

# PhD Thesis

---

## Coping with low data availability for social media crisis message categorisation

Congcong Wang

**Student Number:** 14207199

---

The thesis is submitted to University College Dublin in fulfilment  
of the degree of Doctor of Philosophy in Computer Science

**Supervisor:** Dr. David Lillis

**Members of Research Studies Panel:**

Assoc. Prof. John Dunnion  
Assoc. Prof. Mark Scanlon

**Head of School:** Assoc. Prof. Neil Hurley



UCD School of Computer Science  
College of Science  
University College Dublin

May 2023

# Table of Contents

<b>Abstract</b> . . . . .	<b>xi</b>
<b>1 Introduction</b> . . . . .	<b>1</b>
1.1 Research context . . . . .	1
1.2 Research approaches . . . . .	3
1.3 Research questions . . . . .	5
1.4 Research contributions . . . . .	6
1.5 Thesis outline . . . . .	7
<b>2 Background</b> . . . . .	<b>9</b>
2.1 Machine learning . . . . .	9
2.1.1 Key concepts . . . . .	9
2.1.2 Linear regression . . . . .	13
2.2 Neural networks . . . . .	17
2.2.1 Pre-trained word embeddings . . . . .	20
2.2.2 Convolutional neural network . . . . .	22
2.2.3 Recurrent neural network . . . . .	24
2.2.4 Pre-trained language models . . . . .	26
2.2.4.1 Encoder models . . . . .	26
2.2.4.2 Decoder models . . . . .	32
2.2.4.3 Encoder-Decoder models . . . . .	33
2.3 Conclusions . . . . .	35
<b>3 Crisis Messages Categorisation in Low-data Settings</b> . . . . .	<b>36</b>
3.1 Crisis domain adaptation . . . . .	37
3.1.1 Target data dependent domain adaptation . . . . .	39
3.1.1.1 Instance selection methods . . . . .	39
3.1.1.2 Feature adjustment methods . . . . .	43
3.1.2 Target data independent domain adaptation . . . . .	45
3.2 Crisis few-shot learning . . . . .	47
3.2.1 Prompt-based learning methods . . . . .	48
3.2.2 Augmentation based methods . . . . .	50
3.3 Crisis zero-shot learning . . . . .	52

3.3.1	Indirectly-supervised learning methods . . . . .	52
3.3.2	Weakly-supervised learning methods . . . . .	54
3.4	Related datasets and evaluation metrics . . . . .	56
3.4.1	Informativeness . . . . .	56
3.4.2	Information types . . . . .	57
3.4.3	Beyond crisis . . . . .	62
3.5	Conclusions . . . . .	64
<b>4</b>	<b>Many-to-many Crisis Domain Adaptation . . . . .</b>	<b>66</b>
4.1	Introduction . . . . .	66
4.2	Many-to-many adaptation . . . . .	67
4.2.1	Single-task learning with machine learning and neural networks . . . . .	68
4.2.1.1	Traditional machine learning as the classifier . . . . .	70
4.2.1.2	Neural networks as the classifier . . . . .	71
4.2.1.3	Evaluation and discussion . . . . .	72
4.2.2	Multi-task learning with pre-trained language models . . . . .	75
4.2.2.1	Method . . . . .	75
4.2.2.2	Experiments and results . . . . .	78
4.2.2.3	Error analysis . . . . .	86
4.3	Conclusions . . . . .	88
<b>5</b>	<b>Many-to-one and one-to-one Crisis Domain Adaptations . . . . .</b>	<b>90</b>
5.1	Introduction . . . . .	90
5.2	Many-to-one and one-to-one adaptations . . . . .	90
5.2.1	Event-aware sequence-to-sequence pre-trained language models . . . . .	91
5.2.2	Experiments setup . . . . .	92
5.2.2.1	Datasets . . . . .	92
5.2.2.2	Adaptation scenarios . . . . .	93
5.2.2.3	Training details . . . . .	94
5.2.3	Discussions and Findings . . . . .	94
5.2.3.1	One-to-one adaptation . . . . .	95
5.2.3.2	Many-to-one adaptation . . . . .	99
5.3	Conclusions . . . . .	101
<b>6</b>	<b>Augmentation for Crisis Few-shot Learning . . . . .</b>	<b>103</b>
6.1	Introduction . . . . .	103
6.2	Self-Controlled Text Augmentation (STA) . . . . .	104
6.2.1	Self-controlled method . . . . .	105
6.2.1.1	Prompts-based multi-task training in seq2seq Models . . . . .	107
6.2.1.2	Data generation, self-checking and selection . . . . .	108
6.2.2	Experiments . . . . .	109

6.2.2.1	Dataset . . . . .	109
6.2.2.2	Training details . . . . .	109
6.2.2.3	Baselines . . . . .	110
6.2.3	Results and discussion . . . . .	111
6.2.3.1	Classification performance . . . . .	111
6.2.3.2	Ablation Study . . . . .	112
6.2.3.3	Investigation of Lexical Diversity and Semantic Fidelity	113
6.2.4	Generalisation to domains beyond crisis . . . . .	115
6.2.5	Summary . . . . .	117
6.3	Iterative self-controlled augmentation . . . . .	119
6.3.1	Iterative self-controlled method . . . . .	120
6.3.1.1	Iterative augmentation . . . . .	120
6.3.1.2	Generation de-duplication . . . . .	121
6.3.1.3	Model refinement using unlabelled data . . . . .	122
6.3.2	Experiments . . . . .	123
6.3.2.1	Experimental Setup . . . . .	123
6.3.2.2	Baselines . . . . .	124
6.3.3	Results and discussion . . . . .	125
6.3.3.1	Downstream classification . . . . .	125
6.3.3.2	Quality of generated texts . . . . .	125
6.3.3.3	Generalisation to domains beyond crisis . . . . .	127
6.3.4	Summary . . . . .	128
6.4	Conclusions . . . . .	130
<b>7</b>	<b>Using Pseudo-labelled Data For Crisis Zero-shot Learning . . . . .</b>	<b>132</b>
7.1	Introduction . . . . .	133
7.2	Method . . . . .	133
7.2.1	Label expansion . . . . .	134
7.2.2	Pseudo label assignment . . . . .	135
7.2.3	Pre-training and self-training . . . . .	137
7.3	Experiments . . . . .	138
7.3.1	Datasets . . . . .	138
7.3.2	Experimental details . . . . .	139
7.3.3	Baselines . . . . .	139
7.3.4	Results and discussion . . . . .	141
7.3.4.1	Comparison with baselines . . . . .	141
7.3.4.2	Ablation study . . . . .	142
7.3.4.3	Pseudo-labelled data quality . . . . .	142
7.3.4.4	Generalisation to domains beyond crisis . . . . .	143
7.4	Conclusion . . . . .	146

<b>8 Conclusion</b>	<b>147</b>
8.1 Summary of contributions	147
8.2 Future directions	150
<b>A Appendix</b>	<b>171</b>
A.1 Information types of TREC-IS	171

# **Statement of Original Authorship**

---

"I hereby certify that the submitted work is my own work, was completed while registered as a candidate for the degree stated on the Title Page, and I have not obtained a degree elsewhere on the basis of the research presented in this submitted work."

# Dedication

---

To my grandmother for her love.

# Acknowledgements

---

The pursuit of a Ph.D. degree can be a lengthy and isolating journey, one that would not have been possible without the help and support of numerous individuals. Firstly, I would like to extend my sincere gratitude to my supervisor, David Lillis. His unwavering support, patience, and encouragement have been invaluable. As a mentor, he provides me with constructive feedback on my research and as a friend, he inspires me when I experience feelings of self-doubt. Without his assistance, it would have been difficult to imagine how I could have made it this far on this journey.

I am grateful for the guidance and supervision provided by Paul Nulty in some of my research projects. His deep understanding of the domain and innovative ideas have been invaluable to my research work. Without his dedicated efforts, I would have faced significant challenges in completing some of my research work.

I am also grateful to my research committee members, John Dunnion and Mark Scanlon, for their guidance and support throughout my research. Their constructive feedback and suggestions have been invaluable in moving my work forward. Additionally, I would like to express my appreciation to the professors and staff in the school who have provided timely assistance and support whenever I have reached out to them.

Furthermore, I extend my gratitude to Tri Kurniawan Wijaya, Gonzalo Fiz Pontiveros, Steven Derby, and Puchao Zhang from Huawei IRC. Their valuable insights and suggestions have contributed to the improvement of my research work.

Lastly, I am deeply appreciative of my friends and family, whose unwavering support has been instrumental in my journey. I would like to extend my heartfelt gratitude to my colleague and dear friend Chidubem Iddianozie, who has been like a brother to me and has provided invaluable assistance not only in my research but also in my personal life. I am also grateful for the presence of my girlfriend Xiaoyu Du in my life since the beginning of this journey. It has been a wonderful experience to have her by my side, offering kindness, support, and inspiration during the tough times. I wish to express my special thanks to my parents for their unrelenting efforts in providing me with a good education since childhood. Their financial and emotional support have made a significant difference in shaping the person I am today. Lastly, I am indebted to my grandmother for her upbringing and nurturing during my formative years. I realise that words alone cannot fully convey my gratitude for her kindness and care.

# List of Abbreviations

---

$s$	An input sequence or instance
$t$	Token or word
$\mathbf{x}$	Vector representing input features or representations of a sequence or sample
$x_i$	The i-th feature of a sequence or sample
$\mathbf{X}$	Matrix representing input features or representations of a dataset
$Y$	The class or label space
$y$	A class or label
$n$	Number of sequences or samples in a dataset
$m$	Number of classes
$k$	Feature or embedding size
$T$	Number of tokens of a sequence
$\theta$	Model parameters
$p$	Probability density function
<b>W</b>	Weight matrix
<b>b</b>	Bias
$\alpha$	Learning rate
$J$	Loss or objective function
$\mathcal{T}$	Task
$D$	Domain

# List of Publications

---

The thesis mainly discusses the following published articles:

- **Congcong Wang**, and David Lillis. ISA: Iterative Self-controlled Augmentation for Few Shot Text Classification. *Experiments finished and to be submitted* 2023.
- **Congcong Wang**, Gonzalo Fiz Pontiveros, Steven Derby and Tri Kurniawan Wijaya. STA: Self-controlled Text Augmentation for Improving Text Classification. *Submitted and under review*, 2023.
- **Congcong Wang**, Paul Nulty, and David Lillis. Using Pseudo-Labelled Data for Zero-Shot Text Classification. In Proceedings of the *27th International Conference on Natural Language & Information Systems (NLDB 2022)*, Valencia, Spain, June 2022.
- **Congcong Wang** and David Lillis. UCD-CS at TREC 2021 Incident Streams Track. In *Proceedings of the Thirty Text REtreival Conference (TREC 2021)*, Gaithersburg, MD, USA, 2022.
- **Congcong Wang**, Paul Nulty, and David Lillis. Crisis Domain Adaptation Using Sequence-to-Sequence Transformers. In *Proceedings of 18th International Conference on Information Systems for Crisis Response and Management, pages 655–666 (ISCRAM 2021)*, Blacksburg, VA (USA), Virginia Tech, 2021.
- **Congcong Wang**, Paul Nulty, and David Lillis. Transformer-based Multi-task Learning for Disaster Tweet Categorisation. In *Proceedings of 18th International Conference on Information Systems for Crisis Response and Management, pages 705–718 (ISCRAM 2021)*, Blacksburg, VA (USA), Virginia Tech, 2021.
- **Congcong Wang** and David Lillis. Multi-task transfer learning for finding actionable information from crisis-related messages on social media. In *Proceedings of the Twenty-Ninth Text REtreival Conference (TREC 2020)*, Gaithersburg, MD, USA, 2021.
- **Congcong Wang** and David Lillis. Classification for Crisis-Related Tweets Leveraging Word Embeddings and Data Augmentation. In *Proceedings of the Twenty-Eighth Text REtreival Conference (TREC 2019)*, Gaithersburg, MD, USA, 2020.

The following articles are related, but will not be thoroughly discussed in this thesis:

- **Congcong Wang** and David Lillis. A Comparative Study on Word Embeddings in Deep Learning for Text Classification. In *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval (NLPiR 2020)*, Seoul, South Korea, Dec. 2020.
- **Congcong Wang** and David Lillis. UCD-CS at W-NUT 2020 Shared Task-3: A Text to Text Approach for COVID-19 Event Extraction on Social Media. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 514–521, Online, Nov. 2020. Association for Computational Linguistics.

# Abstract

---

During crisis situations, social media allows people to quickly share information, including messages requesting help. This can be valuable to emergency responders, who need to categorise and prioritise these messages based on the type of assistance being requested. However, the high volume of messages makes it difficult to filter and prioritise them without the use of computational techniques. Fully supervised filtering techniques for crisis message categorisation typically require a large amount of annotated training data, but this can be difficult to obtain during an ongoing crisis and is expensive in terms of time and labour to create.

This thesis focuses on addressing the challenge of low data availability when categorising crisis messages for emergency response. It first presents domain adaptation as a solution for this problem, which involves learning a categorisation model from annotated data from past crisis events (source domain) and adapting it to categorise messages from an ongoing crisis event (target domain). In many-to-many adaptation, where the model is trained on multiple past events and adapted to multiple ongoing events, a multi-task learning approach is proposed using pre-trained language models. This approach outperforms baselines and an ensemble approach further improves performance. In one-to-one or many-to-one adaptation, this research studies which combination of past events to include in the model to achieve the best adaptation performance for a particular target event. An approach using sequence-to-sequence pre-trained language models is proposed that incorporates event information for crisis message categorisation, and it is found to outperform existing state-of-the-art methods. The study also finds that using past events that are more similar to the target event tends to lead to better adaptation performance, while using dissimilar events does not improve performance.

However, crisis domain adaptation is only effective when the categorisation task is the same for both the source and target event and there is sufficient annotated data available from the source event. To address the situation where there is very limited labelled data is available relating to the target event, the research presents a self-controlled augmentation approach and an optimised iterative self-controlled augmentation approach to generate additional crisis data for model training. These approaches are able to generate high quality crisis data, leading to better classification performance compared to other methods in the few-shot learning scenario. Additionally, the research presents

a method for training a categorisation model in a zero-shot setting, where there is no time to annotate any data for the new event. This involves matching label names with the unlabelled data of the target event and creating a pseudo-labelled dataset with high confidence for model training. The results show that this approach is able to effectively pseudo-label the unlabelled data, resulting in better performance compared to other zero-shot methods. The proposed few-shot and zero-shot approaches are also tested in other domains such as emotion and topic classification, and demonstrate superior generalisation performance compared to baselines in these domains.

This thesis contributes to the crisis informatics research by coping with low annotated data availability of emerging events for crisis message categorisation on social media. The approaches presented in the thesis are developed in close association with real-world situations and show top performance in experiments. The approaches have the potential to be used in practice for timely and effective humanitarian aid response.

# Chapter 1: Introduction

---

## 1.1 Research context

The widespread use of social media and the abundance of user-generated content in recent years has made it a valuable tool for connecting people. In the context of emergency situations, social media has gained attention as a means of communication and coordination for emergency response agencies and humanitarian organisations, which aim to provide timely and efficient responses to time-sensitive events such as natural disasters or human-induced hazards [33].

According to [39], natural disasters result in an average of 50,000 deaths worldwide each year. Delays in obtaining critical information during an emergency can not only cause additional property damage but also put lives at risk. Therefore, the timely delivery of emergency response is essential in minimising the impact of such events. Social media's ability to provide real-time communication between those affected by an incident and emergency aid centers can help improve situational awareness [31,134,135], which refers to the effective and accurate understanding of how an incident is unfolding, enabling response services to take timely preventive measures to address the situation. To be specific, emergency services can use social media in several ways to improve situational awareness, for example by monitoring social media platforms for messages and posts about the crisis to gain real-time information about the situation on the ground, or by analysing the sentiment and emotional content of social media posts to understand the overall impact of the crisis on the community, or by capturing location information associated with social media posts to identify and respond to specific areas in need of assistance.

According to a study, 69% of people believe that emergency response operators should monitor their social media accounts and respond promptly during a crisis [100]. Another study found that about 10% of emergency-related posts on Twitter (called “tweets”) are important and about 1% are critical [86], highlighting the potential of social media for emergency response. Traditionally, tracking emergencies on social media has been done manually, with human workers filtering and classifying messages as critical or not [63,103]. While this approach can achieve high precision, it is very labour-intensive and costly in terms of time, especially during crises when the number

of relevant messages often increases rapidly. This has motivated the development of computational techniques for emergency tracking and response. [49,86,87,101,102,159].

Taking Twitter as an example<sup>1</sup>, this research introduces the workflow of automatic crisis message categorisation for emergency response with two phases, namely initial filtering and further classification, as presented in Figure 1.1. It begins with a “post stream” of messages describing an ongoing crisis. These tend to be noisy and numerous, and are initially filtered by monitoring hashtags or keywords related to the crisis. This initial filtering aims to find those that are informative to users aid needs (i.e. what assistance the users are trying to seek from emergency responders). However,

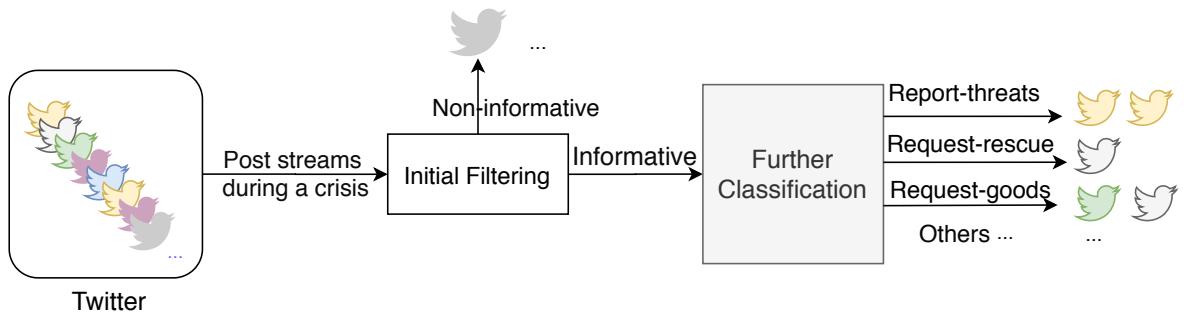


Figure 1.1: The workflow of automatic crisis message categorisation for emergency response on social media.

only finding the informative messages is insufficient for efficient emergency response because the categories of aid needs have different priorities (e.g. search and rescue is more critical than weather and location). This motivates the next phase: further classification is needed to identify these categories. The categories refer to information types [85] that represent different user needs during a crisis. For example, a message could be requesting rescue (for oneself or others) or asking for donations of goods or money. Based on the definition, each informative message after the initial filtering is further classified into one or more of the information types and finally pushed to corresponding emergency operators in accordance with the requirements of their roles.

The problem of social media crisis message categorisation falls into the area of text classification, which is a form of natural language processing (NLP). Text classification has been widely studied from traditional machine learning approaches [121] such as Naïve Bayes to more recent transfer learning methods using pre-trained language models such as BERT and T5 [24, 112, 117, 132]. However, most of these methods are based on fully-supervised learning, which requires a large amount of annotated data for model building. For example, to create a model that can classify the topic of a new piece of news, a dataset of news articles that have been labelled with their topics are needed to train the model. In the domain of crisis response, it is often difficult to

---

<sup>1</sup>Throughout this thesis, unless stated otherwise, Twitter is the default social media platform being discussed due to its high popularity and timeliness.

use fully-supervised learning techniques because annotated data of emerging events is not readily available. This is because crises happen suddenly and it can be costly and time-consuming to manually label a dataset of crisis-related social media messages. As a result, there is a need for methods that can categorise crisis messages effectively even when there is limited labelled data available. This research aims to address this problem by developing methods for categorising social media crisis messages.

## 1.2 Research approaches

As mentioned above, the difficult and time-consuming nature of annotating data from ongoing crisis events makes it challenging to create categorisation models for emergency response using fully-supervised approaches. This research addresses this challenge by proposing three scenarios that explore different sources of data that is feasible to use for model development.

The first scenario, called “crisis domain adaptation”, involves using annotated data from past crisis events (source events) to create a categorisation model that is capable of operating when no annotated data from ongoing events (target events) is available. This is feasible because although it still requires time and human labour, annotating data for past crises is easier and less pressing than annotating data for ongoing events and in most cases, new events have strong similarities to some past event or events.

The second scenario, called “crisis few-shot learning”, involves developing the model using a small amount of labelled data from a target event. This is feasible because it requires little effort to annotate a small amount of data for ongoing crises.

The third scenario, called “crisis zero-shot learning”, involves creating the model using only unlabelled data from a target event, which is easy to obtain as the event unfolds. Figure 1.2 presents the three research scenarios that are discussed in the following text.

Crisis domain adaptation involves adapting a model trained on labelled data from source events (source domain) to classify messages from target events (target domain) that were not seen during training. This raises two research challenges. The first, referred to as many-to-many adaptation, arises when responders want to deploy a categorisation model to target events without knowing the details of the crisis events. It involves training the model on the source domain of multiple events and adapting it to the target domain of any event. Additionally, it is also of interest to determine the best way to select suitable past events to be the source domain and adapt the model to a target event when there is basic knowledge about the events, such as the type

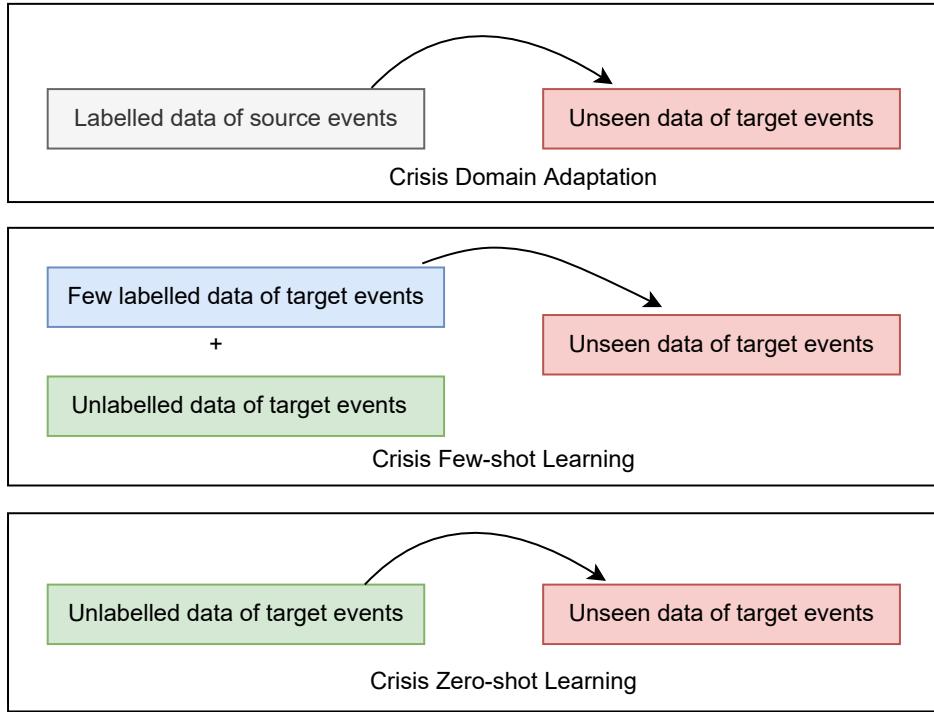


Figure 1.2: Three research scenarios of this research: crisis domain adaptation, crisis few-shot learning and crisis zero-shot learning.

(flood or fire) and location. This is known as many/one-to-one adaptation, and is a more targeted adaptation to a specific target event than many-to-many adaptation. In this scenario, the goal of the research is to create models that are based on pre-trained language models in order to improve performance in both types of adaptation.

In emergency response, responders take appropriate action based on the categorisation of crisis messages into specific types of aid needs (information types). In crisis domain adaptation, messages from ongoing events are categorised into information types by a model that is trained on annotated data from past events with the same set of information types. However, in reality, responders may need to look for new information types for emerging events. For example, in a fire event, responders may ask for information about “firefighting”, but this type of information may not have been included in the annotations for past events such as terrorist attacks and floods. This makes it difficult to adapt the model trained on past events, which do not include the new information types, to a target event with the new information types. This motivates the research to go beyond crisis domain adaptation.

The crisis few-shot learning scenario poses the question of how to build a categorisation model that can operate with only a limited amount of labelled target data, but that may have access to larger quantities of unlabelled data relating to the target event. Since it is not practical to annotate a large volume of target data for a new event, a small amount of annotated training data can be easily collected at the start of the new

event. The crisis few-shot learning approach uses this small labelled dataset to build the model, making it a more feasible solution with regard to annotation efforts. As the crisis event progresses, a growing quantity of unlabelled data from the event becomes available and can be used to further refine the model. In this scenario, the goal of the research is to improve the performance of a categorisation model by using pre-trained language models to augment the small labelled dataset, then fine-tuning the model with unlabelled target data.

The crisis zero-shot learning scenario poses the question of how to classify unseen crisis messages using only unlabelled target data, which is inexpensive to obtain. This is challenging because the model must operate without any reliance on supervision resources in the domain to map an information type to an unseen crisis message. Despite the difficulty, this approach has the advantage of not requiring any labelled data to build the categorisation model, making it the most efficient solution in terms of annotation efforts. The goal of this research in this scenario is to use information types and a corpus of unlabelled target data to improve the performance of crisis message categorisation.

## 1.3 Research questions

The following are the primary research questions that this thesis aims to address based on the presented research scenarios:

- Regarding crisis domain adaptation, how can pre-trained language models be utilised to enhance the performance of crisis messages categorisation? For many-to-many adaptation, what are the machine learning models and pre-trained language models that can improve the categorisation performance? For many/one-to-one adaptation, what factors should be considered when selecting past events to serve as the source domain and adapting the model to a target event?
- In crisis few-shot learning, what techniques can be used to identify new information types that may be required for crisis messages categorisation in emerging events? How can unlabelled target data be used to improve the performance of categorisation in crisis few-shot learning settings?
- In crisis zero-shot learning settings, how can information types and a corpus of unlabelled target data be used to improve the performance of crisis message categorisation?

- Last, what are the limitations of using each of the proposed scenarios for building categorisation models in crisis response, and how can these limitations be addressed?

## 1.4 Research contributions

Based on the aforementioned research scenarios and questions, the contributions of this thesis are summarised as follows.

- In practice, it is important to address the challenge of low data availability for categorising social media crisis messages during emerging events. To study this issue systematically, in this thesis, three research scenarios are proposed based on the level of data availability: crisis domain adaptation, crisis few-shot learning, and crisis zero-shot learning. Each of these scenarios plays a crucial role in effective and timely emergency response.
- In the crisis domain adaptation scenario, a multi-task learning (MTL) approach using pre-trained language models like BERT is proposed for many-to-many crisis domain adaptation. The results indicate that the MTL method performs better than state of the art baseline methods, and an ensemble approach based on the MTL approach further improves performance.
- For many/one-to-one adaptation, a method called CAST is proposed that uses sequence-to-sequence pre-trained language models like T5 incorporating event information for crisis message categorisation. Comparison with existing state-of-the-art crisis domain adaptation approaches shows that CAST, which has the least dependence on target data, also outperforms these methods in terms of effectiveness. In addition, the combination of similar events is found to be more effective for adaptation compared to the addition of dissimilar events to the training set.
- For crisis few-shot learning, a self-controlled augmentation (STA) approach is proposed to generate training crisis messages using a few labelled seed messages for crisis messages categorisation. When tested in low data regimes, STA outperforms state-of-the-art augmentation approaches. The generated messages produced by STA demonstrate better quality compared to state-of-the-art approaches. To further improve STA, an iterative mechanism and a de-duplication mechanism (referred to as ISA) are added to the STA pipeline. ISA is very effective in improving the categorisation performance when tested in few-shot settings

and its performance can be further improved by leveraging a target unlabelled to refine the categorisation model. STA and ISA not only show state-of-the-art performance in the crisis domain, but also exhibit strong performance on other text classification domains such as emotion or news topic classification.

- For crisis zero-shot learning, an approach called P-ZSC is proposed that pseudo-labels an unlabelled corpus for model training. P-ZSC not only performs better than other crisis-related approaches, but also has better generalisation to non-crisis-related text classification tasks. When the pseudo-labelled data produced by P-ZSC is examined, it is found to be as effective as a supervised approach using hundreds of manually annotated samples in both crisis categorisation and non-crisis classification tasks, suggesting the potential to save significant time and effort in training a model for these tasks.
- While the proposed methods still have some distance to cover before they can match the performance of fully-supervised approaches with abundant training data, these latter approaches serve as an interesting benchmark, representing the upper limit of what can currently be achieved under ideal circumstances. However, the proposed methods do outperform their corresponding baselines, making them a significant improvement over current state-of-the-art adaptation, zero, and few-shot approaches, and bringing the community closer to closing the gap between these approaches and their fully-supervised counterparts.

## 1.5 Thesis outline

The outline of this thesis is summarised as follows.

- **Chapter 2: Background.** As crisis message categorisation falls under the umbrella of text classification, this chapter provides a general background review of text classification. It begins by introducing the basics of machine learning and relevant methods, followed by a discussion of deep neural network methods such as word embeddings and pre-trained language models. These methods are essential for building computational models for text classification.
- **Chapter 3: Crisis Message Categorisation in Low-data Settings.** This chapter focuses on a review of the literature on crisis message categorisation in low-data settings. It covers existing methods in the literature for crisis domain adaptation, few-shot learning, and zero-shot learning. While some of these

methods were initially developed for general text classification, they can be easily adapted for use in the crisis domain, making them the major works used as baselines in this research. The chapter also reviews relevant datasets and evaluation metrics used in the research.

- **Chapter 4: Many-to-many Crisis Domain Adaptation.** This chapter presents various methods for adapting to crisis domain data in a many-to-many setting. It begins by introducing single-task methods and then moves on to multi-task methods that utilise pre-trained language models. The chapter provides detailed explanations of each method and offers insights into how machine learning and deep learning techniques can be applied to achieve many-to-many adaptation.
- **Chapter 5: Many-to-one and One-to-one Crisis Domain Adaptations.** This chapter introduces CAST, a method that uses sequence-to-sequence models and incorporates event information for many-to-one and one-to-one domain adaptations. The details of the methods, the experimental setup for testing its effectiveness, and key insights are included in this chapter.
- **Chapter 6: Augmentation for Crisis Few-shot Learning.** This chapter provides details on the self-controlled (STA) and iterative self-controlled (ISA) augmentation techniques for crisis few-shot learning. It compares these techniques to existing baselines, including both augmentation and other few-shot approaches in low data regimes. The chapter also includes ablation studies and an examination of the generated data produced by the methods and baselines.
- **Chapter 7: Using Pseudo-labelled Data For Crisis Zero-shot Learning.** This chapter introduces the P-ZSC method for crisis zero-shot learning, which involves the use of pseudo-labelled data. The chapter discusses the process of label expansion, pseudo label assignment and refinement on unlabelled target data. In addition to categorisation results, the chapter includes the results of an ablation study and a study of the quality of the pseudo-labelled data.
- **Chapter 8: Conclusion.** This chapter summarises the achievements of this research, outlines the major contributions of the study, and discusses future directions for addressing the challenge of low data availability in the categorisation of social media crisis messages.

# Chapter 2: Background

---

This chapter presents the background knowledge that the subsequent chapters will build upon. As the core task in this research is to apply machine learning methods for social media crisis messages categorisation, this chapter focuses on the background of machine learning for text classification in the area of natural language processing (NLP). Regarding the content structure, the basic elements of machine learning are reviewed and linear regression (a preliminary form of neural networks) for supervised text classification is introduced as the first part of this chapter (Section 2.1). Neural Networks are a specific category of machine learning models that have been widely and successfully applied to text classification. They are introduced as the second part of this chapter, including convolutional neural networks, recurrent networks and pre-trained language models (Section 2.2).

## 2.1 Machine learning

This section first reviews key concepts of machine learning at a high level and then introduces a specific example of machine learning approaches—linear regression that can be regarded as a simple neural network. In the subsection of linear regression, the process of building a logistic regression model for text classification is detailed.

### 2.1.1 Key concepts

According to the definition by Mitchell (1997) [93], machine learning is viewed as: “A computer program is said to learn from experience  $E$  with respect to some class of tasks  $T$  and performance measure  $P$ , if its performance at tasks in  $T$ , as measured by  $P$ , improves with experience  $E$ .” From this definition therefore, there are major three concepts regarding machine learning: the task  $T$ , the performance  $P$  and the experience  $E$ .

**Text classification as the task:** The tasks of machine learning ( $T$ ) are various. It can be a task of enabling a robot to play a game, a task of finding human faces

in a picture or a task of identifying emotion in human speech. For example, in this research, the task is to classify aid types for user-posted messages during crises on social media. In formal terms, the crisis message is described as the *example*, *instance*, *sample* or *input* that is processed by the machine learning system to perform the task. To enable the machine to understand the example, it is represented as a vector  $\mathbf{x} \in \mathbb{R}^k$  of  $k$  features, where each feature  $x_i$  refers to the value for a particular attribute of the example. For instance, at its simplest, the features can be occurrence counts of words in the crisis message. This thesis concentrates on crisis message categorisation, which can be considered to be a text classification task. The following provides a high-level description of how a machine learning system is applied for text classification. Basically, in text classification, the machine program is asked to identify which of  $m$  categories the example  $\mathbf{x}$  belongs to. The  $m$  categories are known as the *classes* or *labels* comprising the *label space*  $Y$  that are defined as the *output* of the machine program. Depending on  $Y$ , text classification exists in multiple forms. If the number of classes  $m$  is 2, it is called as a *binary classification* task. If  $m > 2$ , it is defined as a *multi-class classification* task. In cases where multiple classes may be identified for the example, it is referred to as a *multi-label classification* task. To achieve the classification, the learning algorithm needs to produce a *hypothesis* function that maps the input to the output, i.e.,  $y = h(\mathbf{x})$  where  $y \in Y$ . One typical type of mapping function is to estimate the probability distribution over the classes given the input i.e.,  $h(\mathbf{x}) = p(y|\mathbf{x})$ . To categorise the example, the classes with high probabilities are usually assigned.

**Performance evaluation in text classification:** When the machine learning system is carrying out on a task, it is important to measure how well the system performs so that its performance ( $P$ ) can be compared to another system. To measure its performance, some evaluation metrics are required. In the context of text classification, the most straightforward metric to evaluate the performance is *accuracy*. It is obtained by the number of true predictions divided by the number of all predictions by the system, which is formulated as follows:

$$\text{Acc} = \frac{1}{n} \sum_i^n I(y_i = \hat{y}_i) \quad (2.1)$$

where  $n$  stands for the number of all predictions and  $I$  is the indicator function that is 1 if the actual label  $y_i$  is matched with the predicted label  $\hat{y}_i$  and 0 otherwise. In some cases, as an alternative to accuracy, the *error rate* is used to measure the performance, which is calculated as the proportion of incorrect predictions out of all predictions. For multi-class classification, another metric *F1 score* is usually used to measure the performance on a per class basis, which is the harmonic mean of *precision* and *recall*,

defined as follows.

$$\text{Precision} = \frac{TP}{TP + FP}, \text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{TP}{TP + \frac{1}{2}(TP + FN)}$$
(2.2)

where  $TP$ ,  $FP$  and  $FN$  stands for the number of true positives, false positives and false negatives respectively. At a per-class level, an example is said to be a true positive when it is classified to the class that it actually belongs to, a false positive when it is classified to the class that it does not belong to and a false negative when it is not classified to the class that it actually belongs to. Hence, it is easy to know from the equation that precision measures the proportion of true positives out of all positive predictions and recall measures the proportion of true positives out of actual positive examples of the class. To consider both precision and recall, the F1 score is used as the harmonic mean of precision and recall for a more balanced summarisation of the system's performance. Since the aforementioned F1 is calculated on a per class basis, some averaging methods are normally used to get a global performance evaluation for all classes. There are three commonly-used averaging methods for this purpose: *micro-F1*, *macro-F1* and *weighted-F1*. The micro-F1 metric computes a global average F1 score by counting the sums of the TPs, FNs, and FPs, formulated as follows:

$$\text{micro-F1} = \frac{\sum_{i=0}^m TP_i}{\sum_{i=0}^m TP_i + \frac{1}{2}(\sum_{i=0}^m TP_i + \sum_{i=0}^m FN_i)}$$
(2.3)

Where  $m$  is the number of classes. The macro-F1 metric is straightforward, calculated by taking the mean of all the per-class F1 scores.

$$\text{macro-F1} = \frac{1}{m} \sum_{i=0}^m F1_i$$
(2.4)

As seen from the equation, the macro-F1 metric simply averages the per-class F1 scores without taking into account the number of examples of each class (also known as *support*). The weighted-F1 is a variant of the macro-F1, and it is computed by taking the mean of all per-class F1 scores while considering each class's support.

$$\text{weighted-F1} = \sum_{i=0}^m r_i F1_i$$
(2.5)

where  $r_i$  refers to the ratio of examples of the  $i$ th class out of all examples. Regarding the choice of the averaging methods, it is dependent on the characteristics of the examples that are classified by the system. If the examples are imbalanced across all classes, the micro average is a straightforward choice for reporting the overall performance of the system regardless of the class. If the examples tend to be balanced across all classes and the classes are equally important, the macro average would be a good choice as it treats all classes equally. In cases where the examples are imbalanced and the classes with more examples need to be paid more importance, the weighted averaging is preferred.

As described above, for a machine learning system, it is able to perform a specific task such as assigning classes to textual input examples and its performance on the task can be measured by the evaluation metrics. To enable the system perform the task, it has to learn from the experience  $E$ .

**Learning from the experience:** In the learning process of a machine program, the experience normally refers to a dataset consisting of the examples or data points that the program can learn from. For examples, in the crisis messages categorisation task, the experience indicates the dataset containing crisis messages. Depending on whether the data is annotated or not, machine learning can be broadly divided into two categories: *supervised learning* and *unsupervised learning* [38]. The former says that every example in the dataset is assigned with one or more labels that are used to teach the program what to output given the example. It is termed “supervised” since the labels are annotated by humans (i.e., human supervision for teaching the program to learn from the examples). In contrast, in unsupervised learning, the learning algorithm only sees the examples without any labels being designated and the goal is to identify patterns within the dataset containing the data points on its own, such as clustering users into groups of similar interest in a recommendation system. It is termed “unsupervised” since there is no external guidance for the program to perform that task. Situated between supervised learning and unsupervised learning, *semi-supervised learning* is also widely studied in the literature of machine learning where the learning algorithm is provided with a combination of labelled data and unlabelled data to perform a task.

As discussed in Section 1.2, this research proposes three scenarios to solve the low data problem for crisis messages categorisation. Although the focus of this research is on settings where there is a lack of available annotated data, the proposed methods are in essence based on supervised learning. For crisis domain adaptation, the algorithm is learnt on the labelled data of source events and then adapted to a new event. In crisis few-shot learning, the proposed methods apply augmentation techniques to obtain more labelled examples based on a small quantity of originally-provided labelled examples. Hence, the algorithm is learnt on the augmented data that is labelled. Al-

though initially there is no labelled data for crisis zero-shot learning, this research proposes a method that uses label names with the unlabelled data to obtain a pseudo-labelled dataset on which the algorithm is learnt. Overall, the idea is to convert a semi-supervised or unsupervised task into a situation where supervised approaches can be used. As such, the remaining background review will be centered around supervised machine learning algorithms. Machine learning has developed with various algorithms proposed in the literature, from traditional machine learning algorithms such as linear regression or naïve bayes to neural networks algorithms such as pre-trained language models. The proposed methods in this research are mainly built upon the neural networks algorithms due to their strong performance in text classification. Hence, the focus of the discussion of neural network models is on their use within text classification. As the cornerstone of neural networks, the introduction to linear regression comes first as follows.

## 2.1.2 Linear regression

This section presents a linear model in the context of regression tasks, and then demonstrates how a similar approach can be applied to classification tasks. To better describe the process of applying linear regression for text classification, an entire dataset is represented by  $\mathbf{X} \in \mathbb{R}^{n \times k}$  containing  $n$  examples (data points) of  $k$  features where  $\mathbf{x}_i$  stands for the input features of the  $i$ -th example in the dataset. In supervised learning, the input  $\mathbf{x}_i$  is associated with a label  $y_i \in Y$ . For a machine learning algorithm in text classification, the objective is to build a model that predicts the label  $y$  given the input  $\mathbf{x}$  through an estimation of conditional probability distribution using a set of *parameters* or *weights*  $\theta$ , denoted as  $p(y|\mathbf{x}; \theta)$ . The common way to achieve the estimation is known as *maximum likelihood estimation* (MLE) where the optimal parameters  $\hat{\theta}$  are obtained by maximising the likelihood of the data  $p(y|\mathbf{X}; \theta)$

$$\hat{\theta} = \arg \max_{\theta} p(y|\mathbf{X}; \theta) \quad (2.6)$$

which can be transformed into:

$$\hat{\theta} = \arg \max_{\theta} \sum_{i=1}^n \log p(y|\mathbf{x}_i; \theta) \quad (2.7)$$

For a linear regression model, the objective is to predict a numeric value  $\hat{y} \in \mathbb{R}$  given the input features  $\mathbf{x} \in \mathbb{R}^k$  using a vector  $\theta \in \mathbb{R}^k$  of parameters and a *bias*  $b \in \mathbb{R}$ ,

formulated as follows.

$$\hat{y} = \theta^\top \mathbf{x} + b \quad (2.8)$$

where  $\hat{y}$  is the predicted value of  $y$ . To learn the parameters  $\theta$  in regression tasks, the mean square error (MSE) function is commonly used to measure the difference between the model's prediction  $\hat{y}$  and the true value  $y$ , formulated as follows [14].

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad (2.9)$$

The *error* function is also known as the *objective* or *cost* function. That says, to find the optimal  $\hat{\theta}$ , the objective is to minimise the error or cost.

$$\hat{\theta} = \arg \min_{\theta} \frac{1}{n} \sum_{i=1}^n (\theta^\top \mathbf{x}_i + b - y_i)^2 \quad (2.10)$$

**Logistic regression:** The aforementioned describes linear regression in the regression problem where the output is a numeric value. It can be easily adapted to classification tasks by converting the predicted numeric value to a value between 0 and 1 implying the conditional probability of  $y$  given  $\mathbf{x}$ . Linear regression becomes *logistic regression* when it is applied for classification in this way. The conversion function is known as the *sigmoid* or *logistic* function denoted as  $\sigma$ , which is defined as follow.

$$\sigma(x) = \frac{1}{1 + e^{-x}} \quad (2.11)$$

The sigmoid function takes an input  $x$  and converts it into a value between 0 and 1. The converted value is close to 1 when the input goes for positive infinity and 0 when it approaches negative infinity. In binary classification, there are two classes: class 0 and class 1. Logistic regression models the conditional probability  $\hat{p}(y|\mathbf{x}; \theta)$  as follows:

$$\begin{aligned} \hat{p}(y = 1|\mathbf{x}; \theta) &= \sigma(\theta^\top \mathbf{x}) = \frac{1}{1 + e^{-\theta^\top \mathbf{x}}} \\ \hat{p}(y = 0|\mathbf{x}; \theta) &= 1 - \hat{p}(y = 1|\mathbf{x}; \theta) \end{aligned} \quad (2.12)$$

which can be rewritten as:

$$\hat{p}(y|\mathbf{x}; \theta) = (\hat{y})^y(1 - \hat{y})^{1-y} \quad (2.13)$$

By applying maximum likelihood estimation, the parameters  $\theta$  can be estimated as follows:

$$\begin{aligned}\hat{\theta} &= \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^n \hat{p}(y_i | \mathbf{x}_i; \theta) \\ &= \underset{\theta}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^n -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y}_i)\end{aligned} \quad (2.14)$$

In linear regression, MSE is used as the loss function to find the optimal  $\hat{\theta}$ . Similarly, the last part of Equation 2.14 known as the *binary cross entropy* (BCE) loss [51] is minimised to find the optimal  $\hat{\theta}$ . Once the optimal  $\hat{\theta}$  are obtained, the classifier is finished building and can then be used to predict class 1 or 0 for an example via Equation 2.12. For multi-label multi-class classification involving  $m$  classes, it can be done by building  $m$  such binary classifiers. Basically, for each classifier, a separate set of parameters  $\theta_i$  for the  $i$ -th class are learnt through Equation 2.12 first. Then in predictions, the  $i$ -th binary classifier with the learnt  $\theta_i$  applies Equation 2.12 to decide whether the  $i$ -th class is assigned to an example (assigned if being predicted to be class 1 and not if class 0). For single-label multi-class classification, instead of the sigmoid function in Equation 2.12, the *softmax* function is used to obtain a categorical distribution:

$$\hat{p}(y_i | \mathbf{x}; \theta) = \frac{e^{\theta_i^\top \mathbf{x}}}{\sum_{j=1}^m e^{\theta_j^\top \mathbf{x}}} \quad (2.15)$$

The softmax function normalises the model output for  $i$ -th class to be a value between 0 and 1, indicating the conditional probability of the  $i$ -th class given the example  $\mathbf{x}$ . Given the example  $\mathbf{x}$  over all classes, the *categorical cross-entropy* function (CCE) [51] is used as the loss function to calculate the difference between the empirical conditional probability  $p(y|\mathbf{x})$  and the model predicted probability  $\hat{p}(y|\mathbf{x})$ , defined as follows:

$$J(\theta) = - \sum_{i=1}^m p(y_i | \mathbf{x}) \log \hat{p}(y_i | \mathbf{x}; \theta) \quad (2.16)$$

The optimal  $\theta$  can be found by minimising the loss/objective function on all the  $n$  examples, formulated as follows:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^m -p(y_j | \mathbf{x}_i) \log \hat{p}(y_j | \mathbf{x}_i; \theta) \quad (2.17)$$

In describing logistic regression for classification, one question remained unanswered is how to update the parameters  $\theta$  so they can be the optimal. Denoting the objective function as  $J(\theta)$ , one common optimisation procedure is *gradient descent* that updates the parameters  $\theta$  in the opposite direction of the gradients  $\nabla_{\theta} J(\theta)$  with a *learning rate*  $\alpha$  in *mini-batches*. For details on gradient descent and more optimisation procedures such as Adam [59] and Adagrad [28], an overview of these procedures can be found in [116].

**Training and generalisation:** So far, the process of how to apply logistic regression for classification has been described. It is important to note that the aforementioned entire dataset  $\mathbf{X}$  refers to the *observed data* (also known as *training data*) for building the classifier. In machine learning, the process of learning the parameters  $\theta$  for the classifier using the observed data is called as the *training* stage. When applying the classifier to make predictions for the unobserved test data, it is called as the *testing* or *inference* stage. In practice for building a machine learning model, the dataset  $\mathbf{X}$  is usually split into two subsets: training set and development/validation set. They are drawn from  $\mathbf{X}$  and hence they are assumed to have the same distribution. The training set is used as the seen data to learn the model's parameters  $\theta$ , the development set is treated as the unseen data to develop the model in choosing hyper-parameters such as the learning rate or batch size and reflecting the model's generalisation capability. The model's error on the training set is called *training error* and on the validation set is called *test* or *generalisation error*. For a machine learning model, the objective is to make the training error small and the gap between training and test error small. A high training error indicates that the model is *underfitting* on the data or the model has a high *bias* and a big gap between training and test error implies the model is *overfitting* on the data or the model has a high *variance* (poor generalisation capability). To prevent underfitting, common solutions include increasing the training data or increasing the capacity of the model (its ability to fit on various functions). To prevent overfitting, the solution is to decrease the capacity of the model. One common way is through *regularisation*, such as  $l_1$  and  $l_2$  norm (check [38] for details).

## 2.2 Neural networks

To describe neural networks, Figure 2.1 presents an example of a basic neural network for classification. Recalling linear regression, for input  $\mathbf{x}$ , the output  $\hat{y}$  is estimated by a linear transformation function using the parameters  $\theta$  and bias  $b$  (Equation 2.7). When it is adapted to logistic regression for classification, the conditional probability  $p(y|\mathbf{x})$  is estimated by applying the sigmoid function or the softmax function on the output of linear regression (Equation 2.10 and 2.15). Logistic regression can be viewed as a simple neural network with one *layer* where the sigmoid or softmax function is known as the *activation function*.

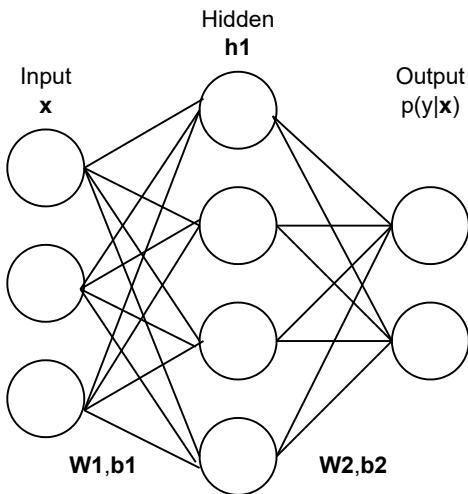


Figure 2.1: A basic neural network for classification

For a neural network, a layer refers to a function that maps an input to an output with learnable parameters  $\mathbf{W}$ <sup>1</sup> and bias  $\mathbf{b}$ . To increase the capacity of a neural network model, there are usually multiple layers stacked together where the output of a previous layer is used as the input of the next layer. To interpret the term “neural”, each layer consists of many units used to process the information from the previous layer and forward it to the next layer, which acts similarly to a biological neuron. As the number of layers increases to a big number, the network is said to be deep in its architecture and in this case neural networks are deep neural networks.

The number of layers is defined as the *depth* of a neural network. For example, Figure 2.1 presents a neural network of three layers. The first layer refers to the *input layer*, the middle layer is the *hidden layer*, and last layer is the *output layer*. In the literature, a neural network of this form is known as the multilayer perceptron network (MLP). In mathematics, it can be described as follows:

---

<sup>1</sup>The  $\mathbf{W}$  notation is used here as an alternative for  $\theta$  used in linear regression.

$$\begin{aligned}\mathbf{h}_1 &= g_1(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) \\ p(y|\mathbf{x}) &= g_2(\mathbf{W}_2 \mathbf{h}_1 + \mathbf{b}_2)\end{aligned}\tag{2.18}$$

In this basic neural network, the input  $\mathbf{x}$  is fed forward to next layer to obtain a hidden representation  $\mathbf{h}_1$  that is then fed forward to next layer to obtain the output  $p(y|\mathbf{x})$ , which is known as *forward propagation*. In the equation,  $g_1$  and  $g_2$  refer to the activation function of the hidden layer and output layer respectively. The activation function is normally a non-linear function to apply a non-linear transformation on the output of that layer. Commonly-used activation functions of the hidden layer include *Rectified Linear Unit* (ReLU) and the *tanh* function, which are defined as follows.

$$\begin{aligned}\text{ReLU}(x) &= \max(0, x) \\ \tanh(x) &= \frac{e^x - e^{-x}}{e^x + e^{-x}}\end{aligned}\tag{2.19}$$

Regarding the activation function of the output layer for classification, it is usually a sigmoid function (binary or multi-label) or a softmax function (multi-class). In a neural network, the output of last layer before applying the activation function is known as the *logits* output.

In Equation 2.19,  $\mathbf{W}s$  refer to the learnable parameters or weights and  $\mathbf{b}s$  are the bias vectors of the neural network model used to estimate the final conditional probability output  $p(y|\mathbf{x})$ . In this example,  $\mathbf{x} \in \mathbb{R}^n$  where  $n = 3$ , there are four nodes in the middle hidden layer and the output is binary, hence  $\mathbf{W}_1 \in \mathbb{R}^{3 \times 4}$ ,  $\mathbf{b}_1 \in \mathbb{R}^4$  and  $\mathbf{W}_2 \in \mathbb{R}^{4 \times 2}$ ,  $\mathbf{b}_2 \in \mathbb{R}^2$ . Similar to linear regression, to find the optimal weights  $\mathbf{W}s$  and  $\mathbf{b}s$ , an objective function is first defined and then the weights are updated through gradient descent on the objective function. In regression, MSE is usually used as the objective function. In classification, binary cross entropy is used as the objective function (binary or multi-label) and categorical cross entropy is used as the objective function (multi-class). Like linear regression, the optimisation is done by a process of gradient descent, which takes derivatives of the objective function with respect to the weights for changing the weights. This is usually referred to as *back-propagation* in the literature [116].

**Neural networks in text classification:** When applying neural networks for text classification, in order to enable the model to understand texts, a piece of text has to be represented numerically. To achieve this, each word in the input text is represented by a vector. The vector representations for each word are known as *word embeddings*, which are used as the initial features of the input text. Word embeddings can be classified into two categories: sparse and dense. A one-hot embedding, which represents a word in a vocabulary with a vector that has a 1 at the position corresponding to the word

and 0s elsewhere, is the most basic example of a sparse embedding. On the other hand, dense word embeddings are typically pre-trained and represent a word with a fixed-length vector of continuous real numbers. Once the individual words of the input text are represented by word embeddings, the representation of the whole text is then learnt based on the initial word features through the neural network models. Figure 2.2 demonstrates the general architecture of neural network models for text classification where a textual input example is viewed as a sequence of tokens  $s : \{t_1, t_2, \dots, t_T\}$  (the tokens usually indicate words or word pieces). To begin, a vocabulary is constructed that includes all the tokens in the training examples, where each token in the vocabulary is associated with an id. The text indexer maps the tokens to their corresponding ids before they are passed to the embedding module. The embedding module then converts the tokens into vectors based on their corresponding ids, denoted by  $\mathbf{x} \in \mathbb{R}^{T \times k}$  where  $k$  implies the embedding dimension. In the case of spare one-hot embedding,  $\mathbf{x} \in \mathbb{R}^k$  where the feature dimension  $k$  is the vocabulary size. However, in neural network models, dense pre-trained word embeddings are commonly used, introduced later in section 2.2.1.

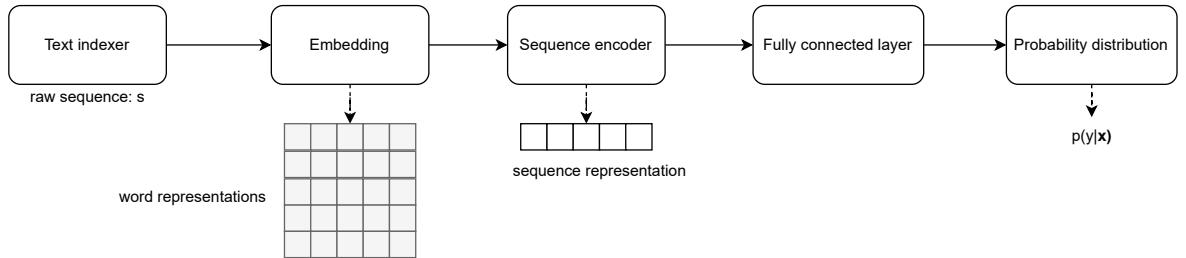


Figure 2.2: The general architecture of neural network models for text classification

After the tokens are embedded, the word representations are fed to the next module which here is called the sequence encoder. Given the classification objective, the sequence encoder aims to learn a good deep representation for the sequence or loosely speaking a good summary of the individual word representations. After being encoded, the sequence representation is forwarded to a fully connected layer (i.e., a linear transformation layer) to output the logits across all classes. This is then converted by the activation function to a probability distribution to capture the probability of the input  $s$  represented by  $\mathbf{x}$  belonging to class  $y$ . In a neural network, the layers can be customised and connected in different ways to learn representations of data for a target problem. This brings many variations of deep models for sequence representation learning. The variations primarily include convolutional neural networks (CNN) and long short-term memory based recurrent neural network (LSTM-RNN) and pre-trained language models. The details of them are covered in Section 2.2.2 2.2.3 and 2.2.4.

## 2.2.1 Pre-trained word embeddings

Pre-trained word embedding is a way to represent a word with fixed-length vectors of continuous real numbers [10, 91, 92, 106]. It maps a word in a vocabulary to a latent vector space where words with similar contexts are in proximity. Through word embedding, a word is converted to a vector that summarises both the word's syntactic and semantic information. Word embeddings are used as input feature representations for neural network models in text classification, which corresponds to the embedding module in Figure 2.2. Figure 2.3 presents a taxonomy of pre-trained word embeddings. Broadly speaking, there are two main types of word embeddings, namely context-independent and context-dependent embeddings. The typical examples of the former are word2vec [92], GloVe [106] and FastText [10]. These are known as classic word embeddings, which learn representations through language model (LM) based shallow neural networks or co-occurrence matrix factorisation [160]. The learned representations are characterised by being distinct for each word without considering the word's context. Hence, they are usually pre-trained and stored in the form of downloadable files which can be directly applied to text classification tasks <sup>2</sup>.

In comparison to context-independent word embeddings, context-dependent methods learn different embeddings for the same word with different contextual use. For example, for the homonym “bank”, its embedding changes depending on whether it is used in a river-related context or finance-related one. This feature has made this type of embeddings to become the mainstream. Examples of context-dependent include ELMo [107], Flair [1], BERT [24], etc. Below gives a short introduction to some specific word embedding approaches.

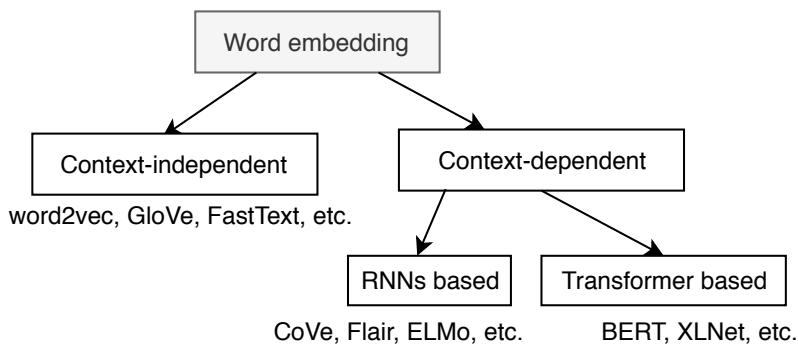


Figure 2.3: A taxonomy of pre-trained word embeddings

**Classic embeddings:** This type of embeddings are pre-trained over very large corpora and shown to capture latent syntactic and semantic features.

- **word2vec** [92]. This applies either of two model architectures to produce word vectors, namely continuous bag-of-words (CBOW) and skip-gram (SG). Both

<sup>2</sup>For example, GloVe can be downloaded from: <https://nlp.stanford.edu/projects/glove/>

methods are trained based on a neural prediction-based model. CBOW trains a model that aims to predict a word given its context, while SG does the inverse, namely to predict the context given its central word.

- **GloVe** [106] learns efficient word representations by performing training on aggregated global word-word co-occurrence statistics from a corpus. It is known for learning good general language features by capturing words’ co-occurrences globally across corpora.
- **FastText** [10] learns word representations through a neural LM. Unlike GloVe, it embeds words by treating each word as being composed of character n-grams instead of a word whole. This allows it to not only learn rare words but also out-of-vocabulary words.

**Contextualised embeddings:** Unlike classic embeddings, this type of embeddings are known for capturing word semantics in context.

- **ELMo** [107] learns contextualised word representations based on a neural LM with a character-based encoding layer and two BiLSTM layers (these are discussed later in Section 2.2.3). The character-based layer encodes a sequence of characters of a word into the word’s representation for the subsequent two BiLSTM layers that leverage hidden states to generate the word’s final embedding.

**Flair** [1]. This trains a model for producing contextualised word embeddings using neural character LM (1-layer BiLSTM), leading to lower computational resources and stronger character-level features. Although its effectiveness has been recognised in sequence labelling tasks, its performance in text classification remains unexplored.

**BERT** [24] is a recently proposed Transformer-based [132] language representation model trained on a large cross-domain corpus. Unlike ELMO, which learns representations through bidirectional LMs (i.e., simply the combination of left-to-right and right-to-left representations), BERT applies a Masked LM to predict words that are randomly masked. Instead of the recurrent learning of a sequence, BERT uses the Multi-Head Cross-Attention mechanism [6, 132] to learn global dependencies between the words of a sequence, leading to it being more parallelisable and exhibiting superior performance (these are discussed later in Section 2.2.4). Through a process of fine-tuning, BERT has been demonstrated to achieve state-of-the-art results for a range of NLP tasks. Other similar embeddings optimised for BERT include DistilBERT [118], RoBERTa [80] and ALBERT [61].

Among so many choices of word embeddings, it is important to know how to select them for text classification. As a preliminary step before undertaking the work that constitutes the main contribution of the thesis, an initial study was conducted to investigate the factors influencing the choice of word embeddings for classification tasks [138]. In this study, the aforementioned word embeddings are used with a CNN and a BiLSTM sequence encoder (see Section 2.2.2 and 2.2.3) for four benchmarking text classification tasks. To summarise the major findings of this work, the results show that CNN as the sequence encoder outperforms BiLSTM in most situations, especially for document context-insensitive datasets. This study recommends choosing CNN over BiLSTM for document classification datasets where the context in sequence is not as indicative of class membership as sentence datasets. For word embeddings, concatenation of multiple classic embeddings or increasing their size does not lead to a statistically significant difference in performance despite a slight improvement in some cases. For context-based embeddings, the results show that BERT overall outperforms ELMo, especially for datasets consisting of long documents. Compared with classic embeddings, both achieve an improved performance for datasets with short texts while the improvement is not observed for longer texts.

Assuming that a word embedding is selected and used to represent the input text at this stage, the next step of the process is to learn a sequence representation for text classification (see Figure 2.2). The following sections introduce various neural network models that can be used for this purpose.

## 2.2.2 Convolutional neural network

Convolutional neural network (CNN) [160, 164] is a variant of feed-forward neural network originally employed in the field of computer vision. It consists of multiple convolutional layers, each of which acts as a local filter to extract features from the input by sliding (or convolving) it, like the cells in visual cortex of the human brain receiving local features (e.g. light or contour, known as receptive fields) of an input image. Figure 2.4 shows the workflow of CNN as an example of sequence representation learning. In the first stage, the input matrix (i.e. the word representations/embeddings) is converted to feature/activation maps through the convolution layer. The convolution process consists of element-wise multiplications between the input matrix and the defined local filters which are analogous to windows. Mathematically, the local filters are simply arrays of weights, and their number and region size can be customised. In this example, there are two sizes of such filters: 2 and 3, each of which has 3 filters <sup>3</sup>.

---

<sup>3</sup>Here the number of filters is set to be 3 for demonstration simplicity and it is much more than this in practice.

For the 3 filters with region size 3, each slides from the top to bottom of the input matrix (element-wise and sum if the sliding stride is 1) and output 3 feature maps (or representation) consisting of vectors with each size being  $T - 3 + 1$  (here  $T$  stands for the number of input tokens). This is the same for the 3 filters with region size 2, outputting 3 feature maps consisting of vectors with each size being  $T - 2 + 1$ .

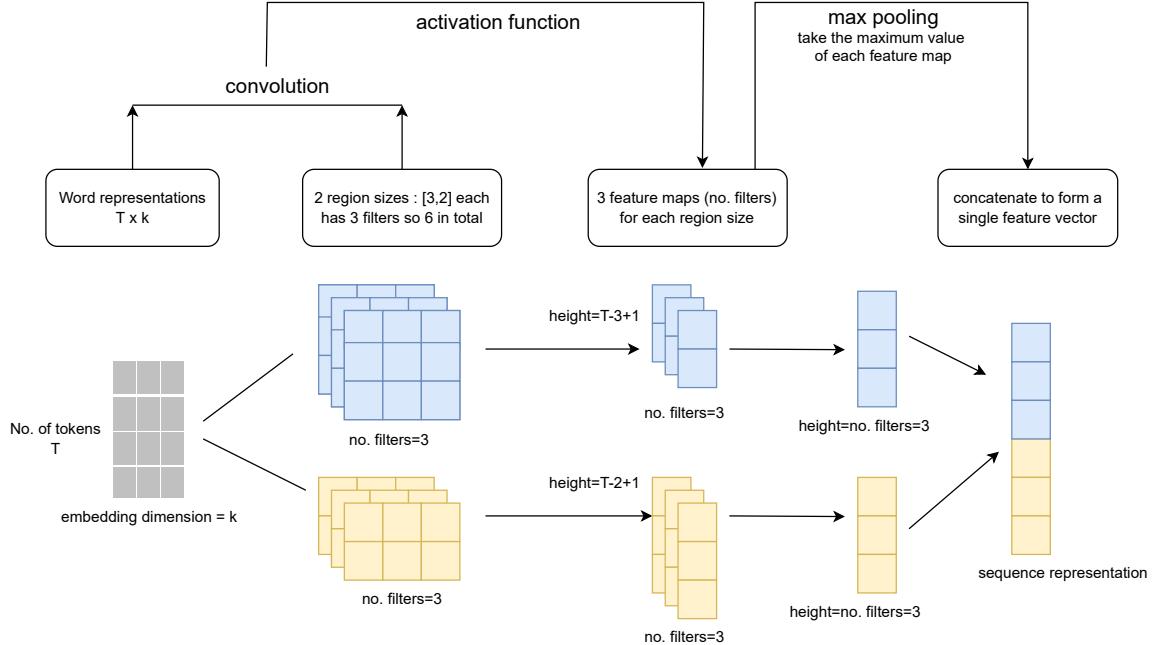


Figure 2.4: CNN as an example of sequence representation learning

Following the convolutional layer, a pooling (or sub-sampling) layer is applied to reduce the spatial size of the representation, while retaining the most salient information of the previous representation. For instance, max pooling in Figure 2.4 refers to the sub-sampling operation that takes only the maximum number out of the vectors of 3 filters of each size and thus outputs a single vector with height being 3 (i.e. the number of filters). In practice, the convolution and pooling processes are usually conducted multiple times to learn deeper representations. In this simple example, the sequence representation is eventually generated by concatenating the two vectors from the pooling layer. CNN is a widely-used model as a feature extractor for learning representations in text classification [20, 26, 58, 162] because of its focus on finding local clues within textual data. Taking a document as an example, there are often phrases or n-grams that are in different places but are very informative as to which topic the document relates to. CNN is good at finding such local indicators, irrespective of the positions.

## 2.2.3 Recurrent neural network

Recurrent neural networks (RNNs) are a class of neural networks that process data in a sequential way [123]. Its sequential attribute makes it well suited to NLP tasks such as text generation and text classification. In basic terms, given an input text sequence represented by  $T$  word vectors, each unit of RNNs takes each word vector at a time step and processes it to next unit. A time step refers to a specific point in the processing of the sequence, at which the RNN processes a single word vector. Hence, each unit not only takes the input of current time step but the output from previous time step. Depending on the number of tokens in the input sequence, the number of units in RNNs is dynamic. In RNNs, the parameters and activation function are usually shared across all time steps, which is known as *tying* or *sharing* of parameters in neural networks. For a RNN, depending on the task, it usually has different output format. For a text generation task, a RNN model is used to train a language model where the model predicts the next word given the previous sequence of words. Hence the output in this case is a sequence of vectors where each vector indicates the word prediction at that time step. For text classification, RNNs simply output a sequence representation (a vector) summarising the whole sequence, which is then used to get the probability distribution across all classes.

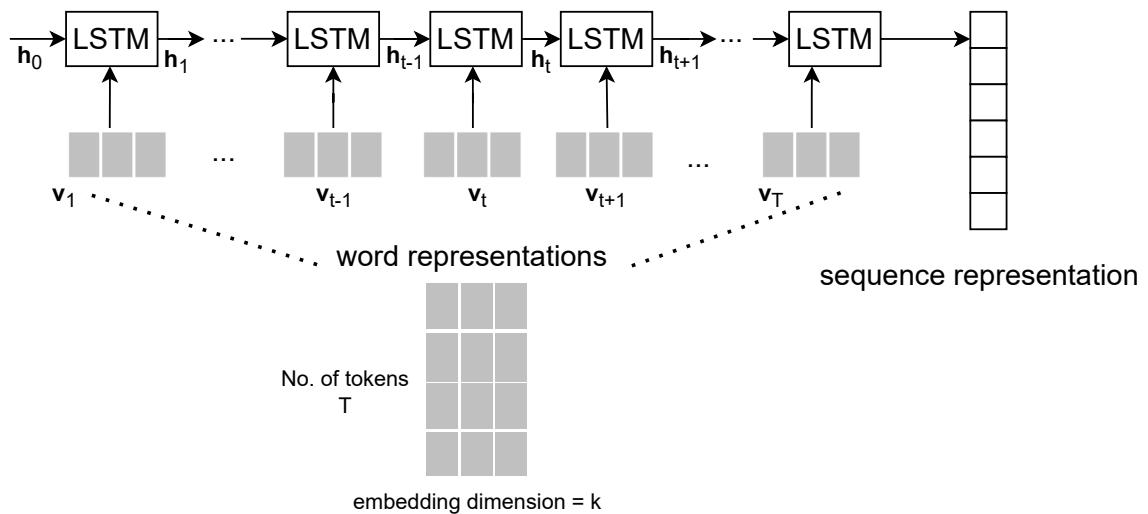


Figure 2.5: LSTM as memory cell of RNN for sequence representation learning

Regarding the unit of RNNs, long-short term memory (LSTM) [45, 168] is a memory-based unit in a RNN. Although there are many other variations of RNN-based memory units, such as GRU [18], only LSTM is detailed below due to its popularity and ability to maintain information for long sequences <sup>4</sup>. Figure 2.5 shows the uni-directional (left-to-right) workflow of LSTM as memory cell of RNN for sequence representation

<sup>4</sup>Check [45] for the gradient vanishing and exploding problem

learning. In a general sense, it encodes the input matrix to a sequence representation. At each time step  $t$  in a token vector  $\mathbf{v}_t$ , it summarises important information in the sequence as far as a hidden state  $\mathbf{h}_t$ , using the previous hidden state  $\mathbf{h}_{t-1}$  as input. It does so by remembering or forgetting information through a set of transition functions (or so-called “gates”) in the unit. Ultimately it summarises the sequence by outputting a single vector at the last time step. The LSTM transition functions are defined as follows [168]:

$$\begin{aligned}\mathbf{e}_t &= \sigma(\mathbf{W}_i \cdot [\mathbf{h}_{t-1}; \mathbf{v}_t] + \mathbf{b}_i) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_f \cdot [\mathbf{h}_{t-1}; \mathbf{v}_t] + \mathbf{b}_f) \\ \mathbf{q}_t &= \tanh(\mathbf{W}_q \cdot [\mathbf{h}_{t-1}; \mathbf{v}_t] + \mathbf{b}_q) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_o \cdot [\mathbf{h}_{t-1}; \mathbf{v}_t] + \mathbf{b}_o) \\ \mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{e}_t \odot \mathbf{q}_t \\ \mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)\end{aligned}\tag{2.20}$$

where  $\sigma$  and  $\tanh$  stand for sigmoid and hyperbolic tangent activation functions respectively. Element-wise multiplication is denoted by  $\odot$  and ; refers to the concatenation of vectors. The parameters  $\mathbf{Ws}$  and bias  $\mathbf{bs}$  are shared across all time steps and normally are randomly initialised before training (the initial hidden state  $\mathbf{h}_0$  is usually initialised with zeros). To better understand how the transition works,  $\mathbf{f}_t$  can be described as the forget gate that controls what information from the old memory cell should be forgotten. The input gate  $\mathbf{e}_t$  is used to control what new information is to be stored in the current memory cell and  $\mathbf{o}_t$  is described as an output gate to control what to output based on the memory cell  $\mathbf{c}_t$  (the initial memory state  $\mathbf{c}_0$  is usually initialised with zeros). In LSTM-based RNNs, the above process happens recurrently from time 0 to  $T$ .

Unlike a LSTM, which summarises information in one direction from left to right, bidirectional LSTM (BiLSTM) sees the context of a token at time step  $t$  in both left-to-right and right-to-left directions. In BiLSTM, the right-to-left process applies the same the transition functions as in the left-to-right process. In practice, there could be multiple layers of LSTM units stacked upon each other to learn deeper representations of the input data. In addition, depending on the problem domain, there exists different ways of generating the final sequence representation, such as concatenation of the mean of the outputs at each time step (bag-of-means) [55] or only the output at the last time step as in the case of Figure 2.5.

Given the extensive studies in recent years on deep models for text classification, CNN and LSTM are just two examples of many in the literature. For example, [166] combines CNN with LSTM for text classification; [156] applies hierarchical attention net-

works; [168] improves text classification by integrating bidirectional LSTM with two-dimensional max pooling; and [107] applies a bi-attentive classification network (BCN) with pre-trained contextualised ELMo embedding, achieving strong performance in text classification.

## 2.2.4 Pre-trained language models

More recently, transfer learning has gained great success in language understanding, which pre-trains on large textual corpora to gain a pool of general knowledge through unsupervised learning and then fine-tunes on the given training dataset (knowledge transfer) through supervised learning for a specific language task such as text classification [117]. The pre-training usually refers to the language modelling step and the fine-tuning refers to the downstream task training step. The major efforts into transfer learning include ULMFiT [48], ELMo [107] and transformer-based pre-trained language models (PLMs) [24, 111, 112, 132]. The former two are pre-trained language models based on recurrent networks. One drawback of such language models is that the sequential nature of recurrent networks is not good for parallelisation within training sequences. This is critical when the length of any sequence is very long where batching across the training sequences is limited by memory constraints [132]. As an important step towards tackling this issue, Ashish et al. 2017 [132] proposed the Transformer architecture that dispenses with recurrence and convolutions entirely and instead applies an attention mechanism for sequence representation learning. The Transformer mainly consists of an encoder and a decoder, which was originally proposed for machine translation. It is noted that in the context of machine translation, the input and output sequences are in different natural languages but share a common semantic representation. The encoder in the Transformer is used to learn the representation of a source sequence, while the decoder generates a target sequence based on the source representations from the encoder. Since the onset of Transformer, much work combining it with the idea of transfer learning has been introduced in recent years, achieving great success in various language processing tasks such as text classification and text generation. This line of work can be divided into three categories: encoder-based, decoder-based, encoder-decoder based.

### 2.2.4.1 Encoder models

The representative work in this category is BERT [25]. Depending on the transformer encoder, it is pre-trained using big unlabelled text data in an unsupervised (or usually referred to as self-training) manner on two tasks: masked language modeling (MLM)

and next sentence prediction (NSP). To better describe BERT’s pre-training, Figure 2.6 depicts the workflow of BERT at the pre-training stage.

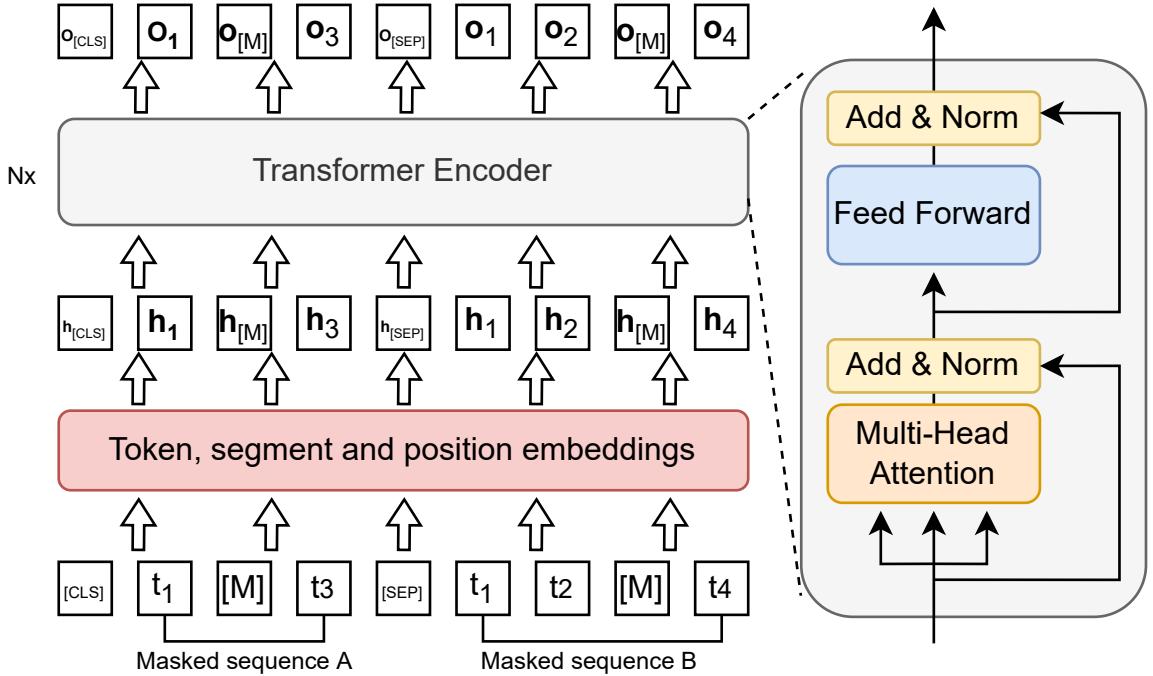


Figure 2.6: The workflow of transformer-encoder based BERT at pre-training

To achieve the MLM and NSP tasks, a pair of sequences A and B are used as the inputs and the pair are masked. The masking is conducted on the tokens of the input pair where a small portion of randomly-chosen tokens are masked (denoted as the symbol [M] in the figure) while the remainder are unchanged (denoted as  $t_1, t_2, t_3, t_4$ ). In addition to the masked tokens, two special tokens are used in BERT: [CLS] and [SEP]. The former is the special classification token whose final output state is used as the aggregate sequence representation for classification tasks (the NSP task at pre-training and downstream classification tasks at fine-tuning). The latter is the separation token used to distinguish sequence A from B. In BERT, the input pair are tokenised by WordPiece [152], which takes a word part instead of a whole word as a token. Once the input tokens are prepared, the next step is to represent them with vectors via the embedding layer. In BERT, each token  $t_i$  is represented by the element-wise addition of three types of embeddings: token, segment and position embeddings.

$$\mathbf{h}_i = \text{Embed}_{\text{tok}}(t_i) + \text{Embed}_{\text{seg}}(t_i) + \text{Embed}_{\text{pos}}(t_i) \quad (2.21)$$

where  $\mathbf{h}_i \in \mathbb{R}^{d_{\text{model}}}$  ( $d_{\text{model}}$  is the embedding dimension or the model’s hidden state size). The token embedding is used to represent the unique language information of each token, i.e., its content. The segment embedding consists of two vectors where the first

one indicates that  $t_i$  is from segment A and the second implies that it is from segment B. Hence, the purpose of segment embedding is again to enable the model to distinguish the two input sequences. The position embedding adds the positional information of each token to its representation so that the model is aware of the positions of each token.

Having represented the input tokens with vectors, let  $\mathbf{H}$  be the hidden state that is a matrix representing all tokens. As introduced earlier, BERT is based on the transformer encoder relying on an attention mechanism to learn deep representations for the input sequences. The transformer encoder is a building block of BERT, which can be viewed as a layer of a neural network with two sub-layers. The first is known as the scaled dot-product cross-attention sub-layer and the second refers to a simple linear feed-forward sub-layer. In each sub-layer, there is a residual connection between the input and output, along with a normalisation on the output. The attention sub-layer plays a crucial role in allowing the representations of individual tokens to depend on one another. Before being fed to the attention sub-layer, the representation of each token is treated independently. The scaled dot-product attention mechanism is then used to allow a token to “communicate” with other tokens, namely enabling the dependency between the token representations. The attention is said to be cross or bi-directional since a token is not only “communicated” with the tokens on the left but also the tokens on the right. Mathematically, it is achieved by introducing three matrices: the keys  $\mathbf{K}$ , the queries  $\mathbf{Q}$  and the values  $\mathbf{V}$ . To obtain them, they are simply linearly transformed from the hidden representations  $\mathbf{H}$ , calculated as follows:

$$\begin{aligned}\mathbf{Q} &= \mathbf{H}\mathbf{W}^Q \\ \mathbf{K} &= \mathbf{H}\mathbf{W}^K \\ \mathbf{V} &= \mathbf{H}\mathbf{W}^V\end{aligned}\tag{2.22}$$

where  $\mathbf{H} \in \mathbb{R}^{L \times d_{\text{model}}}$  and  $L$  refers to the number of input tokens or the sequence length. The weights  $\mathbf{W}^Q, \mathbf{W}^K, \mathbf{W}^V \in \mathbb{R}^{d_{\text{model}} \times d_{\text{model}}}$  are learnable parameters of the model. Using the keys, queries and values, the output of the attention function can be viewed as a weighted sum of the values. The weights are calculated by a softmax function on the scaled dot-product of the queries with the keys, presented as follows:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}\tag{2.23}$$

As the equation shows, the dot-product is scaled by being divided by the root square of the keys’ dimension size  $d_k$  (it is equal to  $d_{\text{model}}$  in this case). To know the importance

of other tokens to a token, the token attends to information of other tokens, which is quantified by the attention weights. The attention weights can be viewed as the similarity scores between the token’s key and the queries of other tokens, computed by the scaled dot-product. As a result, the tokens with higher similarities pay attention to the token with greater importance (higher weight). In reality, instead of applying such a single attention function on the  $d_{\text{model}}$ -dimensional  $\mathbf{Q}$ ,  $\mathbf{K}$  and  $\mathbf{V}$ , the multi-head attention approach is used to capture multiple aspects of a token’s attention to other tokens, calculated as follows:

$$\begin{aligned} \mathbf{H}_{\text{att}} &= \text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) \mathbf{W}^O \\ &\text{where } \text{head}_i = \text{Attention}\left(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V\right) \end{aligned} \quad (2.24)$$

where  $\mathbf{W}_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$ ,  $\mathbf{W}_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$ ,  $\mathbf{W}_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$  and  $\mathbf{W}^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$  are the learnable parameters of the model. In this case,  $d_k = d_v = d_{\text{model}} / h$  where  $h$  indicates the number of heads and  $d_v$  is the values’ dimension size. The multi-head attention concatenates individual attentions computed by the attention function on the projected keys, queries and values, which allows a token to jointly attend to information of other tokens in multiple aspects. Let the output of multi-head attention be denoted as  $\mathbf{H}_{\text{att}}$ . It is added to the previous hidden representations  $\mathbf{H}$  known as the residual connection, followed by layer normalisation [5]:

$$\mathbf{H} = \text{LayerNorm}(\mathbf{H} + \mathbf{H}_{\text{att}}) \quad (2.25)$$

Now the hidden state  $\mathbf{H}$  is re-assigned by the normalised addition of the previous hidden state and the attention output. Then this is fed to the position-wise feed-forward sub-layer that consists of two linear transformation with a ReLU activation in between. The first projects the hidden state to an intermediate dimension and the second projects it back to the original dimension  $d_{\text{model}}$ , presented as follows:

$$\mathbf{H}_{\text{ff}} = \text{FFN}(\mathbf{H}) = \text{ReLU}(\mathbf{H}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2 \quad (2.26)$$

where  $\mathbf{W}_1 \in \mathbb{R}^{d_{\text{model}} \times d_{\text{ff}}}$ ,  $\mathbf{W}_2 \in \mathbb{R}^{d_{\text{ff}} \times d_{\text{model}}}$  and  $d_{\text{ff}}$  stands for the intermediate dimension that is usually greater than  $d_{\text{model}}$ . Similar to the attention sub-layer, the residual connection along with a layer normalisation is then conducted on the hidden state and the transformation output.

$$\mathbf{H} = \text{LayerNorm}(\mathbf{H} + \mathbf{H}_{\text{ff}}) \quad (2.27)$$

So far the transformer encoder of BERT has been described and it can be viewed as a layer of a neural network. In BERT, there are usually multiple such layers stacked together for the purpose of learning deep representations. In other words, the output of the first layer  $\mathbf{H}$  is used as the input for the next layer whose components remain the same as described previously: the attention sub-layer and transformation sub-layer. This repeats  $N$  times where  $N$  is the number of hidden layers<sup>5</sup>. For the output of the last layer, there are deep hidden representations for each input token, denoted as  $\mathbf{o}_i$  (see Figure 2.6). There are two types of tokens whose output states require special attention. The output state of [CLS] can be treated as the aggregate representation of the input sequences and hence it is used for the NSP task. During the pre-training stage, the task of Next Sentence Prediction (NSP) involves classifying whether a given sequence B follows a given sequence A in the unlabeled text corpus. If sequence B is the next sentence in the corpus after sequence A, it is considered a positive example for the NSP task. If sequence B is randomly sampled from the corpus and does not immediately follow sequence A, it is considered a negative example for the NSP task. The output states of the masked tokens [M]s go through a softmax layer for computing the loss of the MLM task where the true labels are the original tokens before being masked. To pre-train a BERT, the losses of the two tasks are joined together as the objective function to optimise the model’s parameters via back-propagation.

When it comes to downstream tasks fine-tuning, the parameters of BERT are initialised with the pre-trained parameters so that it inherits the general language knowledge learnt from the pre-training stage. To fine-tune BERT for a sequence pair classification task such as textual entailment, the input remains the same as the input for pre-training except for the masking. For a sequence classification task such as crisis tweets categorisation, still without the masking, the input only contains a single sequence (i.e., only sequence A). To be consistent with the pre-training, the fine-tuning normally takes the output state of [CLS] as the sequence representation for text classification. To fit BERT on various classification tasks via fine-tuning, a task head such as a linear layer is usually added on the top of BERT, which projects the sequence representation to the class space that estimates the probability distribution across all classes. Taking text classification as an example, the loss function can be written as follows where  $\mathbf{W}_o$  and  $\mathbf{b}_0$  are the weights and bias of the last linear layer projecting the output hidden state of [CLS] to class space.

$$J(s) = - \sum_{i=1}^m p(y_i|s) \log \hat{p}(y_i|s) \quad (2.28)$$

$$\hat{p}(y|s) = \text{softmax}(\mathbf{o}_{[\text{CLS}]}\mathbf{W}_o + \mathbf{b}_0)$$

---

<sup>5</sup>Depending on the versions of BERT,  $N$  varies. The base version of BERT contains 12 such layers and the large version has 24 layers.

Based on the transformer encoder, BERT adopts the MLM and NSP tasks for pre-training, resulting in great success in a wide range of language tasks via fine-tuning. This has prompted much follow-up work to optimise it. The optimisation is mainly seen in the literature regarding the memory use, computational cost and design choices of BERT. Since BERT has been introduced, many variants have been proposed to address certain shortcomings and incorporate improvements. The following is a selection of the most prominent among these.

- **DistilBERT** [118] is a distilled version of BERT. Compared to the original BERT, it has fewer trainable parameters and is thus lighter, cheaper and faster during training and inference. Given the size of the reduced model, the original paper reports that it still keeps comparative language understanding capabilities and performance on downstream tasks.
- **RoBERTa** [80] is an optimised variant of BERT with several changes made. First, it uses a byte-level BPE as the tokenizer [136] and uses the MLM as the only pre-training task (NSP is removed). Besides this, some changes are made to the hyper-parameters for pre-training, including much larger mini-batches and learning rates, etc. The results show that these changes help boost the downstream performance, which indicates that the previously design choices for BERT are sub-optimal.
- **ALBERT** [64] is a derivative of BERT that is mainly optimised for memory efficiency with two parameter-reduction techniques. First, it splits the embedding matrix in a BERT-like architecture into smaller matrices (separation of the embedding size and the hidden size), leading to reduced memory use while mathematically maintaining equivalent effect. In addition, it uses repeated layers, where the parameters are shared across different BERT hidden layers. This optimisation results in a much smaller memory footprint although the computational cost remains similar to the original BERT (the same iteration through all hidden layers is still required).
- **ELECTRA** [19] maintains essentially the same architecture and size as the original BERT except for a change in the embedding matrix as in ALBERT. What makes it stand out is that it adopts a different pre-training approach. Unlike the MLM pre-training objective used in BERT, it trains a generator using the MLM objective to replace tokens in a sequence and meanwhile it trains a discriminator with the objective of identifying which tokens were replaced by the generator in the sequence. It is shown to outperform other transformers on language understanding benchmarks when using the same amount of computational power.

- **DeBERTa** [42] is optimised upon RoBERTa using fewer training data by introducing two novel techniques for pre-training: disentangled attention mechanism and an enhanced mask decoder. In DeBERTa, the representation for each token consists of two vectors that encode its content and position respectively. The disentangled attention calculates the attention weights between tokens by applying disentangled matrices on the representations of their contents and positions. Besides this, instead of using the output softmax as in BERT, DeBERTa applies an enhanced mask decoder to predict the masked tokens for pre-training. The results show that DeBERTa achieves improved performance on many downstream tasks as compared to RoBERTa even using half of its training data.

#### 2.2.4.2 Decoder models

Unlike encoder-based PLMs, the decoder-based PLMs have been widely studied in the literature given their strong generalisation capability in text generation. The most well-known models of this type are the Generative Pre-Training (GPT) family [11, 110, 111]. Compared to BERT-like models, GPTs are pre-trained on the traditional language modelling task with the architecture being similar to the transformer decoder. Regarding the architecture, similar to BERT, the encoder of a GPT consists of two sub-layers: the attention sub-layer and the feed-forward layer. In contrast with the cross attention sub-layer in BERT, the attention sub-layer of GPT is called a “masked self-attention layer”. The major difference is that the masked self-attention only allows the current token to pay attention to its previous tokens, i.e., it is only informed by the tokens on the left side instead of both sides as in the cross attention of BERT (unidirectional versus bidirectional). In mathematical terms, the attention weights for the current token are computed by the dot product between the keys and queries of only the tokens on the left side (see Equation 2.23).

Regarding the pre-training task of GPTs, the language modelling is referred to as the next token prediction task or casual language task (CLM) distinguished from MLM. In general terms, CLM takes a sequence of previous tokens as the input (known as the context window) and uses the context to predict the next token. At the pre-training stage, the loss function for pre-training a GPT is defined as follows:

$$J_1(s) = - \sum_{i=1}^T \log \hat{p}(t_i | t_1, \dots, t_{i-1}; \theta) \quad (2.29)$$

The objective of CLM is to maximise the likelihood of current token  $t_i$  given previous tokens  $t_1, \dots, t_{i-1}$ . When adapting it to a downstream task via fine-tuning, the loss function can be defined as follows:

$$J_2(s) = - \sum_{i=1}^T \log \hat{p}(y | t_1, \dots, t_{i-1}; \theta) \quad (2.30)$$

This defines the objective of generating the target token  $y$  (i.e., the next token  $t_i$ ) given the context tokens  $t_1, \dots, t_{i-1}$ , which applies to downstream tasks that are based on sequence generation, such as question answering. In particular for a text classification task, the equation can be simplified as follows:

$$J_2(s) = - \log \hat{p}(y | t_1, \dots, t_T; \theta) \quad (2.31)$$

where the target token  $y$  refers to the label or class name of the input sequence  $\{t_1, \dots, t_T\}$ . To summarise GPTs in fine-tuning, the downstream task can be viewed as a conditional generation task that coincides with the pre-training CLM task.

The GPT family of language models includes GPT-1 [110], GPT-2 [111], GPT-3 [11], and ChatGPT<sup>6</sup>, etc. These models differ in terms of both the amount of training data they use and their model size. For instance, GPT-3 has 175B parameters, which is 10 times more than GPT-2 (1.5B parameters) and GPT-2 has 10 times more parameters than GPT-1 (117M parameters). In terms of data size, GPT-3 also uses significantly more text data for pre-training than GPT-2, which itself uses more data than GPT-1. ChatGPT is fine-tuned from a large-scale GPT model (GPT-3.5) using reinforcement learning from human feedback, excelling at answering user questions in a dialogue setting. The development of GPTs has seen the trend of large language models for various downstream tasks driven by the finding in the literature that larger language models with more data brings better generalisation capability [57].

#### 2.2.4.3 Encoder-Decoder models

The architecture of encoder-decoder PLMs is more like a standard Transformer comprising an encoder and a decoder [132]. The encoder-decoder PLMs are usually referred to as sequence-to-sequence (seq2seq) models since they take a sequence as the input and generate a sequence as the output. In many cases, the input sequence is also called the source sequence and the output sequence is called the target sequence. At a high level, the encoder learns to encode the source sequence to a vector that can represent its contextualised linguistic features. Conditional on the source representation, the decoder then learns to generate the prediction words iteratively. Mathematically, given a source sequence  $s^a: \{t_1^a, t_2^a, \dots, t_{T^a}^a\}$ , the seq2seq model generates predictions, i.e., the

---

<sup>6</sup><https://openai.com/blog/chatgpt/>

target sequence  $s^b: \{t_1^b, t_2^b, \dots, t_{T^b}^b\}$  through a parameterised estimation of conditional probability distribution as follows.

$$J(s^a, s^b) = - \sum_{i=1}^{T^b} \hat{p}(t_i^b | t_{1:i-1}^b, f(t_{1:T^a}^a; \theta_e); \theta_d) \quad (2.32)$$

Where  $f(t_{1:T^a}^a; \theta_e)$  refers to the mapping function from the source sequence  $s^a$  to its contextualised representation, learnt by the encoder with parameters  $\theta_e$ . Likewise,  $\theta_d$  is the parameters for the decoder to learn the function of conditional generation:  $\hat{p}(\cdot)$ . From this equation, it is easy to know that the next token of the target sequence  $t_i^b$  is generated conditional on both the source sequence and previous tokens of the target sequence.

To look into the inner parts of the architecture of decoder-encoder PLMs, the encoder consists of hidden layers with each layer containing a cross-attention sub-layer and a feed-forward sub-layer, which corresponds to the encoder layer described in Section 2.2.4.1. The decoder consists of hidden layers with each layer containing a masked self-attention sub-layer, an encoder-decoder cross attention sub-layer and a feed-forward sub-layer where the self-attention sub-layer and feed-forward sub-layer coincide with the decoder layer as described in Section 2.2.4.2. The self-attention sub-layer is the same as the masked self attention used in GPTs, ensuring the next token that is generated is conditional on the previous tokens of the target sequence. The encoder-decoder attention is similar to the cross attention used in BERT. The difference is that the attention weights in the encoder-decoder attention are calculated between the keys of the encoder’s tokens and the queries of the decoder’s tokens. This attention ensures that the target sequence is generated conditional on the source sequence.

Due to the seq2seq attribute, the encoder-decoder PLMs can be easily adapted to various language tasks that can be viewed as a text-to-text problem. For text classification, the document becomes the source textual sequence and the label becomes the textual target sequence. For question answering, the question can be the source sequence and the answer can be the target sequence. This also applies to other language tasks such as text generation, summarisation, etc. By converting the input examples of various tasks to the text-to-text format, the encoder-decoder PLMs hence can easily be trained on multiple tasks at the same time (i.e., multi-task learning) based on the Equation 5.2. One important such work is T5 [113], which is pre-trained on a multi-task combination of unsupervised and supervised tasks where each task is converted into a text-to-text format. For unsupervised tasks, T5 adopts a denoising mechanism

as the pre-training objective using big unlabelled text data where the source sequence is partially corrupted and the objective is to generate the target sequence that predicts the corrupted parts of the source sequence. For supervised tasks, T5 relies on existing annotated resources of various bench-marking language tasks and converts them into a text-to-text format for pre-training.

By pre-training in the manner of T5, it has been shown to exhibit strong performance in downstream tasks via fine-tuning. This has prompted many follow-ups of T5 for different use cases. To just name a few of many; mT5 [155] is a multilingual T5 model pre-trained on big text data from more than 100 languages; byT5 [154] is a T5 model pre-trained on byte sequences instead of sub-word token sequences, boosting performance in tasks that are sensitive to spelling and pronunciation; LongT5 [41] is a variant of T5 designed especially for long sequences; BART [68] is pre-trained based on different training objectives, which introduces noise to the input by means of various operations including token deletion, order shuffling, text in-infilling, etc., exhibiting strong performance in downstream generation tasks.

## 2.3 Conclusions

This chapter has introduced the general background of this research. The background review put its focus on text classification as the crisis messages categorisation task is in essence a classification problem. Considering that the major contributions of this thesis utilise neural network based approaches, this chapter has covered the general ideas of machine learning and the basics of neural networks. In addition, much of the content of this chapter has focused on the use of various neural networks including CNN, RNN and PLMs for text classification. They are important models upon which the contributions of this research are built. In particular, the introduction to the Transformer-based PLMs (Section 2.2.4) provides essential background of the crisis messages categorisation problem in low-data settings, which will be reviewed in the next chapter.

# **Chapter 3: Crisis Messages**

## **Categorisation in Low-data Settings**

---

As mentioned in Chapter 1, it is difficult to obtain labelled data relating to emerging crisis events and it can take time to annotate data, so there is a need for methods that can be used to categorise social media messages related to crises in low-data settings. This chapter reviews existing methods that can be used for this purpose and thus be used as baselines for evaluating the work presented in the later chapters of this thesis. The review first covers a family of methods called “domain adaptation”, which involves adapting a model trained on labelled data from one domain (the source domain, a distribution of data) to another domain (the target domain, a different distribution of data). When the methods are used for crisis message categorisation, known as “crisis domain adaptation” and discussed in Section 3.1, it involves adapting a model trained on labelled data from past crisis events (i.e. the source domain) to categorise messages from a current crisis (i.e. the target domain). The review also covers existing methods for training a model on a small amount of labelled data from the target domain, known as “few-shot learning”. These methods are relevant for crisis message categorisation when labelled data from the past events is not available and it is feasible to annotate a small dataset for emerging events so the model can be trained on the small labelled dataset. These methods are discussed in Section 3.2. To further reduce or eliminate the reliance on labelled target data, the review also covers existing methods for training a model without any labelled data from the target domain, known as “zero-shot learning”. These methods are relevant for crisis message categorisation when no time and annotation effort is available for annotation so the model can be trained only on unlabelled data from emerging events. These methods are discussed in Section 3.3. Finally, related datasets and evaluation metrics used by this research are covered in Section 3.4.

## 3.1 Crisis domain adaptation

In crisis response, it can be difficult to use computational models to categorise messages from a new crisis event when labelled data for that event is not immediately available. To overcome this challenge, crisis domain adaptation involves building a model for a categorisation task using data from one or more past events (the source domain) and adapting it to categorise messages from one or more current events (the target domain). In order for this to be effective, the categorisation task must be the same between the source and target domains, meaning that the set of categories used for categorisation must be the same. For example, if the model was trained to categorise messages from past flooding events by the type of aid requested, it would need to use the same set of aid categories to categorise messages from a current earthquake event. The following presents the details of crisis domain adaptation.

**Definition:** Given a categorisation (classification) task  $\mathcal{T}$  with a set of pre-defined aid categories, a source domain is represented by the data of  $n$  events  $D_s : \{D_s^1, D_s^2, D_s^n\}$  and a target domain is represented by the data of  $m$  events  $D_t : \{D_t^1, D_t^2, D_t^m\}$ . Domain adaptation trains a model on  $D_s$  and tests the model on  $D_s$  within the same task  $\mathcal{T}$ . For simplicity,  $D_s$  is said to be the source data and  $D_t$  is the target data. The categorisation task  $\mathcal{T}$  may be composed of multiple sub-tasks, each with its own set of categories, and domain adaptation can still be applied as long as the sub-tasks and their corresponding categories remain the same between the source and target domains. This is known as multi-task adaptation. Depending on the difference between the source domain and target domain, domain adaptation can be divided into in-domain and cross-domain adaptation. In-domain adaptation implies that the  $m$  events in the target domain are the same as the  $n$  events in the source domain, which is equivalent to fully supervised learning (training and testing on the same domain). Cross-domain adaptation emphasises that the  $m$  events in the target domain are different from the  $n$  events in the source domain (i.e., different data distribution such as hurricane versus flooding), which is the focus of this research<sup>1</sup>. There are several categories of domain adaptation, including many-to-many, many-to-one, and one-to-one adaptation. In many-to-many and many-to-one domain adaptation (also known as multi-source domain adaptation), the source domain has multiple events ( $n > 1$ ) and the target domain has one or more events ( $m \geq 1$ ). One-to-one adaptation is similarly defined.

There are various methods in the domain adaptation literature that can be used when there is a varying amount of data available, including labelled source data, limited target data, or no target data. These methods can be grouped into categories based

---

<sup>1</sup>Unless stated, domain adaptation implies cross-domain adaptation in the remaining content of this thesis.

on the amount of data they require, with methods that require more data being listed first.

- **Semi-supervised domain adaptation:** In addition to the usage of labelled source data, semi-supervised domain adaptation uses a small amount of labelled target data plus a larger quantity of unlabelled target data. To differentiate it from semi-supervised learning, it additionally uses labelled source data for model building.
- **Supervised domain adaptation:** In contrast to semi-supervised adaptation, supervised domain adaptation uses labelled source data and only a small amount of labelled target data with no unlabelled target data. The difference with supervised learning is that it uses labelled source data and a small quantity of labelled target data.
- **Unsupervised domain adaptation:** Unsupervised domain adaptation is a branch of domain adaptation research that focuses on using labelled data from the source domain and unlabelled data from the target domain to build a model. This is of particular interest in real-world situations where it is costly to obtain labelled data for the target domain.
- **Target data independent domain adaptation:** This gives the strictest limitation to the availability of target data. It uses labelled source data for domain adaptation, which avoids using any form of data related to the target domain. Although limited work has been conducted for domain adaptation that requires no target data, it is deemed an important research problem in areas such as crisis message categorisation where the acquisition of unlabelled data of target crises is not only very expensive but also impractical given the inherent time constraints (i.e. because it is not practical to wait for enough data to be gathered as an event unfolds before starting to categorise messages). With this motivation, the major contributions of this research in domain adaptation fall in this category.

This section reviews major approaches that have been proposed for domain adaptation in these categories. Among the approaches, some are proposed for non-crisis related tasks (such as movie reviews sentiment classification), which can be easily transferred to crisis message categorisation, and others are directly proposed for crisis message categorisation. To emphasise the importance of target data independent domain adaptation (the major focus of this research), the former three categories are reviewed as a joint category: target data dependent domain adaptation.

### **3.1.1 Target data dependent domain adaptation**

The central idea behind domain adaptation is to align the source and target domain distributions or reduce the shift between the source and target data distributions [148]. The alignment refers to the process of bringing the data from the two domains into a common space where they can be compared and used to train a model. This can involve techniques such as projecting the data onto a shared latent space or matching the distributions of the data in the two domains. The goal of aligning the data is to make it easier to transfer knowledge from the source domain to the target domain, so that a model trained on the source domain can be more effective at making predictions on the target domain.

In target data dependent domain adaptation, the goal is to make the source domain similar to the target domain using labelled data from the source domain, unlabelled data from the target domain, and/or labelled data from the target domain. There are two main approaches to achieving this in the literature: methods based on instance selection, and methods based on feature adjustment. These methods aim to bridge the gap between the source and target domains in order to improve the performance of a model trained on the source domain on the target domain.

#### **3.1.1.1 Instance selection methods**

Instance selection methods aim to adapt a classifier to a new domain by choosing a subset of instances from both the source and target domains for the adaptation process. When there are few labelled target data and unlabelled target data available, a common way to implement instance selection for domain adaptation is to use a two-step process [53, 82, 128, 131]. In the first step, a subset of instances is selected from both the source and the target domains. This can be done using various methods, such as selecting the most representative or informative instances, or using a search algorithm to find the subset of instances that maximises the performance of the classifier. In the second step, weights are assigned to the selected instances based on their relative importance or relevance for the adaptation process. This can be done using various techniques, such as assigning higher weights to instances from the target domain or to those that are more difficult to classify correctly. The final classifier is then trained on the combined set of selected and weighted instances.

When only unlabelled target data is available, the target data can be selected by pseudo-labelling the unlabelled target data using a classifier trained on the source data, and then added with their pseudo labels to the training data [21, 44, 71, 72, 105, 130]. Here, pseudo labels refer to predicted labels by the classifier and these predicted labels are treated as if they were ground truth labels. There are two approaches

to pseudo-labelling unlabelled target data: using hard labels, which are determined through self-training, or using soft labels, which are obtained through expectation-maximisation [69]. Hard labels are discrete class assignments, while soft labels are probabilities or confidence scores indicating the likelihood that an instance belongs to a particular class.

The self-training method [16] can be described with two steps. The first step is to train a classifier on the labelled source data that assigns hard labels (e.g., 0 or 1) to the unlabelled target with high confidence predictions. The second step is to combine the hard-labelled data with the source data for training the classifier. This two-step process repeats until the classifier converges or reaches the designated maximum number of iterations. In the expectation-maximisation (EM) method, the process can be divided into an expectation step and a maximisation step. In the expectation step, the expected likelihood of the target data is evaluated using a classification model with parameters estimated from the labelled source data. In the maximisation step, the parameters are optimised by maximising the expected likelihood evaluated in the expectation step. The goal is to find the set of parameters that best explains the observations in the target data. To maximise the expected likelihood, there are soft labels (e.g., 0.6 or 0.4) assigned to the unlabelled instances. In a similar way to the self-training method, EM is also an iterative method for instance selection and the iterative process does not stop until convergence or reaching the maximum number of iterations. The following gives a review of important instance selection methods that relate not only to crisis messages categorisation but other non-crisis related tasks.

In the work by [21], a EM-based domain adaptation approach was proposed for news topic classification using Naïve Bayes classifiers [114]<sup>2</sup>. Basically, this approach first estimates the priors (the parameters of a Naïve Bayes model) using the labelled source data and then an EM algorithm is applied to tune the model based on the distribution of unlabelled data from the target domain. Their experimental results found that the approach performs better than the traditional supervised algorithms including Support Vector Machine (SVM [98]) and Naïve Bayes classifiers, especially when the source domain and target domain are similar.

Another work by [53] proposed an instance selection (and weighting) framework for domain adaptation assuming the availability of labelled source data, a small set of labelled target data and a large amount of unlabelled target data. This work takes into account three major aspects to improve the adaptation performance, which is tested on several major NLP tasks including part of speech tagging, spam filtering, etc. First, it decides whether a training instance  $x$  in the source domain is “misleading” or not by

---

<sup>2</sup>This work studied three domains represented by news of three resources with unified labels: 20 Newsgroups, SRAA and Reuters-21578.

comparing its distribution for a class  $y$  in the target domain  $p_t(y|x)$  using labelled target data with its corresponding distribution in the source domain  $p_s(y|x)$  labelled source data. Besides, when assigning higher weight to labelled target instances than labelled source instances (more information from the target domain), this is more effective than excluding "misleading" source instances. Last, it is necessary to augment the training data with confidently predicted target instances.

In addition, in this work [130], they proposed a two-step EM based approach for domain adaptation for the task of sentiment analysis<sup>3</sup>. First, to better utilise the source domain knowledge, they proposed Frequently Co-occurring Entropy (FCE) to filter out non-generalisable (i.e., domain-specific) features and only keep the important features that have similar co-occurring probability in the source and target domains. Next, to better incorporate the target domain knowledge, they proposed Adapted Naïve Bayes (ANB), which is a weighted variant of the multinomial Naïve Bayes Classifier. The main idea behind ANB is to use a weighted EM algorithm to bring confidently pseudo-labelled target instances to the training set where the weight is shifted from the source domain to the target domain.

In another work by [105], they studied domain adaptation in the context of sentiment analysis on Twitter from the perspective of finding good source instances that can inform the target domain. To achieve this, they proposed two iterative algorithms based on EM and Rocchio SVM to select good labelled source instances. Unlike previous works [21, 53, 130] requiring unlabelled target data, this work assumes the availability of source domain data and a small amount of labelled target data. In the EM iteration, they train a SVM classifier on the labelled target data and use it to pseudo-label source instances . Last, only the confidently-classified source instances are added back to the training set.

Another work by [44] used a weighted Naïve Bayes classifier based on the EM algorithm, which is similar to the previous works [43, 130], for domain adaptation in the task of splice site prediction. When it comes to the question of how to select the unlabelled target data (i.e., to assign pseudo labels to the unlabelled target instances), they comparatively studied three approaches: using hard labels only via self-training, using soft labels only via EM and using a combination of soft and hard labels. The experimental results in the task of splice site prediction indicate that to use soft labels is generally better than the other alternatives.

Inspired by the aforementioned works, some works have been found in the literature for the study of domain adaptation in the task of crisis messages categorisation. For

---

<sup>3</sup>They studied three domains represented by reviews of three resources with binary sentiment labels: Education Reviews, Stock Reviews and Computer Reviews.

example, the work by [72] was considered to be the first work in adapting general domain adaptation approaches to the crisis domain. They studied domain adaptation in the task of classifying disaster-related tweets where an old disaster is viewed as the source domain and a new disaster is treated as the target domain. They make use of the unlabelled target data for crisis domain adaptation based on the iterative EM algorithm and a weighted Naïve Bayes classifier. In their proposed approach, they first train the Naïve Bayes classifier on the labelled source data (training data) and then use it to soft-label the unlabelled target data. To adapt the classifier from the source domain to the target domain, the soft-labelled instances are then added to the training set and assigned with more weight for the next iteration of training. The iterative process continues until the classifier converges or reaching a designated number of iterations.

In another work by [71], the problem of crisis domain adaptation was studied similarly to their previous work [72]. The difference is that they use unlabelled target data for crisis domain adaptation based on a weighted classifier and the iterative self-training algorithm instead of the EM algorithm as in [72]. In this work, the unlabelled target instances are hard-labelled by the pre-trained classifier. When combining the hard-labelled instances with the original labelled source instances, only the most confidently classified instances are added to the training set. In experiments, they set up the classifier with different choices including Naïve Bayes, SVM, Random Forests and Logistic regression. The results show it gives overall better performance when the classifier is Naïve Bayes than other choices. In addition, they compared the NB-based self-training approach with the NB-based EM approach and supervised NB using labelled source data only. It is found that the NB-based self-training approach overall outperforms the other two approaches.

Building upon previous works [71, 83], this work [82] proposed a hybrid domain adaptation approach for crisis tweet classification. The hybrid approach adopts the iterative self-training algorithm as in [71] for building the classifier. However, it takes two extra steps to select only a better-aligned subset of labelled source data before the self-training step. The first is to apply the Least Squares Non-Negative Matrix Factorization (LSNMF) [65] to reduce the instance-feature data matrix representing the source-target pairs of events to a low-dimensional space. In selecting the subset of labelled source data, KNN [40] is then applied on the reduced matrix and used to select only the top  $k$  nearest instances for each instance of the unlabelled target data. They found that the NB self-training using unlabelled target data outperforms other classifier choices and NB using labelled source data only, which are findings similar to those of [71].

Instead of using traditional ML models as in [71], this work [70] combined self-training with deep learning models including a CNN model and pre-trained language mod-

els (a BERT and a BERTweet, whaich is a variant of BERT pre-trained on English tweets [96]) for crisis domain adaptation. In experiments with self-training, it is found that BERTweet overall outperforms CNN and BERT. In experiments with and without self-training, the results show that self-training with unlabelled target data helps improve the adaptation performance, especially in situations where a large amount of target unlabelled data is available. This indicates the benefit of using unlabelled target data for crisis domain adaptation as compared to without it, which coincides with previous works [71, 82].

### 3.1.1.2 Feature adjustment methods

Another branch of target data dependent domain adaptation methods is based on feature adjustment. It looks into different approaches of aligning the source domain with the target domain; that is to change the feature representations. Given features such as bag-of-words fixed vocabulary representations or continuous representations, the core idea behind these methods is to keep as many cross-domain features (generalisable) as possible while simultaneously filtering out domain-specific features. Depending on whether the generalisable features are learnt through representation learning using deep learning models, the feature adjustment methods can be divided into two categories; traditional based and deep learning based.

Among the traditional based methods, these works by [8, 9] proposed structural correspondence learning (SCL) for domain adaptation in the tasks of part of speech tagging and sentiment analysis. SCL is adopted to choose features (known as pivotal features) that can help the classifiers distinguish between domains. By using the labelled source data and unlabelled target data, pivotal features are chosen in two ways: based on their common frequencies in both domains and mutual information in the situations such as sentiment analysis where frequently occurring words may vary significantly across domains. When some labelled target data is available, SCL uses it to correct the misalignment of the pivot features. Similarly, with the use of few labelled target data, this work by [22] studied multi-source domain adaptation by adopting a simple method to represent the target and source data where the feature representations are divided into three types: source domain-specific, target domain-specific and general. Regarding finding generalisable features for domain adaptation, this work by [54] proposed a simple approach for domain adaptation with two-steps. The first step is to find a set of generalisable features across domains and the second step is to only keep the target domain-specific features. In a similar way to SCL, this work by [104] introduced a spectral feature alignment (SFA) method to align domain-specific words across domains into unified clusters with the help of domain-independent words acting as a bridge. In SFA, the clusters are used to reduce the gap between domain-specific words,

which help train the classifiers for the target domain. Another important traditional feature-based method was by [129] who proposed CORrelation Alignment (CORAL) for domain adaptation for object recognition and sentiment analysis tasks. CORAL is a simple unsupervised domain adaptation approach to align the data distribution of the source domain with that of target domain. It minimises domain shift by aligning the covariance of source and target distributions. In the CORAL approach, the covariance statistics are calculated for each domain and the distributions are aligned by adjusting the source features using the covariance of the target features. This method was then adapted and used with a self-training iterative algorithm for adaptation in a crisis domain. [74].

Unlike the traditional methods aiming to obtain the domain-invariant features from bag-of-words fixed vocabulary feature representations, deep learning based methods apply neural networks to learn continuous feature representations for both domains. In the literature relating to feature based target data dependent domain adaptation, this is usually achieved by the autoencoder networks or domain adversarial networks.

An autoencoder is a type of neural network that learns a latent representation for an input via an encoder and then reconstructs the representation via an decoder in an unsupervised manner [17]. To apply it to the domain adaptation problem, the encoder can be trained on the source data and unlabelled target data and used to obtain the feature representations of both domains. Subsequently, a supervised classifier can be trained on the representations for the target domain classification. For example, this work [37] introduced Stacked Denoising Auto-encoders (SDA) for domain adaptation for the task of sentiment analysis. In SDA, multiple autoencoders are stacked together as a neural network model and the model is trained by denoising or reconstructing the latent representations from the encoder whose input is corrupted. Once the model is trained, the encoder is applied to obtain the latent representations of both domains. The latent representations are then used as the input representations for an SVM classifier. Other works using autoencoders in a similar way for sentiment classification domain adaptations can be found in [17, 167, 169, 170, 171]. Another important autoencoder-based work is by [36] who proposed deep reconstruction classification networks (DRCN) for domain adaptation. The DRCN consists of an autoencoder that is co-trained with a source classification network. The encoder of the autoencoder is shared with the source classification network and thus can learn information from both domains. Finally, the source classification network trained on source domain data is adapted to the target domain. Built on top of DRCN, this work [76] used a RNN (LSTM) network as the autoencoder for crisis domain adaptation relating to the task of disaster tweet classification. It was found that the reconstruction task can bring benefits to the adaptation performance as compared to the classification task alone.

As an alternative to autoencoder based methods, domain adversarial neural networks (DANN) [34] achieve domain adaptation by training a feature extraction network with the supervised source classification network at the same time. The feature extraction network aims to learn domain-invariant features by maximising the loss of a domain classifier that is used to distinguish between the source and target domains. To achieve the joint learning, the loss of the source classification network is combined with the loss of the domain classifier as the objective function for optimising the networks' parameters via backpropagation. Based on this adversarial idea, some related works have been conducted in the literature. For example, this work [78] proposed an end-to-end adversarial memory network for cross-domain sentiment classification where an attention mechanism is used to capture pivotal features. Another work [27] adapted BERT for domain adaptation, achieving state-of-the-art performance in cross-domain sentiment classification. To enable BERT to be domain-aware and distil domain-specific features, BERT is post-trained on a target domain masked language task and a novel domain-distinguishing pre-training task in a self-supervised manner. By doing this, the adversarial training is then conducted to derive the robust domain-invariant features. Given its strong performance, DANN has been also adapted and applied to the crisis domain in recent years. This work [77] used VGG-19 as the backbone of DANN for disaster damage images identification where the adversarial training is used to find a transformation that makes the source and target data indistinguishable. In another work, this work [2] proposed a CNN based domain adversarial network with graph-based semi-supervised learning for crisis domain adaptation in the task of crisis message relevance identification. The network applies a joint training of a supervised, a semi-supervised and a domain classifier on data of both domains to learn good domain-invariant representations. The semi-supervised classifier is trained to learn latent representations by predicting contextual nodes in a graph that encodes similarity between labelled and unlabelled training examples. The domain classifier is trained to distinguish the domains and the supervised classifier is trained for the main classification task. The experimental results show significant improvements of this work over baselines.

### 3.1.2 Target data independent domain adaptation

Target data independent domain adaptation methods assume no availability of any form of target data, including labelled and unlabelled data. Multiple previous studies have shown that the performance of domain adaptation with no target data is inferior to that with large amounts of unlabelled target data [2, 70, 71, 82]. This indicates that domain adaptation can be difficult without incorporating the information from the target domain. Despite this challenge, target data independent domain adaptation is considered to be an important research problem in the area of rapid crisis messages

categorisation on social media. This is mainly because, in real-word crisis response, the need for categorising a new crisis is urgent and the target data is not readily available when the new crisis occurs. Target data independent domain adaptation is proposed to meet this use case. From the perspective of building a computational model, the research question becomes how to build the model trained on past events (source domains) and adapt it to a new event (target domain).

Given its importance and that existing works in this direction are limited, this research mainly focuses on target data independent domain adaptation. Among the existing works [70, 73, 97], one common feature is that they use pre-trained word embeddings as generalised representations for crisis tweets, followed by a classification model or a sentence encoder for classification. Since the word embeddings are usually pre-trained on general texts such as Wikipedia pages, books or news, the general language information learnt from the pre-trained stage can be used to represent textual sequences in different domains<sup>4</sup>. In the literature, the commonly-used pre-trained word embeddings include Word2Vec, GloVe, and FastText (see Section 2.2.1 for further discussion). Once the tweets are represented by the word embeddings, the subsequent classification can be achieved by a simple traditional ML model such as Naïve Bayes, which uses the representations as the input features. Another way to achieve classification is to apply a sentence encoder to learn a contextual sequence representation from the word representations. The sentence encoder normally refers to neural network models such as a CNN, LSTM or BERT. In such an encoder, a classification linear layer can be easily added on the top of the sentence encoder and hence the classification is done along with the sentence representation learning. Below gives a review of important existing works related to target data independent domain adaptation.

The work by [97] studied the problem of rapid classification of crisis-related tweets. This work is conducted to particularly address the real-world crisis situation where no target data is available when a new target crisis occurs. When there is no target data available at the beginning of the crisis occurrence, they found that a CNN model with crisis-specific pre-trained word embeddings outperforms traditional ML models and a CNN with general pre-trained word embeddings such as Word2Vec. However, to further improve the adaptation performance, they suggest using unlabelled target data by either regularised adaptation or instance selection as the target crisis unfolds over time.

Another work by [73] identified the problem of no target data being available for domain adaptation in time-critical situations as disasters occur. They proposed using pre-trained word embeddings as the generalised representations for crisis tweets. They

---

<sup>4</sup>Within the problem domain of this research, these textual sequences are messages relating to crisis events.

explored a wide range of both word embeddings and sentence encodings with traditional ML algorithms for crisis adaptation classification tasks. They found that general pre-trained GloVe embeddings overall outperform other embeddings and the GloVe embeddings trained on crisis data bring better results on more specific crisis tweet classification tasks.

Another work by [70] comparatively studied target data independent and target data dependent domain adaptations using CNN and BERT models with self-training. Another similar study of applying LSTM with various word embeddings for domain adaptation is found in [69]. The experimental results show that self-training with a large quantities of unlabelled target data boosts the performance as compared to without it. When there no target data is available, the results show that the pre-trained language model BERT performs much better than CNN and LSTM. Besides this, it further improves the performance when the BERT is pre-trained on tweets generally (i.e. not specifically on crisis-related tweets).

## 3.2 Crisis few-shot learning

In rapid crisis response, domain adaptation achieves crisis message categorisation by adapting a model trained on past crisis events (source domains) to a current new crisis event (target domain). However, one premise of domain adaptation is that the categorisation task of target events has to be exactly the same as that of the source events. In other words, the pre-defined aid types of the source data need to be consistent with those of the target data so that the adaptation is possible. Although this requirement is satisfied in some real-world situations, it is insufficient in situations where the responders seek new aid types in an unfolding event that were not considered for past events, even though annotated data with different labels is available. For example, the responders may look for a new water-based rescue aid type in a storm while this aid type has never been recorded (or been relevant to) past events such as fire.

This marks the importance of crisis few-shot learning in rapid crisis response, which uses no labelled source data but a small quantity of labelled or unlabelled target data for crisis message categorisation. To achieve crisis few-shot learning, a small data set comprising a few equal number of instances per class from the target event and an unlabelled corpus of the target event are also used when available. The former is relatively inexpensive to obtain as the annotation for a few instances can be done quickly at the beginning of a new crisis with little human labour. The latter is easy to acquire as the new crisis unfolds. Crisis few-shot learning asks the question of how

to build a classification model using the few labelled target instances as well as the unlabelled target instances for unseen crisis message categorisation.

By examining the literature, crisis few-shot learning has been rarely studied. However, many works on few-shot learning have been conducted for other NLP tasks and these works can be adapted to the crisis domain and used as strong baselines for the research into crisis few-shot learning. The works on few-shot learning can be grouped into two main categories: prompt-based learning methods and augmentation-based methods. This section gives a review of them separately.

### 3.2.1 Prompt-based learning methods

Prompt-based methods for few-shot learning borrows the idea of transfer learning using pre-trained language models (PLMs). In the standard method of fine-tuning PLMs for downstream classification tasks, the labelled instances of target tasks are fed to the models directly. However, when the labelled instances are few, the fine-tuning performance can be poor, due to having insufficient data to learn from. Although this problem can be alleviated by very big PLMs such as GPT-3 [11], which uses few instances as the input context to generate the classification results directly with no need for fine-tuning, they are too large to be deployed in domains that require efficiency such as rapid crisis response. Given small or medium-sized PLMs, prompt-based learning conducts few-shot fine-tuning for downstream tasks in a different way. Instead of feeding the raw instances to the models, prompt-based learning reconstructs them via a set of prompt templates before feeding them to the models. The idea behind this is to reconstruct the instances so that they can be fine-tuned in a way that is more similar to the pre-training, leading to better transfer learning from pre-training. Three works have been identified that are particularly important in the area of using prompt-based learning with PLMs for few-shot classification. These are strong benchmarks for use in this research, and are discussed as follows.

Pattern-Exploiting Training (PET) was introduced by [120] — a semi-supervised approach that trains language models on a given task by reformulating few instances as cloze-style questions. The cloze-style questions are natural-language sequences encoding some form of task description where the label field is masked for prediction similar to the pre-training MLM task in PLMs such as BERT and RoBERTa. In PET, a set of hand-created task-specific patterns (i.e., prompt templates) are used to convert the training instances so that the PLMs can be fine-tuned in a way similar to the pre-training<sup>5</sup>. To leverage the unlabelled data of a given task, PET trains

---

<sup>5</sup>When applying PET to crisis few-shot learning, a rescue-related message may appear as “I am on

multiple PLMs on the few training data converted by different patterns and then applies the ensemble of PLMs to soft-label the unlabelled data. This coincides with the previously-mentioned self-training in domain adaptation where unlabelled data are pseudo-labelled first. Similarly, PET subsequently uses the soft-labelled data to train a classifier in a supervised fashion. By testing PET in experiments, it exhibits state-of-the-art performance in many general language tasks and even better performance than supervised methods in several tasks. However, one drawback of PET is its reliance on the unlabelled data and the hand-designed task-specific patterns.

To address this limitation of PET, the work by [35] proposed a technique for better few-shot fine-tuning of medium or small-sized language models (LM-BFF). In reformulating an input sequence with a few task demonstrations are appended to the template-based prompt as the input context for language model training. These demonstrations comprise the instances sampled from the few shot data of classes other than the class of the current input instance. Additionally, the templates for constructing the input are not hand-designed but automatically generated by leveraging a seq2seq language model (T5). They only use a few annotated instances as supervision and also explore automatically generated prompts and fine-tuning with demonstrations, avoiding the use of unlabelled data and making it a task-agnostic method. In experiments, LM-BFF showed much better performance than standard fine-tuning on many tasks and improved performance especially with the demonstrations as compared to PET. Although LM-BFF outperforms standard fine-tuning on small datasets, there is still a gap to match its performance with that of standard fine-tuning on larger datasets. It is also demonstrated with high variance for the results of LM-BFF due to the randomness in the selection of a small amount of data for model training, which is a common issue of many existing few-shot methods. In addition, there is evidence to indicate that LM-BFF favors certain tasks, e.g., the task generalisation problem.

Similarly, with focus on prompt design, the work by [161] proposed a different approach to make small language models for better few-shot learners. They proposed Differentiable prompt (DART) that uses a few unused tokens in the language model, serving as the differentiable template and label tokens so that they can be optimised with backpropagation. DART uses differentiable prompts for few-shot learning, aiming to generate more discriminative representations than the fixed prompts that are used in PET. In addition, DART also applies an auxiliary fluency constraint object to ensure the association among the prompts embeddings. An ablation study shows that the differentiable templates and labels and the fluency constraint object all help improve the

---

a boat, rescue needed!”. Assuming a pattern designed in this form: “⟨TEXT⟩. It’s about ⟨LABEL⟩”, the message is then converted by this pattern to a fill-in-the-blank style “fill-in-the-blank” question: “I am on a boat, rescue needed! It’s about \_\_\_” where the underlined blank text “\_\_\_” implies the label field masked for prediction.

performance. It achieves better performance than LM-BFF without demonstrations and comparable performance to LM-BFF plus demonstrations. However, the instability problem and task generalisation problem still exist as open research questions, as stressed in previous few-shot learning works.

### 3.2.2 Augmentation based methods

Given few-shot data, prompt-based learning methods reformulate the input by using prompts to improve transfer learning from PLMs. Augmentation based methods instead tackle the few-shot learning problem by augmenting the original few-shot data. Data augmentation is an idea to bring better performance by increasing the quantity of training data. It was originally used in computer vision to obtain synthetic training images by various ways of modifying the original images such as rotation, cropping, shearing or scaling, etc [125]. In contrast to images, data augmentation in NLP (i.e., text augmentation) is deemed to be a non-trivial research problem since minor modifications to a piece of text may lead to a drastic change in its semantic meaning. This brings two major challenges. First, to bring new knowledge to the training data, the augmented synthetic instances are expected to be as diverse as possible to the original training instances, known as the “lexical diversity” challenge. Apart from being diverse, the synthetic samples also have to be label-aligned regarding their semantics (i.e., their semantic meanings should coincide with the labels that are assigned to them, otherwise confusing the model), known as the “label alignment” or “semantic fidelity” challenge.

Data augmentation for text classification has been widely developed in the literature. The work by [163] demonstrated that replacing words or phrases with lexically similar words such as synonyms or hyponyms/hypernyms is an effective way to perform text augmentation with minimal loss of generality. The authors identify the target words according to a predefined geometric distribution and then replace words with their synonyms from a thesaurus. Similarly, the work by [147] proposed Easy Data Augmentation (EDA) for text classification that generates new instances from the original training data with four simple operations; synonym replacement, random insertion, random swap, and random deletion, while another work by [32] further explore these substitution techniques, particularly for text generation. The work by [145] instead exploit the distributional knowledge from word embedding models to randomly replace words or phrases with other semantically similar concepts. The work by [61] build upon this idea by replacing words based on the context of the sentence, which they achieve by sampling words from the probability distribution produced by a bi-directional LSTM-RNN language model at different word positions. Back translation

(BT) is another method for text augmentation [122, 124]. Here, new samples are generated by translating the original sentence to an intermediate language, before it is eventually translated back to the original target language. Sometimes, back translation with hops (BT-hops) is used where the original sentence is translated to multiple intermediate languages before it is translated back to the original language. More recently, to bring better label alignment, most works have used pretrained transformer-based language models for text augmentation by performing conditional text augmentation to obtain new sentences from the original training data. For example, the work by [151] proposed Conditional Contextual Augmentation (CBERT) that leverages the masked language model of BERT conditioned on label surface names to replace words contextually. Similarly, another work by [62] used a seq2seq model BART depending on the label names to replace text spans contextually (BART-Span). One common feature among these works is that they increase the quantity of training data by reformulating or translating the original data. The reformulation usually involves word substitutions with synonyms and the translation results in sentence rephrasing. This feature makes them good for obtaining new instances with good label alignment but difficult to bring diverse (new knowledge) instances to the original training data.

To ensure diversity, recent years have seen another line of work in text augmentation: generation-based methods. Here, novel instances are generated by generative language models trained on the original few training instances. For example, the work by [62] proposed fine-tuning GPT-2 for text augmentation. In fine-tuning GPT-2, the raw instances are concatenated with their corresponding class names as the input. At inference time, a new sequence of a class is generated by using the class name along with several seed words of a sampled original sequence as the input context. Although the GPT-2 in this work is able to generate instances that are diverse in terms of semantics, it is found that the generation that is conditional on labels and seed words still brings noise to the training data (the label misalignment problem). Taking the noise into account, another work by [4] (GPT-2- $\lambda$ ) introduced a post-denoising step to filter the noisy generated sentences. In this work, a new sequence of a class is generated that only depends on the class name. The post-denoising step involves training a classifier on the original training data and then using the classifier to make predictions for the generated sentences. The generated sentences with high-confidence predictions are selected and added to the training data. By testing different classification models, they found that using BERT as the classifier results in the best performance overall. It is also found that the post-denoising step helps to alleviate the label misalignment problem, thus resulting in better final performance.

## 3.3 Crisis zero-shot learning

Crisis few-shot learning is introduced for the real-world use case where the target categorisation task is different from the source categorisation task and so domain adaptation is not possible. Crisis zero-shot learning is introduced for the same purpose but to extend the use cases of crisis few-shot learning. In few-shot learning, a small quantity of instances per class are assumed to have been annotated. Although this annotation burden is small compared to fully-supervised learning, it is still time-consuming in some specific domains such as crisis categorisation. Additionally, to complete the annotation the class distribution is usually assumed to be uniform, i.e., equal numbers of instances per class are annotated. This uniform class distribution assumption may deviate significantly from the actual distribution, leading to bias in the classification models. To replace crisis few-shot learning, crisis zero-shot learning is considered to be a more challenging research problem as it achieves crisis message categorisation when there is no annotated data (i.e., zero labelled instances) from the target domain.

Considering the importance of crisis zero-shot learning, this research further investigates the potential of performing crisis message categorisation without any labelled data. Similar to crisis few-shot learning, little work has been found in the literature for crisis zero-shot learning, but many zero-shot learning studies have been conducted for other text classification tasks. This research in crisis zero-shot learning is inspired by the general works on zero-shot learning and hence they are used as strong baselines in the later chapters of this thesis. This section reviews them by dividing them into two categories: Indirectly-supervised learning methods and Weakly-supervised learning methods.

### 3.3.1 Indirectly-supervised learning methods

Given a classification task of a target domain, zero-shot learning aims to build a classifier that predicts test instances from that domain without supervision (i.e. without learning from any previously-annotated training instances). The lack of labelled target data makes it impossible to train the classifier in a directly-supervised way. This has motivated much research to train classifiers in an indirectly or distantly-supervised way. Indirectly-supervised methods transfer classifiers trained on external resources (e.g., labelled or unlabelled data from other domains) to the target domain. Similar to few-shot learning, zero-shot learning addresses the issue of different tasks between domains. Hence, the tasks across domains have to be unified so that the transfer learning with indirect supervision is possible. This task unification is normally achieved

by introducing a matching model between text pairs. Basically, the matching model is first trained on some external datasets comprising text pairs and its objective is to learn the semantic similarity between two pieces of text. When tested for the target classification task, the matching model assigns a class to a piece of text according to the similarity between the text and the class that is represented by its surface name or a description of its name. The following gives details of important such methods in the literature.

The work by [133] proposed a very simple method for zero-shot learning. They used pre-trained word embeddings to represent n-grams of testing instances (sentences or documents) and the classes' surface names. To assign one or more classes to a testing instance, the cosine similarity between the instance's n-grams and each class is calculated and used as the semantic similarity score for classification. Instead of relying on readily-available pre-trained embeddings, another work by [109] proposed three different LSTM-based neural networks to learn the relationship between text pairs consisting of news headlines along with their Search Engine Optimisation (SEO) tags. In this work, millions of news headlines along with SEO tags are used as the external resources to train the matching model that can be applied to classification tasks where each testing instance with the classes comprise the text pair for similarity calculation. Another work is done by [158], which proposed an entailment approach that applies pre-trained BERT to learn the relationship between text pairs that consist of the premises and hypotheses from three textual entailment datasets (RTE/MNLI/FEVER). The entailment datasets are external supervision resources with annotations associated with text pairs to indicate whether the premise entails the hypothesis (high semantic similarity) or contradicts the hypothesis (low semantic similarity). Hence, they are good resources for a model to learn semantic similarities between text pairs. Other similar works using indirect supervision resources for zero-shot learning can be found in [67] and [99], who studied zero-shot relation extraction by transforming it into a machine comprehension and textual entailment problem respectively.

In the aforementioned works, to assign a class to a testing instance, the instance has to be paired with every class's description, which leads to linear computation complexity as the number of classes increases. For optimisation, the work by [94] applied siamese networks and label tuning to tackle this inefficiency issue of the entailment approach at inference time. Another work to address the inefficiency issue is found in [108], who achieved zero-shot model adaptation to new classification tasks via generative language modelling. They trained GPT-2 on text pairs consisting of Reddit posts and multiple titles including correct ones and incorrect ones. Here, the text pairs are formatted as multiple-choice questions and the model's prediction is to identify the correct class given a testing instance. In addition, the work by [23] attempted to overcome the inference inefficiency issue by distilling the entailment matching-based models from [157, 158].

This method assumes the availability of unlabelled target data. The matching model is first used to pseudo-label the unlabelled target data with soft labels and then use it for target classification model training in a standard supervised way. This method is closely related to weakly-supervised learning, which is introduced next.

### 3.3.2 Weakly-supervised learning methods

Indirectly-supervised learning methods achieve zero-shot learning by building a matching model to learn semantic similarities between text pairs. Weak supervision based methods assume the availability of unlabelled target data and pseudo-label the unlabelled data as the training data for the target classification model. Hence, the quality of pseudo-labelled data usually determines the final performance of the classification model. The pseudo-labelled can be generated using class representations or obtained by a matching method between unlabelled instances and classes. The classes are typically expanded before they are matched with the unlabelled instances in order to represent them with richer meanings. Two methods of expansion are typically used. The classes can be expanded computationally by the addition of related words or phrases known as (extremely weak supervision [90]) or by adding a small number of human-provided seed words (known as weak seed-word driven supervision [146]). A lot of research has been done in the literature to examine class expansion and produce better-quality pseudo-labeled data. The most significant of these publications are reviewed below; they will serve as the research's foundation in subsequent chapters.

The work by [89] proposed a framework for Weakly-Supervised Neural Text Classification (WeSTClass) with two stages. The first stage is to use seed information to obtain the pseudo-labelled data for model pre-training<sup>6</sup>. In WeSTClass, the seed information can be adaptive to either the raw class surface names or human-provided seed words. Instead of matching unlabelled instances with the classes, they model the class semantics as a spherical distribution and generate pseudo instances of each class where class-related terms are diversified and concatenated to form a pseudo instance. After the model is pre-trained on the pseudo instances, the second stage refines the model by training it on the real unlabelled data; the predictions by the model on the unlabelled data are normalised and used as the ground truths to tune the model itself, known as the self-training stage. The results show that the self-training stage helps to substantially improve the final performance.

In another work by [88], they proposed a contextualised weakly-supervised (ConWea) approach for text classification. ConWea is a seed-word driven supervision method that

---

<sup>6</sup>To note, this pre-training is different to the pre-training in transfer learning.

converts user-provided seed words for each class into contextualised seed words (i.e., multiple interpretations of the same word) using their contextualised representations from pre-trained contextualised word embedding models such as ELMo and BERT. The contextualised seed words are used to create a contextualised corpus where the original seed words occurring in the unlabelled instances are replaced with their contextualised seed words. To refine and expand the contextualised seed words, a classifier is trained on the contextualised corpus, which works with a comparative ranking component to help filter out ambiguous words and add highly class-indicative keywords in an iterative manner. The training process stops after convergence has occurred, where no more ambiguous or class-indicative keywords remain. The final classifier is then used to make predictions for the test data of the target task.

Regarding extremely weak supervision that does not involve using human-provided seed words to enrich the classes, the work by [90] proposed to use label names only for text classification (LOTClass). In this work, the pseudo-labelled data is obtained by a token-level matching between the unlabelled instances and the classes. Before matching, a class is represented by its vocabulary (i.e., expansion) instead of its surface name and the vocabulary consists of the top-50 predicted words at the positions of the class’s surface name occurrences in the unlabelled instances by leveraging the masked language modelling (MLM) task of BERT. To expand the class effectively in this way, the exact class name should occur within the unlabelled instances, which is also essential to the subsequent matching step. In terms of matching, in a similar way to MLM, they proposed a masked category prediction (MCP) head on BERT where a token is assigned to a class if its top predicted words highly overlap with the class’s vocabulary. Here, the classification model comprising the MCP head and BERT is trained to predict the implied classes of the class-assigned tokens with them masked. In a similar way to [89], the model is finally self-trained on real unlabelled data, leading to further improved performance.

X-Class is another extremely weak supervision approach to text classification, proposed by [146]. They used class names only for text classification by building class-oriented instance representations first and then using the Gaussian Mixture Model (GMM) clustering algorithm [29] to obtain the pseudo-labelled data. X-Class can be broken down into three components: class-oriented instance representation estimation, instance-class alignment through clustering and classifier training based on confident classes. The first component obtains class-oriented instance representations by applying a tailed attention mechanism on the representations of instances and the class names based on pre-trained language models (specifically BERT). The second refers to pseudo-labelling where instances are aligned with the classes with confidence scores by GMM clustering. Finally, BERT is trained on the pseudo-labelled instances with high confidence scores.

## 3.4 Related datasets and evaluation metrics

In the domain of crisis message categorisation on social media, many benchmarking datasets have been created for research purposes. These datasets are important resources for testing the effectiveness of the methods proposed in this research. Referring back to the process of real-world crisis response introduced in Section 1.1, the datasets are reviewed by organising them into two categories: informativeness and information types. The informativeness datasets are used for the initial filtering, which is a binary classification problem aiming at finding crisis messages that are informative to users' aid needs. The information types datasets are used for further classification, aiming to categorise the informative messages into specific aid types known as information types such as “search and rescue”, “volunteer”, “donations” etc. One goal of this research is to investigate to what extent the proposed methods can also generalise to text classification tasks that are not specific to crisis messages. Therefore, a list of benchmarking datasets relating to other domains, such as emotion classification and topic categorisation, are used for testing the generalisation ability of the proposed methods. Based on the evaluation of previous works that use these datasets, most use the standard classification metrics such as Accuracy or F1 and a few use a tailored set of metrics for performance reporting and comparison. Hence, the tailored metrics associated with the datasets are reviewed in this section as well.

### 3.4.1 Informativeness

This section describes two crisis datasets commonly used for informativeness identification in domain adaptation: **nepal\_queensland** [71] and **CrisisT6** [101]. Table 3.1 presents basic information about the two datasets.

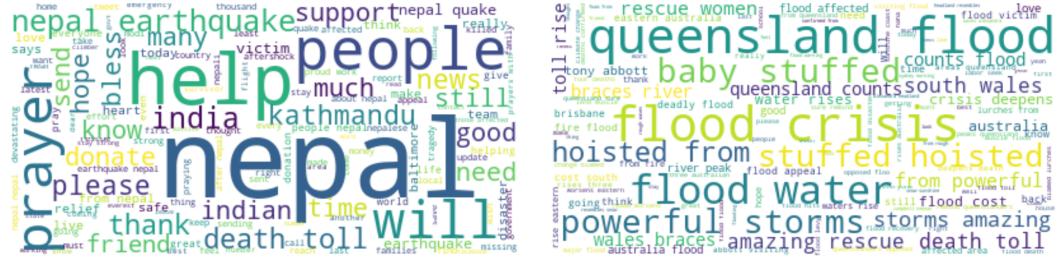
Dataset	No. of events	No. of classes	Class dist.	Data size	Task type
nepal_queensland	2	2	balanced	~21k	single-label, binary
CrisisT6	6	2	balanced	~60k	single-label, binary

Table 3.1: Basic information of informativeness datasets

- **nepal\_queensland** is a real-world crisis Twitter dataset collected during the 2015 Nepal earthquake (NE)<sup>7</sup> and the 2013 Queensland floods (QQF)<sup>8</sup>, originally proposed in [71] for crisis domain adaptation. Figure 3.1 depicts the word clouds of word distributions for the two events. This dataset consists of approximately

<sup>7</sup>[https://en.wikipedia.org/wiki/April\\_2015\\_Nepal\\_earthquake](https://en.wikipedia.org/wiki/April_2015_Nepal_earthquake)

<sup>8</sup>[https://en.wikipedia.org/wiki/Cyclone\\_Oswald](https://en.wikipedia.org/wiki/Cyclone_Oswald)



(a) NE: Nepal Earthquake

(b) QF: Queensland Floods

Figure 3.1: Word clouds of two events from nepal\_queensland

21,000 annotated tweets equally distributed across the two events, sampled from millions of tweets crawled through the Twitter API<sup>9</sup> using event-specific keywords and hashtags. In this dataset, each tweet is annotated by one of two classes (which are well balanced): *relevant* if the tweet contains aid-related information such as donations or food requests and *not\_relevant* if not.

- **CrisisT6** was originally released by [101]. It is a collection of approximately 60,000 annotated tweets posted during six crisis events with approximately 10,000 per event. The word clouds of six events are presented in Figure 3.2. They are real-world crisis events occurred between 2012 and 2013, which can be summarised in three broad categories: Flood (AF: Alberta Floods 2013<sup>10</sup> and QF: Queensland Floods 2013<sup>11</sup>), Hurricane (OT: Oklahoma Tornado 2013<sup>12</sup>) and SH: Sandy Hurricane 2013<sup>13</sup> and Explosion (BB: Boston Bombings 2012<sup>14</sup> and WTE: West Texas Explosion 2013<sup>15</sup>). Similarly, the tweets are sampled before annotation from around 10 million tweets collected through the Twitter API using keywords and geographical regions or coordinates. Each tweet in this dataset is annotated by one of two classes according to relatedness (a similar concept to informativeness): *on-topic* if the tweet is related to the event and *off-topic* if not and the tweets are balanced in terms of class distribution.

### 3.4.2 Information types

This section introduces three crisis datasets for information types classification: **TREC-IS** [12, 85, 87], **HumAID** [3] and **Situation** [81, 127]. These are suitable resources for

<sup>9</sup><https://developer.twitter.com/en/docs/twitter-api>

<sup>10</sup>[https://en.wikipedia.org/wiki/2013\\_Alberta\\_floods](https://en.wikipedia.org/wiki/2013_Alberta_floods)

<sup>11</sup>[https://en.wikipedia.org/wiki/Cyclone\\_Oswald](https://en.wikipedia.org/wiki/Cyclone_Oswald)

<sup>12</sup>[https://en.wikipedia.org/wiki/2013\\_Moore\\_tornado](https://en.wikipedia.org/wiki/2013_Moore_tornado)

<sup>13</sup>[https://en.wikipedia.org/wiki/Hurricane\\_Sandy](https://en.wikipedia.org/wiki/Hurricane_Sandy)

<sup>14</sup>[https://en.wikipedia.org/wiki/Boston\\_Marathon\\_bombing](https://en.wikipedia.org/wiki/Boston_Marathon_bombing)

<sup>15</sup>[https://en.wikipedia.org/wiki/West\\_Fertilizer\\_Company\\_explosion](https://en.wikipedia.org/wiki/West_Fertilizer_Company_explosion)



Figure 3.2: Word clouds of six events from CrisisT6

the studies of crisis domain adaptation, few-shot and zero-shot learning. Table 3.3 presents the basic information of these corpora, followed by a detailed description below.

**TREC-IS** is the dataset that has arisen as part of the challenges posed by the TREC Incident Streams (IS) track [85, 87]. The IS track is an annual task run as part of the Text REtrieval Conference (TREC)<sup>16</sup>, aiming to encourage more mature social media-based emergency response technology. It was launched in 2018 and has run in many editions since then<sup>17</sup>. In each edition after the first, the data from past editions was used as the training data and a new unlabelled corpus was used as the test data (and the training data for following edition). Hence, the TREC-IS dataset are accumulated, and are divided into subsets based on the track’s editions. By the time the track ended, it had accumulated four subsets of editions with annotations: 2018, 2019A, 2019B, and 2020A. Table 3.2 shows the important statistics of the four subsets.

Subsets	Data size	No. events	Avg. length
2018	17581	15	35
2019A	7098	6	37
2019B	9122	6	48
2020A	6658	15	42
Total	40459	42	40

Table 3.2: Statistics of TREC-IS sub-datasets where *No. events* means the number of events for a set. *Avg. length* refers to the average length of instances.

It is notable that each subset consists of tweets relating to different crisis events and that the events do not overlap between the subsets, which makes the dataset a suitable resource for many-to-many crisis domain adaptation. Combining all the subsets, the

<sup>16</sup><https://trec.nist.gov/>

<sup>17</sup>The track ended in 2021 and the editions include 2018, 2019A, 2019B, 2020A, 2020B, 2021A and 2021B.

TREC-IS dataset consists of approximately 40,000 annotated tweets across 42 crisis events. The events occurred between 2012 and 2020 including the COVID-19 pandemic [12]. For the full details of these events, the reader is referred to the home page of the IS track<sup>18</sup>. Regarding annotation, the dataset is annotated by two types of classes: “information types” and “priority levels”.

For information types, each tweet is assigned with one or more information types (thus this is a multi-label classification task). There are 25 information types defined in this dataset where 6 are defined as “actionable” and the remaining 19 as “non-actionable”. Actionable types are related to more urgent aid needs such as *a request for search and rescue* or *asking for a particular service or physical good*, etc. Non-actionable tweets tend to be less urgent including *asking people to volunteer to help the response effort*, *reporting first party observation*, etc. For a complete description of the information types, the reader is referred to Appendix A.1. In addition to information types, each tweet is also labelled with one of four priority levels indicating the urgency level of the tweet: *critical*, *high*, *medium* or *low*. As indicated by the annotation in this dataset, participants in the IS track are asked to develop systems that are capable of categorising the information types and estimating the priority of tweets in a supplied test set.

Regarding evaluation metrics, given that the tweets are annotated by both information types and priority levels, the IS track proposed a set of tailored metrics to uniformly evaluate the performance of the participating systems on the tasks of information type classification and priority estimation. The metrics can be broadly divided into four categories: **Ranking**, **Alerting Worth**, **Information Feed Categorisation** and **Prioritisation**, described as follows.

- **Ranking** (range 0 to 1): In this category, NDCG [52] is the priority-centric metric used to evaluate the quality of submitted test tweets ranked by their priority scores. By default, it measures the top 100 submitted tweets per event and reports the average across all events as the final score. A high NDCG score implies that the system has achieved a good quality of priority-based ranking. It is noted that this metric was not used for the 2019 editions, but was later introduced from 2020.
- **Alerting Worth** (range  $-1$  to  $1$ ): This is inspired by the alerting use case in a real-world emergency response system (i.e, identifying aid needs that require urgency). It not only measures the effectiveness of a system in generating true alerts but also penalises the system in generating consecutive false alerts that would make end users lose trust in the system. Two components of Alerting Worth are AW-HC and AW-A, which measure the effectiveness of true alerts

---

<sup>18</sup>[https://www.dcs.gla.ac.uk/~richardm/TREC\\_IS/](https://www.dcs.gla.ac.uk/~richardm/TREC_IS/)

within the scope of tweets annotated to be critical or high, and within the scope of all priority-level tweets respectively.

- **Information Feed Categorisation** (range 0 to 1): This is used to evaluate the aspect of information type classification performance by a system. To better reflect a system’s utility to emergency response officers, it consists of three specific metrics, Actionable F1, All F1, and Accuracy, denoted by CF1-H, CF1-A, and Cacc respectively. CF1-A macro-averages the F1 scores across all information types, while CF1-H macro-averages the F1 scores only across the 6 actionable types as presented in Table A.1. The two metrics indicate the performance of information type categorisation by only taking the target class per information type into account. Cacc computes the categorisation accuracy micro-averaged across information types.
- **Prioritisation** (range 0 to 1): This is applied to measure the performance of priority level predictions, consisting of two specific metrics: PErr-H and PErr-A. In IS 2019, PErr-H and PErr-A were used to measure the root mean squared error (RMSE) between the predicted priority scores and actual priority levels for actionable and all information types respectively. However, since IS 2020, the priority predictions were changed to categorical priority levels instead of numeric scores as in 2019. The metrics were thus changed to Actionable F1 (PErr-H) and All F1 (PErr-A) respectively. Both are computed by averaging the macro-F1 scores on priority label predictions per information type. Unlike PErr-A, which averages the F1 scores for all information types, PErr-H averages the F1 scores for actionable types only.

As a general reference, the metrics in the category of information feed categorisation indicate the effectiveness of the information type classification task while the rest evaluate the effectiveness of the priority estimation task. Considering the characteristics of this dataset, this research uses it as the resource for the study of many-to-many crisis domain adaptation (Chapter 4) where the subsets of previous years are used for model training and the subsets of late years are used for testing. To align with the IS track, the tailored metrics are used to compare the proposed methods in this research with baselines.

**HumAID** [3] is another dataset suitable for information type categorisation. This dataset consists of English tweets collected from 19 real-world disaster events. The events happened from 2016 to 2019, and various event types including wildfires, floods, hurricanes and so on<sup>19</sup>. In contrast to TREC-IS, the dataset consists of approximately 77,000 tweets annotated by information type with splits of training, development and

---

<sup>19</sup>[https://crisisnlp.qcri.org/humaid\\_dataset](https://crisisnlp.qcri.org/humaid_dataset)

test set, described as “humanitarian categories” in the associated paper [3]. The data in Figure 3.3 illustrates the distribution of word lengths for all examples within the dataset. It is evident from the graph that a majority of the text samples are brief, containing words between 10 and 50 words in length. In addition, each instance of this dataset is annotated by one of 11 humanitarian categories (and thus represents a single-label, multi-class classification problem)<sup>20</sup>. The categories include highly urgent aid needs such as *displaced people and evacuations, missing or found people* as well as less urgent aid needs such as *sympathy and support, caution and advice*. Among the 11 humanitarian categories, there are three non-informative categories: *don’t know can’t judge, not humanitarian* and *other relevant information*. In order to use this as a resource for information type categorisation after the informativeness identification, this research uses a modified version of the dataset where the instances of the three non-informative categories are removed. Figure 3.4 shows the word clouds of the 8 informative humanitarian categories of all examples within the dataset. This version only contains the informative categories whose surface names can effectively represent the meanings of the corresponding aid needs, which is well suited to the proposed methods of this research for crisis few-shot (Chapter 6) or zero-shot learning (Chapter 7) as opposed to the version with the non-informative categories.

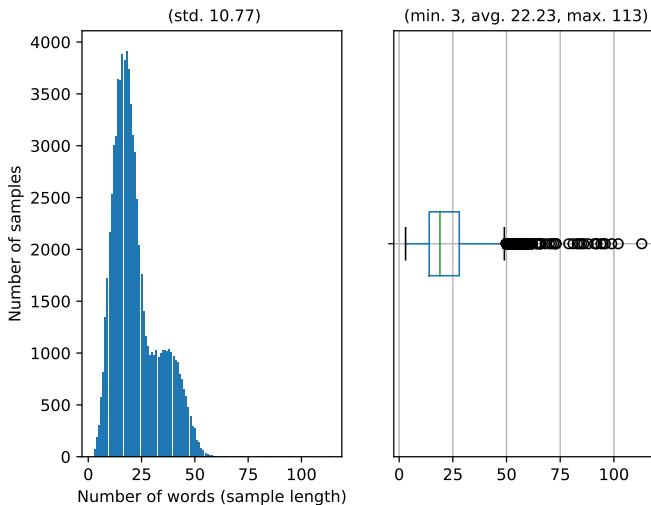


Figure 3.3: Word length statistics of HumAID

**Situation** [81, 127] is a multi-lingual dataset for detecting aid needs from crisis tweets written in a variety of natural languages. Although this dataset does not specify the crisis events from which the tweets are, the tweets are from a mixture of several crisis events. The dataset was originally designed for low-resource situation detection where annotated data is unavailable, which makes it a good resource to evaluate crisis

---

<sup>20</sup>The complete categories are *rescue volunteering or donation effort, sympathy and support, requests or urgent needs, infrastructure and utility damage, injured or dead people, caution and advice, displaced people and evacuations, missing or found people, don’t know can’t judge, not humanitarian, other relevant information*

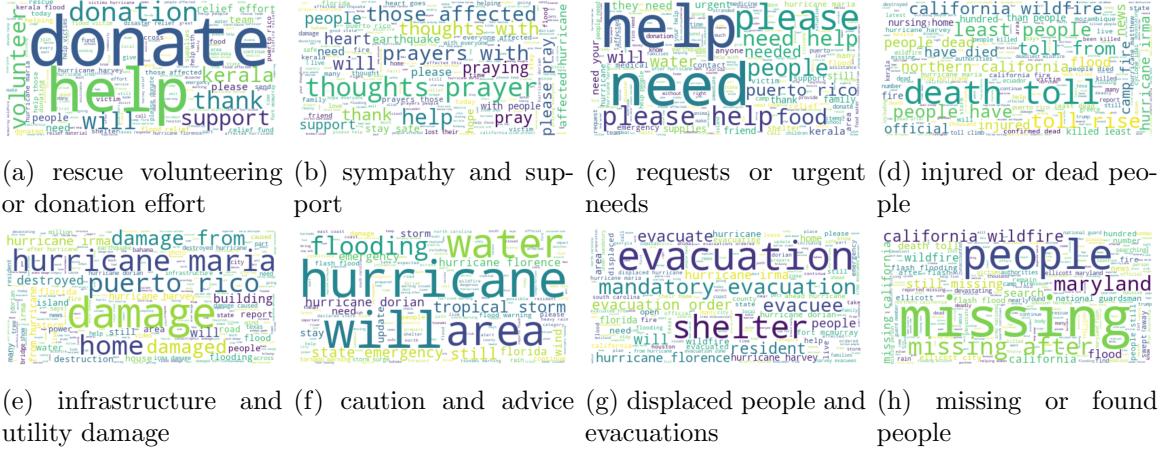


Figure 3.4: Word clouds of 8 informative humanitarian categories from HumAID

Dataset	No. of events	No. of classes	Class dist.	Data size	Task type
TREC-IS 42	25		imbalanced	~40k	multi-label, multi-class
HumAID 19	11		imbalanced	~77k	single-label, multi-class
Situation N/A	11		imbalanced	~6k	multi-label, multi-class

Table 3.3: Basic information of information types datasets

zero-shot learning. Unlike TREC-IS and HumAID, this dataset is annotated by a more general set of information types, known as “situation types” [127]. There are 11 situation types defined in this dataset: *shelter*, *search*, *water*, *utilities*, *terrorism*, *evacuation*, *regime change*, *food*, *medical*, *infrastructure*, and *crime violence*. By their surface names, it can be seen that some of them represent a general description of an event such as *crime violence* and *terrorism*, which are different to the specific aid needs in TREC-IS and HumAID. In a similar way to TREC-IS, each instance of this dataset is annotated by one or more of the 11 situation types (i.e., representing a multi-label classification task). Given the scope of this research, the English version of the dataset comprising approximately 6,000 annotated instances in splits of training, development and test set released by [81] is used. In this research, this dataset is used for zero-shot learning, which will be introduced in Chapter 7.

### 3.4.3 Beyond crisis

In crisis zero-shot and few-shot learning, the proposed methods are also evaluated on domains other than the crisis domain. Although the methods are proposed with the motivation of crisis message categorisation, the extension to other domains indicates that this research emphasises the generalisation capability of the methods across different domains. For this purpose, a list of benchmarking datasets beyond crises are used in this research. Table 3.4 presents the basic information of the datasets.

As this shows, these datasets are varied in multiple aspects. For example, they are from different domains including question type identification, emotion detection, sentiment analysis and news topic categorisation. To be specific, this research uses **Topic**, **UnifyEmotion** and **Emotion** as extended datasets in the study of crisis zero-shot learning and **SST-2**, **TREC**, **Emotion**, **AgNews**, **TwitterSentiment** in the study of crisis few-shot learning. The datasets are detailed as follows.

Dataset	Domain	No. of classes	Class distribution	Data size
TREC	question types identification	6	imbalanced	~6k
Emotion	tweets emotion detection	6	imbalanced	~20k
UnifyEmotion	mixed messages emotion detection	10	imbalanced	~48k
TwitterSentiment	tweets sentiment analysis	3	imbalanced	~60k
SST-2	short messages sentiment analysis	2	balanced	~70k
Topic	news topic categorisation	10	balanced	~1.4m
AgNews	news topic categorisation	4	balanced	~127k

Table 3.4: Datasets beyond crisis domain

- **TREC** [75] is a dataset for question type classification. The instances of this dataset comprises approximately 6,000 questions in short sentences. Approximately 4,500 questions are collected by Hovy et al. 2001 [47] and the rest are from TREC 8, TREC 9 and TREC 10, which are annual task runs similar to TREC-IS proposed by the Text REtrieval Conference (TREC). The questions are originally annotated with 6 coarse class labels and 50 fine class labels. This research uses the version with 6 coarse class labels: *Abbreviation*, *Description*, *Entities*, *Human beings*, *Locations* and *Numeric values* and uses the original splits of this dataset.
- **Emotion** [119] is a dataset for emotion detection comprising English Twitter messages that are pre-processed by removing hashtags and mentions. There are approximately 20,000 messages annotated with six emotion types: *sadness*, *joy*, *love*, *anger*, *fear* and *surprise*. This research uses the original splits of the dataset published by [119].
- **UnifyEmotion** [60] is another dataset for emotion detection. This dataset is constructed by combining multiple public emotion datasets from multiple domains including tweets, fairytales, artificial sentences, etc. Despite similar annotation, this dataset does not contain instances that overlap with Emotion. Instances are originally annotated by 9 emotion types plus a *none* label if no emotion applies: *surprise*, *guilt*, *fear*, *anger*, *shame*, *love*, *disgust* and *sadness* and *joy*. This research uses the version of the dataset from [158] without the “none” instances and uses the splits of the version released by [158]
- **TwitterSentiment** [115] is a sentiment classification dataset that consists of cleaned tweets, released by SemEval-2017 Task 4 Sentiment Analysis in Twitter.

It consists of approximately 60k instances annotated with three class labels: *positive*, *neutral* and *negative*. This research uses the original splits of the dataset published by [115].

- **SST-2** [126] is a benchmarking dataset for binary sentiment classification. It consists of approximately 70,000 short movie reviews annotated with *positive* and *negative* labels. This research uses the original splits of the dataset published by [126].
- **Topic** is a large-scale dataset for news topic categorisation. It refers to the Yahoo dataset released by [163] and consists of approximately 1.4 million news articles annotated by 10 news topics (well-balanced): *Education & Reference*, *Society & Culture*, *Sports*, *Entertainment & Music*, *Politics & Government*, *Computers & Internet*, *Family & Relationships*, *Science & Mathematics*, *Health* and *Business & Finance*. This research uses the original splits of the dataset published by [163].
- **AgNews** [162] is a dataset comprising approximately 1 million news summaries (short documents) collected by more than 2,000 news sources, which can be used for multiple research purposes such as clustering, classification and information retrieval. In its version for news topic categorisation, approximately 127,000 instances are annotated with four news topics: *World*, *Sports*, *Business* and *Sci/Tech*. This research uses the original splits of this version released by [162].

## 3.5 Conclusions

This chapter conducted a review of the specific domain of crisis message categorisation. Based on the proposals of this research, the review is conducted in three parts under the context of low-data availability for crisis messages categorisation: crisis domain adaptation, crisis few-shot learning and crisis zero-shot learning. For domain adaptation, both target data dependent and independent approaches are introduced. These approaches serve as important inspirations for the proposed methods in this research for crisis domain adaptation. In addition, important existing state-of-the-art zero-shot and few-shot methods are reviewed, which are closely related to the proposed zero-shot and few-shot methods proposed in this research. They also serve as strong baselines against which the proposed methods are compared. Finally, a list of benchmarking datasets within the crisis domain and beyond crises are reviewed. They are important resources for testing the effectiveness of the proposed methods in this research as comparing to baselines. Next, this thesis will introduce the details of the proposed methods in crisis domain adaptation (see Chapter 4 and Chapter 5), crisis few-shot

learning (see Chapter 6) and crisis zero-shot learning (see Chapter 7) respectively.

# Chapter 4: Many-to-many Crisis Domain Adaptation

---

## 4.1 Introduction

In real-world crisis message categorisation, labelled data relating to an emerging new event is not readily available. It is costly in terms of both time and human labour to annotate such a labelled data set, in a situation where timeliness is essential to emergency response. When there is a lack of labelled data for a new event, it is difficult to build a computational model that can learn the categorisation task for the event directly in a fully-supervised fashion. This highlights the necessity for domain adaptation to perform crisis message categorisation. Domain adaptation, which is a key aspect of transfer learning, involves learning to perform the same tasks in different domains where a model is trained on labelled data from a source domain plus unlabelled data from a target domain in some cases (check Section 3.1). When applying domain adaptation in the context of crisis message categorisation, it is known as Crisis Domain Adaptation. In that situation, one or more past crisis events are used as the source domain and one or more new emerging events are considered to be the target domain. Assuming that the categorisation task (i.e. the set of labels to be applied) is the same, crisis domain adaptation studies the problem of building a model using the labelled data of past events and adapting the model to perform categorisation on new text instances referring to a new event.

Based on this motivation, this chapter presents a series of studies for many-to-many crisis domain adaptation. It refers to the real-world use case in crisis domain adaptation where multiple source events are used to train a model that can be applied to many target events while the latter refers to the adaptation where one or more source events are used to train a model that is applied to only one target event. It first introduces several approaches for a crisis message categorisation task comprising two sub-tasks: information types classification and priority estimation for many crisis events, which can be viewed as a many-to-many crisis domain adaptation problem (Section 4.2). This includes single-task learning with machine learning and neural networks (Section 4.2.1) as well as a multi-task learning technique with pre-trained lan-

guage models (Section 4.2.2). The former achieves the categorisation task by learning machine learning models and neural networks separately on the two sub-tasks and the latter achieves the categorisation task by learning pre-trained language models jointly on the two sub-tasks.

The work presented in this chapter has previously been presented in published works [137, 140, 142].

## 4.2 Many-to-many adaptation

This research conducts many-to-many crisis domain adaptation according to the timescales and tasks of the Incident Streams (IS) track. The IS track is a research initiative as part of the Text REtrieval Conference (TREC)<sup>1</sup>, which proposes a crisis message categorisation task consisting of two sub-tasks: information type classification and priority estimation. Its aim is for the community to explore more mature social media-based emergency response technology. The track ran annually from 2018 to 2021 inclusive and two editions were run in several of these years. In every edition, a training set and test set containing tweets from many crisis events are available. Figure 4.1 demonstrates the task process as many-to-many crisis domain adaptation. Overall, the TREC-IS dataset (from Section 3.4.2) is the combination of the datasets used for these challenges. Here, unlabelled test tweets from many new events are viewed as the data of target domains and labelled training data from many past events are viewed as the data of source domains. For a participating system, they can develop different methods leveraging the source data and test their methods on the target data by submitting runs to the track. The submitted runs of participating systems are subsequently judged by human assessors using the tailored evaluation metrics as introduced in Section 3.4.2.

Given the characteristics of this track, this research has explored a series of methods for many-to-many crisis domain adaptation by submitting runs to the track between 2019 and 2020. In the 2019 editions of this track, the methods mainly fall into the category of single-task learning with traditional machine learning models such as Logistic Regression and simple neural network models such as LSTM. Having learnt experience from the 2019 editions, the methods evolved to a multi-task learning approach using pre-trained language models such as BERT, leading to state-of-the-art performance in the 2020 editions. This section presents the methods and major findings for crisis message categorisation in the context of many-to-many crisis domain adaptation.

---

<sup>1</sup><https://trec.nist.gov/>

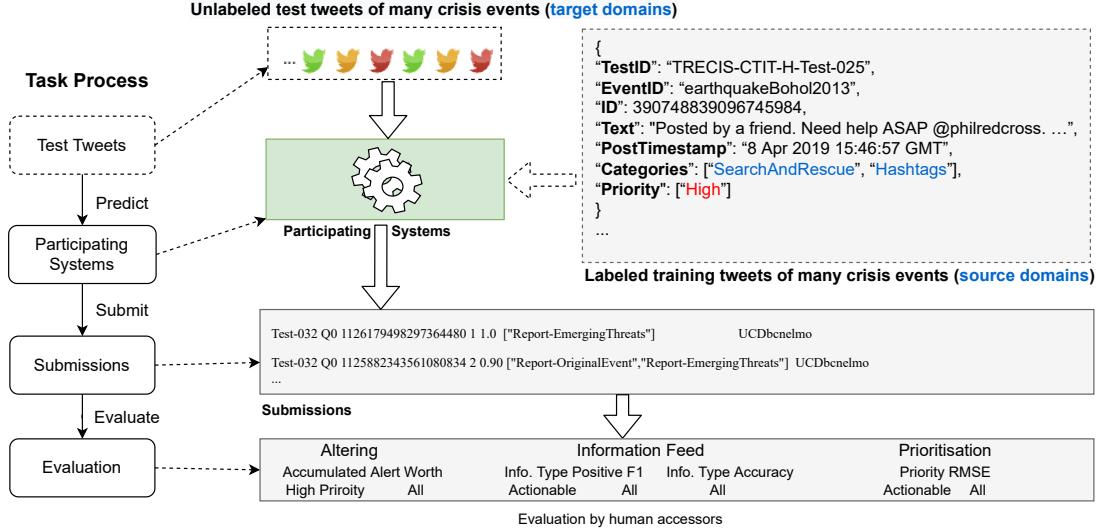


Figure 4.1: The TREC Incident Streams (IS) track as many-to-many adaptation

## 4.2.1 Single-task learning with machine learning and neural networks

The TREC IS track ran twice in 2019 (Edition 2019-A and 2019-B). In both editions, the information type classification and priority estimation sub-tasks are learnt separately by two models. In Edition A, the models mainly consisted of traditional machine learning algorithms with hand-crafted features as input representations. After gaining some experience from the first participation, the participation at Edition B aimed to explore the potential of simple neural networks such as LSTM for the two sub-tasks. In 2019-A, the models used the subset 2018 of the TREC-IS dataset (see Table 3.2 in Section 3.4.2) as the training set from which a 10% portion is sampled as the validation set. In 2019-B, the subset 2018 is used as the training set and the subset 2019A as the validation set.

Figure 4.2 presents the overall system architecture of single-task learning for information type classification and priority estimation. The system began by preprocessing the tweets from a training set. The preprocessing was applied for all runs in both editions for data cleaning and refining. The preprocessing steps included removing URLs and punctuation (like &, @, etc.). In addition, an in-domain out-of-vocabulary dictionary from [50] was used to correct any misspelled words in tweets.

Having been preprocessed, dataset analysis found that the classes of TREC-IS were severely imbalanced (it contains few tweets with more urgent information types and more critical priority levels), therefore data augmentation strategies are applied. After augmentation, the tweets were fed to train two models separately for information type classification and priority estimation. The architecture also allows for different models

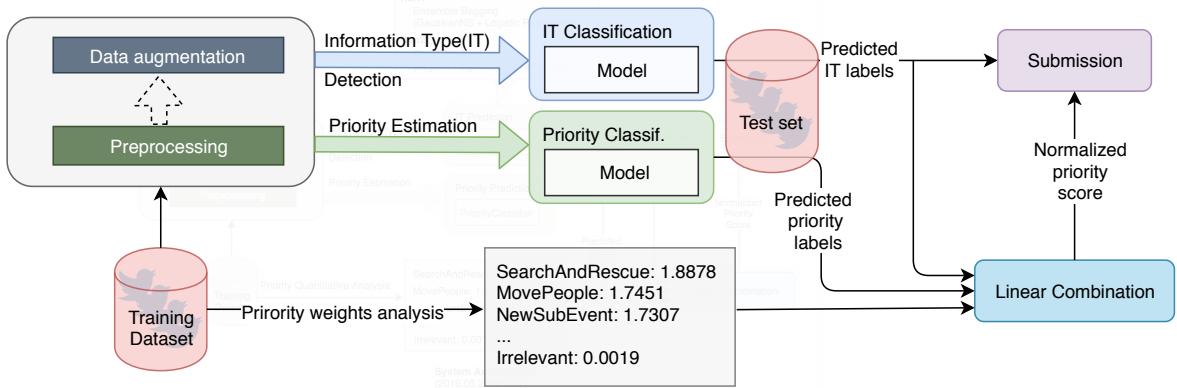


Figure 4.2: Single-task learning for information types classification and priority estimation

to be incorporated (here machine learning models and simple neural network models). Once the models were trained, they were used to make predictions for the unlabelled tweets in the test set. At this stage, each test tweet was predicted with one or more information types and one priority label. Instead of using the model predicted priority, it integrates with the predicted information types via a linear combination, defined as follows:

$$p_i = (1 - \lambda) \times w_i + \lambda \times \delta(\hat{p}_i) \quad (4.1)$$

where, given a tweet  $i$ ,  $p_i$  refers to its final predicted priority score and  $w_i$  refers to the average priority weight based on the predicted information types for the tweet, which is obtained by looking up an information-type-to-weight table as presented in Figure 4.2. The look-up table is constructed by quantitatively analysing all tweets in the training set; the weight for an information type is averaged over the priority scores of the tweets that belong to that information type. In the equation,  $\delta$  is the mapping function that maps the model predicted priority label  $\hat{p}_i$  to the numeric priority score according to the mapping schema: low=0.25, medium=0.5, high=0.75 and critical=1.0. In the formula,  $\lambda$  is a parameter that can be adjusted to give more or less weight to the model predicted priority label and analysed priority weight and it is set to be 0.5 in practice, to give equal contribution. The intuition behind the linear combination is that the information type predictions are used to inform the priority prediction as the information types can implicitly reflect the priority levels such as a request of search and rescue implying a critical priority. Also, priority is quantified to a score between 0 and 1 with this combination.

#### 4.2.1.1 Traditional machine learning as the classifier

In the TREC-IS 2019-A edition, four runs based on different variations of the aforementioned architecture were submitted, namely UCDrunEL1, UCDrunEL2, UCDrunELFB3 and UCDrunELFB4. All the runs were implemented based on machine learning models using feature representations. In all the runs, the up-sampling augmentation technique SMOTE [15] was leveraged for mitigating the imbalanced dataset problem and binary relevance was used for information types (multi-label) classification. In addition, a Linear Regression model was trained for predicting priority levels. The runs differed in how they represented tweets and trained their information type classifiers with different scenarios. Table 4.1 summarises the combinations of four scenarios for the runs. Below are detailed the four scenarios.

	Simple ensemble	Actionable-specific
Word2vec	UCDrunEL1	UCDrunEL2
Feature boosting	UCDrunELFB3	UCDrunELFB4

Table 4.1: Four runs at TREC-IS 2019-A

**Word2vec.** For runs with this scenario, the tweets are represented by an in-domain word2vec-based word embeddings known as crisis word embeddings [50], which are pre-trained on 52 million crisis-related tweets using a Continuous Bag Of Words (CBOW) architecture with negative sampling along with 300 word representation dimensionality. To generate a tweet representation, the bag-of-means strategy was used, which averaged the vectors of all individual words to output a single vector per tweet.

**Feature boosting.** This scenario combined the pre-trained word2vec representing a tweet with 21 other hand-crafted features. These features were extracted based on the commonly-applied practices in the literature of crisis tweet classification [7, 95]. They are based on the content of tweets, and include binary values (e.g. the presence of URLs, hashtags or emoticons), and numeric counts of important words in a tweet (e.g. the count of emotionally-positive or negative words, numbers, instructive words like “donate”, “call”, “search”). The specific features chosen for a tweet were the number of hashtags (numeric), named entity count (numeric), digit count (numeric), number of important crisis-related verbs with reference to the crisis lexicon [101] (e.g. trapped, stuck, move, etc., numeric), sentiment polarity (categorical, -1, 0 or 1), tweet length (word\_length, char\_length, numeric), ratio of uppercase letters (numeric) and retweet check (binary, 0 or 1).

**Simple ensemble.** In this scenario, two classifiers worked together to decide the information types to be assigned to a tweet: a Logistic Regression (LR) classifier and a Naïve Bayes classifier. Prior to deciding which machine learning models to employ

in the ensemble approach, an initial experiment was conducted to evaluate a range of candidates, including Logistic Regression (LR), Naïve Bayes, Decision Tree, Random Forest, SVM, among others. The outcomes indicated that Logistic Regression (LR) and Naïve Bayes exhibited the highest performance among the candidates, and therefore, they were chosen for the ensemble approach. Additionally, the experiment revealed that the combined approach (simple ensemble) outperformed the individual models, which is why the two models were used together to make predictions for information type classification. The information types predicted for a tweet were determined by the averaged predicted probability of the two classifiers where an information type is assigned when the averaged probability is above 0.5.

**Actionable-specific.** Similar to the simple ensemble, this scenario also applied two classifiers to decide the information types to be assigned to a tweet. However, the difference is that the classifiers here were a LR classifier and an actionable classifier. The actionable classifier in the system was a linear Support Vector Machine (SVM) model that was trained only on the training tweets that had actionable types (see Table A.1 in Appendix A.1). Similar to the approach taken in the simple ensemble, an initial experiment was conducted to determine the optimal model for classifying actionable information types. SVM was ultimately selected as it achieved the highest performance among the candidate models. To assign information types for a tweet, the LR classifier first predicted a general list of information types and then the predicted list was forwarded to the actionable classifier so as to add any actionable information types not predicted by the LR classifier.

#### 4.2.1.2 Neural networks as the classifier

In the TREC-IS 2019-B edition, four runs based on different approaches were submitted, namely UCDbaseline, UCDbilstmalpha, UCDbilstmbeta and UCDbcnelmo. Each used a neural network based model (except for UCDbaseline, which reused UCDrunELFB3, consisting of a simple ensemble with feature boosting, with the new data as this run was overall the best in terms of performance among the runs submitted to the 2019-A edition). For the remaining three runs, different models with data augmentation strategies were applied to explore the influence they have on the performance. The following describes the technical variations of each run.

**UCDbilstmalpha.** To address the class imbalance problem, this run applied the text generation strategy GPT-2 [111] for data augmentation. This extracts rare/less-represented tweet samples with respect to information types and priority in the training set and then uses the extracted samples as conditional samples for GPT-2 to generate synthetic samples. Table 4.2 gives some examples of generated samples by GPT-2

given raw critical tweets from the training set as the conditional samples. To represent a tweet, this run used GloVe in the embedding layer and a standard BiLSTM as the classification models. For information type classification, *binary cross entropy* was used as the objective function. For priority level classification, *categorical cross entropy* was used as the objective function.

Conditional raw sample	Generated sample
LANDSLIDE!... road blocked in Costa Rica after M7.6 earthquake	1 car submerged on a hillside and one car in water
More Than 1,000 People Are Now Missing In The California Wildfires	The fires are currently raging, causing extensive damage across vast areas of California
If you are interested in helping on the Earthquake relief	efforts, please call the local offices
Over 350 killed in Kerala floods	Thousands of people were trapped in the city

Table 4.2: Generation of tweets with “critical” priority by GPT-2

**UCDbilstmbeta.** Unlike UCDbilstalpha, this run did not apply data augmentation. Instead, a variant of the *binary cross entropy* from Equation 2.14 is applied  $-w_i[y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})]^2$  where  $w_i$  as a weight factor for handling class imbalance. Unlike in other runs where it is set to be 1, in this run it is  $w_i = \frac{|y_{\max}|}{|y_i|}$  where  $|y_i|$  is the number of samples of information type  $y_i$  appearing in the training data and  $|y_{\max}|$  is the number of samples associated with the most common information type. Like UCDbilstalpha, BiLSTM is used as the classification models for information type classification and priority estimation.

**UCDbcnelmo.** The last run used an attention-based neural network architecture as the classification models, i.e. the Bi-attentive classification (BCN) network by [84]. In [107], BCN was combined with ELMo embeddings to achieve strong performance in sentiment classification. BCN+ELMo applies the contextualised ELMo word representations in the embedding layer of BCN. Based on the promising performance that BCN+ELMo achieved in similar tasks, it was adapted to the crisis domain as the last run. Like UCDbilstalpha, this run also leveraged the GPT-2 based data augmentation strategy for alleviating the class imbalance problem.

#### 4.2.1.3 Evaluation and discussion

In evaluation, all participating groups submit their runs to the IS track whose organisers evaluate the runs of all participating groups uniformly using their proposed metrics (see Section 3.4.2). The results of all the 8 runs described above are reported below. Results in bold are the best in their column. In addition to the presence of scores that the

<sup>2</sup><https://pytorch.org/docs/stable/generated/torch.nn.BCELoss.html>

submitted runs achieved, the median scores of all participating groups for each metric are also included for comparison<sup>3</sup>.

Table 4.3 presents the returned evaluation results for the four runs at TREC-IS 2019-A as well as the median and best scores in each metric from all participating runs. Generally, the runs in most metrics achieved performance above the median. In particular, actionable RMSEs (the lower score the better performance) for the runs (Act. (PErr-H)) are a lot lower than the median and the actionable F1 of UCDrunELFB3 (Act. (CF1-H)) is notably higher than the median (0.1180 vs 0.0459). The weakest aspect of the runs was in the information type (IT) accuracy, with no run above the median (although UCDrunELFB4 was not substantially below median performance). To compare UCDrunELFB3 with UCDrunEL1, and UCDrunELFB4 with UCDrunEL2, it is shown that the use of feature boosting in UCDrunELFB3 and UCDrunELFB4 did not help in AAW but did improve actionable F1 (0.1180 vs 0.0970 and 0.0918 vs 0.0884) and All F1 (0.1827 vs 0.1703 and 0.1668 vs 0.1505). Likewise, if comparing UCDrunEL2 with UCDrunEL1, the use of an actionable classifier in UCDrunEL2 did not help improve information type F1 but did improve both AAW and information type accuracy. In summary, the runs are some of the better performing ones in terms of information type classification, but are somewhat too conservative in terms of alerting. In contrast to a lot of other participating runs, the runs have a better distribution across classes (there are no complete class failures), likely due to the application of SMOTE for mitigating the class imbalance problem.

Runs	Alerting		Information Feed			Prioritisation	
	AAW		IT Positive F1		IT Acc.	RMSE	
	High Pri. (AW-HC)	All (AW-A)	Act. (CF1-H)	All (CF1-A)	All (Cacc)	Act. (PErr-H)	All (PErr-A)
UCDrunEL1	-0.8744	-0.4442	0.0970	0.1703	0.7784	0.1149	0.0617
UCDrunEL2	<b>-0.8556</b>	<b>-0.4382</b>	0.0884	0.1505	0.8324	<b>0.1144</b>	0.0633
UCDrunELFB3	-0.9343	-0.4700	<b>0.1180</b>	<b>0.1827</b>	0.7853	0.1171	<b>0.0603</b>
UCDrunELFB4	-0.9268	-0.4676	0.0918	0.1668	0.8519	0.1207	0.0623
Median	-0.9493	-0.4855	0.0459	0.15995	<b>0.8539</b>	0.1615	0.07545
Best	0.2983	0.2572	0.1695	0.2825	0.9039	0.1132	0.0563

Table 4.3: Evaluation results of four runs at IS 2019-A based on the TREC-IS evaluation metrics described in Section 3.4.2. Best of the four submitted runs in columns are **bolded**.

Table 4.4 presents the returned evaluation results for the four runs at TREC-IS 2019-B. In general, the runs exhibit superior results compared to the median for the majority of metrics. In fact, for certain metrics like information feed F1 and Actionable RMSE, the runs demonstrate the best or nearly the best performance among all the participating runs. Figure 4.3 compares the runs among the top 13 participating runs out of 32 runs in total. All the four runs appear in the top 13 in both information feed and prioritisation. In particular, UCDbaseline achieves the best (better than any other participating runs) performance in finding actionable information types as indicated by the actionable positive F1. Comparatively, UCDbilstmalpha is somewhat weak

---

<sup>3</sup>The overview paper of this track in 2019 [85] includes the full results of all participating groups.

among the four runs. Although it achieves an information type accuracy of 0.86, which is marginally above the median of 0.8583, it does not do well in AAW, which indicates its weakness in correctly finding tweets for alerting. For UCDbilstmbeta, some of its improvements over UCDbilstmbeta are seen. In particular, UCDbilstmbeta achieves good performance in AAW, whose scores (-0.6047 and -0.3332) are increased from the median (-0.9197 and -0.4609) to some extent. It also outperforms UCDbilstmbeta in information type (IT) positive actionable F1. Its performance over UCDbilstmbeta implies that the use of loss weights helps to improve the overall performance. Compared to the other three runs, UCDbcnelmo obtained relatively even good performance across the metrics. It achieved better scores in almost every metric than the median to a good degree except that its information type accuracy is slightly lower than the median.

Runs	Alerting		Information Feed		Prioritisation		
	AAW		IT Positive F1	IT Acc.	Priority RMSE		
	High Pri. (AW-HC)	All (AW-A)	Act. (CF1-H)	All (CF1-H)	All (Caac)	Act. (PErr-H)	All (PErr-A)
UCDbaseline	-0.7856	-0.4131	<b>0.1355</b>	<b>0.2232</b>	0.7495	<b>0.0859</b>	<b>0.0668</b>
UCDbilstmbeta	-0.9287	-0.4677	0.0614	0.171	<b>0.86</b>	0.1521	0.0893
UCDbilstmbeta	<b>-0.6047</b>	<b>-0.3332</b>	0.1269	0.1676	0.8378	0.1004	0.0822
UCDbcnelmo	-0.6961	-0.3624	0.1099	0.1721	0.8452	0.1036	0.0769
Median	-0.9197	-0.4609	0.0386	0.1361	0.8583	0.1767	0.1028
Best	0.2938	-0.1305	0.1355	0.2343	0.8808	0.0788	0.0544

Table 4.4: Evaluation results of four runs at IS 2019-B based on the TREC-IS evaluation metrics described in Section 3.4.2. Best of the four submitted runs in columns are **bolded**.

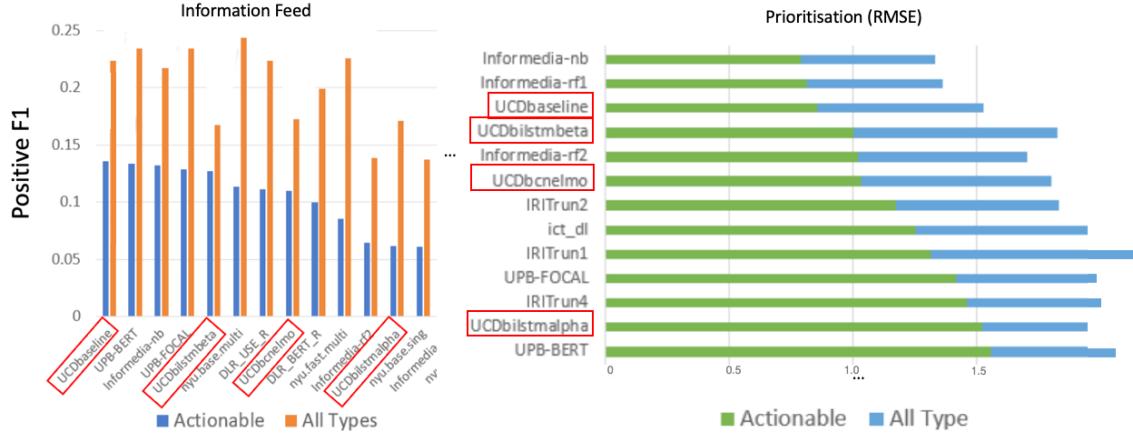


Figure 4.3: The runs at TREC-IS 2019-B among the top 13 participating runs out of 32 runs

To summarise the runs in IS 2019, the runs in Edition B achieved overall better performance than the runs in Edition A as training data increased. It is also interesting to notice that UCDrunELFB3 from Edition A, rerun on Edition B denoted as UCDbaseline, achieved very strong performance in finding actionable information types and estimating priority comparing to other runs. This indicates the benefit of the linear combination (Equation 4.1, where information types are used to inform priority) as well as the simple ensemble strategy with hand-crafted features (multiple machine learning classifiers for information type predictions) applied in the UCDbaseline run. This motivated this research to propose a multi-task learning (MTL) approach followed by a

simple ensemble combining multiple multi-task learners for information types classification and priority estimation in IS 2020 and 2021. For the runs in 2019 Edition B, it is also found that UCDbcnelmo achieves overall good performance, which implies the potential of attention-based neural networks for the IS task. This guided the MTL approach to be based on Transformer-encoder pre-trained language models such as BERT.

## 4.2.2 Multi-task learning with pre-trained language models

After reflecting on the experience from IS 2019, and the results of the runs submitted, improved solutions were sought for IS 2020. Here, this research proposes a multi-task learning (MTL) approach that learns the information type classification task and priority estimation task jointly by fine-tuning Transformer-encoder pre-trained language models such as BERT. Subsequently, an ensemble approach combining multiple such fine-tuned models is used for information type classification and priority estimation. In this section, the MTL approach followed by the ensemble approach is introduced and its effectiveness is discussed in comparison with single task learning approaches and other participating systems in IS 2020 and 2021.

### 4.2.2.1 Method

Figure 4.4 depicts the overview architecture of the MTL approach for the information type classification task and the priority estimation task.

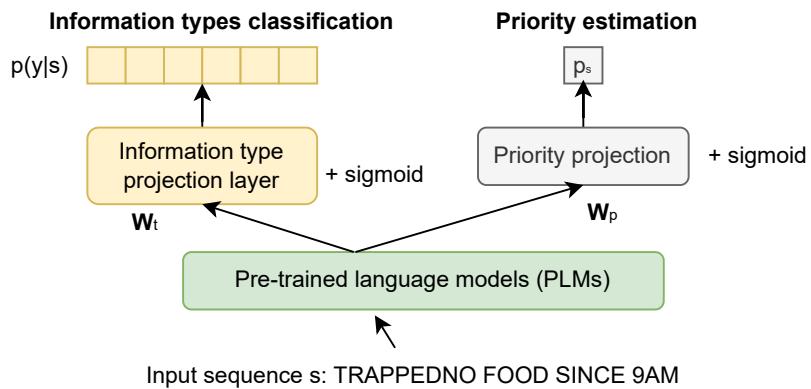


Figure 4.4: Overview architecture of Transformer-based multi-task learning (MTL) approach

**Information type classification:** The objective of information type classification is to predict the probability:  $p(y_j|s)$  referring to the likelihood of an input sequence  $s$

(a tweet message) being assigned to the information type  $y_j$ . Since  $s$  can be assigned to one or more information types, it is taken as a multi-label classification problem, estimated by the following equation.

$$p_{(y|s)} = \sigma(f(s)) \quad (4.2)$$

where  $\sigma$  is the sigmoid function (Equation 2.11) and  $p_{(y|s)}$  is the estimated probability distribution across all information types  $y : \{y_1, \dots, y_j, \dots, y_m\}$  and  $m$  is the number of information types. In the proposed method, an information type  $y_j$  is assigned to  $s$  when  $p_{(y_j|s)} > 0.5$ .

To learn the function  $f(\cdot)$ , a neural network architecture incorporating a pre-trained language model (PLM) and an information type projection layer is deployed. Here, BERT is used as an example of a PLM, although the architecture supports the use of other PLMs as an alternative. BERT represents the sequence first by its output vector at the [CLS] token, denoted by  $\text{PLM}_{\text{CLS}}(s)$  and then this representation is used as the input of the projection layer. The projection process is formulated as follows.

$$f(s) = \text{PLM}_{\text{CLS}}(s) \times \mathbf{W}_t \quad (4.3)$$

Where  $\mathbf{W}_t \in \mathbb{R}^{d_{\text{model}} \times m}$  is the learnable parameters of the projection linear layer and  $d_{\text{model}}$  is the hidden state dimension of the Transformer encoder based PLM (i.e., the dimensionality of the output vector at the [CLS] token).

**Priority estimation:** Similar to information type classification, the priority estimation task is treated as a regression task (due to the dependency between priority levels). In priority estimation, a numeric score  $0 \leq p \leq 1$  is assigned to the input sequence  $s$  to quantify its priority level ( $p_s$ ), which is estimated as follows.

$$p_s = \sigma(g(s)) \quad (4.4)$$

To learn the function  $g(\cdot)$ , a priority projection layer is added to the PLM, which transforms the [CLS] token output vector to the priority score that can be viewed as a numeric scalar, formulated as follows.

$$g(s) = \text{PLM}_{\text{CLS}}(s) \times \mathbf{W}_p \quad (4.5)$$

Where  $\mathbf{W}_p \in \mathbb{R}^{d_{\text{model}} \times 1}$  is the learnable parameters of the priority projection layer and  $d_{\text{model}}$  is the hidden state dimension of the BERT-style Transformer encoder.

**Joint learning:** As mentioned previously, the two tasks share the PLM, i.e., using the same layers of millions of parameters to represent the input sequence  $s$  with the [CLS] output vector (Section 2.2.4.1). The motivation behind this is twofold. Firstly, parameter sharing between multiple tasks is likely to enable one task to share its learnt knowledge with another [165], which is desirable in solving the current problem: information types can inform priority levels and the other around. Secondly, it is faster to make predictions at inference time compared to training two separate models for every task.

To update the parameters of the projection layers as well as the shared parameters of the PLM, the joint loss  $\mathcal{L}_{\text{joint}}$  combining the information type prediction loss  $\mathcal{L}_{\text{it}}$  and priority prediction loss  $\mathcal{L}_{\text{pri}}$  is used as the objective function, defined as follows.

$$\mathcal{L}_{\text{joint}} = \lambda \mathcal{L}_{\text{it}} + (1 - \lambda) \mathcal{L}_{\text{pri}} \quad (4.6)$$

Where  $\lambda$  is a coefficient ( $0 \leq \lambda \leq 1$ ) to adjust the relative weights of  $\mathcal{L}_{\text{it}}$  and  $\mathcal{L}_{\text{pri}}$ . Considering information type prediction to be a multi-label classification problem, *binary cross entropy* is used to define  $\mathcal{L}_{\text{it}}$ . The following formulates the loss computation for one training sequence. For a mini-batch of sequences, the final loss is averaged over the sequences.

$$\mathcal{L}_{\text{it}} = \sum_{y_j \in y} -b(y_j, s) \log(p(y_j|s)) - (1 - b(y_j, s)) \log(1 - p(y_j|s)) \quad (4.7)$$

Where  $b(y_j, s)$  is 0 or 1, indicating if the information type  $y_j$  is assigned to the sequence  $s$  (information type ground truth) and  $p(y_j|s)$  is the probability score for predicting  $s$  to be  $y_j$ . Considering priority estimation to be a regression problem, *mean squared error* is used to define  $\mathcal{L}_{\text{pri}}$  as follows.

$$\mathcal{L}_{\text{pri}} = (m(r_s) - p_s)^2 \quad (4.8)$$

Where  $r_s$  is the ground truth priority of  $s$  and  $m(\cdot)$  is the mapping function converting predicted categorical priority levels ( $p_s$ ) to numeric scores according to the schema:  $\{\text{Critical} : 1.0, \text{High} : 0.75, \text{Medium} : 0.5, \text{Low} : 0.25\}$ . At prediction time, the mapping function is used in reverse to convert the predicted numeric score to a categorical priority level. For example, the priority of  $s$  is predicted to be critical if  $p_s$  lies between

1 and 0.75 inclusive.

#### 4.2.2.2 Experiments and results

Training involves fine-tuning the PLMs to the joint downstream task of information type classification and priority estimation. For hyper-parameter selection, a grid search over learning rate:  $lr \in \{5e-4, 2e-4, 1e-4, 5e-5, 2e-5, 1e-5\}$  and mini-batch size:  $bs \in \{8, 16, 32, 64\}$  is done with the development set. Finally,  $lr$  and  $bs$  are set to be  $5e-5$  and 32 respectively as they perform better empirically. Following the work in a similar domain [79], the Adam optimiser [59] is used to update model parameters and a linear scheduler is applied for dynamically updating the learning rate, with 10% warm-up ratio of the total training steps (12 epochs). Moreover, for the fine-tuning process, the value of  $\lambda$  in Equation 4.6 was set to 0.5, which assigns equal weight to both the information type loss and priority loss. This decision was based on a preliminary study, which discovered that  $\lambda$  of 0.5 yields satisfactory performance in both tasks. To arrive at this conclusion, a grid search was performed over  $\lambda$  values ranging from 0 to 1.0 in increments of 0.1.

Since this approach was proposed in the context of IS 2020 (Edition A), in experiments, the 2020A subset was used as the test set that consists of tweets from 15 crisis events (see Section 3.4.2). For the training and development sets, a data set combining the 2018, 2019A and 2019B subsets was formed first. Then a sampled 10% of the combined set was used as the development set and the remaining 90% as the training set. In evaluation, in addition to the tailored metrics (Section 3.4.2), a harmonic mean metric (**HarM**) was added to the list as an indicator of overall performance across all metrics. In calculating this harmonic mean, AW-HC and AW-A were first normalised to lie between 0 and 1.

In this work, three experiments were conducted to test the effectiveness of the proposed approach in different aspects. The first experiment (Ex1) was conducted to compare it to single-task learning baselines. The second experiment (Ex2) was conducted to explore the use of different PLMs with the approach, and examine their performance both individually and in ensembles. Finally, the approach was tested in the editions of IS 2020 (Ex3). The following presents each experiment along with the discussion of results.

##### Ex1: Single task learning baselines

In the first experiment to explore the difference between the multi-task learning approach and single task learning approach, `BERT_base`<sup>4</sup> is used as a baseline to fine-tune

---

<sup>4</sup><https://huggingface.co/bert-base-uncased>

two models independently for the information type classification and priority estimation tasks in a single task learning (STL) scenario. Specifically, the loss functions of Equation 4.7 and Equation 4.8 are used separately to train two single task models. In contrast, the proposed multi-task learning (MTL) scenario uses the joint loss function of Equation 4.6. In addition to the Transformer-based deep learning baseline, traditional machine learning (ML) algorithms such as Logistic Regression (LR) are considered to be strong baselines in this problem domain, as evidenced in previous work [87, 137]. Hence, another baseline that trains two separate LR-based classifiers for the two single tasks is implemented in the experiment also. To make this baseline as strong as possible, the development set is used to set up the hyper-parameters of the LR classifiers (using grid search) including  $C \in \{0.01, 0.1, 1, 1.0, 10, 100\}$ ,  $ngram\_range \in \{(1), (1,2), (1,3)\}$  and  $weighting \in \{count, TF-IDF\}$ . Following empirical study, the setups ultimately used  $C=10$ ,  $ngram\_range=(1,2)$  and  $weighting=TF-IDF$ .

The performance of the BERT-based and LR-based STL runs compared to the BERT-based MTL run is reported in Table 4.5. It shows that the BERT-based runs perform better than the LR runs except for a marginal decrease in Cacc score. More importantly, the MTL run achieves substantial improvement in NDCG and AW scores over the BERT-based STL run. For example, the BERT\_base + MTL run achieves the best scores in NDCG, AW-HC, AW-A and CF1-H. Although it can be seen that the BERT-based STL runs perform the best in prioritisation, this is at the cost of a loss of NDCG and AW performance. As a whole, the MTL scenario gains an advantage over the STL scenario not only in the overall effectiveness (as illustrated by the HarM score in the last column) but also in avoiding the need to train separate models for separate tasks.

	Priority estimation and ranking						Info. types classification		
	NDCG	AW-H	AW-A	PErr-H	PErr-A	CF1-H	CF1-A	Cacc	HarM
LR + STL	0.4495	-0.4856	-0.2627	0.1718	0.2216	0.0898	0.1527	<b>0.9113</b>	0.2109
BERT_base + STL	0.4393	-0.4057	-0.2148	<b>0.2402*</b>	<b>0.2758*</b>	0.1084	<b>0.1801</b>	0.8960	0.2510
BERT_base + MTL	<b>0.5101</b>	<b>-0.2689*</b>	<b>-0.1569*</b>	0.1923	0.2544	<b>0.1382*</b>	0.1638	0.8937	<b>0.2609</b>

Table 4.5: Comparison between single task learning (STL) with Logistic Regression (LR) and BERT and the multi-task learning (MTL). The numbers in bold represent the highest performance in each column and those annotated with \* indicates that the highest is “confident” compared to the next-highest in its column (Wilson Score Interval [149],  $p < 0.05$ ). The difference between  $c_1$  and  $c_2$  is described as “confident” if their confidence intervals do not overlap.

## Ex2: Transformer selection and ensemble

In the architecture of the proposed approach (Figure 4.4), one important component is the selection of Transformer encoder based PLMs. In the “Method” section above, BERT is used as an example of a PLM. However, since the introduction of BERT, the literature has seen many BERT variants being developed (Section 2.2.4.1). Variants like DistilBERT [118], ALBERT [118], and ELECTRA [19] have been developed to optimise various aspects of BERT such as memory consumption, computation cost or

pre-training representation learning. Studies have shown their promising performance through fine-tuning in downstream tasks such as text classification and reading comprehension. To examine their capabilities in the crisis domain, they are used along with BERT in the experiment.

The upper block of Table 4.6 presents the results of using different PLMs individually with the MTL approach, whereas the lower block reports the results of creating ensembles combining the predictions of multiple PLMs. The number appended to the individual runs refers to the trained model size and the number appended to the ensemble runs indicates the combination of corresponding individual run indices. As this shows, the BERT run has the same model size as the ELECTRA. However, there is no significant difference in performance between the runs when evaluated using the HarM score. Although BERT outperforms ELECTRA in AW, it loses this advantage in prioritisation. In comparison, DistilBERT and ALBERT are relatively smaller in size than BERT and ELECTRA. They still perform well overall, only slightly worse than the BERT and ELECTRA runs, which illustrates their effectiveness with much reduced model sizes (ALBERT in particular).

Run variants	Priority estimation and ranking					Info. types classification			
	NDCG	AW-HC	AW-A	PErr-H	PErr-A	CF1-H	CF1-A	Cacc	HarM
<i>Individual Transformer encoder based PLMs</i>									
1. BERT_base (110M)	0.5101	-0.2689	-0.1569	0.1923	0.2544	0.1382	0.1638	0.8937	0.2609
2. DistilBERT_base (66M)	0.4808	-0.4533	-0.2382	0.9004	0.1191	0.1376	0.1830	0.2110	0.2264
3. ELECTRA_base (110M)	0.5042	-0.4011	-0.2122	0.2059	0.2801	0.1514	0.1742	0.8958	0.2689
4. ALBERT_base-v2 (11M)	0.4669	-0.4118	-0.2190	0.1900	0.2720	0.0568	0.1707	<b>0.9087</b>	0.1923
<i>Ensemble runs</i>									
EnsembleA (1+3)	<b>0.5207</b>	-0.2274	-0.1406	0.1999	0.2560	0.1738	0.1796	0.8722	0.2836
EnsembleB (2+4)	0.4848	-0.3212	-0.1823	0.2081	0.2728	0.1407	0.2041	0.8844	0.2752
EnsembleC (1+2+3)	0.5206	-0.1982	-0.1282	0.2023	0.2589	<b>0.1819</b>	0.1909	0.8621	0.2919
EnsembleD (1+2+3+4)	0.5176	<b>-0.1613*</b>	<b>-0.1148</b>	<b>0.2594*</b>	<b>0.2966</b>	0.1754	<b>0.2084</b>	0.8545	<b>0.3141*</b>

Table 4.6: Individual runs with different PLMs and ensemble runs that leverage the individual runs jointly making predictions for priority and information types.

Of the individual runs, each scores the highest performance in at least one metric, with none showing a significant overall performance improvement over the others. In order to exploit the power of these individual runs, this work proposes a simple ensemble approach that combines the individual runs to jointly make predictions for information types and priority. It is based on the hypothesis that such an ensemble may leverage the distinct benefits of these diverse PLMs to achieve greater overall performance across both tasks. The ensemble approach is described as follows.

**The ensemble approach:** Given a set of individual multi-task learners,  $\{l_1, l_2, \dots, l_n\}$ , the final priority prediction for a tweet is made from the priority predictions by  $\{l_1, l_2, \dots, l_n\}$  according to a priority decision strategy, denoted by  $P_{ds}$ . Three options are set up to evaluate  $P_{ds}$ :  $\{\text{Highest}, \text{Average}, \text{Lowest}\}$ . *Highest* refers to always selecting the highest priority level given by any of the individual predictors. *Lowest*

represents the opposite strategy. The *Average* scenario means taking an average score over all priority predictions in the union and then the final priority prediction is assigned based on this average score. The conversion between priority numeric score and level is applied via the mapping function, namely  $m(\cdot)$  as introduced in Equation 4.8. Regarding information types, the final prediction for each tweet is made from the information type predictions by  $\{l_1, l_2, \dots, l_n\}$  according to an information type decision strategy, denoted by  $I_{ds}$ . Two options are available to determine  $I_{ds}$ :  $\{\text{Union}, \text{Intersection}\}$ . *Union* and *Intersection* refers to always selecting the union and intersection respectively of information types assigned by the individual predictors.

	Priority estimation and ranking					Info. types classification			
	<b>NDCG</b>	<b>AW-H</b>	<b>AW-A</b>	<b>PErr-H</b>	<b>PErr-A</b>	<b>CF1-H</b>	<b>CF1-A</b>	<b>Cacc</b>	<b>HarM</b>
Union-Highest	0.5170	-0.1613	-0.1148	0.2594	0.2966	0.1754	0.2084	0.8545	0.3140
Union-Average	0.5066	-0.2489	-0.1491	0.2475	0.274	0.1754	0.2084	0.8545	0.3036
Union-Lowest	0.4896	-0.5824	-0.2932	0.1302	0.2102	0.1754	0.2084	0.8545	0.2369
Intersection-Highest	0.5178	-0.1613	-0.1148	0.2342	0.2715	0.0303	0.1105	0.9291	0.1387
Intersection-Average	0.5061	-0.2489	-0.1491	0.2184	0.2485	0.0303	0.1105	0.9291	0.1362
Intersection-Lowest	0.4888	-0.5824	-0.2932	0.1321	0.2215	0.0303	0.1105	0.9291	0.1233

Table 4.7: Evaluation results of the EnsembleD run with varying strategies for merging information types and priority levels. Each row is named as  $x - y$  where  $x$  is the information type strategy, i.e.  $I_{ds}$  and  $y$  is the priority level strategy, i.e.  $P_{ds}$ .

In choosing  $P_{ds}$  and  $I_{ds}$ , the ensemble run combining all four individual runs from the top of Table 4.6, namely EnsembleD, is tested with different  $P_{ds}$  and  $I_{ds}$  and the results are reported in Table 4.7. The results show that the *Union* runs substantially outperform the *Intersection* runs in information feed categorisation while yielding the same scores in the remaining metrics. For  $P_{ds}$ , it shows an increased performance in ranking, alert worth, and prioritisation as it changes from *lowest* to *Highest*. Based on the results, in the subsequent experiments,  $I_{ds}$  is set to be *Union* and  $P_{ds}$  is set to be *Highest* as this combination gives the best performance across the metrics.

With this setup, the experiment is then conducted to test the ensemble approach with different sets of  $\{l_1, l_2, \dots, l_n\}$  and the lower block of Table 4.6 demonstrates the results. The EnsembleA run combines the two relatively large models, BERT and ELECTRA, while EnsembleB combines DistilBERT and ALBERT. It shows that each ensemble has overall performance superior to its component models. This indicates that the ensemble approach adds benefit to the performance by leveraging the predictions of individual runs. The best-performing run among all experimental runs so far reported in this work is EnsembleD, with the HarM score reaching 0.3141 as well as achieving strong scores in individual metrics except for in Cacc. However, It is analysed that Cacc is arguably the least important metric due to the heavy imbalance in the TREC-IS information types and the usual problems with accuracy-based metrics in such a scenario. A naïve run that predicts all tweets to have all information types will achieve a Cacc score of

approximately 0.94, while being useless in practical terms and performing extremely poorly in the other metrics. To examine the performance of the ensemble runs in separate tasks, it is found that the ensemble runs outperform the single model runs in both the priority estimation and information type classification tasks. Although the overall performance increases as more individual models combined, the experiment found that the increase becomes marginal when more models than in EnsembleD are combined.

### **Ex3: Results at TREC-IS track**

The MTL-based runs have been found to have an advantage over BERT-based and LR-based STL baselines. A simple ensemble technique has also been proposed, which combines the individual MTL runs to make predictions for information types and priority levels. This ensemble technique improves performance compared to the individual runs. In this section, the performance of this approach is compared to state-of-the-art methods by comparing it to participating runs in the TREC IS track of 2020 (Edition A and B) and 2021 (Edition A and B).

In IS 2020 both Edition A and B, the track proposed two tasks: Task 1 and Task 2. The only difference between Task 1<sup>5</sup> and Task 2 is that Task 2 uses a reduced set of 12 information type labels, which includes 11 important information types<sup>6</sup> from the 25 used in Task 1 (see Table A.1) with the remaining 14 combined into the single category “Other-Any”. Thus Task 2 emphasises a run’s performance in identifying the information types that are most closely related to emergency response. Due to the common features shared between the two tasks, any runs submitted to Task 1 are also evaluated in Task 2.

In the 2020-A edition of TREC-IS, several runs were submitted, one of which was an MTL-based run called `Our_Run1` that is similar to the `BERT_base+MTL` run<sup>7</sup>. Figure 4.5 plots the returned results of `Our_Run1` and the top participating runs, which are evaluated for both Task 1 and Task 2. The performance of `EnsembleD` is also included for comparison to the submitted runs. Although `EnsembleD` was not officially submitted, it was subsequently evaluated using the official evaluation script provided by IS. The plotted results present the performance of the top participating runs from the perspective of four aspects: Ranking, Alert Worth, Information Feed Categorisation and Prioritisation. To view the full results of all participating runs, the reader is encouraged to refer to the track’s overview paper [87].

---

<sup>5</sup>The results reported so far are from Task 1.

<sup>6</sup>The information types include the 6 actionable information types in Table A.1 plus Request-InformationWanted, CallToAction-Volunteer, Report-FirstPartyObservation, Report-Location and Report-MultimediaShare.

<sup>7</sup>The difference is that `Our_Run1` is trained on the entire training set without first removing the development set used in the `BERT_base+MTL`.

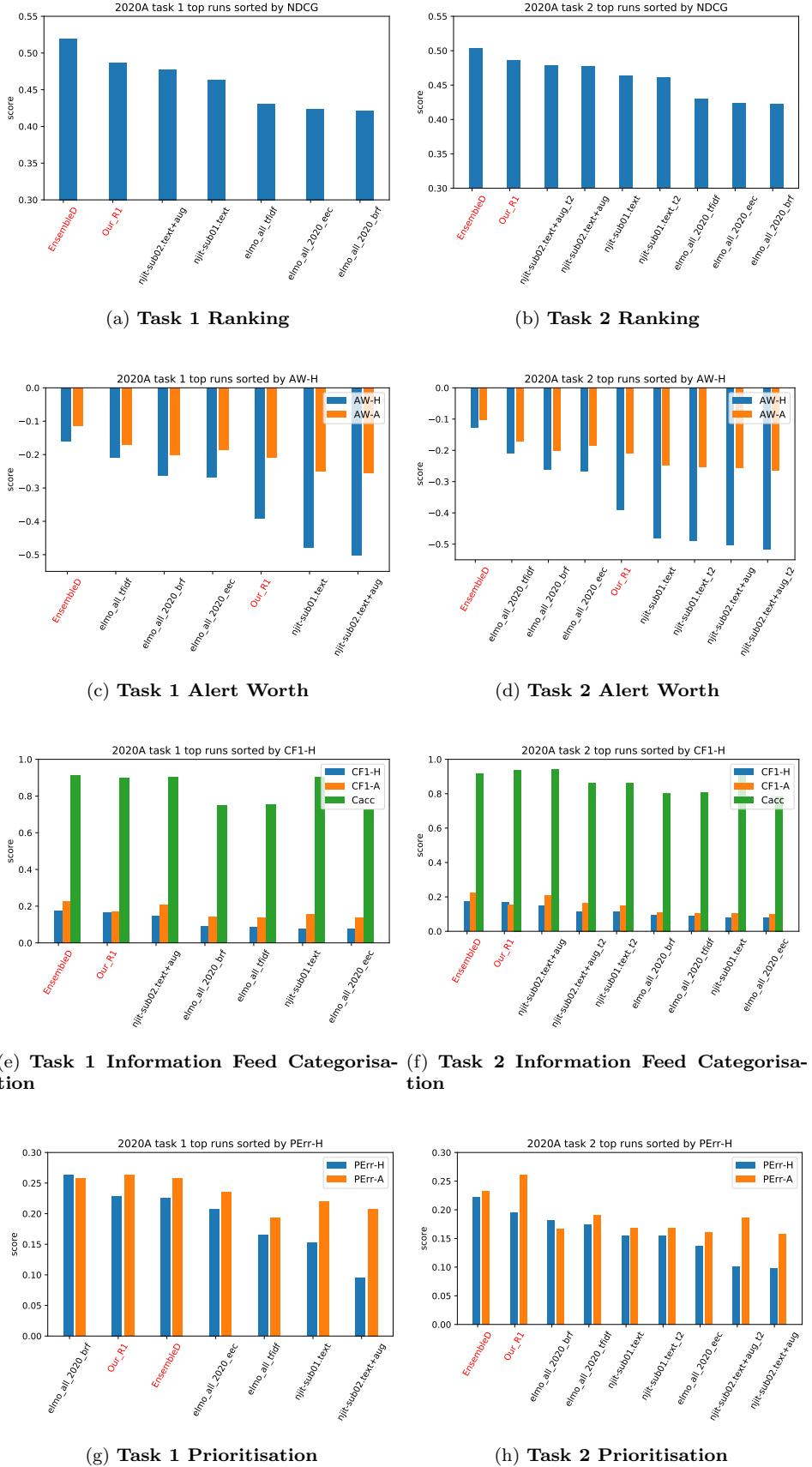


Figure 4.5: Performance comparison between TREC-IS 2020A top participating runs for both task 1 and 2. The runs based on the proposed MTL approach are highlighted in red.

When examining the participating runs, it appears that they frequently achieve high scores in some metrics at the cost of lower scores in others. For example, the participating runs tagging with “elmo” relatively outperform `Our_Run1` in Alert Worth (Figure 4.5c and 4.5d) but fall far behind in Ranking and Information Feed Categorisation. In contrast, the participating runs tagged with “sub” achieve good scores that are near to `Our_Run1` in Information Feed Categorisation (Figure 4.5e and 4.5f) but not in the Alert Worth metrics. Despite the loss in Alert Worth to the `elmo` runs, `Our_run1` outperforms the top participating runs in the rest of metrics for both Task 1 and Task 2 with the only exception of a marginal loss to one `elmo` run in Task 1 Prioritisation (Figure 4.5g). `Our_run1` also achieves the highest HarM score, which implies overall best performance. However, there remains a strong argument that in practical terms different submitted runs are preferable in different situations, depending on the needs of the emergency responders, and that no overall best-performing system has been satisfactorily identified.

The `EnsembleD` run, however, achieves state-of-the-art performance in almost every metric, substantially outperforming the participating runs in most cases. Figure 4.5 indicates that there are only two evaluation figures (other than the less important Cacc discussed previously) where `EnsembleD` does not have the highest performance, with the difference being minor in both cases (the CF1-A from Figure 4.5e and PErr-H from 4.5g).

Given the tendency of participating runs to achieve imbalanced performance across the individual metrics, the `EnsembleD` stands out as a good choice with regards to its effectiveness in different aspects of emergency response, across the range of metrics.

Having established the effectiveness of the approach in IS 2020A, it is further tested in further editions of the IS track. Table 4.8 and 4.9 present the results of participating runs at IS 2020 Edition B Task 1 and 2 respectively, where `EnsemblD` (named as “ucd-run1” in the official submission of this edition) is compared to other participating runs. As can be seen from Table 4.8, in Task 1, the MTL-based ensemble run substantially outperforms other participating runs in both information type classification and prioritisation<sup>8</sup>. In particular, the run is effective in classifying actionable information types. For example, it achieves the top actionable information type F1 score of 0.3215 (CF1-H) and the best actionable priority F1 of 0.2009 (PErr-A). This is further evidenced by the run’s performance in Task 2, as shown in Table 4.9, where the run outperforms the participating runs in every metric. To further examine the run’s performance at the level of individual information types, the information type F1 scores and priority

---

<sup>8</sup>The exception is accuracy, where only a small difference is observed across the participating runs: the MTL-based ensemble run is substantially higher than other participating runs in the remaining metrics.

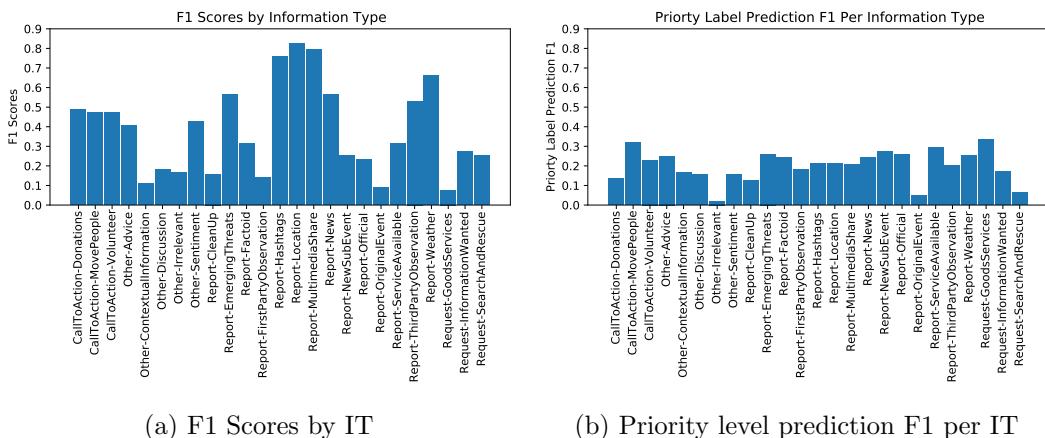
Run	nDCG	CF1-H	CF1-A	Caac	PErr-H	PErr-A
Participating runs						
BJUT-run	0.4346	0.0266	0.0581	0.8321	0.1744	0.0905
njit.s1.aug	0.4480	0.2634	0.3103	<b>0.8655</b>	0.2029	0.1518
njit.s2.cmmnd.t1	0.4475	0.1879	0.2223	0.8475	0.2029	0.1518
njit.s3.img.t1	0.4222	0.1879	0.2223	0.8475	0.1959	0.1417
njit.s4.cml.t1	0.4164	0.1712	0.1465	0.8445	0.1054	0.1064
ufmg-sars-test	0.3634	0.0001	0.0493	0.8337	0.1285	0.1378
EnsembleD (ucd-run1)	<b>0.5033</b>	<b>0.3215</b>	<b>0.3810</b>	0.8520	<b>0.2582</b>	<b>0.2009</b>

Table 4.8: Evaluation results of participating runs at TREC-IS 2020-B Task 1

Run	nDCG	CF1-H	Caac	PErr-A
Participating runs				
BJUT-run	0.4350	0.0472	0.7977	0.1337
njit.s1.aug	0.4487	0.3480	0.8846	0.1838
njit.s2.cmmnd.t1	0.4467	0.2494	0.8612	0.1838
njit.s3.img.t1	0.4215	0.2494	0.8612	0.1708
njit.s4.cml.t1	0.4176	0.1278	0.8360	0.1162
ufmg-sars-test	0.3630	0.0127	0.8419	0.1480
EnsembleD (ucd-run1)	<b>0.5020</b>	<b>0.4036</b>	<b>0.8913</b>	<b>0.2320</b>

Table 4.9: Evaluation results of participating runs at TREC-IS 2020-B Task 2

F1 scores per information type of the run are reported in Figure 4.6. Figure 4.6a shows that the run performs well in categorising some actionable information types, such as “CallToAction-MovePeople” and “Report-EmergingThreats” but performs less well in other actionable information types such as “Request-GoodsService”, as compared to the non-actionable information types. However, examining the priority F1s per information type in Figure 4.6b, it can be seen that the run performs relatively better in priority level prediction for actionable information types than non-actionable information types, where “CallToAction-MovePeople”, “Request-GoodsService” and “Report-ServiceAvailable” are the top 3 that the run achieves in priority F1.



(a) F1 Scores by IT

(b) Priority level prediction F1 per IT

Figure 4.6: Performance visualisation by information types (ITs) of **ucd-run1** in IS 2020B Task 1.

	<b>nDCG</b>	<b>CF1-H</b>	<b>CF1-A</b>	<b>Caac</b>	<b>PErr-H</b>	<b>PErr-A</b>
run1	<b>0.6115</b>	0.215	0.2951	0.8837	0.3032	0.3068
mtl.ens.new	0.5907	0.2579	<b>0.3211</b>	0.8646	0.3052	0.3125
med	0.5695	0.206	0.2823	0.8827	0.2113	0.2175
max	<b>0.6115</b>	<b>0.2815</b>	<b>0.3211</b>	<b>0.8902</b>	<b>0.306</b>	<b>0.3211</b>

Table 4.10: The performance of the submitted runs at IS 2021A where the med and max rows present the median and maximum scores of each metric respectively across all participating runs.

	<b>nDCG</b>	<b>CF1-H</b>	<b>CF1-A</b>	<b>Caac</b>	<b>PErr-H</b>	<b>PErr-A</b>
run1	0.4499	0.2177	0.247	0.8966	0.2376	0.2566
mtl.ens.new	0.4555	<b>0.251</b>	<b>0.2623</b>	0.8753	0.2783	0.2703
med	0.4272	0.1842	0.233	0.8947	0.2107	0.2031
max	<b>0.4791</b>	<b>0.251</b>	<b>0.2623</b>	<b>0.9067</b>	<b>0.2798</b>	<b>0.2756</b>

Table 4.11: The performance of the submitted runs at IS 2021B where the med and max rows present the median and maximum scores of each metric respectively across all participating runs.

In the last year of the TREC IS track, 2021, it changed from two tasks to only Task 1. In IS 2021, the MTL and ensemble methods with some variations are further tested over two editions of the track: 2021A and 2021B. Table 4.10 and 4.11 show the performance of the submitted runs compared to the median and maximum scores across all participating runs<sup>9</sup>. The `run1` is simply the re-run version of the MTL method and `mtl.ens.new` is the re-run version of the MTL-based ensemble method. The difference of the two runs compared to `BERT_base+MTL` and `EnsembleD` is that they use an optimised BERT variant DeBERTa [42] instead of the original BERT [25]. The performance of the two runs was consistent with previous editions, achieving the best or near-best scores in both the information type classification and priority estimation tasks. This offers strong evidence of the effectiveness of the MTL approach for crisis message categorisation in the context of many-to-many domain adaptation.

#### 4.2.2.3 Error analysis

Even if EnsembleD and MTL-based runs perform well in many ways, it is still important to understand the many kinds of faults the system makes. Using the top EnsembleD run on the information type classification task, a qualitative error analysis was carried out to offer insights and aid the community in resolving issues in this area. Examples of incorrect information type predictions made by EnsembleD are shown in Table 4.12 where “IT prediction” denotes the set of information types assigned by EnsembleD and “IT ground truth” lists the information types assigned by human assessors as part of the IS track.

---

<sup>9</sup>Full results can be found in the overview paper [13].

Id	Event	Text	IT prediction	IT ground truth
# 1	gilroygarlicShooting2020	6-year-old killed in Gilroy Report-Factoid, Garlic Festival Shooting Report-Location, <a href="https://t.co/MYgvoneYdCReport">https://t.co/MYgvoneYdCReport</a>	Report-Factoid, Report-Location, Report-MultimediaShare, Report-News, Report-ThirdPartyObservation	Report-Location, Report-MultimediaShare, Report-News
# 2	gilroygarlicShooting2020	Volunteers from Muslim-faith based charity @pennyappeal have turned up at Chapel school with loads of supplies for people evacuated from #Whaley-Bridge. Good on ya, lads. <a href="https://t.co/eBMM3evPAL">https://t.co/eBMM3evPAL</a>	CallToAction-Donations, Report-Hastags, Report-Location, Report-Hashtags, Report-Location, Report-MultimediaShare, Report-ServiceAvailable, Report-ThirdPartyObservation	Report-Hashtags, Report-Location, Report-Hashtags, Report-Location, Report-MultimediaShare, Report-ServiceAvailable, Report-ThirdPartyObservation
# 3	hurricaneBarry2020	Rain water at Mississippi and Santa fe #Denver @9NEWS <a href="https://t.co/a8eekFIODx">https://t.co/a8eekFIODx</a>	Report-Hashtags, Report-Location, Report-Hashtags, Report-Location, Report-MultimediaShare, Report-News	Other-Irrelevant
# 4	baltimoreFlashFlood2020	I'm at Bronycon 2019 in Baltimore, MD <a href="https://t.co/oVXxmZ4JID">https://t.co/oVXxmZ4JID</a>	Other-Irrelevant	Request-SearchAndRescue

Table 4.12: Examples of error analysis for information types

The first two examples show how the information type “Report-ThirdPartyObservatio” is handled. EnsembleD chose this information type in the first example, but it was not present in the ground truth information types. In the second example, EnsembleD did not select this information type, but the ground truth did. It is debatable whether all tweets containing crisis information should be labelled as first-party or third-party observations. That said, the image associated with #2 suggests that this may be a first-party observation, with the author of the tweet posting an image of the volunteers mentioned. It’s also worth noting that EnsembleD chose the “CallToAction-Donation” information type. Although this tweet does not explicitly call for action, the Twitter account referenced (@pennyappeal) is a charity appeal for crisis situations. Both of these observations indicate that context and subtlety make classification of information types difficult.

In example #3, the geographical context is critical. The human assessor deemed it irrelevant because it relates to Denver, Colorado, whereas Hurricane Barry’s main effects were felt primarily in Louisiana. The image attached depicts flood water during the same time period as the hurricane in question. As a result, many of the information types chosen by EnsembleD would have been correct if it had been within the geographical area in question. This example also highlights another challenge in terms of geographical context. Without context, “Mississippi” could refer to a US state or a river, and “Santa Fe” is the capital city of the US state of New Mexico. However, in

this particular example, the names refer to street names in Denver, Colorado, where the image of floodwater was taken. Although this was not the reason for this particular tweet’s misclassification, it does point to a new challenge, implying that language models alone will not be sufficient to solve this problem, and that the incorporation of knowledge maps and other ontologies may be required to provide the necessary context to information type classification systems.

The last example appears to merely illustrate human inaccuracy in the classification of information types. This tweet refers to the user’s attendance at a conference that was over two days before the flash flood being discussed. This highlights the drawbacks of using human assessors as comparison sources. Personal preference, external context, and obvious mistakes make it crucial to analyse systems qualitatively in addition to presenting evaluation metrics.

## 4.3 Conclusions

In this chapter, the research in many-to-many domain adaptation was reported, which was based on the TREC IS track that proposed an information type task and a priority estimation task for categorising crisis-related tweets from many unseen events. At the early stage of this track, the proposed methods were based on single-task learning (STL), which trained machine learning models or simple neural network models separately for the two tasks. Having realised the limited effectiveness of these methods, this research took a further step to explore better solutions as the track evolved. Ultimately, a multi-task learning (MTL) approach that trained a Transformer encoder-based pre-trained language model (PLM) jointly for the two tasks was proposed. By testing this approach in many subsequent IS editions, it was found to achieve strong performance compared to other participating runs. In particular, a simple ensemble technique that leveraged multiple multi-task learners was found to achieve the overall best performance among participating runs. In addition, a qualitative analysis of the predictions made by the MTL-based ensemble system revealed that language models alone may not be sufficient for categorising crisis tweets, as subtle factors such as implicit mentions of charity agencies or ambiguous place names can make the system challenging.

So far, the methods have been developed to build models trained on the entire corpus of training tweets from various distinct events and test them on the test tweets of many new events, thus corresponding to many-to-many crisis domain adaptation. This adaptation work can be useful in addressing the needs of users who want to apply

computational techniques for crisis message categorisation without needing to know the specific types of source and target events. However, in reality, it is also of interest to determine how to select source events for a new event based on their characteristics, specifically the one-to-one or many-to-one crisis domain adaptation problems. In the next chapter, the work of using event-aware sequence-to-sequence PLMs for this purpose of adaptation is presented.

# Chapter 5: Many-to-one and one-to-one Crisis Domain Adaptations

---

## 5.1 Introduction

The work into this many-to-many adaptation helps to meet the user needs when building a computational model for crisis message categorisation without needing to know the specific types of the source and target events (Chapter 4). However, in reality, given a choice of source events, it is also interesting to know how to select appropriate source events for a new event based on their characteristics: the one-to-one and many-to-one adaptation problems. For this purpose, this chapter introduces CAST, a method that utilises pre-trained sequence-to-sequence language models for crisis domain adaptation. The details of CAST are presented in Section 5.2, which includes the methodology in Section 5.2.1, the experimental setup in Section 5.2.2, and discussions of its effectiveness in both many-to-one and one-to-one adaptation settings in Section 5.2.3.

The work presented in this chapter has previously been presented in published work [141].

## 5.2 Many-to-one and one-to-one adaptations

This section presents work using sequence-to-sequence (seq2seq) PLMs for one-to-one and many-to-one crisis domain adaptations, which is abbreviated to CAST. Firstly, this work aims to test the effectiveness of CAST by comparing it to existing adaptation approaches. Secondly, referring to a crisis event as a domain, another important objective of this work is to explore the selection of source domains for adaptation to a target domain, which offers insights on how to efficiently use source data for model training.

In comparison to related work for crisis domain adaptation [2, 71, 79] (Section 3.1), CAST uses only labelled source data without any unlabelled target data, i.e., it is

target data independent. This makes CAST more suited to real-world use cases. This is because a crisis usually focuses on a specific aid-related topic at a certain stage. For example, an earthquake is normally more about “Emerging Threats” than “Donation” at early stages. Hence, when a good-quality representation of the target event distribution is needed, the unlabelled target data needs to be collected as the target crisis unfolds, before training and classification can take place. Instead of using target data, CAST fine-tunes a seq2seq PLM for crisis domain adaptation by adding a task description and an event description to the input sequences that belong to the event, which makes the model aware of the event that the input sequences relate to. The details of CAST are described as follows.

### 5.2.1 Event-aware sequence-to-sequence pre-trained language models

In this work, a Transformer decoder-encoder based model T5 [113] is used as an example of a seq2seq PLM. Unlike the standard way of fine-tuning a seq2seq model for a downstream task (Section 2.2.4.3), CAST is simple and trained by using a task description and/or an event description added to the input sequence (i.e., a crisis-related message), as illustrated in Figure 5.1. The task description essentially refers to a question-form natural language text specifying the target task and the event description refers to basic information of a crisis event including the event name and its occurrence location. Taking an input crisis message from 2015 Nepal earthquake in the task of informativeness classification as an example, the task description and the event description can be formed as “Is this message informative to 2015 Nepal earthquake?”.

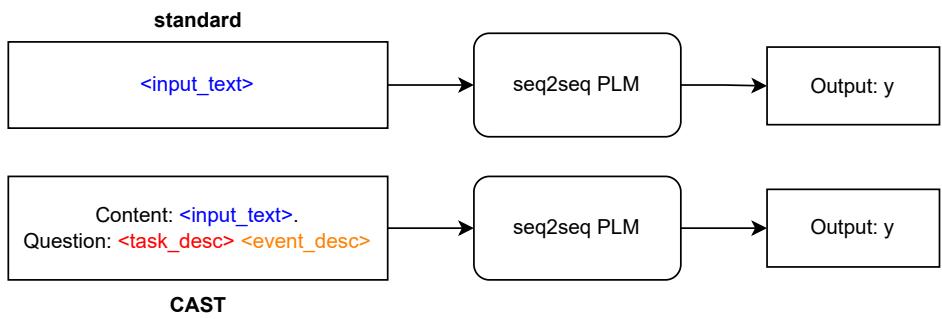


Figure 5.1: The standard and CAST approaches for crisis domain adaptation leveraging seq2seq PLMs.

Mathematically, in the standard way of fine-tuning a seq2seq model for a downstream task, the input sequence  $s: \{t_1, t_2, \dots, t_T\}$  consists solely of the textual content itself. CAST is specifically proposed for crisis domain adaptation, taking into account

both a task description  $T_{desc}$  and an event description  $E_{desc}$ , leading to the new input  $\hat{s}: \{\hat{t}_1, \hat{t}_2, \dots, \hat{t}_{T'}\}$ , formulated as follows.

$$\hat{s}_{1:T'} = \zeta_{\oplus}(s_{1:T}, T_{desc}, E_{desc}) \quad (5.1)$$

Where  $\zeta_{\oplus}$  is the reconstruction function that concatenates  $s$  with  $T_{desc}$  and  $E_{desc}$  in natural language form. By this reconstruction, now the objective/loss function for training the seq2seq model is changed from Equation 5.2 to be:

$$J(\hat{s}, y) = -\hat{p}(y|f(\hat{s}_{1:T'}; \theta_e); \theta_d) \quad (5.2)$$

Here  $y$  refers to the output label text in response to  $T_{desc}$  and  $E_{desc}$  in the input, e.g., “yes” or “no” to “Is this message informative to 2015 Nepal earthquake”. As described, CAST differs from the standard approach in two main aspects. First, it considers  $T_{desc}$ , which is inspired by prior work [139] using a task description in the input example for a COVID-related event extraction task. In addition, CAST considers  $E_{desc}$  for making the model domain-aware when tested on different events.

## 5.2.2 Experiments setup

This section describes the experimental setup for the evaluation of CAST including benchmark datasets, adaptation scenarios and training details.

### 5.2.2.1 Datasets

In this work, the datasets used in the experiments include **nepal\_queensland** [2] and **CrisisT6** [101]. These are two benchmark informativeness classification datasets consisting of crisis tweets from two and six events respectively as described in Section 3.4.1. The two datasets are selected and used in this work since the binary informativeness classification task is compatible with the proposed method and they have equal sizes of instances across different crisis events. Considering that both datasets relate to the informativeness classification task, their labels are unified to the same target labels  $y$ , without altering the fundamental requirements of the task (i.e. to determine whether a message is informative with respect to a particular crisis event). In **nepal\_queensland**, *relevant* is changed to *yes* and *not\_relevant* is changed to *no*, likewise for **CrisisT6**.

Following this unification, the task description  $T_{desc}$  becomes the same for the two datasets.

As **nepal\_queensland** releases splits of training, development and test sets, the same splits are used in the experiments for its in-domain adaptation. For cross-domain adaptation, the training and development sets of source events are used for model training and the test sets of target events are used for testing. As **CrisisT6** releases only a data set for each event, 5-fold cross validation is used in the experiments for its in-domain adaptation. For cross-domain adaptation, the data sets of source events are used for model training and the data sets of target events are used for testing.

### 5.2.2.2 Adaptation scenarios

Using the datasets, the experiments apply two adaptation scenarios for model training, as outlined in Figure 5.1.

**standard:** This scenario represents common practice in the literature for fine-tuning seq2seq Transformers on a downstream task, and is used as a baseline in the experiments. This scenario simply feeds the raw training tweets to the seq2seq model without any additional text being added. Here, a training tweet no matter what crisis event it belongs to is always constructed in the form: “ $\{\text{tweet\_text}\}$ ”.

**postQ:** This scenario represents a particular use case of the CAST method. Referring back to Equation 5.1, postQ takes into account  $T_{desc}$ ,  $E_{desc}$  and  $\zeta_{\oplus}$ . In this scenario,  $T_{desc}$  becomes “*Is this message relevant to*” and  $E_{desc}$  becomes “ $\{\text{location\_name}\} \{\text{crisis\_name}\}$ ”, which are available in the selected datasets. For example, in **nepal\_queensland**,  $E_{desc}$  becomes “*Nepal Earthquake*” or “*Queensland Floods*”. Finally, the function  $\zeta_{\oplus}$  constructs the input sequence to be in the form of a question-answering sequence: “*Content: {tweet\_text}. Question: Is this message relevant to {location\_name} {crisis\_name}?*”<sup>1</sup>. This is determined by testing different variants of the task and event descriptions in a pilot study with CrisisT6, summarised as follows.

- **variant 1.** This is similar to postQ except that  $\{\text{location\_name}\}$  is removed from  $E_{desc}$ , making the input location-agnostic to the model. The final input sequence is constructed like: “*Content: {tweet\_text}. Question: Is this message relevant to {crisis\_name}?*”.
- **variant 2.** This is similar to postQ except that  $\{\text{location\_name}\}$  and  $\{\text{crisis\_name}\}$  are re-arranged such that the input is like: “*Content: {tweet\_text}. Question: Is*

---

<sup>1</sup>For in-domain adaptation, the location name and crisis name are from the source event(s) at both training and testing time. For cross-domain adaptation, the location name and crisis name are from the source event(s) at training time and from the target event at testing time.

*this message relevant to a {crisis\_name} event that occurred in {location\_name}?”.*

- **variant 3.** This variant constructs the input sequence by setting  $T_{desc}$  to be empty such that the input is like: “*Content: {tweet\_text}. Question: {location\_name} {crisis\_name}?*”.

The experimentation on these variants and postQ did not present any noticeable difference in performance. It is interesting to notice that there is no performance difference between variant 3 and postQ where variant 3 simply uses location and crisis name without including  $T_{desc}$  in the extended text. This is because given a specific classification task,  $T_{desc}$  will be the same for all training examples, thus leading to no difference to the model. However, ultimately the variant with  $T_{desc}$  (i.e., postQ) is chosen in the subsequent experiments mainly because this leaves room for expanding CAST to multi-task learning settings where  $T_{desc}$  becomes different for different tasks.

#### 5.2.2.3 Training details

The experiments are conducted to examine the performance of the above two scenarios in many-to-one and one-to-one in-domain and cross-domain adaptations through fine-tuning seq2seq PLMs on the two benchmark datasets. Given a number of existing such seq2seq models (Section 2.2.4.3), the Transformer decoder-encoder based PLM T5 [113] is used as the target model in this study due to its availability of small and medium-sized pre-trained weights and its strong performance in various downstream language tasks. To be specific, the off-the-shelf `t5-small` and `t5-base` weights implying different model sizes are used in this study, which are abbreviated to `small` and `base`<sup>2</sup>. In fine-tuning, the learning rate is set to be  $5e-05$  using Adam optimizer [59], updated by a linear decay scheduler with warmup ratio 10% of the total training steps (12 epochs),, which is based on the work in a similar domain [139].

### 5.2.3 Discussions and Findings

Having conducted extensive experiments with the two selected benchmark datasets, this section reports and discusses the results of CAST compared to baselines in many-to-one and one-to-one adaptations (including both in-domain and cross-domain). In addition, the insights on how to select source data for model training in many-to-one and one-to-one adaptations are presented. Specifically, in the context of real-world crisis domain adaptation, the study of one-to-one adaptation aims to identify

---

<sup>2</sup>`t5-small` and `t5-base` have around 60M and 220M parameters respectively. There are larger versions originally released by the authors, such as `t5-large`, `t5-3B` and `t5-11B`, these are not included given the scope of this research focuses on small and medium-sized PLMs.

the most suitable among a number of candidate source events (for which training sets are available) for adaptation to a new emerging target event (for which training data, either labelled or unlabelled, is not yet available). Similarly, the study of many-to-one adaptation investigates which combination of available source events is most suited to be adapted to an emerging target event.

### 5.2.3.1 One-to-one adaptation

#### Adaptation between two events

Table 5.1 and 5.2 present the in-domain and cross-domain performance respectively on the **nepal\_queensland** dataset across different runs. Regarding the in-domain performance, the standard and postQ runs are compared with the baseline CNN run [2]. It is found that they substantially outperform this baseline for both the Nepal Earthquake and Queensland Floods events. For cross-domain adaptation, two baselines CNN+DA+GE [2] and RNN+AE [76] that use unlabelled target data in training, i.e., target-data-dependent (TDD), are included for comparison. For target data independent (TDI) runs, apart from standard and postQ, two baselines CNN+DA [2] and RNN [76] are reported. To compare standard and postQ with the TDD baselines, they substantially outperform the CNN+DA+GE run. When it comes to the state of the art TDD RNN+AE run, the postQ-base achieves competitive performance, with 87.06 versus 81.18 in NE→QQF and 64.12 versus 68.38 in QF→NE (although the performance is less, it is important to note that the postQ as a TDI approach does not rely on any target data). However, the postQ-base outperforms the state of the art TDI RNN run in cross-domain adaptation with 87.06 versus 55.17 in NE→QQF and 64.12 versus 64.18 in QF→NE. To compare the standard with the postQ in in-domain adaptation (see Table 5.1), it is found that postQ performs very similarly to the standard. This makes sense since postQ is only different from standard in appending an extra task and event description. For in-domain adaptation, the appended text is the same for all training examples during training and at inference time, thus leading to no difference for the model when the extra text exists or not, which explains why they have the same level of performance. Interestingly for cross-domain adaptation, postQ does not achieve performance better than the standard method when using a small model. When using a bigger model (i.e., postQ-base), it seems that postQ performs much better than the standard method (see Table 5.2). This finding is further verified in the subsequent experiments on the **CrisisT6** dataset.

#### Adaptation between more events

Based on the **nepal\_queensland** dataset, some evidence of the effectiveness of the CAST-based runs (particularly for TDI cross-domain adaptation) has been identified

	NE → NE QQQF → QQQF	
CNN [2]	65.11	93.54
<b>standard-small</b>	75.35	96.31
<b>standard-base</b>	74.40	96.83
<b>postQ-small</b>	79.25	96.34
<b>postQ-base</b>	77.57	96.81

Table 5.1: The in-domain adaptation weighted F1 scores for **nepal\_queensland** where **NE**: Nepal Earthquake and **QQF**: Queensland Floods. The CNN run refers to the supervised CNN run from [2] and the runs in bold refer to the **standard** and **postQ** runs with **t5-small** and **t5-base**.

	source → target	NE → QQQF	QQF → NE
TDD	CNN + DA + GE [2]	65.92	59.05
	RNN + AE [76]	81.18	68.38
TDI	CNN + DA [2]	60.94	57.79
	RNN [76]	55.17	64.18
	<b>standard-small</b>	82.43	58.99
	<b>standard-base</b>	77.39	60.25
	<b>postQ-small</b>	78.21	63.75
	<b>postQ-base</b>	87.06	64.12

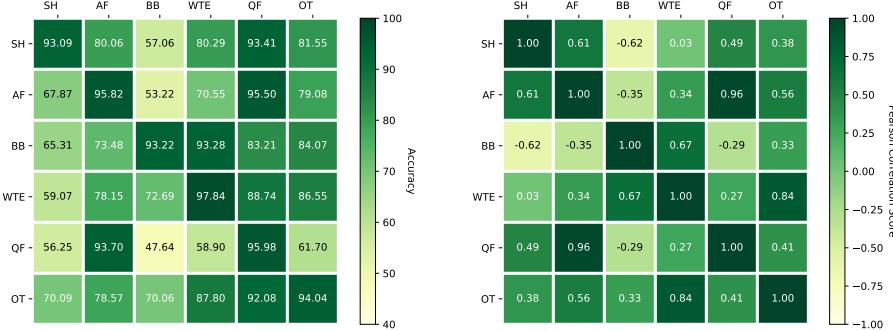
Table 5.2: The cross-domain adaptation weighted F1 scores for **nepal\_queensland**. The runs are presented in two categories: target data dependent (TDD) and target data independent (TDI). The CNN+DA+GE and CNN+DA refer to the CNN baseline runs in [2] and RNN+AE and RNN refer to the RNN baseline runs in [76].

as compared to the state-of-the-art. Next, the experiments are conducted upon the **CrisisT6** dataset that consists of six different crisis events representing a wide range of domains (details can be found in Section 3.4.1).

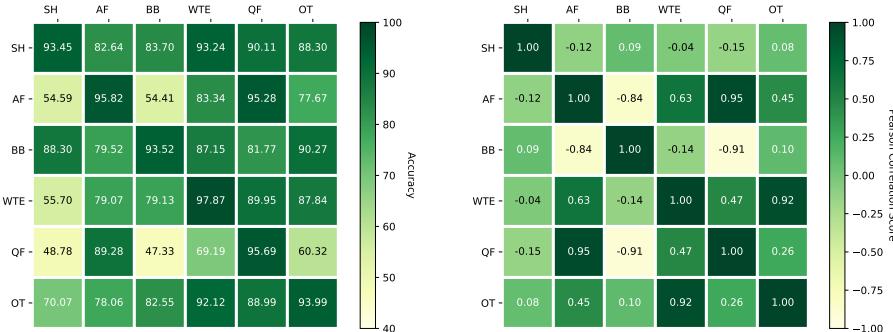
Figure 5.2 reports the results of domain adaptation between the six events using both the standard method and postQ for fine-tuning the small and base models. The sub-figures on the left side, i.e., (a), (c), (e), (g), present the accuracy scores in a matrix where the row represents the source events and the column stands for the target events. Hence, the diagonals refer to the in-domain performance and the rest are cross-domain scores. In addition to the adaptation matrices, a correlation matrix is also added to the right side of each of the adaptation matrices, i.e., (b), (d), (f), (h). The correlation matrices calculate the Pearson correlation between the rows of source events, which helps indicate how correlated two source events are in terms of their applicability to other target events.

Examining the matrices on the left, it is found that both the in-domain and cross-domain performance is consistent with **nepal\_queensland**. For example, in in-domain adaptation, postQ achieves similar performance to the standard with different model sizes<sup>3</sup>. Speaking of cross-domain adaptation, postQ performs competitively with the

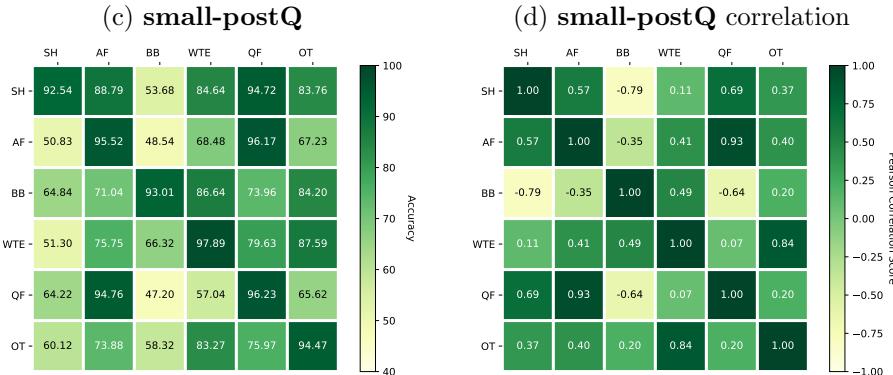
<sup>3</sup>Per [79] that reported the average in-domain state-of-the-art accuracy for **CrisisT6** which is



(a) small-standard



(b) small-standard correlation



(c) small-postQ



(e) base-standard

(f) base-standard correlation

Figure 5.2: Event adaptation accuracy and correlation for **CrisisT6** where SH: Sandy Hurricane, AF: Alberta Floods, BB: Boston Bombing, WTE: West Texas Explosion, QF: Queensland Floods, and OT: Oklahoma Tornado. Here QF is used to stand for Queensland Floods to distinguish it from QQF in **nepal\_queensland**.

95.6, the in-domain scores as seen from the diagonals approximate this state-of-the-art. Since this state-of-the-art was gained based on a random test leave-one-out evaluation which is different from

	$SH \rightarrow QF$	$SH \rightarrow BB$	$SH \rightarrow WTE$	$SH \rightarrow OT$	$SH \rightarrow AF$	$QF \rightarrow BB$
NB-S	76.84	68.66	77.21	80.78	71.06	74.97
NB-ST	82.40	84.06	90.82	87.76	82.57	81.86
<b>postQ-base</b>	94.49	89.14	90.61	94.49	93.42	88.27
	$QF \rightarrow OT$	$QF \rightarrow AF$	$BB \rightarrow OT$	$BB \rightarrow AF$	$BB \rightarrow WTE$	Average
NB-S	84.13	84.35	73.81	78.87	94.77	78.68
NB-ST	85.48	86.91	83.96	86.01	94.82	86.06
<b>postQ-base</b>	89.09	94.90	89.85	76.74	94.67	90.52

Table 5.3: Cross-domain accuracy comparison between postQ-base, NB-S (TDI) and NB-ST (TDD) from [71] who studied the 11 event pairs of CrisisT6.

standard when using the `small` model (see Figure 5.2a and 5.2c). For the `base` model, postQ substantially outperforms the standard in cross-domain adaptation (see Figure 5.2e and 5.2g). As baselines, postQ-base is compared with two approaches presented in [71], using a Naïve Bayes classifier with and without self-training respectively (NB-ST and NN-S). Here, postQ-base is compared with their NN-S and NB-ST runs on 11 event pairs, as presented in Table 5.3. This shows that the postQ-base substantially outperforms the target-data independent NB-S which is consistent with the results reported for `nepal_queensland`. When compared to the TDD NB-ST, postQ-base achieves higher accuracy scores on almost all of the 11 event pairs, resulting in higher overall accuracy for postQ (90.02 versus 86.06 in average accuracy).

Regarding cross-domain adaptation, some interesting points are noticed. The results from [71] present some evidence that similar event pairs (where the same or similar type of crises occurred at different locations) such as Queensland Floods ( $QF \rightarrow$ Alberta Floods ( $AF$ ) and Boston Bombings ( $BB \rightarrow$ West Texas Explosion ( $WTE$ )) are more likely to bring better scores than dissimilar pairs like Queensland Floods ( $QF \rightarrow$ Boston Bombings ( $BB$ )) and Boston Bombings ( $BB \rightarrow$ Alberta Floods ( $AF$ )) (see Table 5.3). This evidence is enhanced in this study also. It is noted that the Alberta Floods ( $AF$ ) and Queensland Floods ( $QF$ ) relate to the same type of crisis (flooding) albeit in different locations at different times. It is interesting that in the standard and postQ runs these two events are **reciprocal**, indicating that either of them as the source event is well-suited to being adapted for the other as the target event. For example, the  $AF \rightarrow QF$  adaptation always achieves accuracy around 95 and  $QF \rightarrow AF$  adaptation achieves accuracy of 89.28 at the worst (Figure 5.2c). Examining their correlation scores on the right, it is found that they are not only reciprocal, but also **highly correlated**, ranging from 0.91 to 0.96 (Figure 5.2h and 5.2b). This lends credence to the idea that similar event types have similar characteristics in terms of their applicability to cross-domain adaptation and could potentially be used interchangeably for a novel target event.

---

5-fold cross validation used in this work and thus this score is only for reference instead of direct comparison.

Another event pair with some similar characteristics are the Boston Bombing (BB) and the West Texas Explosion (WTE). These are perhaps less similar to the floods above in that one was an intentionally planted explosive device whereas the other was a factory fire that later resulted in an explosion. In this situation, it can be seen that whereas BB can be successfully adapted to WTE, the reverse is not the case. This may be related to the observation that BB is itself a difficult target event to adapt to, as evidenced by the fact that the cross-domain adaptation tends to be poorest in general when BB is the target.

Surprisingly, it is found that the Sandy Hurricane (SH) and Oklahoma Tornado (OT) datasets can not only be well adapted to AF and QF in most cases but also adapt well to WTE. This implies that there may be a certain degree of common linguistic features shared between the tornado/hurricane events and the explosion event. It seems that these findings indicate that the more similar a source event is to a target event, the more likely it is to exhibit better adaptation performance for that target event. Now such a question is naturally raised: does combining multiple similar source events add further benefit (many-to-one adaptation)?

### 5.2.3.2 Many-to-one adaptation

Considering the variety of crisis events and training efficiency, the many-to-one experimental runs are based on the **CrisisT6** dataset. The first experiment is leave-one-out cross-domain adaptation where one crisis event is chosen as the target domain and the union of the others as the source domain. Table 5.4 presents the results of fine-tuning using both the standard and postQ approaches. To compare the runs with the baseline by [73], it reveals that the CAST-based postQ run achieves 91.03 versus 89.6 in average accuracy. In addition, from this table it can be seen that there is no substantial difference between standard and postQ when AF, QF and OT are left out (they already achieve above 90% accuracy). However, when leaving out SH and BB, standard performance is substantially lower than for postQ. This experiment indicates that postQ outperforms the standard approach when considering multiple events as the source domain. This is further justified by the next experiment.

The next experiment is conducted to test if more source events bring better adaptation performance, whether they are similar or dissimilar. For this purpose, two event pairs: (QF, AF) and (BB, WTE) are selected. As indicated above, each pair contains two similar event types, while the two pairs themselves are dissimilar. The decision to choose these two pairs as similar event pairs is guided by existing work [71] and the correlation scores that are presented in Figure 5.2.

Table 5.5 presents the results of combinations of multiple source events adapted to

	standard	postQ	
AF+BB+WTE+QF+OT→SH	74.35	AF+BB+WTE+QF+OT→SH	82.16
SH+BB+WTE+QF+OT→AF	95.22	SH+BB+WTE+QF+OT→AF	95.16
SH+AF+WTE+QF+OT→BB	70.18	SH+AF+WTE+QF+OT→BB	88.01
SH+AF+BB+QF+OT→WTE	89.39	SH+AF+BB+QF+OT→WTE	95.94
SH+AF+BB+WTE+OT→QF	95.59	SH+AF+BB+WTE+OT→QF	95.71
SH+AF+BB+WTE+QF→OT	90.52	SH+AF+BB+WTE+QF→OT	89.16
Average	85.88	Average	91.03

Table 5.4: Leave-one-out cross-domain adaptation accuracy using CrisisT6. The last row reports the average score. As a comparison, the best average score reported in the literature is 89.6 [73].

the two pairs. First, it shows that postQ overall outperforms the standard in most situations (accuracy is much higher when BB and WTE are the target events and is at least the same level for QF and AF). However, the more interesting observation from this experiment is that simply increasing the number of source events does not guarantee benefits to the adaptation performance but it depends on what source events are added. For example, when QF is the target event, only a trivial difference is observed between AF-to-QF and leaving QF out (thus combining all other crises as the source domain). A similar pattern is observed when AF is the target event.

	standard		postQ	
	QF	AF	QF	AF
AF	95.5	-	95.28	-
QF	-	93.7	-	89.28
BB+WTE	83.79	74.69	87.27	76.18
SH+OT	93.97	85.05	93.71	83.71
AF+SH+OT	95.32	-	95.76	-
QF+SH+OT	-	95.45	-	95.38
SH+OT+BB+WTE	92.72	82.67	91.44	81.81
AF+SH+OT+BB+WTE	95.59	-	95.71	-
QF+SH+OT+BB+WTE	-	95.22	-	95.16

(a) QF and AF as the target events

	standard		postQ	
	BB	WTE	BB	WTE
WTE	72.69	-	79.13	-
BB	-	93.28	-	87.15
AF+QF	51.24	68.42	50.1	80.56
SH+OT	75.71	89.32	82.14	92.01
WTE+SH+OT	73.3	-	85.3	-
BB+SH+OT	-	92.44	-	96.35
AF+QF+SH+OT	63.94	79.6	78.42	86.29
WTE+AF+QF+SH+OT	70.18	-	88.01	-
BB+AF+QF+SH+OT	-	89.39	-	95.94

(b) BB and WTE as the target events

Table 5.5: Many-to-one cross-domain adaptation on similar event pairs

Table 5.5a also indicates that adding BB+WTE seems not to add any benefit to the performance (indeed this reduces performance when compared with a source domain of SH+OT), and SH+OT can help to a degree (adding SH+OT to QF results in improved cross-domain performance when AF is the target).

Table 5.5b demonstrates a similar outcome. When BB and WTE are the target events, AF+QF contributes little to the adaptation performance. Surprisingly, SH+OT not only helps QF and AF but also helps BB and WTE, which coincides with the one-to-one results as reported in the previous section. Hence, as a recommendation to maximise the adaptation performance to a target event (e.g., AF), it is good to combine its similar events (e.g., QF+SH+OT → AF) as the source domain and exclude dissimilar

events (e.g., BB+WTE)<sup>4</sup>.

## 5.3 Conclusions

The chapter focused on developing adaptation models that learn from a previous crisis or crises in order to classify messages related to a new unfolding crisis situation, i.e., one-to-one and many-to-one crisis domain adaptations. To this end, this chapter presented CAST, which fine-tuned event-aware seq2seq Transformer-based PLMs for one-to-one and many-to-one domain adaptations between crisis events. To test the effectiveness of CAST, extensive experiments were conducted with two benchmark crisis informativeness classification datasets. In the one-to-one adaptation setting, CAST was demonstrated to be effective in cross-domain adaptation, outperforming the state of the art baselines without using any target data. Its advantage over the standard approach was more pronounced when training with a bigger model. In the many-to-one adaptation setting, CAST also outperformed the standard method and baselines in the literature. Interestingly, in the many-to-one adaptation setting, the results indicated that there was merit in choosing a source domain with similar characteristics (i.e. fine-tuning based on a similar type of crisis). If multiple existing similar events were available, these could be combined to form a larger source dataset to improve adaptation performance, while dissimilar events may have harmed classification performance.

Referring back to Chapter 4, the work in the many-to-many adaptation helps guide the community in deploying a computational system for crisis message categorisation without the need for knowledge of specific types of source and target events. Here, the CAST work provides insights on how to select source events for crisis adaptation. However, in real-world crisis message categorisation, crisis domain adaptation only works when the categorisation task for target events is exactly the same as that of source events. In other words, the pre-defined aid types (classes) of the source data must be consistent with those of the target data in order for adaptation to be possible. In a real-world context, when there is a lack of annotated data for a new crisis event that emerges with new classes (due to the time-sensitive nature of such events), it is impractical to apply supervised approaches for categorizing the event. It is also very costly in terms of both human labour and time to annotate data for the new classes. This motivates the research to go beyond crisis domain adaptation and proposes crisis

---

<sup>4</sup>Although there is no direct harm to the performance when including the dissimilar events (see the last two rows of Table 5.5a and 5.5b), it is suggested to exclude them which can help reduce the training size and thus improve the training efficiency.

few-shot learning, which uses no labelled source data but a minimal quantity of labelled or unlabelled target data for crisis message categorisation. In the following chapter, the proposed augmentation approaches using PLMs for this purpose will be introduced.

# Chapter 6: Augmentation for Crisis Few-shot Learning

---

## 6.1 Introduction

The previous two chapters discussed the use of supervised approaches, including domain adaptation, for crisis message categorisation. However, these approaches have limitations in meeting the real-world user needs for crisis response. Domain adaptation only works effectively when the categorisation task for the target event is the same as that of the source event, meaning that the pre-defined aid types (classes) must be the same for both. While this may meet some user needs, it has limited applicability to many real-world scenarios. In practice, responders may need to search for new types of information for emerging events. For instance, in a flood event, responders may request information about “shelter”, but this type of information may not have been included in the annotations for previous events such as terrorist attacks. This makes it challenging to adapt the model trained on previous events. Moreover, the concept of domain adaptation relies on the availability of extensive training data related to past crises. However, in reality, this may not always be feasible, particularly when dealing with a novel crisis event, such as an epidemic with entirely new aid requirements or classifications, where no annotated training data is available due to the absence of similar prior incidents. Additionally, research from the previous chapter suggests that adapting the model using a similar previous event can be successful, but even when the events are similar, there is no guarantee that the results will be good as crises with similar characteristics may still have differences. These considerations have motivated this research to move towards building categorisation models that rely on minimal quantities of training data. The approaches outlined in this chapter aim to achieve “crisis few-shot learning”. This refers to a situation where a very small quantity of labelled messages relating to the target event are required to bootstrap the system. This substantially reduces the burden of manually labelling data so as to be a manageable task in the early stages of an unfolding crisis. In addition to this, larger quantities of unlabelled data relating to the target event are also used, which do not require any manual annotations to be conducted during the crisis.

This chapter presents two augmentation-based approaches for crisis few-shot learning. The first approach, called Self-controlled Text Augmentation (STA), uses sequence-to-sequence (seq2seq) pre-trained language models (PLMs) to generate new crisis messages based on a small training set consisting of a few labelled messages for each class that the classifier is required to identify<sup>1</sup>. To improve the quality of the generated messages in STA, the seq2seq model is used to evaluate the confidence of the generated messages. Only those with high confidence are selected and combined with the original labelled data to create the augmented data for training the categorisation model in a standard supervised manner (Section 6.2). The second approach, called Iterative Text Augmentation (ISA), is an optimized version of STA that further improves the quality of the generated messages by incorporating an iterative mechanism and a de-duplication step into the STA pipeline (Section 6.3).

## 6.2 Self-Controlled Text Augmentation (STA)

As discussed in Section 3.2.2, the key to creating high-quality augmented data is to address the challenges of “lexical diversity” and “semantic fidelity”. Lexical diversity refers to the diversity of the augmented samples in relation to the original samples, which helps to add knowledge to the training data. Semantic fidelity means that, despite being diverse, the augmented samples should have a reliable alignment between their semantic meanings and class labels, so as not to introduce noise into the training data. To tackle these challenges, this section introduces a novel generation strategy called Self-controlled Text Augmentation (STA). STA uses a seq2seq pre-trained language model with a set of task-specific prompts to generate new samples and then select the best ones. The prompts consist of classification and generation templates, allowing the model to learn both a classification task and a generation task jointly. The generation task is used to generate new samples intended to be related to particular classes by being dependent on the class surface name as an input and the classification task is used to choose only the high-confidence samples to be added to the training set, with the aim of including only samples with high semantic fidelity. The following section describes STA in the context of text classification.

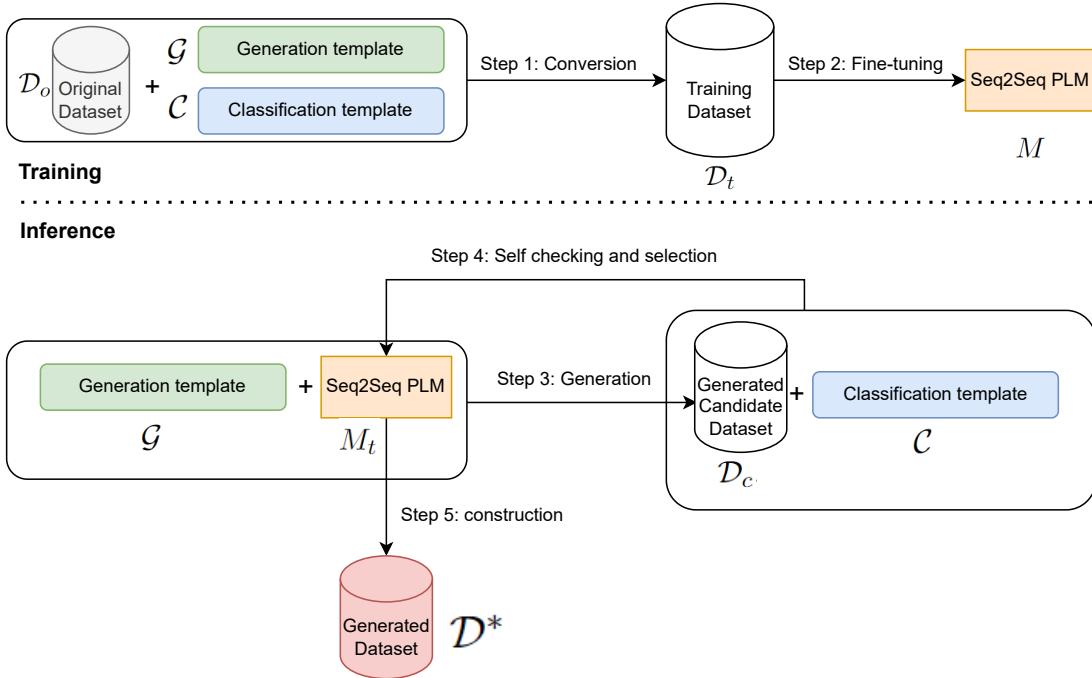


Figure 6.1: The architecture of the Self-controlled Text Augmentation approach (STA).

### 6.2.1 Self-controlled method

Figure 6.1 illustrates the workflow of STA and Algorithm 1 states STA in simple terms. In Figure 6.1, the upper portion outlines the finetuning component of the method (**Training**), whilst the lower portion demonstrates the procedure for generating novel data (**Inference**). In STA, a seq2seq language model is first fine-tuned using a small dataset (i.e. the original dataset) that is converted by two types of prompt templates: generation templates and classification templates. These templates recreate the original dataset in a way that allows the model to be trained on both a generation task and a classification task simultaneously. Once the model is trained, it can use the generation templates to generate new samples for a candidate dataset. However, not all of these candidates are used as augmentations, as some may be noisy. Instead, the model’s predictions for each candidate using the classification templates are used to determine which candidates have high confidence and should be included in the generated dataset. This dataset, along with the original dataset, is then used to train a downstream classification model. The following provides a detailed description of the steps involved in STA, starting with prompts-based multi-task training and ending with data generation and self-checking.

---

<sup>1</sup>A preprint of this STA work can be found in [144].

---

**Algorithm 1** : Self-Controlled Text Augmentation (STA)

---

- Require:** Original dataset  $\mathcal{D}_o$ . Generation model  $M$ . Generation template  $\mathcal{G}$ . Classification template  $\mathcal{C}$ .
- 1: Convert  $\mathcal{D}_o$  to training dataset  $\mathcal{D}_t$  via  $\mathcal{G}$  and  $\mathcal{C}$ .
  - 2: Finetune  $M$  on  $\mathcal{D}_t$  in a generation task and a classification task jointly to obtain  $M_t$ .
  - 3: Use  $\mathcal{G}$  and  $M_t$  to generate candidate dataset  $\mathcal{D}_c$ .
  - 4: Apply  $M_t$  to do classification inference on  $\mathcal{D}_c$  with  $\mathcal{C}$  to select the most confident examples.
  - 5: Combine the final generated dataset  $\mathcal{D}^*$  with the selected examples.
  - 6: Use the combined data for downstream model training
- 

Template	Source sequence ( $s$ )	Target sequence ( $t$ )
Classification	$c_1$ Given {Topic}: $\{\mathcal{L}\}$ . Classify: $\{x_i\}$	$\{y_i\}$
	$c_2$ Text: $\{x_i\}$ . Is this text about $\{y_i\}$ {Topic}?	yes
	$c_3$ Text: $\{x_i\}$ . Is this text about $\{\bar{y}_i\}$ {Topic}?	no
Generation	$g_1$ Description: $\{y_i\}$ {Topic}. Text:	$\{x_i\}$
	$g_2$ Description: $\{y_i\}$ {Topic}. Text: $\{x_j\}$ . Another text: $\{x_i^{0-2}\}$	$\{x_i^{3\dots}\}$

Table 6.1: Prompt templates of STA where  $x_i$  refers to an input sequence and  $y_i$  implies the label surface name of the sequence, “Topic” refers to a simple keyword describing the target task dataset e.g. “disaster aid type” and  $\mathcal{L}$  is the list of all class labels in the dataset. The symbol  $\bar{y}_i$  in  $c_3$  stands for any label in  $\mathcal{L} \setminus \{y_i\}$ , chosen randomly. In  $g_2$ , the  $x_j$  denotes another sample from same class as  $x_i$  (i.e.  $y_j = y_i$ ) chosen randomly and  $x_i^{0-2}$  refers to the first three words of  $x_i$  as the context to generate the remaining words of  $x_i$  i.e.  $x_i^{3\dots}$ .

An example from a disaster aid type (Topic) classification dataset where the classes ( $\mathcal{L}$ ): missing or found people, sympathy and support...	
Text ( $x$ )	<i>UPDATE: Body found of man who disappeared amid Maryland flooding</i>
Label ( $y$ )	<i>missing or found people</i>
Converted examples by classification templates: source( $s$ ), target( $t$ )	
<i>Given disaster aid type: missing or found people, sympathy and support... Classify: UPDATE: Body found of man who disappeared amid Maryland flooding</i>	<i>missing or found people</i>
<i>Text: UPDATE: Body found of man who disappeared amid Maryland flooding Is this text about missing or found people disaster aid type?</i>	yes
<i>Text: UPDATE: Body found of man who disappeared amid Maryland flooding Is this text about negative disaster aid type?</i>	no
Converted examples by generation templates: source( $s$ ), target( $t$ )	
<i>Description: missing or found people disaster aid type.</i>	<i>UPDATE: Body found of man who disappeared amid Maryland flooding</i>
<i>Description: missing or found people disaster aid type.</i>	<i>Search Database from Mati and Rafina areas #Greecefires #PrayForGreece #PrayForAthens</i>
<i>Description: missing or found people disaster aid type.</i>	<i>in mass grave in Sussundenga, at least 60 missing - #Mozambique #CycloneIdai #CycloneIdai</i>
<i>Description: missing or found people disaster aid type.</i>	<i>way as search for missing continues after California wildfires</i>

Table 6.2: The demonstration of an example conversion by the prompt templates in Table 6.1 where the input text is highlighted in blue and label is highlighted in red for readability.

### 6.2.1.1 Prompts-based multi-task training in seq2seq Models

Let  $M$  be a pretrained seq2seq PLM. Such models consist of an encoder-decoder pair: the encoder takes a source sequence  $s$  and produces a contextualised encoding sequence  $\bar{s}$ . For each token  $t_i$  that is to be generated for  $\bar{s}$ , the encoded input sequence and the subsequence  $t$ :  $\{t_1, t_2, \dots, t_{i-1}\}$  (i.e. all tokens in the output sequence prior to  $t_i$ ) are used as the input for the decoder to compute the conditional probability  $p_M(t_i|t_{1:i-1}, \bar{s})$  for  $t_i$  and  $p_M$  is estimated by trainable parameters of the model. The possible target output (a sequence)  $t$ :  $\{t_1, t_2, \dots, t_m\}$  given  $\bar{s}$  is generated via the factorisation:

$$p_M(t_{1:m}|\bar{s}) = \prod_{i=1}^m p_M(t_i|t_{1:i-1}, \bar{s}) \quad (6.1)$$

Let  $\mathcal{D}_o = \{(x_i, y_i)\}_{i=1}^n$  be a small corpus of few-shot data for text classification where  $x_i \in \mathcal{X}$  and  $y_i \in \mathcal{L}$  are an input text instance and its label respectively. The goal is to produce a training dataset  $\mathcal{D}_t$  to finetune  $M$  and ensure that it is primed for generating diverse examples that are faithful to the appropriate label.

Formally, a *template* is a function  $T : \mathcal{X} \times \mathcal{L} \rightarrow V^* \times V^*$  where  $V$  is the vocabulary of  $M$  and  $V^*$  denotes the set of finite sequences of symbols in  $V$ . Given a set of templates  $\mathcal{T}$ , let  $\mathcal{D}_t = \mathcal{T}(\mathcal{D}_o) = \bigcup_{T \in \mathcal{T}} T(\mathcal{D}_o)$ . That is, each sample  $(x_i, y_i) \in \mathcal{D}_o$  is converted to  $|\mathcal{T}|$  samples in the training dataset  $D_t$ . Table 6.1 lists all the templates that are specifically designed for classification and generation purposes and Table 6.2 demonstrates how this conversion is performed.

Crucially, two types of template families are constructed: classification templates  $\mathcal{C}$  and generation templates  $\mathcal{G}$  and the set  $\mathcal{T}$  comprises both of these (i.e.  $\mathcal{T} = \mathcal{C} \cup \mathcal{G}$ ).

- **Classification templates** have the form  $c(x, y) = (f_1(x), f_2(y))$  or  $c(x, y) = (f_1(x, y), f_2(y))$  where  $f_1$  and  $f_2$  refer to functions that convert a piece of text to the source sequence and target sequence respectively. Here, the text  $x \in \mathcal{X}$  is not a part of the target output.
- **Generation templates** have the form  $g(x, y) = (f_1(y), f_2(x))$  or  $g(x, y) = (f_1(x, y), f_2(x))$  where  $f_1$  and  $f_2$  refer to functions that convert a piece of text to the source sequence and target sequence respectively. Here, the label  $y \in \mathcal{L}$  is not a part of the target output.

After obtaining the training dataset  $D_t$  by the conversion,  $M$  can then be finetuned to obtain  $M_t$  (see 6.2.2.2 for details on the training parameters used). The motivation here is that  $D_t$  will ensure the model is finetuned to learn both how to generate a new

piece of text of the domain based on the label description as well as to classify a piece of text relating to the domain. For example, during the training stage, the model can use the first classification template  $c_1$  in Table 6.2.1.1, to predict the label for an original sample  $x_i$ , which is a classification task. Meanwhile, using the first generation template  $g_1$ , the model can generate the original sample  $x_i$  given its label  $y_i$  as input, which is a generation task. At the inference stage, the model can use the generation template  $g_i$  to generate a new sample given a label as input, and then use the classification template  $c_1$  to classify the generated example and check its confidence, which is described next.

### 6.2.1.2 Data generation, self-checking and selection

A two-step process is adopted here: first, candidates are generated and then a fraction of the candidates are selected to be included as augmentations. This process is conducted for each class separately so it is assumed that for the remainder of this section that a label  $y \in \mathcal{L}$  has been fixed for both generation and selection (the same process is applied to other labels also).

The first objective is to generate  $\alpha \times n_y$  samples where  $n_y$  is the original number of samples in  $\mathcal{D}_o$  for label  $y$  and  $\alpha$  is a multiplier hyperparameter to control the ratio of generated samples to original samples. To perform this generation, Equation 6.1 is applied autoregressively to a chosen prefix or source sequence.

Referring back to Table 6.1, there are two choices of generation template sequence ( $g_1$  and  $g_2$ ) that can be used to construct  $s$ . Here  $g_1$  is chosen over  $g_2$  as the former only needs the label (the dataset description is viewed as a constant), i.e.

$$g_1(x, y) = (f_1(y), f_2(x)).$$

which gives the model greater freedom to generate diverse examples.

Thus  $s$  is set to be  $s = f_1(y)$  and  $\alpha \times n_y$  samples are generated using the finetuned model  $M_t$ .

Now a synthetic candidate dataset for label  $y$ ,  $\mathcal{D}_c^y = \{(x_i, y)\}_{i=1}^{\alpha \times n_y}$ , is obtained, which will be refined using a self-checking strategy for selecting the generated samples based on the confidence estimated by the model  $M_t$  itself.

For each synthetic sample  $(x, y)$ , a source sequence is constructed using the template  $c_1(x, y) = (f_1(x), f_2(y)) = (f_1(x), \{y\})$ , that is,  $s$  is set to be  $s = f_1(x)$ . Given  $s$ , a score function  $u$  is defined in the same way as in [120]:

$$u(y|s) = \log p_{M_t}(\{y\}|\bar{s})$$

Equivalently, this is the *logit* computed by  $M_t$  for the sequence  $\{y\}$ . Then the labels in  $\mathcal{L}$  are normalised by applying a softmax over each of the scores  $u(\cdot|s)$ :

$$q(y|s) = \frac{e^{u(y|s)}}{\sum_{l \in \mathcal{L}} e^{u(l|s)}}$$

Finally, the elements of  $\mathcal{D}_c^y$  are ranked by the value of  $q$  and the top  $\beta \times n_y$  samples ( $\beta < \alpha$ ) are selected to form the final generated dataset for the specific label  $y$ :  $D_*^y$ . The overall  $D_* = \bigcup_{y \in \mathcal{L}} D_*^y$  is the union of specific labels. Here,  $\beta$  is called the *augmentation factor* and  $\alpha = 5 \times \beta$  is used. Thus, the self-checking technique selects the top 20% of the candidate examples per class <sup>2</sup> to form the final generated  $D^*$  that is combined with the original dataset  $D_o$  for downstream model training.

## 6.2.2 Experiments

Next, extensive experiments are conducted to test the effectiveness of STA in low-data regimes. This section first describes the experimental setup including dataset and training details, and then outlines the baselines for comparison and discusses the results. Finally, an additional experiment is conducted to examine how STA can be generalised to other types of text classification tasks that are not specifically crisis-related.

### 6.2.2.1 Dataset

In this work, the **HumAID** dataset [3] is used as the benchmark dataset to test the effectiveness of STA. As discussed in Section 3.4.2, this is a single-label multi-class dataset for information types classification, consisting of crisis tweets from 19 events between 2016 to 2019 annotated by information types including sympathy and support, missing or found people, etc. (These labels are easy to understand based on their surface names.). Considering at this stage STA supports single-label multi-class tasks and relies on good surface names, this justifies why this dataset is selected here.

### 6.2.2.2 Training details

When finetuning the seq2seq model, the pre-trained T5 base checkpoint is selected as the starting weights. For the downstream classification task, “bert-base-uncased” is fine-tuned on the original training data with the augmented (generated) samples.

---

<sup>2</sup>This is based on empirical experimental search over {10%, 20%, 30%, 40%, 50%}.

Regarding the pre-trained models, both are from the publicly-released version of the HuggingFace transformers library [150]<sup>3</sup>. For the augmentation factor (i.e.,  $\beta$  in Section 6.2.1.2), the augmentation techniques including STA and the baselines are applied with  $\beta$  between 1 and 5. Since this work focuses on text augmentation for classification in low-data settings, the experiments for both STA and the baselines are conducted on 5, 10, 20, 50 and 100 samples per class randomly sampled from the training set as per [4]. Due to the randomness when sampling data of small size, all experiments are run 10 times so that the average accuracy along with its standard deviation (std.) is reported on the full test set in the evaluation.

In finetuning T5, the learning rate is set to be  $5 \times 10^{-5}$  using Adam [59] with linear scheduler (10% warmup steps), the training epochs to be 32 and batch size to be 16. At generation time, a top-k ( $k = 40$ ) and top-p ( $p = 1.0$ ) sampling technique is used for next token generation [46]. In finetuning downstream BERT, the hyper-parameters are the same as those used in T5 finetuning, with the exception that the training epoch is set to be 20. The training epochs is set to be as large as possible with the aim of finding the best model when trained on a small dataset.

### 6.2.2.3 Baselines

In evaluation, STA is compared against a set of state-of-the-art techniques found within the literature of text augmentation. These approaches include a variety of augmentation procedures from easy reformulation to deep neural text generation. STA is compared to the augmentation techniques as they are directly related to STA in generating samples that can be used in the subsequent study for examining the quality of generated examples.

- **Baseline (No Aug.)** uses the original training data as the downstream model training data. The downstream classification model is directly trained on the original few-shot data with no augmentation applied.
- **EDA** refers to easy data augmentation in [147], which provides official implementations. Hence, their implementations are adapted to the selected dataset of this study for comparison.
- **BT** and **BT-Hops** [30, 124] refer to back-translation techniques. The former is implemented by taking one step back-translation from English to another language that is randomly sampled from the 12 Romance languages provided by the “opus-mt-en-ROMANCE” model <sup>4</sup> from the transformers library [150]. The

---

<sup>3</sup><https://github.com/huggingface/transformers>

<sup>4</sup><https://huggingface.co/Helsinki-NLP/opus-mt-en-ROMANCE>

latter adds random 1 to 3 extra languages to the back-translation using the same model.

- **GPT-2** is a deep learning method proposed in [62] that uses GPT-2 for augmentation. Since no implementation is made available, the methodology described in the paper was followed. This was done by finetuning a GPT-2 base model on the small sampled training data sets (i.e., 5, 10, 20, 50, 100 samples per class to simulate the limited data scenario as in STA) and then the fine-tuned model is used to generate new samples that are conditional on both the label description and the first three words of an existing example.
- **GPT-2- $\lambda$**  is similar to GPT-2 with the addition of the LAMBDA technique from [4], which also does not offer implementations. In a similar way to GPT-2, it is implemented by finetuning a GPT-2 base model on the small training data sets to generate new samples that are later confidence checked by a BERT base model (the LAMBDA component).
- **CBERT** [151] is a strong word-replacement based method for text augmentation, for which an implementation is provided. Their implementation is applied to the selected dataset of this study for comparison.
- **BART-Span** [62] uses the seq2seq BART model for text augmentation, and an implementation is not availalble. It is implemented as described in the paper, by finetuning the BART large model using the label names and the texts of 40% consecutive masked words.

## 6.2.3 Results and discussion

### 6.2.3.1 Classification performance

Table 6.3 presents STA’s accuracy on **HumAID**<sup>5</sup>. It shows the effectiveness of STA in the crisis aid type classification task. It shows that STA outperforms a suite of state-of-the-art augmentation approaches, particularly when using fewer annotated training examples. When a higher number of samples (50-100) are used for training, it is seen that STA is slightly better than the baselines. However, STA is superior to other augmentation techniques when only a small number of examples are used to train the generator (5-10-20). In fact, STA demonstrates a difference of +18.3 and +7.8 compared to the best-performing augmentation baseline when trained on only 5 and 10

---

<sup>5</sup>The results are reported as average accuracy over 10 random experimental runs, with the standard deviation in parentheses. Numbers in bold indicate the highest in each column

samples per class respectively, demonstrating its ability to generate salient and effective training examples from limited amounts of data in the crisis tweet classification task.

Augmentation Method	5	10	20	50	100
Baseline (No Aug.)	29.09(6.64)	37.08(6.42)	60.69(4.0)	80.01(0.93)	83.43(0.97)
EDA	49.49(4.48)	64.4(3.64)	74.69(1.53)	80.7(0.99)	83.53(0.56)
BT	45.76(5.67)	59.14(5.24)	73.47(2.05)	80.42(1.22)	83.1(0.68)
BT-Hops	43.41(6.44)	57.52(5.21)	72.38(2.78)	80.12(1.11)	82.76(1.4)
CBERT	44.8(7.56)	59.46(4.76)	73.43(1.73)	80.34(0.77)	82.69(1.16)
GPT-2	46.02(4.67)	55.72(5.72)	67.28(2.6)	77.82(1.61)	81.13(0.57)
GPT-2- $\lambda$	50.73(8.62)	68.09(6.25)	78.51(1.32)	82.13(1.05)	84.2(0.79)
BART-Span	42.4(7.31)	58.62(6.98)	70.04(3.73)	79.3(1.43)	83.33(0.93)
<b>STA</b>	<b>69.0(3.92)</b>	<b>75.84(3.34)</b>	<b>80.17(1.59)</b>	<b>83.19(0.47)</b>	<b>84.52(1.14)</b>

Table 6.3: Results of STA and baselines on **HumAID** in 5, 10, 20, 50, 100 examples per class. Accuracy is reported here and the numbers in **bold** indicate the highest in columns]

### 6.2.3.2 Ablation Study

To demonstrate the importance of the self-checking procedure, an ablation study is conducted to investigate STA with and without the self-checking step. The results without self-checking are shown in Table 6.4 for **HumAID** (denoted as “STA-nosef”). As seen from this table, the approach demonstrates considerable improvements when the self-checking step is added across all tasks and training sample sizes, further supporting our augmentation technique. In fact, the difference between the two settings is considerable, with an average increase of +5.7 across all training samples sizes. It is hypothesised that the self-checking step more reliably controls the labels of the generated text, which greatly improves training stimulus and thus the performance on downstream tasks.

Of course, there are many possible choices for templates and permutations of template procedures. To further support the use of our multiple prompt templates used in STA (see Table 6.1), another ablation run is done for this purpose and its results are displayed in Table 6.4 (denoted as “STA-twoprompts”). These templates, one for classification ( $c_1$ ) and one for generation ( $g_1$ ), represent a minimalistic approach for performing generation-based augmentation with self-checking without the additional templates outlined in Table 6.1. The results show that the multiple templates used for

STA-nosef	56.44(6.95)	70.23(4.27)	76.26(3.31)	79.35(4.45)	81.76(1.25)
STA-twoprompts	68.71(10.92)	<b>77.61(3.57)</b>	80.06(1.71)	82.86(1.6)	84.31(0.69)
<b>STA</b>	<b>69.0(3.92)</b>	75.84(3.34)	<b>80.17(1.59)</b>	<b>83.19(0.47)</b>	<b>84.52(1.14)</b>

Table 6.4: Ablation study of STA on **HumAID** in 5, 10, 20, 50, 100 examples per class. Accuracy is reported here and the numbers in **bold** indicate the highest in columns

STA provide slight improvements in all training sizes except for the 10-per-class case.

### 6.2.3.3 Investigation of Lexical Diversity and Semantic Fidelity

As stated in Section 6.2, one objective of this work is to generate data with high lexical diversity and semantic fidelity. To determine if STA is capable of producing new samples that outperform the baselines in these two aspects, additional experiments were conducted to analyse the samples produced by STA and the baselines. First, the following two measurements were used to quantify high lexical Diversity and semantic Fidelity.

- **Generated Data Diversity.** The metric used for evaluating diversity is Unique Trigrams [32, 62]. This is determined by calculating the number of unique trigrams as a proportion of the total number of tri-grams in a population. As the aim is to examine the difference between the generated data and the original data, the population consists of both the original and generated training data. For this metric, a higher score indicates better diversity.
- **Generated Data Fidelity.** The semantic fidelity is measured by evaluating how well the generated data retains the semantic meaning of its label. As per [62], it is measured by first finetuning a “BERT-base-uncased” on 100% of the original training data of each classification task. The performance of the classifier on the test set of **HumAID** is 89.69. After the finetuning, to measure the generated data fidelity, the finetuned classifier is used to predict the labels for the generated data and use the accuracy between its predicted labels and its associated labels (i.e., assigned labels by STA and baselines) as the metric for fidelity. Hence, a higher score indicates better fidelity.

To present the quality of generated data in diversity and fidelity, the training data (10 samples per class) along with its augmented data ( $\beta = 1$ ) is used for investigation. Figure 6.2 depicts the diversity versus semantic fidelity of generated data by various augmentation methods for **HumAID**<sup>6</sup>. It is found that generation-based approaches such as GPT-2 or GPT-2- $\lambda$ , achieve strong diversity but less competitive fidelity (i.e. the generated samples are less representative of the labels they are intended to match). On the contrary, easy reformulation methods such as EDA perform well in retaining the semantic meaning but not in lexical diversity. The merit of STA is that it performs well in both diversity and fidelity, as can be seen from its position at the top-right of the figure. Finally, when comparing STA with and without self-checking, it can be seen that each approach produces highly diverse samples, although the self-checking step of

---

<sup>6</sup>The average scores over 10 runs are reported.

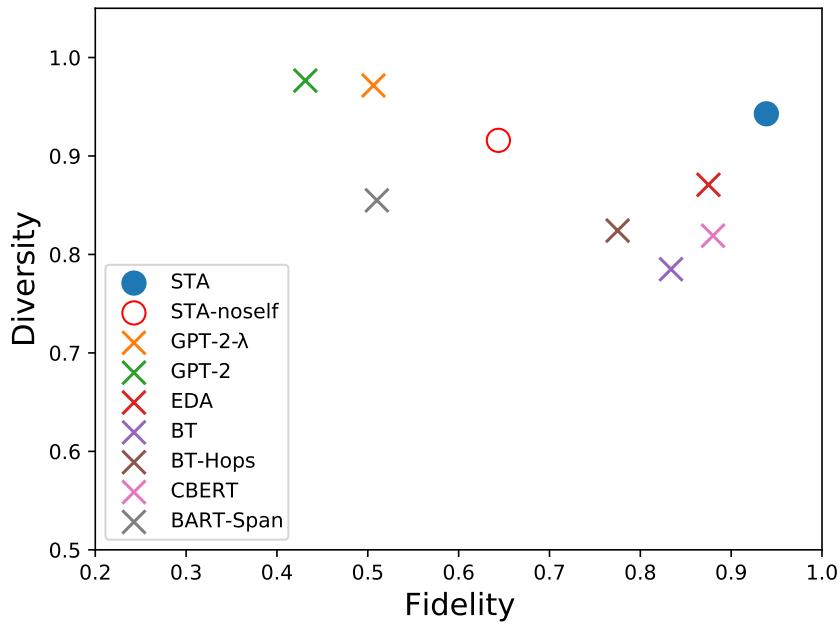


Figure 6.2: Diversity versus semantic fidelity of generated texts by various augmentation methods

STA results in a much higher level of semantic fidelity. This supports the notion that the generation-based approach (STA) is able to produce novel data that is lexically diverse, whilst the self-checking procedure can ensure consistent label retention, which produces a high semantic fidelity in the generated samples.

To conduct a qualitative analysis for the generated samples, Table 6.5 demonstrates some original samples and augmented samples generated by different methods. It is noted that the 5 augmented samples in each block are randomly selected instead of cherry-picked. Here an easy reformulation method EDA, a generation-based method GPT-2- $\lambda$  and STA-noself are used for comparison. As the results indicate, there is some difference between the original training samples and the augmented samples generated by STA and other methods. In comparison, the samples generated by STA tend to be not only diverse but also highly label-relevant (semantic fidelity). For example, despite high fidelity in EDA, it produces new samples by applying simple reformulation operations such as replacing “flooding” with “Body” for the first sample, leading to low diversity. Although GPT-2 $\lambda$  looks good in terms of diversity, some of its generated samples intended for “missing or found people” are more suited to “search and rescue”, resulting in unwanted noise. As a comparison, STA is superior in both aspects. Besides, without the self-checking component, (i.e., STA-noself) some noisy samples are also included (e.g., the second one in the demonstration), which reflects the effectiveness of the self-checking procedure in STA.

Original training examples and augmented examples for “missing or found people” of HumAID

Original	UPDATE: Body found of man who disappeared amid Maryland flooding Open Missing People Search Database from Mati and Rafina areas #Greecefires #PrayForGreece #PrayForAthens @ThinBlueLine614 @GaetaSusan @DineshDSouza case in point, #California Liberalism has created the hell which has left 1000s missing 70 dead,... Heres the latest in the California wildfires #CampFire 1011 people are missing Death toll rises to 71 Trump blames fires on poor ... #Idai victims buried in mass grave in Sussundenga, at least 60 missing - #Mozambique #CycloneIdai #CicloneIdai
EDA	update flooding found of man who disappeared amid maryland Body open missing people search database from mati escape and rafina areas greecefires prayforgreece prayforathens created gaetasusan dineshdsouza hell in point california missing has thinblueline the case which has left s liberalism dead an countless people... heres blames latest in the california wildfires campfire people are missing death toll rises to trump more fires on poor... idai victims buried in mass grave in sussundenga at mozambique missing least cycloneidai ci- cloneidai
GPT-2-lambda	@KezorNews - Search remains in #Morocco after @deweathersamp; there has been no confirmed death in #Kerala #Cambodia - Search & Rescue is assisting Search & Rescue officials in locating the missing 27 year old woman who disappeared in ... @JHodgeEagle Rescue Injured After Missing Two Children In Fresno County #Florence #Florence Missing On-Rescue Teams Searching For Search and Rescue Members #Florence #Florence #DisasterInformer #E RT @LATTAODAYOUT: RT @HannahDorian: Search Continues After Disappearance of Missing People in Florida
STA-noself	Search Database from Matias, Malaysia, missing after #Maria, #Kerala, #Bangladesh #Ker- alaKerala, #KeralaFloods, ... RT @hubarak: Yes, I can guarantee you that our country is safe from flooding during the up- coming weekend! Previous story Time Out! 2 Comments The missing persons who disappeared amid Maryland flooding are still at large. More on this in the next article. the number of missing after #CycloneIdai has reached more than 1,000, reports CNN. RT @adriane@przkniewskiZeitecki 1 person missing, police confirm #CycloneIdai. #CicloneIdai
STA	The missing persons who disappeared amid Maryland flooding are still at large. More on this in the next article. Search Triangle County for missing and missing after #Maria floods #DisasterFire Just arrived at San Diego International Airport after #Atlantic Storm. More than 200 people were missing, including 13 helicopters ... Search Database contains information on missing and found people #HurricaneMaria, hashtag #Firefighter Were told all too often that Californians are missing in Mexico City, where a massive flood was devastating. ...

Table 6.5: The demonstration of original training samples and augmented samples for “missing or found people” of HumAID

## 6.2.4 Generalisation to domains beyond crisis

Having studied the effectiveness of STA for crisis tweet classification, it is interesting to examine the extent to which it can generalise to other domains that are not crisis-related. Three more datasets beyond crisis are used for this purpose.

Following previous augmentation work [4, 62], two bench-marking datasets are used in the experiments: **SST-2** [126] and **TREC** [75]. The **Emotion** [119] dataset is also included to extend the domains of testing STA’s effectiveness. The details of each of these datasets can be found in Section 3.4.

Following the same experimental procedures as for **HumAID**, STA is run on the three

datasets (tasks) and its results compared to the baselines are reported, in addition to the results of an ablation study and an analysis of lexical diversity and semantic fidelity.

By comparison to the baselines, the results on the **SST-2** (Table 6.6), **Emotion** (Table 6.7) and **TREC** (Table 6.8) classification tasks all demonstrate the effectiveness of the augmentation strategy. Similar to **HumAID**, in all cases STA achieves state-of-the-art performance for text augmentation across all low-resource settings. When a higher number of samples (50-100) are used for training, STA is better, as in the cases of **SST-2** and **EMOTION** tasks, or competitive, as in the case of **TREC**. Besides this, the performance of STA is superior to that of the other augmentation techniques in very low data regimes (5-10-20). In fact, across the three tasks, STA on average demonstrates a difference of +6.4 and +3.7 when trained on only 5 and 10 samples per class respectively. To summarise, the results demonstrate the ability of STA to generate salient and effective training samples from limited amounts of data for text classification tasks other than crisis-based.

In the ablation study, it is noticed that the self-checking step and templates used for STA help to improve the downstream classification performance (rows “STA-noself” and “STA-twoprompts”). This is consistent with the results for **HumAID** (see Section 6.2.3.2). In the investigation of semantic fidelity and lexical diversity, Table 6.9 shows the performance of the fully-trained classifier on the test set predicted by BERT that is trained on the whole training data for measuring semantic fidelity. Figure 6.3 depicts the diversity versus semantic fidelity of generated data by various augmentation methods and STA across the three datasets. As can be seen from its position at the top-right of Figure 6.3a, 6.3b and 6.3c, STA again achieves the best combination of semantic fidelity and lexical diversity compared to alternative approaches.

Augmentation Method	5	10	20	50	100
Baseline (No Aug.)	56.5 (3.8)	63.1 (4.1)	68.7 (5.1)	81.9 (2.9)	85.8 (0.8)
EDA [147]	59.7 (4.1)	66.6 (4.7)	73.7 (5.6)	83.2 (1.5)	86.0 (1.4)
BT [30]	59.6 (4.2)	67.9 (5.3)	73.7 (5.8)	82.9 (1.9)	86.0 (1.2)
BT-Hops [124]	59.1 (4.6)	67.1 (5.2)	73.4 (5.2)	82.4 (2.0)	85.8 (1.1)
CBERT [151]	59.8 (3.7)	66.3 (6.8)	72.9 (4.9)	82.5 (2.5)	85.6 (1.2)
GPT-2 [62]	53.9 (2.8)	62.5 (3.8)	69.4 (4.6)	82.4 (1.7)	85.0 (1.7)
GPT-2-λ [4]	55.4 (4.8)	65.9 (4.3)	76.2 (5.6)	84.5 (1.4)	86.4 (0.6)
BART-Span [62]	60.0 (3.7)	69.0 (4.7)	78.4 (5.0)	83.8 (2.0)	85.8 (1.0)
STA-noself	66.7 (5.0)	77.1 (4.7)	81.8 (2.1)	84.8 (1.0)	85.7 (1.0)
STA-twoprompts	69.8 (4.9)	79.1 (3.4)	81.7 (4.5)	<b>86.0 (0.8)</b>	<b>87.5 (0.6)</b>
<b>STA</b>	<b>72.8 (6.2)</b>	<b>81.4 (2.6)</b>	<b>84.2 (1.8)</b>	<b>86.0 (0.8)</b>	87.2 (0.6)

Table 6.6: STA on **SST-2** in 5, 10, 20, 50, 100 examples per class

Augmentation Method	5	10	20	50	100
Baseline (No Aug.)	26.7 (8.5)	28.5 (6.3)	32.4 (3.9)	59.0 (2.6)	74.7 (1.7)
EDA	30.1 (6.2)	33.1 (4.3)	47.5 (5.0)	66.7 (2.7)	77.4 (1.8)
BT	32.0 (3.0)	37.4 (3.0)	48.5 (5.1)	65.5 (2.0)	75.6 (1.6)
BT-Hops	31.3 (2.6)	37.1 (4.6)	49.1 (3.5)	65.0 (2.3)	75.0 (1.5)
CBERT	29.2 (6.5)	32.6 (3.9)	44.1 (5.2)	62.1 (2.0)	75.5 (2.2)
GPT-2	28.4 (8.5)	31.3 (3.5)	39.0 (4.1)	57.1 (3.1)	69.9 (1.3)
GPT-2- $\lambda$	28.6 (5.1)	30.8 (3.1)	43.3 (7.5)	71.6 (1.5)	80.7 (0.4)
BART-Span	29.9 (4.5)	35.4 (5.7)	46.4 (3.9)	70.9 (1.5)	77.8 (1.0)
STA-noself	34.0 (4.0)	41.4 (5.5)	53.3 (2.2)	65.1 (2.3)	74.0 (1.1)
STA-twoprompts	41.8 (6.1)	56.2 (3.0)	<b>64.9 (3.3)</b>	75.1 (1.5)	81.3 (0.7)
<b>STA</b>	<b>43.8 (6.9)</b>	<b>57.8 (3.7)</b>	64.1 (2.1)	<b>75.3 (1.8)</b>	<b>81.5 (1.1)</b>

Table 6.7: STA on **Emotion** in 5, 10, 20, 50, 100 examples per class

Augmentation Method	5	10	20	50	100
Baseline (No Aug.)	33.9 (10.4)	55.8 (6.2)	71.3 (6.3)	87.9 (3.1)	93.2 (0.7)
EDA	54.1 (7.7)	70.6 (5.7)	79.5 (3.4)	89.3 (1.9)	92.3 (1.1)
BT	56.0 (8.7)	67.0 (4.1)	79.4 (4.8)	89.0 (2.4)	92.7 (0.8)
BT-Hops	53.8 (8.2)	67.7 (5.1)	78.7 (5.6)	88.0 (2.3)	91.8 (0.9)
CBERT	52.2 (9.8)	67.0 (7.1)	78.0 (5.3)	89.1 (2.5)	92.6 (1.1)
GPT-2	47.6 (7.9)	67.7 (4.9)	76.9 (5.6)	87.8 (2.4)	91.6 (1.1)
GPT-2- $\lambda$	49.6 (11.0)	70.2 (5.8)	80.9 (4.4)	<b>89.6 (2.2)</b>	<b>93.5 (0.8)</b>
BART-Span	55.0 (9.9)	65.9 (6.7)	77.1 (5.5)	88.38 (3.4)	92.7 (1.6)
STA-noself	45.4 (3.2)	61.9 (10.2)	77.2 (5.5)	88.3 (1.2)	91.7 (0.8)
STA-twoprompts	49.6 (9.0)	69.1 (8.0)	81.0 (5.9)	89.4 (3.0)	93.1 (0.9)
<b>STA</b>	<b>59.6 (7.4)</b>	<b>70.9 (6.6)</b>	<b>81.1 (3.9)</b>	89.1 (2.7)	93.2 (0.8)

Table 6.8: STA on **TREC** in 5, 10, 20, 50, 100 examples per class

	SST-2	Emotion	TREC
Test	91.8	93.5	96.6

Table 6.9: Accuracy (in %) on test set predicted by BERT that is trained on the whole training data for measuring semantic fidelity.

## 6.2.5 Summary

This section introduced a novel strategy (named STA) for text-based data augmentation that used pattern-exploiting training to generate training samples and achieve better label alignment. STA substantially outperformed the previous state-of-the-art augmentation approaches on the crisis tweet classification task across a range of low-resource scenarios. In addition, experiments also showed that STA generalised well to other domains such as sentiment, emotion, and topic classifications. Furthermore, an analysis of the lexical diversity and label consistency of generated samples was provided, demonstrating that the approach produces uniquely varied training samples

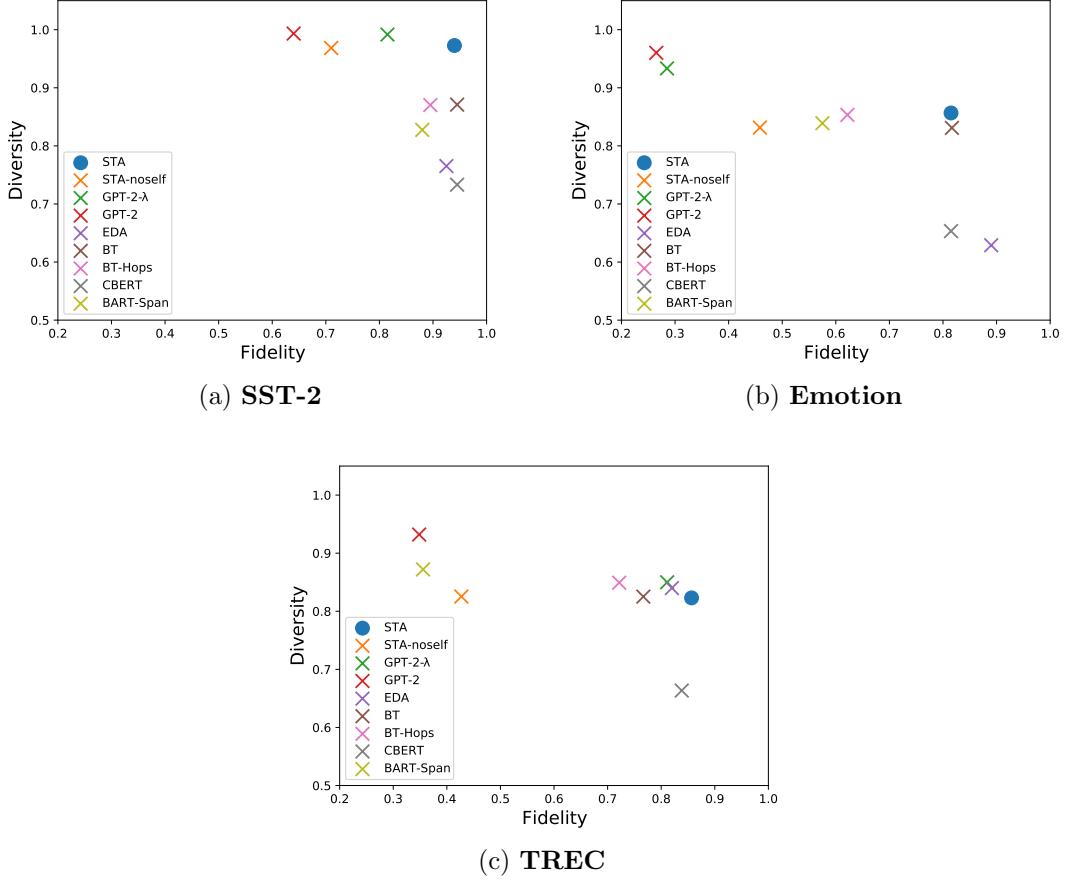


Figure 6.3: Diversity versus semantic fidelity of generated texts by various augmentation methods

with more consistent label alignment than previous work.

This work explored the possibility of using data augmentation for boosting text classification performance when the downstream model was finetuned using pre-trained language models. The results showed that STA consistently performed well across different domains using the same experimental setup, which addressed the limitation stated in the previous work [62] calling for a unified data augmentation technique. However, there remains room to improve STA. In STA, new samples are generated and self-checked using a one-time fine-tuned seq2seq model. This raises the question of whether iteratively fine-tuning the seq2seq model would be beneficial. Additionally, the samples generated by STA are not checked for duplicates among themselves. This prompts the question of whether adding a deduplication component would be useful. The iterative self-controlled augmentation (ISA) method is introduced next as an optimised work of STA.

## 6.3 Iterative self-controlled augmentation

STA is demonstrated to be effective in the crisis message categorisation domain as well as for other text classification tasks. In particular, the self-checking mechanism helps to improve the quality of the generated texts by STA in terms of semantic fidelity. To ensure better quality of generated texts for both lexical diversity and semantic fidelity, iterative self-controlled augmentation (ISA) is introduced in this section. ISA is an optimised version of STA that adds two extra steps to the text generation process: an iterative step and a de-duplication step. Figure 6.4 depicts the overview of ISA and Algorithm 2 describes the overall process of using ISA for few-shot text classification in simple mathematical terms. The text in bold indicates the difference introduced by ISA as compared to STA (see Algorithm 1).

---

**Algorithm 2 : Iterative Self-Controlled Text Augmentation (ISA)**

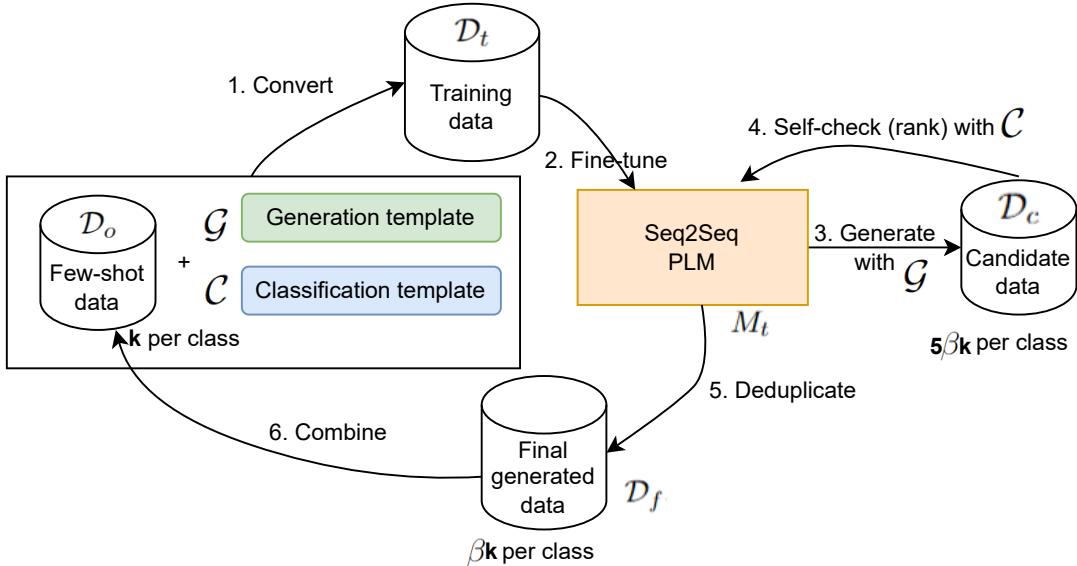
---

**Require:** Few-shot labelled data  $\mathcal{D}_o$ . Generation model  $M$ . Generation template  $\mathcal{G}$ . Classification template  $\mathcal{C}$ . **Unlabelled target data**  $\mathcal{D}_u$ .

- 1: Convert  $\mathcal{D}_o$  to training dataset  $\mathcal{D}_t$  via  $\mathcal{G}$  and  $\mathcal{C}$ .
- 2: Finetune  $M$  on  $\mathcal{D}_t$  in a generation task and a classification task jointly to obtain  $M_t$ .
- 3: Use  $\mathcal{G}$  and  $M_t$  to generate candidate dataset  $\mathcal{D}_c$ .
- 4: Apply  $M_t$  to do classification inference on  $\mathcal{D}_c$  with  $\mathcal{C}$  to **rank  $\mathcal{D}_c$  by prediction confidence**.
- 5: **De-duplicate the confidence-ranked samples through semantic similarity checking to get final generated dataset  $\mathcal{D}_f$ .**
- 6: Combine the final generated dataset with the selected samples.
- 7: **Repeat step 1 to 6  $n$  times to get the final augmented training data  $\mathcal{D}^*$ .**
- 8: Use the augmented data for downstream classification model training
- 9: **Use the trained downstream model to make predictions for the unlabelled target data  $\mathcal{D}_u$ .**
- 10: **Select high-confidence predicted samples for model refinement.**

---

For ISA, to get the augmented data (step 1 – 7), first a seq2seq transformer (generation model) is fine-tuned on the training data that is converted from the original few-shot data by the generation templates and classification templates. The fine-tuned seq2seq model is then used for generating new samples and selecting samples with high confidence of being faithful to the appropriate label. Next, to increase the diversity of the selected samples, a de-duplication step is taken to check their semantic similarities. The selected data is then combined with the few-shot data for next round of model training and new samples are generated. This iterative process is repeated  $n$  times to generate the augmented data for downstream classification training. When training the downstream classifier, it is first trained with the augmented data and then refined on the unlabelled data with high-confidence predictions by the classifier itself (step 8 – 10). The following section introduces the details of ISA.



i: Iterative Self-controlled Augmentation (ISA): 7. repeat step 1-6  $n$  times to get augmented data

ii: Downstream classifier training: using augmented data and unlabeled training data

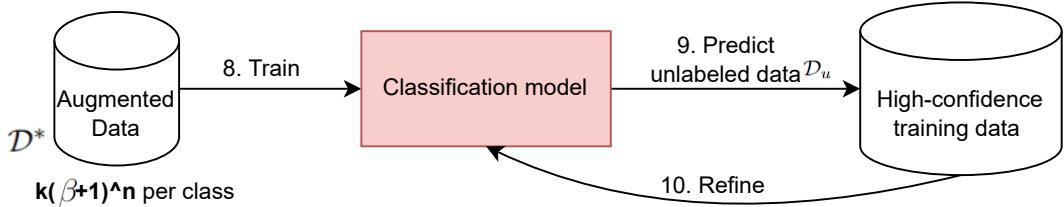


Figure 6.4: The overview of Iterative Self-controlled Augmentation (ISA) approach for few-shot text classification

### 6.3.1 Iterative self-controlled method

Given the difference of ISA compared to STA, this section is divided into three components describing ISA for crisis few-shot text classification: iterative augmentation, generation de-duplication, and model refinement.

#### 6.3.1.1 Iterative augmentation

In a round of iterative augmentation, let  $\mathcal{D}_o$  be the initial dataset for few-shot text classification, known as the seed or few-shot data. It consists of  $k$  samples per class. Similar to STA,  $\mathcal{D}_o$  is first converted to the training data  $\mathcal{D}_t$  by the generation templates  $\mathcal{G}$  and classification templates  $\mathcal{C}$  (see Table 6.1). The seq2seq model is then fine-tuned on this training data and used to generate a candidate dataset  $\mathcal{D}_c$  that consists of  $5\beta k$  samples per class where  $\beta$  refers to the augmentation factor (i.e., how many times more samples are generated for the original  $k$  samples per class). By applying the self-checking mechanism of STA on  $\mathcal{D}_c$ , the  $5\beta k$  samples per class of  $\mathcal{D}_c$  are ranked by the confidence scores predicted by the seq2seq model (See Section 6.2.1.2), represented

by  $\mathcal{D}_{\hat{c}}$ . Then  $\mathcal{D}_{\hat{c}}$  is de-duplicated by semantic similarity checking, forming the final generated data of this round  $\mathcal{D}_f$ , consisting of  $\beta k$  samples per class (introduced later in Section 6.3.1.2). This indicates that the de-duplication step is used to select only the top 20% dissimilar samples from  $\mathcal{D}_{\hat{c}}$  and add them to  $\mathcal{D}_f$ . After obtaining the selected augmented data  $\mathcal{D}_f$  in one round, the union of  $\mathcal{D}_f$  and  $\mathcal{D}_0$  containing  $k(\beta + 1)$  samples per class becomes the training set for the next round of iteration. In ISA, the iteration continues until  $n$  rounds have been completed. After  $n$  rounds of iterations, mathematically, this leads to the final augmented data  $\mathcal{D}^*$  that comprises  $k(\beta + 1)^n$  samples per class. After the iterative augmentation obtaining  $\mathcal{D}^*$ , the next step is to use it along with an unlabelled dataset  $\mathcal{D}_u$  for downstream model training, introduced in Section 6.3.1.3.

### 6.3.1.2 Generation de-duplication

The purpose of de-duplication is to select samples from the confidence-ranked dataset  $\mathcal{D}_{\hat{c}}$  that are as diverse as possible (taking a single iterative round as an example). After the de-duplication, only  $\beta$  samples per class are obtained from the  $5\beta k$  samples per class of  $\mathcal{D}_{\hat{c}}$ , forming the final selected augmented data  $\mathcal{D}_f$ . Algorithm 3 presents the details of the de-duplication process. In this de-duplication,  $\mathcal{D}_r$  is a copy of the seed data  $\mathcal{D}_o$  used as the reference data for checking the semantic similarity between each sample  $x_i$  from  $\mathcal{D}_{\hat{c}}$  and each sample  $x_j$  of  $\mathcal{D}_r$ <sup>7</sup>. To measure the similarity, cosine similarity is used and calculated as follows.

$$s_{i,j} = \text{cosine}(\mathcal{V}(x_i), \mathcal{V}(x_j)), \text{ where } x_i \in \mathcal{D}_{\hat{c}} \text{ and } x_j \in \mathcal{D}_r \quad (6.2)$$

where  $\mathcal{V}$  is a sentence embedding model for vectorising  $x_i$  and  $x_j$ . To determine if  $x_i$  is added to  $\mathcal{D}_f$ ,  $x_i$  is matched with all samples of  $\mathcal{D}_r$  to get the similarity collection:  $S_i = \{s_{i,j}\}|_{j=1}^{\text{len}(\mathcal{D}_r)}$ . To de-duplicate,  $x_i$  is added to  $\mathcal{D}_f$  only when the maximum similarity score is less than or equal to a threshold  $\epsilon$ , formulated as follows.

$$\mathcal{D}_f = \mathcal{D}_f \cup \{x_i\}, \text{ if } \max(S_i) \leq \epsilon \quad (6.3)$$

Once  $x_i$  is selected based on this check, it is also added to the reference data  $\mathcal{D}_r$  for next candidate checking. With this de-duplication, the selected samples in  $\mathcal{D}_f$  are diversified between themselves and also become dissimilar to the seed samples in terms of semantics.

---

<sup>7</sup>Making  $\mathcal{D}_r$  a copy of  $\mathcal{D}_o$  is because  $\mathcal{D}_r$  is dynamically updated in checking against  $\mathcal{D}_{\hat{c}}$  and  $\mathcal{D}_o$  must remain unchanged in order to be combined with  $\mathcal{D}_f$  at the end of one round.

---

**Algorithm 3** Generation De-duplication

---

**Input:** Original few-shot data:  $\mathcal{D}_o$ , Confidence-ranked data:  $\mathcal{D}_c$ , Embedding model:  $\mathcal{V}$

**Output:** Final generated data:  $\mathcal{D}_f$ . No. of candidates for deduplication:  $5\beta k$ , No. after deduplication (20%):  $\beta k$

```
 $\mathcal{D}_r \leftarrow \mathcal{D}_o$ 
 $i \leftarrow 0$ 
 $\mathcal{D}_f \leftarrow \emptyset$ 
while  $i \neq 5\beta k$  do
     $j \leftarrow 0$ 
     $S_i \leftarrow \emptyset$ 
    while  $j \neq \text{len}(\mathcal{D}_r)$  do
         $s_{i,j} \leftarrow \text{cosine}(\mathcal{V}(x_i), \mathcal{V}(x_j))$ , where  $x_i \in \mathcal{D}_c$  and  $x_j \in \mathcal{D}_r$ 
         $S_i \leftarrow S_i \cup \{s_{i,j}\}$ 
         $j \leftarrow j + 1$ 
    end while
    if  $\max(S_i) \leq \epsilon$  then
         $\mathcal{D}_f \leftarrow \mathcal{D}_f \cup \{x_i\}$  #  $x_i$  is selected and added to  $\mathcal{D}_f$  if it hits a low similarity score.
    end if
    if  $\text{len}(\mathcal{D}_f) \geq \beta k$  then
        break # this indicates 20% out of candidates are selected after de-duplication.
    end if
     $i \leftarrow i + 1$ 
end while
```

---

### 6.3.1.3 Model refinement using unlabelled data

After the iterative augmentation with de-duplication, the final augmented dataset  $\mathcal{D}^*$  is obtained for downstream model training. Apart from being trained on  $\mathcal{D}^*$ , the downstream model is also refined on an unlabelled corpus from the target task  $\mathcal{D}^u$ . The lower part of Figure 6.4 shows that the process of downstream model refinement is iterative. Given a downstream model, it is first trained on  $\mathcal{D}^*$ . Then the model is used to make predictions for the unlabelled samples of  $\mathcal{D}^u$ . Among the predicted samples, only the samples with high confidence are selected for refinement (i.e., as the training data for the model in next iteration). The confidence of a sample is determined by its predicted probability by the model and the sample is selected when its confidence is above a threshold  $\sigma$ . For ISA, the refinement continues to iterate while high-confidence samples continue to be identified. The idea behind refinement is to enable the model to self-learn from its own predictions on the unlabelled samples (also known as “self-training” in the literature [89, 90]).

## 6.3.2 Experiments

The purpose of the experiments in STA was to compare its performance to augmentation-based few-shot baselines, using synthetic data generated by STA and the baselines to see how it impacted downstream classification performance. However, it is not yet clear how STA performs compared to prompt-based approaches (Section 3.2.1). Therefore in the following experiments both STA and ISA are compared to these prompt-based approaches. To ensure a fair and parallel comparison, STA and ISA are run using the same experimental setup as the prompt-based approaches, including the use of certain language models and the number of random sampling. In addition, ISA is also compared to the state-of-the-art augmentation-based STA to determine if the synthetic data generated by ISA is an improvement over STA. The following sections provide more details on the experimental setup.

### 6.3.2.1 Experimental Setup

For the generation seq2seq model, instead of the base version for STA, the large version of T5, `t5-large` is chosen in the experiments for ISA. The hyperparameters of fine-tuning `t5-large` remain the same as for STA. To decide the iteration number  $n$ , a grid search over  $n \in \{1, 2, 3, 4, 5\}$  is conducted and finally it is set to be 3 based on its overall good performance with this setup. The augmentation factor  $m$  is set to be 5 based on a search over  $m \in \{1, 2, 3, 4, 5, 6, 7\}$ . Since ISA is proposed for few-shot text classification, the experiments are run in very low-data regimes: the seed data consists of 5 samples per class ( $k_1 = 5$ ). The seed data is sampled from the training set of the target task and the rest is treated as the unlabelled data for model refinement (i.e.,  $D_u$ ).

The de-duplication threshold  $\epsilon$  is set to be 0.9 and the threshold  $\sigma$  for downstream model refinement is set to be 0.95 given their overall good performance in initial experiments. For downstream model training, the hyperparameters remain the same as for STA but the initial model checkpoint is changed from `bert-base` to `roberta-large`<sup>8</sup>, so as to be consistent with the few-shot literature [35, 120, 161]. In the experiments, the parameters are validated based on the final refined model on a development set. It can be challenging to decide the development set as a small such set with great randomness makes it difficult to find the best model checkpoint while a large such set leads to deviation from the real-world few-shot use cases [35]. To overcome this problem 50 examples per class are sampled from the original development set to form the few-shot development set for parameter validation. Additionally, to alleviate the instability of training on small datasets, the experiments are run on 5 splits of the seed

---

<sup>8</sup><https://huggingface.co/roberta-large>

data sampled with different random seeds and hence the performance is reported as the average accuracy score with standard deviation over the 5 runs.

### 6.3.2.2 Baselines

Apart from the baseline without augmentation (No Aug.), STA and three state-of-the-art prompt-based few-shot learning approaches are selected from the literature (Section 3.2.1) to which ISA is compared in the experiments: PET [120], LM-BFF [35], and DART [161].

- **PET** is a semi-supervised approach that fine-tunes language models on a task using cloze-style questions. Hand-crafted prompt templates convert the training data, and multiple models are trained to leverage unlabelled data. The ensemble of models is then used to soft-label unlabelled data.
- **LM-BFF** is a technique for improved few-shot fine-tuning of medium or small-sized language models. Task demonstrations are added to template-based prompts for language model training. Unlike PET, LM-BFF uses automatically generated templates from a seq2seq language model (T5), instead of hand-designed templates.
- **DART** employs unused tokens as differentiable template and label tokens for backpropagation optimization. Its differentiable prompts for few-shot learning aim to generate more discriminative representations than PET’s fixed prompts.

STA is implemented following the same experimental setup as ISA, namely, 5 random runs in 5 examples per class setting using the large T5 and RoBERTa instead of the base T5 and BERT (as were used in the previous experiments). The three prompt-based baselines are common in making relatively smaller language models for better few-shot learners<sup>9</sup>. To adapt them to the crisis categorisation domain, they are implemented by running the official code released by PET<sup>10</sup>, LM-BFF<sup>11</sup>, and DART<sup>12</sup> on target datasets in the few-shot setting of 5 examples per class, as for STA and ISA.

---

<sup>9</sup>This is opposed to large-scale language models such as GPT-3 [11]. Although they exhibit state-of-the-art few-shot classification performance, considering fair comparison and the difficulty in re-implementing them, they are not used as baselines in the experiments.

<sup>10</sup><https://github.com/timoschick/pet>

<sup>11</sup><https://github.com/princeton-nlp/LM-BFF>

<sup>12</sup><https://github.com/zjunlp/DART>

### 6.3.3 Results and discussion

In a similar way to STA, ISA is first tested on the crisis message categorisation task represented by the **HumAID** dataset [3] and then on other domain tasks such as topic classification or sentiment classification represented by **Emotion** [119], **AgNews** [162], **TweetSentiment** [115] and **TREC** [75]. This section reports ISA’s performance on these datasets as compared to the baselines and discusses the quality of generated texts by ISA as compared to STA.

#### 6.3.3.1 Downstream classification

Table 6.10 presents the accuracy and standard deviation (in parentheses) of ISA compared to the baselines on **HumAID** based on 5 runs with 5 examples per class using `roberta-large`. The first row denoted as Baseline (No Aug.) is a simple baseline that uses no augmented data but the original seed data for downstream model training. The table shows that the iterative self-controlled method with de-duplication to augment the seed data in ISA helps improve the downstream performance substantially as compared to the baseline without augmentation and STA with only self-controlled augmentation (78.0 versus 57.0 and 78.0 versus 72.2 respectively). In addition, although STA hits a comparable score with the best-performing prompt-based approach PET (72.2 vs 73.8), it is also found that ISA outperforms PET by a large margin, as indicated by the +4.2 difference (78 vs 73.8). To examine the effect of model refinement on ISA (Section 6.3.1.3), a run denoted as “ISA-refine” that does not use the unlabelled corpus for model refinement is included also. The results show that the refinement helps improve the downstream performance by a small margin, indicated by 78.0 versus 76.1.

	HumAID
Baseline (No Aug.)	57.0 (7.0)
DART [161]	61.2 (5.1)
PET [120]	73.8 (1.5)
LM-BFF [35]	57.1 (14.0)
STA	72.2 (5.1)
ISA-refine	76.1 (3.3)
ISA	<b>78.0 (2.6)</b>

Table 6.10: Accuracy of ISA comparing to baselines on **HumAID** based on 5 runs

#### 6.3.3.2 Quality of generated texts

ISA achieves strong downstream classification performance. It is interesting to know whether there is a correlation between the downstream performance and the quality of the generated texts. Hence, the investigation of the quality of generated texts by

ISA is conducted by experimentation. The investigation serves to answer two major questions. First, how does the quality of texts generated by ISA comparing to those of STA? Second, what effect does the de-duplication have on ISA regarding the quality of generated texts?

The experiment to answer the first question follows the same methodology as was used for STA (Section 6.2.3.3), i.e, to measure the lexical diversity and semantic fidelity of the generated texts. Likewise, Unique Trigrams is used to measure diversity and fully-trained BERT base is used to measure fidelity. Figure 6.5a plots the scores of lexical diversity and semantic fidelity of texts generated by ISA and STA based on 5 runs. It can be seen that ISA is positioned at the right-top of the graph, indicating its superiority in generating texts with both good lexical diversity and semantic fidelity as compared to STA. Furthermore, to conduct a qualitative analysis of the texts generated by the two methods, Table 6.2 lists five randomly-picked samples generated by STA and ISA and five original samples from the label “missing or found people” of **HumAID**. It reveals that despite good diversity of the generated samples by STA as compared to the original samples, there are some near-duplicates among them, such as #1 and #3 (the second block) both describing “Missing in MD After Earthquake”. By contrast, the generated samples by ISA are not only mutually diverse but also different from the original samples.

To study the effect of de-duplication, the experiment is designed simply to run ISA by setting the augmentation factor  $m$  to be 1 and the iteration number  $n$  to be 5. At the end of each iteration, the average diversity and fidelity scores of generated texts by ISA with and without de-duplication are recorded, which are plotted in Figure 6.5b. The results show that ISA with de-duplication achieves better diversity scores while retaining competitive fidelity scores as compared to ISA without de-duplication. This indicates the positive effect of the de-duplication mechanism used in ISA.

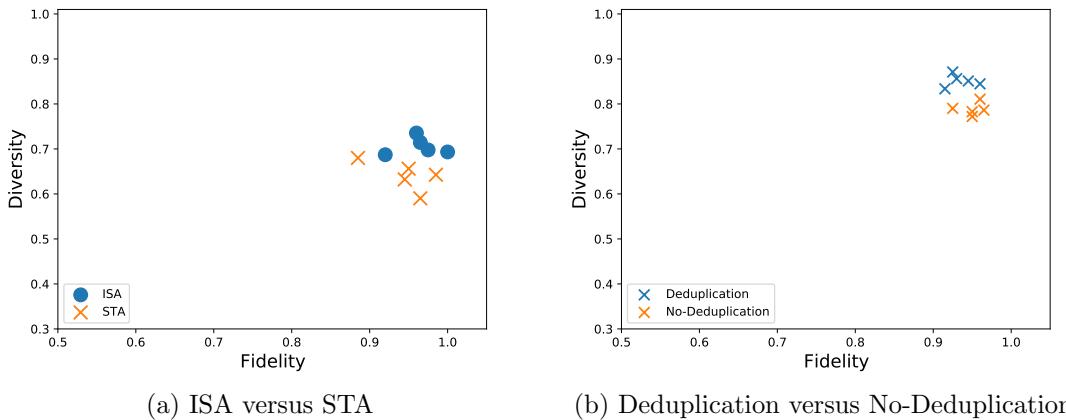


Figure 6.5: The investigation of the quality of generated texts by ISA based on 5 random runs

---

	<p>1: Maryland county grapples with devastation left behind by flash flood One man remained missing after flash flooding tore down historic Main Street in Ellicott Original City, Maryland.</p> <p>2: #California: 200 #missing in wildfires, @BBCWorld reports. #disappeared #MissingPersons</p> <p>3: Catastrophic Floods: 1 Missing in MD From The Weather Channel iPhone App</p> <p>4: TODAY clean up continues in Maryland after the devastating floods.. while the search continues for a missing man. We're tracking the latest on @WMTWTV</p> <p>5: One man is missing after devastating flash flooding swept through Ellicott City, Maryland, turning streets into rivers and carrying away cars.</p>
STA	<p>1: Missing in MD After Earthquake — NBC News — 9to5Mac</p> <p>2: : Said one person is missing after devastating flash flooding swept through Ellicott City, Maryland, turning streets into rivers and carrying away cars.</p> <p>3: Disaster Relief: Missing in MD From The Weather Channel iPhone App</p> <p>4: For information on missing persons in the area of the 2010 earthquake and tsunami click here. #Earthquake</p> <p>5: A large number of people are missing after devastating flash flooding swept through Ellicott City, Maryland, turning streets into rivers and carrying away cars.</p>
ISA	<p>1: In addition to the search for bodies, the FBI launched an extensive manhunt for possible missing people in connection with the devastating 2011 earthquake in Mexico. We are still searching for three missing persons. We will continue to monitor the situation closely.</p> <p>2: RCMP continues search for a missing man after devastating flash flooding swept through Ellicott City, Maryland, turning streets into rivers and carrying away cars.</p> <p>3: Hikers in Mexico City unaccounted for after flash flooding swept through town on Sunday afternoon, leaving behind cars and power lines.</p> <p>4: In the aftermath of Hurricane Florence, more than 2,000 people have been missing since the beginning of the year.</p> <p>5: #Fire and Ice @ChickenFest @CycloneIdai @RachelJ. @CycloneIdai @RachelJ. @CycloneIdai @WWII @WWII @MissingPersons</p>

---

Table 6.11: Original training examples and augmented examples generated by STA and ISA for “missing or found people” of **HumAID** using **t5-large**

### 6.3.3.3 Generalisation to domains beyond crisis

ISA is demonstrated to be very effective in the crisis message categorisation task, similar to STA, it is then tested on more classification tasks. Apart from **Emotion** [119] and **TREC** [75], which are also used in STA, two extra datasets **AgNews** [162] and **TweetSentiment** [115] are added to test ISA in other domains. The former is a news categorisation task and the latter is a ternary sentiment classification task. More details are described in Section 3.4.

Table 6.12 shows the results of ISA compared to baselines on the four datasets. ISA consistently outperforms the baseline without augmentation as well as STA with only

	Emotion	AgNews	TweetSentiment	TREC
Baseline (No Aug.)	30.1 (4.4)	80.3 (3.2)	43.8 (7.2)	65.8 (3.8)
DART [161]	48.1 (2.1)	82.0 (0.9)	54.2 (1.4)	71.3 (2.3)
PET [120]	46.9 (1.2)	<b>86.7 (0.4)</b>	46.0 (0.7)	77.1 (1.7)
LM-BFF [35]	52.9 (7.3)	60.9 (19.8)	51.5 (6.7)	40.2 (7.3)
STA	52.2 (1.4)	83.1 (3.0)	51.7 (5.3)	71.8 (1.6)
ISA-refine	58.8 (2.0)	83.7 (0.8)	<b>57.6 (7.3)</b>	82.6 (3.2)
ISA	<b>61.2 (2.1)</b>	86.1 (0.8)	56.7 (8.8)	<b>84.6 (3.1)</b>

Table 6.12: Accuracy of ISA comparing to baselines on **Emotion**, **AgNews**, **TweetSentiment** and **TREC** based on 5 runs

self-controlled augmentation, implied by the average difference +17.2 and +7.5 respectively. It is also seen that ISA with the refinement outperforms ISA without it (“ISA-refine”) by a small margin, on average +1.5. For the prompt-based few-shot baselines, it is difficult to find one method that performs well across all datasets and instead they each tend to succeed on some datasets while producing poorer results in others. For example, DART performs well in **TweetSentiment** but not for **Emotion**, LM-BFF succeeds in **Emotion** but not for **TREC** and PET is good in **AgNews** and **TREC** but not for the others. However, ISA achieves overall strong performance across all of these datasets, outperforming the prompt-based few-shot baselines substantially in all situations except for a slight loss to PET in AgNews (86.1 versus 86.7). This suggests that ISA is a generalisable approach for text classification of different domains in very low-data regimes.

Regarding the study of the quality of generated texts, the experimental methodology remains the same as for **HumAID**. In measuring semantic fidelity, the accuracy of fully-trained BERT base on the test sets of **AgNews** and **TweetSentiment** is 94.78 and 74.9 respectively. Figure 6.5a shows the results of diversity versus fidelity of texts generated by ISA and STA on datasets beyond crisis. It can be seen that ISA overall achieves better scores in both diversity and fidelity than STA. Figure 6.5b depicts the effect of de-duplication on ISA. It shows that ISA with de-duplication achieves better diversity scores than without de-duplication. This reflects the correlation between the quality of generated texts measured by diversity and fidelity and the downstream performance: the better the quality of generated texts is, the better performance it brings to the downstream model performance.

### 6.3.4 Summary

In this section, ISA is introduced for few-shot text classification. It is built upon STA, adopting an iterative self-controlled method with de-duplication to generate higher-

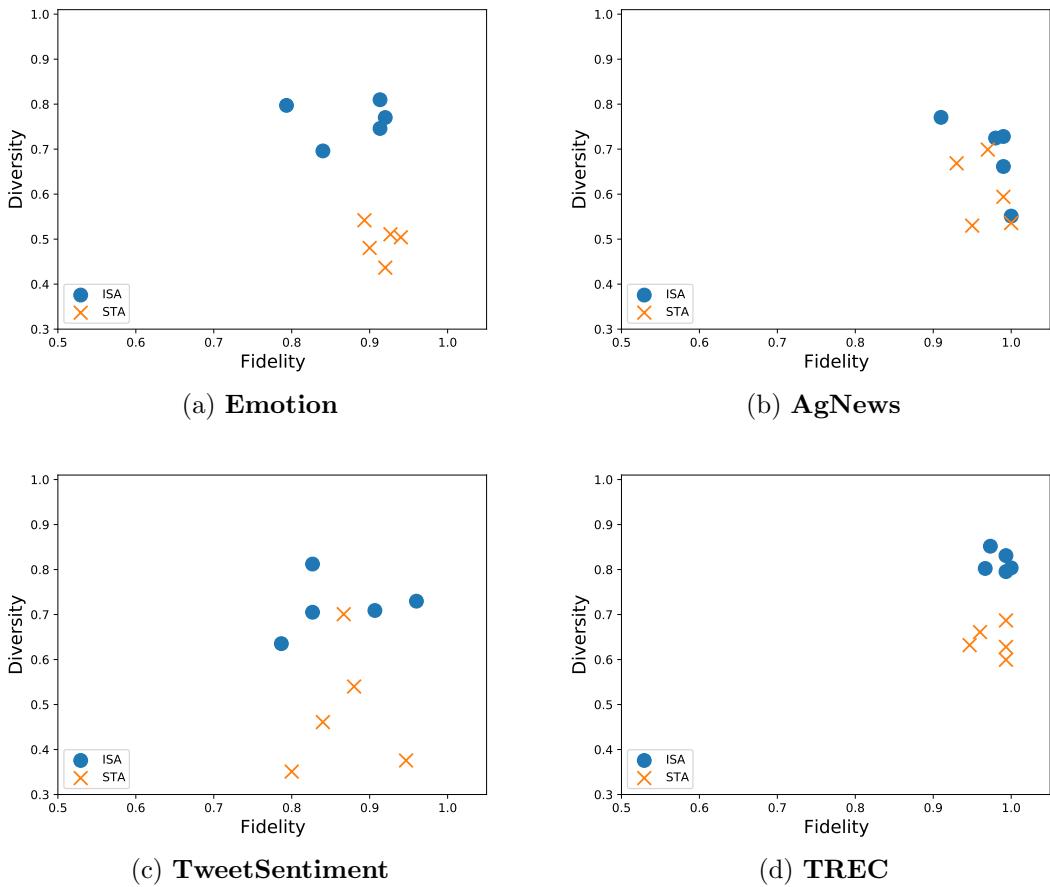


Figure 6.6: Diversity versus semantic fidelity based on 5 runs

quality samples for downstream model training. The results show that it outperforms STA as well as several state-of-the-art prompt-based few-shot baselines in the crisis message categorisation domain as well as for other domains such as sentiment and news classification. The investigation of the quality of texts generated by ISA reveals that ISA is superior in generating texts with good lexical diversity and semantic fidelity compared to the other approaches. The correlation between the quality of generated texts and downstream performance indicates that, for a generation-based augmentation approach for few-shot text classification it is important to ensure good lexical diversity and semantic fidelity for the generated texts. Hence, the future work can aim to further optimise the approach by controlling the quality of generated texts both in terms of lexical diversity and semantic fidelity.

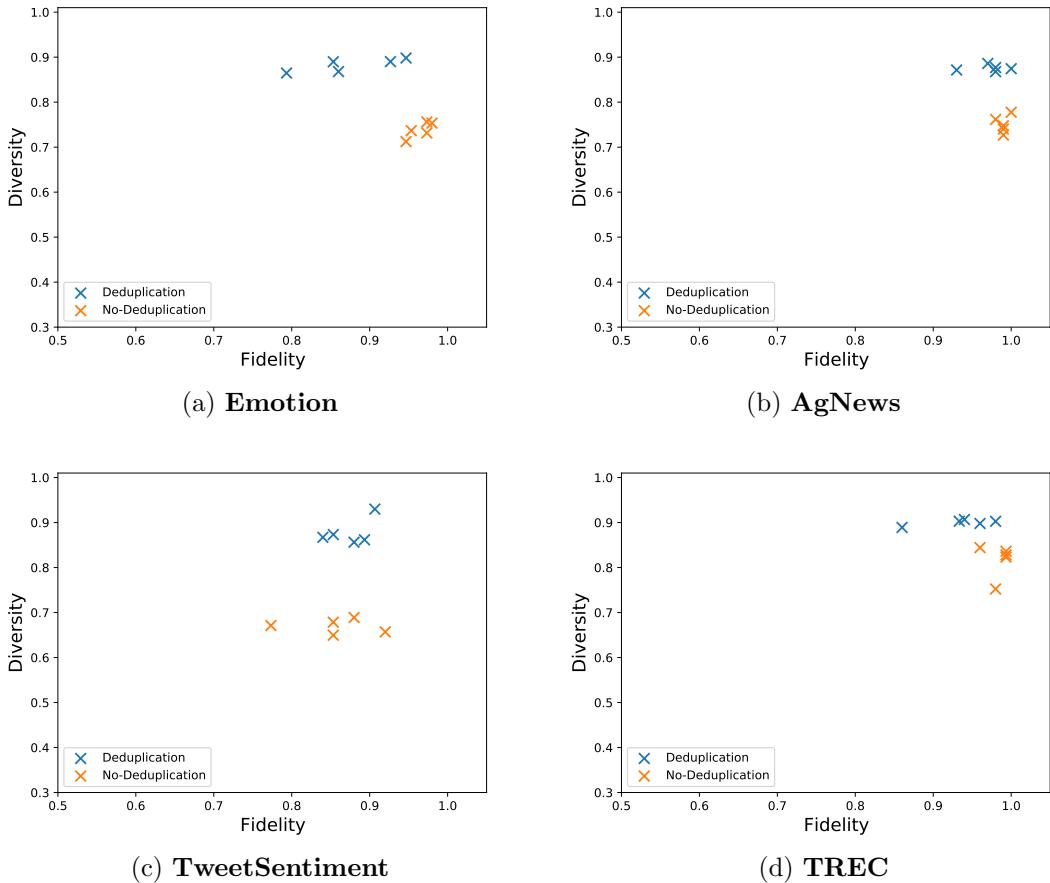


Figure 6.7: Deduplication versus No-Deduplication for ISA based on 5 runs

## 6.4 Conclusions

Various approaches have been proposed for few-shot text classifications within the NLP community and these can be directly adapted to the crisis message categorisation domain [35, 120, 161]. This chapter introduced two novel augmentation approaches for crisis few-shot classification. STA is a self-controlled augmentation method that uses seq2seq pre-trained language models to generate new crisis messages based on only a small quantity of initial seed messages. It is motivated by and developed with the objective of optimising the quality of generated messages, i.e., the lexical diversity and semantic fidelity (label alignment). Experiments comparing STA to existing augmentation baselines showed that STA is capable of generating new crisis messages with superior diversity and fidelity compared to the baselines.

Following this, ISA was proposed, which introduces an iterative mechanism and a deduplication mechanism to STA, aiming to further improve the quality of generated messages. The results indicate that ISA outperforms STA in terms of both diversity and fidelity, leading to improved downstream classification performance. STA was also shown to achieve comparable performance with the best prompt-based method, while

ISA demonstrated the best performance among all the approaches evaluated for crisis message categorisation in few-shot settings. Not only were STA and ISA empirically effective in few-shot crisis message categorisation, further experiments also showed that they had strong generalisation capability to other text classification domains such as emotion or topic classifications.

For crisis message categorisation, STA and ISA use a small quantity of annotated samples (performing well with as few as 5 labelled examples per class) and greater quantities of unlabelled data from the target crisis to generate more data for downstream model training. Although the few-shot seed data is small and can be annotated within a short time in real-world crisis response, it becomes time expensive when the number of pre-defined crisis-related aid classes become large. This is because STA and ISA requires a few annotated messages for every class in order to ensure that new messages can be generated for every class. Additionally, crisis few-short learning assumes equal class distribution, i.e,  $k$  examples per class, which usually does not reflect the real-world distribution.

This raises the question of whether it is possible to completely eliminate the need for annotated target data. In the next chapter, the research will introduce a study on crisis zero-shot learning, which aims to categorise crisis messages without any annotated data. To address this challenge, the study will propose a novel approach called P-ZSC that uses pseudo-labelled data for zero-shot crisis message categorisation.

# Chapter 7: Using Pseudo-labelled Data For Crisis Zero-shot Learning

---

The previous chapter explored the use of two augmentation approaches for categorising crisis messages that require only a few annotated messages per aid type from emerging events. The annotation is relatively easy and efficient when there are only a few aid types, but becomes expensive when many aid types are of interest. Additionally, the methods assume that crisis messages are evenly distributed across aid types, which is rarely the case in real-world scenarios where the class distribution is often influenced by the characteristics of the crisis. For example, during a terrorist attack, requests for blood donations may be more common than requests for shelter or food, whereas the opposite may be true during a storm or flood. These considerations have motivated the current research to focus on crisis message categorisation using zero-shot learning, which involves using only easy-to-obtain unlabelled target data from emerging events.

This chapter first introduces the method for crisis zero-shot learning called P-ZSC (Pseudo-labelled Zero-Shot Categorization) (Section 7.1). The approach involves using pseudo-labelled data to train a model followed by a refinement process for crisis message categorisation (Section 7.2). To obtain the pseudo-labelled data, each sample of the unlabelled data is matched with the descriptions of aid types represented by their surface names such as “search and rescue”. After this matching, a label expansion component is then applied to generate a label vocabulary for each class in order to enhance the representation of the aid classes (Section 7.2.1). Matching is then conducted between the unlabelled samples and the label vocabularies, rather than the raw label surface names. Samples that achieve a high matching score with a label are assigned that label (Section 7.2.2). The pseudo-labelled set is then used to pre-train a downstream classifier, which is finally self-trained on the unlabelled data (Section 7.2.3). Finally, the results of P-ZSC compared to baselines are reported and discussed in experiments (Section 7.3).

The work presented in this chapter has previously been the subject of a peer-reviewed publication [143].

## 7.1 Introduction

This section introduces P-ZSC, a weakly-supervised method for performing zero-shot learning during a crisis. Building upon previous work (Section 3.3.2), P-ZSC uses a simple and effective algorithm to match unlabelled samples with classes to obtain pseudo-labelled data that can then be used to train the classification model. To avoid exact matching, P-ZSC obtains related phrases from a domain-specific unlabelled corpus to create a vocabulary for each class based on their embeddings. A sample is assigned with a particular class when it overlaps significantly with the vocabulary of that class. P-ZSC also uses self-training, a method that has been demonstrated to be effective [90] to refine the categorisation model on the domain-specific unlabelled data after it has been trained on the pseudo-labelled data. The details of P-ZSC are provided below.

## 7.2 Method

Formally, the problem that the system aims to solve is defined as follows: for a classification task  $\mathcal{T}$  (which can be either a single-label or multi-label task), given  $n$  label names<sup>1</sup>  $\mathcal{Y} : \{y_1, y_2, \dots, y_n\}$  and an unlabelled corpus  $\mathcal{D} : \{d_1, d_2, \dots, d_m\}$  containing  $m$  examples from this task domain, the objective is to train a model  $f$  that can assign one or more labels (depending on whether the task is single-label or multi-label classification) from  $\mathcal{Y}$  to an example  $x$  based on its probability estimation over  $\mathcal{Y}$ , namely,  $f(x) : p(y_1|x), p(y_2|x), \dots, p(y_n|x)$ .

As illustrated in Figure 7.1, the system has three stages: label expansion produces a label vocabulary to represent raw labels, pseudo label assignment matches the unlabelled corpus with label vocabularies to obtain a pseudo-labelled dataset, pre-training and self-training train a classification model on the pseudo-labelled dataset and then refines the model on the unlabelled corpus. These stages are described in detail in the following sections.

---

<sup>1</sup>In the context of a crisis message categorisation tasks, the label names are aid types for emergency response. In this description, more general terms are used to indicate its generalisability to other text classification tasks.

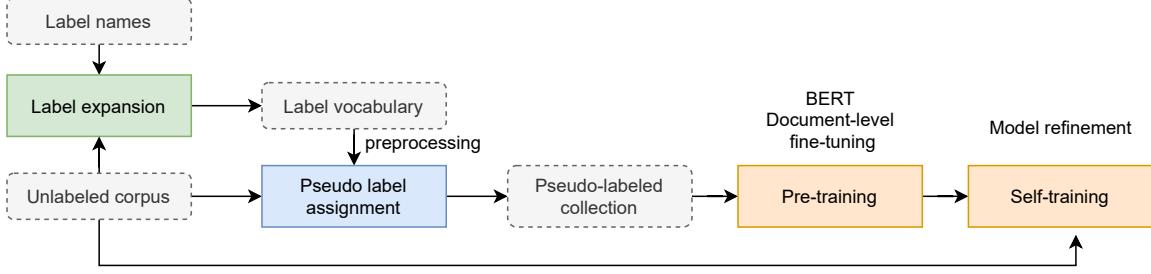


Figure 7.1: The architecture of P-ZSC

## 7.2.1 Label expansion

In a classification task such as crisis message categorisation, the label names tend to consist of only one or two words such as “missing or found”, “goods services” and “donations”, which are insufficient to convey their potentially broad meaning. To enrich a label’s meaning, a simple label expansion (LE) algorithm is proposed to find the most semantically-similar words or phrases to the label from the unlabelled in-domain corpus  $\mathcal{D}$ .

Given  $\mathcal{Y}$  and  $\mathcal{D}$ , first all examples from  $\mathcal{D}$  are combined and split into n-gram phrases consisting of all 1-grams, 2-grams and 3-grams that are found in any examples within  $\mathcal{D}$  (including overlaps):  $\mathcal{G} : \{g_1, g_2, g_3, \dots, g_L\}$ . Then, the similarity between  $y_i \in \mathcal{Y}$  and  $g_i \in \mathcal{G}$  is calculated via a sentence embedding model. Given a sentence embedding model  $E$ ,<sup>2</sup> the matching score  $\hat{s}_{i,j}$  between a label  $y_i$  and an n-gram  $g_j$  is calculated as the cosine similarity between the embeddings of the label and the n-gram respectively:

$$\hat{s}_{i,j} = \text{cosine}(E_{avg}(y_i), E_{avg}(g_j)) \quad (7.1)$$

For any  $y_i$  or  $g_j$  with more than one token,  $E_{avg}$  takes the average pooling as the output embedding. After this, each label  $y_i \in \mathcal{Y}$  has a list of n-grams (i.e. a subset of  $\mathcal{G}$ ) that are ranked by the similarity scores. The vocabulary for each label  $y_i$  is denoted by  $\hat{\mathcal{V}}_i : \{(\hat{v}_{i,k}, \hat{s}_{i,k})\}|_{k=1}^{\hat{K}}$  where  $\hat{v}_{i,k} \in \mathcal{G}$  represents the  $k$ th expanded n-gram in the vocabulary of label  $y_i$  and  $\hat{s}_{i,k}$  is the corresponding matching score.

**Label vocabulary pre-processing:** To maintain the quality of the label vocabulary, a further pre-processing step is taken to optimise the original vocabulary  $\hat{\mathcal{V}}_i$  and a label’s pre-processed vocabulary is denoted as:  $\mathcal{V}_i : \{(v_{i,k}, s_{i,k})\}|_{k=1}^K$ . At this step, only those n-grams in  $\hat{\mathcal{V}}_i$  where  $\hat{s}_{i,k} \geq 0.7$  are selected, maintaining a minimum of 2 n-grams and a maximum of 100 n-grams per label<sup>3</sup>. Then a discounting is applied on

<sup>2</sup>As an example of sentence embeddings models, the widely-applied `deepset/sentence_bert` from the Huggingface model hub [150] is used in this work.

<sup>3</sup>The parameters are fixed for all tasks to ensure actual zero-shot learning.

the n-grams that co-occur across different labels, which is calculated by

$$s_{i,k} = \hat{s}_{i,k} * \log_e \left( \frac{n}{LF(v_{i,k})} \right) \quad (7.2)$$

where  $n$  is the number of labels and  $LF(v_{i,k})$  is the frequency of n-gram  $v_{i,k}$  across the vocabularies of all labels. To know the effect of the label expansion approach on raw surface names, Table 7.1 shows the label vocabularies for the labels from two crisis classification datasets **HumAID** and **Situation**. After this process, the labels are represented by their corresponding vocabularies that convey a broader semantic meaning of their raw surface names. For example, instead of just “infrastructure and utility damage”, it is enriched with new terms such as “property damage”, “power lines”, etc. Notably, the top-ranked expanded phrases mostly consist of 1 and 2-grams, as matching high semantic scores with 3-grams can be challenging due to the complexity of semantic matching. Nevertheless, this work still leverages 3-grams for label expansion, as they may include crucial phrases such as “people evaluated safet”, which can help expand the “evaluation” label well (as shown in Table 7.1).

## 7.2.2 Pseudo label assignment

After expansion, a labelled collection (pseudo-labelled data) is next constructed for model training. Due to the lack of annotated data, the alternative is to construct the collection via the process of pseudo label assignment (PLA). To achieve this, a simple approach for PLA is adopted, which is described as follows:

A document  $d_j \in \mathcal{D}$  is matched with a label’s vocabulary  $\mathcal{V}_i : \{(v_{i,k}, s_{i,k})\}|_{k=1}^K$  (from the previous section) through a cumulative scoring mechanism. To assign a label  $y_i$  to a document  $d_j$ , a matching score between them is first calculated by

$$s_{j,i}^* = \sum_{k=0}^K s_{i,k} \mathbb{I}(v_{i,k} \in \mathcal{G}_j) \quad (7.3)$$

where  $\mathbb{I}(\cdot)$  is the indicator function (evaluating to 1 if  $v_{i,k} \in \mathcal{G}_j$  or 0 otherwise) and  $\mathcal{G}_j$  is the set of n-grams that are contained in document  $d_j$ . For consistency with what was used in label expansion, here the n-grams also range from  $n = 1$  to  $n = 3$ . Now  $s_j^* : \{s_{j,i}^*\}|_{i=0}^n$  refers to the matching score of  $d_j$  with every label from  $\mathcal{Y}$ . To decide if  $d_j$  is assigned one or more labels, a threshold  $\epsilon$  is defined. For single-label tasks, if the maximum value of  $s_j^*$  at index  $i$  is greater than  $\epsilon$ , then  $y_i$  is assigned to  $d_j$ . For

Label	Vocab
rescue volunteering or donation effort	donation help, donate help, help donate, donating help, volunteers helping, volunteering, volunteers help, rescue effort, relief donations, rescue people
sympathy and support	sympathy, condolences, compassion, support please, condolences families, deepest condolences, support relief, please support, provide relief, kindness
requests or urgent needs	urgent need, urgent, needs, need, needing, require, urgently, demand, needed, really need
infrastructure and utility damage	infrastructure damage, infrastructure, power lines, areas impacted, damage caused, damage, property damage, areas affected, power outages, damage amp
injured or dead people	dead people, dead injured, people died, people dead, killed injured, people dying, people killed, died injured, killed people, deaths
caution and advice	advice, careful, warns, warned, safety tips, advised, heed warnings, pray safety, advisory, take care
displaced people and evacuations	displaced people, people displaced, refugees, families displaced, evacuations, evacuation orders, evacuation zones, displaced, evacuation, need evacuate
missing or found people	people missing, missing people, people still missing, reported missing, people lost, number missing, people unaccounted, lost lives, missing, still missing

(a) Label vocabularies for **HumAID** dataset.

Label	Vocab
shelter	shelter, temporary shelter, shelters, shelter food, refuge, food shelter, protection, temporary shelters, makeshift shelters, emergency shelter
search	search, searching, find, search operations, finding, seek, seeking
water	water, water borne, waters, water system, water drinking, water sources, waterborne, water resources, water supplies, water supply
utilities	electricity supply, power lines, power supply, electricity water, electricity, supply systems
terrorism	terrorism, anti terrorism, terrorist groups, terrorist, terrorists, terrorist attack
evacuation	evacuation, evacuate, evacuated, people evacuated, people evacuated safety, evacuated safety, evacuating
regime change	change, changed
food	food, including food, food clothing, food items, food clothes, food work, foods, food supplies, food program, food programme
medical	medical, medical services, medical team, medicine, doctors, medical care, healthcare, medical attention, doctor, hospital
infrastructure	infrastructure, infrastructures, infrastructure including, structures
crime violence	crime, crimes, violence

(b) Label vocabularies for **Situation** dataset.Table 7.1: Top-ranked expanded phrases for crisis-related labels from **HumAID** and **Situation**

multi-label tasks, if the value of  $s_j^*$  at any index is larger than  $\epsilon$ , then the label at that index is assigned to  $d_j$ . Thus only the examples achieving high matching scores (high-confidence) with the labels are likely to be pseudo-labelled. Examples for which no label exceeds the threshold are left unlabelled and are not used in the next stage for pre-training, but can be used for self-training.

At this point, a pseudo-labelled collection is obtained, denoted as  $\hat{\mathcal{D}} : \{(x_i, \mathcal{Y}_i)\}_{i=1}^N$  where  $N$  is the number of pseudo-labelled examples, and  $\mathcal{Y}_i$  refers to the pseudo labels assigned to  $x_i$  and it is a subset of  $\mathcal{Y}$  consisting of a label for single-label tasks and multiple labels for multi-label tasks. To ensure the quality of  $\hat{\mathcal{D}}$ , the threshold  $\epsilon$  should

be chosen carefully. It will result in poor-quality pseudo-labelled data being generated if it is too low, but will result in zero examples for some labels if it is too high. Since the matching score  $s_{i,k}$  is normalised by Equation 7.2,  $\epsilon$  is set to be  $\log_e n$  where  $n$  is the number of labels. However, this value can lead to zero pseudo-labelled examples for insufficiently-expanded labels (e.g., their vocabularies contain few phrases: particularly common in class-imbalanced datasets). Hence,  $\epsilon$  is reduced by half when the label with fewest expanded phrases has fewer than  $10^4$ .

### 7.2.3 Pre-training and self-training

With  $\hat{\mathcal{D}}$  as the supervision data, the next step fits the model  $f$  to the target classification task by learning from the pseudo-labelled data. Instead of training from scratch, the pre-trained `bert-base-uncased` [25] is used as the base model ( $f_\theta^*$ ) and this can easily be replaced by other pre-trained language models [150]. To be specific, for the task  $\mathcal{T}$ , a classification head is added to the base model. The classification head takes the [CLS] output (denoted as  $\mathbf{h}_{[\text{CLS}]}$ ) of the base model as input and outputs the probability distribution over all classes:

$$\begin{aligned} \mathbf{h} &= f_\theta^*(x) \\ p(\mathcal{Y} \mid x) &= \sigma(\mathbf{W}\mathbf{h}_{[\text{CLS}]} + \mathbf{b}) \end{aligned} \tag{7.4}$$

where  $\mathbf{W} \in \mathbb{R}^{h \times n}$  and  $\mathbf{b} \in \mathbb{R}^n$  are the task-specific trainable parameters and bias respectively and  $\sigma$  is the activation function (softmax if  $\mathcal{T}$  is a single-label task, or sigmoid for a multi-label task). In model training, the base model parameters  $\theta$  are optimised along with  $\mathbf{W}$  and  $\mathbf{b}$  with respect to the following cross entropy loss (on the pseudo-labelled data):

$$\mathcal{L}_{pt} = - \sum_{i=0}^n y_i \log p(y_i \mid x) \tag{7.5}$$

Since  $\hat{\mathcal{D}}$  is a subset of  $\mathcal{D}$ , there are many unlabelled examples not seen in the model’s pre-training. As indicated in LOTClass [90], the unlabelled examples can be leveraged to refine the model for better generalisation via self-training. Hence, the unlabelled corpus  $\mathcal{D}$  is then used for model self-training. This is done by splitting it into equal-sized portions (assuming each portion has  $M$  examples) and then each of the examples in a portion is predicted by the model in an iteration with the predictions denoted as

---

<sup>4</sup>This setup is fixed for all tasks and not tuned to different tasks to ensure actual zero-shot learning.

the target distribution  $Q$ . In each iteration, the model is trained on batches of the portion with the current distribution as  $P$ . The model is updated with respect to the following Kullback–Leibler (KL) divergence loss function [56]:

$$\mathcal{L}_{st} = \text{KL}(Q\|P) = \sum_{i=1}^M \sum_{j=1}^n q_{i,j} \log \frac{q_{i,j}}{p_{i,j}} \quad (7.6)$$

where  $q_{i,j}$  and  $p_{i,j}$  refer to the predicted probability of an sample  $d_i$  to an label  $y_j$  of  $Q$  and  $P$  respectively. In deriving the target distribution  $Q$ , it can be applied with either soft labelling [153] or hard labelling [66]. Hard labeling involves using high-confidence predictions as the ground truth labels, which can be prone to error propagation. Soft labeling, on the other hand, calculates the target distribution for each instance, and does not require the use of preset confidence thresholds. As a result, soft labelling overall tends to bring better results [90], therefore the soft labelling strategy is used to derive  $Q$  in this work, formulated as follows:

$$q_{i,j} = \frac{p_{i,j}^2/p_j^*}{\sum_{j'} (p_{ij'}^2/p_{j'}^*)}, p_j^* = \sum_i p_{i,j} \\ p_{i,j} = p(y_j \mid x_i) = \sigma(\mathbf{W}(f_{\theta}^*(x_i))_{[\text{CLS}]} + \mathbf{b}) \quad (7.7)$$

This strategy derives  $Q$  by squaring and normalising the current predictions  $P$ , which helps boost high-confidence predictions while reducing low-confidence predictions.

## 7.3 Experiments

Having described the system components, the following sections describe extensive experiments to demonstrate the effectiveness of the proposed approach.

### 7.3.1 Datasets

In testing P-ZSC for crisis message categorisation, **Situation** [81, 158] is selected in addition to **HumAID** [3]. In contrast with **HumAID**, **Situation** is a multi-label dataset where a crisis tweet is assigned to one or more aid types such as “shelter”, “search”, etc. Details of the datasets can be found in Section 3.4.2. In evaluation, the *label-wise weighted F1* metric is used for **Situation** considering it is a multi-label

class-imbalanced dataset while accuracy is used for **HumAID**, which are also aligned with [158] and [3] respectively. To use them in zero-shot learning settings, the training and development sets of each dataset excluding labels are combined and used as an unlabeled corpus for model training, while the test set is used for testing.

### 7.3.2 Experimental details

In the experiments, to simulate a simple zero-shot setting, the training set of each dataset (pre-split training sets that come with the datasets) is used as the unlabelled corpus  $\mathcal{D}$  and the surface names (e.g. “missing or found”, “shelter”, “search” etc) as the label set  $\mathcal{Y}$ . For model pre-training on the pseudo-labelled data, the **BERT-base-uncased** pre-trained checkpoint is fine-tuned on a batch size of 16 using Adam [59] as the optimiser with a linear warm-up scheduler for increasing the learning rate from 0 to  $5e - 5$  at the first 0.1 of total training steps and then decreasing to 0 for the remaining steps. For self-training, the hyper-parameters remain the same as in [90], with batch size 128 and update interval 50, which results in the number of training examples in each iteration (namely,  $M$ ) being  $50 \times 128$ .

### 7.3.3 Baselines

In these experiments, related existing methods (Section 3.3) are selected as baselines to be compared with P-ZSC. The baselines include indirectly-supervised methods **Label similarity** [133], **Entail-single** [158], **Entail-ensemble** [158], **Entail-Distil** [23] and weakly-supervised methods **ConWea** [88], **X-Class** [146], **WeSTClass** [89] and **LOTClass** [90]. In addition, the fully **Supervised BERT** is added to the list for comparison. The experimental setup for them are described as follows.

- **Label similarity** uses pre-trained word embeddings to compute the cosine similarity between the class label and every 1-gram to 3-gram of the example. In the experiments, BERT base is used as the pre-trained embedding model. For single-label tasks, the label with the highest similarity score is chosen. For multi-label tasks, any label with a similarity score greater than 0.5 is chosen.
- **Entail-single** and **Entail-ensemble** correspond to the best *label-fully-unseen* ZSL single and ensemble results in [158]. As **HumAID** and **Situation** were not used in that paper, their methodology is followed to create a similar setup by fine-tuning three variants of BERT on three inference datasets (RTE/MNLI/FEVER) and then being tested on the datasets. The best of these in each category is

reported as “entail-single” and the ensemble of the three variants is reported as “entail-ensemble”.

- **Entail-Distil** attempts to overcome the inference inefficiency issue in the entailment matching-based methods that require all-versus-all pairwise matching. The training data is pseudo-labelled first by the matching model (`bert-base-uncased` fine-tuned on MNLI) and then the pseudo-labelled data is used for downstream model fine-tuning (`bert-base-uncased`).
- **ConWea** is a contextualised weakly-supervised approach for text classification, which uses a small quantity of human-provided seed words for label expansion. In the experiments, their released code<sup>5</sup> is re-run on the selected datasets using at least 3 seed words. As a comparison, weak human supervision (few seed words) entails this approach unlike P-ZSC using label names only.
- **X-Class** uses label names only by building class-oriented document representations first and then using a Gaussian Mixture Model (GMM) to obtain the pseudo-labelled data. In the experiments, their released code<sup>6</sup> is run on the selected datasets to obtain their corresponding pseudo-labelled for model fine-tuning (`bert-base-uncased`) and the performance is reported on the fine-tuned model.
- **WeSTClass** is configurable to accept up to three sources of supervision. In the experiments, their released code<sup>7</sup> is re-run on the selected datasets using label names as the only supervision resource so as to be consistent with P-ZSC.
- **LOTClass** is another weakly-supervised approach using only label names followed by a self-training component for text classification. In the experiments, their officially-released code<sup>8</sup> is re-run for the selected datasets.
- **Supervised BERT (Sup. BERT)** is included so that a comparison with a fully-supervised approach can be done. This uses `bert-base-uncased`, fine-tuned on the training sets with labels, validated by the development sets and tested on test sets. While a ZSL approach will be unlikely to match the performance of a fully-supervised approach, it is important to illustrate how large that performance gap actually is and to illustrate the degree to which P-ZSC contributes towards closing that gap.

---

<sup>5</sup><https://github.com/dheeraj7596/ConWea>

<sup>6</sup><https://github.com/ZihanWangKi/XClass>

<sup>7</sup><https://github.com/yumeng5/WeSTClass>

<sup>8</sup><https://github.com/yumeng5/LOTClass>

	HumAID Situation	
Indirectly supervised runs		
Label similarity [133]	57.89	40.75
Entail-single [158]	40.96	37.20
Entail-ensemble [158]	41.34	38.00
Entail-Distil [23]	45.81	38.85
Weakly-supervised runs		
ConWea [88]	46.35	25.91
X-Class [146]	45.70	39.27
WeSTClasss [89]	35.39	28.40
LOTClass [90]	15.63	5.85
P-ZSC	<b>67.09</b>	<b>55.02</b>
Sup. BERT	88.89	85.27

Table 7.2: Performance of P-ZSC and baselines on test sets of crisis datasets

### 7.3.4 Results and discussion

In this section, P-ZSC is compared with the baselines in a crisis zero-shot classification setting. To dissect each component of P-ZSC, an ablation study is conducted. Following this, an additional experiment is conducted to investigate the quality of the pseudo-labelled data that is generated by the system. Finally, its generalisation to other domains such as emotion classification is tested and verified.

#### 7.3.4.1 Comparison with baselines

In testing the effectiveness of P-ZSC on the two selected crisis categorisation tasks, the baselines are divided into two categories: indirectly-supervised runs and weakly-supervised runs, as presented in Table 7.2. P-ZSC falls into the category of weakly-supervised runs. The results show that P-ZSC achieves state-of-the-art 67.09 accuracy and 55.02 F1 score for **HumAID** and **Situation** respectively.

P-ZSC substantially outperforms the weakly-supervised methods including Label similarity, Entail-single, Entail-ensemble and Entail-Distil. Label similarity performs well among the baselines of this kind. Although it is a simple matching between the sentence embeddings of label names and document phrases, interestingly it outperforms the entailment methods and Entail-Distil for both datasets. This indicates that semantically matching a document’s phrases with the label names using pre-trained word embeddings can help determine the document’s class. It should also be noted that label similarity and the entailment runs are around  $n$  (number of labels) times slower than Entail-Distil and the weakly-supervised runs. This indicates that P-ZSC as a weakly-supervised approach has advantages in terms of inference efficiency also. Compared to

HumAID Situation		
P-ZSC	67.09	55.02
- self-training	60.73	50.51
- PLA	56.43	46.20
- PLA+LE	55.03	43.26

Table 7.3: Ablation study of P-ZSC on crisis datasets

the best results of weakly-supervised runs (46.35 for **HumAID** by ConWea and 39.27 for **Situation** by X-Class), P-ZSC shows performance improvements of +20.74 and +15.75 for **HumAID** and **Situation** respectively. However, as can be seen from the table, there is still a gap between P-ZSC (unsupervised) and Sup. BERT (the supervised BERT run). This suggests that although the results of P-ZSC constitute substantial progress in crisis zero-shot (label-fully-unseen) text classification, crisis zero-shot learning remains challenging and cannot yet be considered to be a solved problem. On the other hand, the necessity for significant quantities of expensive annotations (both in terms of time and effort) means that a fully-supervised approach is impractical for real-world crisis events.

#### 7.3.4.2 Ablation study

To examine the contribution of each component of P-ZSC, an ablation study is conducted, with the results reported in Table 7.3. This shows the performance of the entire P-ZSC system, and also separate results with the self-training step omitted, with the pseudo-label assignment (PLA) omitted and with both PLA and label expansion (LE) omitted. In each case, the removal of any phase results in a decline in performance for all datasets. This decrease becomes slight when only the self-training phase is removed and becomes much greater when both the PLA and LE phases are removed. This indicates that all phases are important to maintain peak effectiveness.

#### 7.3.4.3 Pseudo-labelled data quality

In the system, the only “supervision” resources for the downstream model training is from the pseudo-labelled data that is obtained via LE and PLA. In the pipeline of P-ZSC, pseudo-labelled data is obtained without any human supervision, using only the label names and an unlabelled corpus relating to the target task. Given the importance of the pseudo-labelled data in the final performance, the pseudo-labelled data is constructed to be a subset of the unlabelled corpus (i.e., only high-confidence label allocations are included in the pseudo-labelled set). To quantify the quality of pseudo-labelled data, an extra experiment is conducted for this purpose using similar methodology to [90]. In this methodology, P-ZSC is compared with the Sup. BERT

run fine-tuned on different numbers of actually-labelled examples per class. From Figure 7.2, it is noticed that the pseudo-labelling of P-ZSC can equal the performance of 23 and 48 actually-labelled examples per class on **HumAID** and **Situation** respectively. Considering that **HumAID** has 8 classes and **Situation** has 11 classes, it therefore accounts for a total of 184 annotated examples for **HumAID** and 528 for **Situation**. This indicates how much human annotation effort can be saved when using P-ZSC for the two crisis categorisation tasks as compared to supervised methods.

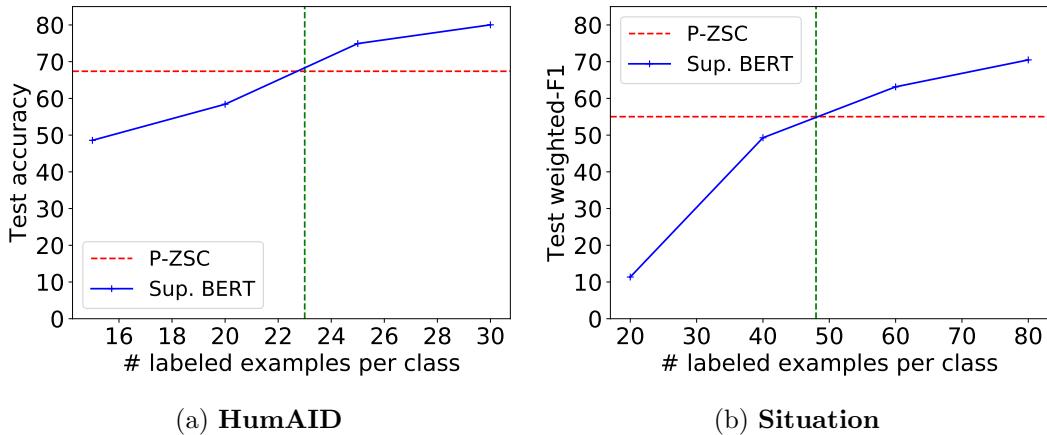


Figure 7.2: The investigation of pseudo-labelled data by P-ZSC on crisis datasets

#### 7.3.4.4 Generalisation to domains beyond crisis

Having verified P-ZSC’s effectiveness on two crisis categorisation tasks, it is interesting to also examine the extent to which it can be generalised to domains beyond crises. Three additional datasets are selected for this purpose: **Topic** [163], **UnifyEmotion** [60] and **Emotion** [119]. More details on the selected datasets can be found in Section 3.4.3. The former two are the benchmarking datasets used in [158] and **Emotion** is another emotion dataset that does not overlap with **UnifyEmotion**. It is noted that **Topic** is class-balanced while the other two are not, and thus the evaluation uses accuracy for measuring P-ZSC’s performance on **Topic** and weighted F1 for the other two. In the experiments, P-ZSC and the selected benchmark approaches are re-run on the three datasets following the same experimental configuration as for the crisis datasets.

Regarding the comparison to baselines, Table 7.4 shows the performance of P-ZSC and baselines on the test sets of the three selected datasets. Overall, P-ZSC outperforms both the indirectly-supervised runs and the weakly-supervised runs. Although it can be seen that the accuracy of P-ZSC on **Topic** is slightly below the best-performing LOT-Class (50.48 vs 52.07), it achieves the highest performance on the other two datasets. On **UnifyEmotion** its performance is similar to the best-performing Entail-Distil, and its **Emotion** performance is substantially better than all of the other approaches.

	Topic	UnifyEmotion	Emotion
<i>Indirectly-supervised runs</i>			
Label similarity [133]	34.62	26.21	56.03
Entail-single [158]	43.80	24.70	49.60
Entail-ensemble [158]	45.70	25.20	50.16
Entail-Distil [23]	44.47	29.43	48.87
<i>Weakly-supervised runs</i>			
ConWea [88]	49.81	21.39	47.34
X-Class [146]	48.12	15.19	42.21
WeSTClass [89]	34.96	15.45	22.54
LOTClass [90]	<b>52.07</b>	7.19	16.82
P-ZSC	50.68	<b>30.22</b>	<b>64.47</b>
Sup. BERT [25]	74.86	40.10	92.02

Table 7.4: Performance of P-ZSC and baselines on test sets of datasets beyond crisis. Due to differing levels of label imbalance, accuracy is used to evaluate the Topic dataset, and weighted F1 is used for the others.

P-ZSC consistently performs well across all datasets and achieves the best overall performance. In contrast, other baselines may perform well on some datasets but not as well on others.

For example, LOTClass performs well in **Topic** but exhibits poorer performance in the other datasets, suggesting that LOTClass does not generalise particularly well. By analysis, to expand labels, LOTClass identifies unlabelled samples with exact-word matches to label names. These are then expanded using BERT masked language modelling (MLM). Masked Category Prediction (MCP) is then used to pseudo-label the unlabelled samples at the token level. For some tasks, this works well since the label names (e.g. “education”, “sports”) are straightforward and usually have enough exact-word matches within unlabelled samples. Thus LOTClass performs well in the **Topic** dataset. However, for datasets like Emotion detection, classes such as “sadness” are more abstract and have a more unbalanced distribution, and are not contained directly in unlabelled samples. This leads to few samples in the label name data subsequently used by MLM and MCP. Thus LOTClass obtains relatively poor results for **UnifyEmotion** and **Emotion**.

Concerning the ablation study on the datasets, Table 7.5 presents the results. The results look similar to those of P-ZSC on the crisis datasets. This further demonstrates that each component of P-ZSC is essential to the final performance. As the results show, the label expansion (LE) component stands out as being particularly important, indicated by the fact that the performance substantially decreases without it.

Finally, the investigation of pseudo-labelled data by P-ZSC on the datasets is carried

out and the results are plotted in Figure 7.3. To summarise the results, the pseudo-labelling of P-ZSC can equal the performance of a fully-supervised BERT classifier with 18, 88 and 58 actually-labelled samples per class on **Topic**, **UnifyEmotion** and **Emotion** respectively. In practical terms, this represents a total of 40, 880 and 348 annotated samples as the datasets have 4, 10, and 6 classes respectively. This shows that there is some room to improve the pseudo-labelled data on datasets like **Topic**. However, there is nonetheless a clear indication that expensive annotations can be dramatically reduced by the use of P-ZSC.

	Topic	UnifyEmotion	Emotion
P-ZSC	50.68	30.22	64.47
- Self-training	44.18	25.35	59.83
- PLA	49.83	29.86	56.35
- PLA + LE	46.33	20.97	48.59

Table 7.5: Ablation study of P-ZSC on datasets beyond crisis

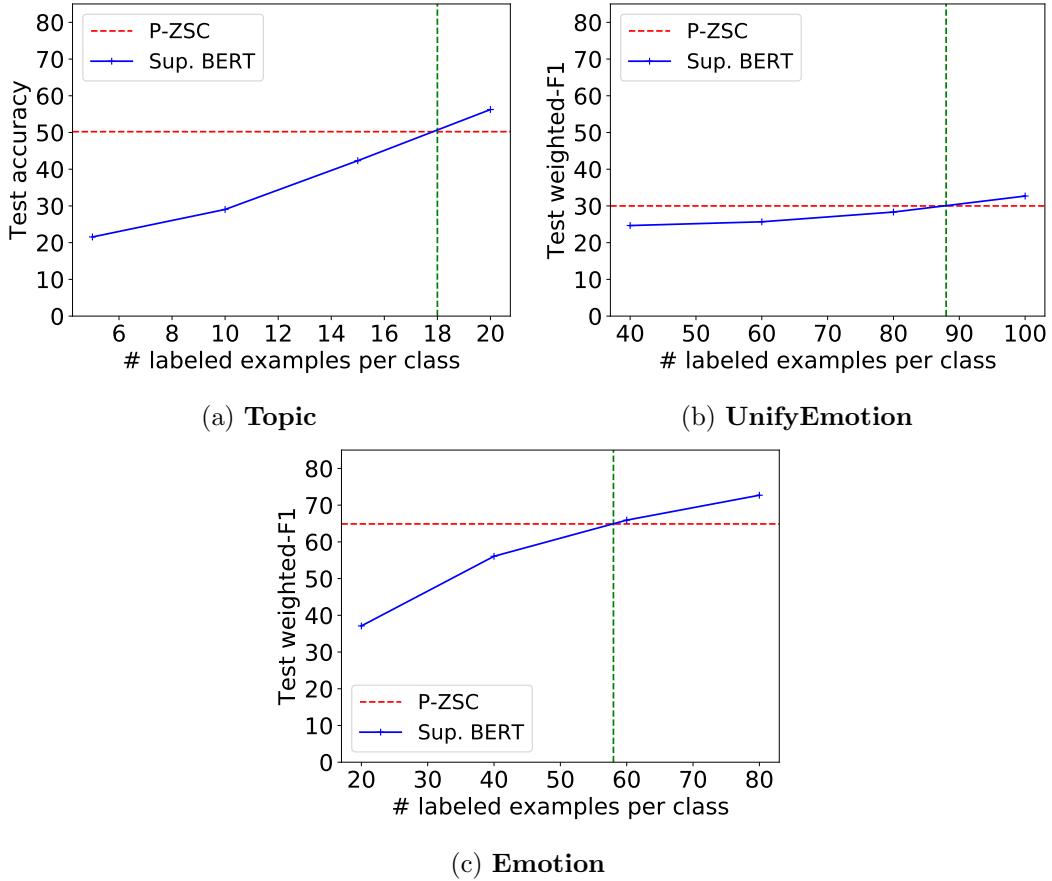


Figure 7.3: The investigation of pseudo-labelled data by P-ZSC on datasets beyond crisis

## 7.4 Conclusion

Motivated by the fact that the class space is usually not known until a crisis occurs and that annotation is a burden (particularly when the class space is large), this chapter presented a novel approach called P-ZSC that used pseudo-labelled data for zero-shot crisis message categorisation. To classify a crisis message into new classes that are not represented in any previously-available training data, P-ZSC obtained a pseudo-labelled data set through a matching mechanism between an unlabelled corpus of the target task and the label vocabularies of new classes. In contrast with crisis few-shot learning, P-ZSC can be considered to be a form of crisis zero-shot learning because it does not rely on any annotated data for model training but only the pseudo-labelled data, which helps eliminate the annotation cost in real-world crisis message categorisation. By comparing P-ZSC with baselines, it achieved state-of-the-art performance not only in the crisis categorisation task but also was shown to generalise better to tasks of other domains. The investigation of pseudo-labelled data showed that the pseudo-labelled data produced by P-ZSC can perform as well as using a supervised approach with hundreds of manually-annotated samples, indicating the potential for significant time and effort savings in training a model for the task of crisis message categorisation.

# Chapter 8: Conclusion

---

## 8.1 Summary of contributions

This thesis has focused on using messages circulated on social media during crisis situations for the purpose of supporting emergency response. The research has specifically developed computational techniques for categorising the messages in text forms (i.e., crisis tweets) based on the types of aid being requested, in order to provide useful information for emergency responders. In existing curated datasets, human annotators have associated large quantities of crisis-related messages with appropriate labels, and so it is common for researchers to experiment with supervised approaches to this classification problem. However, the nature of emerging crises means that this scenario is unrealistic. Primarily, since crises are time-critical, it is not feasible to spend the time required for the levels of annotation that would be required to train a supervised model. Additionally, there is a significant cost (both in terms of money and expertise) associated with organising sufficient levels of annotation. Therefore, in real-world crisis message categorisation, there is usually low availability of labelled data relating to emerging crisis events, leading to the difficulty of model training in a fully supervised way. Hence, this research was carried out in addressing the challenge of low data availability for practical crisis message categorisation through three scenarios.

The first, crisis domain adaptation, involves adapting a categorisation model trained on one domain (past crisis events) to perform well on a different but related domain (emerging events). This can be particularly useful in situations where for the same categorisation task, labelled data from past events are available even though no labelled data relating to emerging events are available.

The second, crisis few-shot learning, involves training a categorisation model to learn from a small number of labelled examples, often just a few labelled examples per class. This can be useful in situations where there is a limited amount of labelled data from emerging events available and a new categorisation task is required.

The third, crisis zero-shot learning, involves training a categorisation model to recognise and classify new classes (aid types) without any labelled examples of those classes. This can be useful in situations when there is no labelled data from emerging events

available and a new categorisation task is required for emerging events. The following summarises major contributions with proposed approaches for the three research scenarios.

In Chapter 4 and 5, the focus was on crisis domain adaptation and the findings of this research have been published in [137, 140, 141, 142]. Specifically, this research first put effort into many-to-many adaptation, which helped to address the need for categorisation models trained on past events being adapted to emerging events without requiring detailed knowledge of those events. To this end, the research presented a multi-task learning (MTL) approach that fine-tuned pre-trained language models (e.g. BERT) for crisis message categorisation, achieving superior performance compared to baselines. An ensemble version of this approach, combining multiple MTL models, was also introduced and demonstrated state-of-the-art performance in experiments. The research also addressed many/one-to-one crisis domain adaptations: developing adaptation models that can learn from a previous crisis or multiple crises in order to categorise messages related to an emerging crisis. For this purpose, the research proposed the use of sequence-to-sequence (seq2seq) pre-trained language models with the embedding of event information (CAST), without the need for any target data. Experimental results showed that CAST outperformed existing state-of-the-art crisis domain adaptation approaches, particularly when it had the least dependence on target data. In addition, the research investigated the selection of source events for a target event adaptation based on the characteristics of the events, and found that combining similar events tended to yield the best performance, as compared to adding dissimilar events to the training set. Overall, the contributions include the development of state-of-the-art approaches for crisis domain adaptation that address user needs for both many-to-many and many/one-to-one adaptations, as well as insights into how to select appropriate source events for a target event adaptation.

In Chapter 6, the research examined crisis few-shot learning, which addresses the need for emergency response systems to handle crisis message categorisation in real-world scenarios where there may be new predefined classes and limited annotated data for emerging events. To address this challenge, the research introduced two novel augmentation approaches: STA and ISA. STA is a self-controlled method that uses seq2seq pre-trained language models to generate new crisis messages based on a small number of labelled messages per class. To improve the quality of generated messages in terms of both lexical diversity and semantic fidelity (label-alignment), the messages were first generated by the seq2seq model to be conditional on the class names and then these messages were fidelity-checked by the model itself. This approach was found to generate messages with superior lexical diversity and semantic fidelity compared to existing baselines. ISA was then proposed as an optimised version of STA that includes an iterative mechanism and duplication mechanism to further improve the quality of

generated messages. The results demonstrated that ISA outperformed STA in both lexical diversity and semantic fidelity and achieved better performance in few-shot crisis message categorisation. In addition, the experiments revealed that both STA and ISA exhibited strong generalisation capabilities across other domains such as emotion and topic classification. Overall, the major contributions of this chapter include the development of two state-of-the-art augmentation approaches for crisis few-shot learning and insights into how to improve the quality of augmented crisis messages in terms of lexical diversity and semantic fidelity to enhance performance in few-shot crisis message categorisation.

While crisis few-shot learning addresses the need for emergency response systems to handle crisis message categorisation when there are new predefined classes and limited annotated data for emerging events, it still requires costly annotation when there is a large number of predefined classes and does not account for unequal class distributions, which are common in real-world scenarios. In Chapter 7, the research explored crisis zero-shot learning, which addresses the challenge of categorising crisis messages without depending on any annotated data and the findings of this research have been published in [143]. To address this challenge, the research proposed a novel approach called P-ZSC that uses pseudo-labelled data for zero-shot crisis message categorisation. P-ZSC obtained the pseudo-labelled data set by matching an unlabelled corpus of the target event with the label vocabularies of new classes, allowing for the categorisation of crisis messages into new classes without the need for labelled data. The results of comparing P-ZSC to baselines showed that it achieved state-of-the-art performance not only in the crisis categorisation domain but also demonstrated strong generalisation capabilities across other domains such as emotion detection and topic classification. It was also found that the pseudo-labelled data produced by P-ZSC was as effective as hundreds of manually annotated samples in both crisis categorisation and non-crisis classification tasks, indicating the possibility of saving a significant amount of time and effort when training a model with P-ZSC for these tasks. When comparing P-ZSC to a fully-supervised BERT model, the results show that a gap still exists in terms of the performance of P-ZSC. However, the fully-supervised scenario is artificial because it relies on pre-annotated data (i.e. whole training dataset) in a crisis situation, which is not practical as a crisis emerges. Due to time constraints and the workload involved, making a dataset like this is not trivial. Hence, a zero-shot approach like P-ZSC is a crucial area of research because it can be realistically deployed in a real crisis situation where users can set up labels to reflect what they are looking for and then start matching messages almost immediately. While P-ZSC does not fully solve this problem, it makes a significant improvement to the state of the art.

In summary, this thesis makes good contributions to the field of crisis informatics by addressing the challenge of limited annotated data availability for categorising crisis

messages on social media during emerging events. The approaches presented in this thesis have been developed in close collaboration with real-world scenarios and have demonstrated state-of-the-art performance in experiments. These approaches have the potential to be applied in practice to provide timely and effective humanitarian aid responses.

## 8.2 Future directions

This research has made progress in improving the categorisation of social media crisis messages in low data settings. However, there is still much work to be done to fully solve this research problem. There are several key areas of future work that should be considered in order to continue advancing in this area. The following identifies limitations of this research and presents a list of future work to consider.

- In the many-to-many adaptation work presented in this thesis, the MTL-based ensemble approach achieved the best performance to date. However, this approach is not very efficient because it involves combining multiple MTL models for crisis message categorisation. Inference efficiency is a critical concern during crises, when messages may come in rapidly, and hence it is an area of future work to optimise the efficiency of the ensemble approach. Techniques such as distillation, pruning, or compression of categorisation models could potentially improve the speed at which messages can be processed, leading to more efficient and timely emergency responses.
- In crisis few-shot learning, STA and ISA use human-designed templates to ensure lexical diversity and semantic fidelity for the generated messages. This makes it challenging to determine which templates are the most effective for optimal performance. As a result, future work could focus on automating the process of finding the optimal set of templates for STA and ISA.
- In P-ZSC, although the pseudo-labelled data represents high quality in some aspects, there remains some room to improve it. For example, the existing matching mechanism simply assigns a label to a message when they are locally matched with each other with high confidence. Hence, the future work can seek to explore better solutions such as global semantic meaning matching for obtaining better quality of pseudo-labelled data.
- It is necessary to develop a dynamic system that can adapt to different levels of data availability. While this research has presented state-of-the-art approaches

for crisis domain adaptation, few-shot, and zero-shot learning, it is important to combine these approaches into a system that can handle both high and low levels of data availability, or that can adapt to changing data availability over time. This includes integrating approaches for crisis message categorisation depending on the amount of available data. For instance, in situations where few labeled data and larger quantities of unlabeled data for emerging events are available, it would be valuable to explore how the zero-shot P-ZSC approach could be used in a semi-supervised manner in conjunction with the augmentation-based few-shot approaches (STA and ISA) to further improve crisis message categorisation.

- The proposed approaches used in this research focus on the raw text content of crisis messages. It is important to incorporate additional sources of information beyond the text of the messages themselves. This could include incorporating images, locations, or knowledge graphs to provide additional context and help improve the accuracy of categorisation.
- This research so far has just handled English crisis messages. It is important to extend this work to other languages, as crisis messages are often written in a variety of languages. Cross-lingual crisis message categorisation is therefore a critical area of future work.
- The proposed approaches operate on the assumption that the crisis messages being learned from, and being classified, are credible and reliable sources of information. Verifying the credibility of crisis messages is also critical, as false or misleading information can cause confusion and hinder effective response. Hence, it would be helpful to add a credibility-checking component before feeding the messages to the categorisation system.
- This research has made strides in decreasing the need for labelled data in the categorisation of real-world social media crisis messages. However, it still requires some labelled and/or unlabelled data for emerging events. One potential area of future work is to optimise large-scale language models like GPT-3 and ChatGPT, which are trained on vast amounts of text data. If these models can be made more efficient, they have the potential to be used for crisis message categorisation of any event without the need for additional training on domain-specific data.

Overall, these are key areas of future work that are important not only for this specific research, but also for the broader community working on social media crisis message categorisation in the face of low data availability.

# Bibliography

---

- [1] Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649, 2018.
- [2] Firoj Alam, Shafiq Joty, and Muhammad Imran. Domain adaptation with adversarial training and graph embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1077–1087, 2018.
- [3] Firoj Alam, Umair Qazi, Muhammad Imran, and Ferda Ofli. Humaid: Human-annotated disaster incidents data from twitter with deep learning benchmarks. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 15, pages 933–942, 2021.
- [4] Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. Do not have enough data? deep learning to the rescue! In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7383–7390, 2020.
- [5] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.
- [6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015.
- [7] Ghazaleh Beigi, Xia Hu, Ross Maciejewski, and Huan Liu. An overview of sentiment analysis in social media and its applications in disaster relief. In *Sentiment analysis and ontology engineering*, pages 313–340. Springer, 2016.
- [8] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Proceedings of the 45th annual meeting of the association of computational linguistics*, pages 440–447, 2007.
- [9] John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128, 2006.

- [10] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017.
- [11] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.
- [12] C Buntain, R McCreadie, and I Soboroff. Incident streams 2020: Trecis in the time of covid-19. In *18th International Conference on Information Systems for Crisis Response and Management, ISCRAM 2021*, pages 621–639, 2021.
- [13] Cody Buntain, Richard McCreadie, and Ian Soboroff. Incident streams 2021 off the deep end: Deeper annotations and evaluations in twitter. In *18th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, pages 584–604, 2021.
- [14] Tianfeng Chai and Roland R Draxler. Root mean square error (rmse) or mean absolute error (mae)?—arguments against avoiding rmse in the literature. *Geoscientific model development*, 7(3):1247–1250, 2014.
- [15] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.
- [16] Minmin Chen, Kilian Q Weinberger, and John Blitzer. Co-training for domain adaptation. *Advances in neural information processing systems*, 24, 2011.
- [17] Minmin Chen, Zhixiang Xu, Kilian Q. Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *Proceedings of the 29th International Conference on International Conference on Machine Learning, ICML’12*, page 1627–1634, Madison, WI, USA, 2012. Omnipress.
- [18] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, 2014.
- [19] Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

- [20] Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1107–1116, 2017.
- [21] Wenyuan Dai, Gui-Rong Xue, Qiang Yang, and Yong Yu. Transferring naive bayes classifiers for text classification. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 540–545. AAAI Press, 2007.
- [22] H DAUME III. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, 2007*, pages 256–263, 2007.
- [23] Joe Davison. Zero-shot classifier distillation.  
[https://github.com/huggingface/transformers/tree/master/examples/research\\_projects/zero-shot-distillation](https://github.com/huggingface/transformers/tree/master/examples/research_projects/zero-shot-distillation), 2021.
- [24] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [25] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.
- [26] Cicero Dos Santos and Maira Gatti. Deep convolutional neural networks for sentiment analysis of short texts. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 69–78, 2014.
- [27] Chunning Du, Haifeng Sun, Jingyu Wang, Qi Qi, and Jianxin Liao. Adversarial and domain-aware bert for cross-domain sentiment analysis. In *Proceedings of the 58th annual meeting of the Association for Computational Linguistics*, pages 4019–4028, 2020.
- [28] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(Jul):2121–2159, 2011.

- [29] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.
- [30] Sergey Edunov, Myle Ott, Michael Auli, and David Grangier. Understanding back-translation at scale. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun’ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 489–500. Association for Computational Linguistics, 2018.
- [31] Mica R Endsley. Toward a theory of situation awareness in dynamic systems. In *Situational Awareness*, pages 9–42. Routledge, 2017.
- [32] Steven Y Feng, Varun Gangal, Dongyeop Kang, Teruko Mitamura, and Eduard Hovy. Genaug: Data augmentation for finetuning text generators. In *Proceedings of Deep Learning Inside Out (DeeLIO): The First Workshop on Knowledge Extraction and Integration for Deep Learning Architectures*, pages 29–42, 2020.
- [33] Julia Daisy Fraustino, Brooke Liu, and Yan Jin. Social media use during disasters: a review of the knowledge base and gaps. National Consortium for the Study of Terrorism and Responses to Terrorism, 2012.
- [34] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.
- [35] Tianyu Gao, Adam Fisch, and Danqi Chen. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*, 2021.
- [36] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *European conference on computer vision*, pages 597–613. Springer, 2016.
- [37] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Domain adaptation for large-scale sentiment classification: A deep learning approach. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 513–520. Omnipress, 2011.
- [38] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.

- [39] Debarati Guha-Sapir, Femke Vos, Regina Below, and Sylvain Ponserre. Annual disaster statistical review. *Centre for Research on the Epidemiology of Disasters*, 2011.
- [40] Gongde Guo, Hui Wang, David Bell, Yaxin Bi, and Kieran Greer. Knn model-based approach in classification. In *OTM Confederated International Conferences” On the Move to Meaningful Internet Systems”*, pages 986–996. Springer, 2003.
- [41] Mandy Guo, Joshua Ainslie, David C. Uthus, Santiago Ontañón, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. Longt5: Efficient text-to-text transformer for long sequences. In Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz, editors, *Findings of the Association for Computational Linguistics: NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 724–736. Association for Computational Linguistics, 2022.
- [42] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. In *International Conference on Learning Representations*, 2020.
- [43] Nic Herndon and Doina Caragea. Empirical study of domain adaptation with naive bayes on the task of splice site prediction. *BIOINFORMATICS (BIOSTEC 2014)*, 1:57–67, 2014.
- [44] Nic Herndon and Doina Caragea. An evaluation of self-training styles for domain adaptation on the task of splice site prediction. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 1042–1047, 2015.
- [45] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. In *Neural computation*, volume 9, pages 1735–1780. MIT Press, 1997.
- [46] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2019.
- [47] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Chin-Yew Lin, and Deepak Ravichandran. Toward semantics-based answer pinpointing. In *Proceedings of the First International Conference on Human Language Technology Research*, 2001.
- [48] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, 2018.

- [49] Muhammad Imran, Shady Elbassuoni, Carlos Castillo, Fernando Diaz, and Patrick Meier. Extracting information nuggets from disaster- related messages in social media. In Tina Comes, Frank Fiedrich, Simon Fortier, Jutta Geldermann, and Tim Müller, editors, *10th Proceedings of the International Conference on Information Systems for Crisis Response and Management, Baden-Baden, Germany, May 12-15, 2013*. ISCRAM Association, 2013.
- [50] Muhammad Imran, Prasenjit Mitra, and Carlos Castillo. Twitter as a lifeline: Human-annotated twitter corpora for NLP of crisis-related messages. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Sara Goggi, Marko Grobelnik, Bente Maegaard, Joseph Mariani, Hélène Mazo, Asunción Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Tenth International Conference on Language Resources and Evaluation LREC 2016, Portorož, Slovenia, May 23-28, 2016*. European Language Resources Association (ELRA), 2016.
- [51] Katarzyna Janocha and Wojciech Marian Czarnecki. On loss functions for deep neural networks in classification. *Schedae Informaticae*, 25:49–59, 2016.
- [52] Kalervo Järvelin and Jaana Kekäläinen. Cumulated gain-based evaluation of ir techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.
- [53] Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in NLP. In John A. Carroll, Antal van den Bosch, and Annie Zaenen, editors, *ACL 2007, Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics, June 23-30, 2007, Prague, Czech Republic*. The Association for Computational Linguistics, 2007.
- [54] Jing Jiang and ChengXiang Zhai. A two-stage approach to domain adaptation for statistical classifiers. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 401–410, 2007.
- [55] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of Tricks for Efficient Text Classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics, 2017.
- [56] James M Joyce. Kullback-leibler divergence. In *International encyclopedia of statistical science*, pages 720–722. Springer, 2011.
- [57] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *CoRR*, abs/2001.08361, 2020.

- [58] Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar, October 2014. Association for Computational Linguistics.
- [59] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.
- [60] Roman Klinger et al. An analysis of annotated corpora for emotion classification in text. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2104–2119, 2018.
- [61] Sosuke Kobayashi. Contextual augmentation: Data augmentation by words with paradigmatic relations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 452–457, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.
- [62] Varun Kumar, Ashutosh Choudhary, and Eunah Cho. Data augmentation using pre-trained transformer models. In *Proceedings of the 2nd Workshop on Life-long Learning for Spoken Language Systems*, pages 18–26, Suzhou, China, December 2020. Association for Computational Linguistics.
- [63] Laura Lambert, Christos JP Moschovitis, Hilary W Poole, and Chris Woodford. *The internet: a historical encyclopedia*, volume 2. ABC-CLIO, 2005.
- [64] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2020.
- [65] Daniel Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. *Advances in neural information processing systems*, 13, 2000.
- [66] Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *In Workshop on challenges in representation learning, ICML*, 2013.
- [67] Omer Levy, Minjoon Seo, Eunsol Choi, and Luke Zettlemoyer. Zero-shot relation extraction via reading comprehension. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 333–342, Vancouver, Canada, August 2017. Association for Computational Linguistics.

- [68] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online, July 2020. Association for Computational Linguistics.
- [69] Hongmin Li. *Domain adaptation approaches for classifying social media crisis data*. PhD thesis, 2021.
- [70] Hongmin Li, Doina Caragea, and Cornelia Caragea. Combining self-training with deep learning for disaster tweet classification. In *The 18th International Conference on Information Systems for Crisis Response and Management (ISCRAM 2021)*, 2021.
- [71] Hongmin Li, Doina Caragea, Cornelia Caragea, and Nic Herndon. Disaster response aided by tweet classification with a domain adaptation approach. *Journal of Contingencies and Crisis Management*, 26(1):16–27, 2018.
- [72] Hongmin Li, Nicolais Guevara, Nic Herndon, Doina Caragea, Kishore Neppalli, Cornelia Caragea, Anna Cinzia Squicciarini, and Andrea H. Tapia. Twitter mining for disaster response: A domain adaptation approach. In Leysia Palen, Monika Büscher, Tina Comes, and Amanda Lee Hughes, editors, *12th Proceedings of the International Conference on Information Systems for Crisis Response and Management, Krystiansand, Norway, May 24-27, 2015*. ISCRAM Association, 2015.
- [73] Hongmin Li, Xukun Li, Doina Caragea, and Cornelia Caragea. Comparison of word embeddings and sentence encodings as generalized representations for crisis tweet classification tasks. *Proceedings of ISCRAM Asia Pacific*, 2018.
- [74] Hongmin Li, Oleksandra Sopova, Doina Caragea, and Cornelia Caragea. Domain adaptation for crisis data using correlation alignment and self-training. *International Journal of Information Systems for Crisis Response and Management (IJISRAM)*, 10(4):1–20, 2018.
- [75] Xin Li and Dan Roth. Learning question classifiers. In *COLING 2002: The 19th International Conference on Computational Linguistics*, 2002.
- [76] Xukun Li and Doina Caragea. Domain adaptation with reconstruction for disaster tweet classification. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1561–1564, 2020.

- [77] Xukun Li, Doina Caragea, Cornelia Caragea, Muhammad Imran, and Ferda Ofli. Identifying disaster damage images using a domain adaptation approach. In Zeno Franco, José J. González, and José H. Canós, editors, *Proceedings of the 16th International Conference on Information Systems for Crisis Response and Management, València, Spain, May 19-22, 2019*. ISCRAM Association, 2019.
- [78] Zheng Li, Yu Zhang, Ying Wei, Yuxiang Wu, and Qiang Yang. End-to-end adversarial memory network for cross-domain sentiment classification. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 2237–2243. ijcai.org, 2017.
- [79] Junhua Liu, Trisha Singhal, Luciënne T. M. Blessing, Kristin L. Wood, and Kwan Hui Lim. Crisisbert: A robust transformer for crisis classification and contextual crisis embedding. In Owen Conlan and Eelco Herder, editors, *HT '21: 32nd ACM Conference on Hypertext and Social Media, Virtual Event, Ireland, 30 August 2021 - 2 September 2021*, pages 133–141. ACM, 2021.
- [80] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *CoRR, abs/1907.11692*, 2019.
- [81] Stephen Mayhew, Shyam Upadhyay, Wenpeng Yin, Lucia Huo, Devanshu Jain, Prasanna Poudyal, Tatiana Tsygankova, Yihao Chen, Xin Li, Nitish Gupta, et al. University of pennsylvania lorehlt 2018 submission. Technical report, Technical report, University of Pennsylvania, 2018., 2018.
- [82] Reza Mazloom, Hongmin Li, Doina Caragea, Cornelia Caragea, and Muhammad Imran. A hybrid domain adaptation approach for identifying crisis-relevant tweets. *International Journal of Information Systems for Crisis Response and Management (IJISCRAM)*, 11(2):1–19, 2019.
- [83] Reza Mazloom, HongMin Li, Doina Caragea, Muhammad Imran, and Cornelia Caragea. Classification of twitter disaster data using a hybrid feature-instance adaptation approach. In Kees Boersma and Brian M. Tomaszewski, editors, *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management, Rochester, NY, USA, May 20-23, 2018*. ISCRAM Association, 2018.
- [84] Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6294–6305, 2017.

- [85] R McCreadie, C Buntain, and I Soboroff. Trec incident streams: Finding actionable information on social media. In *16th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*. 2019.
- [86] Richard McCreadie, Cody Buntain, and Ian Soboroff. TREC incident streams: Finding actionable information on social media. *Proceedings of the International ISCRAM Conference*, 2019-May(May):691–705, 2019.
- [87] Richard McCreadie, Cody Buntain, and Ian Soboroff. Incident streams 2019: Actionable insights and how to find them. In Amanda Lee Hughes, Fiona McNeill, and Christopher W. Zobel, editors, *17th International Conference on Information Systems for Crisis Response and Management, ISCRAM 2020, May 2020*, pages 744–760. ISCRAM Digital Library, 2020.
- [88] Dheeraj Mekala and Jingbo Shang. Contextualized weak supervision for text classification. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 323–333, 2020.
- [89] Yu Meng, Jiaming Shen, Chao Zhang, and Jiawei Han. Weakly-supervised neural text classification. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM 2018)*, volume 2018, 2018.
- [90] Yu Meng, Yunyi Zhang, Jiaxin Huang, Chenyan Xiong, Heng Ji, Chao Zhang, and Jiawei Han. Text classification using label names only: A language model self-training approach. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9006–9017, Online, November 2020. Association for Computational Linguistics.
- [91] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.
- [92] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.
- [93] Tom M. Mitchell. *Machine learning, International Edition*. McGraw-Hill Series in Computer Science. McGraw-Hill, 1997.
- [94] Thomas Müller, Guillermo Pérez-Torró, and Marc Franco-Salvador. Few-shot learning with siamese networks and label tuning. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long*

*Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 8532–8545.  
Association for Computational Linguistics, 2022.

- [95] Venkata Kishore Neppalli, Cornelia Caragea, and Doina Caragea. Deep neural networks versus naive bayes classifiers for identifying informative tweets during disasters. In Kees Boersma and Brian M. Tomaszewski, editors, *Proceedings of the 15th International Conference on Information Systems for Crisis Response and Management, Rochester, NY, USA, May 20-23, 2018*. ISCRAM Association, 2018.
- [96] Dat Quoc Nguyen, Thanh Vu, and Anh-Tuan Nguyen. Bertweet: A pre-trained language model for english tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, 2020.
- [97] Dat Tien Nguyen, Kamla Al-Mannai, Shafiq R. Joty, Hassan Sajjad, Muhammad Imran, and Prasenjit Mitra. Rapid classification of crisis-related data on social networks using convolutional neural networks. *CoRR*, abs/1608.03902, 2016.
- [98] William S Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.
- [99] Abiola Obamuyide and Andreas Vlachos. Zero-shot relation classification as textual entailment. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 72–78, 2018.
- [100] Jolie O’Dell. How we use social media during emergencies, 2011.  
<https://mashable.com/2011/02/11/social-media-in-emergencies>.
- [101] Alexandra Olteanu, Carlos Castillo, Fernando Diaz, and Sarah Vieweg. Crisislex: A lexicon for collecting and filtering microblogged communications in crises. In *Eighth International AAAI Conference on Weblogs and Social Media*, 2014.
- [102] Alexandra Olteanu, Sarah Vieweg, and Carlos Castillo. What to expect when the unexpected happens: Social media communications across crises. In *Proceedings of the 18th ACM conference on computer supported cooperative work & social computing*, pages 994–1009, 2015.
- [103] Leysia Palen and Sophia B Liu. Citizen communications in crisis: anticipating a future of ict-supported public participation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 727–736. ACM, 2007.
- [104] Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. Cross-domain sentiment classification via spectral feature alignment. In

*Proceedings of the 19th international conference on World wide web*, pages 751–760, 2010.

- [105] Viswa Mani Kiran Peddinti and Prakriti Chintalapoodi. Domain adaptation in sentiment analysis of twitter. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.
- [106] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.
- [107] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 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)*, pages 2227–2237. Association for Computational Linguistics, 2018.
- [108] Raul Puri and Bryan Catanzaro. Zero-shot text classification with generative language models. *CoRR, abs/1912.10165*, 2019.
- [109] Pushpankar Kumar Pushp and Muktabh Mayank Srivastava. Train once, test anywhere: Zero-shot learning for text classification. *CoRR, abs/1712.05972*, 2017.
- [110] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training.
- [111] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.
- [112] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.
- [113] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.
- [114] Irina Rish et al. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3, pages 41–46, 2001.

- [115] Sara Rosenthal, Noura Farra, and Preslav Nakov. SemEval-2017 task 4: Sentiment analysis in Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 502–518, Vancouver, Canada, August 2017. Association for Computational Linguistics.
- [116] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.
- [117] Sebastian Ruder. *Neural transfer learning for natural language processing*. PhD thesis, NUI Galway, 2019.
- [118] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *Advances in Neural Information Processing Systems 33 (NIPS)*. 2019.
- [119] Elvis Saravia, Hsien-Chi Toby Liu, Yen-Hao Huang, Junlin Wu, and Yi-Shin Chen. CARER : Contextualized affect representations for emotion recognition. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3687–3697, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.
- [120] Timo Schick and Hinrich Schütze. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269, 2021.
- [121] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.
- [122] Rico Sennrich, Barry Haddow, and Alexandra Birch. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, 2016.
- [123] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, 2020.
- [124] Sam Shleifer. Low resource text classification with ulmfit and backtranslation. *CoRR*, abs/1903.09244, 2019.
- [125] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.
- [126] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for

- semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.
- [127] Stephanie M. Strassel, Ann Bies, and Jennifer Tracey. Situational awareness for low resource languages: the LORELEI situation frame annotation task. In Saptarshi Ghosh, Kripabandhu Ghosh, Tanmoy Chakraborty, Debasis Ganguly, Gareth J. F. Jones, and Marie-Francine Moens, editors, *Proceedings of the First International Workshop on Exploitation of Social Media for Emergency Relief and Preparedness co-located with European Conference on Information Retrieval, SMERP@ECIR 2017, Aberdeen, UK, April 9, 2017*, volume 1832 of *CEUR Workshop Proceedings*, pages 32–41. CEUR-WS.org, 2017.
- [128] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. *Advances in neural information processing systems*, 20, 2007.
- [129] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.
- [130] Songbo Tan, Xueqi Cheng, Yuefen Wang, and Hongbo Xu. Adapting naive bayes to domain adaptation for sentiment analysis. In *European Conference on Information Retrieval*, pages 337–349. Springer, 2009.
- [131] Yuta Tsuboi, Hisashi Kashima, Shohei Hido, Steffen Bickel, and Masashi Sugiyama. Direct density ratio estimation for large-scale covariate shift adaptation. *Journal of Information Processing*, 17:138–155, 2009.
- [132] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.
- [133] Sappadla Prateek Veeranna, Jinseok Nam, Eneldo Loza Mencia, and Johannes Fürnkranz. Using semantic similarity for multi-label zero-shot classification of text documents. In *Proceeding of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges, Belgium: Elsevier*, pages 423–428, 2016.
- [134] Sarah Vieweg, Amanda L Hughes, Kate Starbird, and Leysia Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1079–1088. ACM, 2010.

- [135] Sarah Elizabeth Vieweg. *Situational awareness in mass emergency: A behavioral and linguistic analysis of microblogged communications*. PhD thesis, University of Colorado at Boulder, 2012.
- [136] Changhan Wang, Kyunghyun Cho, and Jiatao Gu. Neural machine translation with byte-level subwords. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9154–9160, 2020.
- [137] Congcong Wang and David Lillis. Classification for Crisis-Related Tweets Leveraging Word Embeddings and Data Augmentation. In *Proceedings of the Twenty-Eighth Text REtrieval Conference (TREC 2019)*, Gaithersburg, MD, 2020.
- [138] Congcong Wang and David Lillis. A Comparative Study on Word Embeddings in Deep Learning for Text Classification. In *Proceedings of the 4th International Conference on Natural Language Processing and Information Retrieval (NLPiR 2020)*, Seoul, South Korea, December 2020.
- [139] Congcong Wang and David Lillis. UCD-CS at W-NUT 2020 shared task-3: A text to text approach for COVID-19 event extraction on social media. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 514–521, Online, November 2020. Association for Computational Linguistics.
- [140] Congcong Wang and David Lillis. Multi-task transfer learning for finding actionable information from crisis-related messages on social media. In *Proceedings of the Twenty-Ninth Text REtrieval Conference (TREC 2020)*, Gaithersburg, MD, USA, 2021.
- [141] Congcong Wang, Paul Nulty, and David Lillis. Crisis Domain Adaptation Using Sequence-to-Sequence Transformers. In Anouck Adrot, Rob Grace, Kathleen Moore, and Christopher W. Zobel, editors, *ISCRAM 2021 Conference Proceedings – 18th International Conference on Information Systems for Crisis Response and Management*, pages 655–666, Blacksburg, VA (USA), 2021. Virginia Tech.
- [142] Congcong Wang, Paul Nulty, and David Lillis. Transformer-based Multi-task Learning for Disaster Tweet Categorisation. In Anouck Adrot, Rob Grace, Kathleen Moore, and Christopher W. Zobel, editors, *ISCRAM 2021 Conference Proceedings – 18th International Conference on Information Systems for Crisis Response and Management*, pages 705–718, Blacksburg, VA (USA), 2021. Virginia Tech.

- [143] Congcong Wang, Paul Nulty, and David Lillis. Using pseudo-labelled data for zero-shot text classification. In *International Conference on Applications of Natural Language to Information Systems*, pages 35–46. Springer, 2022.
- [144] Congcong Wang, Gonzalo Fiz Pontiveros, Steven Derby, and Tri Kurniawan Wijaya. STA: self-controlled text augmentation for improving text classifications. *CoRR*, abs/2302.12784, 2023.
- [145] William Yang Wang and Diyi Yang. That’s so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563, 2015.
- [146] Zihan Wang, Dheeraj Mekala, and Jingbo Shang. X-class: Text classification with extremely weak supervision. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3043–3053, 2021.
- [147] Jason Wei and Kai Zou. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389, 2019.
- [148] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.
- [149] Edwin B Wilson. Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212, 1927.
- [150] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierrick 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. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics.
- [151] Xing Wu, Shangwen Lv, Liangjun Zang, Jizhong Han, and Songlin Hu. Conditional bert contextual augmentation. In *International Conference on Computational Science*, pages 84–95. Springer, 2019.

- [152] Yonghui Wu, 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. Google’s neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.
- [153] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487. PMLR, 2016.
- [154] Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022.
- [155] Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, 2021.
- [156] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 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*, pages 1480–1489. Association for Computational Linguistics, 2016.
- [157] Zhiqian Ye, Yuxia Geng, Jiaoyan Chen, Jingmin Chen, Xiaoxiao Xu, SuHang Zheng, Feng Wang, Jun Zhang, and Huajun Chen. Zero-shot text classification via reinforced self-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3014–3024, 2020.
- [158] Wenpeng Yin, Jamaal Hay, and Dan Roth. Benchmarking zero-shot text classification: Datasets, evaluation and entailment approach. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3905–3914, 2019.
- [159] Kiran Zahra, Muhammad Imran, and Frank O Ostermann. Automatic identification of eyewitness messages on twitter during disasters. *Information processing & management*, 57(1):102107, 2020.

- [160] Lei Zhang, Shuai Wang, and Bing Liu. Deep learning for sentiment analysis: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1253, 2018.
- [161] Ningyu Zhang, Luoqiu Li, Xiang Chen, Shumin Deng, Zhen Bi, Chuanqi Tan, Fei Huang, and Huajun Chen. Differentiable prompt makes pre-trained language models better few-shot learners. In *International Conference on Learning Representations*, 2022.
- [162] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 649–657. Curran Associates, Inc., 2015.
- [163] Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657, 2015.
- [164] Ye Zhang and Byron C Wallace. A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 253–263, 2017.
- [165] Yu Zhang and Qiang Yang. A survey on multi-task learning. *CoRR*, abs/1707.08114, 2017.
- [166] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis C. M. Lau. A c-lstm neural network for text classification. In *CoRR*, abs/1511.08630, 2015.
- [167] Guangyou Zhou, Zhiwen Xie, Xiangji Huang, and Tingting He. Bi-transferring deep neural networks for domain adaptation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 322–332, 2016.
- [168] Peng Zhou, Zhenyu Qi, Suncong Zheng, Jiaming Xu, Hongyun Bao, and Bo Xu. Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3485–3495. The COLING 2016 Organizing Committee, 2016.
- [169] Fuzhen Zhuang, Xiaohu Cheng, Ping Luo, Sinno Jialin Pan, and Qing He. Supervised representation learning: Transfer learning with deep autoencoders. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

- [170] Yftah Ziser and Roi Reichart. Neural structural correspondence learning for domain adaptation. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 400–410, 2017.
- [171] Yftah Ziser and Roi Reichart. Pivot based language modeling for improved neural domain adaptation. In *Proceedings of the 2018 conference of the north American chapter of the association for computational linguistics: Human language technologies, volume 1 (long papers)*, pages 1241–1251, 2018.

# Appendix A: Appendix

---

## A.1 Information types of TREC-IS

	Category	Description
Actionable	Request-GoodsServices	The user is asking for a particular service or physical good
	Request-SearchAndRescue	The user is requesting a rescue (for themselves or others)
	Report-NewSubEvent	The user is reporting a new occurrence that public safety officers need to respond to
	Report-ServiceAvailable	The user is reporting that they or someone else is providing a service
	CallToAction-MovePeople	The user is asking people to leave an area or go to another area
	Report-EmergingThreats	The user is reporting a potential problem that may cause future loss of life or damage
Non-actionable	CallToAction-Volunteer	The user is asking people to volunteer to help the response effort
	CallToAction-Donations	The user is asking people to donate goods/money
	Report-Weather	The user is providing a weather report (current or forecast)
	Report-Location	The post contains information about the user or observation location
	Request-InformationWanted	The user is requesting information
	Report-FirstPartyObservation	The user is giving an eye-witness account
	Report-ThirdPartyObservation	The user is reporting information that they received from someone else
	Report-MultimediaShare	The user is sharing images or video
	Report-Factoid	The user is relating some facts, typically numerical
	Report-Official	An official report by a government or public safety representative
	Report-News	The post is a news report providing/linking to current/continuous coverage of the event
	Report-CleanUp	A report of the clean up after the event
	Report-Hashtags	Reporting which hashtags correspond to each event
	Report-OriginalEvent	A report of the original event occurring.
	Other-ContextualInformation	The post contains contextual information that can help understand the event, but is not about the event itself
	Other-Advice	The user is providing some advice to the public
	Other-Sentiment	The post is expressing some sentiment about the event
	Other-Discussion	Users are discussing the event
	Other-Irrelevant	The post is unrelated to the event or contains no information

Table A.1: Actionable and non-actionable information types in the TREC-IS dataset [85]