to be controlled by domain experts for effective event identification.

4.2.4 Overall Performance

In this section, we report the overall performance of Embed2Detect, comparing it with several recently proposed methods. Since there is no specific dataset to evaluate event detection performance, available methods cannot be directly compared with each other to pick the best baseline method. Therefore, considering the requirements of event detection (Section 1.1.4) and available competitive areas (Section 2.2), we selected three methods as baselines for comparisons. We mainly focused on methods' accuracy, efficiency, and expandability during this selection. We also covered different competitive areas, which can be summarised as social aspect, word acceleration over frequency, unsupervised learning (tensor decomposition and clustering), and segments over uni-grams to strengthen the baselines. All of these methods process the whole textual data in streams without considering only some keywords (e.g. hashtags) to identify temporal events, similar to our approach. More details on selected baseline methods are as follows.

- ***MABED*** (Guille and Favre, 2015): MABED uses anomalous variations in mention creation frequency and their magnitudes to detect events in an offline manner. Since user mentions are links added intentionally to connect a user with a discussion or dynamically during re-tweeting, their anomalies are used to incorporate the social aspect of Twitter into event detection. Given a data stream, MABED returns event spans with words. Since we targeted identifying event windows (i.e. occurrences of events) in this research, for comparisons, we assigned events detected by this method into time windows based on the spans' start times. If a window

got at least one event, it is marked as an event window, and the non-duplicate set of recognised event words is considered as the event words of that window.

- ***TopicSketch*** (Xie et al., 2016): TopicSketch focuses on a sketch-based topic model with tensor decomposition to detect events. It uses word acceleration-based measures for building data sketches to provide the ability to differentiate bursty topics (events) from general topics like car, food, or music. Since events have the ability to encourage people to discuss them intensively, acceleration is proposed with this method as a good measure over frequency for event detection. Given a data stream, TopicSketch returns event detected times and corresponding keywords. We assigned them to time windows based on detection times (similar to the scenario with MABED) to recognise event windows for the comparison.

- ***SEDTWik*** (Morabia et al., 2019): SEDTWik is an extension to Twevent (Li et al., 2012) and targets identifying the text of events in a given corpus with their newsworthiness. This method uses text statistics and user diversity-based measures to calculate the burstiness of segments and clusters bursty segments to identify events. Mainly, text segments are focused on in this research because they are more meaningful and specific than uni-grams. Wikipedia page titles are used as a semantic resource during segment extraction to preserve the informativeness of identified segments. For our experiments, we applied this method to each time window and marked a window as an event window if it has at least one event with high newsworthiness than the defined threshold. All keywords of the recognised events within a window are considered as the event words of that window.

The results obtained for MUNLIV and BrexitVote datasets using Embed2Detect and baseline methods are summarised in Table 4.3 and 4.4. We used time- and keyword-based metrics (Section 3.3) to measure the accuracy of methods. We involved efficient computation techniques with parallel processing to facilitate fast execution of Embed2Detect and used parallel processing with eight workers to measure the reported execution times. For the baseline methods, we used available implementations, which targeted sequential processing. Both the total time taken to process the whole data stream and the average time taken per time window are reported.

TABLE 4.3: Performance comparison of Embed2Detect with baseline methods for MUNLIV. TW stands for Time Window and Average Time indicates the processing time for a 2-minute time window. The best result is in bold and the best baseline result is in italics.

Method	**TW Recall**	**TW Precision**	**TW F1**	**Keyword Recall**	**Execution Time (s)**	
					Total	Average
Embed2Detect	**0.6522**	**0.6522**	**0.6522**	**0.8431**	202	3.5439
MABED	0.4783	0.1930	0.2750	0.3478	**168**	**2.9474**
TopicSketch	0.6087	0.2456	0.3500	0.4000	25492	447.2281
SEDTWik	*0.6522*	*0.2679*	*0.3797*	*0.3857*	1290	22.6316

TABLE 4.4: Performance comparison of Embed2Detect with baseline methods for BrexitVote. TW stands for Time Window and Average Time indicates the processing time for a 30-minute time window. The best result is in bold and the best baseline result is in italics.

Method	**TW Recall**	**TW Precision**	**TW F1**	**Keyword Recall**	**Execution Time (s)**	
					Total	Average
Embed2Detect	**1.0000**	**0.8000**	**0.8889**	**0.9846**	**310**	**28.1818**
MABED	0.6250	0.4545	0.5263	0.4032	532	48.3636
TopicSketch	0.5000	0.3636	0.4211	0.2540	15887	1444.2727
SEDTWik	*0.7500*	*0.5000*	*0.6000*	*0.4262*	702	63.8182

According to the results, Embed2Detect outperformed all the baseline methods returning 27.25% and 28.89% higher Time Window F1 values than

the best baseline method for MUNLIV and BrexitVote datasets, respectively. These results prove that our method has the ability to detect event windows more accurately in diverse domains, specifically, sports and politics, compared to the available methods. Also, the comparatively high Keyword Recall values resulted by Embed2Detect further emphasise the importance of involving syntax and semantics for event detection. According to the Time Window Recall and Precision measures, all methods returned higher recall than precision. When preparing the GT events based on published/news reports, there is a possibility to miss some important events which are only discussed within the social media platform (Aiello et al., 2013; Morabia et al., 2019). Due to that, some actual events can be labelled as false positives, reducing the precision value. Considering Time Window Recall, most methods (except TopicSketch) resulted in higher values with BrexitVote than the MUNLIV dataset. Comparing the GT of the two datasets, BrexitVote has 72.73% of event occurred time windows while MUNLIV has only 40.35%. Due to this bias, high recall can result with the BrexitVote dataset. Theoretically, such bias is captured because of the political domain's slow evolution and less dynamicity.

Following the execution times, for MUNLIV, MABED took 168 seconds, and Embed2Detect took 34 seconds more than MABED. However, for BrexitVote, Embed2Detect completed the execution in 310 seconds – 222 seconds faster than MABED. In terms of average execution time per window, Embed2Detect took nearly 3.54 seconds to process a 2-minute window in the MUNLIV dataset ($\approx 1,724$ tweets) and 28.18 seconds to process a 30-minute window in the BrexitVote dataset ($\approx 14,542$ tweets). This time measures further prove that Embed2Detect is sufficiently fast for (near) real-time event detection. A detailed analysis of intermediate processing time of Embed2Detect with sequential and parallel processing is reported in Appendix A.1.

TABLE 4.5: Best hyper-parameter settings

Method	**Hyper-parameters**	
	MUNLIV	**BrexitVote**
Embed2Detect	$\alpha = 0.23$	$\alpha = 0.16$
	$\beta = 20$	$\beta = 10$
MABED	$k = 150$	$k = 150$
	$p = 20$	$p = 20$
	$\theta = 0.7$	$\theta = 0.6$
	$\sigma = 0.5$	$\sigma = 0.5$
TopicSketch	*detection threshold*$= 60$	*detection threshold*$= 35$
	$B = 5000$	$B = 5000$
SEDTWik	$M = 2$	$M = 2$
	$k = 6$	$k = 6$
	$\tau = 0.7$	$\tau = 0.2$

Hyper-parameters: Since all methods' performance depends on their hyper-parameter configurations, we used a common strategy to identify optimal settings to generate comparable results. We evaluated all possible hyper-parameter settings for each method and selected the best setting with the highest Time Window F1 for comparisons. Table 4.5 summarises the optimal hyper-parameter values obtained. For Embed2Detect, we identified all possible values for α and β, and experimented with all of their combinations to get the best results. For MABED, we optimised the number of events (k), the maximum number of words describing each event (p), weight threshold for selecting relevant words (θ) and overlap threshold (σ). For k and p, starting from a low value, we kept increasing them gradually until the maximum F1, which reduces with further increasing parameter values, is reached. Similarly, for θ and σ, we experimented with the values around the original values reported in initial experiments (Guille and Favre, 2015). For TopicSketch, we decided to optimise only the most critical hyper-parameter due to the high time consumption of this method. Thus, while keeping default values for other

parameters, including the bucket size (B), we gradually increased values for *detection threshold* to obtain the best F1 value. For SEDTWik, we optimised the number of subwindows (M), number of cluster neighbours (k) and newsworthiness threshold (τ). Different values for M are picked considering the time window lengths assigned to each dataset. For k and τ, gradually increasing values were tested to obtain the highest F1 starting from a low value.

4.2.5 Scalability Evaluation

Processing data belonging to a time window within a sufficiently short period is a crucial requirement of Embed2Detect. Thus, we measured the execution time per window with increasing data size to evaluate the scalability of Embed2Detect. The obtained results are plotted in Figure 4.8. As the data size per window (e.g. 1-minute window), 5,000 to 25,000 tweets were considered. Focusing on a filtered data stream, the upper limit of 25,000 tweets per minute is reasonable to depict the real scenario.

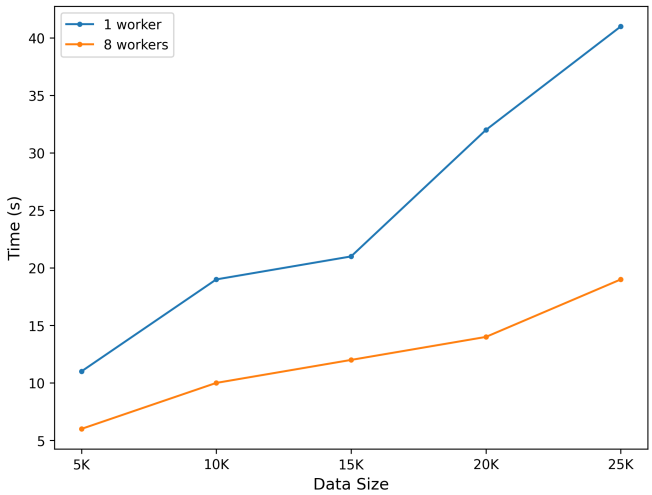


FIGURE 4.8: Execution time on different data sizes with sequential and parallel processing of Embed2Detect

According to the results, the sequential version of Embed2Detect took nearly 10 seconds to process 5,000 tweets, and the time was increased to 41

seconds to process 25,000 tweets. However, the parallel version with eight workers reduced the processing time to 6 seconds for 5,000 tweets and 19 seconds for 25,000 tweets. Also, we noticed that for both implementations, sequential and parallel execution times grow linearly with data size. Due to the linear growth of execution time, we can further guarantee that Embed2Detect can also handle data bursts effectively.

4.2.6 Extension to Other Word Embeddings

Different word embedding models can be used with Embed2Detect instead of our default selection, Skip-gram. Also, we designed the word embedding learner as a separate component to support the easy integration of different embedding models. However, it is important to consider the linguistic coverage, learning time and associated complexities while selecting an embedding model to satisfy the accuracy and efficiency required for real-time event detection. In this section, we discuss the appropriateness of different recently-proposed architectures for word embedding generation in Embed2Detect.

For this analysis, we used fastText, BERT and DistilBERT models. FastText is an improved version of the Skip-gram model, which considers subword information while learning word representations (Bojanowski et al., 2017). Both BERT and DistilBERT are transformer-based models, which are further discussed in Section 6.4.1. According to the recent advances in NLP, transformer-based models gained success in many areas such as language generation (Devlin et al., 2019), question answering (Yang et al., 2019b), named entity recognition (Liang et al., 2020) and machine translation (Ranasinghe et al., 2020). Among different models, BERT (Devlin et al., 2019) attracted a wide attention from the community recently. This model is designed to train from unlabelled text using masked language modelling (MLM) and next sentence prediction

(NSP) objectives and fine-tune for a downstream task as a solution for the high data requirement by deep neural networks. DistilBERT is a distilled version of BERT which is light and fast (Sanh et al., 2019).

TABLE 4.6: Time taken to learn embeddings by different architectures

Time Window Length	**Tweet Count**	**Embedding Learning Time (s)**			
		Skip-gram	fastText	BERT	DistilBERT
2 min.(120 s)	1705	1	12	646	433
30 min.(1800 s)	20133	18	41	21442	11699

Initially, time taken by different architectures to learn word embeddings is measured, and the obtained results are summarised in Table 4.6. Both Skip-gram and fastText models were trained from scratch using Twitter data as suggested by this research. Following the idea presented with transformers, for BERT and DistilBERT, we retrained available models for the MLM objective using our data. As the pre-trained BERT model, *bert-base-uncased* and DistilBERT model, *distilbert-base-uncased* released with HuggingFace's Transformers library[4] (Wolf et al., 2020) were selected. According to the obtained results, classic word embedding models (e.g. Skip-gram and fastText) learn the representations faster than transformer-based models (e.g. BERT and DistilBERT).

Comparing fastText and Skip-gram, fastText took more time because it processes subword information. But, the incorporation of subwords allows this model to capture connections between modified words. For example, consider the following goal-related words found within the top 20 words with high cluster change during a goal score:

Skip-gram- *goal, goalll, rashyyy, scores*

fastText- *goalll, goooaaalll, rashford, rashyyy, @marcusrashford, scored, scores*

[4]HuggingFace models are available on `https://huggingface.co/models`

As can be seen, fastText captures more modified words than Skip-gram. However, we could not run a complete evaluation using fastText embeddings during this research because it requires a manual process since GT keywords only contain the words with conventional (correct) spelling.

Transformer-based models took more time to learn embeddings due to their complex architecture and contextual learning. DistilBERT is faster than BERT; however, its learning time is also not fast enough for real-time processing as it exceeds the tweet generation time. For example, to learn from tweets posted during a 2-minute time window, it took approximately 7.2 minutes. If this model can be further distilled, there is a possibility to achieve the required efficiency to become suitable for real-time processing. However, further distillation can reduce the language understanding capability of the model as there is a 3% reduction in DistilBERT compared to BERT (Sanh et al., 2019).

According to recent literature, transformer-based models performed well on many NLP-related tasks, mainly due to their ability in capturing contextual word senses. BERT is capable of generating different embeddings for the same word depending on its surrounding context. In other words, the main idea behind BERT is capturing spatial changes of words. From the perspective of processing formally written natural language, this is a very useful feature. However, language is mostly informal in social media, and for event detection using social media text, temporal changes of words need to be more focused. Also, considering a particular time window of a filtered data stream, it is rarely possible to have a word with two totally different contextual meanings. Therefore, the context awareness associated with BERT is not much useful for event detection.

Further, contextualised word embeddings could incorporate an additional complexity to the event detection method. For example, during a goal-scoring of a football match, the word *'goal'* will be expressed by the audience of the

winning team and losing team differently. Even though the surrounding contexts are varied, the meaning of the word *'goal'* targeted by event detection is a constant. For such a scenario, BERT will return different embeddings for *'goal'* as illustrated in Figure 4.9. Having multiple embeddings for monosemy words can confuse the clusters and exponentially increase the computational complexity of the method. To overcome these issues, multiple embeddings of a word can be combined using an aggregation method, but it breaks the main objective of contextualised word embeddings. Therefore, we believe it is less appropriate to aggregate context-aware embeddings. Following these findings, we can further emphasise that contextualised word embedding models such as BERT are less suitable to be used with Embed2Detect, due to their complexities which are not useful for event detection.

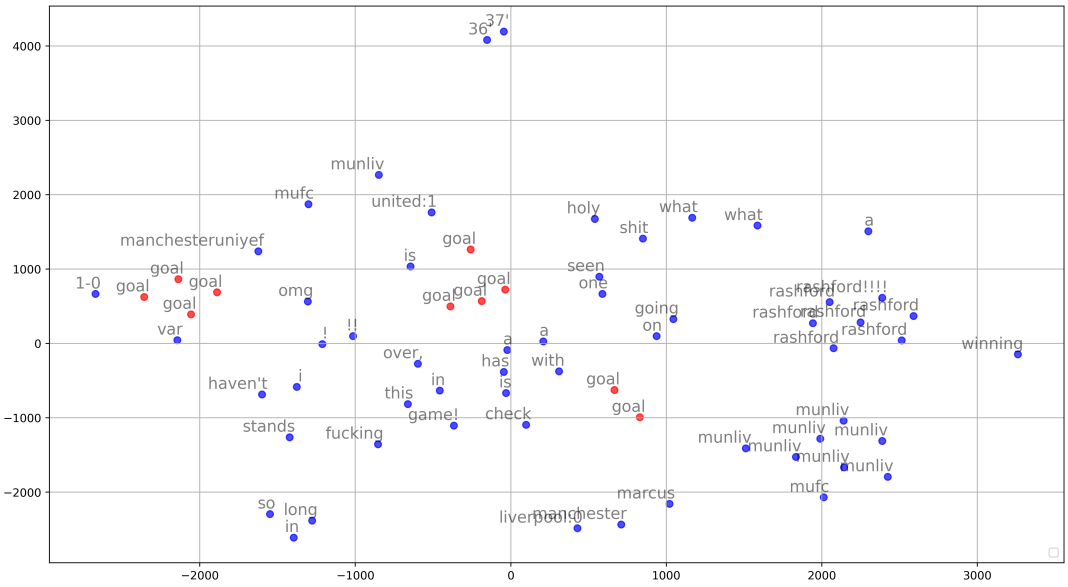


FIGURE 4.9: t-SNE visualisation of sample word embeddings obtained by a *bert-base-uncased* model retrained on first goal scored time window of MUNLIV (16:06-16:08)

4.3 Conclusions

In this chapter, we proposed *Embed2Detect*, a novel method to identify event occurred time windows in social media data streams. Embed2Detect mainly combines the characteristics in word embeddings and dendrograms and also introduces a new text similarity measure named *Dendrogram Level (DL) Similarity* to capture cluster similarity of tokens. Usage of self-learned word embeddings allows our method to capture linguistical features (syntax and semantics) in the targeted data and makes it expandable to any domain, language or platform, unlike the majority of available methods, which are limited to specific languages (e.g. English) and platforms (e.g. Twitter). Also, the inclusion of semantics allows understanding of the relationships between tokens. Since social media text contains different words and word sequences which describe the same idea due to vast and diverse user bases, knowing the relationships between words, differently described similar ideas and their connections can be extracted effectively. Thus, our approach is also capable of reducing the information loss experienced in previous approaches due to the lack of semantic involvement.

According to the experiment results, Embed2Detect notably outperformed several recently proposed methods from different competitive areas. We involved various metrics, Time Window Recall, Precision, F1 and Keyword Recall, to evaluate the methods' accuracy comprehensively. Also, we considered data from two diverse domains (i.e. sports and politics), which have different word usages, audiences and evolution rates, targeting to assess the expandability of methods. For both domains, Embed2Detect returned promising results with comparatively high F1 measures (>65%), emphasising its accuracy

and expandability. Also, we focused on methods' efficiency during the evaluations since event detection in social media is a time-critical operation. Embed2Detect completed processing both datasets within short periods, proving its appropriateness for (near) real-time processing. Furthermore, our analyses confirmed Embed2Detect's ability to handle increasing data volumes successfully, demonstrating its scalability.

More advanced embedding models can be used with the word embedding learner as extensions to Embed2Detect. However, considering the learning time and associated complexities, it is more suitable to use classic embedding models such as Skip-gram than advanced models such as BERT to preserve the method efficiency. In future work, we hope to analyse further the impact of subword and character-based classic word embedding models that can capture the connections between informal or modified text and their formal versions. Such an approach would be helpful in understanding the informal text, which is common in the context of social media. Also, it would be interesting to analyse the method's performance on multilingual data to see its ability to detect events in such a setting. Since social media allow posting in different languages, information from different people groups can be combined to detect events with multilingual processing.

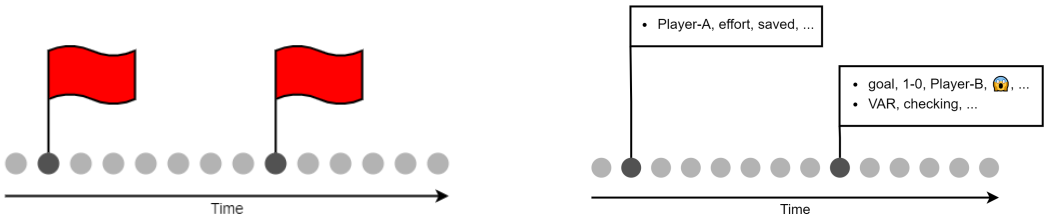
Following the details on coarse-grained level event detection in social media in this chapter, we report our approach for fine-grained level event detection in the next chapter. At the coarse-grained level, we mainly targeted notifying users about events by detecting event windows using Embed2Detect. At the fine-grained level, we target extracting textual details of co-occurred events within event windows to provide more detailed quick updates to users, as described in Chapter 5.

Chapter 5

WhatsUp: Fine-grained Level – Co-occurring Event Identification

In Chapter 4, we proposed *Embed2Detect*, a novel method to notify users about event occurrences in time by identifying event occurred time windows or event windows, covering the coarse-grained level of social media event detection. Upon the notification, users can take necessary actions, such as manual data analysis, to acquire the required event information. However, to automate the complete event detection process, it is important to develop approaches for fine-grained level detection, which extracts event-described text segments following the coarse-grained level detection. As the fine-grained level of social media event detection, we target identifying the text of co-occurred events in event windows. Since multiple events can happen during an event window, co-occurred events are focused. Following this idea, fulfilling the fine-grained level requirement of social media event detection, in this chapter, we propose a novel method named *WhatsUp* to identify the text of co-occurred events in event windows, allowing the concise and quick acquisition of information at events. The outputs targeted by Embed2Detect and WhatsUp are illustrated in Figure 5.1 to clearly contrast the coarse- and fine-grained level event detection in social media focused by this research. The findings reported in this chapter

have been published in (Hettiarachchi et al., 2023b).



(A) Embed2Detect's output (B) WhatsUp's output

FIGURE 5.1: Embed2Detect vs WhatsUp

Considering the limitations in available approaches (Section 2.2.5), we mainly targeted statistics and linguistics in the text to extract event details while designing our approach. To identify event time, we utilised the idea proposed with Embed2Detect, considering its promising results (Chapter 4). In summary, Embed2Detect uses temporal variations of self-learned word embeddings and hierarchical clusters to detect event windows. Following this idea, we propose different strategies which improve its performance along with WhatsUp, including an advanced text similarity measure named *Local Dendrogram Level (LDL) Similarity* and a similarity change calculation technique named *Positive Similarity Change*. To extract fine-grained event text, we propose a novel clustering algorithm that identifies the co-occurred events in each event window. For clustering, we involve linguistical features captured using self-learned word embeddings and their hierarchical relationships in dendrograms and statistical features captured using token frequency-based measures. In summary, WhatsUp considers all the important features in textual data: statistics and linguistics (syntax and semantics) to detect event time and text of co-occurred events effectively. Also, using self-learned word embeddings allows capturing the linguistics specific to the underlying corpus. Further, we preserve the expandability of our approach by only using unsupervised learning techniques independent of domain, language and platform.

We evaluated WhatsUp in two diverse domains using a newly designed comprehensive set of metrics. According to the results, our method outperformed several recently proposed methods proving its effectiveness.

To the best of our knowledge, WhatsUp is the first method that uses underlying linguistics to detect both temporal and textual event details together. In summary, the main contributions of this chapter are as follows.

1. We propose a novel method named *WhatsUp* which detects both temporal (event occurred time windows) and fine-grained textual (co-occurred event word groups within event windows) event details from social media data streams in (near) real-time, considering statistics and linguistics of underlying data.

2. We introduce a localised version of Dendrogram Level (DL) Similarity named *Local Dendrogram Level (LDL) Similarity*, a similarity change calculation technique named *Positive Similarity Change* and a novel clustering approach along with our approach.

3. We utilise self-learned word embeddings and unsupervised learning techniques in WhatsUp to develop a more robust event resolution approach, overcoming the limitations in previous methods, specifically, the less semantic involvement, inability to capture linguistics specific to the underlying data and less expandability to different domains, languages and platforms.

4. We evaluate WhatsUp on recent real datasets from two diverse domains, sports and politics, using a comprehensive set of metrics that we designed to assess both temporal and textual details of detected events and compare the performance with several recently proposed competitive methods emphasising its effectiveness.

5. We release the implementation of WhatsUp as an open-source project to support applications and research in the area of social media event detection[1].

The rest of this chapter is organised as follows. Section 5.1 introduces the proposed approach: WhatsUp for fine-grained event resolution in social media. Section 5.2 comprehensively describes the conducted experiments and obtained results. Finally, Section 5.3 concludes the chapter with a discussion and future research directions.

5.1 Methodology - WhatsUp

As *WhatsUp*, we propose a novel event detection approach to recognise both temporal (event occurred time windows) and fine-grained textual (co-occurred event word groups within event windows) event details. Our approach has three main steps: (1) data processing, (2) event occurred time window identification and (3) event cluster detection as illustrated in Figure 5.2. The first step separates the data stream into time windows and per window, learns word embeddings and extracts statistical information. The next step identifies the event occurred time windows. If any window is identified as an event window, the final step detects co-occurred event clusters in that window. For both event window and cluster detection, we use temporal changes in linguistics (syntax and semantics) captured by self-learned word embeddings and their hierarchical relationships and statistics. To the best of our knowledge, both linguistical and statistical features have not been involved in previous research to identify event time and text of co-occurred events together. Each step is further described in Sections 5.1.1-5.1.3. To explain the concepts used to develop

[1]WhatsUp implementation is publicly available on https://github.com/HHansi/WhatsUp

the methodology, we use sample data from the MUNLIV dataset (Section 3.2) only considering the simplicity and cohesion of events in this domain.

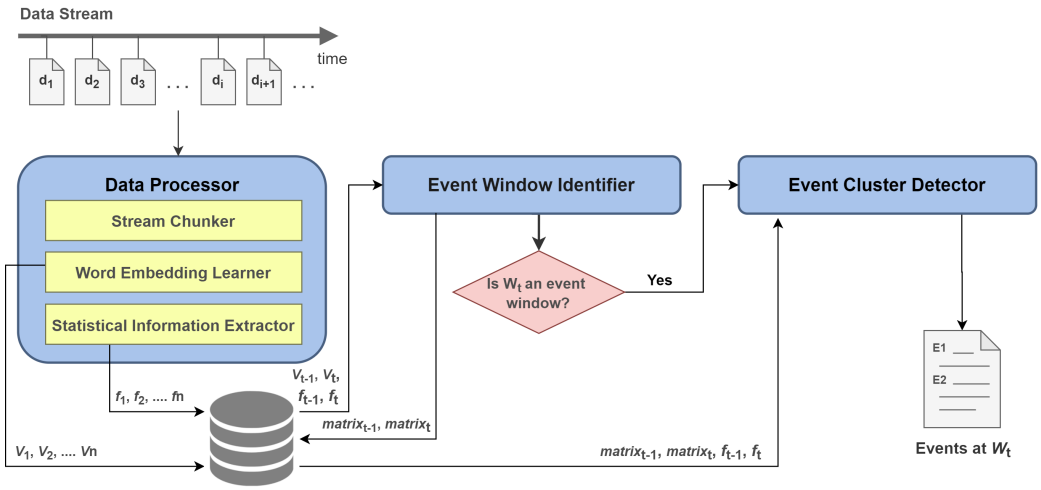


FIGURE 5.2: Overview of the proposed method – WhatsUp

5.1.1 Data Processing

Under data processing, we initially separate the incoming data stream into windows of non-overlapping fixed periods using stream chunker. Following the findings during Embed2Detect developments (Section 4.1.1), we use a sliding window model with a fixed time frame for this data separation. Especially, the ability to adjust the time window length in this model is advantageous for social media processing as it can be adjusted depending on the events' evolution in the targeted domain.

Then, we extract each window's linguistical and statistical details to use with event window identification and cluster detection. To capture linguistics, embedding models are trained per window using the data in that window. This training allows capturing expressions and characteristics unique to the data in each window. We mainly focused on semantic accuracy and efficiency appropriate for real-time processing, while selecting an algorithm to

learn embeddings. Considering the semantic accuracy, transformer-based embeddings (e.g. BERT, XLM-R) showed improved performance in many NLP applications recently (Liang et al., 2020; Ranasinghe et al., 2020). However, they are not suitable for real-time processing due to the long training time and additional computational complexities introduced for event detection by contextual word senses, as we discussed in Section 4.2.6. Therefore, we decided to use Word2Vec models (Mikolov et al., 2013a) for WhatsUp also, considering their ability to learn effective embeddings faster. Among CBOW and Skip-gram architectures, we selected the Skip-gram model considering its high semantic accuracy (Section 3.4.1). However, we facilitate easy integration of any embedding model that fulfils the targeted requirements (i.e. good semantic accuracy and efficient learning) by developing the word embedding learner as a separate component. As statistical details, token frequencies at each time window are calculated.

5.1.2 Event Window Identification

We utilise the idea proposed with Embed2Detect system in Chapter 4 to identify event windows. Primarily, a time window W_t is marked as an event window if its temporal textual change compared to the previous time window W_{t-1} is above a predefined threshold (α). The final change is calculated by aggregating two measures based on the temporal variations in token clusters and vocabularies, capturing underlying linguistics and statistics. Any value between 0 and 1 can be picked for α, as normalised values are involved in change calculation, and it mainly defines the significance of the events targeted by the user. The higher the value, the more significance it has. The overall idea of event window identification is summarised in Algorithm 1 in Chapter 4. It

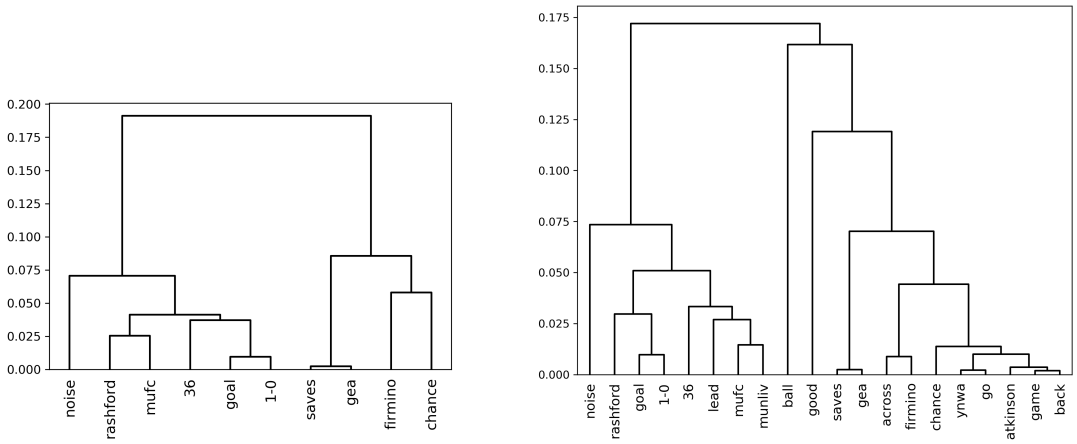
takes a chronological stream of time windows (at least two consecutive windows W_{t-1}, W_t) as the input and returns the windows which are recognised as event windows. Highlighting some limitations recognised in this approach, along with WhatsUp, we propose different strategies to calculate token cluster similarity, temporal similarity change and vocabulary change which improve the performance of event window identification. The proposed improvements to this algorithm are further described in Section 5.1.2.1-5.1.2.3.

Before moving into the calculations, the text needs to be pre-processed for effective results and efficient processing. Similar to Embed2Detect, under pre-processing, we mainly targeted removing the uninformative tokens using general procedures which are not strictly dependent on domains or languages. Punctuation marks and stopwords in the text are removed as uninformative tokens. Also, the tokens with low frequency than a predefined threshold (β) are removed as outlier tokens.

5.1.2.1 Cluster Similarity Calculation

To compute cluster change between consecutive time windows W_{t-1} and W_t, initially, we need to calculate the cluster similarity between tokens in each window. With Embed2Detect, we introduced a new measure named *Dendrogram Level (DL) similarity* to calculate cluster similarity. As the name depicts, this measure is mainly based on the structural relationships of a dendrogram, and we described more details in Section 4.1.3.2. To calculate DL similarities, we require to generate dendrograms per window. We applied hierarchical agglomerative clustering (HAC) to self-learned word embeddings of each window to generate the dendrograms as mentioned in Section 4.1.3.1.

The main limitation we noticed in DL similarity is it can be distorted when noise is added to the dendrogram. To illustrate this effect, we generated two



(A) Dendrogram 1 (B) Dendrogram 2

FIGURE 5.3: Sample dendrograms (y-coordinate denotes the cosine distance and x-coordinate denotes the selected words)

sample dendrograms on data during the first goal of MUNLIV using different word sets. For dendrogram 1 (Figure 5.3a), we only considered the keywords of events: Firmino's attempt saved by Gea and the first goal for MUFC by Rashford. To generate dendrogram 2 (Figure 5.3b), we used more additional words such as *'good'*, *'go'*, *'ynwa'*, *'game'* and *'back'* which are not event-related directly to depict a near-real scenario. As can be seen, the addition of non-relevant words changed the whole structure of the dendrogram, increasing the maximum number of levels which we use as the denominator while calculating DL similarity (Equation 4.3). In measures, dendrogram 1 has 6 (using path $r \rightarrow goal$) and dendrogram 2 has 10 (using path $r \rightarrow game$) maximum levels between root and leaf nodes. To see the impact on DL similarity by such structural changes to the dendrogram, we selected the word *'goal'* and paired it with the player name who scored the goal *'rashford'* and some other player name *'firmino'*. The DL similarity values calculated for both word pairs using dendrogram 1 and 2 are available in Table 5.1. Using dendrogram 1, we received a high similarity for the closer word pair (*'rashford'*-*'goal'*) as expected. However, the values calculated using dendrogram 2 have a low similarity for

that pair, even if the tokens are located very closely. This happened due to the increment of levels with the noise added to the dendrogram 2 by non-event words. In reality, it is infeasible to filter event-related words beforehand to generate less noisy dendrograms, considering that automated event detection is the aim of the methodology. To overcome this limitation, we propose a localised version of DL similarity named *Local DL (LDL) similarity*.

TABLE 5.1: DL similarity values calculated using Dendrogram 1 and 2 (Figure 5.3)

Word Pair	**DL Similarity**	
	Dendrogram 1	**Dendrogram 2**
rashford-goal	3/6 = 0.5	4/10 = 0.4
firmino-goal	1/6 = 0.167	1/10 = 0.1

Local Dendrogram Level (LDL) similarity: The main difference in LDL similarity compared to DL similarity is that it only considers the levels between the root and targeted word pair to calculate the denominator as in Equation 5.1. It prevents unnecessary normalisation of the value due to a rare long path in the dendrogram.

$$LDL\ Similarity_{(w_i,w_j)} = \frac{dl_{(w_i,w_j)}}{max(dl_{r\rightarrow x} : x \in \{w_i, w_j\}) + 1} \tag{5.1}$$

Using LDL similarity, we will receive the similarities in Table 5.2 for the above-considered word pairs. With dendrogram 1, we consider the path $r \rightarrow goal$ to calculate the denominator values of both pairs as it has the maximum number of levels among the targeted words. Similarly, with dendrogram 2, we consider the path $r \rightarrow goal$ for *'rashford'-'goal'* and $r \rightarrow firmino$ for *'firmino'-'goal'*. According to the obtained values, unlike DL similarity, LDL similarity is capable of returning high values for closely located word pairs using both clean

and noisy dendrograms, emphasising its effectiveness. Therefore we use LDL similarity to calculate the cluster similarity between words in this research.

TABLE 5.2: LDL similarity values calculated using Dendrogram 1 and 2 (Figure 5.3)

Word Pair	**LDL Similarity**	
	Dendrogram 1	**Dendrogram 2**
rashford-goal	3/6 = 0.5	4/6 = 0.667
firmino-goal	1/6 = 0.167	1/7 = 0.143

In summary, DL and LDL similarities compute normalised text similarity values within the range of 0–1, based on structural relationships of a dendrogram. Thus, these measures are relative to the dendrogram or underlying corpus rather than absolute, preserving symmetricity. They intend to result in a value closer to 1 if a word pair is similar (or closer) and a value closer to 0 if the pair is distant. However, DL similarity could fail to produce high values for closer words if the dendrogram is distorted with noise, and LDL similarity is proposed to overcome this limitation. Also, even though we only focus on similarities in this research following its requirements, both metrics can convert to distance measures, similar to cosine similarity, by taking $1 - (L)DL\ Sim$. This way, the number of non-shared levels between the word pair becomes prominent, resulting in the difference/distance between words. Contrary to similarity measures, values closer to 1 will represent high distances in distance calculations.

After calculating token similarities, we format them into a matrix to support efficient future computations. The similarity matrix is a square matrix of size $N \times N$ where N is the number of tokens in the vocabulary. Each cell in the matrix $matrix[i, j]$ holds the cluster similarity between tokens w_i and w_j (or $LDL\ sim._{(w_i, w_j)}$). To support matrix comparisons over consecutive time windows, we use one vocabulary to generate consecutive matrices $matrix_{t-1}$ and

$matrix_t$. Since the change at W_t needs to be calculated compared to the previous window W_{t-1} to detect the event occurrences, pre-processed vocabulary at t $vocab_t$ is considered as this common vocabulary during matrix generation. Following this idea, Embed2Detect uses common vocabularies (or $vocab_t$) to generate dendrograms in consecutive time windows. We use the term *relative dendrograms* to refer to such dendrograms.

Relative Dendrograms: To generate relative dendrograms for W_t and W_{t-1}, pre-processed vocabulary at t $vocab_t$ is considered as a common vocabulary. Dendrogram for W_t is generated using $vocab_t$ and word embeddings learned for W_t. Dendrogram for W_{t-1} is generated using $vocab_t$ and word embeddings learned for W_{t-1} ignoring the words in the vocabulary which are not found in the vector space V_{t-1}.

Usage of a common vocabulary can negatively affect the similarity values if $vocab_t$ contains tokens that have a very low frequency at W_{t-1}. If the embedding model has not seen sufficient occurrences, it may result in vectors that are not properly learned for those tokens. This could affect the dendrogram structure and result in incorrect similarity values. A few sample similarity values calculated using relative dendrograms on MUNLIV data are shown in Table 5.3. As illustrated in these examples, relative dendrograms return high similarities for word pairs which are rarely used during time windows. Even though the selected word pairs are emerging at t according to the ground truth, cluster similarity change between W_t and W_{t-1} cannot capture that state using the resulting negative ('gini'-'movement': -0.2807) and small positive ('goal'-'1-0': 0.1308) values. To overcome this issue, we propose to use non-relative dendrograms.

TABLE 5.3: Sample LDL similarity values using relative dendrograms

w_i	w_j	f_{t-1}: w_i-w_j	f_t: w_i-w_j	$LDL\ sim._{t-1}$	$LDL\ sim._t$
gini	movement	10-9	40-37	0.9474	0.6667
goal	1-0	16-2	339-244	0.7692	0.9000

Non-Relative Dendrograms: Non-relative dendrogram generation does not use a common vocabulary as relative dendrograms. Dendrograms for W_{t-1} and W_t will be generated using pre-processed vocabularies $vocab_{t-1}$ and $vocab_t$, respectively. By using the preprocessed vocabularies of each window, the involvement of low-frequent/rare words for dendrogram generation can be eliminated, overcoming the issue mentioned above with relative dendrograms. If non-relative dendrograms are used to calculate the similarities between the word pairs in Table 5.3, due to low frequency, 'movement' and '1-0' will not be included in $vocab_{t-1}$ and similarity between both word pairs at W_{t-1} will be zero. For W_t, the same similarities will be calculated, and cluster similarity change of both pairs will be large positive values ('gini'-'movement': 0.6667. 'goal'-'1-0': 0.9) capable of capturing the emerging state of words.

In summary, for WhatsUp, we use LDL similarity calculated on non-relative dendrograms to measure the cluster similarity of tokens overcoming some major limitations recognised with DL similarity and relative dendrograms.

5.1.2.2 Similarity Change Calculation

After calculating cluster similarities of tokens at each time window, the temporal change of similarities needs to be measured. The change calculation is mainly based on the *diffMatrix* which holds the token similarity differences between W_t and W_{t-1} ($matrix_t - matrix_{t-1}$). With Embed2Detect, we used the

Absolute (Abs.) Similarity Change to calculate the overall similarity change (Section 4.1.3.1).

The idea of overall similarity change calculation is to differentiate event and non-event windows by recognising the high changes in event windows. An event can happen suddenly introducing all the event keywords or make a sudden change to an ongoing discussion introducing a few new keywords to the data stream (Adedoyin-Olowe et al., 2016). If all the keywords are newly introduced when an event occurs, they will show a positive cluster change compared to the previous time window. If some high impact happened to an ongoing discussion by an event, newly introduced words would become closer to already existing words weakening some connections they had previously. Both positive and negative similarity changes should be expected in such a situation. Also, during a time window when no event happened, interest in an ongoing discussion can fade, resulting in some negative cluster changes. If the absolute changes are considered, negative and positive values will be similarly treated even though they hold valuable details regarding temporal event evolution. This issue can be simply solved by considering the non-absolute (*Non-abs.*) changes. But, the high availability of general discussions in social media data can unnecessarily lower the overall non-absolute change. Considering all these facts and the requirement of identifying emerging events, we propose a new calculation named *Positive Similarity Change* in this research as described below.

Positive (Pos.) Similarity Change: Under this calculation, a weighted average of positive values in *diffMatrix* is used to measure the overall similarity change as illustrated in Algorithm 2. As the weight, the proportion of positive values in the upper triangle of the *diffMatrix* is used. Only the positive values

are focused because positive changes occur when tokens become closer at W_t than W_{t-1} or word groups appear representing events.

Algorithm 2: Positive Similarity Change Calculation

input: *diffMatrix:* matrix of temporal word similarity differences
output: *posChange*: positive similarity change

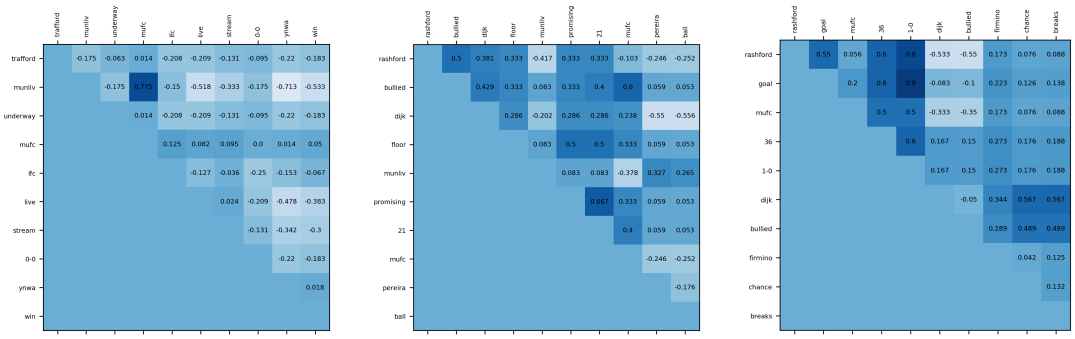
```
positiveValues ← []; // empty array to keep positive values
N ← length(diffMatrix_rows);
// Get positive values in the upper triangle of diffMatrix
for i=1 to N do
    for j=i+1 to N do
        if diffMatrix[i][j] > 0 then
            positiveValues.add(diffMatrix[i][j]);
        end
    end
end
// Calculate overall positive change
average ← ∑ positiveValues / length(positiveValues);
proportion ← length(positiveValues) / ((N × (N − 1)) / 2);
posChange ← average × proportion;
return posChange
```

TABLE 5.4: Sample events from MUNLIV

Time window	Event	Description
15:34-15:36 (W_{t_a})	-	No important event reported
15:52-15:54 (W_{t_b})	Foul	Foul by Marcus Rashford(MUFC) on Virgil van Dijk(LFC)
16:06-16:08 (W_{t_c})	Goal	First goal by Marcus Rashford(MUFC)

To explain the effectiveness of positive similarity change compared to absolute and non-absolute changes, we selected a few sample time windows with event details from the MUNLIV dataset, which are summarised in Table 5.4. Figure 5.4 shows the *diffMatrices* generated for each of these windows. To maintain simplicity, we only used the important keywords of the selected events to generate matrices. However, we used dendrograms generated considering all data belong to the targeted window to calculate cluster similarities

illustrating the real scenario. At W_{t_a}, no important events happened and *diffMatrix* in Figure 5.4a also has low values. At W_{t_b}, an emerging event happened and it is captured by the *diffMatrix* in Figure 5.4b with a majority of high positive values. At W_{t_c}, an ongoing discussion is altered by an event, and thus the *diffMatrix* in Figure 5.4c has a combination of negative and positive values. Overall similarity changes calculated for each of these matrices using Absolute, Non-Absolute and Positive calculations are summarised in Table 5.5.



(A) *diffMatrix* at W_{t_a} (B) *diffMatrix* at W_{t_b} (C) *diffMatrix* at W_{t_c}

FIGURE 5.4: *diffMatrices* generated for time windows W_{t_a}, W_{t_b} and W_{t_c} in Table 5.4

TABLE 5.5: Comparison of similarity change calculation methods

Similarity Change	***diffMatrix***		
	(A)	(B)	(C)
Abs.	0.2005	0.3388	0.2981
Non-abs.	-0.1467	0.2786	0.2092
Pos.	0.0269	0.3087	0.2537

According to the resulting values in Table 5.5, Abs. change returns a high value for W_{t_a} even though there are no events in that time window. Non-abs. change can effectively capture the non-existence of events (at W_{t_a}) and suddenly happened events introducing all the keywords (at W_{t_b}), but it returns a low value when an event occurred altering an ongoing discussion (at W_{t_c}). Pos. change showcases the ability to effectively capture all these scenarios with

low value at W_{t_a} and high values at W_{t_b} and W_{t_c}. Therefore we propose to use Pos. similarity change to calculate overall text similarity change over time.

In summary, Abs., Non-abs. and Pos. change compute an overall value for a matrix within the ranges of 0–1, -1–1 and 0–1, incorporating different impacts based on their attributes. Specifically, Abs. change treats both positive and negative values similarly, and Non-abs. change treats them separately, but the majority sign could dominate it. Distinctly, Pos. change particularly targets positive values of a matrix. If a large proportion of a matrix has positive values, it will return a high value; otherwise, it will return a low value. Even though we target positive values in this research as they represent words that became closer over time, this metric can also be easily customised for the opposite sign (negative).

5.1.2.3 Vocabulary Change Calculation and Aggregation

In addition to the token similarity changes, vocabulary change is considered covering the statistical details in text to calculate overall change. In Embed2Detect, vocabulary change is calculated based on the temporal token variations (Section 4.1.3.3). We referred to this calculation as *Vocabulary Change (VC)* following Equation 4.4. Here, we propose to involve frequency changes of tokens in addition to their temporal variations to calculate vocabulary change. With this modification, we can only focus on the significant changes excluding the slight modifications that happened to the discussions.

Frequency Difference-based Vocabulary Change (FDVC): For this calculation, we consider new tokens that appeared in the vocabulary of W_t with a

frequency difference above the threshold β, which is used to filter outlier tokens following the Equation 5.2.

$$FDVC_{(t-1,t)} = \frac{|w : w \in vocab_t, w \notin vocab_{t-1} \; and \; f(w)_t - f(w)_{t-1} > \beta|}{|vocab_t|} \quad (5.2)$$

In summary, VC and FDVC return normalised change values within 0–1. A value closer to 0 indicates that $vocab_{t-1}$ and $vocab_t$ are almost the same. If many new tokens are added to the $vocab_t$, VC will return a high value without considering their significance or frequency difference. Thus, this calculation will treat tokens that newly appeared once (mostly due to typing mistakes or background discussions) and multiple times (mostly due to events) similarly. Overcoming this limitation, FDVC only returns high values if a large proportion of $vocab_t$ holds new tokens with higher frequency differences above a threshold compared to $vocab_{t-1}$, capturing event occurrences. Thus, we propose to use FDVC along with WhatsUp.

After calculating the temporal textual change using two measures: similarity change and vocabulary change, these values need to be aggregated to calculate the overall change. Similar to the Embed2Detect approach, we consider maximum and average calculations as aggregation methods to preserve the simplicity of WhatsUp. However, the average calculation outperformed the maximum calculation with the proposed improvements to textual change measures (Section 5.2.1).

5.1.3 Event Cluster Detection

After identifying the event windows, the next phase detects co-occurring events/event clusters within those windows to extract fine-grained event details. Since one or more events would have been reported within an event window, it is important to detect the details of each event separately to let

users get quick and concise event updates. For clustering, we propose a novel method which focuses on identifying important keywords to extract clusters using the tokens closer to the keywords. Basically, our method has two steps: (1) token ranking (identifying important keywords) and (2) cluster generation, being flexible for easy adjustments for different events. To rank the tokens, we assign a weight to each token considering its textual similarity change over time, as sudden incidents within data streams result in textual changes. If a token has a high positive change value, it means that many words became closer to that token over time, indicating that it is an important keyword of an event or discussion. Thus, we sort the tokens with positive textual change in descending order to identify the important event keywords. We only consider the words in the vocabulary for this step, excluding emojis and non-alphanumeric characters, because the target is to rank keywords. We use LDL similarities calculated using self-learned word embeddings and their hierarchical relationships and token frequency-based measures for the weight calculation, capturing underlying linguistics and statistics as described in Section 5.1.3.1. Then, token groups are identified using linguistically nearby tokens to the ranked keywords as such groups represent the ideas discussed together. Similar to token weighting, we use LDL similarity between tokens to extract nearby tokens, capturing the linguistical relationships, as described in Section 5.1.3.2. Also, as additional information helpful to users, we occupy each cluster with a novelty measure that indicates the cluster's newness or newsworthiness. After detecting the clusters, we prune them to filter events following the approaches in Section 5.1.3.3. The overview of our event cluster detection approach is illustrated in Algorithm 3.

Unlike the majority of previous research (Nur'aini et al., 2015; Nguyen et al., 2019; Comito et al., 2019a; Comito et al., 2019b), our algorithm targets clustering tokens (not documents). As tokens, we use words and other useful

Algorithm 3: Event Cluster Detection

input: W_t^d : detected event window, s or m : cluster detection threshold, η or κ : cluster pruning threshold

output: $E_{W_t^d}$: co-occurred events (event-described token clusters) at W_t^d

$vocab_t \leftarrow$ vocabulary at W_t^d;
$weightedWords \leftarrow \{\}$; `// empty dictionary to keep word-weights`
$C \leftarrow []$; `// empty array to keep clusters`
`//` **Rank tokens which showed a temporal textual change**
for *w in* $vocab_t$ **do**
    $weight \leftarrow$ weight calculated for w using temporal textual change;
    if $weight > 0$ **then**
        $weightedWords.add(w, weight)$;
    end
end
$weightedWords.sort(descending = True)$;
`//` **Generate clusters**
for *w in weightedWords* **do**
    if *w not appeared in any token cluster in* C **then**
        $c \leftarrow$ nearby tokens to w based on s or m;
        $c \leftarrow [w, c]$;
        $Novelty_c \leftarrow$ novelty calculated on c;
        $C.add(c, Novelty_c)$;
    end
end
`//` **Prune clusters**
$E_{W_t^d} \leftarrow$ pruned clusters from C as events based on η or κ;
return $E_{W_t^d}$

symbols such as emojis because they play a crucial role in expressing public opinions in social media nowadays. When the token level is considered, multiple events described in a single document can be separated, and connections between tokens that do not appear together in documents can be captured. Due to the recent increments made to character limits by social media services (e.g. Twitter increased 140 character limit to 280 in 2017), it is possible

to have multiple event details in a single document. Also, due to the diversity of users, the same event can be expressed differently in multiple documents requiring capturing token relationships to get complete event information. Furthermore, our algorithm has advantages compared to the traditional clustering algorithms. Unlike flat clustering algorithms (e.g. K-means), our algorithm does not require the number of clusters to be predefined, which is unpredictable due to the dynamicity of data streams. Unlike hierarchical algorithms (e.g. hierarchical agglomerative, hierarchical divisive), our algorithm is capable of assigning the same token to multiple clusters. In reality, there can be situations when the same token is shared among multiple events (e.g. 'goal' is shared between events: *team MUFC scored a goal* and *VAR for the goal is in progress*).

5.1.3.1 Token Weight Calculation

To rank tokens, we assign them weights based on their temporal textual change (a high weight for a high change). We experimented with the following measures to calculate the token weight involving linguistics and statistics. To make the similarity-based computations faster, we use the *diffMatrix* generated during event window identification (Section 5.1.2.2), as a row of the matrix $diffMatrix[w]$ contains the values $\{LDL\ sim._{(w,w_i)} : w_i \in vocab_t\}$.

1. ***Pairwise Similarity Difference (PSD):*** The highest similarity difference of the token pairs which contain the targeted token.

$$PSD_w = max(LDL\ sim._{(w,w_i)} : w_i \in vocab_t)$$

2. ***Average Similarity Difference (ASD):*** Average similarity difference of all token pairs which contain the targeted token. We use the proposed positive similarity change calculation to calculate the average considering its effectiveness. In Algorithm 2, we showed how to apply this calculation

for a matrix, but similarly, it can also use for a row of the matrix (similarities between all token pairs which contain the targeted token).

$$positiveValues = \{LDL\ sim._{(w,w_i)} : w_i \in vocab_t\ and\ LDL\ sim._{(w,w_i)} > 0\}$$

$$ASD_w = \frac{\sum positiveValues}{|positiveValues|} \times \frac{|positiveValues|}{|vocab_t|}$$

3. ***Frequency Difference (FD):*** Normalised frequency difference of the targeted token.

$$FD_w = \frac{f(w)_t - f(w)_{t-1}}{max(f(w)_t, f(w)_{t-1})}$$

4. ***Average Aggregations (AVG-*-FD):*** Average of similarity-based value and FD. Since there are two similarity-based measures, average aggregation also returns two measures *AVG-PSD-FD* and *AVG-ASD-FD*.

5. ***Maximum Aggregations (MAX-*-FD):*** Maximum of similarity-based value and FD. Similar to the average aggregation, this also has two measures *MAX-PSD-FD* and *MAX-ASD-FD*.

According to the experiment results described in Section 5.2.2, we found that AVG-ASD-FD outperforms the other weight measures.

5.1.3.2 Cluster Generation

In a text corpus, a token becomes closer to another token linguistically if used in the same context or to describe the same idea. Following this idea, we propose generating clusters based on tokens nearby to the top-ranked keywords, which had high temporal textual changes. To measure the similarity between tokens during cluster generation, we use LDL similarity, measured using dendrograms generated on word embeddings, allowing to capture both token relationships in the hierarchical structure and the vector space. Additionally, we require a threshold to extract nearby words, and we consider two threshold types as follows.

1. ***Similarity-based Threshold (s):*** Consider all tokens with similarity $\geq s$ as nearby tokens. This is similar to the similarity thresholds used by hierarchical and online clustering algorithms.

2. ***Count-based Threshold (m):*** Consider m most similar tokens as nearby tokens.

We conducted experiments using both thresholds and revealed that they have different qualities, which are helpful for users depending on their interests and domains specificities as described in Section 5.2.2.

After identifying the clusters, we assign a novelty measure for each cluster to improve its informativeness. The novelty of cluster C is calculated as the average of token weights belonging to it (Equation 5.3). Overall, it measures the temporal textual change of that cluster compared to the previous time window and indicates the event significance.

$$Novelty_c = \frac{\sum Weight_w : w \in c}{|c|} \tag{5.3}$$

5.1.3.3 Cluster Pruning

The detected clusters can be emerging events or background discussions. Therefore, they need to prune to extract event clusters. The commonly used approach in previous research is to assign clusters a value that indicates their importance and filter out the less important clusters using a threshold (Li et al., 2017a; Comito et al., 2019a; Morabia et al., 2019). Following this idea, we suggest pruning clusters using a threshold for cluster novelty (η). Nonetheless, such a threshold is highly dependent on the underlying data. Even within the same domain, using a static value for multiple time windows will be incorrect if the nature of data and their evolution are different in those time windows.

Also, due to the dynamicity and diversity of events, it will be difficult to pick such a threshold value by analysing previous data and heuristics.

We propose a keyword count-based technique to prune clusters considering all these drawbacks. Rather than pruning the generated clusters, our approach targets limiting non-event cluster generation. Given a count κ, this will only take the top κ keywords in the ranked tokens to use with the cluster generation step in Algorithm 3. Knowing the data and possible event generation rates, κ can be defined, and it is not very dependent on other factors as a novelty-based threshold. The experiments we conducted further emphasise the effectiveness of κ than η in Section 5.2.3.

Algorithm 4: WhatsUp

input: D' : filtered data stream ($[d_1, d_2, ...d_{i-1}, d_i, ...]$), l : time window length,
α : change threshold, s or m : cluster detection threshold,
η or κ : cluster pruning threshold
output: E : detected co-occurred events in event windows

$E \leftarrow []$; `// empty array to keep co-occurred events in event windows`
`//` **Data preprocessing**
$W \leftarrow D'$ separated into l long time windows;
for W_t *in* W **do**
    $V_t \leftarrow$ vector space learned from W_t;
    $vocab_t \leftarrow$ vocabulary at W_t;
    $f_t \leftarrow$ token frequencies of $vocab_t$;
    $W_t \leftarrow \{V_t, vocab_t, f_t\}$;
end
`//` **Event window identification**
$W^d \leftarrow$ Algorithm 1(W, α);
`//` **Event cluster detection**
for W_t^d *in* W^d **do**
    $E_{W_t^d} \leftarrow$ Algorithm 3$(W_t^d, s$ or m, η or $\kappa)$;
    $E.add(E_{W_t^d})$;
end
return E

Concluding Section 5.1, Algorithm 4 summarises the top-level idea of WhatsUp, combining all the steps described above.

5.2 Experimental Study

This section summarises the experiments we conducted using WhatsUp on sports and political domains using MUNLIV and BrexitVote datasets introduced in Section 3.2. We used the updated version (V2) of ground truth (GT) for these experiments. For evaluations, both time and text-based metrics introduced in Section 3.3 were used. While comparing the results, we prioritised Time Window F1, followed by Event Recall, Event Relevance and Keyword Recall because event measures were calculated considering the GT events of the detected event windows and keyword measures were calculated considering the matched GT events. We implemented WhatsUp in Python to conduct experiments, and it is publicly available on GitHub[2].

Initially, we analysed the effectiveness of strategies we proposed for event window identification (Section 5.2.1). Then, the performance of event cluster detection is evaluated (Section 5.2.2). Following Section 5.2.2, the overall performance of WhatsUp is evaluated and compared with several recently proposed methods from different competitive areas (Section 5.2.3). Since, similar to Embed2Detect, word embedding learning and the event window identification are the most computationally complex operations in WhatsUp, the discussion in Section 4.1.5 confirms its appropriateness for real-time processing. However, for further validity, we thoroughly analysed the efficiency of our method and reported the results along with overall performance. During

[2]WhatsUp implementation is publicly available on https://github.com/HHansi/WhatsUp

all experiments, we followed a common strategy to identify optimal hyper-parameters to generate unbiased comparable results, as results are highly biased to the hyper-parameter configurations. We evaluated the results of all possible hyper-parameter settings and picked the best results to report and compare. Finally, Section 5.2.4 summarises the findings for hyper-parameter sensitivity analysis. We used a Common KVM CPU @ 2.40GHz with 16GB RAM for all experiments.

5.2.1 Event Window Identification

For event window identification, we utilised the idea of Embed2Detect and proposed different strategies to improve the performance. Thus, we mainly focused on comparing the effectiveness of the proposed strategies with the original idea in this section. For evaluations, we only used time-based metrics defined in Section 3.3 because these experiments only identify event windows. To measure time-based metrics without event clusters, we considered all tokens which showed any temporal textual change within an event window as event words, similar to Embed2Detect. The whole group of event tokens is considered as a single cluster and matched it with multiple GT events following the same criteria defined for event matching. Initially, we analysed the impact on event window identification by the improvements to cluster change calculation. The obtained results are summarised in Table 5.6 and 5.7.

We consider the Time Window F1 to compare the results since it computes the harmonic mean between recall and precision. According to the results in Table 5.6, for MUNLIV, LDL similarity performed better than DL similarity for all strategies. Also, non-relative dendrograms performed better than relative dendrograms. Considering the similarity change calculations, Non-abs. change returned the lowest F1 values while Pos. change performed best. A

TABLE 5.6: Evaluation results: Time Window Recall (R), Precision (P) and F1 of event window identification with different strategies for MUNLIV. T and F indicate relative and non-relative dendrograms. The best results are in bold and Embed2Detect's results are with ‡.

Similarity Change	Relative Dendro.	DL Similarity			LDL Similarity		
		R	P	F1	R	P	F1
Abs.	T	0.8800‡	0.5000‡	0.6377‡	0.8400	0.5385	0.6563
Abs.	F	0.8800	0.5238	0.6567	0.8800	0.5238	0.6567
Non-Abs.	T	0.0400	1.0000	0.0769	0.0800	1.0000	0.1481
Non-Abs.	F	0.5600	0.6087	0.5833	0.6800	0.6071	0.6415
Pos.	T	0.2400	0.8571	0.3750	0.5200	0.5000	0.5098
Pos.	F	0.7200	0.5806	0.6429	**0.7200**	**0.6429**	**0.6792**

TABLE 5.7: Evaluation results: Time Window Recall (R), Precision (P) and F1 of event window identification with different strategies for BrexitVote. T and F indicate relative and non-relative dendrograms. The best results are in bold and Embed2Detect's results are with ‡.

Similarity Change	Relative Dendro.	DL Similarity			LDL Similarity		
		R	P	F1	R	P	F1
Abs.	T	1.0000‡	1.0000‡	1.0000‡	1.0000	1.0000	1.0000
Abs.	F	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
Non-Abs.	T	0.3636	1.0000	0.5333	0.0909	1.0000	0.1667
Non-Abs.	F	0.6364	1.0000	0.7778	1.0000	1.0000	1.0000
Pos.	T	0.7273	1.0000	0.8421	0.8182	1.0000	0.9000
Pos.	F	0.9091	1.0000	0.9524	**1.0000**	**1.0000**	**1.0000**

similar pattern is also found from BrexitVote results (Table 5.7), but they are not very differentiative due to high values. High results are returned mainly because almost all the time windows in this dataset have events. However, newly introduced strategies (LDL similarity on non-relative dendrograms and Pos. similarity change) improved the F1 by 4.15% for MUNLIV while maintaining the 100% F1 for BrexitVote.

Moreover, we analysed the impact by combining vocabulary change calculation using maximum and average aggregations, and its improvements

TABLE 5.8: Evaluation results: Time Window Recall (R), Precision (P) and F1 of event window identification for different aggregations. Best results are in bold.

Aggregation		**MUNLIV**			**BrexitVote**		
Method	Values	**R**	**P**	**F1**	**R**	**P**	**F1**
Average	LDL sim., VC	0.6800	0.7391	0.7083	1.0000	1.0000	1.0000
Maximum	LDL sim., VC	0.6800	0.7083	0.6939	1.0000	1.0000	1.0000
Average	LDL sim., FDVC	**0.7200**	**0.7500**	**0.7347**	**1.0000**	**1.0000**	**1.0000**
Maximum	LDL sim., FDVC	0.8000	0.6667	0.7273	1.0000	1.0000	1.0000

with cluster change calculation. The obtained results are available in Table 5.8. According to the results, average aggregation performed best. Among vocabulary change calculations, FDVC got higher results than VC. Following the theoretical exposure described in Section 5.1.2 and the obtained results, we can conclude that the performance of event window identification can be improved using LDL similarity on non-relative dendrograms, Pos. similarity change and average aggregation using FDVC. We use this combination for the following experiments (Sections 5.2.2, 5.2.3).

5.2.2 Event Cluster Detection

The method proposed for clustering consists of two main steps: (1) token ranking and (2) cluster generation, excluding the final cluster pruning step, which filters the event clusters. In this section, we report the evaluations of these two main steps. For token ranking, we experimented with the effectiveness of different weighting mechanisms described in Section 5.1.3.1 and the obtained results are summarised in Table 5.9. The target of token ranking is finding important event keywords to use with cluster generation. Therefore to measure the effectiveness of each weighting mechanism, we selected the top n tokens from the ranked list and measured the time-based metrics. For MUNLIV, we set n to 20 and for BrexitVote, to 100 for these experiments. These values are

chosen based on each dataset's average vocabulary size per time window. According to the results, among similarity-based weights, ASD obtained better results than PSD. This emphasises the importance of involving all similarity changes of a token for weight calculation. FD also returned relatively high F1 values for both datasets. However, with the average (AVG) aggregation of ASD and FD, we could further improve the results involving underlying linguistics and statistics. Based on the results, we can conclude that AVG-ASD-FD is the best weighting mechanism among others. For later experiments, we only use this mechanism (AVG-ASD-FD).

TABLE 5.9: Evaluation results: Time Window Recall (R), Precision (P) and F1 of token ranking. Best results are in bold.

Token Weight	**MUNLIV**			**BrexitVote**		
	R	**P**	**F1**	**R**	**P**	**F1**
PSD	0.2000	0.2083	0.2041	0.0909	0.0909	0.0909
ASD	0.4000	0.4167	0.4082	0.2727	0.2727	0.2727
FD	0.6400	0.6667	0.6531	0.3636	0.3636	0.3636
AVG-PSD-FD	0.3200	0.3333	0.3265	0.1818	0.1818	0.1818
AVG-ASD-FD	**0.6800**	**0.7083**	**0.6939**	**0.4545**	**0.4545**	**0.4545**
MAX-PSD-FD	0.1600	0.1667	0.1633	0.0909	0.0909	0.0909
MAX-ASD-FD	0.6400	0.6667	0.6531	0.3636	0.3636	0.3636

TABLE 5.10: Evaluation results of token clustering. TW stands for Time Window. Best results are in bold.

Dataset	**Method**	**TW Recall**	**TW Precision**	**TW F1**	**Event Recall**	**Event Relevance**	**Keyword Recall**
MUNLIV	Proposed(s)	0.7200	0.7500	0.7347	0.7742	0.2500	0.7813
	Proposed(m)	0.7200	0.7500	**0.7347**	**0.7742**	**0.3304**	**0.8125**
	HAC	0.5600	0.5833	0.5714	0.6129	0.2447	0.9423
BrexitVote	Proposed(s)	1.0000	1.0000	**1.0000**	**1.0000**	**0.4444**	**0.7917**
	Proposed(m)	1.0000	1.0000	1.0000	1.0000	0.1810	0.7053
	HAC	1.0000	1.0000	1.0000	1.0000	0.0762	0.6316

Then we evaluated the effectiveness of the cluster generation step. Since we use dendrograms generated using hierarchical agglomerative clustering (HAC) to measure text similarities during the clustering, we compared our approach with HAC. The obtained results for both datasets are summarised in Table 5.10. There are two variants of the proposed approach, which use a similarity-based threshold (s) and a count-based threshold (m). According to the results, both variants of the proposed approach outperformed HAC for both datasets. For BrexitVote, similar Time Window F1 and Event Recall values were returned by HAC, but Event Relevance and Keyword Recall values are lower than the values returned by our approach. However, all the experiments return low Event Relevance values because we only performed cluster generation in this stage without further pruning to extract event clusters. Considering Event Relevance and Keyword Recall, the proposed approach with m performed best for MUNLIV and s performed best for BrexitVote. In the sports domain, considering a particular match, all events have the same structure with nearly the same number of keywords per event. Therefore m-based approach is capable of performing well in the sports domain. Unlike this, different events in different structures are discussed in the political domain. These events can have different keyword counts, and thus, the s-based approach is more appropriate for such a domain than the m-based approach.

5.2.3 Overall Performance

In this section, we report the overall performance of WhatsUp, comparing it with several recently proposed methods. We mainly focused on their accuracy, efficiency and expandability during the selection. Also, we considered different competitive areas such as social aspect, segments over unigrams, clustering and topic modelling to strengthen the baselines. More details of the selected

methods are as follows.

- ***MABED*** (Guille and Favre, 2015): MABED uses anomalous variations in mentions to detect events in an offline manner, as described in Section 4.2.4. This method uses word co-occurrences and their temporal dynamics to extract event text. Previously, we modified the output to event windows, but with WhatsUp, we evaluated the resulting events.

- ***SEDTWik*** (Morabia et al., 2019): SEDTWik clusters bursty segments in a data corpus involving Wikipedia titles, text statistics and user diversity-based measures, as described in Section 4.2.4. This uses the Jarvis-Patrick algorithm to identify event clusters. We applied this method to each time window and only considered the events with high newsworthiness than a defined threshold for our experiments.

- ***LDA-based Sub-Event Tracking (LDA-SET)*** (Unankard and Nadee, 2020): LDA-SET identifies topics per window using Latent Dirichlet Allocation (LDA) and then analyses their temporal evolution to capture event transitions. The temporal evolution is mainly assessed using topic similarity (common keyword count-based measure) and topic word count. Among the captured transitions, we only considered *form* events (or new topics) for the experiments because we only evaluate such events in this research.

The results obtained for MUNLIV and BrexitVote datasets using WhatsUp and baseline methods are summarised in Table 5.11 and 5.12. For WhatsUp, we experimented with both similarity-based (s), and count-based (m) clustering approaches as they showed different qualities useful for different domains

and users. To extract event clusters, we pruned the clusters using both thresholds: novelty (η) and keyword count (κ) to see the effectiveness of each technique. Combining both clustering and cluster pruning thresholds, in total, for WhatsUp, there are four combinations as shown in the results tables. The bellow section of each table lists the experimented baseline methods and obtained results.

TABLE 5.11: Performance comparison of WhatsUp with baseline methods for MUNLIV. TW stands for Time Window and Average Time indicates the processing time for a 2-minute window. The best result is in bold and the best baseline result is in italics.

Method	TW Recall	TW Precision	TW F1	Event Recall	Event Relevance	Keyword Recall	Average Time(s)
WhatsUp(s, κ)	**0.7200**	**0.7500**	**0.7347**	**0.7742**	**0.4400**	**0.7813**	3.6140
WhatsUp(m, κ)	0.7200	0.7500	0.7347	0.7742	0.4231	0.8125	3.7193
WhatsUp(s, η)	0.7200	0.7500	0.7347	0.7742	0.2836	0.7619	3.8596
WhatsUp(m, η)	0.7200	0.7500	0.7347	0.7742	0.3743	0.8125	3.7368
MABED	0.5200	0.3095	0.3881	0.5161	0.2867	0.6444	3.9825
SEDTWiK	0.4800	0.2308	0.3117	0.5161	0.3148	0.4889	23.8297
LDA-SET	*0.9600*	*0.4211*	*0.5854*	*0.9677*	*0.2391*	*0.8442*	3.7807

TABLE 5.12: Performance comparison of WhatsUp with baseline methods for BrexitVote. TW stands for Time Window and Average Time indicates the processing time for a 30-minute window. The best result is in bold and the best baseline result is in italics.

Method	TW Recall	TW Precision	TW F1	Event Recall	Event Relevance	Keyword Recall	Average Time(s)
WhatsUp(s, κ)	**1.0000**	**1.0000**	**1.0000**	**1.0000**	**0.6585**	**0.7396**	32.1818
WhatsUp(m, κ)	1.0000	1.0000	1.0000	1.0000	0.2463	0.5464	35.4545
WhatsUp(s, η)	1.0000	1.0000	1.0000	1.0000	0.5152	0.7917	32.4545
WhatsUp(m, η)	1.0000	1.0000	1.0000	1.0000	0.2671	0.6489	33.6364
MABED	*0.9091*	*0.9091*	*0.9091*	*0.8889*	*0.6067*	*0.5682*	51.2727
SEDTWiK	0.7273	0.7273	0.7273	0.7407	0.7500	0.3784	68.2666
LDA-SET	0.8182	0.8182	0.8182	0.8889	0.5962	0.4471	17.5000

According to the results, for MUNLIV, all four combinations of WhatsUp outperformed the baselines returning 14.93% higher Time Window F1 than

the best baseline method. Similarly, for BrexitVote, all combinations of WhatsUp obtained 9.09% higher Time Window F1 than the best baseline method, proving that our approach can more accurately extract temporal event information in diverse domains than the available methods. We cannot involve event- and keyword-based metrics for comparisons between methods because they are calculated based on the event windows identified by each method. When methods detect different event windows, different GT events will be considered for the event and keyword-based evaluations (Section 3.3.2). Using WhatsUp, for both datasets, we received the same Event Recall for all combinations. However, there are variations in Event Relevance and Keyword Recall depending on the used combination. Overall, in both datasets, κ-based pruning performed better than η-based pruning. Following the insights in Section 5.2.2, for MUNLIV, both s- and m-based clustering performed well with κ, but for BrexitVote, only s-based clustering performed well in terms of Event Relevance and Keyword Recall. In summary, these experiments reveal that WhatsUp with (s, κ) combination performs more effective event detection in diverse domains, specifically, sports and politics, than other combinations and recent baseline methods. Further, we report qualitative analysis of a sample of detected events in Appendix A.2 focusing on word coverage and novelty measure, which further emphasise the effectiveness of event details detected by WhatsUp.

Additionally, the processing time is also a critical factor in performing event detection in (near) real-time. Targeting this requirement, we involved efficient computation techniques for our approach and also incorporated parallel processing for all time-consuming operations. To measure the reported processing times in Table 5.11 and 5.12, we used parallel processing with eight workers. According to the measured times, on average, WhatsUp took 3.72 seconds to process a 2-minute time window ($\approx 1,724$ tweets) and 33.43 seconds to process

a 30-minute time window ($\approx 14,542$ tweets). Compared to the majority of the baseline methods, our approach performed faster in both datasets. Considering the specifications of the used machine and measured processing times, we can state that WhatsUp is sufficiently fast for (near) real-time detection. Also, parallel processing provides the capability to handle increasing data amounts successfully. A detailed analysis of intermediate processing times comparing the sequential and parallel processing is available in Appendix A.3.

TABLE 5.13: Best hyper-parameter settings

Method	**Hyper-parameters**	
	MUNLIV	**BrexitVote**
WhatsUp	$\alpha = 0.12$	$\alpha = 0.12$
	$\beta = 20$	$\beta = 10$
	$(s, \kappa) = (0.4, 15)$	$(s, \kappa) = (0.6, 130)$
	$(m, \kappa) = (25, 15)$	$(m, \kappa) = (35, 50)$
	$(s, \eta) = (0.4, 0.1)$	$(s, \eta) = (0.6, 0.08)$
	$(m, \eta) = (25, 0.12)$	$(m, \eta) = (35, 0.2)$
MABED	$k = 150$	$k = 150$
	$p = 25$	$p = 35$
	$\theta = 0.7$	$\theta = 0.6$
	$\sigma = 0.5$	$\sigma = 0.5$
SEDTWik	$M = 2$	$M = 3$
	$k = 5$	$k = 3$
	$\tau = 0.6$	$\tau = 0.3$
LDA-SET	$k = 15$	$k = 30$
	min. probability$= 0.01$	*min. probability*$= 0.01$
	threshold match$= 0.7$	*threshold match*$= 0.6$

Hyper-parameters: For all methods, we used the optimal hyper-parameter settings to generate comparable results since the performance is dependent

on the parameters configurations. Table 5.13 summarises the optimal hyper-parameter values obtained. For WhatsUp, we optimised the change threshold (α) and frequency threshold (β), considering all possible values to identify event windows. To detect event clusters, four-parameter combinations of cluster word similarity (s) or cluster word count (m) and novelty (η) or keyword count (κ) were optimised following all possibilities. For m, we defined a maximum value of 25 for MUNLIV and 35 for BrexitVote, considering the GT event word counts. For MABED, we optimised the number of events (k), the maximum number of words describing each event (p), weight threshold for selecting relevant words (θ) and overlap threshold (σ). For k and p, incremental values have been experimented with until we receive the optimal result. For θ and σ, we experimented with the values around the original values used in the initial experiments (Guille and Favre, 2015). For SEDTWik, we optimised the number of subwindows (M), number of cluster neighbours (k) and newsworthiness threshold (τ). Values for M were picked based on the time window lengths of datasets. For k and τ, incremental values were experimented with until receiving the optimal result. For LDA-SET, we optimised the number of topics (k), the minimum probability to filter topic words and threshold match, which compares the events temporally. We experimented with incremental values for all of them until we received the optimal result.

5.2.4 Parameter Sensitivity Analysis

WhatsUp requires hyper-parameters during word embedding learning, event window identification and event cluster detection. They should be mainly picked depending on the characteristics of the targeted domain and user preferences on resulting events. Different parameters make different levels of impact on the output, as described below.

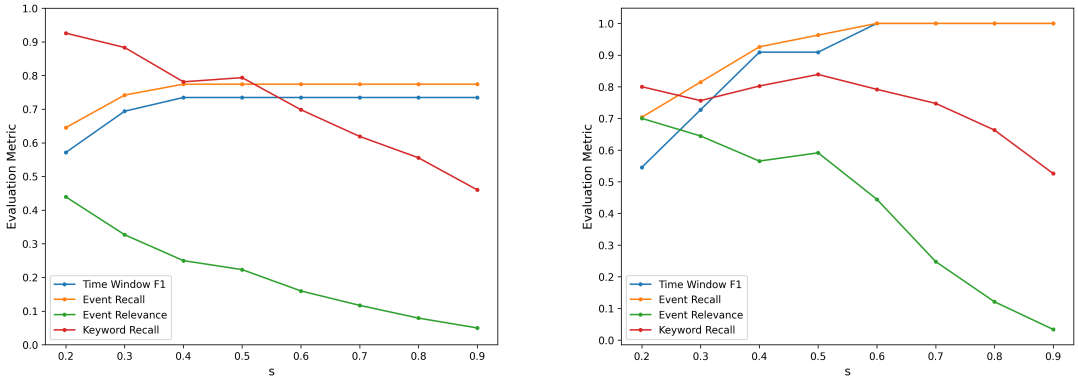
Word Embedding Learning: To train Word2Vec models during embedding learning, three hyper-parameters: minimum word count, context size, and vector dimension are required. The minimum word count removes tokens with less total frequency than the count. The context size defines the number of words around the target word to consider during learning. The vector dimension represents the number of dimensions in the final word embeddings. Following a comprehensive analysis conducted in Section 4.2.3.1, we fixed 1 for minimum word count, 5 for context size and 100 for vector dimensions as the optimal setting appropriate for time window data.

Event Window Identification: To identify event windows, two hyper-parameters: change threshold (α) and frequency threshold (β) are required. α mainly indicates the significance level of targeted events, and β is used as a frequency threshold to remove outlier tokens. A value for α can be picked using the domain knowledge and analysing a few past time windows. In addition to the domain-specific characteristics, personal preferences also need to be considered. β is also a domain-dependent parameter since the inclusion and frequency of outlier tokens vary with the domain. A detailed analysis of these parameters have been conducted with Embed2Detect in Section 4.2.3.2 and the heuristics suggested by it are applicable for WhatsUp also since it uses Embed2Detect's core idea for event window identification.

5.2.4.1 Event Cluster Detection

To generate clusters, we proposed to use either a similarity-based threshold (s) or a word count-based threshold (m). According to the overall performance, s-based clustering performed best for MUNLIV and BrexitVote. Any value between 0-1 can be picked for s since it considers the LDL similarity between tokens. To provide a clearer idea, we plotted the variations of evaluation metrics

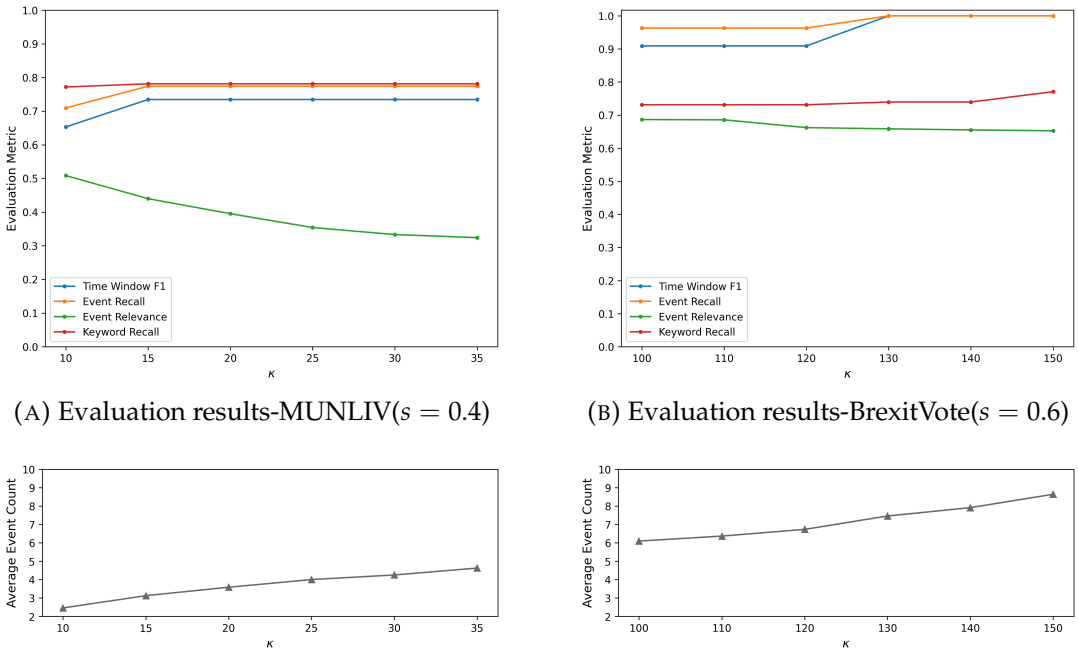
with varying s on both datasets in Figure 5.5. According to the obtained results, Time Window F1 and Event Recall increase with increasing s until they obtain their maxima for both datasets. $s \approx 0$ generates huge clusters by grouping less similar tokens as well and fails to identify event clusters separately. Thus, optimal results are obtained when s is sufficiently large for the cluster separation. Contrarily, Event Relevance and Keyword Recall decrease with increasing s. When $s \approx 1$, very small clusters will be generated due to the consideration of very similar tokens. Thus, unnecessary separation of event clusters occurs, reducing the keyword coverage per cluster. Therefore it is important to pick a middle value for s. For MUNLIV, a value between 0.4-0.6 and BrexitVote, a value between 0.6-0.8, is appropriate. Following these insights, we can summarise that if the targeted domain is highly evolving and short time windows with fewer data are considered, a comparatively low value will be the optimal s. A relatively high value will be optimum for a domain with opposite characteristics. Additionally, as described in Section 5.1.3, if the targeted domain has events with a similar structure, m-based clustering can be used, and the average number of keywords per event can be set as m straightforwardly.



(A) Evaluation results- MUNLIV (B) Evaluation results- BrexitVote

FIGURE 5.5: Evaluation results for varying similarity threshold (s) values

After generating the clusters, they need to be pruned to extract important events. We proposed two approaches using novelty (η) and keyword count (κ)



(A) Evaluation results-MUNLIV($s = 0.4$)

(B) Evaluation results-BrexitVote($s = 0.6$)

(C) Average event counts- MUNLIV

(D) Average event counts- BrexitVote

FIGURE 5.6: Evaluation results and average event counts per event windows for varying keyword count (κ) values

for pruning and, among them, the κ-based approach performed best in sports and political domains. As κ, the possible keyword count per time window should be set. For MUNLIV, a low value than BrexitVote is preferred because MUNLIV has shorter time windows with fewer possible events per window. To provide a clearer insight, we plotted evaluation results and resulted in cluster counts with varying κ values in Figure 5.6. According to the results, for both datasets, Time Window F1 and Event Recall achieve their optimal state with sufficiently large κ to capture all important events. When κ is lower, important events can be missed. However, Keyword Recall shows a near-constant behaviour, and Event Relevance shows a slight drop with increasing κ. Since the number of capturing events grows with increasing κ (Figure 5.6c, 5.6d), it is possible to capture non-relevant events too. Thus, when picking a value for κ, the focus should be on the lowest possible keyword count, which is sufficiently large to capture relevant events.

5.3 Conclusions

In this chapter, we proposed a novel event resolution method named *WhatsUp* for co-occurring event detection in social media. Our approach focuses on the automatic detection of both temporal and fine-grained textual (text of co-occurred events) event details involving statistical and linguistical features of underlying data. To capture statistics, token frequency-based measures were used and to capture linguistics, self-learned word embeddings and their hierarchical relationships in dendrograms were used. In summary, WhatsUp considers all the important features in textual data needed for effective event detection. Also, to the best of our knowledge, no prior work involved both statistics and linguistics in detecting both temporal and textual event details together. Further, the usage of unsupervised techniques makes our approach expandable to any domain, language or platform. The involvement of self-learned word embeddings also supports expandability as well as understanding data-specific linguistics.

WhatsUp returned promising results on conducted experiments. We used several recently proposed methods from different competitive areas to involve stronger baselines for comparisons. Also, we designed metrics that cover both temporal and textual aspects of events to comprehensively evaluate the accuracy of methods and used data from two diverse domains (i.e. sports and politics) to assess the expandability. Overall, WhatsUp outperformed all the baselines in terms of event time and text detection, emphasising its accuracy and expandability. Following these results, it is also safe to assume that our approach has low information loss than available approaches, which is possible due to the insufficient involvement of linguistics. Considering the processing time, WhatsUp took a comparatively short time for both datasets, proving

its efficiency appropriate for (near) real-time processing. Targeting the scalability, embedding learning and time window identification are the most computationally complex steps of WhatsUp, which utilise the core idea of Embed2Detect that has proven to be scalable for large data volumes.

In future work, we plan to mine sentiments of events to enhance their informativeness because knowing public opinion would be helpful in situations such as crises, political debates and product launches to take necessary immediate actions. Considering the lack of data availability for event sentiment analysis, we have already prepared and released a sentiment dataset named *TED-S* (Hettiarachchi et al., 2022b) covering MUNLIV and BrexitVote tweets and plan to use it with our future developments. Further, we believe it is useful to integrate a text summarisation technique to return a summary of each event cluster. When a similarity-based threshold is used for clustering, it is possible to grow clusters massively in some instances. A summary will be helpful in such situations to provide the user with a quick insight into the event. Also, we plan to extend our approach to detect the evolution of events over time to understand the event's progress and analyse the impact of time window length for event resolution. Additionally, it would be interesting to evaluate the method's performance on multilingual data. Since social media allow posting in different languages, information from different people groups can be combined with multilingual event detection.

With this chapter, we conclude Part I of the thesis. In this part, we proposed novel event detection methods (i.e. Embed2Detect and WhatsUp) for coarse- and fine-grained level detection from social media data, which returned promising results, outperforming several recently proposed competitive methods. We particularly involved self-learned word embeddings and their hierarchical relationships in dendrograms in our approaches to facilitate the capturing of underlying linguistics, overcoming a critical limitation found

in previous research. As far as we know, this is the first effort to use underlying linguistics to detect temporal and textual event details from social media data streams. Thus, we believe our findings will lead future research in social media event detection in a new direction. Then, we move to news media event detection, the other targeted area of this research, in the next part. Similar to Part I, Part II describes our approaches for coarse- and fine-grained level detection from news media data involving underlying linguistics.

Part II

News Media Event Detection

Chapter 6

Introduction to News Media Event Detection

Initiating Part II of the thesis, this chapter introduces news media event detection and provides an overview of this part. It mainly describes the important aspects to consider to effectively detect events from different data levels of news media, following the previous work discussed in Chapter 2 and the characteristics of textual data. Furthermore, this chapter defines the targeted problem, details the resources and concepts utilised for event detection by this part of the research and provides an overview of the remaining chapters (Chapters 7 and 8) of this part.

Similar to social media services, online news agencies also generate a vast amount of data, as described in Chapter 1. However, most of this data is unstructured and cannot be easily understood. Also, the high data generation makes manual information extraction harder. Thus, automated intelligent mechanisms are crucial for effectively extracting the information available in news media (Balali et al., 2020). Addressing this requirement, various approaches have been proposed by previous research, targeting different levels of data granularity: document, sentence and word levels as discussed in Chapter 2 (Section 2.3).

The techniques used by available approaches mainly range from traditional

machine learning (ML) to deep learning (DL). The earlier work extensively relied on language-specific linguistic tools, resources and features, only focusing on high-resource languages such as English (Naughton et al., 2010; Hong et al., 2011). These approaches mainly suffered from expandability issues and inabilities to support low-resource languages. With the evolution of deep neural networks and their effectiveness, later research focused more on DL-based approaches to detect events (Chen et al., 2015; Lindén et al., 2018; Liu et al., 2019a). This mostly eliminated the requirement to rely on linguistic tools, resources and features. However, deep networks require more instances for the training process, limiting their applicability when training data is scarce. The other major challenge experienced by both traditional ML- and DL-based approaches is handling text ambiguity. For example, the phrase 'classrooms empty' could refer to a teachers' strike, and without knowing the context, it cannot be recognised (Hürriyetoğlu et al., 2021b). Also, the word *'workers'* in the sentences in Figure 6.1 plays three different roles. The first sentence does not describe any event, but the other two describe events expressed by the words (triggers) *'strike'* and *'vandalised'*. Thus, *'workers'* in sentence (1) is not event-related. However, *'workers'* in sentences (2) and (3) hold event arguments, participant and target, respectively. Thus, it is crucial to focus on textual context to resolve such ambiguities while extracting event details.

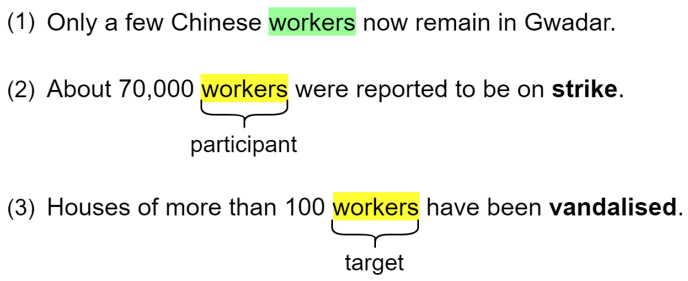


FIGURE 6.1: Sample sentences from news articles with word *'workers'*. Bold text represents the triggers in event-described sentences. Word *'workers'* is highlighted in yellow if it represents an event argument and in green otherwise.

Considering the crucial requirement to understand the textual context and the limitations in available methods, we design approaches based on transformers to detect events in news media. Transformers are designed with the ability to fine-tune for a downstream task by utilising their original knowledge in the language model (Devlin et al., 2019). Thus, unlike deep neural networks, transformer-based models can learn effectively using comparatively few training instances. Also, transformer architecture is capable of capturing contextual details in the text to disambiguate word senses. We aim to extract events, defining two data granularities useful for different user groups. At the coarse-grained level, we target identifying event-described news articles as described in Chapter 7. Such a system filters articles with targeted/interesting events so that they can be further analysed manually to obtain event details without possible machine errors, being helpful in situations where sensitive information is involved. At the fine-grained level, we target extracting event-described sentences and words as described in Chapter 8. This automates the manual analysis of event-described articles, returning complete event information helpful in situations where full automation is preferred. Rather than limiting to high-resource languages like English, we also apply our approaches to low-resource languages and investigate how their performance can be improved using different learning strategies and state-of-the-art transformer models.

The main contributions of this part of the thesis are as follows.

1. We propose a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformers for event-described news article identification, becoming the winning solution for the English language in subtask 1 of *Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE) 2021-Task 1*.

2. We propose a novel learning strategy named *Two-phase Transfer Learning*, involving different levels of data granularity and the capabilities of state-of-the-art transformer models and apply it to sentence and token level tasks of news media event detection to evaluate and discuss its effectiveness and applicability.

3. We empirically evaluate how the performance of news media event detection at the document, sentence and token levels can be improved for high- and low-resource languages involving different learning strategies and the characteristics of state-of-the-art transformer models.

4. We release our method implementations as open-source projects to support applications and research in the area of news media event detection[1].

The rest of this chapter is organised as follows. Section 6.1 defines the problem targeted by this research. Section 6.2 describes the datasets we used for evaluations. Section 6.3 details the evaluation metrics we involved. Section 6.4 summarises the background concepts we used, especially about the transformer models. Finally, Section 6.5 provides a summary of this chapter introducing the following chapters in Part II of the thesis.

6.1 Problem Definition

The problem targeted by this part of the research is automatically detecting events in news articles. Generally, news articles could contain various events from diverse domains. Rather than focusing on all of them, previous research mostly focused on specific events such as natural disasters (Nugent et al.,

[1]Links to the GitHub repositories are provided in bellow Chapters 7 and 8

2017), economic events (Lefever and Hoste, 2016) and political events (Hürriyetoğlu et al., 2021a). Specific focus allows the algorithm to learn the characteristics of the targeted domain and make more accurate predictions. Also, in reality, users are more interested to know events from particular domains more accurately rather than knowing all the events from different domains. Following this tendency and requirement, we also focus on extracting specific event details in this research. Also, we aim to design algorithms which do not rely on features from a particular domain or can be customised easily for different domains.

Similar to Part I of this thesis, we mainly divide news media event detection into two parts based on data granularity to address different user requirements. At the coarse-grained level, we aim to filter news articles that contain interesting events, targeting the users who look for sensitive events and prefer to manually analyse the filtered articles to obtain event details without possible machine errors. Briefly, given a set of news articles, the targeted system needs to recognise the event and not-event articles. Based on the Global Contentious Politics Dataset (GLOCON)[2] annotation manual, we define an event article more comprehensively using the Definition 6.

Definition 6 *Event Article:* An article that solely mentions or contains the details of an event(s), covering actors, time and location.

At the fine-grained level, we aim to extract event sentences and words from event articles targeting the users who prefer to follow a fully automated process to obtain comprehensive event information. Initially, we target recognising whether a sentence is an event sentence or not, analysing all the sentences

[2]Details of GLOCON are available on https://glocon.ku.edu.tr/

in an event document. Following Automatic Content Extraction (ACE) Program[3] and GLOCON annotation manuals, we use Definition 7 to introduce the idea behind event sentences concisely.

Definition 7 *Event Sentence:* A sentence that describes an event or contains an expression (word or phrase) which directly refers to an event.

Then we target extracting event words (triggers and arguments), analysing the token level of event sentences as the finest-grained information. Previous research treated event trigger and argument extraction as separate (Chen et al., 2015; Yang et al., 2019a) as well as joint (Lin et al., 2020; Awasthy et al., 2021) tasks. Considering the recent applications, benefits of mutual learning and resource limitations, we aim to build a joint system in this research. We define an event trigger and argument using Definitions 8 and 9, following ACE and GLOCON manuals.

Definition 8 *Event Trigger:* The main word that most clearly expresses an event occurrence.

Definition 9 *Event Argument:* An entity, temporal expression, or value that serves as a participant or attribute of an event.

In summary, we aim to develop approaches to automatically extract coarse-grained (event articles) and fine-grained (event sentences and words) event details from news articles in this part of the research. We target maintaining the generalisability of our approaches to easily support different languages, including low-resource languages and diverse domains.

[3]Details of ACE are available on https://www.ldc.upenn.edu/collaborations/past-projects/ace

6.2 Datasets

To conduct the experiments and evaluations, we used the multilingual version of the GLOCON gold standard dataset (Hürriyetoğlu et al., 2021b), which was released by the CASE 2021 workshop (Hürriyetoğlu et al., 2021a), considering its recency, open-availability and coverage. Section 6.2.1 reports more details about the data coverage, sizes and distributions of the GLOCON dataset. The data cleaning steps we followed are described in Section 6.2.2.

6.2.1 Data Collection

The GLOCON dataset targeted socio-political events covering demonstrations, industrial actions, group clashes, political violence, armed militancy and electoral mobilisations. Multiple news sources were used to collect data from four languages: English, Portuguese, Spanish and Hindi, at different levels of granularity: document, sentence and token. For simplicity, we refer to these languages using their ISO 369-1 codes[4] in the following content.

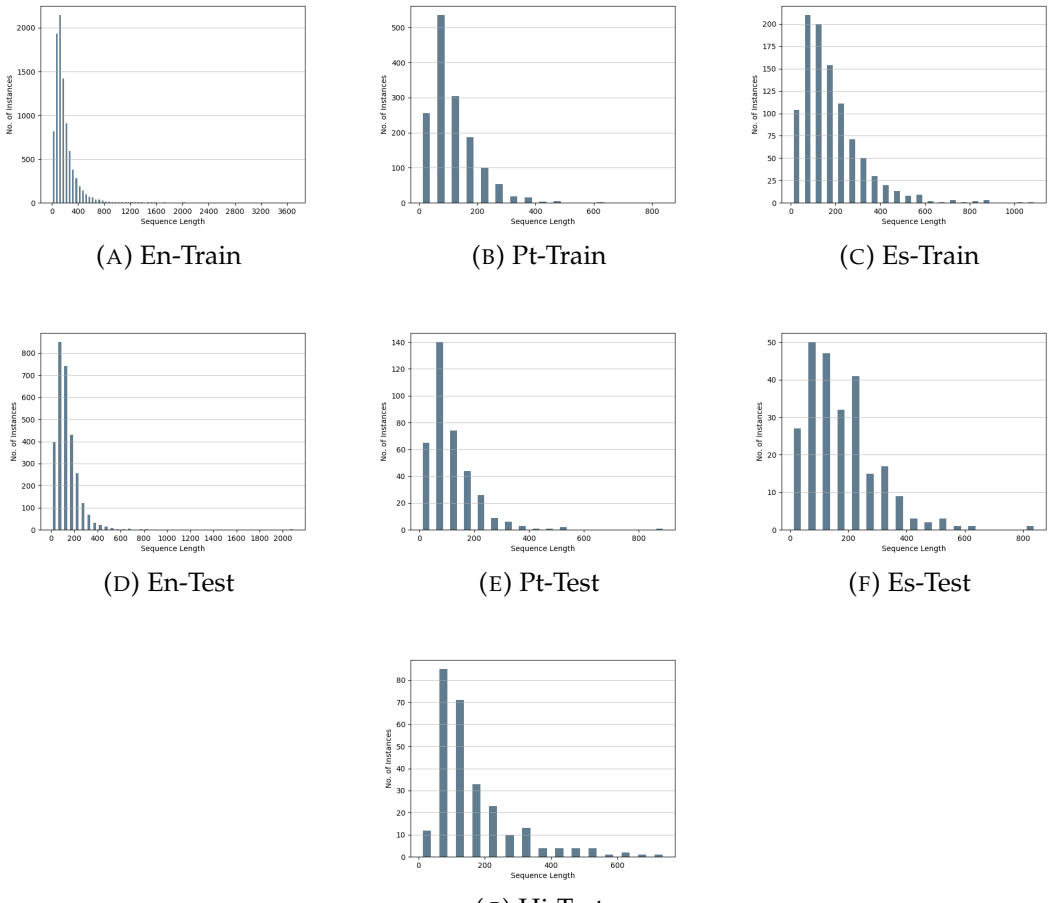
TABLE 6.1: Number of instances/samples in different data granularities. An instance contains a document at the document level and a sentence at the sentence and token levels.

Language	Document Level		Sentence Level		Token Level	
	Train	Test	Train	Test	Train	Test
English (En)	9313	2971	22481	1290	3248	311
Portuguese (Pt)	1485	372	1001	1445	87	192
Spanish (Es)	994	250	2613	686	87	190
Hindi (Hi)	-	268	-	-	-	-

For the coarse-grained level experiments (event article identification), we used the document level data of the GLOCON dataset. These data consist of an identifier, document text and binary label, which indicates whether that

[4]Language codes are available in ISO 369-1 Registration Authority Website on https://www.loc.gov/standards/iso639-2/php/code_list.php

particular article describes/contains an event or not, per instance. Only one-third of document text from the beginning of each document was available with this dataset to respect the news sources' copyright and prevent possible copyright issues. We followed a few language-independent steps to clean the data, as mentioned in Section 6.2.2. The sizes of cleaned data are summarised in Table 6.1. Overall, the document level data contained train and test sets for three languages (English, Portuguese and Spanish) and only a test set for Hindi. Additionally, the distribution of document sequence lengths (i.e. the number of tokens per document) in each split is illustrated in Figure 6.2.



(A) En-Train (B) Pt-Train (C) Es-Train

(D) En-Test (E) Pt-Test (F) Es-Test

(G) Hi-Test

FIGURE 6.2: Sequence length histograms of train and test splits of document level data

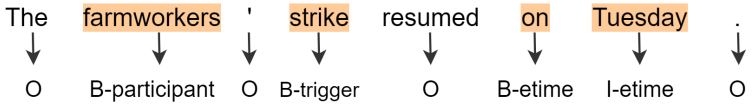
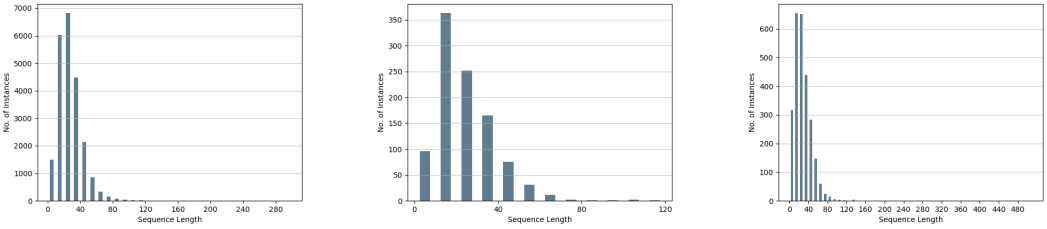


FIGURE 6.3: Sample of token level labels in BIO format

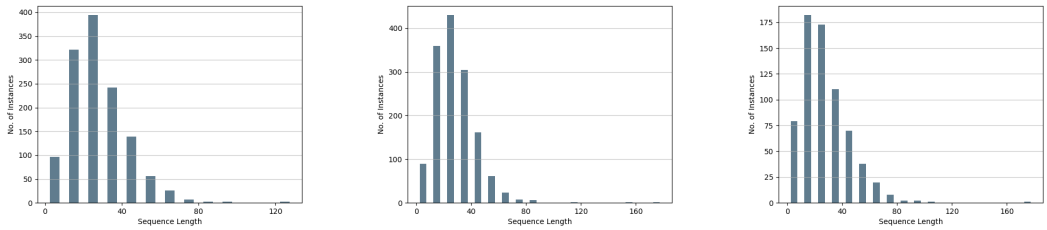
For the fine-grained level experiments (event sentence and word extraction), we used sentence and token level data of the GLOCON dataset. The sentence level data contained an identifier, sentence text and binary label, which indicates whether that particular sentence describes/contains an event or not, per instance. The token level data were formatted into BIO (Beginning, Inside, Outside) format, which is considered the standard for information extraction tasks (Ramshaw and Marcus, 1995), based on event triggers and arguments, as shown in the sample in Figure 6.3. The data cleaning steps we followed with sentence and token level data are described in Section 6.2.2. Both of these levels contained data from three languages, excluding Hindi. The data distribution of cleaned datasets over these languages is summarised in Table 6.1. Furthermore, we illustrate the sequence length distribution of sentences (i.e. the number of tokens per sentence) in Figure 6.4. This provides an overview for both sentence and token level data because the token level was composed using a subset of sentence level data.

Overall, document and sentence levels have a higher number of instances/labelled samples than the token level, mainly due to the data annotation complexities at the token level. Considering the languages, comparatively, English has more instances than others at all the granularities explaining its wide usage and data availability. Thus, for the targeted tasks, we consider English as a high-resource language and others as low-resource languages.

The class distributions over document and sentence level data are shown in Figure 6.5. As can be seen, there are more non-event documents/sentences

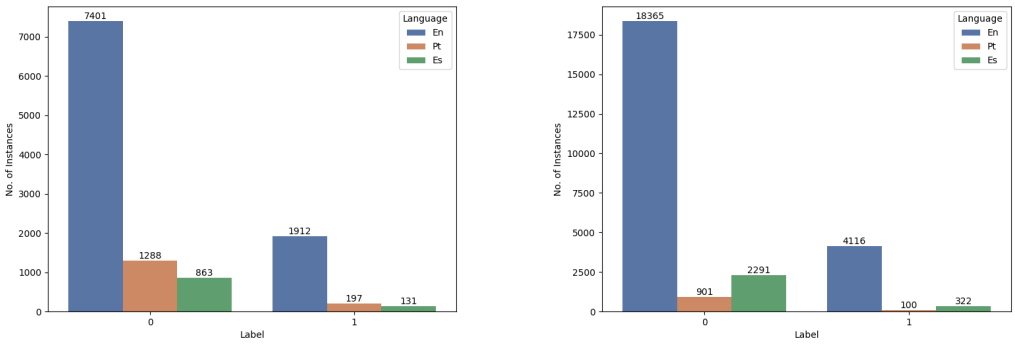


(A) En-Train (B) Pt-Train (C) Es-Train



(D) En-Test (E) Pt-Test (F) Es-Test

FIGURE 6.4: Sequence length histograms of train and test splits of sentence level data



(A) Document level (B) Sentence level

FIGURE 6.5: Label distribution of document and sentence level data

than event documents/sentences. Since this imbalance depicts the real scenario and provides more training samples from the targeted domain to the models, we directly experimented with these data without any pruning. Token level data were provided with labels indicating event triggers and arguments. Overall, there are six argument types, and the details of their distributions are given in Table 6.2. More details about the classes and the annotation processes

TABLE 6.2: Label distribution of token level data. Spans are the text spans/ordered sequences of tokens corresponding to each label.

Label	**Number of Spans**		
	En	**Pt**	**Es**
trigger	4595	122	127
participant	2663	73	79
place	1570	61	14
target	1470	32	52
organizer	1261	19	23
etime (event time)	1209	41	32
fname (facility name)	1201	48	39

are available with GLOCON annotation manuals[5]. We only represent the class distributions in training datasets in the diagrams mentioned above because test data labels were not released when we conducted our experiments. We used the CodaLab pages[6] set up by CASE 2021 task organisers to evaluate our results.

6.2.2 Data Cleaning

We applied a few language-independent techniques to clean the data from the GLOCON dataset. For document level data, since documents cannot be very short, we removed the documents with very low sequence lengths (<5 tokens). Also, we removed the URLs and replaced the repeating symbols more than three times (e.g. =====) with their three occurrences (e.g. ===) because they are uninformative. All these removals were automated using text pattern matching based on regular expressions. Analysing the sentence and token data, we noticed some instances shared among training and testing splits

[5]Details of GLOCON are available on https://glocon.ku.edu.tr/, and annotation manuals can be directly accessed from https://github.com/emerging-welfare/general_info/tree/master/annotation-manuals

[6]CodaLab page for Shared Task on Multilingual protest news detection CASE 2021 is available on https://competitions.codalab.org/competitions/31247. Its additional scoring page is available on https://competitions.codalab.org/competitions/31639

of these levels. Since such occurrences could affect the performance of our approach, we removed those instances from the training splits. For example, if a sample in token level test data split is available in sentence level train data split, we removed it from the sentence level train split. Also, we removed URLs and repeating symbols from the sentence level data, following the same approach as with the document level, considering their uninformativeness. We did not apply any further processing for token level data, which were already cleaned.

6.3 Evaluation Metrics

To evaluate the performance of event detection approaches designed for news media, we used different variants of the F1 score, which are appropriate for document, sentence and token levels, following CASE 2021 event detection shared task (Hürriyetoğlu et al., 2021a). Generally, F1 is calculated as the weighted harmonic mean of precision and recall as a positively oriented score. In the below equations, TP, FP and FN refer to the true positive, false positive and false negative counts, respectively.

$$Precision = \frac{TP}{TP + FP} \tag{6.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{6.2}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{6.3}$$

We used macro averaged F1 (Macro F1) for document and sentence level evaluations. It is the unweighted mean of F1 scores calculated for each label as in Equation 6.4. n represents the total number of classes, and $F1_i$ represents the

per-class $F1$.

$$Macro\ F1 = \frac{\sum_{i=1}^{n} F1_i}{n} \tag{6.4}$$

We used the F1 measure introduced in Conference on Computational Natural Language Learning (CoNLL) 2003 shared task (Tjong Kim Sang and De Meulder, 2003) for the token level evaluations. This score also follows the Equation 6.3 but considers text spans/ordered sequences of tokens and their labels to compute TP, FP and FN values. It marks a span as correct only if it exactly matches the actual label/labelled span.

6.4 Theoretical Background

We adapt approaches utilising transformer-based language models for news media event detection in this research, considering their recent successful applications in many NLP tasks including question answering (Devlin et al., 2019; Yang et al., 2019b), offensive language identification (Ranasinghe et al., 2019; Ranasinghe and Hettiarachchi, 2020), machine translation (Ranasinghe et al., 2020) and named entity recognition (Liang et al., 2020).

6.4.1 Transformer-based Language Model

Transformer architecture was originally proposed to overcome the limitations in Recurrent Neural Network (RNN)-based sequence-to-sequence models. Due to the recurrence and sequential nature of RNNs, they fail to process long sequences well while focusing on important words. Attention mechanisms have become popular, targeting this limitation, considering their ability to model dependencies without relying on their distances in the input and output sequences. Following this trend, transformer architecture was proposed solely based on the attention mechanisms (Vaswani et al., 2017). Later research

suggested separating the encoder of the transformer architecture to use it as a language model which generates contextual text representations (Radford et al., 2018; Devlin et al., 2019).

Among the transformer-based language models, Bidirectional Encoder Representations from Transformers (BERT) model (Devlin et al., 2019) became popular recently, mainly due to its bidirectional nature, which improved the performance of many NLP tasks. This bidirectional architecture captures left-to-right and right-to-left relationships in text, incorporating context from both sides. Thus, it can generate linguistically powerful contextual language representations and perform well on both sentence and token level tasks. Following BERT, more architectures such as DistilBERT (Sanh et al., 2019), RoBERTa (Liu et al., 2019b), multilingual-BERT (mBERT) (Devlin et al., 2019) and XLM-RoBERTa (XLM-R) (Conneau et al., 2020) were proposed recently, involving knowledge distillation, robust learning techniques and multilinguality/cross-linguality.

The general architecture of a transformer-based language model is shown in Figure 6.6, and we will refer to such model as a *'transformer'* for the simplicity. It mainly consists of an input layer, multi-layer bidirectional transformer encoder and output layer. Being an encoder, it takes a text sequence as the input and returns its representations/embeddings, which can use to learn downstream tasks while preserving the linguistical features of the original text.

Transformer Input Format: Allowing to handle various downstream tasks, transformers are designed to take a single text sequence or a pair of sequences as the input. Different special tokens such as [CLS] and [SEP] are used to indicate the input text's organisation. [CLS] is added as the first token. If two sequences exist in the input, [SEP] is placed in between to indicate the separation. Following the raw text formattings, the text needs to convert to a token

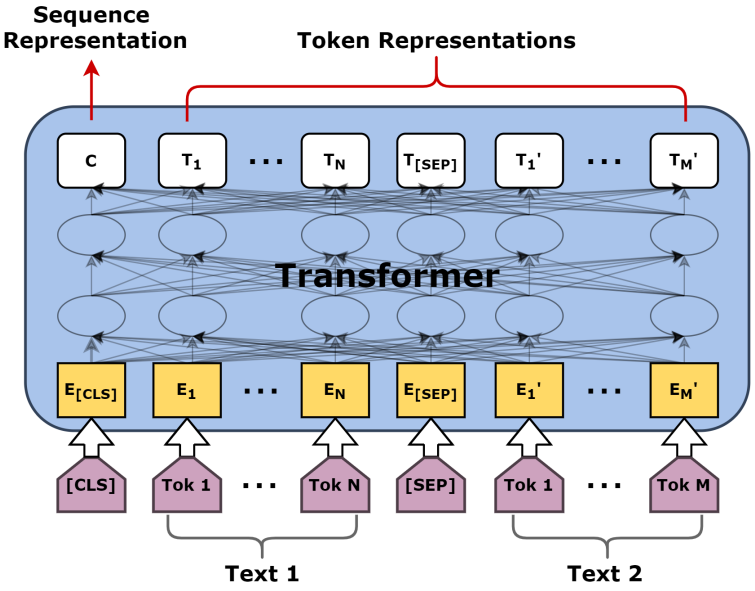


FIGURE 6.6: Transformer architecture

embedding using a tokeniser. Additionally, a segment embedding that holds boolean values (0 and 1), separating the segments and a position embedding with increasing numbers from 0, indicating the token positions are required to populate the final input. The sum of these three embeddings forms the input to a transformer model (Figure 6.7).

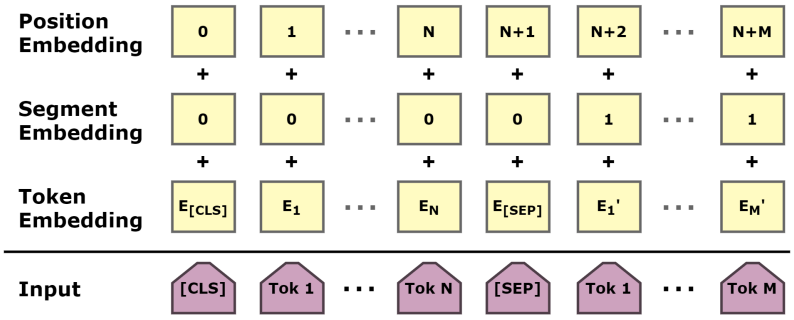


FIGURE 6.7: Transformer input representation

Transformer Output Format: The final hidden state of a transformer encoder provides representations for each token in the input. The first token ([CLS])'s output holds a representation corresponding to the entire sequence, which can

be used as a contextual sequence embedding or with sequence-based predictions. The other outputs contain token representations per input token, which can be used as contextual word embeddings or with token-based predictions.

Transformers are designed with two training steps: pre-training and fine-tuning. During pre-training, model trains on unlabelled data over different tasks to build the language model. The commonly used tasks for pre-training are masked language modelling (MLM) and next sentence prediction (NSP).

- ***MLM:*** MLM randomly masks some percentage (e.g. 15%) of the input tokens and trains the model to predict those masked tokens. For predictions, the output vectors of the masked tokens are fed into a softmax layer over the vocabulary. Since this task trains the model to predict targeted words in a multi-layered context focusing on both directions, it helps to learn bidirectional representations.

- ***NSP:*** NSP is a binary task which predicts whether two sentences are consecutive in a monolingual corpus. When selecting sentence samples, equal weight is given to both classes by having 50% of consecutive and random/non-consecutive sentences. This task is mainly involved in helping the model understand relationships between sentences, which are not directly captured by language modelling, targeting downstream tasks such as question answering and natural language inference.

Fine-tuning typically happens based on a downstream task. Depending on the task, an appropriate additional layer(s) like a classification head must add to the top of the output layer. During the fine-tuning, model initialises first with its pre-trained parameters and fine-tunes all its parameters, including the parameters in the newly added layers, using the labelled data from the downstream task. Initialisation with pre-trained parameters transfers the model's

original knowledge, allowing to learn the downstream task effectively. Deviating from this idea, some research used transformer embeddings to input text into different ML and DL models (Balali et al., 2020; Lu et al., 2022) or freeze the weights in the transformer during the fine-tuning and only train the parameters of newly added layers (Büyüköz et al., 2020). However, overall, fine-tuning the transformer along with the additional layers performed best for downstream tasks, as it customises a generic language model for a specific task or domain data in addition to fine-tuning the direct task-related layers (Büyüköz et al., 2020; Merchant et al., 2020). Thus, we also follow the complete fine-tuning process in this research to adapt pre-trained transformers for news event detection.

6.5 Summary

Automated intelligent mechanisms are crucial to effectively extract information from news media, mainly considering the high generation and unstructured nature of data. Further emphasising this fact, previous research has proposed various approaches for this information extraction task, ranging from traditional machine learning to deep learning, capturing different levels of information. However, we noticed that most available approaches focus only on high-resource languages such as English. Also, there was less focus on handling text ambiguity which plays a key role in extracting information/events from the text. Targeting these gaps, in this research, we aim to develop methods based on transformers to detect events at different levels of data granularity in news media, introducing new research directions. We mainly involve transformers in our approaches due to their knowledge transferability, cross-linguality and context awareness helpful for making accurate predictions for high- and low-resource languages while resolving text ambiguities effectively.

Our focus is also motivated by the transformer-based models' state-of-the-art performance in many NLP applications.

In the next few chapters, we describe our approaches to news media event detection. Similar to Part I of this thesis, we mainly aimed at two levels of granularity (coarse and fine) while designing our approaches considering different user requirements. Chapter 7 describes the coarse-grained approach and Chapter 8 describes the fine-grained approach. At the coarse-grained level, we target recognising event-described news articles, allowing users to take further actions of their choice, such as manual analysis to extract event details. At the fine-grained level, we target extracting event-described sentences and words from event-described articles to automate the complete event detection process.

Chapter 7

TRED: Coarse-grained Level – Event Article Identification

As described in Chapter 6, this research focuses on detecting news media events based on two levels of data granularity, coarse and fine, considering different information requirements. Mainly, coarse-grained level detection targets to notify users about event occurrences and fine-grained level targets to extract event-described text segments at event occurrences. As the coarse-grained level of news media event detection, we focus on identifying event-described news articles or event articles. Due to the high volume of news media data generation in terms of article count and length, it is impractical to manually go through all available articles to filter the ones with interesting events. Also, the requirement to carefully go through all the content to understand the context described in an article to resolve language ambiguities makes this task further complex. Considering all these associated complexities, to fulfil the coarse-grained level requirement of news media event detection, in this chapter, we propose a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformer models to automate event article identification. We also analyse how the proposed architecture can be utilised for cross-lingual predictions. The findings reported in this chapter have been published in (Hettiarachchi et al., 2021a).

TRED is mainly motivated by the recent successful application of transformers in many NLP tasks (Section 6.4). Transformer-based approaches obtained state-of-the-art performance in many tasks, mainly due to their knowledge transferability and context awareness. However, transformers were not popularly involved in document classification tasks due to their inability to process long sequences. Overcoming this limitation, sparse attention mechanisms were introduced recently along with long-sequence transformers (Beltagy et al., 2020; Zaheer et al., 2020). We propose using these transformers for event article identification along with TRED. We also investigate how pre-trained transformer models can be effectively utilised in cross-lingual event article identification, reporting a comprehensive experimental study covering four languages: English, Portuguese, Spanish and Hindi. We involve different pre-trained transformers, including long-sequence, monolingual and multilingual models for our experiments, along with several popularly used learning strategies: monolingual, multilingual, zero-shot and transfer learning. We mainly target analysing the impact by input sequence length, cross-linguality of the transformer model and involved data on high- and low-resource language predictions in event article identification.

To the best of our knowledge, this is the first study to analyse the performance of long-sequence transformers in event article identification. This claim is further supported by the results of *Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE) 2021-Task 1*. In summary, the main contributions of this chapter are as follows.

1. We propose a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformers for event-described news article identification.

2. We empirically evaluate how the performance of event article identification can be improved for different languages involving transformer models and different learning strategies, answering the following research questions:

   RQ1: Do higher input sequence lengths always improve the performance of transformer-based event article identification?

   RQ2: Can a multilingual transformer, which is only fine-tuned for a particular language, outperform a monolingual transformer of that language in event article identification?

   RQ3: Can a high-resource language improve the event article identification performance of a low-resource language using the cross-linguality in transformer models?

3. Our approach won subtask 1 of CASE 2021-Task 1 for the English language while being within the top four solutions for other languages: Portuguese, Spanish and Hindi out of 13 teams[1].

4. We release the implementation of our approach as an open-source project to support related research and applications[2].

The rest of this chapter is organised as follows. Section 7.1 introduces TRED for event article identification along with transformer models. Section 7.2 describes the experimental setup we used for our experiments. Section 7.3 comprehensively describes the conducted experiments and obtained results, along with discussions which address the targeted research questions. Finally, Section 7.4 summarises the conclusions with aimed future work.

[1]More detains about the shared task and leader board are available on https://competitions.codalab.org/competitions/31247

[2]Our implementation is publicly available on https://github.com/HHansi/EventMiner

7.1 Methodology - TRED

We propose a *TRansformer-based Event Document classification architecture (TRED)* to identify event-contained news articles based on the sequence classification architecture described in Section 7.1.1. Even though transformers have been used in a wide range of NLP applications as mentioned in Section 6.4, they are not popularly involved in document classification tasks due to their inability to process long sequences. Overcoming this limitation, long-sequence transformer models were proposed recently. However, to the best of our knowledge, these models have not been used with news media event detection due to their recency, and our study is the first to analyse their performance in this domain. We also use multilingual transformers with TRED to analyse its performance at cross-lingual identifications. More details about the transformer models and learning techniques we used are described in Section 7.1.2 and 7.1.3.

7.1.1 Transformer-based Sequence Classifier

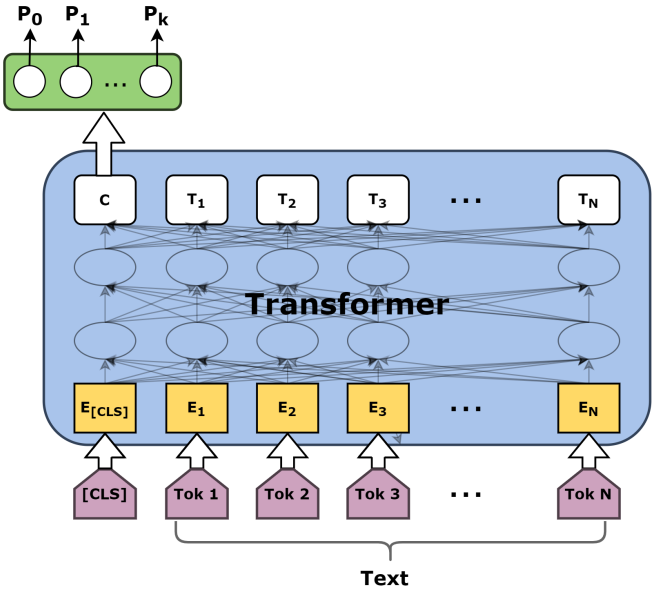


FIGURE 7.1: Transformer-based sequence classification architecture

We treat event article identification as a sequence classification problem, considering the text in an article as a sequence. We propose using the transformer-based architecture shown in Figure 7.1 for sequence classification. Requiring to process a single sequence per instance, we only need the special token [CLS] without [SEP] to format the inputs for this architecture (Section 6.4.1). To perform the classification task (or fine-tuning), we feed the final hidden state of [CLS] token, which represents the entire sequence to a softmax layer. A softmax layer contains k neurons equivalent to the number of classes targeted by the classifier. Each neuron follows the softmax activation function in Equation 7.1 returning probabilities per class (P_i). z_i and z_j represent input and output vectors. After calculating the class probabilities, we pick the class with maximum probability as the final prediction.

$$P_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{7.1}$$

7.1.2 Transformer Models

Typically, most of the transformer models, such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019b), can only process sequences up to the length of 512. This limitation is mainly introduced due to the full self-attention operation used by these architectures, which scales quadratically with the sequence length (Beltagy et al., 2020). However, this could negatively affect event article processing considering the high possibility of having lengthier articles. Thus, we mainly focus on the recently released improved models: Longformer (Beltagy et al., 2020) and BigBird (Zaheer et al., 2020), which can process longer sequences with TRED.

Longformer (Beltagy et al., 2020): Longformer architecture uses a sparse attention mechanism, which scales linearly with the sequence length, reducing

the time and memory complexities in the full self-attention. The proposed mechanism is a combination of local windowed and global attention. Two variants of local attention named sliding window and dilated sliding window were used, focusing on local and longer contexts. The global attention was added to a few locations for learning task-specific representations, which cannot capture using windowed attention. Using this sparse mechanism, the Longformer model can process sequences up to 4,096 or 8x of the length supported by previous transformers. Also, the original study's experiments showed that Longformer could learn long-range contexts effectively.

BigBird (Zaheer et al., 2020): BigBird architecture also uses a sparse attention mechanism with complexity linear to the number of tokens. This is a combination of random, window and global attention mechanisms. Similar to the Longformer architecture, random blocks and sliding windows were used, targeting the local context. Global attention was involved, targeting global information. BigBird can handle sequences of up to 4,096 using similar hardware used by previous transformers, which are limited to the length of 512. Additionally, the original study's experiment results revealed that BigBird generates better contextual representations than the limited length models, such as RoBERTa, by learning from longer sequences.

However, Longformer and BigBird only support the English language currently. Therefore, to conduct multilingual experiments, we decided to use multilingual BERT (mBERT) and XLM-RoBERTa (XLM-R) models, which support limited sequences of length up to 512, considering these models' language coverage and successful applications (Karthikeyan et al., 2020; Conneau et al., 2020).

mBERT (Devlin et al., 2019): mBERT follows the same architecture and is pre-trained in the same way as the monolingual BERT model (Devlin et al., 2019). However, mBERT is pre-trained on Wikipedia text from the top 104 languages, making it a multilingual model. Some languages were sub-sampled and super-sampled, accounting for their data availability in Wikipedia while preparing the training data corpus. In addition to using a multilingual corpus, mBERT does not rely on any cross-lingual objectives or aligned data but exhibited surprising cross-lingual abilities in recent work (Karthikeyan et al., 2020).

XLM-R (Conneau et al., 2020): XLM-R architecture is proposed following the Cross-lingual Language Model (XLM) (Conneau and Lample, 2019). However, rather than using the translation language modelling (TLM) objective used by XLM to leverage parallel data to obtain cross-lingual inference, XLM-R uses MLM objective (Section 6.4.1) from the RoBERTa model (Liu et al., 2019b). The unsupervised nature of MLM allowed the model to train on a large amount of data, significantly boosting its performance. As the pre-training corpus, XLM-R used cleaned CommonCrawl data (Wenzek et al., 2020), covering 100 languages, including low-resource languages, considering the relatively limited scale in Wikipedia data. Following these slight modifications to XLM, XLM-R became the state-of-the-art for cross-lingual language understanding outperforming other models, including mBERT.

7.1.3 Learning Strategies

We involve various language-based learning strategies described below for the fine-tuning of TRED. They have been involved in different areas such as machine translation (Zoph et al., 2016), offensive language identification (Ranasinghe and Zampieri, 2020) and word sense disambiguation (Hettiarachchi and

Ranasinghe, 2021), considering their diverse characteristics. We mainly target analysing the effectiveness of each strategy in identifying event articles and the cross-linguality of the proposed architecture.

1. ***Monolingual Learning:*** Monolingual learning fine-tunes a model using data from a single language. This is the common learning strategy, and mostly, it performs well for high-resource languages with the provision of enough data to fine-tune a transformer (Devlin et al., 2019).

2. ***Multilingual Learning:*** Multilingual learning fine-tunes a model in multiple languages simultaneously, using a training dataset composed of different languages. This strategy can supply more training data to the model, overcoming data scarcity in low-resource languages (Ranasinghe et al., 2020). Also, in general, this learning can help optimise the model effectively for different languages capturing their interconnections, unlike monolingual learning. Additionally, a model that supports multiple languages is more resource-effective and easily manageable than a monolingual model collection. However, this learning is only applicable to multilingual transformers.

3. ***Language-based Zero-shot Learning:*** Language-based zero-shot learning uses a model fine-tuned for the same task in another language(s) to make predictions. This is more useful in scenarios where no training data are available for a particular language and is especially beneficial for low-resource languages (Ranasinghe et al., 2020; Hettiarachchi and Ranasinghe, 2021). The cross-lingual abilities of the transformer model play a crucial role in the success of this strategy.

4. ***Language-based Transfer Learning:*** Language-based transfer learning is a variant of transfer learning (TL) (Weiss et al., 2016) that transfers knowledge from one language to another. This strategy fine-tunes a model learned in a particular language for the same task in another language. Popularly, models trained on high-resource languages are fine-tuned for low-resource languages following this idea (Zoph et al., 2016; Ranasinghe and Zampieri, 2020).

7.2 Experimental Setup

This section summarises the experimental setup used with TRED for event article identification. We used the document level data of the GLOCON gold dataset, which is introduced in Section 6.2, for our experiments. This dataset has data from four languages: English (En), Portuguese (Pt), Spanish (Es) and Hindi (Hi), and we consider English as a high-resource language and others as low-resource languages based on the data availability. We used the Macro F1 measure described in Section 6.3 to assess the accuracy of built models. Considering the targeted task and languages, we involved several pre-trained transformer models, including long-sequence and multilingual models with TRED, and their details are summarised in Section 7.2.1. Also, the hyper-parameter configurations we used for our experiments are available in Section 7.2.2. We followed a common convention to format training data combinations based on the learning strategies (Section 7.1.3) while reporting results as described in Section 7.2.3. We implemented the neural network architectures in Python, using PyTorch (Paszke et al., 2019) and Huggingface (Wolf et al., 2020) libraries. Our implementation is publicly available on GitHub[3]. We used a GeForce RTX 3090 GPU to conduct all experiments.

[3]The public GitHub repository is available on https://github.com/HHansi/EventMiner

7.2.1 Pre-trained Transformers

As mentioned in Section 7.1.2, we involved long- and limited-sequence monolingual and multilingual pre-trained transformer models for our experiments with TRED. We especially targeted evaluating and comparing the performance of these models in event article identification. As long-sequence models, we used Longformer (*longformer-base-4096*) (Beltagy et al., 2020) and BigBird (*bigbird-roberta-large, bigbird-roberta-base*) (Zaheer et al., 2020) pre-trained transformers. Even though we targeted using large versions of the models, we had to use Longformer's base version for all experiments and BigBird's base version for a few long-sequence experiments due to resource limitations. Also, these models only support English currently. Thus, for other languages' monolingual experiments, we used limited-sequence models, BERTimbau (*BERTimbau-large*) (Souza et al., 2020) and BETO (*BETO-cased*) (Canete et al., 2020) trained in Portuguese and Spanish, which are variants of BERT (Devlin et al., 2019). Also, we involved BERT (*bert-large-cased*) model trained in English to compare with long-sequence model performance. As the multilingual models also, limited-sequence models, mBERT (*bert-base-multilingual-cased*) and XLM-R (*xlm-roberta-large*) (Conneau et al., 2020) trained in 104 and 100 languages, including the targeted languages were involved considering the unavailability of long-sequence multilingual models. We used HuggingFace's model repository[4] (Wolf et al., 2020) to obtain all of these pre-trained transformers. Table 7.1 summarises more details about these models' design, and Table 7.2 summarises the targeted language coverage in data used to pre-train these models.

[4]HuggingFace models are available on `https://huggingface.co/models`

TABLE 7.1: Pre-trained transformer model details, including the number of trained languages (#Lgs.), maximum sequence (Max. Seq.) length, layers (L), hidden states (H_m), attention heads (A), vocabulary size (V) and total parameters (#Params). Under tokenisers, WPM and SPM refer to WordPiece (Wu et al., 2016) and SentencePiece (Kudo and Richardson, 2018) models, BPE refers to Byte Pair Encoding (Sennrich et al., 2016), and bBPE refers to byte-level BPE.

Model	#Lgs.	Tokeniser	Max. Seq. Length	L	H_m	A	V	#Params
BERT	1	WPM	512	24	1024	16	30k	335M
BERTimbau	1	WPM	512	24	1024	16	30k	330M
BETO	1	BPE	512	12	1024	16	32k	110M
Longformer(Base)	1	bBPE	4096	12	768	12	50k	149M
BigBird(Base)	1	SPM	4096	12	768	12	50k	128M
BigBird(Large)	1	SPM	4096	24	1024	16	50k	360M
mBERT	104	WPM	512	12	768	12	110k	172M
XLM-R	100	SPM	512	24	1024	16	250k	550M

TABLE 7.2: Coverage of targeted languages by data corpora used to pre-train transformer models. Token counts are in Billions, and estimations of token percentages per language compared to the full corpus are within brackets. For the mBERT model, we approximated the given values using extrapolation (Conneau et al., 2020; Lee et al., 2021), considering the unavailability of statistics.

Model	Tokens (B)			
	En	Pt	Es	Hi
BERT	3.30 (100%)	-	-	-
BERTimbau	-	2.68 (100%)	-	-
BETO	-	-	3 (100%)	-
Longformer	6.5 (100%)	-	-	-
BigBird	19.2 (100%)	-	-	-
mBERT	1.73 (28.8%)	0.18 (3%)	0.60 (10.1%)	0.07 (1.2%)
XLM-R	55.61 (18.8%)	8.41 (2.8%)	9.37 (3.2%)	1.72 (0.6%)

7.2.2 Hyper-parameters

To maintain the consistency among architectures to generate comparable results, we used a common set of hyper-parameters for our experiments. We also believe this will provide a good starting configuration for researchers willing to explore our approach further. We used the batch size of four, the learning

rate of $1e^{-5}$ with Adam optimiser and epochs of three with an early stopping patience of 10, considering the computational complexities associated with transformers. We set evaluation steps allowing 6-12 evaluations per training epoch depending on the size of the training dataset. A split of 10% from training data is used for these evaluations, and the rest is used for training. To mitigate the impact on results by the randomness associated with deep neural networks, we used the majority-class self-ensemble approach, which also helps improve the overall performance (Hettiarachchi and Ranasinghe, 2020). With this setting, per experiment, we trained three models initialised with different random seeds and took the majority vote of model predictions as the final prediction. We limit the ensemble model count to three due to the high computational complexities in processing long sequences using transformers. We varied the input sequence length during our experiments to analyse its impact on the final outcome. Thus, more details about the used sequence lengths are available in Section 7.3.

7.2.3 Training Formats

TABLE 7.3: Training data formats with learning strategies

Strategy	Format	Description
Monolingual learning	L_1	learn from data in language L_1
Language-based TL	$L_1 \rightarrow L_2$	learn the same task from data in language L_1 and then from data in language L_2
Multilingual learning	$L_1+L_2+..+L_n$	learn from data in multiple languages $L_1, L_2, ...L_n$ simultaneously

In our experiments, we involved all language-based learning strategies introduced in Section 7.1.3. For simplicity, we use a common convention to

format training data depending on the learning strategy to report our results consistently, as described in Table 7.3.

7.3 Results and Discussion

This section mainly presents evaluation results and findings of event article identification, targeting the accuracy in predictions. The transformer-based sequence classification architecture introduced in Section 7.1.1 was used with different pre-trained transformers to build models in TRED. Additionally, Section 7.3.1 discusses the efficiency and scalability of proposed models.

Initially, we evaluated the performance of monolingual classifiers. We used the monolingual transformers: BERT, Longformer, BigBird, BERTimbau and BETO to build these models based on the languages in the training data. We especially focused on analysing the effect of input sequence length on predictions. However, as mentioned in Section 7.2.1, only for English, there are pre-trained long-sequence models which can handle up to 4,096-length sequences. Thus, for other languages, we had to limit the maximum sequence length to 512 during our experiments. For English, we experimented up to the sequence length of 900, considering the sequence length distributions of targeted data (Figure 6.2). Table 7.4 summarises the results we obtained using different monolingual models with varying sequence lengths.

According to the results in Table 7.4, for En, the Longformer-based model performed better with long input sequences, but both BigBird- and BERT-based models performed better with short sequences. Overall, long-sequence models (Longformer and BigBird) outperformed the BERT-based model even with short input sequences ($\leq$512). Among long-sequence models, in the majority, BigBird made more accurate predictions than Longformer. For Pt, the BERTimbau-based model with the longer input sequence (512) performed best

TABLE 7.4: Evaluation results: Macro F1 of document classification using monolingual transformers with different input sequence lengths. The best results per language are in bold. Due to resource limitations, we had to use the base versions of the transformers for some experiments, and they are marked with *.

Lang.	Transformer	Sequence Length			
		256	512	700	900
En	Longformer	0.8168*	0.8162*	0.8184*	0.8329*
	BigBird	**0.8339**	0.8276	0.8230*	0.8225*
	BERT	0.8110	0.8096	-	-
Pt	BERTimbau	0.7405	**0.8121**	-	-
Es	BETO	**0.7214**	0.7064	-	-

with a 7.16% increment of F1, but for Es, the BETO-based model with the shorter sequence length (256) performed best with a 1.5% increment of F1. Based on our results, we answer RQ1 as follows.

RQ1: *Do higher input sequence lengths always improve the performance of transformer-based event article identification?*

During our experiments, we analysed the performance (or accuracy) of transformer-based models with varying input sequence lengths. We noticed different impacts on the final predictions by the sequence length, mainly depending on the pre-trained transformer model. For En, we compared the performance of long-sequence models (Longformer and BigBird) with the BERT model, which can handle limited sequences up to the length of 512. Comparatively, long-sequence models were pre-trained on long input sequences and large corpora (Table 7.2). Therefore, these models can effectively fine-tune for a downstream task even seeing less amount of data (short sequences), outperforming the BERT model. Comparing Longformer and BigBird, in addition to the variations in attention mechanisms, BigBird was pre-trained on a larger corpus than Longformer (Table 7.2). This helps the fine-tuning process of the

BigBird model, resulting in slightly improved F1 values than Longformer, especially with short input sequences. However, increasing the sequence length, we noticed an increment of F1 values with Longformer and a decrement with BigBird. This could mainly happen depending on the connections between pre-trained knowledge and fine-tuning data. Contrary to Longformer, the BigBird-based model gets slightly confused with increasing sequences. For limited-sequence models (BERT, BERTimbau and BETO), En and Es resulted in slightly better F1 with short input sequences (256), but Pt resulted in notably better F1 with long sequences (512). Considering the average F1 from all languages per sequence length, for 256 and 512, we get 0.7576 and 0.7761, respectively. This further emphasises that if the transformer has seen fewer data during the pre-training process, it requires to see more data (long sequences) during the fine-tuning process for effective learning.

In summary, high input sequence lengths do not always improve the performance of transformer-based event article identification. Using long-sequence models (e.g. BigBird), we can get good results with short input sequences, reducing the processing and memory requirements. However, overall, high input sequences can improve document classification performance for limited-sequence transformers such as BERT.

For multilingual experiments, we used multilingual transformers: mBERT and XLM-R, which can process up to 512-length sequences with TRED due to the unavailability of long-sequence multilingual models. Based on the findings for RQ1, we fixed the input sequence to 512 during these experiments. We involved different learning strategies introduced in Section 7.1.3 to assess how they can improve the model's accuracy on event document identification with high- and low-resource languages. While applying these strategies,

we only allowed transfer and zero-shot learning from high-resource to low-resource languages because the contrary approach is not reasonable. Table 7.5 summarises the results we obtained.

TABLE 7.5: Evaluation results: Macro F1 of document classification using multilingual transformers with different learning strategies. *Language* indicates the language of test data. The best results per language are in bold, and zero-shot learning scenarios are marked with ‡.

Strategy	**Transformer**	**Training Data**	**Language**			
			En	Pt	Es	Hi
monolingual learning	mBERT	Pt	-	0.7885	-	-
		Es	-	-	0.5768	-
		En	0.7998	0.7437‡	0.6648‡	0.5084‡
	XLM-R	Pt	-	0.4647	-	-
		Es	-	-	0.7333	-
		En	0.8245	**0.8277**‡	0.7333‡	0.8047‡
transfer learning	mBERT	En→Pt	0.7896	0.8015	0.6667‡	0.6977‡
		En→Es	0.8027	0.7633‡	0.6719	0.6090‡
	XLM-R	En→Pt	0.8184	0.8268	0.7556‡	**0.8160**‡
		En→Es	0.8095	0.7899‡	0.6947	0.7685‡
multilingual learning	mBERT	En+Pt	0.8092	0.7853	0.6595‡	0.4396‡
		En+Es	0.8120	0.7380‡	0.6760	0.3520‡
		En+Pt+Es	0.8029	0.7809	0.6889	0.5902‡
	XLM-R	En+Pt	0.8244	0.8207	0.7172‡	0.8126‡
		En+Es	0.8006	0.7599‡	0.7172	0.7944‡
		En+Pt+Es	**0.8248**	0.8028	**0.7598**	0.7948‡

According to the results in Table 7.5, most of the multilingual models fine-tuned in a particular language did not perform better than the monolingual models trained in that language (Table 7.4). However, with zero-shot learning, the XLM-R model trained on the high-resource (En) language made more accurate predictions for low-resource languages (Pt and Es), outperforming monolingual and mBERT-based models. Overall, transfer learning

a low-resource language after the high-resource language improved the results of most mBERT-based models but reduced the results of most XLM-R-based models compared to the models which only fine-tuned on En. A similar trend is also noticed with multilingual learning scenarios. However, XLM-R-based models generally outperformed the mBERT-based models in all learning strategies. Following these results, we answer RQ2 and RQ3 below.

RQ2: *Can a multilingual transformer, which is only fine-tuned for a particular language, outperform a monolingual transformer of that language in event article identification?*

To answer this question, we focus on the results in Table 7.4 and under monolingual learning of Table 7.5 except the zero-shot learning scenarios. Among the transformers we used, the mBERT model is approximately pre-trained on fewer data than that particular language data used by monolingual models and the XLM-R model (Table 7.2). We also received the lowest F1 values from mBERT-based models in our experiments. Comparing the monolingual models with the XLM-R model, XLM-R has seen more data per language during the pre-training. Thus, it has a large vocabulary size, which leads to a high number of parameters to learn during the fine-tuning (Table 7.1). Therefore, when a small amount of data is available for fine-tuning, XLM-R cannot learn the downstream task effectively, like in the scenario with Pt. However, XLM-R-based models performed on par or better than monolingual models for En and Es. These results also indicate that when a high proportion of language data has seen during the pre-training, it positively affects fine-tuning.

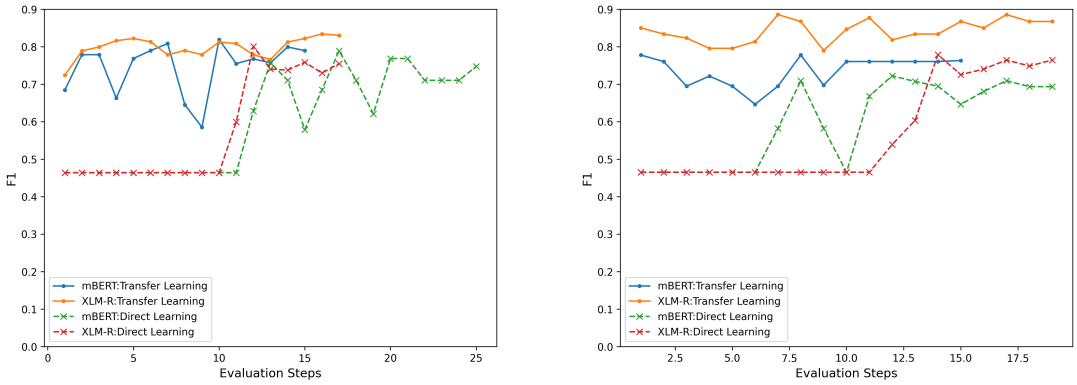
In summary, our results suggest that for a particular language, if a sufficient amount of data is provided for fine-tuning and that language covers an adequate proportion in model pre-trained data, a multilingual transformer can perform on par or better than a monolingual transformer of that language in

event article identification. The fine-tuning data amount is relative to the instance count as well as the sequence length.

RQ3: *Can a high-resource language improve the event article identification performance of a low-resource language using the cross-linguality in transformer models?*

Targeting this question, we involved different learning strategies to analyse the cross-lingual capabilities of the multilingual transformers (mBERT and XLM-R) in document classification. According to the results (Table 7.5), with zero-shot learning, the multilingual models, which only learned the high-resource language (En), performed on par or better than the models that learned corresponding low-resource languages (Pt and Es), in majority cases. The XLM-R model fine-tuned in En outperformed the mBERT-based and monolingual models that learned Pt or Es. Also, it resulted in a good F1 (≈80%) for Hi language, which does not have any training instances. These results further emphasise the high data requirement for multilingual models during the fine-tuning mentioned in RQ2's findings. Also, they reveal that by utilising the cross-linguality of transformers, effective predictions can make for low-resource languages, only learning high-resource languages.

We also analysed the impact of transfer learning a low-resource language after a high-resource language on document classification performance. Following our results, such learning is not much helpful for a high-resource language, but it improves the low-resource language performance than only learning from that particular language. This claim is further supported by the evolution of Macro F1 scores computed on validation datasets during the training process's evaluation steps. As shown in Figure 7.2, with transfer learning (TL), multilingual models obtain high F1 values much earlier than with direct learning. This indicates even with few training instances, a model can learn well using the knowledge obtained from the high-resource language and

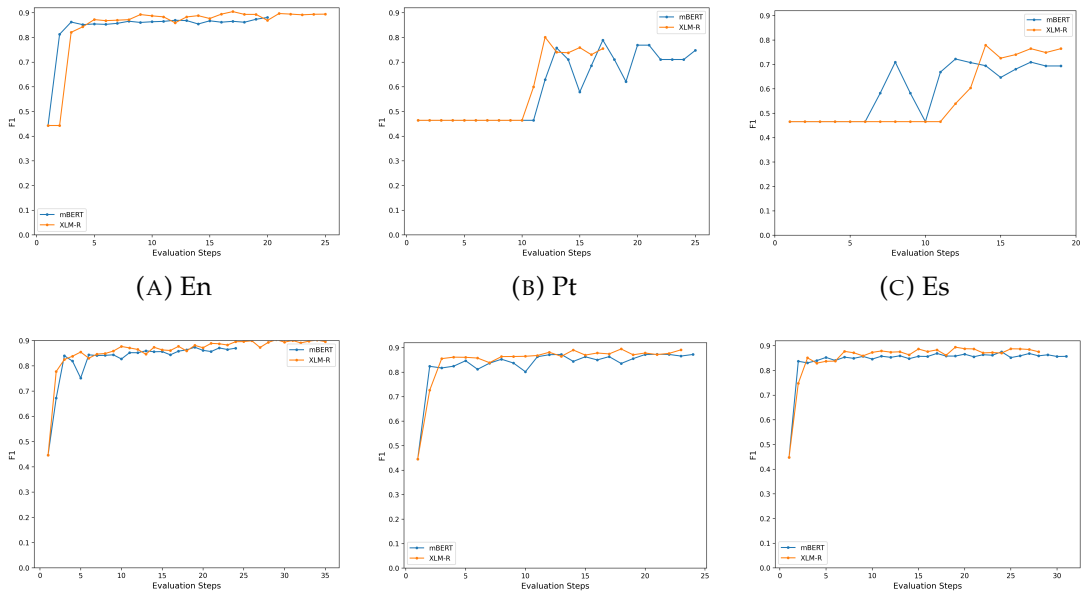


(A) Pt vs En→Pt (B) Es vs En→Es

FIGURE 7.2: Macro F1 scores for the validation sets at different evaluation steps of training processes, which involved direct language learning (Pt, Es) and transfer learning from a high-resource language (En→Pt, En→Es) with the sequence classification model with mBERT and XLM-R transformers.

the transformer's cross-lingual abilities. Also, our zero-shot learning results under TL indicate that even transfer learning another low-resource language could improve the performance of a low-resource language. Overall, compared to fine-tuning on a high-resource language, TL showed improvements in mBERT-based models (+4.61% F1 on average) and slight drops in XLM-R-based models (-1.26% F1 on average). If the low-resource language datasets are very small compared to the high-resource language data, TL will not make notable improvements on final predictions, explaining the behaviour of these XLM-R models. However, XLM-R outperformed mBERT models in all settings, emphasising its superior cross-linguality.

Furthermore, we experimented with multilingual learning, which is especially advantageous in situations where there are few training instances from low-resource languages that are insufficient for monolingual or transfer learning. It resulted in a combination of improvements and deteriorations compared to learning only a high-resource language and transfer learning a low-resource language afterwards. Especially for Hi, there are significant drops in F1 in mBERT models. Overall, the multilingual results suggest that learning



(A) En (B) Pt (C) Es

(D) En+Pt (E) En+Es (F) En+Pt+Es

FIGURE 7.3: Macro F1 scores for the validation sets at different evaluation steps of training processes, which involved monolingual (En, Pt, Es) and multilingual (En+Pt, En+Es, En+Pt+Es) learning with the sequence classification model with mBERT and XLM-R transformers. During multilingual learnings, a composition of samples from each language is used as the validation set.

from long sequences from multiple languages could sometimes confuse the model, depending on the languages' interconnections and the transformer's pre-trained knowledge. However, similar to TL, Figure 7.3 illustrates that multilingual learning also effectively helps the fine-tuning process than learning the languages separately. Also, according to the results of En and Es, multilingual learning applied XLM-R models outperformed all other combinations, emphasising the potential of this strategy.

Based on the findings, we can conclude that high-resource languages can improve low-resource languages' event article identification performance using the cross-linguality in transformer models. Overall, comparing the multilingual transformers, XLM-R is more effective in cross-lingual predictions than mBERT. Considering the learning strategies, with zero-shot learning, XLM-R

models that were trained only on a high-resource language could make effective predictions for low-resource languages, outperforming other monolingual settings. Also, transfer and multilingual learning improved the model learning process more than learning low-resource languages separately. However, sometimes, combined languages could also slightly confuse the model depending on the fine-tuning data and the transformer's original knowledge, as seen in our results.

We submitted the best results obtained using our approaches to subtask 1: *Document Classification* of CASE 2021 shared task 1: *Multilingual Protest News Detection* (Hürriyetoğlu et al., 2021a). The official results of the competition show that our approach won the first place in the English language while being within the top four solutions for other languages: Portuguese (rank-2), Spanish (rank-4) and Hindi (rank-3)[5]. These results further emphasise the novelty and effectiveness of our approach for event article identification.

7.3.1 Efficiency and Scalability

In addition to being accurate, event detection approaches need to be efficient and scalable for large data volumes (Section 1.1.4). The efficiency of a classification model mainly relies on the inference speed as the training/fine-tuning can be done offline without interfering with its later usage. We analysed the inference speed of each transformer-based model, and obtained results are reported in Appendix B.1. In summary, our analysis indicated that on a GPU (GeForce RTX 3090), all the built models make predictions within less time than a second. On a CPU (Intel(R) Xeon(R) @ 2.30GHz), the process took a little longer but was limited to a maximum of 7 seconds with an average of 4.12 seconds, emphasising the efficiency of the models.

[5]The leader board of CASE 2021 shared task is publicly available on https://competitions.codalab.org/competitions/31247

There is no direct impact of increasing data volumes on an already trained classifier. However, classifiers should be fast enough to make predictions on large data volumes without any considerable delay. According to the efficiency analyses done on a GPU and CPU, our models are sufficiently fast to handle a query within a very short period. Also, these experiments indicated that the inference speed could be improved more by increasing the machine's computational power. Furthermore, it is also possible to have multiple model instances to parallelly process more data, considering the models' memory utilisation (Appendix B.1). Overall, these findings suggest the built models are sufficiently scalable for handling large data volumes.

7.4 Conclusions

In this chapter, we proposed a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformer models to identify event-described news articles. We especially involved transformers, considering their transferability, context awareness and state-of-the-art performance in many NLP tasks. We also analysed how the proposed architecture can be utilised for cross-lingual event article identification on high- and low-resource languages, using different pre-trained transformer models and learning strategies, opening wider research avenues.

We used the document level data of the GLOCON gold standard dataset and several monolingual and multilingual pre-trained transformers, which support long sequences as well as limited sequences of length 512 for our experiments. We submitted the best results we obtained to subtask 1 of CASE 2021 shared task 1, and they won first place in English and ranked within the top four solutions for Portuguese, Spanish and Hindi, emphasising the

novelty and effectiveness of our approach. Our results show high input sequence lengths do not always improve the performance of transformer-based event article identification, answering RQ1. This finding suggests the sequence length should be picked mainly depending on the pre-trained transformer in addition to the targeted task. Since low sequence lengths reduce the model's computational requirements during the training, this finding will also help to build resource-efficient and effective models. Also, our experiments indicate if sufficient data exist for fine-tuning and the corresponding language covers an adequate proportion of the transformer's pre-trained data, a multilingual model can outperform a monolingual model, addressing RQ2. Following RQ3, our experiments reveal that high-resource languages can improve low-resource languages' event article identification performance using the cross-linguality in transformer models, especially with zero-shot and multilingual learning. Among the multilingual transformers we involved, XLM-R outperformed mBERT models in most cases, emphasising its effectiveness in cross-lingual predictions. These findings will be beneficial from the viewpoint of real applications because a multilingual event detection model can support multiple languages in a more resource-efficient manner than maintaining a set of monolingual models per language. Also, they provide a basis for possible language-based extensions in this area, combining high- and low-resource languages.

Overall, the high F1 measures (>75%) received in four languages, despite the training data availability, prove the built models' accuracy and expandability to different languages. Also, the approach proposed to build models is independent of any domain- or language-specific features, being able to be applied to any data, further emphasising its expandability. In terms of efficiency, inference speeds of all models are sufficiently fast to process data as they appear via news media. Furthermore, models' resource utilisation and speed

indicate that they can scale successfully to support efficient predictions on an increased data volume, fulfilling all the essential requirements of an event detection mechanism.

By the time we conducted our experiments, there were no pre-trained multilingual long-sequence transformers available, but recent research released a multilingual longformer (Sagen, 2021), which can be utilised in future experiments. Using this multilingual long-sequence transformer, we plan to analyse the performance of TRED on cross-lingual predictions. Also, we aim to extend our research to more languages, covering high- and low-resource languages and analyse how well TRED can handle the predictions. It is also worth exploring how the interconnections between languages can be utilised to improve long-sequence predictions.

Following the details on coarse-grained level event detection in news media in this chapter, we report our approaches for fine-grained level event detection in the next chapter. At the coarse-grained level, we mainly targeted notifying users about events by identifying event articles using a transformer-based approach (TRED) involving long-sequence transformers. At the fine-grained level, we target extracting event-described text segments (sentences and words) from event articles to provide detailed event information to users using an automated process, as described in Chapter 8.

Chapter 8

TTL: Fine-grained Level – Event Sentence and Word Extraction

In Chapter 7, we proposed *TRED* using long-sequence transformers to notify users about events by identifying event-described news articles or event articles, covering the coarse-grained level of news media event detection. After identifying event articles, users need to manually go through the content of these articles to extract important event information. However, it is helpful to develop approaches for fine-grained level detection, which extracts event-described text segments following the coarse-grained level detection, to automate the complete event detection process. As the fine-grained level of news media event detection, we target extracting event-described sentences (event sentences) and words (triggers and arguments) from event articles. For example, Figure 8.1 shows a non-event and event sentence from an event article with corresponding event words. Event sentence extraction helps to locate the important sentences within an article/document so that they can be further processed to extract event triggers and arguments. Following this idea, fulfiling the fine-grained level requirement of news media event detection, in this chapter, we propose a novel learning strategy named *Two-phase Transfer Learning (TTL)* based on transformers, which can transfer knowledge between tasks in different data granularity (i.e. sentence or token), to extract event sentences

and words with mutual support. We also analyse how the proposed strategy can be utilised for cross-lingual predictions. The findings reported in this chapter have been published in (Hettiarachchi et al., 2023a).

(1) Ipid spokesman said he was not immediately aware of the case, but would respond later in the week.

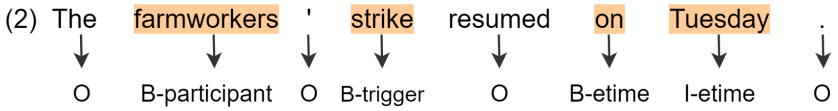


FIGURE 8.1: Sample sentences from a news article which was recognised as an event article. The first sentence does not describe the details of the targeted protest event, but the second sentence describes the highlighted details (event trigger and arguments: participant and event time).

Similar to the coarse-grained level approach, our approaches for fine-grained extractions are also mainly motivated by the recent success of transformers (Section 6.4). Transformer-based approaches have been proposed for sentence and word/token level event extraction tasks recently, setting the state-of-the-art performance (Section 2.3). However, to the best of our knowledge, all the available approaches considered sentence and token level detection as two separate tasks and built separate models per task, ignoring their interconnections, which would be helpful for mutual learning. Targeting this gap, we propose TTL, which allows a transformer to learn a task, following another related task in different data granularity (i.e. sentence or token), transferring knowledge from the first task. We apply this learning for sentence and token level event detection tasks involving different pre-trained transformer models and analyse their performance and involved tasks' transferability.

We also investigate how pre-trained transformer models can be effectively used in cross-lingual event sentence and word extraction, reporting a comprehensive experimental study covering three languages: a high-resource language (English) and two low-resource languages (Portuguese and Spanish) (Section 6.2). We involve different pre-trained transformers, including

monolingual and multilingual models for our experiments, along with several language-based learning strategies: monolingual, multilingual, transfer and zero-shot learning. We further extend these analyses with TTL to investigate its performance with other language-based learning strategies. Figure 8.2 illustrates a summary of learning strategies we devised in this study, including the explored applications using different data types. To maintain simplicity, we did not include zero-shot learning in this diagram because it is applicable to all other strategies.

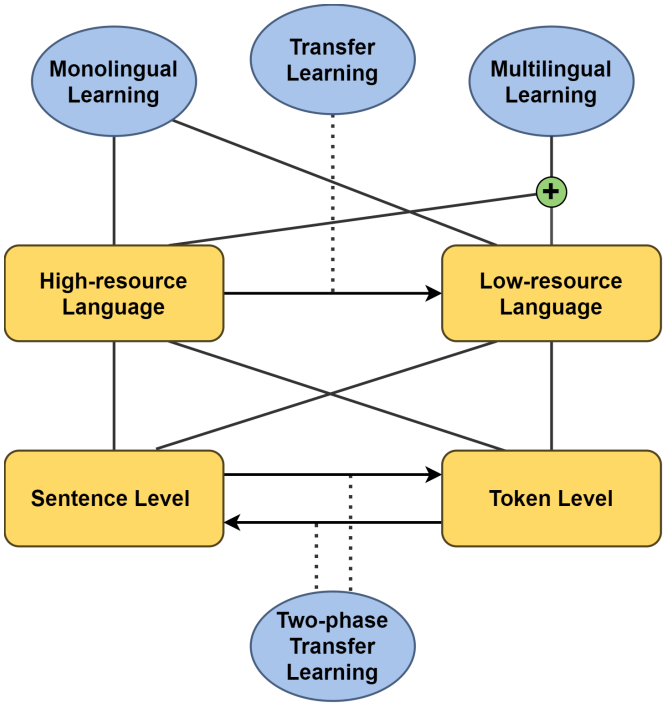


FIGURE 8.2: Learning strategies involved in this study

To the best of our knowledge, this is the first attempt to transfer knowledge from different data granularities (i.e. sentence and token levels) and analyse its performance on event sentence and word extraction in news media. Also, we report the first comprehensive experimental study on cross-lingual event detection at the sentence and token levels, as far as we are aware. In summary, the main contributions of this chapter are as follows.

1. We propose a novel learning strategy named *Two-phase Transfer Learning (TTL)*, involving different levels of data granularity and the capabilities of state-of-the-art transformer models.

2. We apply the proposed strategy to sentence and word/token level tasks of news media event detection and discuss its effectiveness and applicability.

3. We empirically evaluate how the performance of news media event detection at the sentence and token levels can be improved for different languages involving the cross-linguality in transformer models and language-based learning strategies along with TTL, answering the following research questions:

   RQ1: Can an event detection model based on a multilingual transformer, which is only fine-tuned for a particular language, outperform a model based on a monolingual transformer of that language?

   RQ2: Can a high-resource language improve the event detection performance of a low-resource language using the cross-linguality in transformer models?

   RQ3: Can TTL using sentence and token level event detection tasks improve the performance of involved tasks in monolingual and multilingual settings?

4. We release the implementation used for our experiments as an open-source project[1] to support related research and applications.

The rest of this chapter is organised as follows. Section 8.1 introduces the proposed approach for event sentence and word extraction along with TTL.

[1]Our codebase is publicly available on https://github.com/HHansi/MultiEventMiner

Section 8.2 describes the experimental setup we used for our experiments. Section 8.3 comprehensively describes the conducted experiments and obtained results along with discussions which address the targeted research questions. Finally, Section 8.4 summarises the conclusions with aimed future work.

8.1 Methodology

Transformer-based approaches have recently been proposed for sentence and word/token level event detection tasks, setting the state-of-the-art performance (Section 2.3). Following these trends, we use the transformer-based architectures described in Section 8.1.1 for sentence and token level predictions in this research. However, as far as we know, none of the available methods accounts for the interconnections between sentence and token levels, targeting possible mutual improvements, especially in settings where labelled data are scarce, such as low-resource language predictions. Focusing on this gap, we propose a novel *Two-phase Transfer Learning (TTL)* strategy combining the characteristics of traditional transfer learning, multi-task learning and transformers in Section 8.1.2. Using this approach, we aim to transfer knowledge from data at different granularities (i.e. sentence and token levels). Also, to the best of our knowledge, this is the first attempt to transfer knowledge from different data granularities to identify events in news text.

8.1.1 Transformer-based Classifiers

Similar to our event article identification approach, we treat event sentence identification as a sequence classification problem. Thus, we use the transformer-based sequence classification architecture proposed in Section

7.1.1, considering a sentence as an input sequence, for event sentence classification. For token level predictions, we use the transformer-based sequence labelling architecture shown in Figure 8.3. Similar to the document/sentence level model, this also requires processing a single sequence per instance. Thus, we only need the special token [CLS] to format the input text. To perform the token level classification, we feed the final hidden state of each token to a separate softmax layer. As described in Section 7.1.1, a softmax layer contains k neurons equivalent to the number of classes targeted by the classifier, which follow the softmax activation function (Equation 7.1). Each of these neurons returns probabilities per class, and we pick the class with maximum probability as the final prediction.

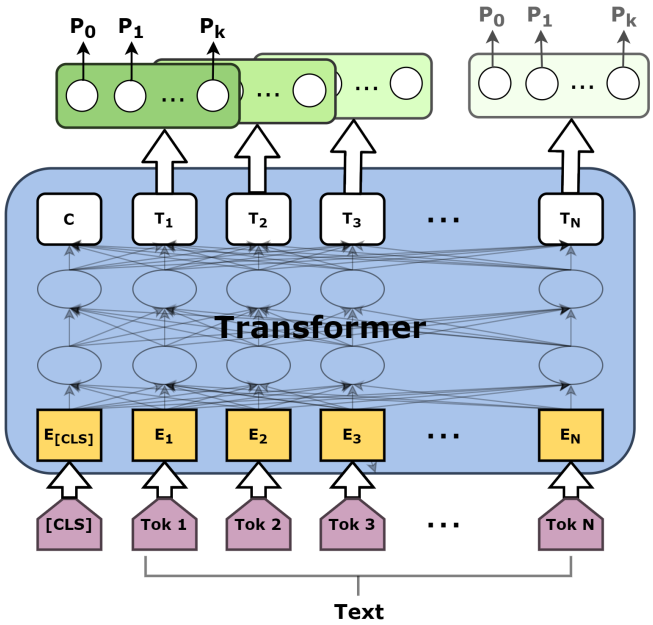


FIGURE 8.3: Transformer-based token classification architecture

8.1.2 Two-phase Transfer Learning (TTL)

Transfer learning (TL) is the process of improving a target predictive function of task T_t at a target domain D_t using the related knowledge gained from a

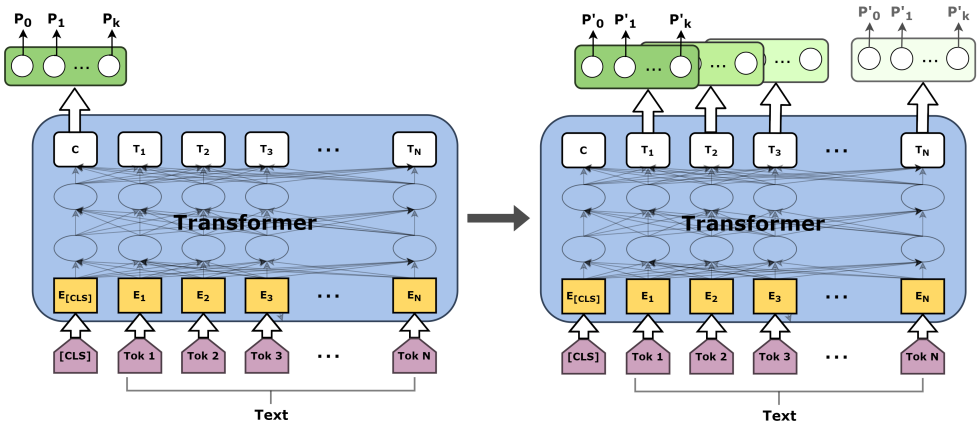
task T_s at a source domain D_s where $D_s \neq D_t$ or $T_s \neq T_t$ (Weiss et al., 2016). This knowledge transfer also helps mitigate overfitting and underfitting problems that arise with deep neural networks due to data limitations, allowing to use such networks for a wide range of tasks where training data is scarce (He et al., 2020; Ranasinghe et al., 2020). Mainly, there are two TL types based on the consistency between the source and the target feature and label spaces (Zhuang et al., 2021). If both source and target feature and label spaces are equivalent ($X_s = X_t$ and $Y_s = Y_t$), it is named homogeneous TL, and if either feature spaces or label spaces are not equivalent ($X_s \neq X_t$ and/or $Y_s \neq Y_t$), it is named heterogeneous TL. Comparatively, homogeneous TL is commonly used in previous research, but heterogeneous TL is more advantageous considering its ability to learn from different feature/label spaces (Day and Khoshgoftaar, 2017). However, most available solutions handle the heterogeneity by transforming feature/label spaces into common spaces with the possibility of losing important information in data or original data structure (Shi et al., 2010; Moon and Carbonell, 2016).

The concept of multi-task learning (MTL) is popularly used in recent research to handle heterogeneous tasks (Cruz et al., 2020; Mathew et al., 2021). MTL optimises a model for more than one task simultaneously leveraging the generalisation across all tasks (Zhang and Yang, 2021). It learns the interconnections between tasks rather than transferring knowledge from a related task as with TL. Also, this learning does not require space transformations similar to heterogeneous TL. However, this strategy requires shared training instances across all tasks, which are unavailable in many scenarios, including low-resource language-based predictions.

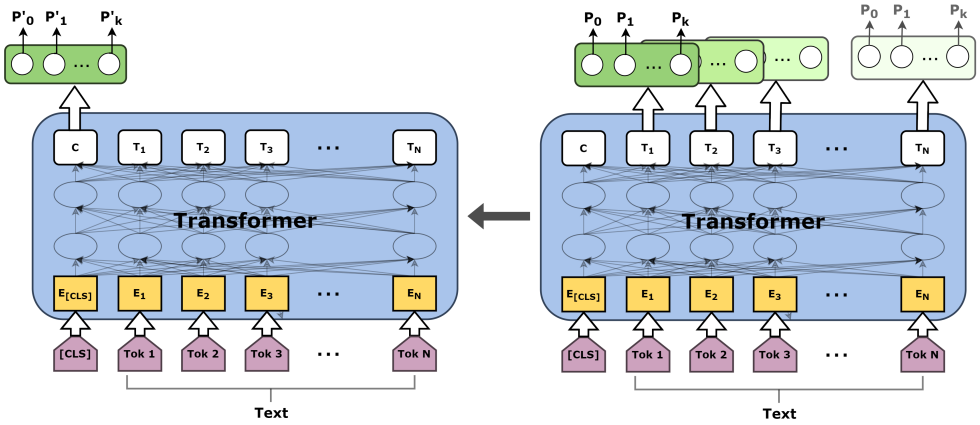
Considering the above limitations in heterogeneous TL and MTL, we propose a hybrid strategy named *TTL* in this research. We mainly utilise the characteristics of transformers for our approach. Transformer models are originally designed with the ability to fine-tune a pre-trained language model for a downstream task by adding an additional output layer (Section 6.4.1). This approach allows transferring the knowledge from the language model to the downstream task predictions. Following this idea, we propose fine-tuning a pre-trained transformer for two related tasks in two sequential phases, unlike the simultaneous learning that happens with MTL. We add different output layers to the model depending on the targeted task at each phase but share the transformer weights among the tasks allowing the phase-2 task to learn from the phase-1 task in addition to the original language model.

With TTL, we target transferring knowledge from different levels of data granularity (i.e. sentence and token level) in news event detection. We aim to analyse how the relationships and data sizes at different levels affect learning and model performance. Sentence and token levels have intermediate relationships, specifically from the fine-grained (token) level to the coarse-grained (sentence) level, which helps derive the final labels. For example, if a sentence has an event trigger, it is an event sentence. Considering the data sizes, there is a tendency to have more labelled data at the sentence level than the token level due to the data annotation complexities at token data (Hürriyetoğlu et al., 2021b). We use the transformer architectures introduced in Section 8.1.1 for sentence and token level classifications with TTL as shown in Figure 8.4.

For the sentence to token level transfer learning, the transformer model is initially fine-tuned for the sentence level task by feeding the output of [CLS] to a softmax layer, which predicts probabilities per class in the sentence level, $P_0, P_1, ...P_k$, as illustrated in Figure 8.4a. Then, the fine-tuned transformer weights are again fine-tuned for the token level task by feeding the output



(A) TTL: sentence level to token level



(B) TTL: token level to sentence level

FIGURE 8.4: Two-phase classification architectures

of each token to separate softmax layers, which predicts the token level class probabilities, $P'_0, P'_1, ...P'_k$, utilising the transformer's pre-trained and phase-1 fine-tuned/sentence level knowledge. The same architectures are trained conversely for the token to sentence level transfer learning as shown in Figure 8.4b. Initially, the transformer model is fine-tuned for the token level task by adding multiple softmax layers per token and then fine-tuned again for the sentence level task using a single softmax layer over the [CLS] output, transferring the transformer's pre-trained and phase-1 fine-tuned/token level knowledge for sentence level predictions.

8.2 Experimental Setup

This section summarises the experimental setup of our architectures for event sentence and word (trigger and argument) extraction. We used the sentence and token level data of the GLOCON gold dataset, introduced in Section 6.2 for our experiments. Over these levels, this dataset has data from three languages: English (En), Portuguese (Pt) and Spanish (Es). Among these languages, we consider English as a high-resource language and others as low-resource languages based on data availability. We used Macro F1 and CoNLL 2003 F1 measures described in Section 6.3 to evaluate the models' accuracy. Considering the targeted languages, we involved four popular transformer models, including a multilingual model for our experiments as detailed in Section 8.2.1. The hyper-parameter configurations we used are available in Section 8.2.2. For model fine-tuning, we involved all language-based learning strategies introduced in Section 7.1.3 to analyse their performance in cross-lingual news event detection at the sentence and token levels and their impact on TTL. We followed a common convention to format training data combinations along with the learning strategies while reporting results, which is explained in Section 8.2.3. We implemented all the neural network architectures in Python, using PyTorch (Paszke et al., 2019), FARM[2] and Huggingface (Wolf et al., 2020) libraries. Our implementation is publicly available on GitHub[3]. We used a GeForce RTX 3090 GPU to conduct all experiments.

8.2.1 Pre-trained Transformers

We used three monolingual pre-trained transformer models and one multilingual model based on the targeted languages for our experiments. As

[2]FARM is available on `https://github.com/deepset-ai/FARM`
[3]Our codebase is publicly available on `https://github.com/HHansi/MultiEventMiner`

monolingual models, BERT (*bert-large-cased*) (Devlin et al., 2019), and its variants, BERTimbau (*BERTimbau-large*) (Souza et al., 2020) and BETO (*BETO-cased*) (Canete et al., 2020) models trained in English, Portuguese and Spanish were used. As the multilingual model, XLM-R (*xlm-roberta-large*) (Conneau et al., 2020) model trained in 100 languages, including the targeted languages, was used. Previous research had commonly used Multilingual BERT (mBERT) and XLM-R models for multilingual experiments, but according to the results in Chapter 7 and recent studies (Ranasinghe et al., 2020; Hettiarachchi et al., 2021a), mostly, the XLM-R model outperformed the mBERT model, proving its superior cross-linguality. Therefore, we only use the XLM-R model for our experiments in this chapter. We used HuggingFace's model repository[4] (Wolf et al., 2020) to obtain all these pre-trained transformers. More details about these models' design and language coverage in pre-training data are available in Tables 7.1 and 7.2 in Chapter 7.

8.2.2 Hyper-parameters

We used a common set of hyper-parameters for our experiments to generate comparable results, maintain consistency among architectures and provide a good starting configuration for researchers willing to explore our approaches further. For all models, we fixed the maximum sequence length to 128, considering the sequence length distribution of targeted data. We used the batch size of eight, the learning rate of $1e^{-5}$ with Adam optimiser and epochs of three with an early stopping patience of 10, considering the computational complexities associated with transformers. We set evaluation steps allowing 6-13 evaluations per training epoch depending on the size of the training dataset. A split of 10% from training data is used for these evaluations, and the rest is

[4]HuggingFace models are available on https://huggingface.co/models

used for training. To mitigate the impact on results by the randomness associated with deep neural networks, we used the majority-class self-ensemble approach, which also helps improve the overall performance, following recent trends (Hettiarachchi et al., 2021a; Awasthy et al., 2021). With this setting, per experiment, we trained five models initialised with different random seeds and took the majority vote of model predictions as the final prediction.

8.2.3 Training Formats

We involved all language-based learning strategies described in Section 7.1.3 along with TTL for our experiments. To maintain consistency in result formats, we use the convention introduced in Section 7.2.3 to format training data depending on the learning strategy while reporting results in this chapter. Additionally, we use the format of $L(1) - L(2)$, which indicates learning the first phase (task 1) from data in language/language combination $L(1)$ and the second phase (task 2) from data in language/language combination $L(2)$, for TTL.

8.3 Results and Discussion

This section presents the evaluation results of event sentence and word (trigger and argument) extraction targeting the accuracy of predictions made using proposed architectures and learning strategies. While applying language-based learning strategies (Section 7.1.3), we only allowed transfer and zero-shot learning from high-resource to low-resource languages because the contrary approach is not reasonable. We report the results of transformer-based sequence and token classification architectures only learning the corresponding level of data under one-phase learning (Section 8.3.1). In this section, we

address the research questions: RQ1 and RQ2, analysing both sentence and token level results. Following this, Section 8.3.2 reports the results of two-phase architectures, which learn both sentence and token level data sequentially, and answers the RQ3 based on our overall findings.

8.3.1 One-phase Learning

We report and discuss the results of transformer-based sequence and token classification architectures by learning one-phase (sentence or token level data) in Sections 8.3.1.1 and 8.3.1.2, respectively.

8.3.1.1 Event Sentence Identification

For one-phase learning of event sentence identification, we conducted experiments using transformer-based sequence classification architecture (Figure 7.1), involving the language-based learning strategies introduced in Section 7.1.3. To build classifiers, we used monolingual transformers: BERT, BERTimbau and BETO and the multilingual transformer: XLM-R. We refer to the models based on monolingual transformers as monolingual models and the models based on multilingual transformers as multilingual models in the below content for simplicity. Table 8.1 summarises the results we obtained. The combinations that are impossible (e.g. making predictions in En using a monolingual model, BERTimbau, which only supports Pt) are marked with '-' in the table.

According to results in Table 8.1, monolingual models trained in a particular language outperformed the multilingual models trained in that language for high-resource (En) and low-resource (Pt and Es) languages. However, with zero-shot learning, the multilingual model trained on the high-resource language made more accurate predictions for low-resource languages than monolingual models. A similar trend is also noticed with the multilingual models,

TABLE 8.1: Sentence level results: Macro F1 using transformer-based sequence classification architecture. *Strategy* and *Language* indicate the language-based learning strategy and language of test data. NT shows the models which were not trainable due to data limitations. Zero-shot learning scenarios are marked with ‡, and the best results per language are in bold.

Strategy	**Transformer**	**Training Data**	**Language**		
			En	Pt	Es
monolingual learning	BERTimbau	Pt	-	0.7068	-
	BETO	Es	-	-	0.7958
	BERT	En	0.8253	-	-
	XLM-R	Pt	-	NT	-
	XLM-R	Es	-	-	0.4814
	XLM-R	En	0.7900	0.8518‡	0.8121‡
transfer learning	XLM-R	En→Pt	0.7991	0.8429	0.7547‡
		En→Es	0.8174	**0.8871**‡	0.8199
multilingual learning	XLM-R	En+Pt	**0.8307**	0.8585	0.7547‡
		En+Es	0.8265	0.8596‡	0.8305
		En+Pt+Es	0.8127	0.8665	**0.8448**

which transfer learned a low-resource language after the high-resource language. The multilingual model performance could be further improved with multilingual learning than with monolingual and multilingual models trained using other learning strategies. Based on our results, we answer RQ1 and RQ2, focusing on event sentence identification below.

RQ1: *Can an event detection model based on a multilingual transformer, which is only fine-tuned for a particular language, outperform a model based on a monolingual transformer of that language?*

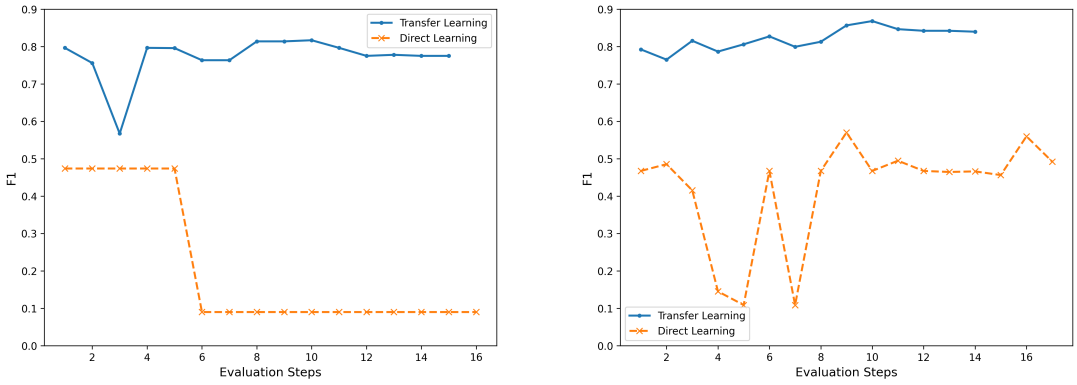
During our experiments, we analysed models' performance (or accuracy) based on monolingual and multilingual transformers for identifying event sentences in three languages (En, Pt and Es). Comparatively, the monolingual transformers we used (i.e. BERT, BERTimbau and BETO) were pre-trained on

fewer data than that particular language data used by the multilingual transformer (i.e. XLM-R) (Table 7.2). However, the larger the vocabulary size, the transformer requires to learn more parameters during the fine-tuning (Table 7.1). Thus, as can be seen in our results in Table 8.1 under monolingual learning, when a few training instances are available for fine-tuning, monolingual models can learn better to identify event sentences than the multilingual models, even though the monolingual transformers have seen fewer data during language modelling. For the high-resource language (En) with 22,481 training instances, the monolingual model improved the Macro F1 by 3.5% more than the multilingual model. Monolingual models showed more improvements than the multilingual model when the training data size became smaller. For Pt, the XLM-R-based model did not converge (behaved as a majority class classifier), but BERTimbau returned 70.68% Macro F1, and for Es, BETO returned 31.44% higher Macro F1 than XLM-R.

In summary, multilingual transformer typically requires more training data to fine-tune for the event sentence identification task than the monolingual models, considering the parameter counts. Thus, if data are insufficient for fine-tuning, multilingual models cannot perform better than monolingual models. This claim is further supported by the variations in F1 gaps between multilingual and monolingual models for different languages with different training data sizes mentioned above. The higher the data size, a low gap is returned, indicating that the multilingual model can perform on par or better than the monolingual models if enough training data exists, agreeing with the conclusions made by the XLM-R model's original study (Conneau et al., 2020).

RQ2: *Can a high-resource language improve the event detection performance of a low-resource language using the cross-linguality in transformer models?*

Targeting this question, we involved different learning strategies to analyse the cross-lingual capabilities of the multilingual transformer model we chose (XLM-R). With zero-shot learning, the multilingual sentence classification model, which only learned the high-resource language (En), outperformed the monolingual models that learned corresponding low-resource languages (Table 8.1). Agreeing with our findings for RQ1, the XLM-R model fine-tunes well when sufficient training instances are provided. Utilising its cross-linguality, effective predictions can make for low-resource languages, learning high-resource languages.

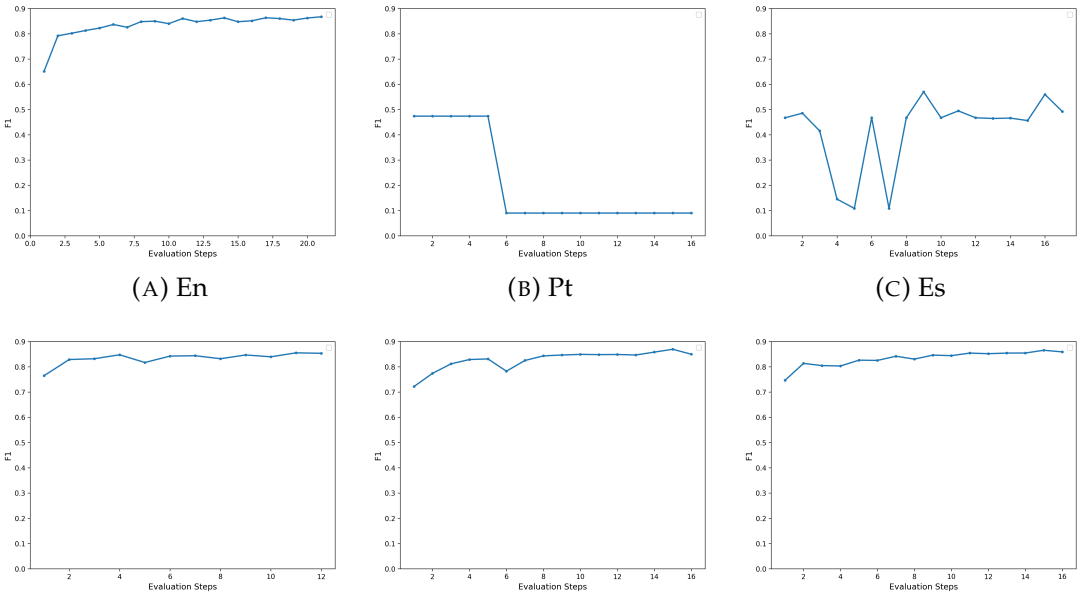


(A) Pt vs En→Pt (B) Es vs En→Es

FIGURE 8.5: Macro F1 scores for the validation sets at different evaluation steps of the sentence level training processes, which involved direct language learning (Pt, Es) and transfer learning from a high-resource language (En→Pt, En→Es) with the sequence classification model with XLM-R transformer.

Also, we obtained improved sentence classification results for low-resource languages from multilingual models, which transfer learned the low-resource language (Pt or Es) after the high-resource language (En) (Table 8.1). As shown in Figure 8.5, with transfer learning (TL), multilingual models return high Macro F1 values from the beginning of evaluation steps, unlike with direct learning. This indicates that even with few training instances, a model can

learn well following the knowledge obtained during high-resource language training and the cross-lingual abilities of the transformer. Even for the scenario with Pt where the model did not converge with direct learning due to data limitations, TL returned Macro F1 scores of around 80% throughout the evaluations emphasising its effectiveness (Figure 8.5a). However, no notable improvements are recognised, mostly comparing the multilingual models which transfer learned from the high-resource language and models which only learned the high-resource language. This indicates that if the low-resource language datasets are very small compared to the high-resource language data, they cannot significantly impact the model performance via TL.



(A) En (B) Pt (C) Es

(D) En+Pt (E) En+Es (F) En+Pt+Es

FIGURE 8.6: Macro F1 scores for the validation sets at different evaluation steps of the sentence level training processes, which involved monolingual (En, Pt, Es) and multilingual (En+Pt, En+Es, En+Pt+Es) learning with the sequence classification model with XLM-R transformer. During multilingual learnings, a composition of samples from each language is used as the validation set.

Furthermore, we experimented with multilingual learning. Mostly, models fine-tuned using multilingual learning outperformed the models which only learned the high-resource language or transfer learned a low-resource

language for sentence level predictions (Table 8.1). Additionally, Figure 8.6 illustrates how Macro F1 values vary over evaluation steps with monolingual and multilingual learning. With monolingual learnings, the high-resource language (En) has a high F1 value from the second evaluation step, but other low-resource languages have very low F1 values over all steps. However, with multilingual learning, for all combinations, models return high F1 values (approximately $\geq$80%) throughout all evaluations (Figure 8.6d, 8.6e, 8.6f). These results reveal that a cross-lingual model can train well on each language (or adjust its parameters appropriate for multiple languages) when it sees all language data together rather than seeing the languages separately. Also, this way allows the effective utilisation of low-resource language data irrespective of the data size, unlike the scenario with TL.

In summary, these findings lead to a positive answer to RQ2. High-resource languages can improve the event sentence identification performance of low-resource languages using the cross-linguality in transformer models. Zero-shot learning can be effectively applied using a multilingual model that is fine-tuned only on high-resource language data for a scenario with no training data available for a low-resource language. When few training instances are available for low-resource languages, a multilingual model can be fine-tuned effectively by combining all the data using multilingual learning, outperforming the language-based TL approach.

8.3.1.2 Event Trigger and Argument Extraction

For event trigger and argument extraction, we utilised transformer-based token classification architecture (Figure 8.3) along with the language-based learning strategies introduced in Section 7.1.3. However, we had to skip a few strategies due to training data limitations. For low-resource languages (Pt and

Es), token level data are minimal (<100 instances), and thus, monolingual and language-based TL experiments could not be conducted. Therefore, we only required the English transformer model: BERT and the multilingual model: XLM-R for token level experiments. However, similar to the above section, we refer to the BERT-based models as monolingual models and XLM-R-based models as multilingual models to maintain consistency and generality of the content. Table 8.2 summarises the results we obtained.

TABLE 8.2: Token level results: CoNLL 2003 F1 using transformer-based token classification architecture. *Strategy* and *Language* indicate the language-based learning strategies and language of test data. Zero-shot learning scenarios are marked with ‡, and the best results per language are in bold.

Strategy	**Transformer**	**Training Data**	**Language**		
			En	Pt	Es
monolingual learning	BERT	En	0.7517	-	-
	XLM-R	En	0.7511	0.7043‡	0.6461‡
multilingual learning	XLM-R	En+Pt	**0.7678**	0.7389	0.6587‡
		En+Es	0.7540	0.7151‡	0.6700
		En+Pt+Es	0.7616	**0.7441**	**0.6752**

Comparatively, token level predictions are less accurate than sentence level predictions, emphasising the complexity of the token level task. According to Table 8.2 results, for the high-resource language (En), the monolingual model performed slightly better than the multilingual model. For low-resource languages also, good F1 scores (≥65%) could be obtained with zero-shot learning on the multilingual model trained on the high-resource language. The involvement of multilingual learning further improved the results of high- and low-resource languages, effectively utilising the few labelled instances available with low-resource languages.

Following our results, we answer RQ2, focusing on event trigger and argument extraction below. Due to training data limitations, we could not train

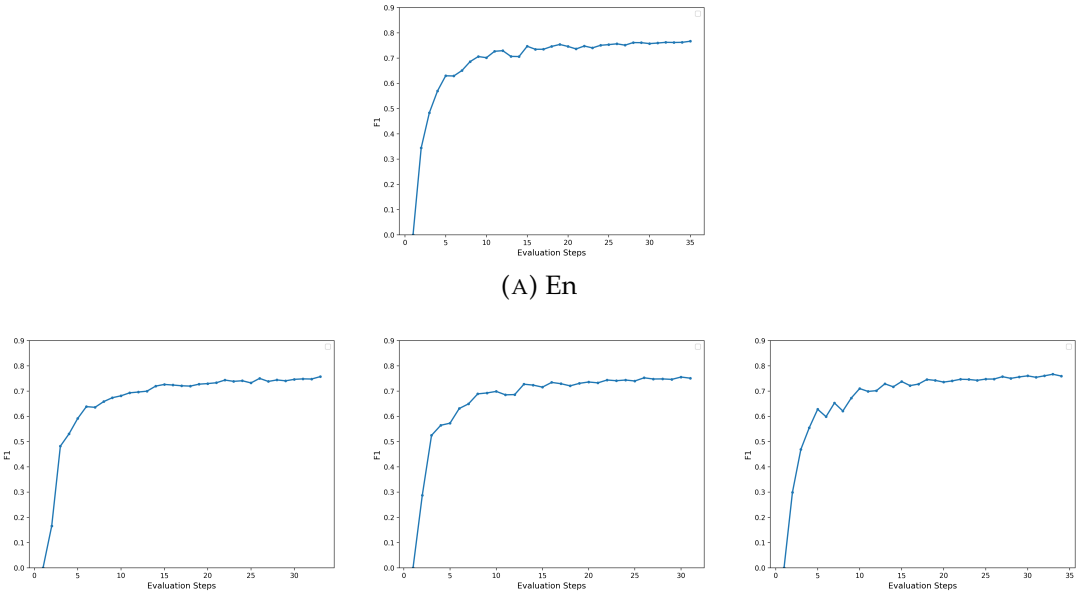
monolingual models for low-resource languages to compare with multilingual models and thus skip addressing RQ1 for this task. However, for En, the monolingual model slightly improved over the multilingual model, which is only fine-tuned using that language, agreeing with our finding for RQ1 based on sentence level results.

RQ2: *Can a high-resource language improve the event detection performance of a low-resource language using the cross-linguality in transformer models?*

Similar to sentence level analysis, we used different learning strategies with the selected multilingual transformer model (XLM-R) to address this question, focusing on event trigger and argument extraction. However, language-based TL could not be applied since there are insufficient training instances from low-resource languages to learn separately. With zero-shot learning, the multilingual token classification model, which only learned the high-resource language (En), returned good results (F1 scores $\geq$65%) for low-resource languages, as can be seen in our results in Table 8.2. These results clearly highlight the cross-linguality of the XLM-R model, which can effectively utilise for low-resource language token level predictions with no training data.

The token level results were further improved with multilingual learning (Table 8.2), similar to our findings with event sentence identification. Multilingual learning allowed the model to learn using the high-resource language (En) data and the few training instances of low-resource languages (Pt and Es), which are insufficient to build separate models or apply language-based TL. We also analysed how the CoNLL F1 scores vary over the evaluation steps of each learning setting (Figure 8.7). However, we do not have monolingual models from each language to compare with. Also, we cannot see clear distinctions in the F1 scores between the En and multilingual models, similar to the sentence level analysis. When low-resource language data are limited, the

validation split at each setting is almost identical to the En validation split. Thus, we see nearly constant behaviour of F1 scores across all settings. Even though the improvements are not clearly visible over the evaluation steps of the training phase, the final predictions on test data emphasise the effectiveness of multilingual learning.



(A) En

(B) En+Pt (C) En+Es (D) En+Pt+Es

FIGURE 8.7: CoNLL 2003 F1 scores for the validation sets at different evaluation steps of the token level training processes, which involved monolingual (En) and multilingual (En+Pt, En+Es, En+Pt+Es) learning with the token classification model with XLM-R transformer. During multilingual learnings, a composition of samples from each language is used as the validation set.

In summary, we can also provide a positive answer to RQ2 based on the token level results. High-resource languages can improve the event trigger and argument extraction performance of low-resource languages, using the cross-lingual capabilities of transformers. Zero-shot learning can be effectively used in scenarios with no training data. It is effective to use multilingual learning when few training instances are available from low-resource languages irrespective of their count.

8.3.2 Two-phase Learning

In this section, we report and discuss the results of TTL along with the pre-trained transformer models and language-based learning strategies we involved with one-phase learning. For event sentence identification, we trained the model for the token level task before the sentence level task using the proposed architecture in Figure 8.4b. The opposite learning sequence is followed for the event trigger and argument extraction (Figure 8.4a). Tables 8.3 and 8.4 summarise the results we obtained.

TABLE 8.3: Sentence level results: Macro F1 using two-phase classification architecture, which learns the sentence level task following the token level task. *Strategy* and *Language* indicate the language-based learning strategy and language of test data. Zero-shot learning scenarios are marked with ‡, and the best results per language are in bold. Highlighted cells indicate improved F1 scores than only learning sentence data.

Strategy	**Transformer**	**Training Data**	**Language**		
			En	Pt	Es
monolingual learning	BERT	En – En	0.8049	-	-
	XLM-R	En – Pt	-	0.4997	-
	XLM-R	En – Es	-	-	0.7638
	XLM-R	En – En	0.7879	0.8543‡	0.8034‡
transfer learning	XLM-R	En – En→Pt	0.6028	0.5568	0.5763‡
		En – En→Es	0.6985	0.8404‡	0.7964
multilingual learning	XLM-R	En – En+Pt	0.8219	0.8646	0.8088‡
		En+Pt – En+Pt	0.8085	0.8658	0.8094‡
		En – En+Es	0.7884	0.8749‡	0.8255
		En+Es – En+Es	**0.8352**	0.8631‡	0.8328
		En – En+Pt+Es	0.7943	0.8708	0.8186
		En+Pt+Es – En+Pt+Es	0.8149	**0.8772**	**0.8404**

As can be seen in Table 8.3, TTL (learning token level task before sentence level task) improved the accuracy of low-resource language predictions at the sentence level in the majority of cases. Multilingual models trained on the high-resource language token data before training on low-resource language

sentence data outperformed the multilingual models, which only learned low-resource language sentence data. Also, the multilingual models, which learned the high-resource language token and sentence data, returned higher F1 values for low-resource languages than the scores of monolingual models, which only learned the sentence level of that particular language. However, combining language-based TL with TTL did not improve the results for any language. Contrarily, with multilingual learning, TTL performed better in most cases than only learning sentence data.

TABLE 8.4: Token level results: CoNLL 2003 F1 using two-phase classification architecture, which learns the token level task following the sentence level task. *Strategy* and *Language* indicate the language-based learning strategy and language of test data. Zero-shot learning scenarios are marked with ‡, and the best results per language are in bold. Highlighted cells indicate improved F1 scores than only learning token data.

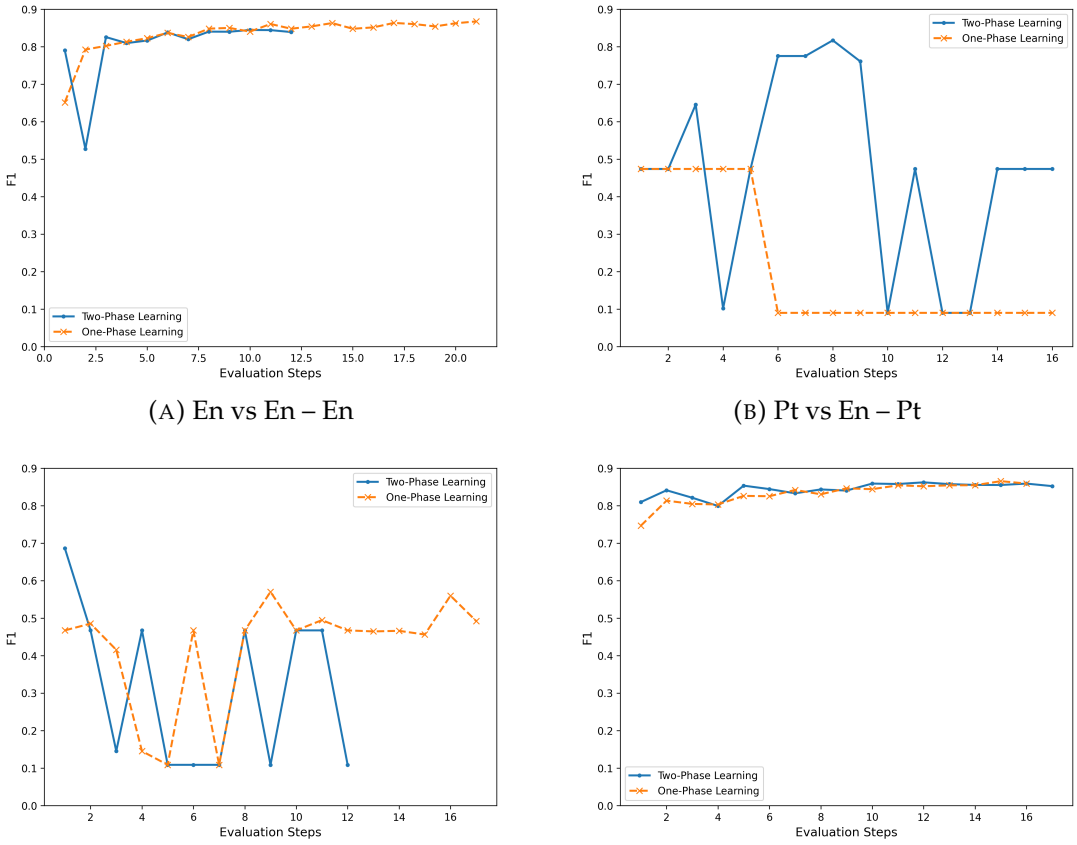
Strategy	**Transformer**	**Training Data**	**Language**		
			En	Pt	Es
monolingual learning	BERT	En – En	0.7513	-	-
	XLM-R	En – En	0.7566	0.6930‡	0.6266‡
multilingual learning	XLM-R	En – En+Pt	0.7548	0.7222	0.6478‡
		En+Pt – En+Pt	0.7542	0.7275	0.6377‡
		En – En+Es	0.7568	0.7164‡	0.6652
		En+Es – En+Es	0.7525	0.7012‡	0.6567
		En – En+Pt+Es	0.7575	0.7364	0.6620
		En+Pt+Es – En+Pt+Es	**0.7599**	**0.7441**	**0.6780**

Following the results in Table 8.4, overall, TTL (learning sentence level task before token level task) did not improve the token level predictions even though more instances are available with sentence data. However, monolingual learning could improve the multilingual model performance for the high-resource language with TTL rather than only learning token data. Also, on a few occasions, applying TTL with multilingual learning improved the results compared to the models that only learned token data. Based on the results, we answer RQ3, focusing on the tasks event sentence and word extraction below.

RQ3: *Can TTL using sentence and token level event detection tasks improve the performance of involved tasks in monolingual and multilingual settings?*

We analysed the performance of TTL involving the tasks: event sentence identification and event trigger and argument extraction at two data granularities: sentence and token level. Our experiments showed improvements in the sentence level predictions in most cases using the models which learned token level beforehand (Table 8.3). Further analysis on variations in Macro F1 values over model evaluation steps also confirmed that TTL from token level helps sentence level learning. As can be seen in Figure 8.8, for monolingual and multilingual learning, the sentence level learning process begins with high F1 scores or achieves high F1 scores in a few steps with TTL than the scores obtained by learning the sentence level task directly. However, in most cases, token level predictions were not improved by learning sentence data beforehand, even though more training instances are available at the sentence level (Table 8.4). As shown in Figure 8.9, during the model training process also, TTL behaves similar to learning token data directly. Since token level labels directly help resolve sentence level labels, learning the token data help the model to improve sentence level predictions. Contrarily, token labels cannot be predicted by seeing sentence labels. Thus, learning sentence labels beforehand does not help the model much with token level predictions, even though the instance count is high.

In summary, TTL can improve the performance of a task in monolingual and multilingual settings by learning a related task that can help derive the targeted labels during the first phase. In other terms, this strategy can mainly be used to improve the performance of a coarse-grained task based on a related fine-grained task. This also reveals that the relatedness between tasks is more crucial for this learning than the training dataset sizes. This strategy would be more helpful in scenarios requiring predictions for low-resource languages



(A) En vs En – En

(B) Pt vs En – Pt

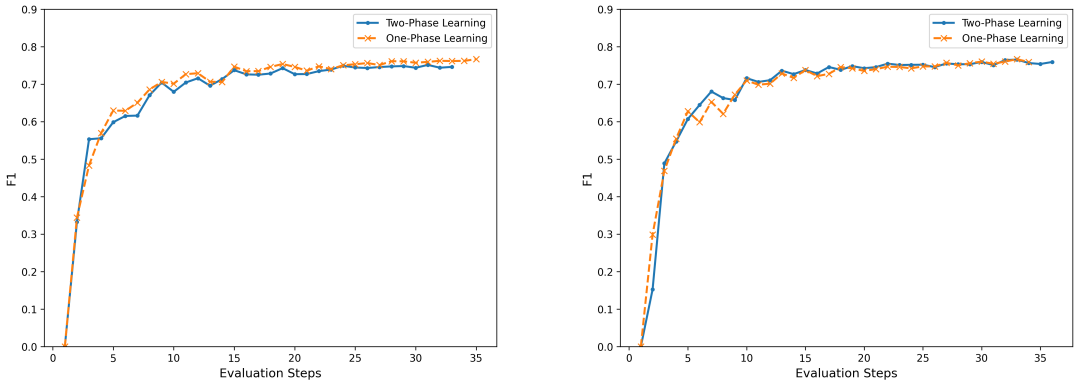
(C) Es vs En – Es

(D) En+Pt+Es vs En+Pt+Es – En+Pt+Es

FIGURE 8.8: Macro F1 scores for the validation sets at different evaluation steps of the sentence level training processes using the sequence classification model (one-phase learning) and two-phase classification model (two-phase learning) with XLM-R transformer. For multilingual learning, a composition of samples from each language is used as the validation set.

with few or no training instances, as data from other languages prepared for related tasks can be used.

However, learning two phases requires more training time or computational resources than learning one phase. Yet, the training process has no impact on the final model's size or inference time, which are critical for its later usage, as these factors only depend on the model architecture. Our analysis reported in Appendix B.1 further confirmed this fact. Overall, all built models took less time than a second on a GPU and a maximum of 7 seconds on a CPU



(A) En vs En – En (B) En+Pt+Es vs En+Pt+Es – En+Pt+Es

FIGURE 8.9: CoNLL 2003 F1 scores for the validation set at different evaluation steps of the token level training process using the token classification model (one-phase learning) and two-phase classification model (two-phase learning) with XLM-R transformer. For multilingual learning, a composition of samples from each language is used as the validation set.

to make a prediction. Therefore, if the training process helps improve the final predictions, the additional time it takes can be neglected, considering the possible later usages of the model for many effective predictions.

8.4 Conclusions

In this chapter, we proposed a novel learning strategy named *Two-phase Transfer Learning (TTL)*, allowing transformer models to learn from different levels of data granularity. Transformers are especially involved in our approach, considering their transferability, cross-linguality, context awareness and state-of-the-art performance in many NLP applications. We applied TTL to news event detection and analysed how it can improve sentence and word/token level tasks by transferring knowledge. Also, to the best of our knowledge, this is the first effort to report a comprehensive experimental study on cross-lingual event detection, covering sentence and token level tasks and their transferability.

We used sentence and token level data of the multilingual version of GLOCON gold standard dataset and several monolingual and multilingual pre-trained transformer models for our experiments. Our findings show that if sufficient training data exist, a multilingual transformer-based model can outperform a monolingual model, answering RQ1 of this research. Also, our experiments indicate that high-resource languages can improve the event sentence and word extraction performance of low-resource languages, using cross-linguality in transformer models, especially with zero-shot and multilingual learning, addressing RQ2. These findings will also be beneficial for real-world applications because maintaining a multilingual transformer-based model which can handle multiple languages is more resource-efficient than having several monolingual models per language. Following RQ3, with the involvement of TTL, we could further improve the model performances in monolingual and multilingual settings. However, the relatedness of tasks is more crucial with this learning than the training data sizes. Thus, we noticed more improvements by learning the sentence level task after the token level task since the token data can help derive the labels of sentences. Additionally, the ability to learn from different language data at different granularities helps build effective models for low-resource languages, utilising available data.

Overall, the relatively high F1 measures returned by sentence (Macro F1 >84%) and token (CoNLL 2003 F1 >67%) level experiments in three languages, even with low training resources, prove the accuracy of proposed approaches and expandability to multiple languages. The independence from any domain- or language-specific features despite the complexity of the task further emphasises the approaches' expandability. Considering the efficiency and scalability, findings reported with the coarse-grained level (event article identification) approach (Section 7.3.1) indicate that the sentence and token level approaches are also sufficiently fast and scalable, as these factors for a

transformer-based classifier do not depend on the training process or classification heads (Appendix B.1).

In future work, we plan to extend our research to more languages and analyse how their interconnections can be utilised to improve the performance of sentence and token level event detection tasks. Also, in this work, we only focused on the languages supported by available pre-trained transformer models such as XLM-R. To fill this gap, we aim to construct datasets for not supported languages and evaluate their performance in future. Considering TTL, we designed it in a generic manner applicable to any related sentence and token level classification tasks without limiting it to event detection. Thus, we also plan to investigate its applicability in different domains and research areas thoroughly.

With this chapter, we conclude Part II of the thesis. In this part, we proposed using long-sequence transformer models and a novel transformer-based learning strategy (TTL) for coarse- and fine-grained level event detection from news media data, respectively, which returned promising results. We particularly focused on capturing underlying linguistics while designing these approaches, similar to our focus during Part I of this research (social media event detection). With the involvement of transformers, we could allow the proposed architectures to learn contextual relationships in the text to capture linguistics while resolving text ambiguities and utilise the knowledge of pre-trained models for event detection tasks, improving the overall performance. Also, we believe TTL, which is proposed to transfer knowledge from different data granularities (i.e. sentence and token levels), will pave a new path for future studies and be useful for many research areas, considering its novelty and broad applicability.

Chapter 9

Final Remarks and Perspectives

This research focused on developing event detection approaches for textual data from social media data streams and news media articles. Social and news media event detection has recently gained wide attention from researchers due to the informativeness of data and their remarkable growth in generation. Without an automated intelligent mechanism, it is impractical to process data manually to extract information, even at the current data generation rates. Since there will be access to more data in future, according to the predictions, a large proportion of the data will be useless unless proper mechanisms are developed. Thus, the efforts to improve the current state or qualities of event detection approaches would be invaluable.

In this research, we defined the problem of event detection based on two data granularities, coarse- and fine-grained levels. The main target of coarse-grained level detection is to notify users about event occurrences, giving them space to take necessary future actions of their choice. Contrarily, fine-grained level detection mainly focuses on identifying comprehensive event details to automate the complete process. Since there can be different types of data with varying levels of sensitivity unique to the users depending on their information requirements, it is beneficial to focus on different information levels while

developing event detection approaches. Furthermore, we believe this categorisation will encourage researchers to look at event detection from a more detailed perspective.

We analysed the progress of previous work in event detection and its requirements to specify the focus of this research. Overall, social and news media event detection evolved differently, mainly due to the variations in these media and the information they generate. Efficient unsupervised techniques were found to be more beneficial for social media event detection. Contrarily, supervised approaches were popularly used for news media event detection. However, underlying linguistics needs to be understood appropriately to extract information from text successfully. According to the recent NLP trends, prediction-based text embedding models (e.g. Word2Vec, BERT, etc.) were found to be the most effective approaches to capturing linguistics. Based on these facts, we mainly focused on how the characteristics of prediction-based text embeddings can be utilised to develop effective event detection mechanisms while defeating the critical limitations in available approaches in this research and comprehensively described our work in this thesis.

Overall, we provide a summary of our work in social and news media event detection in this chapter, closing the thesis. Section 9.1 provides an overview of the whole thesis, revisiting the aim and objectives of this research. Section 9.2 details the achievements made during this study. Finally, Section 9.3 describes some potential directions for future work.

9.1 Overview

This research aimed to investigate how different capabilities of prediction-based text embedding learning approaches can be utilised for effective event

detection from textual data in social media data streams and news media articles while overcoming the limitations of traditional approaches. To achieve the aim, five main objectives, introduced in Chapter 1, are followed. More details on each objective's fulfilment are stated below.

1. ***Conduct a thorough literature review covering the areas of social media event detection and news media event detection to understand the strengths and weaknesses of available approaches:*** We conducted a thorough literature review covering the areas of social and news media event detection, fulfilling this objective, and described the findings in Chapter 2 of this thesis. We mainly focused on recent work in these areas (primarily within the last decade) to understand the latest evolutions and state-of-the-art methods. Based on the gained knowledge, we noticed a considerable lack in involving linguistical features for social media event detection, which could lead to a critical information loss due to the characteristics of textual data. Similarly, in news media event detection, capturing long-range dependencies and interconnections between different data levels and supporting low-resource languages are recognised as the major challenges that exist with state-of-the-art methods, which need to be resolved to increase the accuracy and usability of the methods. Inspired by these findings, we focused on how the capabilities of prediction-based text embedding learning approaches can be effectively utilised to fill these gaps in the rest of this research.

2. ***Analyse the characteristics of social media data streams and limitations in the available approaches and, focusing on the findings, develop effective methods for event detection by appropriately utilising the competencies of embedding learning approaches:*** We described our efforts on

social media event detection in Part I of this thesis. Following the literature review findings, we further analysed the characteristics of social media data streams and the impact of the available approaches' limitations for event detection, as detailed in Chapter 3. Addressing the targeted limitations, we developed two novel methods, *Embed2Detect* and *WhatsUp*, for coarse- and fine-grained (i.e. event window and co-occurring event identification) event detection in social media. The proposed methods are described in Chapters 4 and 5. We mainly involved self-learned word embeddings and their hierarchical relationships in dendrograms to capture underlying text linguistics in our approaches, fulfilling this objective. To the best of our knowledge, this is the first time of using linguistics to detect both temporal and textual event details from social media data streams. Thus, we believe our findings will lead future research in social media event detection in a new direction. Also, the proposed approaches will be useful for wide research and application areas, as they are solely based on unsupervised techniques, which are easily expandable to any domain, language or platform.

3. ***Evaluate the approaches proposed for social media event detection using recent real datasets and compare the performance with state-of-the-art methods:*** We empirically evaluated the proposed approaches for social media event detection, *Embed2Detect* and *WhatsUp*, fulfilling this objective, and reported the findings in Chapters 4 and 5 of this thesis. To support evaluations, we prepared and published a recent real dataset, *TED*, covering two diverse domains (i.e. sports and politics), considering the limited availability of data for social media event detection, and described its details in Chapter 3. We also designed a comprehensive set of metrics to evaluate temporal and textual event details in a unified way,

as detailed in Chapter 3. Furthermore, we used several recently proposed methods from different competitive areas to provide stronger baselines for comparisons during the experiments. In summary, the approaches proposed by this research returned promising results on conducted experiments, outperforming the recently proposed methods, emphasising the proposed approaches' effectiveness and applicability.

4. ***Analyse the characteristics of news media articles and limitations in the available approaches and, focusing on the findings, develop effective methods for event detection by appropriately utilising the competencies of embedding learning approaches:*** We described our work in news media event detection in Part II of this thesis. Similar to the scenario with social media, following the literature review findings, we further analysed the characteristics of news media data and the impact of the available approaches' limitations for event detection, as detailed in Chapter 6. Addressing the limitations, we proposed a *TRansformer-based Event Document classification architecture (TRED)* using long-sequence transformer models for coarse-grained event detection (i.e. event article identification) in Chapter 7. As far as we know, this is the first time involving long-sequence transformers for event detection. Targeting the fine-grained event detection (i.e. event sentence and word extraction), we proposed a novel learning strategy, *Two-phase Transfer Learning (TTL)*, utilising transformer capabilities, which transfers knowledge from different data granularities (i.e. sentence and token levels) in Chapter 8, fulfilling this objective. We believe this learning technique will pave a new path for future studies and be useful for many research areas, considering its novelty and broad applicability.

5. ***Evaluate the approaches proposed for news media event detection using recent real datasets and compare the performance with state-of-the-art methods:*** We empirically evaluated the proposed architectures for news media event detection, fulfilling this objective, and reported the findings in Chapters 7 and 8 of this thesis. During the experiments, we mainly focused on how the performance of proposed architectures can be improved for different languages and data granularities using the transformer models' capabilities and different learning strategies. We involved four languages (i.e. English, Portuguese, Spanish and Hindi) for the experiments using GLOCON gold standard dataset (Hürriyetoğlu et al., 2021b), covering high- and low-resource scenarios. Overall, the proposed architectures returned promising results in all languages, irrespective of data availability, revealing the possibilities to improve the predictions in a low-resource language using high-resource language data and at a particular granularity (e.g. sentence) using data from another granularity (e.g. token). We believe these findings will help expand event detection into more domains and languages while being useful for a wide range of research which involves transformer models.

9.2 Achievements

This research aimed achieving four crucial strengths in the systems designed for social and news media event detection as described in Chapter 1 (Section 1.1.4). Their successful achievement can be summarised as follows.

Accuracy: Achieving a high degree of accuracy in results/predictions is vital to confirm a system's potential to solve an actual problem successfully. We involved a wide variety of evaluation metrics, which capture textual event

details from different data granularities: document to token level and temporal details, to conduct comprehensive evaluations appropriate for each component developed by this research. According to the obtained results, Embed2Detect and WhatsUp, the novel approaches proposed for social media event detection, outperformed recent competitive methods, confirming their high accuracy in capturing coarse- and fine-grained event details in data streams. Similarly, TRED and TTL, along with different pre-trained transformer models and language-based learning strategies, improved the accuracy of predictions than state-of-the-art methods, proving the proposed approaches' ability to successfully extract document, sentence and token level event details from news media articles.

Efficiency: Efficiency is also a crucial strength required by event detection mechanisms to extract information in (near) real-time before they get outdated. For social media event detection, we followed unsupervised approaches, mainly considering the requirement to capture newly arising events along with the dynamicity in data streams. Thus, the designed approaches need to process large data volumes to extract events. Even though the larger the data size, the longer the processing time, our experiments revealed that the proposed approaches are capable of completing the process in a very short time than the time taken to generate data, proving their efficiency. For news media event detection, we used supervised approaches, mainly considering the pre-definitive nature of events reported in this media. Thus, the proposed approaches' efficiency solely relied on inference speeds, which were estimated to be a few seconds on a CPU and faster on a GPU than on a CPU, confirming their real-timeliness.

Expandability: Event detection approaches need to be easily expandable to any domain, language or platform to extract information from diverse data available via social and news media. All the methods proposed in this research for social and news media event detection are independent of domain-, language- or platform-specific features, fulfilling this requirement. To further confirm this fact, we experimented with the proposed social media event detection approaches on two diverse domains and obtained competitive results outperforming the recently proposed approaches. Similarly, we analysed the performance of news media event detection approaches in multiple languages and received relatively high accuracy values even with low training resources, proving the methods' expandability, including cross-lingual capabilities.

Scalability: Event detection mechanisms should be scalable to process large data volumes, to support real applications and future usages, considering the current data generation and its extensive growth estimated in future. Since this research proposes unsupervised approaches for social media event detection, which require processing data streams to extract events, high data generation directly impacts their performance. Thus, we comprehensively analysed their scalability theoretically and practically to ensure this critical requirement is satisfied. Both the computational complexity estimations and results with increasing data sizes indicated that the proposed approaches are sufficiently scalable to handle large data volumes efficiently. On the contrary, the proposed news media event detection approaches have no direct impact from high data generation, as they are supervised approaches that make predictions on incoming queries based on pre-trained models. However, the prediction/inference process should be sufficiently fast, as confirmed by the experiments, to handle large data volumes effectively. These experiments also indicated possibilities

to improve model inference speeds further, emphasising the approaches' scalability. We involved CPUs with average specifications for all our experiments, which are affordable to the majority.

Also, this work holds the following key strengths, which are helpful for other research in social and news media event detection.

Replicability: Being replicable guarantees the reliability of research outcomes while letting other researchers use them with their experiments or further explore in the same directions. All the datasets used for the experiments in this research, including the ones prepared as parts of this work, are publicly available to be used by any researcher. Similarly, the implementations/codebases used for all conducted experiments have also been made publicly available to confirm their replicability and support related research.

Competitiveness: Competitiveness is important for research outcomes to support solving problems and draw new research avenues. We always involved recent real datasets and state-of-the-art/recent competitive methods for the experimental studies conducted under this work to ensure their up-to-date competitiveness. According to these studies, proposed social media event detection approaches resulted in effective results, outperforming the recent competitive methods. For news media event detection, the proposed approach for document level extractions became the winning solution of CASE 2021 shared task 1 for the English language while being within the top four solutions for other languages. The other approaches proposed for fine-grained event detection of news media also returned more competitive results than the state-of-the-art methods.

9.3 Future Directions

There are many potential directions to extend this work in future. We already discussed the future directions targeting the scope of each sub-component within each contribution chapters (Chapters 4, 5, 7 and 8). Additionally, we describe some ideas for future work considering the bigger picture of this work below.

- **Code-mixed Language Processing:** We had to limit the experiments of this research to a few languages due to resource limitations. Still, we guaranteed the expandability of the proposed methods to different languages by not involving any language-specific features. However, in addition to processing multiple languages, the ability to process code-mixed languages is also important as they are popularly used in social media and to a certain extent in news media, especially by non-native English speaking countries (Saumya et al., 2021; Qureshi et al., 2019). Thus, it is worth exploring the performance of proposed approaches in code-mixed languages.

- **Other Domain Exploration:** In this research, we explored the proposed methods' performance in multiple diverse domains (i.e. sports, politics and protest). However, it would be interesting to extend the evaluations into more domains such as economy, disasters and epidemiology, considering their importance and demand (Amen et al., 2022; Prasad et al., 2023).

- **Other Corpora Development:** As a part of this research, we prepared and published social media event datasets covering two diverse domains (i.e. sports and politics), mainly due to the lack of recent data availability. We also annotated the data with sentiment labels to facilitate event

sentiment-based research. This initiative can be further extended to more domains and languages, including code-mixed languages, to support a wide range of future research.

- **Event Summarisation:** This research proposed approaches to extract event details from social media data streams and news media articles. The output would be more useful and easy to follow if natural language summaries were generated per event using the extracted details (Babu and Badugu, 2023).

- **Event Tracking:** This research only targeted the detection of events. However, it would be interesting to analyse the temporal evolution of detected events by tracking them. Event tracking would be especially beneficial for social media event analysis, considering the data dynamicity and user requirement to gain up-to-date knowledge (Houghton et al., 2019). In news media also, it is advantageous to analyse the event progression mainly in the long term to study the event behaviours and their impact.

- **Combined Knowledge Generation:** This research focused on separate event detections from social and news media. The obtained outputs can be merged to generate combined knowledge. Since social and news media have different characteristics, such a combination would hold a wide range of useful aspects. Additionally, the public opinions which can be extracted from social media also can be integrated to improve the informativeness of the outcome.

- **Detailed Visualisations:** The proposed approaches currently save the outputs as token lists or JSON objects, which are unfamiliar to non-technical people. Thus, designing an appropriate output visualisation

strategy, which is simple, straightforward, detailed and visually appealing, is an important area to focus on in future work to support the effective utilisation of event detection approaches by different people groups.

- **Web-based Application:** The main idea behind developing event detection mechanisms is to allow people to extract event details from social and news media effectively. The developed components need to be hosted as services via an application to fulfil this requirement. Unlike a standalone application, a web application would be more advantageous for an initial step, as it centralises the application, facilitating easy upgrades. Also, it provides access via the web through different devices, open to different people groups, regardless of their technical knowledge.

Bibliography

Adedoyin-Olowe, Mariam, Mohamed Medhat Gaber, Carlos M. Dancausa, Frederic Stahl, and João Bártolo Gomes (2016). "A rule dynamics approach to event detection in Twitter with its application to sports and politics". In: *Expert Systems with Applications* 55, pp. 351–360. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2016.02.028. URL: https://www.sciencedirect.com/science/article/pii/S0957417416300598.

Aiello, Luca Maria, Georgios Petkos, Carlos Martin, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Göker, Ioannis Kompatsiaris, and Alejandro Jaimes (2013). "Sensing Trending Topics in Twitter". In: *IEEE Transactions on Multimedia* 15.6, pp. 1268–1282. DOI: 10.1109/TMM.2013.2265080.

Aldhaheri, Abdulrahman and Jeongkyu Lee (2017). "Event detection on large social media using temporal analysis". In: *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1–6. DOI: 10.1109/CCWC.2017.7868467.

Alkhamees, Nora and Maria Fasli (2016). "Event detection from social network streams using frequent pattern mining with dynamic support values". In: *2016 IEEE International Conference on Big Data (Big Data)*, pp. 1670–1679. DOI: 10.1109/BigData.2016.7840781.

Allan, James, Ron Papka, and Victor Lavrenko (1998). "On-Line New Event Detection and Tracking". In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '98. Melbourne, Australia: Association for Computing Machinery, 37–45. ISBN: 1581130155. DOI: 10.1145/290941.290954.

Aloufi, Samah and Abdulmotaleb El Saddik (2018). "Sentiment Identification in Football-Specific Tweets". In: *IEEE Access* 6, pp. 78609–78621. DOI: 10.1109/ACCESS.2018.2885117.

Amen, Bakhtiar, Syahirul Faiz, and Thanh-Toan Do (2022). "Big data directed acyclic graph model for real-time COVID-19 twitter stream detection". In: *Pattern Recognition* 123, p. 108404. ISSN: 0031-3203. DOI: /10.1016/j.patcog.2021.108404.

Anandkumar, Animashree, Rong Ge, Daniel Hsu, Sham M Kakade, and Matus Telgarsky (2014). "Tensor decompositions for learning latent variable models". In: *Journal of machine learning research* 15, pp. 2773–2832.

Antoniak, Maria and David Mimno (2018). "Evaluating the Stability of Embedding-based Word Similarities". In: *Transactions of the Association for Computational Linguistics* 6, pp. 107–119. DOI: 10.1162/tacl_a_00008. URL: https://aclanthology.org/Q18-1008.

Atefeh, Farzindar and Wael Khreich (2015). "A Survey of Techniques for Event Detection in Twitter". In: *Computational Intelligence* 31.1, 132–164. ISSN: 0824-7935. DOI: 10.1111/coin.12017.

Awasthy, Parul, Jian Ni, Ken Barker, and Radu Florian (Aug. 2021). "IBM MNLP IE at CASE 2021 Task 1: Multigranular and Multilingual Event Detection on Protest News". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 138–146. DOI: 10.18653/v1/2021.case-1.18. URL: https://aclanthology.org/2021.case-1.18.

Babu, GL Anand and Srinivasu Badugu (2023). "A Survey on Automatic Text Summarisation". In: *Proceedings of Third International Conference on Advances in Computer Engineering and Communication Systems*. Springer, pp. 679–689. ISBN: 978-981-19-9228-5.

Balali, Ali, Masoud Asadpour, Ricardo Campos, and Adam Jatowt (2020). "Joint event extraction along shortest dependency paths using graph convolutional networks". In: *Knowledge-Based Systems* 210, p. 106492. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2020.106492. URL: https://www.sciencedirect.com/science/article/pii/S0950705120306213.

Basile, Angelo and Tommaso Caselli (2020). "Protest Event Detection: When Task-Specific Models Outperform an Event-Driven Method". In: *Lecture Notes in Computer Science*. Springer International Publishing, pp. 97–111. ISBN: 978-3-030-58219-7. DOI: 10.1007/978-3-030-58219-7_9.

Beltagy, Iz, Matthew E Peters, and Arman Cohan (2020). "Longformer: The long-document transformer". In: *arXiv preprint arXiv:2004.05150*.

Bengio, Yoshua, Réjean Ducharme, Pascal Vincent, and Christian Janvin (2003). "A Neural Probabilistic Language Model". In: *Journal of Machine Learning Research* 3, 1137–1155. ISSN: 1532-4435. DOI: 10.5555/944919.944966.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan (2003). "Latent Dirichlet Allocation". In: *Journal of Machine Learning Research* 3, 993–1022. ISSN: 1532-4435. DOI: 10.5555/944919.944937.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov (2017). "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. DOI: 10.1162/tacl_a_00051. URL: https://aclanthology.org/Q17-1010.

Büyüköz, Berfu, Ali Hürriyetoğlu, and Arzucan Özgür (May 2020). "Analyzing ELMo and DistilBERT on Socio-political News Classification". English. In: *Proceedings of the Workshop on Automated Extraction of Socio-political Events from News 2020*. Marseille, France: European Language Resources Association (ELRA), pp. 9–18. ISBN: 979-10-95546-50-4. URL: https://aclanthology.org/2020.aespen-1.4.

Canete, José, Gabriel Chaperon, Rodrigo Fuentes, Jou-Hui Ho, Hojin Kang, and Jorge Pérez (2020). "Spanish Pre-Trained BERT Model and Evaluation Data". In: *Proceedings of Practical ML for Developing Countries Workshop at International Conference on Learning Representations (ICLR 2020)*. URL: https://pml4dc.github.io/iclr2020/papers/PML4DC2020_10.pdf.

Castillo, Carlos, Marcelo Mendoza, and Barbara Poblete (2011). "Information Credibility on Twitter". In: *Proceedings of the 20th International Conference on World Wide Web*. WWW '11. Hyderabad, India: Association for Computing Machinery, 675–684. ISBN: 9781450306324. DOI: 10.1145/1963405.1963500.

Çelik, Furkan, Tuğberk Dalkılıç, Fatih Beyhan, and Reyyan Yeniterzi (Aug. 2021). "SU-NLP at CASE 2021 Task 1: Protest News Detection for English". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 131–137. DOI: 10.18653/v1/2021.case-1.17. URL: https://aclanthology.org/2021.case-1.17.

Chen, Chen and Vincent Ng (Dec. 2012). "Joint Modeling for Chinese Event Extraction with Rich Linguistic Features". In: *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*. Mumbai, India: The COLING 2012 Organizing Committee, pp. 529–544. URL: https://aclanthology.org/C12-1033.

Chen, Junyang, Zhiguo Gong, and Weiwen Liu (2019). "A Nonparametric Model for Online Topic Discovery with Word Embeddings". In: *Information Sciences* 504.C, 32–47. ISSN: 0020-0255. DOI: 10.1016/j.ins.2019.07.048.

Chen, Xi, Xiangmin Zhou, Timos Sellis, and Xue Li (2018). "Social event detection with retweeting behavior correlation". In: *Expert Systems with Applications* 114, pp. 516–523. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.08.022. URL: https://www.sciencedirect.com/science/article/pii/S095741741830530X.

Chen, Yubo, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao (July 2015). "Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks". In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Beijing, China: Association for

Computational Linguistics, pp. 167–176. DOI: 10.3115/v1/P15-1017. URL: https://aclanthology.org/P15-1017.

Choi, Hyeok-Jun and Cheong Hee Park (2019). "Emerging topic detection in twitter stream based on high utility pattern mining". In: *Expert Systems with Applications* 115, pp. 27–36. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.07.051. URL: https://www.sciencedirect.com/science/article/pii/S0957417418304767.

Comito, Carmela, Agostino Forestiero, and Clara Pizzuti (2019a). "Bursty Event Detection in Twitter Streams". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 13.4. ISSN: 1556-4681. DOI: 10.1145/3332185.

Comito, Carmela, Agostino Forestiero, and Clara Pizzuti (2019b). "Word Embedding Based Clustering to Detect Topics in Social Media". In: *IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. WI '19. Thessaloniki, Greece: Association for Computing Machinery, 192–199. ISBN: 9781450369343. DOI: 10.1145/3350546.3352518.

Conneau, Alexis, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov (July 2020). "Unsupervised Cross-lingual Representation Learning at Scale". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 8440–8451. DOI: 10.18653/v1/2020.acl-main.747. URL: https://aclanthology.org/2020.acl-main.747.

Conneau, Alexis and Guillaume Lample (2019). "Cross-lingual Language Model Pretraining". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett. Vol. 32. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf.

Corney, David, Carlos Martin, and Ayse Göker (2014). "Spot the Ball: Detecting Sports Events on Twitter". In: *Advances in Information Retrieval*. Ed. by Maarten de Rijke, Tom Kenter, Arjen P. de Vries, ChengXiang Zhai, Franciska de Jong, Kira Radinsky, and Katja Hofmann. Cham: Springer International Publishing, pp. 449–454. DOI: 10.1007/978-3-319-06028-6_40.

Cruz, Jan Christian Blaise, Julianne Agatha Tan, and Charibeth Cheng (May 2020). "Localization of Fake News Detection via Multitask Transfer Learning". English. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, pp. 2596–2604. ISBN: 979-10-95546-34-4. URL: https://aclanthology.org/2020.lrec-1.316.

Dadgar, Seyyed Mohammad Hossein, Mohammad Shirzad Araghi, and Morteza Mastery Farahani (2016). "A novel text mining approach based on TF-IDF and Support Vector Machine for news classification". In: *2016 IEEE International Conference on Engineering and Technology (ICETECH)*, pp. 112–116. DOI: 10.1109/ICETECH.2016.7569223.

Dai, Xiang-Ying, Qing-Cai Chen, Xiao-Long Wang, and Jun Xu (2010). "Online topic detection and tracking of financial news based on hierarchical clustering". In: *2010 International Conference on Machine Learning and Cybernetics*. Vol. 6, pp. 3341–3346. DOI: 10.1109/ICMLC.2010.5580677.

Day, Oscar and Taghi M Khoshgoftaar (2017). "A survey on heterogeneous transfer learning". In: *Journal of Big Data* 4.1, pp. 1–42. DOI: 10.1186/s40537-017-0089-0.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (June 2019). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: https://aclanthology.org/N19-1423.

Doddington, George, Alexis Mitchell, Mark Przybocki, Lance Ramshaw, Stephanie Strassel, and Ralph Weischedel (May 2004). "The Automatic Content Extraction (ACE) Program – Tasks, Data, and Evaluation". In: *Proceedings of the Fourth International Conference on Language Resources and Evaluation (LREC'04)*. Lisbon, Portugal: European Language Resources Association (ELRA). URL: http://www.lrec-conf.org/proceedings/lrec2004/pdf/5.pdf.

Edouard, Amosse, Elena Cabrio, Sara Tonelli, and Nhan Le-Thanh (Sept. 2017). "Graph-based Event Extraction from Twitter". In: *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*. Varna, Bulgaria: INCOMA Ltd., pp. 222–230. DOI: 10.26615/978-954-452-049-6_031.

Erp, MGJ van, G Rizzo, and R Troncy (2013). "Learning with the Web: Spotting Named Entities on the intersection of NERD and Machine Learning". In: *CEUR Workshop Proceedings*. CEUR Workshop Proceedings, pp. 27–30.

Ertugrul, Ali Mert, Burak Velioglu, and Pinar Karagoz (2017). "Word Embedding Based Event Detection on Social Media". In: *Hybrid Artificial Intelligent Systems*. Ed. by Francisco Javier Martínez de Pisón, Rubén Urraca, Héctor Quintián, and Emilio Corchado. Cham: Springer International Publishing, pp. 3–14. ISBN: 978-3-319-59650-1. DOI: 10.1007/978-3-319-59650-1_1.

García, Marcos Antonio Mouriño, Roberto Pérez Rodríguez, Manuel Vilares Ferro, and Luis Anido Rifón (2016). "Wikipedia-Based Hybrid Document Representation for Textual News Classification". In: *2016 3rd International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pp. 148–153. DOI: 10.1109/ISCMI.2016.31.

Godin, Fréderic, Baptist Vandersmissen, Wesley De Neve, and Rik Van de Walle (July 2015). "Multimedia Lab @ ACL WNUT NER Shared Task: Named Entity Recognition for Twitter Microposts using Distributed Word Representations". In: *Proceedings of the Workshop on Noisy User-generated Text*. Beijing, China: Association for Computational Linguistics, pp. 146–153. DOI: 10.18653/v1/W15-4322. URL: https://aclanthology.org/W15-4322.

Grishman, Ralph (2015). "Information Extraction". In: *IEEE Intelligent Systems* 30.5, pp. 8–15. DOI: 10.1109/MIS.2015.68.

Guille, Adrien and Cécile Favre (2015). "Event detection, tracking, and visualization in twitter: a mention-anomaly-based approach". In: *Social Network Analysis and Mining* 5.1, p. 18. DOI: 10.1007/s13278-015-0258-0.

Gürel, Alaeddin and Emre Emin (Aug. 2021). "ALEM at CASE 2021 Task 1: Multilingual Text Classification on News Articles". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 147–151. DOI: 10.18653/v1/2021.case-1.19. URL: https://aclanthology.org/2021.case-1.19.

Hasan, Mahmud, Mehmet A. Orgun, and Rolf Schwitter (2019). "Real-Time Event Detection from the Twitter Data Stream Using the TwitterNews+ Framework". In: *Information Processing & Management* 56.3, 1146–1165. ISSN: 0306-4573. DOI: 10.1016/j.ipm.2018.03.001.

Hassan, Abdalraouf and Ausif Mahmood (2017). "Deep learning for sentence classification". In: *2017 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, pp. 1–5. DOI: 10.1109/LISAT.2017.8001979.

Hassan, Abdalraouf and Ausif Mahmood (2018). "Convolutional Recurrent Deep Learning Model for Sentence Classification". In: *IEEE Access* 6, pp. 13949–13957. DOI: 10.1109/ACCESS.2018.2814818.

He, Xin, Yushi Chen, and Pedram Ghamisi (2020). "Heterogeneous Transfer Learning for Hyperspectral Image Classification Based on Convolutional Neural Network". In: *IEEE Transactions on Geoscience and Remote Sensing* 58.5, pp. 3246–3263. DOI: 10.1109/TGRS.2019.2951445.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (Aug. 2021a). "DAAI at CASE 2021 Task 1:

Transformer-based Multilingual Socio-political and Crisis Event Detection". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 120–130. DOI: 10.18653/v1/2021.case-1.16. URL: https://aclanthology.org/2021.case-1.16.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2021b). "Embed2Detect: Temporally Clustered Embedded Words for Event Detection in Social Media: Extended Abstract". In: *2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–2. DOI: 10.1109/DSAA53316.2021.9564110. URL: https://ieeexplore.ieee.org/document/9564110.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2022a). "Embed2Detect: Temporally clustered embedded words for event detection in social media". In: *Machine Learning* 111, pp. 49–87. DOI: 10.1007/s10994-021-05988-7.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2023a). "TTL: transformer-based two-phase transfer learning for cross-lingual news event detection". In: *International Journal of Machine Learning and Cybernetics*. DOI: 10.1007/s13042-023-01795-9.

Hettiarachchi, Hansi, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2023b). "WhatsUp: An event resolution approach for co-occurring events in social media". In: *Information Sciences* 625, pp. 553–577. ISSN: 0020-0255. DOI: 10.1016/j.ins.2023.01.001.

Hettiarachchi, Hansi, Doaa Al-Turkey, Mariam Adedoyin-Olowe, Jagdev Bhogal, and Mohamed Medhat Gaber (2022b). "TED-S: Twitter Event Data in Sports and Politics with Aggregated Sentiments". In: *Data* 7.7. ISSN: 2306-5729. DOI: 10.3390/data7070090. URL: https://www.mdpi.com/2306-5729/7/7/90.

Hettiarachchi, Hansi and Tharindu Ranasinghe (Nov. 2020). "InfoMiner at WNUT-2020 Task 2: Transformer-based Covid-19 Informative Tweet Extraction". In: *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*. Online: Association for Computational Linguistics, pp. 359–365. DOI: 10.18653/v1/2020.wnut-1.49. URL: https://aclanthology.org/2020.wnut-1.49.

Hettiarachchi, Hansi and Tharindu Ranasinghe (Aug. 2021). "TransWiC at SemEval-2021 Task 2: Transformer-based Multilingual and Cross-lingual Word-in-Context Disambiguation". In: *Proceedings of the 15th International Workshop on Semantic Evaluation (SemEval-2021)*. Online: Association for Computational Linguistics, pp. 771–779. DOI: 10.18653/v1/2021.semeval-1.102. URL: https://aclanthology.org/2021.semeval-1.102.

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997). "Long Short-Term Memory". In: *Neural Computation* 9.8, pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf.

Hong, Yu, Jianfeng Zhang, Bin Ma, Jianmin Yao, Guodong Zhou, and Qiaoming Zhu (June 2011). "Using Cross-Entity Inference to Improve Event Extraction". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, pp. 1127–1136. URL: https://aclanthology.org/P11-1113.

Houghton, James P, Michael Siegel, Stuart Madnick, Nobuaki Tounaka, Kazutaka Nakamura, Takaaki Sugiyama, Daisuke Nakagawa, and Buyanjargal Shirnen (2019). "Beyond Keywords: Tracking the evolution of conversational clusters in social media". In: *Sociological Methods & Research* 48.3, pp. 588–607.

Hu, Tiancheng and Niklas Stoehr (Aug. 2021). "Team "NoConflict" at CASE 2021 Task 1: Pretraining for Sentence-Level Protest Event Detection". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 152–160. DOI: 10.18653/v1/2021.case-1.20. URL: https://aclanthology.org/2021.case-1.20.

Huang, Jiajia, Min Peng, Hua Wang, Jinli Cao, Wang Gao, and Xiuzhen Zhang (2017). "A probabilistic method for emerging topic tracking in microblog stream". In: *World Wide Web* 20.2, pp. 325–350. DOI: 10.1007/s11280-016-0390-4.

Hürriyetoğlu, Ali, Osman Mutlu, Erdem Yörük, Farhana Ferdousi Liza, Ritesh Kumar, and Shyam Ratan (Aug. 2021a). "Multilingual Protest News Detection - Shared Task 1, CASE 2021". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 79–91. DOI: 10.18653/v1/2021.case-1.11. URL: https://aclanthology.org/2021.case-1.11.

Hürriyetoğlu, Ali, Erdem Yörük, Osman Mutlu, Fırat Duruşan, Çağrı Yoltar, Deniz Yüret, and Burak Gürel (June 2021b). "Cross-Context News Corpus for Protest Event-Related Knowledge Base Construction". In: *Data Intelligence* 3.2, pp. 308–335. ISSN: 2641-435X. DOI: 10.1162/dint_a_00092. eprint: https://direct.mit.edu/dint/article-pdf/3/2/308/1963469/dint_a_00092.pdf.

Huynh, Trung, Yulan He, Alistair Willis, and Stefan Rueger (Dec. 2016). "Adverse Drug Reaction Classification With Deep Neural Networks". In: *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, pp. 877–887. URL: https://aclanthology.org/C16-1084.

Irsan, Ivana and Masayu Khodra (2019). "Hierarchical multi-label news article classification with distributed semantic model based features". In: *International Journal of Advances in Intelligent Informatics* 5.1, pp. 40–47. ISSN: 2548-3161. DOI: 10.26555/ijain.v5i1.168. URL: https://ijain.org/index.php/IJAIN/article/view/168.

James, Josh (2019). *Data never sleeps 7.0. 2019*. URL: https://www.domo.com/learn/data-never-sleeps-7.

James, Josh (2021). *Data Never Sleeps 9.0*. URL: https://www.domo.com/learn/infographic/data-never-sleeps-9.

Jarvis, R.A. and E.A. Patrick (1973). "Clustering Using a Similarity Measure Based on Shared Near Neighbors". In: *IEEE Transactions on Computers* C-22.11, pp. 1025–1034. DOI: 10.1109/T-C.1973.223640.

Jing, How, Yu Tsao, Kuan-Yu Chen, and Hsin-Min Wang (Oct. 2013). "Semantic Naïve Bayes Classifier for Document Classification". In: *Proceedings of the Sixth International Joint Conference on Natural Language Processing*. Nagoya, Japan: Asian Federation of Natural Language Processing, pp. 1117–1123. URL: https://aclanthology.org/I13-1158.

Kalyan, Pawan, Duddukunta Reddy, Adeep Hande, Ruba Priyadharshini, Ratnasingam Sakuntharaj, and Bharathi Raja Chakravarthi (Aug. 2021). "IIITT at CASE 2021 Task 1: Leveraging Pretrained Language Models for Multilingual Protest Detection". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 98–104. DOI: 10.18653/v1/2021.case-1.13. URL: https://aclanthology.org/2021.case-1.13.

Karthikeyan, K, Zihan Wang, Stephen Mayhew, and Dan Roth (2020). "Cross-Lingual Ability of Multilingual BERT: An Empirical Study". In: *International Conference on Learning Representations*. URL: https://openreview.net/pdf?id=HJeT3yrtDr.

Kemp, Simon (2022). *Digital 2022: Global Overview Report*. URL: https://datareportal.com/reports/digital-2022-global-overview-report.

Kudo, Taku and John Richardson (Nov. 2018). "SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing". In: *Proceedings of the 2018 Conference on Empirical Methods in*

Natural Language Processing: System Demonstrations. Brussels, Belgium: Association for Computational Linguistics, pp. 66–71. DOI: 10.18653/v1/D18-2012. URL: https://aclanthology.org/D18-2012.

Kumar, Rama Bharath, Bangari Shravan Kumar, and Chandragiri Shiva Sai Prasad (2012). "Financial news classification using SVM". In: *International Journal of Scientific and Research Publications* 2.3, pp. 1–6.

Kwak, Haewoon, Changhyun Lee, Hosung Park, and Sue Moon (2010). "What is Twitter, a Social Network or a News Media?" In: *Proceedings of the 19th International Conference on World Wide Web*. WWW '10. Raleigh, North Carolina, USA: Association for Computing Machinery, 591–600. ISBN: 9781605587998. DOI: 10.1145/1772690.1772751.

Lample, Guillaume, Alexis Conneau, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou (2018). "Word translation without parallel data". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=H196sainb.

Lau, Jey Han, Nigel Collier, and Timothy Baldwin (Dec. 2012). "On-line Trend Analysis with Topic Models: #twitter Trends Detection Topic Model Online". In: *Proceedings of the 24th International Conference on Computational Linguistics (COLING)*. Mumbai, India: The COLING 2012 Organizing Committee, pp. 1519–1534. URL: https://aclanthology.org/C12-1093.

Lawrence, S., C.L. Giles, Ah Chung Tsoi, and A.D. Back (1997). "Face recognition: a convolutional neural-network approach". In: *IEEE Transactions on Neural Networks* 8.1, pp. 98–113. DOI: 10.1109/72.554195.

Le, Quoc and Tomas Mikolov (2014). "Distributed Representations of Sentences and Documents". In: *Proceedings of the 31st International Conference on Machine Learning*. Ed. by Eric P. Xing and Tony Jebara. Vol. 32. Proceedings of Machine Learning Research 2. Bejing, China: PMLR, pp. 1188–1196. URL: https://proceedings.mlr.press/v32/le14.html.

Le Nguyen, Hoai Nam and Quoc Ho Bao (2015). "A Combined Approach for Filter Feature Selection in Document Classification". In: *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 317–324. DOI: 10.1109/ICTAI.2015.56.

Lee, Chanhee, Kisu Yang, Taesun Whang, Chanjun Park, Andrew Matteson, and Heuiseok Lim (2021). "Exploring the Data Efficiency of Cross-Lingual Post-Training in Pretrained Language Models". In: *Applied Sciences* 11.5. ISSN: 2076-3417. DOI: 10.3390/app11051974. URL: https://www.mdpi.com/2076-3417/11/5/1974.

Lefever, Els and Véronique Hoste (May 2016). "A Classification-based Approach to Economic Event Detection in Dutch News Text". In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. Portorož, Slovenia: European Language Resources Association (ELRA), pp. 330–335. URL: https://aclanthology.org/L16-1051.

Li, Chenliang, Aixin Sun, and Anwitaman Datta (2012). "Twevent: Segment-Based Event Detection from Tweets". In: *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*. CIKM '12. Maui, Hawaii, USA: Association for Computing Machinery, 155–164. ISBN: 9781450311564. DOI: 10.1145/2396761.2396785.

Li, Jianxin, Zhenying Tai, Richong Zhang, Weiren Yu, and Lu Liu (2014). "Online Bursty Event Detection from Microblog". In: *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, pp. 865–870. DOI: 10.1109/UCC.2014.141.

Li, Qi, Heng Ji, and Liang Huang (Aug. 2013). "Joint Event Extraction via Structured Prediction with Global Features". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Sofia, Bulgaria: Association for Computational Linguistics, pp. 73–82. URL: https://aclanthology.org/P13-1008.

Li, Quanzhi, Armineh Nourbakhsh, Sameena Shah, and Xiaomo Liu (2017a). "Real-Time Novel Event Detection from Social Media". In: *2017 IEEE 33rd International Conference on Data Engineering (ICDE)*, pp. 1129–1139. DOI: 10.1109/ICDE.2017.157.

Li, Quanzhi, Sameena Shah, Xiaomo Liu, and Armineh Nourbakhsh (2017b). "Data sets: Word embeddings learned from tweets and general data". In: *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 11. 1, pp. 428–436.

Liang, Chen, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang (2020). "BOND: BERT-Assisted Open-Domain Named Entity Recognition with Distant Supervision". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. New York, NY, USA: Association for Computing Machinery, 1054–1064. ISBN: 9781450379984. URL: https://doi.org/10.1145/3394486.3403149.

Liddy, Elizabeth D (2001). "Natural language processing". In: *Encyclopedia of Library and Information Science*. NY: Marcel Decker, Inc. URL: https://surface.syr.edu/istpub/63/.

Lin, Ying, Heng Ji, Fei Huang, and Lingfei Wu (July 2020). "A Joint Neural Model for Information Extraction with Global Features". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, pp. 7999–8009. DOI: 10.

18653/v1/2020.acl-main.713. URL: https://aclanthology.org/2020.acl-main.713.

Lindén, Johannes, Stefan Forsström, and Tingting Zhang (2018). "Evaluating Combinations of Classification Algorithms and Paragraph Vectors for News Article Classification". In: *2018 Federated Conference on Computer Science and Information Systems (FedCSIS)*, pp. 489–495.

Liu, Shulin, Yang Li, Feng Zhang, Tao Yang, and Xinpeng Zhou (June 2019a). "Event Detection without Triggers". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 735–744. DOI: 10.18653/v1/N19-1080. URL: https://aclanthology.org/N19-1080.

Liu, Xiuwen, Jianming Fu, and Yanjiao Chen (2020). "Event evolution model for cybersecurity event mining in tweet streams". In: *Information Sciences* 524, pp. 254–276. ISSN: 0020-0255. DOI: 10.1016/j.ins.2020.03.048. URL: https://www.sciencedirect.com/science/article/pii/S0020025520302280.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019b). "Roberta: A robustly optimized bert pretraining approach". In: *arXiv preprint arXiv:1907.11692*.

Lu, Shudong, Si Li, Yajing Xu, Kai Wang, Haibo Lan, and Jun Guo (2022). "Event detection from text using path-aware graph convolutional network". In: *Applied Intelligence* 52.5, pp. 4987–4998. DOI: 10.1007/s10489-021-02695-7.

Maas, Andrew L., Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts (June 2011). "Learning Word Vectors for Sentiment Analysis". In: *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA: Association for Computational Linguistics, pp. 142–150. URL: https://aclanthology.org/P11-1015.

Maaten, Laurens van der and Geoffrey Hinton (2008). "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.86, pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze (2008). *Introduction to Information Retrieval*. USA: Cambridge University Press. ISBN: 0521865719.

Mathew, Binny, Punyajoy Saha, Seid Muhie Yimam, Chris Biemann, Pawan Goyal, and Animesh Mukherjee (2021). "HateXplain: A Benchmark Dataset

for Explainable Hate Speech Detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 17, pp. 14867–14875.

McCreadie, Richard, Craig Macdonald, Iadh Ounis, Miles Osborne, and Sasa Petrovic (2013). "Scalable distributed event detection for Twitter". In: *2013 IEEE International Conference on Big Data*, pp. 543–549. DOI: 10.1109/BigData.2013.6691620.

McMinn, Andrew J., Yashar Moshfeghi, and Joemon M. Jose (2013). "Building a Large-Scale Corpus for Evaluating Event Detection on Twitter". In: *Proceedings of the 22nd ACM International Conference on Information & Knowledge Management*. CIKM '13. San Francisco, California, USA: Association for Computing Machinery, 409–418. ISBN: 9781450322638. DOI: 10.1145/2505515.2505695.

Mehta, Sneha, Mohammad Raihanul Islam, Huzefa Rangwala, and Naren Ramakrishnan (2019). "Event Detection Using Hierarchical Multi-Aspect Attention". In: *The World Wide Web Conference*. WWW '19. San Francisco, CA, USA: Association for Computing Machinery, 3079–3085. ISBN: 9781450366748. DOI: 10.1145/3308558.3313659.

Mellin, Jonas and Mikael Berndtsson (2009). "Event Detection". In: *Encyclopedia of Database Systems*. Ed. by LING LIU and M. TAMER ÖZSU. Boston, MA: Springer US, pp. 1035–1040. ISBN: 978-0-387-39940-9. DOI: 10.1007/978-0-387-39940-9_506.

Merchant, Amil, Elahe Rahimtoroghi, Ellie Pavlick, and Ian Tenney (Nov. 2020). "What Happens To BERT Embeddings During Fine-tuning?" In: *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*. Online: Association for Computational Linguistics, pp. 33–44. DOI: 10.18653/v1/2020.blackboxnlp-1.4. URL: https://aclanthology.org/2020.blackboxnlp-1.4.

M'hamdi, Meryem, Marjorie Freedman, and Jonathan May (Nov. 2019). "Contextualized Cross-Lingual Event Trigger Extraction with Minimal Resources". In: *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*. Hong Kong, China: Association for Computational Linguistics, pp. 656–665. DOI: 10.18653/v1/K19-1061. URL: https://aclanthology.org/K19-1061.

Mikolov, Tomás, Kai Chen, Greg Corrado, and Jeffrey Dean (2013a). "Efficient Estimation of Word Representations in Vector Space". In: *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://arxiv.org/abs/1301.3781.

Mikolov, Tomas, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur (2010). "Recurrent neural network based language model." In: *Interspeech*. Vol. 2. 3. Makuhari, pp. 1045–1048.

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean (2013b). "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. Lake Tahoe, Nevada: Curran Associates Inc., 3111–3119. DOI: 10.5555/2999792.2999959.

Müllner, Daniel (2011). "Modern hierarchical, agglomerative clustering algorithms". In: *CoRR* abs/1109.2378. URL: http://dblp.uni-trier.de/db/journals/corr/corr1109.html#abs-1109-2378.

Mohamed, Tawfik A, Neamat El Gayar, and Amir F Atiya (2007). "A co-training approach for time series prediction with missing data". In: *International Workshop on Multiple Classifier Systems*. Springer, pp. 93–102. DOI: 10.1007/978-3-540-72523-7_10.

Moon, Seungwhan and Jaime Carbonell (2016). "Proactive transfer learning for heterogeneous feature and label spaces". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 706–721. DOI: 10.1007/978-3-319-46227-1_44.

Morabia, Keval, Neti Lalita Bhanu Murthy, Aruna Malapati, and Surender Samant (June 2019). "SEDTWik: Segmentation-based Event Detection from Tweets Using Wikipedia". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 77–85. DOI: 10.18653/v1/N19-3011. URL: https://aclanthology.org/N19-3011.

Mu, Lin, Peiquan Jin, Lizhou Zheng, En-Hong Chen, and Lihua Yue (2018). "Lifecycle-Based Event Detection from Microblogs". In: *Companion Proceedings of the The Web Conference 2018*. WWW '18. Lyon, France: International World Wide Web Conferences Steering Committee, 283–290. ISBN: 9781450356404. DOI: 10.1145/3184558.3186338.

Munkres, James (1957). "Algorithms for the assignment and transportation problems". In: *Journal of the Society for Industrial and Applied Mathematics* 5.1, pp. 32–38.

Naughton, M., N. Stokes, and J. Carthy (2010). "Sentence-Level Event Classification in Unstructured Texts". In: *Information Retrieval* 13.2, 132–156. ISSN: 1386-4564. DOI: 10.1007/s10791-009-9113-0.

Nguyen, Son, Bao Ngo, Chau Vo, and Tru Cao (2019). "Hot Topic Detection on Twitter Data Streams with Incremental Clustering Using Named Entities and Central Centroids". In: *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, pp. 1–6. DOI: 10.1109/RIVF.2019.8713730.

Nguyen, Thien Huu, Kyunghyun Cho, and Ralph Grishman (June 2016). "Joint Event Extraction via Recurrent Neural Networks". In: *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. San Diego, California: Association for Computational Linguistics, pp. 300–309. DOI: 10.18653/v1/N16-1034. URL: https://aclanthology.org/N16-1034.

Nugent, Tim, Fabio Petroni, Natraj Raman, Lucas Carstens, and Jochen L. Leidner (2017). "A comparison of classification models for natural disaster and critical event detection from news". In: *2017 IEEE International Conference on Big Data (Big Data)*, pp. 3750–3759. DOI: 10.1109/BigData.2017.8258374.

Nur'aini, Khumaisa, Ibtisami Najahaty, Lina Hidayati, Hendri Murfi, and Siti Nurrohmah (2015). "Combination of singular value decomposition and K-means clustering methods for topic detection on Twitter". In: *2015 International Conference on Advanced Computer Science and Information Systems (ICACSIS)*, pp. 123–128. DOI: 10.1109/ICACSIS.2015.7415168.

Panagiotou, Nikolaos, Ioannis Katakis, and Dimitrios Gunopulos (2016). "Detecting events in online social networks: Definitions, trends and challenges". In: *Solving Large Scale Learning Tasks. Challenges and Algorithms*, pp. 42–84. DOI: 10.1007/978-3-319-41706-6_2.

Pandey, Chandra, Zina Ibrahim, Honghan Wu, Ehtesham Iqbal, and Richard Dobson (2017). "Improving RNN with Attention and Embedding for Adverse Drug Reactions". In: *Proceedings of the 2017 International Conference on Digital Health*. DH '17. London, United Kingdom: Association for Computing Machinery, 67–71. ISBN: 9781450352499. DOI: 10.1145/3079452.3079501.

Papadopoulos, Symeon, Yiannis Kompatsiaris, Athena Vakali, and Ploutarchos Spyridonos (2012). "Community detection in social media: Performance and application considerations". In: *Data mining and knowledge discovery* 24, pp. 515–554. DOI: 10.1007/s10618-011-0224-z.

Pappagari, Raghavendra, Piotr Zelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak (2019). "Hierarchical Transformers for Long Document Classification". In: *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 838–844. DOI: 10.1109/ASRU46091.2019.9003958.

Parida, Shantipriya, Petr Motlicek, and Satya Ranjan Dash (2021). "German News Article Classification: A Multichannel CNN Approach". In: *Advances*

in Systems, Control and Automations. Springer, pp. 263–271. ISBN: 978-981-15-8685-9. DOI: 10.1007/978-981-15-8685-9_27.

Paszke, Adam, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc. DOI: 10.5555/3454287.3455008.

Peng, Min, Shuang Ouyang, Jiahui Zhu, Jiajia Huang, Hua Wang, and Jianming Yong (2018). "Emerging Topic Detection from Microblog Streams Based on Emerging Pattern Mining". In: *2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design ((CSCWD))*, pp. 259–264. DOI: 10.1109/CSCWD.2018.8465166.

Pennington, Jeffrey, Richard Socher, and Christopher Manning (Oct. 2014). "GloVe: Global Vectors for Word Representation". In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1532–1543. DOI: 10.3115/v1/D14-1162. URL: https://aclanthology.org/D14-1162.

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (June 2018). "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, pp. 2227–2237. DOI: 10.18653/v1/N18-1202. URL: https://aclanthology.org/N18-1202.

Piskorski, Jakub and Roman Yangarber (2013). "Information extraction: Past, present and future". In: *Multi-source, multilingual information extraction and summarization*. Springer, pp. 23–49. DOI: 10.1007/978-3-642-28569-1_2.

Pollard, Carl and Ivan A. Sag (1988). *Information-Based Syntax and Semantics: Vol. 1: Fundamentals*. USA: Center for the Study of Language and Information. ISBN: 0937073245.

Prasad, Rajesh, Akpan Uyime Udeme, Sanjay Misra, and Hashim Bisallah (2023). "Identification and classification of transportation disaster tweets using improved bidirectional encoder representations from transformers". In: *International Journal of Information Management Data Insights* 3.1, p. 100154. ISSN: 2667-0968. DOI: 10.1016/j.jjimei.2023.100154.

Qureshi, Sumera, Shoukat Ali Lohar, Syed Waqar Ali Shah, and Javed Iqbal Mirani (2019). "Investigating Code-Mixing in the Headlines and Advertisements of Sindhi Newspapers: Perceptions of Editors and Readers". In: *International European Extended Enablement in Science, Engineering & Management (IEEESEM)* 7.10. ISSN: 2320-9151.

Radford, Alec, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever (2018). "Improving language understanding by generative pre-training". In: *Technical Report, OpenAI*. URL: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf.

Rahane, Akshay and Sahil Pawar (2020). *Evolution of Natural Language Processing*. URL: https://www.freshgravity.com/evolution-of-natural-language-processing/.

Ramshaw, Lance and Mitch Marcus (1995). "Text Chunking using Transformation-Based Learning". In: *Third Workshop on Very Large Corpora*. URL: https://aclanthology.org/W95-0107.

Ranasinghe, Tharindu and Hansi Hettiarachchi (Dec. 2020). "BRUMS at SemEval-2020 Task 12: Transformer Based Multilingual Offensive Language Identification in Social Media". In: *Proceedings of the Fourteenth Workshop on Semantic Evaluation*. Barcelona (online): International Committee for Computational Linguistics, pp. 1906–1915. DOI: 10.18653/v1/2020.semeval-1.251. URL: https://aclanthology.org/2020.semeval-1.251.

Ranasinghe, Tharindu, Constantin Orasan, and Ruslan Mitkov (Dec. 2020). "TransQuest: Translation Quality Estimation with Cross-lingual Transformers". In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, pp. 5070–5081. DOI: 10.18653/v1/2020.coling-main.445. URL: https://aclanthology.org/2020.coling-main.445.

Ranasinghe, Tharindu and Marcos Zampieri (Nov. 2020). "Multilingual Offensive Language Identification with Cross-lingual Embeddings". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, pp. 5838–5844. DOI: 10.18653/v1/2020.emnlp-main.470. URL: https://aclanthology.org/2020.emnlp-main.470.

Ranasinghe, Tharindu, Marcos Zampieri, and Hansi Hettiarachchi (2019). "BRUMS at HASOC 2019: Deep Learning Models for Multilingual Hate Speech and Offensive Language Identification". In: *In Proceedings of the 11th annual meeting of the Forum for Information Retrieval Evaluation*. URL: http://ceur-ws.org/Vol-2517/T3-3.pdf.

Re, Francesco, Daniel Vegh, Dennis Atzenhofer, and Niklas Stoehr (Aug. 2021). "Team "DaDeFrNi" at CASE 2021 Task 1: Document and Sentence Classification for Protest Event Detection". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 171–178. DOI: 10.18653/v1/2021.case-1.22. URL: https://aclanthology.org/2021.case-1.22.

Rosenthal, Sara, Pepa Atanasova, Georgi Karadzhov, Marcos Zampieri, and Preslav Nakov (Aug. 2021). "SOLID: A Large-Scale Semi-Supervised Dataset for Offensive Language Identification". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, pp. 915–928. DOI: 10.18653/v1/2021.findings-acl.80. URL: https://aclanthology.org/2021.findings-acl.80.

Rosenthal, Sara, Noura Farra, and Preslav Nakov (Aug. 2017). "SemEval-2017 Task 4: Sentiment Analysis in Twitter". In: *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. Vancouver, Canada: Association for Computational Linguistics, pp. 502–518. DOI: 10.18653/v1/S17-2088. URL: https://aclanthology.org/S17-2088.

Roux, Maurice (2018). "A comparative study of divisive and agglomerative hierarchical clustering algorithms". In: *Journal of Classification* 35.2, pp. 345–366. DOI: 10.1007/s00357-018-9259-9.

Saeed, Zafar, Rabeeh Ayaz Abbasi, Onaiza Maqbool, Abida Sadaf, Imran Razzak, Ali Daud, Naif Radi Aljohani, and Guandong Xu (2019). "What's happening around the world? a survey and framework on event detection techniques on twitter". In: *Journal of Grid Computing* 17.2, pp. 279–312. DOI: 10.1007/s10723-019-09482-2.

Sagen, Markus (2021). "Large-Context Question Answering with Cross-Lingual Transfer". MA thesis. Uppsala University, Department of Information Technology, p. 45.

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf (2019). "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv preprint arXiv:1910.01108*.

Saumya, Sunil, Abhinav Kumar, and Jyoti Prakash Singh (Apr. 2021). "Offensive language identification in Dravidian code mixed social media text". In: *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages*. Kyiv: Association for Computational Linguistics, pp. 36–45. URL: https://aclanthology.org/2021.dravidianlangtech-1.5.

Sayyadi, Hassan, Matthew Hurst, and Alexey Maykov (2009). "Event detection and tracking in social streams". In: *Third International AAAI Conference on Weblogs and Social Media*. DOI: 10.1609/icwsm.v3i1.13970.

Schakel, Adriaan M. J. and Benjamin J. Wilson (2015). "Measuring Word Significance using Distributed Representations of Words". In: *CoRR* abs/1508.02297. arXiv: 1508.02297. URL: http://arxiv.org/abs/1508.02297.

Schinas, Manos, Symeon Papadopoulos, Georgios Petkos, Yiannis Kompatsiaris, and Pericles A. Mitkas (2015). "Multimodal Graph-Based Event Detection and Summarization in Social Media Streams". In: *Proceedings of the 23rd ACM International Conference on Multimedia*. MM '15. Brisbane, Australia: Association for Computing Machinery, 189–192. ISBN: 9781450334594. DOI: 10.1145/2733373.2809933.

Sennrich, Rico, Barry Haddow, and Alexandra Birch (Aug. 2016). "Neural Machine Translation of Rare Words with Subword Units". In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, pp. 1715–1725. DOI: 10.18653/v1/P16-1162. URL: https://aclanthology.org/P16-1162.

Sha, Lei, Feng Qian, Baobao Chang, and Zhifang Sui (2018). "Jointly Extracting Event Triggers and Arguments by Dependency-Bridge RNN and Tensor-Based Argument Interaction". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1. URL: https://ojs.aaai.org/index.php/AAAI/article/view/12034.

Shearer, Elisa and Jeffrey Gottfried (2017). "News use across social media platforms 2017". In: URL: https://www.journalism.org/2017/09/07/news-use-across-social-media-platforms-2017/.

Shi, Xiaoxiao, Qi Liu, Wei Fan, Philip S. Yu, and Ruixin Zhu (2010). "Transfer Learning on Heterogenous Feature Spaces via Spectral Transformation". In: *2010 IEEE International Conference on Data Mining*, pp. 1049–1054. DOI: 10.1109/ICDM.2010.65.

Škrlj, Blaž, Jan Kralj, and Nada Lavrač (2020). "Embedding-based Silhouette community detection". In: *Machine Learning* 109.11, pp. 2161–2193. DOI: 10.1007/s10994-020-05882-8.

Small, Sharon Gower and Larry Medsker (2014). "Review of information extraction technologies and applications". In: *Neural Computing and Applications* 25.3-4, pp. 533–548. DOI: 10.1007/s00521-013-1516-6.

Souza, Fábio, Rodrigo Nogueira, and Roberto Lotufo (2020). "BERTimbau: pretrained BERT models for Brazilian Portuguese". In: *9th Brazilian Conference*

on Intelligent Systems, BRACIS, Rio Grande do Sul, Brazil, October 20-23 (to appear). Rio Grande, Brazil: Springer-Verlag, 403–417. ISBN: 978-3-030-61376-1. DOI: 10.1007/978-3-030-61377-8_28.

Sutanto, Taufik and Richi Nayak (2018). "Fine-grained document clustering via ranking and its application to social media analytics". In: *Social Network Analysis and Mining* 8.1, pp. 1–19. DOI: 10.1007/s13278-018-0508-z.

Takaffoli, Mansoureh, Farzad Sangi, Justin Fagnan, and Osmar R. Zaïane (2011). "MODEC - Modeling and Detecting Evolutions of Communities". In: *Fifth international AAAI conference on weblogs and social media*. Ed. by Lada A. Adamic, Ricardo Baeza-Yates, and Scott Counts. The AAAI Press, pp. 626–629. URL: http://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/view/2853.

Tjong Kim Sang, Erik F. and Fien De Meulder (2003). "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*, pp. 142–147. URL: https://aclanthology.org/W03-0419.

Tsai, Pauray S. M. (2009). "Mining Frequent Itemsets in Data Streams Using the Weighted Sliding Window Model". In: *Expert Systems with Applications* 36.9, 11617–11625. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2009.03.025.

Unankard, Sayan and Wanvimol Nadee (2020). "Sub-Events Tracking from Social Network based on the Relationships between Topics". In: *2020 Joint International Conference on Digital Arts, Media and Technology with ECTI Northern Section Conference on Electrical, Electronics, Computer and Telecommunications Engineering (ECTI DAMT NCON)*, pp. 1–6. DOI: 10.1109/ECTIDAMTNCON48261.2020.9090732.

van der Meer, Toni G.L.A. and Piet Verhoeven (2013). "Public framing organizational crisis situations: Social media versus news media". In: *Public Relations Review* 39.3, pp. 229–231. ISSN: 0363-8111. DOI: 10.1016/j.pubrev.2012.12.001. URL: https://www.sciencedirect.com/science/article/pii/S0363811113000027.

Van Oorschot, Guido, Marieke Van Erp, and Chris Dijkshoorn (2012). "Automatic Extraction of Soccer Game Events from Twitter." In: *Proceedings of the Workhop on Detection, Representation, and Exploitation of Events in the Semantic Web (DeRiVE 2012)*, pp. 21–30.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin (2017). "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc. URL: https://proceedings.

neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Vivek Kalyan, Sureshkumar, Tan Paul, Tan Shaun, and Martin Andrews (Aug. 2021). "Handshakes AI Research at CASE 2021 Task 1: Exploring different approaches for multilingual tasks". In: *Proceedings of the 4th Workshop on Challenges and Applications of Automated Extraction of Socio-political Events from Text (CASE 2021)*. Online: Association for Computational Linguistics, pp. 92–97. DOI: 10.18653/v1/2021.case-1.12. URL: https://aclanthology.org/2021.case-1.12.

Wang, Zhengjue, Chaojie Wang, Hao Zhang, Zhibin Duan, Mingyuan Zhou, and Bo Chen (2020). "Learning Dynamic Hierarchical Topic Graph with Graph Convolutional Network for Document Classification". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, pp. 3959–3969. URL: https://proceedings.mlr.press/v108/wang20l.html.

Weiler, Andreas, Michael Grossniklaus, and Marc H. Scholl (Sept. 2016). "Survey and Experimental Analysis of Event Detection Techniques for Twitter". In: *The Computer Journal* 60.3, pp. 329–346. ISSN: 0010-4620. DOI: 10.1093/comjnl/bxw056. eprint: https://academic.oup.com/comjnl/article-pdf/60/3/329/11043116/bxw056.pdf.

Weiss, Karl, Taghi M Khoshgoftaar, and DingDing Wang (2016). "A survey of transfer learning". In: *Journal of Big data* 3.1, pp. 1–40. DOI: 10.1186/s40537-016-0043-6.

Wenzek, Guillaume, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave (May 2020). "CCNet: Extracting High Quality Monolingual Datasets from Web Crawl Data". English. In: *Proceedings of the 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, pp. 4003–4012. ISBN: 979-10-95546-34-4. URL: https://aclanthology.org/2020.lrec-1.494.

Wolf, Thomas, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush (Oct. 2020). "Transformers: State-of-the-Art Natural Language Processing". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. URL: https://aclanthology.org/2020.emnlp-demos.6.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean (2016). "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". In: *CoRR* abs/1609.08144. arXiv: 1609.08144. URL: http://arxiv.org/abs/1609.08144.

Xie, Wei, Feida Zhu, Jing Jiang, Ee-Peng Lim, and Ke Wang (2016). "TopicSketch: Real-Time Bursty Topic Detection from Twitter". In: *IEEE Transactions on Knowledge and Data Engineering* 28.8, pp. 2216–2229. DOI: 10.1109/TKDE.2016.2556661.

Xu, Xiaowei, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger (2007). "SCAN: A Structural Clustering Algorithm for Networks". In: *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '07. San Jose, California, USA: Association for Computing Machinery, 824–833. ISBN: 9781595936097. DOI: 10.1145/1281192.1281280.

Yang, Sen, Dawei Feng, Linbo Qiao, Zhigang Kan, and Dongsheng Li (July 2019a). "Exploring Pre-trained Language Models for Event Extraction and Generation". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, pp. 5284–5294. DOI: 10.18653/v1/P19-1522. URL: https://aclanthology.org/P19-1522.

Yang, Wei, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin (June 2019b). "End-to-End Open-Domain Question Answering with BERTserini". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*. Minneapolis, Minnesota: Association for Computational Linguistics, pp. 72–77. DOI: 10.18653/v1/N19-4013. URL: https://aclanthology.org/N19-4013.

Yang, Yiming, Tom Pierce, and Jaime Carbonell (1998). "A Study of Retrospective and On-Line Event Detection". In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '98. Melbourne, Australia: Association for Computing Machinery, 28–36. ISBN: 1581130155. DOI: 10.1145/290941.290953.

Yang, Zichao, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy (June 2016). "Hierarchical Attention Networks for Document Classification". In: *Proceedings of the 2016 Conference of the North American Chapter*

of the Association for Computational Linguistics: Human Language Technologies. San Diego, California: Association for Computational Linguistics, pp. 1480–1489. DOI: 10.18653/v1/N16-1174. URL: https://aclanthology.org/N16-1174.

Yilmaz, Seyhmus and Sinan Toklu (2020). "A deep learning analysis on question classification task using Word2vec representations". In: *Neural Computing and Applications* 32, pp. 2909–2928. DOI: 10.1007/s00521-020-04725-w.

Yin, Jie, Andrew Lampert, Mark Cameron, Bella Robinson, and Robert Power (2012). "Using Social Media to Enhance Emergency Situation Awareness". In: *IEEE Intelligent Systems* 27.6, pp. 52–59. DOI: 10.1109/MIS.2012.6.

Zaheer, Manzil, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed (2020). "Big Bird: Transformers for Longer Sequences". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., pp. 17283–17297. URL: https://proceedings.neurips.cc/paper/2020/file/c8512d142a2d849725f31a9a7a361ab9-Paper.pdf.

Zhang, Lemei, Peng Liu, and Jon Atle Gulla (2019). "Dynamic attention-integrated neural network for session-based news recommendation". In: *Machine Learning* 108.10, pp. 1851–1875. DOI: 10.1007/s10994-018-05777-9.

Zhang, Yu and Qiang Yang (2021). "A Survey on Multi-Task Learning". In: *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1. DOI: 10.1109/TKDE.2021.3070203.

Zhou, Y. and S. Goldman (2004). "Democratic co-learning". In: *16th IEEE International Conference on Tools with Artificial Intelligence*, pp. 594–602. DOI: 10.1109/ICTAI.2004.48.

Zhuang, Fuzhen, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He (2021). "A Comprehensive Survey on Transfer Learning". In: *Proceedings of the IEEE* 109.1, pp. 43–76. DOI: 10.1109/JPROC.2020.3004555.

Zoph, Barret, Deniz Yuret, Jonathan May, and Kevin Knight (Nov. 2016). "Transfer Learning for Low-Resource Neural Machine Translation". In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, pp. 1568–1575. DOI: 10.18653/v1/D16-1163. URL: https://aclanthology.org/D16-1163.

Appendix A

Social Media Event Detection

A.1 Embed2Detect Processing Time

TABLE A.1: Embed2Detect intermediate processing time on Common KVM CPU @ 2.40GHz - MUNLIV

Step	**Time(s): 1 worker**		**Time(s): 8 workers**	
	Total	Average	Total	Average
Stream Chunker	66	1.1579	64	1.1228
Word Embedding Learner	114	2.0000	90	1.5789
Event Window Identifier	35	0.6140	34	0.5965
Event Word Extractor	0	0.0000	0	0.0000
Full Process	230	4.0351	202	3.5439

TABLE A.2: Embed2Detect intermediate processing time on Common KVM CPU @ 2.40GHz - BrexitVote

Step	**Time(s): 1 worker**		**Time(s): 8 workers**	
	Total	Average	Total	Average
Stream Chunker	28	2.5455	27	2.4545
Word Embedding Learner	168	15.2727	78	7.0909
Event Window Identifier	562	51.0909	139	12.6364
Event Word Extractor	29	2.6364	29	2.6364
Full Process	824	74.9091	310	28.1818

We did an intermediate analysis to understand the complexities of Embed2Detect's components. The obtained results are summarised in Table A.1

and A.2. Total time reports the time taken to process whole corpora (57 2-minute windows of MUNLIV and 11 30-minute windows of BrexitVote), and average time reports the time taken per window. Comparatively, word embedding learner and event window identifier are the most time-consuming components in Embed2Detect. Their time could be notably reduced with parallel processing, making Embed2Detect more appropriate for real-time processing. However, even with sequential processing (or single worker), the complete process can be run in a very short time than the time taken to generate data.

A.2 WhatsUp Event Clusters

In addition to quantitative analyses reported in Chapter 5, we also analysed the quality of events detected by WhatsUp, considering the word coverage and novelty measures. Table A.3 summarises a few event windows with detected events that match with GT events. We only report events from the MUNLIV dataset, considering their briefness and simplicity than the BrexitVote events. For MUNLIV, both s- and m-based clustering performed well with κ (Section 5.2.3). Among them, we used (m, κ) combination to extract events because m fixes the number of words in a cluster easing the analysis and reporting.

As shown in Table A.3, detected events include the majority of keywords in GT events. Additionally, event-related misspelt words, other/background words, and emojis are also recognised, highlighting the potential of involved linguistical features. Furthermore, the novelty measure provides a positive insight into the newness of an event. However, comparing novelty values across different event windows is less effective as they highly depend on the underlying data. In the given samples, novelty varied from 0.30 to 0.44 for top-ranked events within the windows. BrexitVote events also indicated a similar nature

TABLE A.3: Sample events detected by WhatsUp for MUNLIV. TW, Nov. and R stand for time window, novelty and rank, respectively. Rank indicates the event position within the corresponding time window based on the descending order of event novelties.

TW	GT Event	Detected Event	Nov.	R
16:52-16:54	Marcus Rashford is one-on-one with Virgil van Dijk on the right touchline and gets fortunate as the LFC defender slips.	bullied, see, mufc, love, marcus, rashford, 21, dijk, puts, van, floor, mulive, promising, start, old, trafford, chances, far, rt, counter-attack, either, followers, side, 😂, applying	0.4013	1
16:56-16:58	Scott McTominay (MUFC) right footed shot from outside the box is saved in the centre of the goal by Alisson.	hosts, mctominay, 0-0, distance, effort, alisson, fires, goalwards, saves, afternoon, old, trafford, 😂 fixture, like, smells, 👎, love, marcus, rashford, see, van, man, mufc, reds	0.3024	1
17:06-17:08	Goal by Marcus Rashford (MUFC)! Manchester United 1, Liverpool 0.	goooaaalll, @marcusrashford, munliv, mufc, man, marcus, rashford, 1-0, 36, goal, utd, 😱, gives, lead, 0, 1, liverpool, manchester, united, ⚪, ⚽, 🔴, 0-0, buuut	0.4430	1
	VAR review is in progress.	var, f**king, great, love, see, atkinson, back, game, win, going, martin, origi, daniel, dijk, range, van, today, far, players, @manutd, come, front, f**k, get, ggmu	0.2908	6
17:14-17:16	Sadio Mané (LFC) scores but the goal is ruled out after a VAR review.	44, disallows, let, mulive, var, equalises, mane, ruled, sadio, fans, loving, man, utd, goal, handball, overturned, disallowed, mane's, 1-0, golazo, hosts, remains, every, f**k, get	0.3914	1

in word coverage and novelty measure. However, they formed larger clusters with fewer emojis, following the event complexities and audience in the political domain.

A.3 WhatsUp Processing Time

TABLE A.4: WhatsUp(s, κ) intermediate processing time on Common KVM CPU @ 2.40GHz - MUNLIV

Step	**Time(s): 1 worker**		**Time(s): 8 workers**	
	Total	Average	Total	Average
Stream Chunker	66	1.1579	69	1.2105
Word Embedding Learner	114	2.0000	87	1.5263
Statistical Information Extractor	1	0.0175	1	0.0175
Event Window Identifier	33	0.5789	31	0.5439
Event Cluster Detector	8	0.1404	8	0.1404
Full Process	235	4.1228	206	3.6140

TABLE A.5: WhatsUp(s, κ) intermediate processing time on Common KVM CPU @ 2.40GHz - BrexitVote

Step	**Time(s): 1 worker**		**Time(s): 8 workers**	
	Total	Average	Total	Average
Stream Chunker	31	2.8182	32	2.9091
Word Embedding Learner	192	17.4545	73	6.6364
Statistical Information Extractor	3	0.2727	3	0.2727
Event Window Identifier	573	52.0909	152	13.8182
Event Cluster Detector	72	6.5455	65	5.9091
Full Process	899	81.7273	354	32.1818

We analysed the time taken by each step to understand the intermediate complexities of WhatsUp. Table A.4 and A.5 summarises the obtained results. Total time indicates the time taken to process full corpora (57 2-minute windows of MUNLIV and 11 30-minute windows of BrexitVote), and average time indicates the time taken per window. Word embedding learner and event window identifier are the most time complex components of WhatsUp. However, they are sufficiently fast to complete the full process in a very short time than

the time taken to generate data, even with sequential processing. With parallel processing, these components' speed can be further increased to facilitate (near) real-time processing.

Appendix B

News Media Event Detection

B.1 Transformer Resource Utilisation

TABLE B.1: Memory usage and inference speed of transformer-based models built for news media event detection on GeForce RTX 3090 GPU and Intel(R) Xeon(R) CPU @ 2.30GHz

Transformer	Disk Usage (MB)	GPU		CPU	
		RAM (MB)	Time (s)	RAM (MB)	Time (s)
BERT	1274	6539	0.0562	3295	3.7190
BERTimbau	1277	6540	0.0560	3312	6.5957
BETO	420	5482	0.0481	1587	1.8834
Longformer(Base)	570	5270	0.1599	2292	3.0372
BigBird(Base)	489	5485	0.2063	1716	4.5594
BigBird(Large)	1372	5296	0.4265	3482	4.3118
mBERT	682	5503	0.1385	2101	1.9229
XLM-R	2150	7680	0.7040	5018	6.9053

The involvement of large pre-trained language models/transformers to build text classifiers could affect the usability of the resulting models. Therefore we analysed the memory usage and inference speed of built classifiers, and our findings are summarised in Table B.1. Space requirements or inference time of a model do not vary depending on the fine-tuned data or classification heads. Thus, the reported values are common for document, sentence and token level models built using each transformer. Overall, the model size ranges from 400 MB to 2,000 MB, depending on the transformer model utilised. The

inference is clearly faster on a GPU with less time than a second. A CPU also takes a few seconds (<7 seconds), confirming the efficiency of these models appropriate for (near) real-time predictions.