


School of Electrical Engineering, Computing and
Mathematical Sciences


Advanced Deep Learning Methods for Enhancing
Information Compliance Checking

Chongyi Liu 

This thesis is presented for the Degree of
Doctor of Philosophy
of
Curtin University

Dec 2023

To the best of my knowledge and belief this thesis contains no material previously published by any other person except where due acknowledgement has been made. This thesis contains no material which has been accepted for the award of any other degree or diploma in any university.

Chongyi Liu 

Acknowledgements

I would like to express my sincere gratitude to Associate Professor XU Honglei and Professor WANG Xiangyu for their invaluable guidance and unwavering support throughout the entire research process.

Scientific Editor HENG Julian (Remotely Consulting, Australia) provided professional proofreading and editing services in the preparation of this thesis (Certificate No. 8Lh1Ez0B)

List of Abbreviations

AI Artificial Intelligence

AMI Artificial Machine Intelligence

BERT Bidirectional Encoder Representations from Transformers

BiLSTM Bidirectional Long Short-Term Memory

CAD Computer-Aided Design

CBOW Continuous Bag of Words

CNN Convolutional Neural Network

CRF Conditional Random Field

GCNs Graph Convolutional Networks

GloVe Global Vectors for word representation

GNN Graph Neural Network

GPT Generative Pre-training Transformer

HLG Heterogeneous Linguistics Graph

LSTM Long Short-Term Memory

MSIP Multi-Step Information Propagation

MWA Multi-source Word-Aligned

NER Named Entity Recognition

NLP Natural Language Processing

NNLM Neural Network Language Model

RNN Recurrent Neural Network

TF-IDF Term Frequency-Inverse Document Frequency

Abstract

Information checking, such as Computer-Aided Design (CAD) drawing compliance checking, remains a considerable task that amounts to a significant workload from domain specialists. While modern programmatic methods have simplified this process, manual input specifications that are required in the process still represent a time-consuming and error-prone step. The present research in this thesis tackles this challenge by focusing on the development and refinement of advanced deep learning algorithms, primarily in the Natural Language Processing (NLP) sphere, as an innovative and time-saving solution. Although the study's context revolves around information drawing compliance, the principal aims of this thesis have implications for the advancement of the underlying deep learning methodologies that are applicable across a variety of fields.

One of the key objectives of this thesis is to improve the efficiency and robustness of NLP techniques, allowing algorithms to automatically interpret and extract crucial data from documents written in human languages. To achieve this, various machine learning techniques were studied, developed, and tested, such as Graph Convolutional Networks (GCNs), Term Frequency-Inverse Document Frequency (TF-IDF) measures, and pre-trained language models like Bidirectional Encoder Representations from Transformers (BERT). The algorithms were not only evaluated for their ability to process text but were also stress-tested under conditions of varying complexities and scales to evaluate their broad applicability and scalability.

The present research introduces a novel concept, termed Document-Relational

GCNs. Unlike traditional GCNs, which only operate at word-document and word-word levels, the proposed method detailed in this thesis incorporates document-document relations as features, adding complexity and richness to the adjacency matrix graph used for text classification. This model uses cumulative TF-IDF scores to generate these document-document relations and has been rigorously evaluated using five benchmark databases.

Another significant scientific contribution from this thesis is the introduction of GraphNorm in Heterogeneous Linguistics Graph (HLG) models. The technique employs dynamic normalization layers for each Graph Neural Network (GNN) module in the HLG model. This novel approach optimizes the learning process by retaining more reliable node information and mitigating the noise, thus making the model more resilient to graph modifications.

The research also introduces innovative position encoding strategies within a multi-granularity contextualized model, thereby advancing the process of Document-Level Event Role Filler Extraction. Utilizing Global Vectors for word representation (GloVe) and BERT embeddings along with sinusoidal and relative position encodings, the research achieves a richer, more contextualized representation of words, which is crucial for tasks like event extraction.

Evaluation of all proposed methods used multiple datasets and metrics, including accuracy and F1 scores. Findings indicate significant improvements in text classification tasks, model training efficiency, and resilience against noisy data and graph modifications. This research has wide-ranging implications, including the automation of labour-intensive tasks across diverse fields, which opens the door for future breakthroughs in machine learning and Artificial Intelligence (AI).

Contents

Acknowledgements	v
List of Abbreviations	vii
Abstract	ix
1 Introduction	1
1.1 Background	2
1.2 Study Framework	5
1.3 Thesis Structure	6
2 Background	9
2.1 NLP	9
2.2 NLP Information Checking	10
2.3 TF-IDF	11
2.4 Word Embeddings	13
2.4.1 Bag-of-words Model	14
2.4.2 Pre-trained Word Vectors	15
2.4.2.1 Continuous Bag of Words (CBOW) Model	16
2.4.2.2 Skip-gram Model	16
2.4.2.3 Word2vec	17

2.4.2.4	GloVe	18
2.4.3	Pre-trained Language Model - BERT	20
2.4.3.1	Transformer	20
2.4.3.2	BERT	21
2.4.3.3	Strengths and Innovations	22
2.4.3.4	Limitations and Challenges	22
2.4.3.5	Application in Information Checking	23
2.5	GCNs	23
2.6	Normalization	24
2.6.1	Batch Normalization	25
2.6.2	Layer Normalization	26
2.6.3	Instance Normalization	26
2.7	Event Role Filler Extraction	26
2.7.1	Sequence Tagging	27
2.7.2	Bidirectional Long Short-Term Memory (BiLSTM)-Conditional Random Field (CRF) Model	27
2.8	Challenges and Gaps in Information Checking	28
2.8.1	Justification for Research Focus	29
3	Document-Relational Graph Convolutional Networks	33
3.1	Introduction	34
3.2	Related Work	37
3.2.1	Motivation and Detailed Analysis	37
3.2.1.1	Motivation Behind Document-Relational GCNs	37
3.2.1.2	Challenges Addressed	38
3.2.2	Graph Generation	38

3.2.3	GCNs Model	39
3.3	Proposed Method	40
3.4	Experiment	43
3.4.1	Datasets	44
3.4.2	Experiment Parameters	45
3.4.3	Hidden Layers	45
3.4.4	Justification for Comparative Techniques	45
3.4.5	Experiment Results	47
3.4.6	In-depth Analysis	48
3.5	Conclusion	49
4	A GraphNorm module in Heterogeneous Linguistics Graph Model	51
4.1	Introduction	52
4.2	Related Work	55
4.2.1	HLG model	55
4.2.2	Graph Norm	58
4.3	Proposed Method	60
4.3.1	Motivation	60
4.4	Experiments	63
4.4.1	Datasets	65
4.4.2	Word Segmentation	66
4.4.3	Environment and Parameters	67
4.4.4	Justification for Comparative Techniques	67
4.4.5	Experiment Results	68
4.5	Conclusion	70
5	Expanding the Capabilities of Document-Level Event Role Filler	

Extraction with Innovative Position Encoding Methods in Multi-Granularity Contextualized Models	73
5.1 Introduction	74
5.2 Related Work	77
5.2.1 Sequence Tagging	78
5.2.2 BiLSTM-CRF models	78
5.2.2.1 BiLSTM	79
5.2.2.2 BiLSTM-CRF	80
5.2.3 Multi-Granularity Reader	81
5.2.3.1 Embedding Layer	82
5.2.3.2 BiLSTM Layer	82
5.2.3.3 Fusion Layer	83
5.2.4 Position Encoding	84
5.3 Proposed Method	84
5.3.1 Motivation and Analysis	85
5.3.2 Sinusoidal Position Encoding	86
5.3.3 Relative Position Encoding	86
5.3.4 Word embedding	87
5.3.4.1 GloVe Embedding	88
5.3.4.2 Contextualized Embeddings	88
5.3.4.3 Embedding Combination	88
5.4 Experiment	89
5.4.1 Environment and Parameters	89
5.4.2 Dataset	90
5.4.3 Justification of Datasets Used	90
5.4.4 Evaluation Methods	91

5.4.5	Techniques Used in Comparison	92
5.4.6	Experiment Result	93
5.5	Analysis	95
5.6	Conclusion	97
6	Conclusion	99
6.1	Overview of The Previous Methods	99
6.2	Significance of the thesis	101
6.2.1	Text Classification on GCNs	101
6.2.2	Text Classification on HLG	102
6.2.3	Information Extraction	102
6.3	Future Researches	103
	Bibliography	107

List of Figures

1.1	Approach for Automated Rule Extraction [138]	5
2.1	CBOW and Skip-gram Models [89]	17
2.2	Words Vectors in a Dimensional Space	19
2.3	Transformer Structure [119]	30
2.4	Pre-training and Fine-tuning Procedures for BERT [27]	31
3.1	Relations Between the Number of Hidden Layers and Test Accuracy	46
4.1	MSIP Learning Operations [69]	56
4.2	Dynamic Graph Norms for HLG model	61
4.3	Development Sets Accuracy in Epochs	70
5.1	BIO Token-tag Sequences	78
5.2	BiLSTM Model	79
5.3	BiLSTM-CRF Model	80
5.4	Three Layers of BiLSTM-CRF Models	82
5.5	Multi-Granularity Reader [30]	83
5.6	Word Embedding is GloVe and BERT Concatenation plus Positional Encoding	88

Chapter 1

Introduction

Information checking like engineering design drawings compliance checking is a considerable task that amounts to a significant workload carried out by specialists. To save time and to improve productivity, program-based information compliance checking systems have been introduced. However, these checking systems still require personnel to provide input specifications manually, which is both time-consuming and error-prone [72]. To solve this issue, this thesis focuses on research to develop and refine deep learning algorithms that can be utilized for compliance checking tasks. The main aim of this thesis is to enhance the efficiency and robustness of NLP techniques powered by these deep learning methods to suit a specified task. A key objective is to advance the capability of these algorithms to automatically interpret and extract vital information from complex documents written in human language. While information compliance checking serves as a contextual example, the primary goal here is not the application itself, but the advancement of the underlying deep learning methods. The research in this thesis will also include the formulation of novel testing methodologies to verify the performance and robustness of the enhanced algorithms developed, thereby ensuring their broad applicability and scalability across various complex tasks and domains. Overall, this research will result in advancements in the field of deep learning, with potentially wide-ranging implications for the automation and

improvement of labour-intensive tasks in numerous fields. The study stands to redefine understanding of what is achievable with deep learning, and forging a path through which future breakthroughs in machine learning and AI can be made.

1.1 Background

A well-designed compliance information must comply with a series of specific criteria. Users, such as compliance information checkers, typically function by inspecting specific design properties or data to ensure these meet road design criteria. Krish [59] developed a generative design method for multiple criteria compliance information design. However, consistency in the quality and criteria to validate design drawings is very difficult to maintain from person-to-person because of each designer's own experiences, habits and level of professional experience. Thus, checking criteria of designed information, which is also called compliance checking, is essential [93].

It is becoming increasingly difficult for related people to inspect road information criteria manually. According to Belsky and colleagues [6], the analysis and parsing of compliance text are both far more difficult and time consuming to carry out after such information are completed. In addition, road information criteria checking is even more complex than general information criteria checking because road design requires its bespoke criteria in order to meet local conditions and safety standards. Moreover, different countries and cities have different criteria and national standards to abide by. Amin and Amador [3] developed a road pavement criteria-based system in Canada, while Jamroz and colleagues [48] introduced a tool for road infrastructures safety design criteria in Poland. It has been reported that the critical analysis of information and their checking is a waste of a lot of time, depending on the designer's expertise and size of the designed file for checking [29]. This explains, at least, partially, why there is great challenge for a criteria checking person or body to complete all checks manually.

Based on the above challenges, it is necessary to develop a computer program-based tool, which can serve as an automated solution to check against criteria. Such information checker should ensure that a specific design property or data element meets road design criteria. According to Goguelin and coworkers [32], a suitable tool for drawing design can help designers comply with their design criteria in an effective and efficient way. Although the use of systems like AutoCAD (Autodesk, San Rafael, CA, USA) has the advantage of APIs to access entities and data in information files as an automated, user-programmable tool to check the design criteria of road information more easily, it remains difficult to make the computer program understand Road design criteria documents and files. Typically, road design criteria are written in human languages, such as English. Thus, understanding human language becomes a key challenge to develop an automated road information compliance checking system.

NLP could be a method to apply in order to solve this challenge. According to Cambria and White [14], NLP provides a theoretical basis for computer systems to understand human languages, and the development of NLP is described to have arisen from three stages, from 1950 to 2100, which are the Syntactics, Semantics and Pragmatics stages. Now it is in the stage semantics.

In the Semantics stage, there are 3 types of NLP which are Endogenous NLP, Taxonomic NLP and Noetic NLP [14]. Endogenous NLP uses a machine learning method to process language in documents which only rely on the inner knowledge of the documents. It focuses on the meaning of an individual word or sentence. Research by Daubler and colleagues [26] is an example of using endogenous NLP, a method that is straightforward, effective and that can save time and personnel [77]. For applications using road design compliance documents, only the knowledge itself needs to be realized, and not external knowledge, thus Endogenous NLP could be useful for carrying out criteria extraction in such documents.

Creating a document checking system for this task is not trivial. It requires deep domain knowledge, expertise in NLP, and a well-defined data pipeline for

training, testing, and deploying the model. Additionally, it's also important to continuously monitor and update the model as new standards or criteria, or both, are introduced in the road design industry. Despite these challenges, the payoff of an automated tool for compliance information checking is highly significant, reducing manual labour, time and potential errors in the design process from manual checking. My PhD research has led to significant advancements in deep learning algorithms, specifically tailored for complex text classification and information extraction tasks. These innovations offer profound implications for domains requiring precise information extraction and classification, including compliance checking. This thesis showcases the development of novel methodologies that substantially enhance the capabilities of NLP techniques. Although the potential applications of this research span a wide array, the essence of this work is to demonstrate how these deep learning advancements can be pivotal in automating and improving compliance checking processes across various industries. The primary challenge addressed in the study will be the development of more robust, efficient, and interpretable deep learning algorithms. Innovative techniques will be explored to improve various aspects of deep learning models, such as training efficiency, generalization capabilities, robustness to different data distributions, and interpretability of the models. My research will also delve into novel architectures and learning techniques and push the boundaries of what is currently achievable with deep learning.

My research is centred on enhancing deep learning algorithms to impact various sectors significantly. It aims to refine these algorithms for broader applicability, offering advanced solutions for intricate challenges faced across different text tasks. This study will provide insight into any field where deep learning is used, so as to lead to more efficient, robust, and interpretable models that can handle complex tasks with increased accuracy and speed. To succeed, the research requires a robust data pipeline for training, testing, and improving the deep learning models. This can involve designing new benchmarking methods to accurately

measure the performance of the developed algorithms. Positioned at the cutting edge of deep learning research, my study introduces innovative methodologies to transform how deep learning algorithms are understood and utilized, offering novel approaches to complex problem-solving in real-world contexts.

1.2 Study Framework

Understanding document content is crucial to this thesis. A computer-aided method is important to recognise and understand related document, extract key words and sentences, and transfer processed information to downstream purposes. Zhang and El-Gohary [138] introduced an automated information transformation method of an NLP for regulatory compliance checking in the construction domain. The flow of automated compliance information checking is shown in fig. 1.1. My study focuses on advancing deep learning methodologies in the context

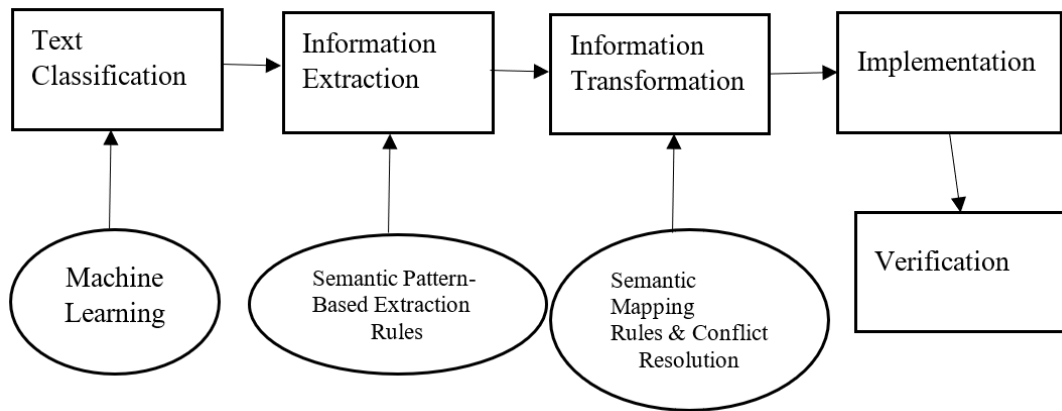


Figure 1.1: Approach for Automated Rule Extraction [138]

of Text Classification and Information Extraction, two crucial tasks in the automation of document content understanding.

In the field of Text Classification, GCNs will be further researched, a contemporary approach that has garnered significant attention in the field of Artificial Machine Intelligence (AMI) and NLP. The potential of GCNs to build more sophisticated accompanying graph structures than traditional neural networks for

feature engineering is acknowledged in the field. In particular, this study seeks to push the current boundaries by adding cumulative TF-IDF document-document relations as features in the graph. The aim is to create complex and rich relation-based adjacent matrix graphs as features to achieve superior accuracy in text classification, thus optimizing the use of GCNs in this context. In parallel, this study will also examine the capabilities of pre-trained language models such as BERT, known for their ability to capture complex linguistic information in large text corpora. However, there are limitations in handling complex semantic relationships as well as noisy or incomplete data. Thus, GraphNorm will be introduced and evaluated, which is an innovative method aimed at enhancing the performance of HLG models. By employing dynamic normalization layers for each GNN module in the HLG model, GraphNorm is designed to increase model accuracy and minimize memory consumption during the training process.

In the area of Information Extraction, this study recognizes the significant potential of Document-Level Event Role Filler Extraction. Specifically, innovative position encoding strategies are explored within a multi-granularity contextualized model. By integrating sinusoidal and relative position encodings at different levels of granularity in the model's architecture, this study will attempt to capture the contextual interdependencies and the internal hierarchical structure in text more effectively. This approach is expected to provide a richer, more contextualized representation of words, essential for tasks like event extraction.

1.3 Thesis Structure

My research will explore and enhance state-of-the-art deep learning techniques and their applications to Text Classification and Information Extraction in the following chapters:

In Chapter 2, foundational groundwork is established for this study. This includes introducing key concepts and terminologies, such as Deep Learning and NLP, which form the backbone of the research. This sets the stage for a deeper

understanding of the later sections where the novel methods will be tested to enhance Text Classification and Information Extraction.

In Chapter 3, the application of GCNs [55] is examined for text classification, highlighting their ability to create complex and rich relation-based adjacent matrix graphs as features for training, and propose an innovative method to optimize this process. Also, a novel document-relational GCNs model that incorporates cumulative TF-IDF document-document relations as features, is presented. By doing this, the aim is to enhance the accuracy of text classification in an attempt to redefine the standard methods of using GCNs in this context.

In Chapter 4, attention is turned to the use of pre-trained language models such as BERT [27] for capturing complex linguistic information. Despite the power of these models, these often struggle to handle complex semantic relationships and noisy or incomplete data. In response to this, the integration of GraphNorm, a new method designed to improve the performance of HLG models [69], is proposed. By introducing dynamic normalization layers for each GNN module in the HLG model, the goal is to increase model accuracy, enhance word and sentence representation learning during training, and to handle complex semantic relationships more effectively.

Chapter 5 focuses on the extraction of information from document-level events. We propose a method of integrating novel position encoding strategies within a multi-granularity contextualized model for improved Document-Level Event Role Filler Extraction [30]. Our approach captures contextual interdependencies and internal hierarchical structures in the text, providing a richer, more contextualized representation of words, crucial for tasks such as event extraction.

In summary, the aim of this thesis is to explore, innovate and enhance current deep learning techniques for Text Classification and Information Extraction. Our research contributes to the broader goal of evolving the capabilities of automated document content understanding systems, thereby opening new avenues in the field of Natural Language Processing.

Chapter 2

Background

In this chapter, some NLP-based concepts and algorithms will be introduced. These algorithms are essential basic components of the models developed and described within this thesis.

2.1 NLP

NLP is a specialized area within AI and linguistics, dedicated to the processing and utilization of human natural language. This field encompasses diverse elements and procedures, including cognition, comprehension, and language generation [95]. The process of natural language cognition and understanding involves converting input language into symbols and relationships, which are then processed according to the intended purpose [83]. On the other hand, a natural language generation system transforms computer data into human-like language. Thus, NLP plays a crucial role in enabling computers to comprehend and generate natural language, facilitating applications such as information retrieval, sentiment classification and language translation [56]. In the scope of this research, NLP serves as the foundational technology for automating the extraction and interpretation of complex documents, in the context of information compliance checking. This area of research has achieved state-of-the-art progress in NLP by focusing on

development of deep learning algorithms to enhance its efficiency and robustness for compliance checking.

2.2 NLP Information Checking

The advent of NLP has ushered in transformative changes in the realm of information checking, particularly in domains where accuracy and efficiency are paramount. Recent advancements in deep learning have significantly expanded the capabilities of NLP applications, making it possible to automate and enhance the compliance checking of complex documents, such as legal contracts, regulatory filings, and engineering drawings, with unprecedented precision and speed [63, 39].

The significance of NLP in automating information checking cannot be overstated. Traditional methods, which relied heavily on manual review by domain experts, were not only time-consuming but also prone to errors due to human fatigue and oversight [91]. The integration of NLP has fundamentally changed this landscape. For instance, the potential of machine learning models is demonstrated to automate the verification of legal documents against regulatory standards, showcasing a significant reduction in processing time and error rate compared to traditional methods [143]. Similarly, NLP techniques are applied to the compliance checking of engineering designs, illustrating how deep learning models can effectively interpret and validate CAD drawings against established criteria [103].

Despite these advancements, the application of NLP in information checking faces several challenges. One of the primary limitations is the handling of context and semantic complexity in human language [1]. While models like [7] and GPT [127] have made substantial strides in understanding context, their application in specialized domains such as legal or technical document analysis still encounters difficulties due to the unique vocabulary and stylistic nuances of such texts. For example, there are challenges of extracting and interpreting technical standards

from engineering documents, where the model’s performance was hindered by the specialized language and the intricate relationships between different document sections [76].

Moreover, the issue of data scarcity and the need for domain-specific training sets is a notable obstacle. Deep learning models require vast amounts of annotated data to learn effectively, yet, in many specialized fields, such data is scarce or proprietary, limiting the development and training of models. Research on using NLP for financial compliance checking [99] elucidated this challenge, pointing out the need for more robust methods of data augmentation and transfer learning to mitigate the lack of domain-specific datasets.

In light of these challenges, this thesis seeks to explore innovative approaches to enhance the effectiveness and robustness of NLP techniques for information checking tasks. Specifically, it aims to develop methodologies that improve the context-awareness and domain adaptability of NLP models. By focusing on the development of deep learning algorithms that can better handle the semantic complexity of specialized texts and leveraging novel techniques for data augmentation, this research endeavours to bridge the gap between the current capabilities of NLP applications and the demanding requirements of information checking in specialized domains.

This pursuit is not only crucial for advancing the state of NLP research but also holds significant practical implications. By enhancing the accuracy and efficiency of automated information checking systems, the proposed research stands to offer substantial benefits across various industries, from reducing operational costs and time frames to improving compliance and reducing the risk of errors.

2.3 TF-IDF

TF-IDF is a statistical method employed to evaluate the importance of a word within a document set or corpus [53]. The significance of a word is directly proportional to its frequency within a specific document but inversely proportional to

its occurrence across the entire corpus. Search engines commonly utilize various forms of TF-IDF weighting as a measure to assess the relevance of a document to a user’s query.

The core concept of TF-IDF is illustrated in the following scenario: if a word or phrase appears frequently in a single article (high Term Frequency) but infrequently in others, it is considered to possess substantial category discrimination, making it suitable for classification purposes.

Breaking down the components of TF-IDF:

- Term Frequency (TF): This represents the frequency of a word in a document divided by the total number of words in that document. The rationale is that the more often a term appears in a document, the more significant it is.
- Inverse Document Frequency (IDF): This gauges the importance of a word by considering how many documents contain that word. If a word is prevalent across all documents, it is less distinctive and, therefore, less important. IDF is calculated as the logarithm of the total number of documents divided by the number of documents containing the term.

Thus, the equation of TF-IDF is represented by equation (2.1)

$$\begin{aligned}
 \text{TF}_{i,j} &= \frac{N_{i,j}}{\sum_{k \in j} N_{k,j}} \\
 \text{IDF}_i &= \log \frac{|D|}{|j : t_i \in d_j| + 1} \\
 \text{TFIDF}_{i,j} &= \text{TF}_{i,j} \times \text{IDF}_i
 \end{aligned} \tag{2.1}$$

where $\text{TF}_{i,j}$ represents the frequency of the term (word) i in the document j , and IDF_i corresponds to the logarithmic value of the total number of documents divided by the number of documents containing the word i .

While TF-IDF offers simplicity, speed in computation, and ease of understanding, it has drawbacks. It may fall short in comprehensively measuring the importance

of a word in an article, especially when there is a lack of important words [101]. Additionally, TF-IDF struggles with the issue of synonyms. To address the contextual structure of words, the use of vectors may be necessary.

2.4 Word Embeddings

In NLP, words, which form sentences, are the most fine-grained component of natural languages. Of course, sentences form paragraphs, chapters, and documents. Therefore, to deal with NLP problems, it is important to focus on words first. Developing a way to make computers read words is a significant challenge. Reading a word like an image is a complex task as a word can have a plethora of meanings and information. For this task, vectorisation is a key method to solve meanings of different words, while word embedding is a transformation method that converts incomputable and unstructured words into computable and structured vectors.

Word embeddings are vectorized descriptions of words designed to encapsulate their semantic significance [86]. These embeddings are generated using algorithms like Word2Vec [86], GloVe [96], FastText [10] and BERT [27]. Unlike traditional one-hot encoding, which represents each word as a unique high-dimensional vector, word embeddings map words into a lower-dimensional, continuous vector space. Words that appear in similar contexts have vectors that are close to each other in this space. Word embeddings find extensive application in diverse NLP tasks, including but not limited to named entity extraction [61], sentiment classification [61], language translation [112], recommendation systems [23] and question-answering systems [100]. They have become a foundational component in the field of NLP.

2.4.1 Bag-of-words Model

Bag-of-words model, a straightforward yet widely utilized information representation in NLP and information extraction, simplifies the expression of sentences or entire documents. In this model, the representation involves a collection of words, disregarding both grammar and the sequential arrangement of the words [35]. Each word's occurrence within the document is autonomous and does not rely on the presence of other words. TF-IDF, as mentioned above, is a popular and sophisticated example of the bag-of-words models. Other methods, such as one-hot encoding, are also classical examples of Bag-of-words model.

Despite its popularity, the Bag-of-words model has clear drawbacks. Specifically, as the document's vocabulary expands, the number of words utilized in each sentence remains limited, typically a dozen or so at most. This results in sparse matrices for each sentence, potentially straining memory and computing resources. Furthermore, since the model treats words individually, it overlooks the sequence and relationships between words in a sentence. Consequently, there are instances where the representation of sentence meaning may lack accuracy [95].

Another example of Bag-of-words model is n-gram [15, 87]. An n-gram refers to a consecutive sequence of n words extracted from voice or text in an NLP dataset. The term 'n-gram' is derived from the word 'gram', which is often used in linguistics to denote a unit or particle. In the context of text processing, these 'items' are typically words, but they can also be characters or other sub-words units. Also, N-grams represent sequences of words by grouping contiguous sets of n words together. The idea behind n-grams is to capture local linguistic structures or patterns. The value of n determines how many words are grouped together in each sequence. Though words sequence is added to sentences, the n-gram lacks long-term dependence and can only model a limited length of words. As the sentence length increases, the parameter space grows exponentially. The n-gram model data is sparse and it has poor generalization ability.

2.4.2 Pre-trained Word Vectors

A language model produces word vectors through the training of the Neural Network Language Model (NNLM), with the word vectors serving as an unintended outcome of the model. The fundamental idea behind the NNLM involves predicting words occurring in a given context, essentially learning co-occurrence statistics through this contextual prediction. [51]

Pre-trained word vectors are essentially word embeddings that have been previously trained on a large dataset, often encompassing a broad range of vocabularies and context. These vectors serve as a form of “knowledge base” that can be plugged into various NLP models to enhance their performance. Because they are trained on extensive data, NNLMs encapsulate rich semantic and syntactic information about words, capturing various relationships such as similarity, opposition, and many more nuanced aspects of meaning.

The primary advantage of using pre-trained word vectors is that they can significantly reduce the amount of time and computational resources required for developing NLP models. Training word embeddings from scratch can be computationally expensive, intensive and time-consuming. Utilizing pre-trained embeddings can overcome these challenges so as to move directly to subsequent applications such as sentiment classification and named entity extraction.

The generation of these pre-trained vectors often involves training a neural network-based language model on a large corpus. Algorithms like Word2Vec, GloVe, FastText, and more recently, transformer-based methods like BERT and Generative Pre-training Transformer (GPT), are commonly used for this purpose. During this training, the model learns to predict a word based on its context (or vice versa), effectively learning the co-occurrence statistics between words. Once trained, the weight matrices in these models serve as the word vectors.

These pre-trained word vectors can often be fine-tuned further on a specific task or dataset to make them more suited to the problem being solved, thereby achieving even greater performance outcomes. Overall, pre-trained word vectors have

been a significant catalyst in the rapid development and deployment of efficient and effective NLP applications.

2.4.2.1 CBOW Model

CBOW model is a neural network approach for generating word embeddings, commonly used in the Word2Vec algorithm. With CBOW, the objective is to forecast a target word by considering the surrounding context words. This process entails encoding these words, calculating the average of the vectors representing the context words, and subsequently utilizing this average as input for a neural network. The network then tries to predict the target word. The model is trained iteratively on a large corpus, adjusting its internal weights based on prediction errors to improve the quality of the word embeddings. CBOW is computationally efficient and is widely used in various NLP tasks, although its averaging approach can sometimes lose nuanced word relationships [89].

2.4.2.2 Skip-gram Model

Skip-gram model [90] utilizes similar method as CBOW which uses context to infer a word. However, rather than predicting the current word from its context, Skip-gram endeavours to classify the current word to the fullest extent by leveraging another word in the same sentence. Specifically, Skip-gram employs each current word as input for a log-linear classifier, incorporating successive projection layers to predict a set number of words occurring both before and after the current word. It has been observed that enhancing the range enhances the quality of the resulting word embeddings, albeit at the cost of increased computational complexity. Given that words farther away from the current word typically exhibit lower correlation, Skip-gram mitigates this by assigning less weight to words at a greater distance, achieved through reduced sampling from these distant words in the training examples. Figure 2.1.

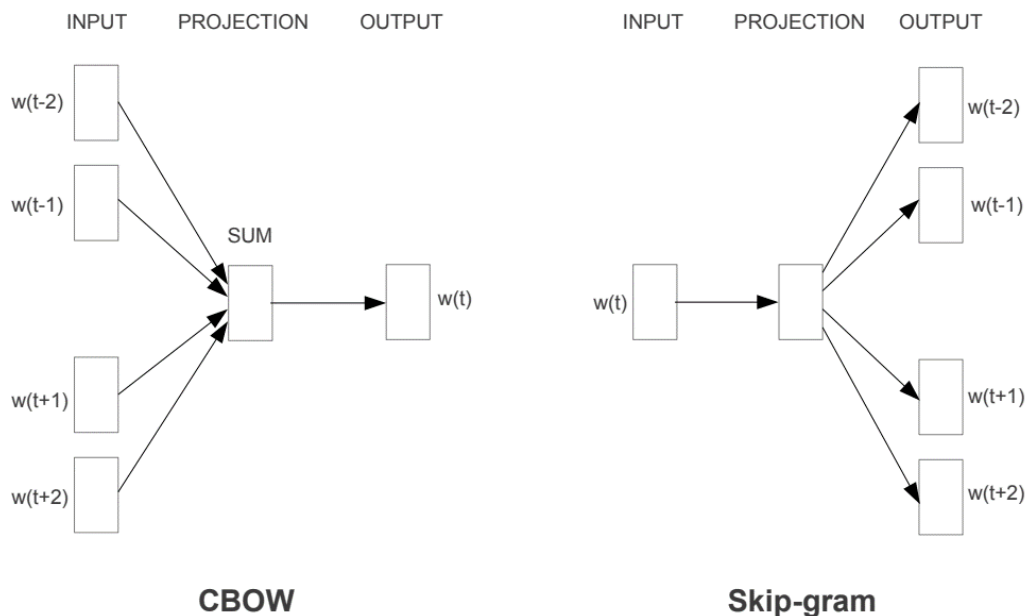


Figure 2.1: CBOW and Skip-gram Models [89]

2.4.2.3 Word2vec

In the development of a sophisticated word embedding technique, Word2vec [86] utilizes a series of interconnected models. These models are two-layer and shallow neural networks specifically crafted to reconstruct the linguistic context of words. The process involves training on a substantial text corpus and subsequently constructing a vector space, typically comprising hundreds of dimensions, where each distinct word in the corpus is associated with a corresponding vector in this space. Word2vec has the flexibility to utilize either of two model architectures (CBOW and Skip-gram) for generating distributed representations of all words within a trained corpus. In both architectures, the Word2vec utilizes context sliding dual windows of words and considers both given words while traversing the entire corpus. The order of context words does not affect predictions. In the CBOW model, it anticipates the current word based on a context window of surrounding words, whereas the skip-gram architecture, as previously discussed, employs the current word to forecast surrounding windows of context words. Skip-gram mod-

els assign greater significance to nearby context words compared to those farther away. CBOW demonstrates efficiency in the training process, while skip-gram proves more effective for less common words.

Word2vec provides a simple and relatively low dimension solution for tasks involving the presentation of millions of words. As presented in Figure 2.2, similar words generate similar vectors in the dimensional space. Given a large enough dataset, Word2Vec is trained from a large dataset and can then go on to make robust estimates about the meaning to a word through a vector based on their context occurrences. These estimates yield words are related to other words from vector expression in the corpus. For instance, the word vectors for 'boy' and 'girl' exhibit a high degree of similarity. Through vectorization of word embeddings, one can identify close approximations of word similarities. Illustrated in Figure 2.2, the high-dimensional embedding vector of 'boy,' the embedding vector of 'man,' and the embedding vector of 'woman' all result in vectors that closely align with the embedding vector of 'girl.' Thus, it is common to express a word in a corpus using a unique vector. And simple increase in dimensions of vectors can generate rich features for every word in a corpus.

2.4.2.4 GloVe

GloVe is an unsupervised machine learning algorithm for generating word embeddings [96]. It has gained prominence for its ability to capture intricate relationships between words purely based on their co-occurrence statistics. Unlike other well-known word embedding algorithms such as Word2Vec, GloVe operates on an aggregated global word-word co-occurrence matrix, exploiting both the global statistical and local semantic information of a text corpus. The core idea is to make sense of how frequently each pair of words appears together, thereby encapsulating a broad array of semantic and syntactic relationships.

The initial stage in GloVe involves building the matrix that represents the co-occurrence of words. For every word in the corpus vocabulary, the algorithm

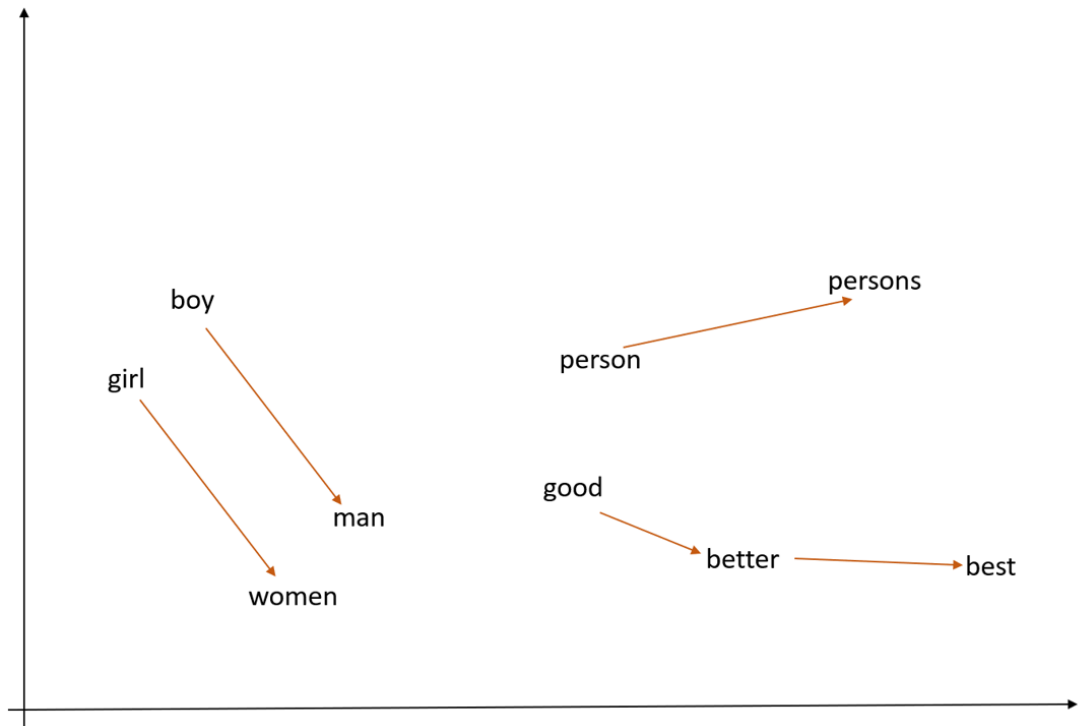


Figure 2.2: Words Vectors in a Dimensional Space

tracks how frequently it co-occurs with every other word within a defined window size. This creates a matrix where the rows and columns represent target words and context words respectively. The value of each matrix cell represents how often these two words (target word and context word) are close to each other in the corpus. This co-occurrence matrix serves as the basis for creating embeddings.

Once the co-occurrence matrix is built, GloVe employs matrix factorization techniques to derive the lower-dimensional word vectors. The objective of the algorithm is to discover optimal word vectors that effectively represent co-occurrence probabilities. It achieves this by minimizing an objective function that measures the disparity between logarithm of co-occurrence counts and dot product of word vectors. A distinct advantage here is computational efficiency. Matrix factorization methods are well-established and can be optimized for performance, allowing GloVe to scale well with the size of the dataset.

GloVe's unique point is its ability to amalgamate the benefits of both global sta-

tistical methods and local context window methods. The global statistical aspect allows it to capture relationships over the entire corpus, making the embeddings more robust. On the other hand, the local window feature allows it to capture various semantic and syntactic nuances, like plurals, synonyms, or antonyms, thus creating a richer embedding space.

2.4.3 Pre-trained Language Model - BERT

In previous section, models that featured pre-trained word vectors were introduced to express words which can simply do downstream NLP tasks. However, a word's pre-trained vector is a static vector which has disadvantages that make it difficult to optimize algorithms for a specific task, and this approach is typically difficult to use to solve polysemous word problems. Thus, a dynamic language model is needed to apply to NLP tasks. Generally, dynamic models can learn not only previous knowledge such as pre-trained corpus but also now knowledge from higher-level information from downstream tasks. BERT is an example of such a dynamic model, and will be introduced in this section. BERT [27] stands as a pre-trained language representation model that distinguishes itself by moving away from conventional straightforward NLP models or integration of two NLP models for pre-training. Instead, it asserts itself as a novel masked language model capable of generating profound bidirectional language representations.

2.4.3.1 Transformer

The Transformer [119] is a neural network architecture commonly employed for tasks in natural language processing. Diverging from its forerunners, namely Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), it enables enhanced parallelization, resulting in expedited training times. The Transformer consists of an Encoder-Decoder structure, each made of multiple identical layers. A notable attribute is its 'attention mechanism', which make the model pay attention to distinct segmentations of the input words for output

generation. This architecture excels at handling long-distance dependencies and is highly scalable, forming the foundation for models like GPT and BERT. BERT comprises a series of Bidirectional Transformer components. The Transformer model utilizes the attention mechanism to accelerate the training process. It is a deep learning model built entirely on the self-attention mechanism, designed for efficient parallel computing. The inherent complexity of the model contributes to its superior accuracy and performance compared to the previously prevalent RNN neural network. The Transformer structure is as Figure 2.3.

The internal details of Transformer are concealed through a black box function, revealing only its inputs and outputs. Moreover, the Transformer architecture can be stacked to create a more extensive neural network. BERT can likewise be conceptualized as the stacking of Transformer encoders.

2.4.3.2 BERT

Figure 2.4 introduces different pre-trained BERT models to apply to different downstream tasks of NLP. The input for BERT consists of representations corresponding to each token. In the Figure 2.3, there are token and token representation blocks. with the purpose of addressing specific downstream classification tasks, a designated classification token (*CLS*) is inserted at the start of each input text. The output of the last Transformer layer linked to this classification token is employed to consolidate the representation information for the entire text.

Given that BERT is a pre-trained model adaptable to various NLP tasks, the input text should accommodate either a single sentence (for tasks like text sentiment classification and sequence labeling) or more than two sentences (for tasks such as text summarization, natural language inference, and question answering). The model possesses the capability to discern the sentence boundaries, achieved by inserting segmentation tokens ([SEP]) after each sentence in the sequence tokens. Additionally, a learnable segment token is added to each token representation to indicate whether it belongs to the first sentence or the second sentence.

The output of a BERT model is proportional to the number of input tokens. The classification token (*CLS*), denoted as C , mirrors the output of the final Transformer layer and incorporates information from other tokens within the same layer. For certain tasks like question answering and sequence labeling, the input is forwarded to an extra output layer for prediction. In sentence-level tasks such as natural language inference and sentiment classification, the classification token C is fed into an additional output layer, clarifying the insertion of a specific classification token before each token sequence.

2.4.3.3 Strengths and Innovations

The primary strength of BERT lies in its pre-training on a large corpus of text, which enables the model to develop a rich understanding of language syntax and semantics before being fine-tuned for specific tasks. This pre-training aspect has allowed for significant improvements in NLP applications with limited domain-specific training data, by transferring the general language understanding developed during pre-training to specific tasks [27, 76].

Moreover, the adaptability of BERT to a wide range of domains and languages further underscores its versatility. Its architecture facilitates fine-tuning for specific tasks without substantial modifications, making it a powerful tool for domain-specific applications, including legal document analysis.

2.4.3.4 Limitations and Challenges

Despite their strengths, pre-trained models like BERT also present challenges. One significant limitation is their requirement for substantial computational resources for training and fine-tuning, which can be a barrier for researchers and practitioners without access to high-performance computing facilities. Additionally, while BERT excels at understanding context within the text, it sometimes struggles with highly specialized or technical language outside of its training corpus, limiting its effectiveness in certain domain-specific applications without

further adaptation or domain-specific pre-training [7, 64].

2.4.3.5 Application in Information Checking

The capabilities of BERT and similar models hold significant potential for enhancing information checking processes. By understanding context and semantics at a deep level, these models can automate the extraction, classification, and verification of information from complex documents with a high degree of accuracy. For instance, in the domain of legal document verification, BERT’s ability to discern subtle differences in language can be leveraged to identify non-compliance with regulatory standards or to verify the accuracy of information presented in contracts [16].

2.5 GCNs

The abbreviation GCNs refers to Graph Convolutional Networks, a category of neural networks explicitly crafted to process graphs as input data. Unlike conventional neural networks like CNN [54], which are well-suited for grid-like data format like pictures, and RNN, designed for sequential data like time series or text, GCNs are specifically tailored for data that can be represented in the form of graphs.

Graphs collectively represent a very flexible and expressive data structure that can model a wide variety of real-world phenomena, including social networks, molecular structures, transportation systems, and much more. Every node in the diagram symbolizes an entity, and each connection signifies a relationship between these entities. Nodes and connections may possess specific attributes and features.

GCNs research is primarily focused on data with Euclidean domains. A distinctive characteristic of Euclidean domain data is their regular spatial structures, such as the regular squares in images or the one-dimensional sequences in audio

data. This regularity allows features to be represented by one-dimensional or two-dimensional matrices, making GCNs more efficient for processing.

When aiming to represent a node, a convenient and effective approach is to consider its surrounding nodes, including neighbors and their neighbors. In the context of graph node representation, the fundamental concept is that each node continuously adjusts its state influenced by its neighbors and distant points until a final equilibrium is achieved. The strength of influence from neighbors is directly proportional to the closeness of their relationship. There are multiple forms of GCNs, the form used in this thesis will be introduced.

$$\begin{aligned}
 H_{l+1} &= r(\tilde{A}H_lW_l) \\
 \tilde{A} &= D^{-1/2}AD^{-1/2} \\
 D_{ii} &= \sum_j A_{ij} \\
 r(x) &= \max(0, x)
 \end{aligned}
 \tag{2.2}$$

Where H is an input layer. A is an adjacent matrix. W_l is the l -th layer weight matrix. $r(\cdot)$ is an activation function which is ReLU. This concept is grounded in the understanding that a node’s characteristics are interconnected with those of all its neighbouring nodes. The multiplication of the adjacency matrix A by the feature matrix H is akin to aggregating the features of a node’s neighbouring nodes. Such multi-layer hidden layer superposition can utilize the information of multi-layer neighbors. Self-degree matrix D is introduced to solve the self-propagation problem.

2.6 Normalization

Normalization methods in deep learning aim to make the training of neural networks more stable and efficient by ensuring that the input features or activations have consistent statistical properties. The normalization offers a host of advantages that collectively contribute to more effective and efficient model training.

These methods accelerate the convergence rate, enable higher learning rates, mitigate issues such as vanishing and exploding gradients, and reduce sensitivity to weight initialization [47]. Additionally, they provide a mild form of regularization that improves generalization to unseen data, simplify hyperparameter tuning, and make it easier to train complex, deep architectures. Overall, normalization serves to stabilize, speed up, and enhance the performance of neural network training [4].

2.6.1 Batch Normalization

Batch Normalization [47] is designed to tackle the issue of internal covariate shift, wherein the distribution of inputs for each layer undergoes changes throughout the training process. This makes the training process slow and unstable. BatchNorm normalizes the activations of each layer (or alternatively, the inputs to each layer) by making them zero-mean and unit-variance. BatchNorm is widely used in CNN and also applicable in fully connected layers. However, it's less effective in RNN. A batch *bof* size m :

$$\begin{aligned}
 \mu_b &= \frac{1}{m} \sum_{i=1}^m x_i \\
 \sigma_b^2 &= \frac{1}{m} \sum_{i=1}^m (x_i - \mu_b)^2 \\
 \hat{x}_i &= \frac{x_i - \mu_b}{\sqrt{\sigma_b^2 + \varepsilon}} \\
 y_i &= \gamma \hat{x}_i + \beta
 \end{aligned} \tag{2.3}$$

where μ_b and σ_b^2 are mean and variance of the batch. γ and β are learnable parameters, and ε is a small constant to prevent division by zero. \hat{x}_i is the normalized result. y_i is the final normalization result after dynamic scale and shift.

2.6.2 Layer Normalization

Layer Normalization is designed to be effective regardless of the mini-batch size. It normalizes the sum of the activations in a single layer for each individual example. Layer Normalization is particularly useful for sequence models like RNN and Long Short-Term Memory (LSTM), where batch sizes can be dynamic. For each feature vector x across all dimensions:

$$\begin{aligned}\mu &= \text{mean}(x) \\ \sigma^2 &= \text{variance}(x) \\ \hat{x} &= \frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}} \\ y_i &= \gamma \hat{x} + \beta\end{aligned}\tag{2.4}$$

2.6.3 Instance Normalization

Instance Normalization is mostly used in style transfer and generative models. It normalizes each individual instance in each feature channel independently. Instance Normalization is mainly used in tasks that involve image generation, such as style transfer [78] and Generative Adversarial Networks [24]. For each instance in each feature channel:

$$\begin{aligned}\mu &= \text{mean}(x) \\ \sigma^2 &= \text{variance}(x) \\ \hat{x} &= \frac{x - \mu}{\sqrt{\sigma^2 + \varepsilon}}\end{aligned}\tag{2.5}$$

2.7 Event Role Filler Extraction

Event Role Filler Extraction is a specific task within NLP that focuses on recognizing and categorizing entities or phrases in text that play particular roles in events [103]. This involves recognizing the event itself and understanding the roles of various components (or 'fillers') in the context of that event. This task

is essential for many NLP applications, including but not limited to:

- Information Extraction: Populating databases with structured information from unstructured text.
- Text Summarization: Understanding the key events in a document to produce a condensed version.
- Question Answering: Answering queries about who did what, when, and where.
- Knowledge Graph Population: Adding factual information to knowledge graphs based on text.

Event Role Filler Extraction usually involves multiple sub-tasks like Named Entity Recognition (NER) and Relationship Extraction. Advanced machine learning models like CRF, RNN, and Transformers may be employed for this task.

2.7.1 Sequence Tagging

Sequence tagging is an NLP task in which each token, such as a word or sub-word, in a sequence is tagged with a label. Common examples of sequence tagging include Part-of-Speech (POS) tagging, NER and Event Role Filler Extraction. In Event Role Filler Extraction, the goal is to label each token in the sequence based on its role in the event described by the sentence or paragraph.

2.7.2 BiLSTM-CRF Model

A BiLSTM-CRF model combines BiLSTM networks with a CRF for sequence tagging tasks.

BiLSTM networks are an extension of traditional RNN that can capture long-range dependencies in the text and take into account both past and future context. This is achieved by having two LSTM layers run in parallel: One processes the sequence from left to right, while the other processes it from right to left. The

outputs of both LSTMs are then typically concatenated, capturing information from both directions for each token.

CRF is a probabilistic graphical model that is often used for labeling or parsing of sequential data. In the context of sequence tagging, a CRF layer can take the sequence of word embeddings, which are often enriched by BiLSTM, as input and calculate the most probable labels for each token in the sequence, considering not just the individual classification scores but also the transitions between consecutive labels.

2.8 Challenges and Gaps in Information Checking

The evolution of NLP and deep learning technologies has significantly impacted the field of information checking, yet there remain substantial challenges and gaps that current methodologies struggle to overcome. This thesis proposes the development of advanced techniques to address these issues, focusing on the following key areas:

Current deep learning models, while powerful, often suffer from a lack of interpretability, efficiency issues, and robustness in handling diverse and complex datasets. The complexity of construction drawing reviews presents a substantial challenge that requires more sophisticated models for accurate compliance checking [103]. This thesis aims to address these limitations by enhancing the robustness and efficiency of deep learning algorithms, making them more suitable for practical applications in information checking.

The automation of document content understanding necessitates effective text classification and information extraction techniques. However, the semantic complexity of documents and the scarcity of domain-specific training data limit the effectiveness of current models. By advancing deep learning methodologies, this research seeks to improve the accuracy and applicability of these models in real-

world information checking tasks.

The extraction of event roles at the document level is crucial for understanding complex document structures and contents [121]. Existing methods, however, often fail to capture the full context of events, leading to inaccuracies in information extraction [99]. This research recognizes the significant potential of improving document-level event role filler extraction techniques to enhance the precision and depth of information analysis.

2.8.1 Justification for Research Focus

The identified challenges and gaps in the literature underscore the necessity of this thesis's focus. The limitations of current deep learning and NLP models in terms of efficiency, robustness, interpretability, and domain adaptability directly impact their effectiveness in information checking tasks. By addressing these issues, this research aims to significantly advance the field of NLP and deep learning, contributing to the development of more sophisticated and practical tools for automated information checking.

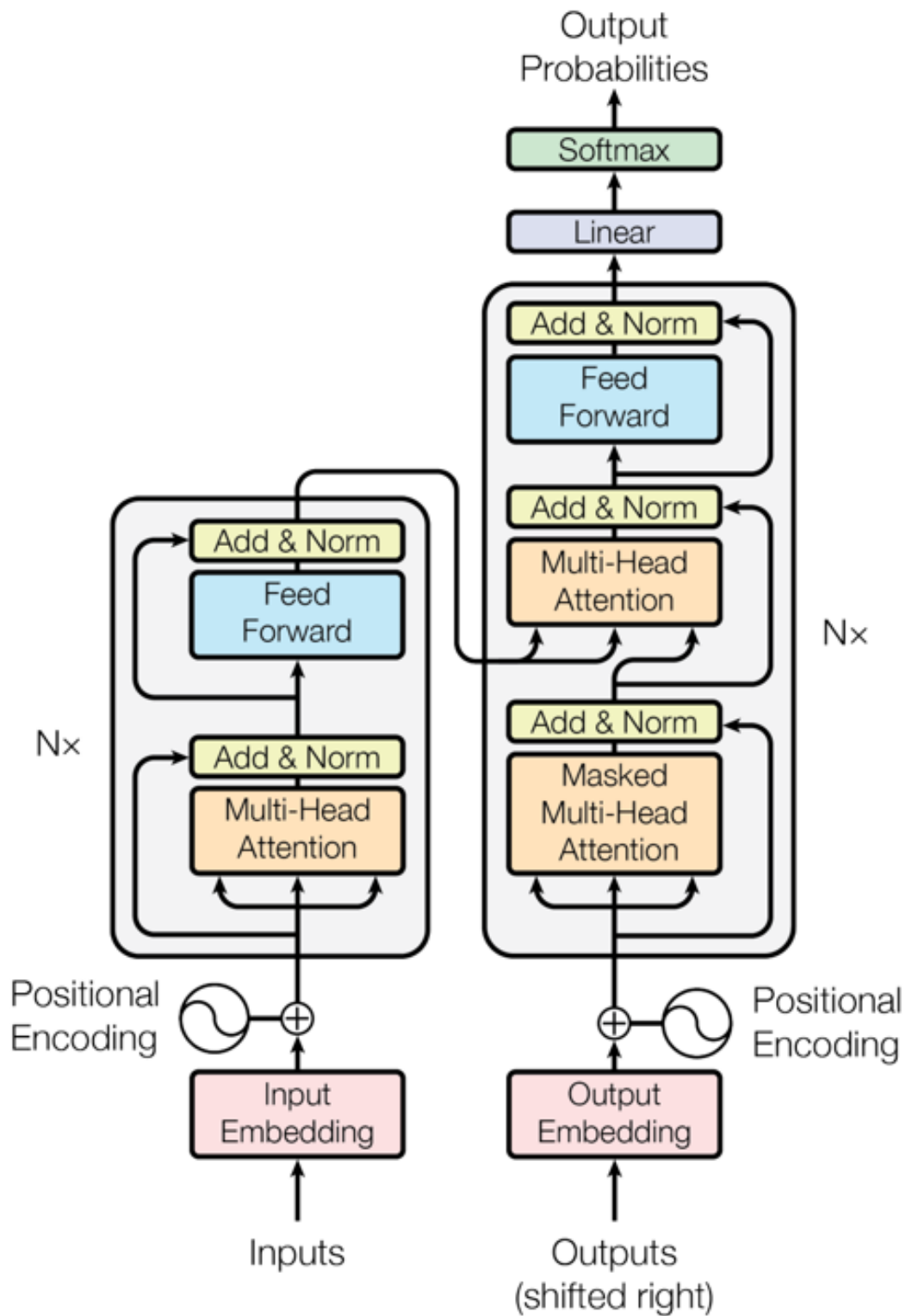


Figure 2.3: Transformer Structure [119]

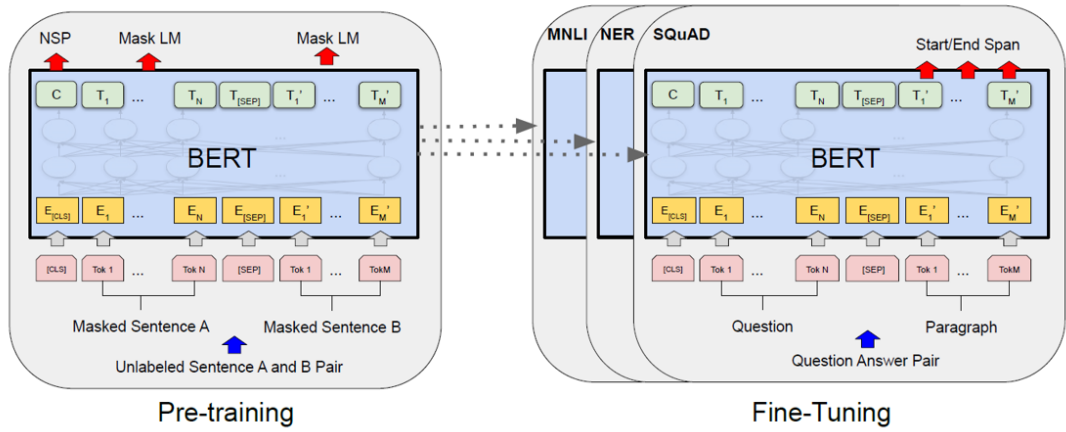


Figure 2.4: Pre-training and Fine-tuning Procedures for BERT [27]

Chapter 3

Document-Relational Graph Convolutional Networks

GCNs have received a lot of attention in the field of AMI and NLP research since these can build more sophisticated accompanying graph structures than traditional neural networks that perform feature engineering. In this approach, a graph is used as a feature in a neural network because it is easy to find relations among nodes. In text classification applications, GCNs can create complex and rich relation-based adjacent matrix graphs as features to be trained. The existing methods, on the other hand, only generate adjacent matrix graphs in GCNs at the word-document and word-word levels as features which lack the exploitation of document-document relationships. In this chapter, a document-relational GCNs method is proposed to achieve an outperformance in accuracy in text classification by adding cumulative TF-IDF document-document relations as features. Our approach aims to enrich the relational features between documents and improve text classification performance. Additionally, a hyperparameter tuning experiment is conducted to find the optimal configuration for the GCNs model. We compare the performance of the method against existing text GCNs models and synonym-augmented data sets across five benchmark text classification datasets. Experimental results in this chapter show that the proposed method achieves

superior classification accuracy in four out of five datasets. Our findings suggest that the incorporation of document-document relationships serves as an effective strategy for improving text classification models.

3.1 Introduction

Text classification represents one of the most important tasks in deep learning problems which can automatically categorize or label text documents into predefined classes or topics. This task is crucial in various NLP applications such as sentiment classification [52], semantic segmentation [146], rule text based recommendation [128] and news classification [66]. Many researchers have attempted to solve text classification problems in different text ranges, including at document [133] and sentence levels [18].

To achieve better classification performance, significant efforts have been made to accelerate the process of model development from machine learning methods [20], [109] to deep learning methods [54], [146], [73]. And the development of deep learning computing efficiency [130], [37] techniques have been assisting in reducing the running time of deep learning methods. Compared to other methods for text classification, deep learning methods have achieved better effective and efficient performance owing to the rapid development of computer hardware and deep learning research.

Topic classification is an important text classification application that uses machine learning [65, 97, 124], and deep learning [5, 49, 136] methods. For example, in road construction documents, a text classification program needs to be developed to categorize the regulatory compliance language. Document inspection for road construction regulatory compliance begins with text classification, as previously described [105, 138]. As different forms of code compliance documents have varied roles in terms of topics, topic classification is an appropriate solution for code compliance text classification.

News classification is one of the most popular and widely used applications in

the field of topic classifications. Every minute of the day, there are countless news releases that individuals cannot keep up with them. However, it is possible to classify news in a manner that allows humans to comprehend it more quickly and correctly. On Facebook and Twitter, for example, a plethora of news is constantly disseminated. Ostensibly, news should be classified into their own categories [106]. As a result, citizens and social media companies should contribute to structuring news data. To comprehend data using the Internet structural approach, Trieu and colleagues [115] developed a word vector technique for news classification based on Twitter data. Furthermore, false news accounts for a significant proportion of online news releases as well. Recently, fake news classification has improved, which can help consumers detect false news [8, 11, 31]. In addition to authentic news reports, news classification can assist healthcare companies in extracting relevant data from social media news [82]. Therefore, news classification is worth investigating to improve its performance effectively and efficiently.

In recent years, graph networks have received considerable interest in deep learning research [129], and the use of graph networks for text classification has produced excellent performance results [9, 126, 74]. Because many applications build huge amounts of data in a structured fashion, graphs have the benefit of storing rich relations between nodes [144]. This type of structural data can then be used as features in a deep learning model to obtain robust results for a realistic relation. The development of deep learning networks has assisted the development of GCNs, such as CNN [60, 13]. To obtain better citation classification performance in an effective and efficient manner, Kipf and Welling [55] developed a GCNs model that can build relations of objects throughout the entire dataset. The GCNs model was then used to classify documents by Yao and colleagues [134], who achieved improved performance in five distinct benchmark datasets, including a news dataset. However, in their GCNs model, they only generated the feature graph using document-word relations. Tang and colleagues [114] constructed a text classification integration model based on GCNs

and achieved competitive results. A document-document link was added into the feature graph to seek an improved result for text classification.

Until recently, there have been only a few techniques that could be used to enhance graph features. Meta-path is one such technique, as it is a type of text classification characteristic that can be used to create document relations. Wang and colleagues [122] utilized a meta-path approach to build linkages between documents, and a machine learning classifier produced superior classification results in two topic-based datasets. The meta-path approach was used by Ding and colleagues [28] and was applied to enrich the knowledge graph for the entire dataset. Moreover, a knowledge graph establishes relationships between documents in a dataset through entities and their interactions represented within the graph [132]. These entities correspond to concepts, objects, or events within the documents, and their relationships are the links that connect them, providing a structured understanding of how documents are related. This process enables more efficient information retrieval, recommendation systems, and data integration, as the graph structure facilitates the discovery of hidden connections and insights among the dataset's documents. In summary, it can be concluded that the link relation between the generated documents in the dataset has better performance on topic sensitive documents. Therefore, text classification effect can be improved from the perspective of feature engineering to enhance the knowledge network in the entire dataset. In conclusion, the link relations between the documents produced in the dataset performs better on the topic sensitive documents to enhance the text classification.

An effective way to enhance the performance of text classification is to build further on feature engineering. Input data forms the basis of feature engineering to achieve good performance, and almost all models require data to drive [79, 80]. There are two feature engineering methods for searching relations inside a dataset, such as graph networks [129, 9, 126, 74] and increasing the size of one dataset, such as data augmentation [57, 22]. Zhang and colleagues [139] found an effective

way to perform data augmentation which simply replaced words or phrases in the dataset by their synonymous ones.

We propose to add document-document link weights to the data graph to improve text classification performance, inspired by the notion of adding visual information to features through meta-paths in terms of feature engineering. Also, the method will be compared with text graph networks and synonym data augmentation. The contributions of this study are as follows:

- We introduce a novel approach to text classification by adding document-document link weights to the data graph to improve its performance. This is achieved through cumulative TF-IDF values for document-document relations.
- The study explores the tuning of appropriate hyperparameters in the document-relational GCNs model to achieve better text classification results.

The remaining parts of this chapter are organised as follows. Previous GCNs models are presented in Section 2. The proposed cumulative TF-IDF document-relational GCNs model is established in Section 3, while Section 4 provides details of results from experiments, as well as their analyses. Finally, the conclusions of this study and the prospects for future work are presented in Section 5.

3.2 Related Work

In this section, previous works on document classification will be described briefly for both graph generation and GCNs model construction.

3.2.1 Motivation and Detailed Analysis

3.2.1.1 Motivation Behind Document-Relational GCNs

The advent of GCNs marked a significant leap in leveraging graph structures for deep learning applications, particularly in text classification tasks. Traditional

neural networks, while effective in various scenarios, often struggle to encapsulate the complex, relational information inherent in textual data. Prior models primarily focused on word-document and word-word relationships, overlooking the rich insights that could be gleaned from document-document interactions. This oversight forms the crux of our motivation: to harness these under explored relationships and enrich the feature set available for text classification, thereby addressing a notable gap in current methodologies [55, 33].

3.2.1.2 Challenges Addressed

Existing methods reveal a notable limitation in capturing the full spectrum of relationships within textual data. For instance, the word-document and word-word adjacency matrices, while powerful, provide a limited view, neglecting the potential contextual depth offered by document-document relationships [62]. This limitation not only constrains the model’s understanding of the corpus but also impacts its ability to accurately classify texts in more nuanced or closely related categories.

3.2.2 Graph Generation

Yao and colleagues [134] constructed a large heterogeneous graph for an entire dataset. The total number of nodes in the corpus was equal to the number of documents and unique words. Thus, the dimensions of the graph adjacent matrix was defined as $(documents + uniquewords) \times (documents + uniquewords)$. Furthermore, in the graph, there were two types of edge relations namely document-word and word-word. TF-IDF was applied to determine the values of document-word edges. The TF-IDF algorithm is in equation (2.1)

Word-word relations are represented by another sort of edge relation representation. To deal with these, traditional approaches have been utilized to count word occurrences in a confined context region, which is known as word co-occurrence statistics [41, 104, 142]. The word-word link weight value in the

constructed graph is calculated using the point-wise mutual information (PMI) method, which is based on the word co-occurrence concept. To track and calculate the frequencies of single and double words co-occurrences, a convolutional window (a fixed length with 20 words) is used. The PMI may be computed using the following formula:

$$\begin{aligned} \text{PMI}(i, j) &= \log \frac{p(i, j)}{p(i)p(j)} \\ p(i, j) &= \frac{\#W(i, j)}{\#W} \\ p(i) &= \frac{\#W(i)}{\#W} \end{aligned} \tag{3.1}$$

where $W(i)$ counts the number of slide windows that contain word i , $W(i, j)$ counts the number of slide windows that contain words i and j , and $\#W$ is the total number of windows sliding past all the documents. A greater PMI value implies a higher correlation between the two words, whereas a lower value suggests a lower correlation between two words. In the graph, only positive PMI values are utilized to indicate the edge of the word-word relation. The negative PMI value is set to zero, indicating that there is no edge in the node pair.

3.2.3 GCNs Model

The document graph in the previous subsection was used as a feature in the GCNs model for further training and testing. Kipf and Welling [55] developed a GCNs model expressed by a multilayer neural network that directly operated on graphs as features. Let $G = (N, E)$ denote the document graph, where N and E are the nodes and edges, respectively. The number of nodes is the sum of the number of documents and unique vocabulary terms. Adjacent matrix A is used to represent the graph. Then, the adjacent matrix A must be normalised as $\tilde{A} = D^{-0.5}AD^{-0.5}$, where D is the degree matrix of A and $D_{ii} = \sum_j A_{ij}$. There is a feature matrix X followed by a normalised adjacent matrix \tilde{A} . In the GCNs model for text classification, the feature matrix is usually set to be an identity matrix as $X = I$, so X is not shown in the future equations. The normalised

adjacent matrix \tilde{A} is a feature of the model. The forward propagation equations for the text classification version are as follows:

$$\begin{aligned}
L_1 &= r(\tilde{A}W_0) \\
L_{j+1} &= r(\tilde{A}L_jW_j) \\
\tilde{A} &= D^{-1/2}AD^{-1/2} \\
D_{ii} &= \sum_j A_{ij} \\
r(x) &= \max(0, x)
\end{aligned} \tag{3.2}$$

where L is the hidden or result layer matrix, j is the layer number, \tilde{A} is the normalised adjacent matrix of the graph, and $r(x)$ is an activation function of the neural network, called ReLU.

3.3 Proposed Method

In this section, cumulative TF-IDF edges method is developed to connect document-document nodes, which differs from the graph that simply uses word-document relations. We generate document relations by adding the multiplication of TF-IDF values of the same words.

$$A'_{ij} = \log(A(\text{row}_i) \cdot A(\text{column}_j)) \tag{3.3}$$

where i, j are document indices, $A(\text{row}_i)$ is the row vector for document i , and $A(\text{column}_j)$ is the column vector for document j . The multiplication of document vectors i and j shows that the TF-IDF values of two documents with the same vocabulary are multiplied and added together. Then the edges that are greater than or equal to 3 are saved because only a large value of document-document relation has a positive effect on text classification performance. Other edge relations of the graph, such as word-word and document-word relations, are identical to those in [134]. Thus, the total adjacent matrix can be defined as follows:

$$A'_{ij} = \begin{cases} \log(A(\text{row}_i) \cdot A(\text{column}_j)) & i, j \text{ are documents} \\ \text{PMI}(i, j) & i, j \text{ are words} \\ \text{TF-IDF}_{i,j} & i \text{ is document, } j \text{ is word} \\ 1 & i = j \\ 0 & \text{otherwise} \end{cases}$$

Document-document, word-document, and word-word relational values comprise adjacency matrix A'_{ij} . The document-document relation is obtained from the equation (3.3). The word-document relation is the TF-IDF value from the equation (2.1). The word-word relation is PMI value from the equation (3.1). The adjacent matrix A'_{ij} has a square shape. The document and vocabulary together constitute the size of the adjacency matrix. The details of how to construct the neighbouring matrix of the graph are shown in Pseudo Code 1.

The adjacent matrix of graph A'_{ij} is then sent to the GCNs model in the next step. Previous studies ([68], [55], [134]) showed that a GCNs model with two layers performs better than that with one layer, whereas a GCNs model with more than two layers does not perform better than that with two layers. Thus, a two-layer GCNs model is applied to train and test the text classification model in this study. Let

$$\begin{aligned} Z &= \text{softmax}(\tilde{A} \cdot r(\tilde{A}W_0)W_1), \\ \text{softmax}(x_i) &= \frac{\exp(x_i)}{\sum_i \exp(x_i)} \\ \tilde{A} &= D^{-1/2}A'D^{-1/2} \\ D_{ii} &= \sum_j A'_{ij} \end{aligned} \tag{3.4}$$

where \tilde{A} is the same as represented in (3.2), W_0 and W_1 are two weight matrices to learn, and $\text{softmax}(\cdot)$ is a probability normalization function that can convert the resultant vector elements into a probability distribution. For each vector, the number of elements is the number of document classes that must be classified.

Pseudo Code 1: Build an adjacent matrix of the graph

```
windows  $\leftarrow$  empty list;
window_size  $\leftarrow$  20;
for doc in documents do
    windows.add (every 20 words);
    for word in doc do
        |  $TF[doc, word] \leftarrow$  number of the word in the doc;
        |  $IDF[doc, word] \leftarrow$ 
            |  $\log(\text{document number}/\text{documents with the word});$ 
        |  $adj[doc, word] \leftarrow TF[doc, word] * IDF[doc, word];$ 
    end for
end for
word_window_freq  $\leftarrow$  empty list;
for window in windows do
    | for word in window do
    | |  $word\_window\_freq[word] \leftarrow word\_window\_freq[word] + 1;$ 
    | end for
end for
word_pair_count  $\leftarrow$  empty list;
for window in windows do
    | for wordPair in window do
    | |  $word\_pair\_count[wordPair] \leftarrow word\_pair\_count[wordPair] + 1;$ 
    | end for
end for
num_window  $\leftarrow$  number of windows;
for key in word_pair_count do
    |  $i \leftarrow$  first word in key;
    |  $j \leftarrow$  second word in key;
    |  $count \leftarrow word\_pair\_count[key];$ 
    |  $word\_freq\_i \leftarrow word\_window\_freq[i];$ 
    |  $word\_freq\_j \leftarrow word\_window\_freq[j];$ 
    |  $pmi \leftarrow \log((count/num\_window)/(word\_freq\_i * word\_freq\_j / (num\_window * num\_window)));$ 
    |  $adj[i, j] \leftarrow pmi;$ 
end for
for each word-doc relation in adj do
    |  $adj[doc, doc] \leftarrow$ 
    |  $adj[\text{doc-words row vector}] \cdot adj[\text{words-doc column vector}];$ 
end for
```

Thus, the total of all elements in a vector after the *softmax* computation is 1, and the highest value position represents the text classification class. Then, the cross-entropy loss function is applied to calculate the error distance between the model forward propagation result Z and the target label:

$$\text{Loss} = - \sum_{d \in Yd} \sum_{c=1}^C Y_{dc} \log Z_{dc} \quad (3.5)$$

where Yd is the label document index, and C is the number of document class dimensions of the outcome feature. Y_{dc} is a vector label and Z_{dc} is a vector result. They represent the target label and predicted result of a document, respectively.

The loss function has two weights that must be learned. The GCNs model does forward propagation and calculates the loss function each time. The derivatives of two weights (W_0 and W_1) are then determined. To reduce loss value, the W_0 and W_1 are updated in the opposite direction as the derivatives. The optimal weights (W_0 and W_1) are discovered, and it can now approach the best document classification prediction using the present model and inputs.

3.4 Experiment

In this section, for text classification accuracy, the updated document-document GCNs will be compared to the original text GCNs and augmented data with text GCNs. Synonym data augmentation [139] is used to enlarge the five datasets which is double the size of each dataset. Furthermore, accuracy of the model is verified by filtering alternative maximum values of document-document graph weights in adjacent matrix of the graph. In the updated document GCNs model, the performance for different numbers of hidden layer dimensions is evaluated.

3.4.1 Datasets

The selection of the five benchmark datasets, 20ng, R8, R52, Ohsumed and MR, for this study is grounded in their diversity, scale, and the unique challenges they present in text classification. This variety ensures that the performance and robustness of our proposed Document-Relational GCNs technique can be thoroughly evaluated across different contexts and types of textual data, thus providing a comprehensive understanding of its applicability and effectiveness.

- **20ng.** Dataset of 20 newsgroup records and 20 different types of news. There are a total of 18846 documents and 42757 vocabulary items. 11314 documents are used for training and 7532 for testing.
- **R8** and **R52.** Two subsets of Reuters data. There are 8 and 20 classes in each, respectively. R8 is equipped with 5485 train documents, 2189 test documents, and 7688 vocabulary items. R52 has 6532 train documents, 2568 test documents, and 8892 vocabulary items in its database.
- **Ohsumed** [38]. Derives from the "MEDLINE10" medical database, which contains titles and abstracts from 270 medical journal articles published between 1987 and 1992. Diseases are divided into 23 categories.
- **MR.** Dataset of movie reviews with only two sentiment classes. There are 10662 documents in total, with half of them being positive reviews and the other half being negative reviews.

The details of the datasets are summarised in Table 3.1. The number of training nodes is the sum of the number of true training documents and the number of valid training documents. The training documents are divided into two categories: actual training and validation. 90% of training documents are for actual training, while 10% are for validation.

Table 3.1: Datasets Summary

Dataset	20ng	R8	R52	Ohsumed	MR
# Actual train	10183	4937	5879	3022	6398
# Valid	1131	548	653	335	710
# Words	42757	7688	8892	14157	18764
# Test	7532	2189	2568	4043	3554
# Total	61603	15362	17992	21557	29426
# Classes	20	8	52	23	2

3.4.2 Experiment Parameters

The PyTorch library is one of the most widely used deep learning frameworks. Both GCNs models (document-relational GCNs and original text GCNs model) are run on the PyTorch deep learning framework. In the training model, the weights for both models are updated using the Adam optimizer [25]. As previously stated, the drop out rate is 0.5, and the GCNs model has two layers.

3.4.3 Hidden Layers

Yao and colleagues [134] chose 200 hidden layer dimensions of the GCNs model in their experiment. According to experiment, it is difficult to obtain good test performance with as few as 200 hidden layer dimensions. Consequently, a number of hidden layer dimensions are investigated, as shown in Fig. 3.1. When 1200 and 1500 hidden layers are applied in the document-relational GCNs model, better test accuracy results are achieved.

3.4.4 Justification for Comparative Techniques

To streamline the justification for the inclusion of Document-Relational Graph Convolutional Networks (Doc-Relational GCNs), Text Graph Convolutional Networks (Text GCNs), and Synonym-Augmented GCNs in the comparative study, we focus on the unique contribution and rationale behind each approach within

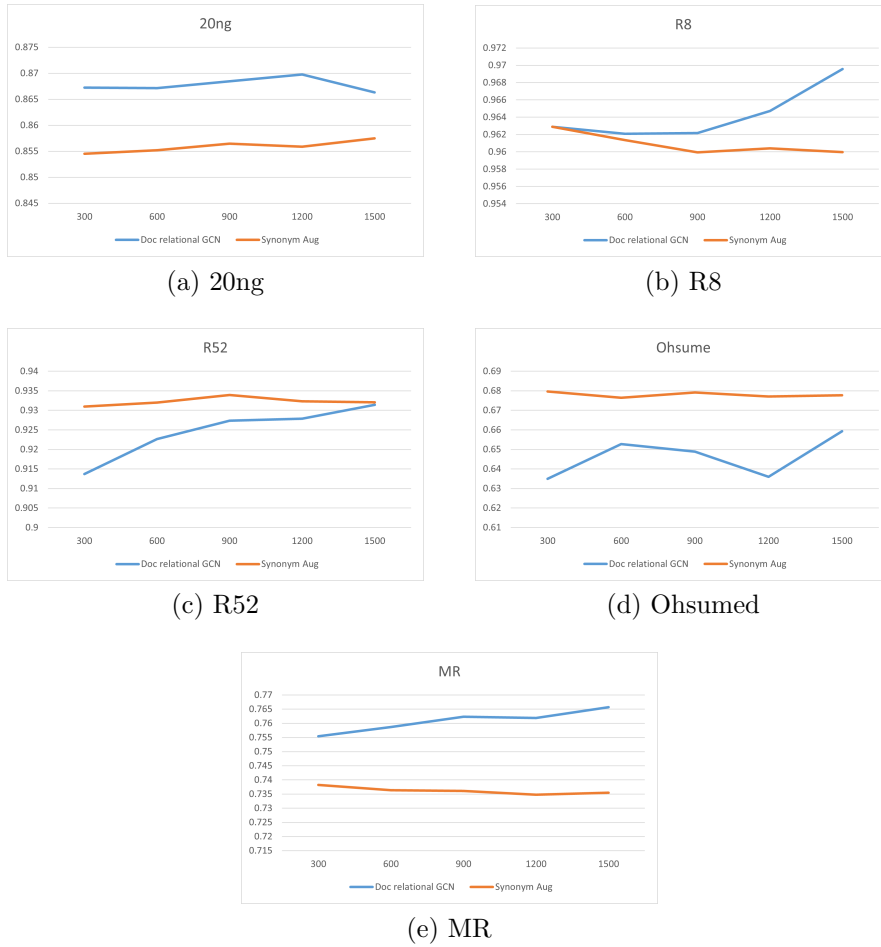


Figure 3.1: Relations Between the Number of Hidden Layers and Test Accuracy

Table 3.2: Comparison of test accuracy among document-relational GCNs, text GCNs and synonym augmented text GCNs

Dataset	Doc-Relational GCNs	Text GCNs [134]	Synonym aug GCNs
20ng	86.8%	85.81%	85.7%
R8	96.96%	96.8%	96.3%
R52	93.19%	92.47%	93.2%
Ohsumed	65.9%	66.99%	68.0%
MR	76.57%	76.18%	73.8%

the context of text classification.

The foundational premise of our comparative study lies in exploring the multi-

faceted nature of graph-based models in text classification, specifically through the lens of document-document relationships (Doc-Relational GCNs), global word-document relations (Text GCNs), and synonym-enhanced semantic networks (Synonym-Augmented GCNs). Each technique addresses distinct aspects of the text classification challenge.

Doc-Relational GCNs are motivated by the potential of leveraging under explored document-document relationships, aiming to enrich text classification models with deeper contextual insights and thematic connections between documents. This approach seeks to overcome the limitations of models that focus solely on word-document interactions, providing a more holistic understanding of the corpus.

Text GCNs, capitalize on the integration of global word co-occurrence and word-document relationships. Their strength lies in harnessing both local and global textual contexts, offering a comprehensive semantic representation that enhances classification accuracy.

Synonym-Augmented GCNs extend the graph model by incorporating synonyms, aiming to tackle the challenges of vocabulary diversity and polysemy. This method enriches the semantic representation within the graph, potentially leading to more nuanced understanding and classification of texts.

3.4.5 Experiment Results

The test accuracy comparison for the document-relational GCNs and the original model without document-document relation edges is presented in Table 3.2. Our new model is even more accurate than the original GCNs model, which performed impressively in the previous experiment [134]. In four out of five datasets, the document-relational GCNs model improves 0.2%-1% accuracy to text GCNs. For the 20ng dataset, the model exhibits a noticeable performance improvement (1%) in terms of test accuracy compared to other datasets. Because of the new document-document relation in the graph, it performs better for the 20ng dataset

than the others. The synonym augmented GCNs performs better in the R52 and Ohsumed datasets. TF-IDF stores important words that can represent document topics and contains information about document-document relations. It also increases the weight of topic words in documents on the same subject. Because the same news category belongs to the same topic, 20ng is both a news and topic classification dataset. In a hyperparameter experiment with hidden layers, overall, there is a tendency towards increasing accuracy with an increasing number of hidden layers in the document-relational GCNs model. The most accurate results are obtained with the number of hidden layers ranging from 1200 to 1500 in the four datasets. We hardly obtain good results in both test accuracy and expected hidden layers for the Ohsumed dataset because using words link to stand for document is not sensitive to medical abstract documents. Overall, the cumulative TF-IDF document-document relation in the GCNs model can achieve much higher test accuracy in most datasets related to topic document classification, and a specific figure of the hidden layer hyperparameter can be used for model parameter tuning.

3.4.6 In-depth Analysis

The proposal to integrate document-document relationships into GCNs stems from an analytical observation: texts within the same or related categories often share thematic or semantic similarities that are not fully captured through direct word-level interactions. By constructing a cumulative TF-IDF weighted graph that highlights these document-document relationships, our approach seeks to introduce a new dimension of relational understanding. This methodological innovation allows the model to perceive the dataset as an interconnected web of documents, where the strength of each connection enriches the model's feature set, leading to more nuanced classifications.

3.5 Conclusion

We have introduced a new document-relational GCNs in this study, which provides a cumulative TF-IDF document-document relation for text classification. By testing its performance on five benchmark datasets, it is discovered that with document-document relation edges in the adjacent matrix, the document-relational GCNs model could improve the overall accuracy comparing to the original text GCNs and the synonym augmented GCNs. Synonym augmented GCNs had higher accuracy in the medical dataset. In addition, different numbers of hidden layers are experimented to determine the optimal parameters of the model. We could observe that excessive hidden layers did not help the original text GCNs and data-augmented GCNs had higher accuracy. However, in the document-relational GCNs model, a large number of hidden layers had a better ability to store text and graph features, which helped text classification. Our future research plan is to determine the best percentage of document-document edges to use when filtering useful or important document relation features. We will also use a more sophisticated algorithm to calculate document-document relations, as well as to filter document-word or word-word relations to simplify the graph.

Chapter 4

A GraphNorm module in Heterogeneous Linguistics Graph Model

Pre-trained language models, such as BERT, have revolutionized the field of NLP, providing powerful tools capable of capturing complex linguistic information in large text datasets. However, these models, though incredibly effective, often require substantial computational resources and struggle with handling complex semantic relationships and noisy or incomplete data. Graph-based approaches, such as HLG models, provide an alternative strategy for tackling these challenges. HLG models utilize graph structures to represent linguistic relationships within and across different linguistic levels, like characters, words, and sentences, which can capture hierarchical and complex semantic information more effectively. However, these models can be sensitive to graph modifications and noisy data points, which could distort the learned representations and lead to performance degradation. We introduce a GraphNorm module, an innovative method aimed at enhancing the performance of HLG model. The primary objective of GraphNorm is to increase model accuracy and minimize memory consumption during the training process. Our approach employs dynamic normalization layers for

each GNN module in the HLG model, facilitating the retention of reliable node information from the graph while mitigating noise. By regulating the influence of outliers and noisy data points, GraphNorm increases the model’s resilience to graph modifications, supporting its ability to handle different scenarios. Our results demonstrate that GraphNorm improves the stability and performance of the HLG model, demonstrating potential for a wide range of applications involving five benchmark datasets.

4.1 Introduction

Deep learning is recognized as one of the most successful types of neural networks, and is referred to as a “deep” method because of numerous hidden layers that give the network great depth. Classic deep learning models have been drawing a great attention and revolutionizing such as CNN [54], RNN [126], and auto encoders [120]. Also, the development of deep learning and big data has been boosting computer hardware development, which, subsequently, has enabled large amounts of training data to be made available and made linguistic feature extraction from Euclidean space data a possibility [129].

The deep learning method has numerous applications on Euclidean space data, such as image, video and speech data. CNN is one of the classic examples that applies the model in Euclidean space data. Primarily, CNN trained images data effectively since images data is invariant of their structure, can be considered as rectangular grid-like and it is continuous in every pixel [12].

However, there are also many applications or databases that store non-Euclidean space data such as graph data and stream data. Significant efforts have been made by researchers to take advantage of neural network deep learning to solve problems of non-Euclidean space data such as social networks [137] and knowledge graphs [34]. A characteristic of non-Euclidean data is that data arrangement is not continuous and relatively random. Specifically for a given data point, it is difficult to define its neighbour nodes. Also, the number of neighbour nodes of

different nodes is different [94]. Thus, it is difficult for CNN to define the same convolution operation on this type of data as on data such as images. Moreover, the arrangement of nodes in each sample data may be different, such as molecular screening [42], which is obviously an application of graph data. Also, while the number of atoms connected in different molecular structures may be different, it is difficult to define the Euclidean distance. Therefore, this type of data can no longer be regarded as a sample point in the Euclidean sample space. Researchers are actively finding ways to learn non-Euclidean data in modern neural networks. GNN can perform the types of operations to learn non-Euclidean data. GNN is an embedding algorithm in the sense that it generates a single node embedding by transferring, transforming, and aggregating feature information of nodes on the entire graph. Inspired by CNN, Graph Convolutional Networks have been developing rapidly. An image can be seen as a special case of a graph, where pixels are connected by neighbouring pixels. A feature which is reminiscent of 2D convolution is that graph convolution can be performed by weighted averaging of node neighbourhood information [129]. However, the GCNs' training is full-batch, which is difficult to expand to a large-scale network, and the convergence is relatively slow. Thus, cluster-GCNs [19] have been developed which are capable of reducing memory requirements and computation time. To solve the parsing error problem, a graph ensemble method has been introduced [44]. Before sending data embeddings into the graph neural network, multiple syntax analysers are utilized to analyze the input, so that multiple graphs can be generated, and multiple graphs can be combined to obtain a graph structure with a smaller graph diameter. The graph is processed so that the model no longer depends on a specific syntactic classifier which increases the robustness of the model.

In recent years, pre-trained language model methods represented by BERT [27] have been widely used in various NLP tasks. A typical pre-trained language model application method can be attributed to a two-stage model comprising pre-training and fine-tuning. First unsupervised and self-supervised pre-training

on a large-scale unlabelled corpus is carried out, and then it migrates to specific downstream tasks through supervised training. For Chinese natural language processing, researchers have proposed various pre-trained language models for Chinese language characteristics, such as ERNIE [111] and Glyce [85], which make use of Chinese word segmentation and Chinese characters to improve the effect of pre-training tasks. Li and colleagues [70] proposed a Multi-source Word-Aligned (MWA) model based on the motivation of integrating Chinese word segmentation into the pre-training language model, trying to integrate vocabulary-level features into the native pre-training language model. Unlike other focus and pre-training work, MWA carried out the integration of external information in the stage of fine-tuning. MWA uses more tokenizers to get better results, however, word segmentation noise increases as more tokenizers are introduced.

Based on a MWA model motivation, an enhanced module and adapter have been introduced for HLG [69]. HLG shows a certain deionising effect when in use with multiple tokenizers to achieve greater accuracy and a faster training process. However, due to the fact that HLG is a graph ensemble method which generates multiple graph operations in the model based on different syntactic classifiers [2], an identical method of normalization in the model is insufficient for the learning process of that large model. As such, a GraphNorm method is proposed to integrate in the HLG model for an improved learning effect. Moreover, some chopping and simplified methods are used to optimize the structure of the model. The presentation of this chapter is as follows. It begins with an introduction to GraphNorm, an innovative normalization method designed to improve the performance of HLG models. Application of this method is aimed at enhancing model accuracy during the training process. Particularly, GraphNorm adds dynamic normalization layers to each GNN module in the HLG model. This approach retains reliable node information from the graph and mitigates noise, increasing the model’s resilience to graph modifications.

The research in this chapter shows that GraphNorm has the potential to general-

ize well across a variety of datasets. It achieves superior accuracy in benchmark datasets, and this suggests that GraphNorm could be beneficial for a wide range of NLP applications.

4.2 Related Work

This section will introduce HLG model and GraphNorm module.

4.2.1 HLG model

HLG [69] is a Chinese linguistics structure heterogeneous graph model which is inspired by MWA and multi-graph ensemble method. MWA assigns each character in the same word a unified attention weight after aggregation through the mixed pooling [135] strategy; on the other hand, in order to reduce the errors caused by different tokenizers, multiple tokenizers are used, and the results of all tokenization strategies are merged.

Developers of MWA propose the use of more tokenizers to achieve better results. However, the addition of more token information without an upper limit cannot continuously bring about performance improvements because such an approach will bring more word segmentation error information, which presents as noise. What HLG can achieve in this context is to make the correct word segmentation structure have greater influence in the model and, as such, let the influence of wrong word segmentation information in the model be ignored as much as possible.

In the HLG model, a Multi-Step Information Propagation (MSIP) is applied to learn linguistic knowledge. Also, MSIP can integrate Chinese word segmentation information to the pre-trained model. According to the developer of HLG, there are three major operations in the MSIP structure which are summarization, concretization and skip connection. Figure 4.1 shows the information-learning operations within the MSIP structure.

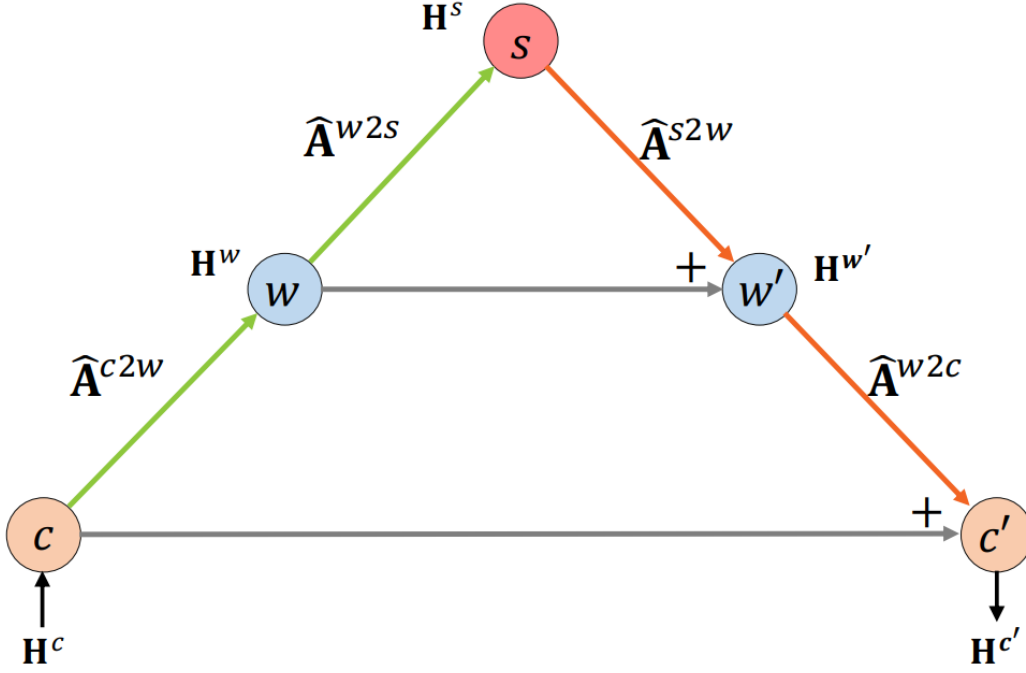


Figure 4.1: MSIP Learning Operations [69]

Coloured circles represent character, word or sentence representations. Green, orange and gray lines describe summarization, concretization and skip connection respectively.

The summarization operation is to gain words and sentences representations. There are two steps, one is from character level to word level, and the other is from word level to sentence level. Given the graph G , the representative character level input matrix is H^c . The summarisation operation equation is as follow:

$$\begin{aligned} H^w &= r(\hat{A}^{c2w} H^c W^{c2w}) \\ H^s &= r(\hat{A}^{w2s} H^w W^{w2s}) \end{aligned} \quad (4.1)$$

where H^w and H^s are input or hidden layers in the information learning operations, W^{c2w} and W^{w2s} are weight parameters that need to be learned by the model.

Concretization is a reverse operation of summarization which can obtain de-

tailed information from the high level information representations, such as partition information from sentences to words and finally to characters. In order to operate reverse operation. \hat{A}^{s2w} and \hat{A}^{w2c} can be obtained by following equations:

$$\begin{aligned}\hat{A}^{s2w} &= Norm((\overline{A}^{w2s})^T) \\ \hat{A}^{w2c} &= Norm((\overline{A}^{c2w})^T)\end{aligned}\tag{4.2}$$

where \overline{A}^{w2s} and \overline{A}^{c2w} are predefined relations matrices for word to sentence and character to word respectively. Then concretization equations are as below:

$$\begin{aligned}\overline{H}^{w'} &= r(\hat{A}^{s2w} H^s W^{s2w}) \\ \overline{H}^{c'} &= r(\hat{A}^{w2c} H^{w'} W^{w2c})\end{aligned}\tag{4.3}$$

where $H^{w'}$ and $H^{c'}$ are hidden layers in the information learning operations, W^{s2w} and W^{w2c} are weight parameters that need to be learned by the model.

Skip connection is a supplementary method to solve problems of concretization because, typically, it is difficult to obtain low level representations from a high level, such as getting representation from sentences level to word level. Thus, skip connection operations are added within summarization and concretization to obtain low level representations. The equations are as follow:

$$\begin{aligned}H^{w'} &= \overline{H}^{w'} + r(H^w W^w) \\ H^{c'} &= \overline{H}^{c'} + r(H^c W^c)\end{aligned}\tag{4.4}$$

where W^w and W^c are weight parameters matrices. r is an activation function like *ReLU*. $H^{c'}$ is the output matrix of MSIP which is the learned knowledge of the graph.

4.2.2 Graph Norm

Generic GNN structure with normalization layers can be represented as follow:

$$H^n = r(\text{Norm}(W^n H^{n-1} \tilde{A})) \quad (4.5)$$

where r is an activation function. W^n is current layer weight matrix. H^{n-1} is last layer graph information. \tilde{A} is adjacent matrix after normalization treatment. There are multiple normalization method that can be utilized in GNN such as LayerNorm [4], BatchNorm [47] and InstanceNorm [117].

LayerNorm has a robust effect on Recurrent Neural Networks. However, BatchNorm is effective at CNN and GCNs. Generally, before training, it is necessary to normalize the data to make the distribution consistent, but in the training process of a deep neural network, it is usually trained with each batch sent to the network, so that each batch has a different distribution. In addition, in order to solve the internal covariate shift problem inherent to this, the definition of this problem was proposed along with the batch normalization. During the training process, the data distribution will change, which will bring difficulties to the learning of the next layer of network. Therefore, applying batch normalization forcibly pulls the data back to the normal distribution with a mean value of 0 and a variance of 1, so that the data distribution is not only consistent, but the gradient also disappears. In addition, an internal covariate shift and covariate shift are two separate issues. The former pertains to a shift inside the network, and the latter relates to a shift for input data.

BatchNorm focuses on normalizing each batch to ensure that the data distribution is consistent, because the results in the discriminant model depend on the overall distribution of the data. However, in text features stylization, the generated result mainly depends on a certain text instance, therefore normalizing the whole batch is not suitable for text. InstanceNorm can speed up model convergence and maintain independence between each text features instance which

focuses on height and width of matrices. The equation is as follow:

$$\begin{aligned}
 y_{tijk} &= \frac{x_{tijk} - \mu_{ti}}{\sqrt{\sigma_{ti}^2 + \varepsilon}} \\
 \mu_{ti} &= \frac{1}{HW} \sum_{l=1}^W \sum_{m=1}^H x_{tilm} \\
 \sigma_{ti}^2 &= \frac{1}{HW} \sum_{l=1}^W \sum_{m=1}^H (x_{tilm} - \mu_{ti})^2
 \end{aligned} \tag{4.6}$$

where μ is the range of mean, σ is standard deviation.

Although InstanceNorm has a beneficial effect on GNN, to simply normalize the values of each feature dimension in the graph does not consistently lead to improvements. For regular graphs, inducing the standard shift by subtracting the mean may cause loss of information on the graph structure. In graphical data, statistics after mean aggregation can sometimes contain structure information. Eliminating the mean can reduce the expressiveness of the neural network. The issue may not happen in the image domain. The average statistics of image data contains global information such as brightness. Removing such information from the image will not change the semantics of the objects therein, and thus do not hurt classification performance. Guided by this insight, the author modified the current normalization method with a learnable parameter to automatically control how much the mean to preserve in the shift operation which combined with the graph-wise normalization. The GraphNorm equation is as follow:

$$y_{ij} = \gamma_j \cdot \frac{x_{ij} - \alpha_j \mu_j}{\sqrt{\sigma_j^2 + \varepsilon}} + \beta_j \tag{4.7}$$

where α is learnable weight matrix to control mean in normalization operation.

4.3 Proposed Method

In this section a GraphNorm method will be proposed to facilitate HLG model to obtain an improved outcome.

The GraphNorm technique is introduced as an innovative normalization method aimed at enhancing the performance of HLG models, especially in the context of NLP tasks. This technique addresses the challenges associated with graph-based learning, particularly when dealing with noisy, incomplete, or complex semantic relationships within data. The motivation behind GraphNorm and a detailed analysis of its workings are discussed below, with references to relevant literature to provide a comprehensive understanding of its significance and application. GraphNorm can improve the stability and robustness of GNNs to alleviate noise and disturbance in the training process. Also, GraphNorm reduces the impact of outliers or noisy data points by preventing them from dominating graph representations. This makes the GNN more resilient to graph modifications and enhances its ability to handle real-world scenarios with imperfect or incomplete data. To retain dependable node information from the graph and eliminate noise from HLG in dynamic level, dynamic normalization layers are introduced for each GNN module in the HLG model. Figure 4.2 shows the structure of a typical HLG model with dynamic normalizations.

4.3.1 Motivation

The proliferation of graph-based models, notably GNN, has opened new avenues in handling data that inherently forms non-Euclidean structures, such as social networks, molecular structures, and linguistic relationships [129]. However, these models often grapple with the erratic nature of graph data, which may include noisy information or irregular graph structures. This variability can significantly impact the learning process, leading to unstable training and suboptimal model performance.

In the realm of NLP, where the objective is to capture and utilize complex linguistic relationships, these challenges are accentuated. Traditional pre-trained language models like BERT have shown remarkable success in capturing linguistic information in text datasets. Yet, they are computationally intensive and may not effectively handle the nuanced semantic relationships and the non-uniformity present in natural languages [27].

The introduction of HLG models represents a strategic move to leverage graph structures for encapsulating linguistic relationships across different levels (characters, words, sentences) more effectively. However, the sensitivity of these models to graph modifications and the presence of noisy data points necessitate a robust normalization technique that can stabilize the learning process and enhance model performance [125].

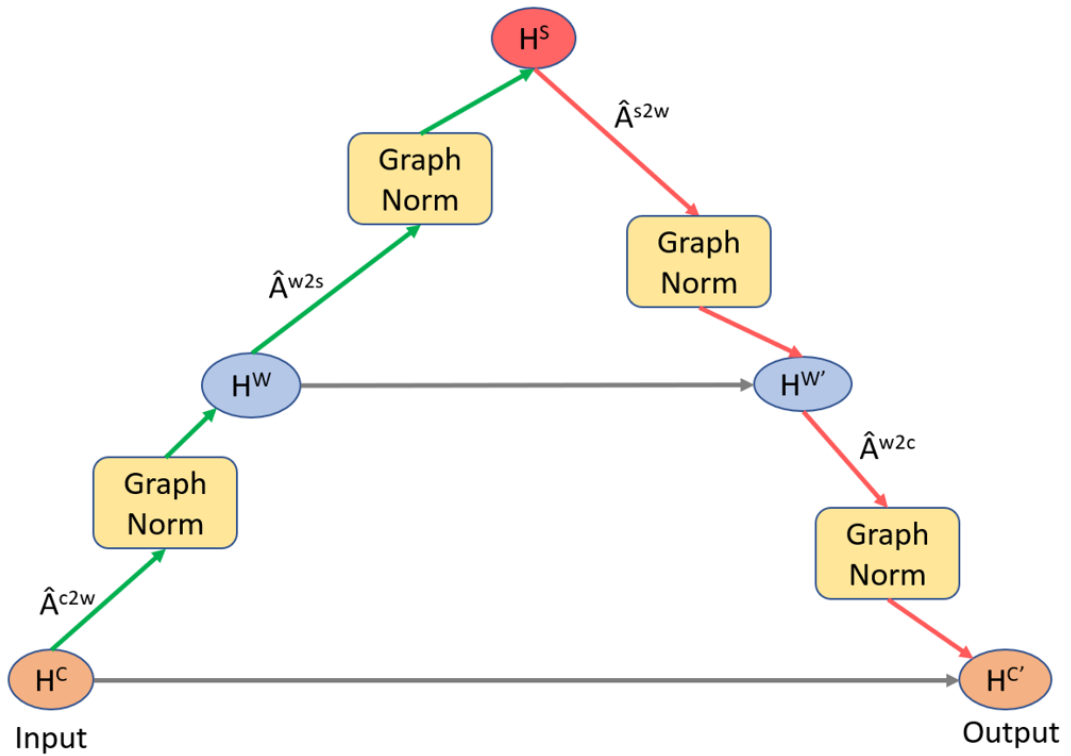


Figure 4.2: Dynamic Graph Norms for HLG model

The GraphNorm method is added after every GNN module to increase model robustness. Firstly, the input nodes feature H^c which will go through a GNN

module. Then the output hidden layer will be normalized by the GraphNorm module. Thus, output is the word representations H^w . The equation is as follow:

$$\begin{aligned}
H^{c2w} &= \hat{A}^{c2w} H^c W^{c2w} \\
H^{c2w_norm} &= \gamma \cdot \frac{H^{c2w} - \alpha \cdot \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \\
H^w &= r(H^{c2w_norm})
\end{aligned} \tag{4.8}$$

where \hat{A}^{c2w} is the adjacent matrix after normalization. H^c is the character representation. W^{c2w} is the learnable weight matrix in this module. μ is the mean of current scope. σ is standard deviation. γ is overall coefficient weights of the normalization. β is bias weights of the normalization. α is mean scale weights in the normalization. α , β and γ are learnable parameters in the normalization module. They can control how much mean and overall information they should keep in the normalization. r is an activation function *ReLU*.

Then, similarly, the output word representation matrix H^w will go through the next summarisation module as follow:

$$\begin{aligned}
H^{w2s} &= \hat{A}^{w2s} H^w W^{w2s} \\
H^{w2s_norm} &= \gamma \cdot \frac{H^{w2s} - \alpha \cdot \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \\
H^s &= r(H^{w2s_norm})
\end{aligned} \tag{4.9}$$

where H^w is the word representation matrix in last module. H^s is the sentence representation matrix which is output in this module. This is a successive module after character to word module. Thus, the normalization module is same as the last.

After obtaining these sentence representations, the model will then have a reverse operation from sentence representations to character representations. The first reverse operation is from sentence representations to word representations. The

equations are as follow:

$$\begin{aligned}
\overline{H}^w &= \hat{A}^{s2w} H^s W^{s2w} \\
\overline{H}^{s2w.norm} &= \gamma \cdot \frac{\overline{H}^w - \alpha_j \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \\
\overline{H}^{w'} &= r(\overline{H}^{s2w.norm}) \\
H^{w'} &= \overline{H}^{w'} + r(H^w W^w)
\end{aligned} \tag{4.10}$$

This is a reverse operation module. Word representations $\overline{H}^{w'}$ are obtained from the sentence representations H^s . Also, the skip connection operation is added with the word representation in the previous front operation result. Thus, a final word representation $H^{w'}$ is obtained after the skip connection operation. Then the same operation to obtain character representations is as follow:

$$\begin{aligned}
\overline{H}^c &= \hat{A}^{w2c} H^{w'} W^{w2c} \\
\overline{H}^{w2c.norm} &= \gamma \cdot \frac{\overline{H}^c - \alpha_j \mu}{\sqrt{\sigma^2 + \varepsilon}} + \beta \\
\overline{H}^{c'} &= r(\overline{H}^{w2c.norm}) \\
H^{c'} &= \overline{H}^{c'} + r(H^c W^c)
\end{aligned} \tag{4.11}$$

The final step module is to operate from word representations $H^{w'}$ to character representations $H^{c'}$. The details of how to do GraphNorm in HLG model are shown in Pseudo Code 2.

4.4 Experiments

Our proposed GraphNorm method through HLG has several advantages compared to existing HLG models. GraphNorm leverages the graph structure of the HLG to capture and encode intricate linguistic relationships within Chinese text. By considering the dependencies and connections between linguistic units, such as characters, words and sentences. GraphNorm enhances the ability of the model

Pseudo Code 2: Implement GraphNorm in HLG in a batch of data

```
graphs ← a batch graphs;
tensors ← the batch tensors;
feats ← empty;
nodeslist ← empty;
for graph in graphs do
    nodes ← number of nodes in graph;
    feat ← select first number of lines of tensors;
    nodeslist.add(nodes);
    feats.add(feat);
end for
featDimention ← number of a feat vector;
nodesNumber ← sum of nodesList;
batchSize ← length of batchList;
indexList ← nodesNumber by featDimention empty matrix;
meanList ← empty;
for nodes in nodesList do
    featureVector ← select feats based on nodes index in indexList;
    sumFeatureVector ← sum of featureVector;
    meanFeatureVector ←
        sumFeatureVector / number of featureVector;
    meanList.add(meanFeatureVector);
end for
subtract ← feats - meanList * meanWeight;
stdList ← empty;
for nodes in nodesList do
    featureVector ← select feats based on nodes index in indexList;
    sumFeatureVector ← sum of featureVector;
    stdFeatureVector ←  $\text{sqrt}(\text{subtract}^2 + 1 * 10^{-6})$ ;
    stdList.add(stdFeatureVector);
end for
normedFeature ←  $\text{subtract} / (\text{std} * \text{stdWeight}) + \text{biasWeight}$ ;
```

to understand the contextual and semantic information presenting in the text. Chinese text often contains noise, ambiguity, and inherent linguistic challenges [17]. GraphNorm helps pretrained models handle these issues by leveraging the graph structure. By considering multiple paths and alternative routes within the graph, GraphNorm enables the models to navigate through ambiguous contexts and make more robust predictions, mitigating the impact of noise and ambiguity in Chinese text.

4.4.1 Datasets

A detailed justification for each chosen dataset (ChnSenti, Weibo, THUCNews, LCQMC, XNLI) was provided, emphasizing their diversity, relevance, and the unique challenges they present for text classification. This diversity ensures a comprehensive evaluation of GraphNorm’s effectiveness across different contexts and linguistic challenges.

- **ChnSenti** is a comprehensive and widely-used resource for sentiment analysis in Chinese hotel review text. There are more than 11,546 hotel reviews in the data including 5,786 positive reviews and more than 5,760 negative reviews. The dataset is separated to a 9146 train set, and 1200 validation and test sets. Sentiment, opinion and comment tendency analysis can be carried out using this review dataset.
- **Weibo** dataset is a comprehensive collection of social media posts from the popular Chinese microblogging platform, Weibo. It comprises diverse user-generated posts which has 100,000 train sets, 9,988 evaluation sets and 10,000 test sets. The dataset offers a valuable resource for researchers, data scientists, and social media analysts interested in studying and understand online behaviors, trends, and sentiments within the Chinese social media landscape.
- **THUCNews** [110] dataset consists of a large collection Chinese news ar-

Table 4.1: Datasets Summary

Dataset	ChnSenti	Weibo	THUCNews	LCQMC	XNLI
# Train	9,146	100,000	50000	238,765	392,702
# Valid	1,200	9,988	5,000	8,801	2,490
# Test	1,200	10,000	10,000	12,499	5,010
# Total	11,564	119,988	65,000	260,065	400,202
# Classes	2	2	10	2	3

ticles. The dataset aims to provide a diverse range of text samples that represent different domains and subjects. There are 50,000 train sets, 5,000 evaluation sets and 10,000 test sets. 10 topics are selected in THUCNews dataset which include real estate, technology, finance, games, entertainment, fashion, politics, household, education and sports.

- **LCQMC** [75]. dataset provides a valuable resource for researchers and developers working in the field of NLP, specifically in the domain of question matching and semantic textual similarity in the Chinese language. There are 238,765 train sets, 8,801 evaluation sets and 12,499 test sets.
- **XNLI**. [21] dataset provides a rich and diverse collection of sentence pairs in multiple languages, and offers an excellent benchmark for developing and evaluating cross-lingual NLP models. In the experiments set out in this thesis, only the Chinese components to the XNLI dataset were utilized, which comprised 392,702 train sets, 2,490 evaluation sets and 5,010 test sets.

The datasets summary are as below:

4.4.2 Word Segmentation

The datasets are preprocessed by representation with the Chinese BERT dictionary index for every character. Sentences are segmented to words by using

Chinese word segmentation (CWS) tools, as explained below. CWS tools are invaluable resources for segmenting Chinese text into individual words or lexical units. With their linguistic rules, statistical models, and machine learning algorithms, these tools enable accurate and efficient analysis of Chinese language data. To compare model performance with HLG, three word segmentation tools are utilized, which are THULAC [71], NLPIR [145] and pyhanlp [36] in the data processing of the five datasets.

4.4.3 Environment and Parameters

The model was run in a desktop computer with 16GB random access memory (RAM) and 6G video memory. Python 3.9 is utilized for the programming language. Pytorch 2.0 is used as a framework that provides a flexible and efficient platform for building and training deep learning models.

During the training process, it is essential to fine-tune parameters in the model. Indeed, several parameters, such as learning rate and batch size, need to be fine tuned to optimize the model’s performance. Here learning rate of the model is fine-tuned to be $3.5 * 10^{-5}$ to enable the GraphNorm module to learn appropriately. Also, setting an appropriate learning rate is essential for finding the optimal weight values that minimize the loss function. However, using a fixed learning rate throughout training may lead to suboptimal results, such as slow convergence, overshooting, or getting trapped in local minima. To ballance BERT weights, model weights and GraphNorm weights, CosineAnnealingLR method has been added to dynamically adjust the learning rate over time.

4.4.4 Justification for Comparative Techniques

The selection of comparative techniques (GraphNorm HLG vs. HLG) was explained by highlighting the importance of benchmarking the proposed GraphNorm enhancement against the baseline HLG model. This comparison aims to demonstrate the incremental improvements offered by GraphNorm in handling

Table 4.2: Comparison test accuracy among GraphNorm HLG and HLG

Dataset	GraphNorm HLG	HLG
ChnSenti	95.42%	95.0%
Weibo	96.97%	96.34%
THUCNews	94.54%	94.45%
LCQMC	88.31%	88.85%
XNLI	67.78%	66.85%

hierarchical linguistic graph structures for NLP tasks.

4.4.5 Experiment Results

The performance of deep learning models is evaluated by its accuracy, defined as a measure of the proportion of correct predictions made by the model compared to the total number of predictions. A higher accuracy indicates that the model is capable of capturing and understanding the underlying patterns in the data effectively. The accuracies in five bench-mark datasets in GraphNorm-HLG model are presented in Table 4.2:

The experiment compares GraphNorm HLG and HLG models on the above five datasets. The results indicate that the GraphNorm HLG model generally outperforms the HLG model in terms of test accuracy, demonstrating its effectiveness in various NLP applications.

Overall, the results suggest that the GraphNorm HLG method generally outperforms the traditional HLG method. In 4 out of the 5 datasets, ChnSenti, Weibo, THUCNews, and XNLI, the GraphNorm-HLG method demonstrates improved accuracy. On the ChnSenti dataset, the GraphNorm-HLG model has an accuracy of 95.42%, which is marginally better than the accuracy of the HLG model, at 95.0%. This indicates that for this specific dataset, the incorporation of GraphNorm techniques into the HLG model provides a marginal improvement. The Weibo dataset results further corroborate the relative effectiveness of the

GraphNorm-HLG model. Here, the accuracy rises to 96.97%, outperforming the accuracy of the HLG model at 96.34%. This underlines the potential robustness of the GraphNorm-HLG model across different datasets, implying a certain level of generalization. On the THUCNews dataset, the accuracy for both models are near-equal, with GraphNorm-HLG scoring a 94.54% accuracy, slightly higher than HLG model, at 94.45%. While the difference may not appear substantial, this result once again is in favour of GraphNorm-HLG over HLG. In the case of the LCQMC dataset, the HLG model performs better, with an accuracy of 88.85% compared to the GraphNorm-HLG's 88.31%. This result is a notable deviation, and suggests that the GraphNorm technique applied to HLG may not always lead to improved performance. There may be dataset-specific factors that cause this deviation. Finally, on the XNLI dataset, the GraphNorm-HLG model 67.78% accuracy is superior to HLG, at 66.85%. Despite the lower overall scores on this dataset, the results suggest that the GraphNorm-HLG model is overall more accurate in 4 out of 5 datasets.

Development set accuracy is a measure of a model's performance that can predict the correct output given a set of inputs. A high development set accuracy means the model is effective at making correct predictions on that particular set of data. Figure 4.3 shows accuracies of development sets in different epochs during training process for the five datasets. Based on the provided data which details the performance of GraphNorm-HLG and HLG models across five different datasets, the GraphNorm-HLG model tends to outperform the HLG model in majority of cases across these datasets and runs. The performance of the HLG and GraphNorm-HLG models can vary across different run epochs and development sets. Yet, while GraphNorm-HLG appears to be superior to HLG in the development accuracy at the end of epochs, this study finds that there are rare but notable situations in which the HLG model has superior accuracy.



Figure 4.3: Development Sets Accuracy in Epochs

4.5 Conclusion

This research chapter has introduced a novel method of layer normalization, utilizing GraphNorm, with the goal to address the improvement of accuracy in HLG models. GraphNorm’s experimental application here has resulted in marked improvements in the performance of the HLG model across a wide range of benchmark datasets, thus affirming this approach to improve its efficiency and robustness.

GraphNorm has been found to preserve graph-level representations and encapsulates hierarchical information, effectively solving the prevalent problems associated with standard layer normalization techniques. Among these issues are

increased memory usage and reduced performance in larger graph structures. Empirical evaluations further exemplify GraphNorm’s superiority over batch normalization and layer normalization concerning memory usage, computational speed, and the quality of the results, advocating for its wider adoption in NLP. A noteworthy point is the consistent performance of GraphNorm during development runs. The model demonstrated robustness and stability, delivering steady performance improvements over iterations. This consistency in development runs underlines the utility of GraphNorm in diverse real-world scenarios, even when dealing with significant variations in graph structures.

Looking ahead, GraphNorm’s versatility has also been demonstrated and seamless integration with various graph networks can be experimented in the future works. Although the primary focus of GraphNorm is currently on HLG models, data in this chapter highlights its potential application in other graph-based models. Thus, the future application of GraphNorm to drive improvements in model performance across a multitude of applications is promising, with potential applications developed to benefit different sectors such as finance, healthcare, technology, and social media. The opportunity to optimize GraphNorm for more efficient computational time and memory usage, to evaluate its performance with larger, more complex datasets, and to explore its impact on different facets of NLP tasks are attractive directions for future research.

Through the findings and potential future directions outlined here, the overarching aim is to push the boundaries of graph-based NLP models, specifically HLG models. It is anticipated that the insights derived from this research will spur on further developments and wider application of GraphNorm and similar normalization techniques in the future.

Chapter 5

Expanding the Capabilities of Document-Level Event Role Filler Extraction with Innovative Position Encoding Methods in Multi-Granularity Contextualized Models

The research in this chapter presents an innovative solution for Document-Level Event Role Filler Extraction, achieved by introducing novel position encoding strategies within a multi-granularity contextualized model. A notable innovation of this work lies in the integration of sinusoidal and relative position encodings at different levels of granularity in the model's architecture. These novel encoding methods add substantial value by capturing the contextual interdependencies and the internal hierarchical structure within text. Through the unique combination of GloVe word embeddings, context-aware BERT embeddings and these advanced position encodings, a richer, more contextualized representation of

words is achieved. Utilizing the strengths of BiLSTM-CRF models, this chapter shows that such an approach effectively encapsulates both sequential and contextual aspects of the text, crucial for tasks like event extraction. The experiments demonstrate superior performance of this method compared to existing benchmark datasets. As such, the work in this chapter demonstrates the potential of incorporating innovative position encoding strategies within a multi-granularity framework.

5.1 Introduction

Event role filler extraction, also referred to as event extraction, is a task in NLP that involves detecting instances of pre-defined types of events in a given text and identifying the entities or arguments that are associated with these events in particular roles [113]. For instance, given an event type of attack, potential roles might include attacker, target, time and location. Traditional approaches have focused on sentence-level event extraction, however, with the ever-increasing complexity and volume of real-world data, there is currently a significant need for document-level event role filler extraction to effectively manage and process this growing body of data. Moreover, document-level extraction provides greater extensive context and is helpful to better understand complex event dynamics [131] and is an approach that can understand and interpret events as interconnected entities distributed in text, rather than isolated events within a single sentence. Despite these advantages in its use, there are challenges and problems in the development of event role filler extraction. Notably, natural language is inherently ambiguous which means that many words have multiple meanings based on their context. This ambiguity makes it challenging to determine the correct interpretation of a sentence or document [83]. Recognizing entities within the document is a critical first step in event role filler extraction, and any errors at this stage can propagate through the process. Some named entities might not be properly recognized due to the diversity and ambiguity of natural language [92]. Another

challenge of event role filler extraction is dependency parsing and semantic role labeling. It is essential to correctly identify the semantic roles of different entities in the text which can be complicated by language ambiguity, complex sentence structures, and long-range dependencies [140].

To solve such challenges of event role filler extraction, researchers have been striving to develop sophisticated deep learning models. RNN [61] and transformers [119], have demonstrated their superior performance in named entity recognition tasks. They capture contextual information effectively which leads to more accurate entity recognition. Moreover, transformer-based models, such as BERT [27] and GPT [98], have shown great promise in handling language ambiguity. These models can capture context effectively and provide meaningful representations to better understand the semantic roles of the words in the text.

To handle the event role filler extraction effectively, a combined BiLSTM-CRF model has been proposed [46] [108]. The BiLSTM can learn context-rich representations of the words in a document that incorporate information from both preceding and succeeding words. The CRF can then use these representations to predict the most likely sequence of labels that identify the event role fillers. Moreover, the CRF layer in the BiLSTM-CRF model takes the whole output sequence into account when making predictions, as opposed to making independent decisions for each token. This helps the model to produce coherent labels for the sequence, considering the dependencies between adjacent labels. This is critical in Event Role Filler Extraction, where the labels of the words often depend on each other. Therefore, the BiLSTM-CRF model for document-level event role filler extraction demonstrates the effectiveness of hierarchical BiLSTM-CRF models in handling the challenges of document-level event role filler extraction.

Based on the BiLSTM-CRF model, to capture dependencies on all levels of text, a multi-granularity contextual encoding method [30] has been developed. This approach captures context at different granularities such as word-level, sentence-level, paragraph-level and document-level, which provides a comprehensive un-

derstanding of the events. Though the multi-granularity contextual encoding tries to handle long-Range dependencies by considering the entire document context, the deficiency of positional information in the model might not handle long-range dependencies that occur when role fillers and events are mentioned far apart in the text.

Position encoding involves adding additional information to the input embeddings that the model can use to distinguish the position of each word in the sequence. In the design of BiLSTM-CRF models, positional information is implicitly captured through the sequence order in which inputs are fed into the model. However, adding explicit positional encoding could provide additional benefits, especially in complex tasks such as event role filler extraction. In the original Transformer model [119], this was achieved through absolute position encodings generated by sinusoidal functions. These functions output unique values for each position, providing a distinctive signature that the model can use to identify word order which can be beneficial for handling long-range dependencies in text. Moreover, a more sophisticated relative positioning method has been introduced to enhance the model’s ability to handle long-range dependencies. This concept is particularly useful for tasks that need to consider the directionality or order of the sequence, such as text generation, machine translation, and event role filler extraction. Relative position encoding can enhance the model’s ability to understand the semantics of a sequence based on the positional relationships among the tokens.

Positional embeddings help models understand the order of tokens within a sequence, and understand the relationships between different parts of a sentence or document, which can be important in tasks that involve understanding complex event [67]. Also, positional embeddings can improve the model’s ability to handle long sequences [141]. This is particularly relevant in document-level tasks, where the model needs to process long documents and understand the relationships between entities and events that might be located far away from each other.

To enhance better event role filler extraction effectiveness, this thesis chapter proposes to add sinusoidal positioning and relative positioning methods to the multi-granularity contextual encoding. The contributions are as follows:

Firstly, two positional encoding strategies are applied, namely sinusoidal and relative positional encodings, for the first time in the domain of Document-Level Event Role Filler Extraction. This innovative application of encoding methods enhances the understanding of the contextual interdependencies and the hierarchical structure in a given text.

Secondly, a combination of GloVe word embeddings, BERT embeddings and the new positional encodings is introduced, which results in a richer, more contextualized representation of words. This enhanced representation is particularly crucial for tasks like event extraction.

Thirdly, the proposed model in this chapter, which harnesses the strengths of BiLSTM-CRF models and new positional encoding strategies, is found to display superior performance on event role filler extraction tasks. This superiority is demonstrated through experimental results using the MUC-4 dataset, where the model outperforms existing benchmarks.

Furthermore, the relative position encoding approach adopted in this work shows high performance across multiple event role categories. This indicates the broad applicability and generalization of the model to various tasks and scenarios.

Our analysis of the results provides valuable insights into which position encoding strategy works best for specific tasks and event role categories. These insights offer guidelines for future task-specific model adaptation and offer new avenues for further research.

5.2 Related Work

In the following, previous work on three major model parts, namely sequence tagging [81], BiLSTM-CRF models [46] and multi-granularity reader [30] for Event

Role Filler Extraction, is briefly introduced.

5.2.1 Sequence Tagging

Sequence Tagging can be defined as an NLP technique through which computers or models can understand text as a sequence of word-tag information. A text comprises words, and each word has a specific meaning to be understood by computers. Thus, each word in the sentence or sequence is tagged by a specific meaning label. This 'word-tag' information method represents input that enables the model to understand text in a sophisticated way. Normally, the tag definition is a dataset-specific. To solve this problem of document-level event role extraction, one important task is to transform a document into paired token-tag sequences [118]. A common token-tag format is the BIO (Beginning, Inside, Outside) format, and examples of paired token-tag sequences are shown in Fig. 5.1. The *B* tag indicates that a word is the beginning of a words group or chunk. When the words group has more than one word, the next word must be an *I* tag. There are multiple *I* tags to label until the words group ends.

...	the	guatemala	army	denied	today	that	guerrillas	attacked	the	...
...	O	O	O	O	O	O	B-perp_individual_id	O	O	...
...	"	presidential	farm	,	located	on	the	pacific	side	...
...	O	O	B-phys_tgt_id	O	O	O	O	O	O	...
...	ce	##re	##zo	has	been	staying	since	2	february	...
...	B-	hum_tgt_name	I-hum_tgt_name	I-hum_tgt_name	O	O	O	O	O	...
...	mouth	##piece	of	the	guatemala	##n	national	revolutionary	unity	...
...	O	O	O	O	B-perp_organization_id	I-perp_organization_id	I-perp_organization_id	I-perp_organization_id	I-perp_organization_id	...
...	a	guerrilla	column	attacked	the	farm	2	days	ago	...
...	O	B-perp_individual_id	I-perp_individual_id	O	O	B-phys_tgt_id	O	O	O	...
...	against	the	farm	,	and	that	president	ce	##re	...
...	O	O	B-phys_tgt_id	O	O	O	O	B-hum_tgt_name	I-hum_tgt_name	...

Figure 5.1: BIO Token-tag Sequences

5.2.2 BiLSTM-CRF models

A set of sophisticated BiLSTM-CRF models is utilized to extract entities or tags within the sentence level [46]. BiLSTM recognises word tags of sentences which can find relations between a word and a tag [123]. While CRF can be used to

find relations among tags in a sentence [116]. Thus, the combination of BiLSTM and CRF models are proven high accuracy of NER.

5.2.2.1 BiLSTM

To memorise long term feature of datasets, RNN has been utilized to predict current output [88]. However, the effect of RNN on the semantic information capture of long sentences is very limited. RNN often confronts computational issues, such as gradient vanishing and exploding, which cannot run modern models effectively [102]. A more efficient LSTM model [40] are utilized to solve the issues of word-sequence tagging.

LSTM can learn the semantic information of a sentence, but it assumes that the semantic information of each word is only related to the semantic information of the previous word in the sentence, and has nothing to do with the meaning of the subsequent word [45]. This assumption may not be satisfied in the actual scenario, so BiLSTM is introduced to solve the problem. The model takes word embedding as input and learns the semantic information of the sentence from the forward direction and the reverse backward direction respectively. The results are spliced using concatenation operation, such that the semantic information learned by BiLSTM satisfies that the word depends on both the meaning of the word before the sentence and the meaning of the word after the sentence. The BiLSTM model graph is shown on Fig. 5.2

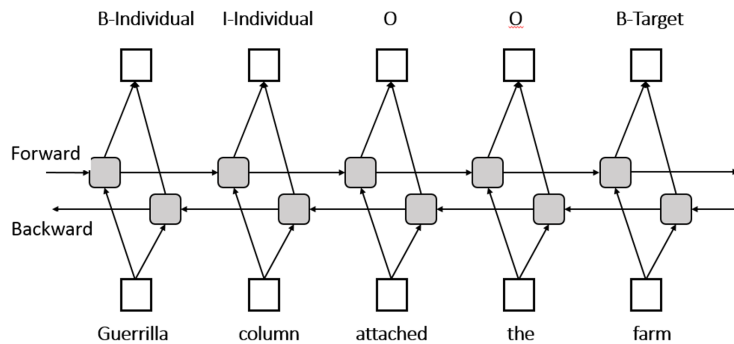


Figure 5.2: BiLSTM Model

5.2.2.2 BiLSTM-CRF

CRF is conditional random field, which is a discriminative undirected graphical model. Given observation sequence. The CRF network is connected after the LSTM. The LSTM can effectively use the semantic features of the input words, and the CRF layer can effectively use the sentence-level annotation information. At the same time, the state transition matrix of the CRF layer is added to the parameters to be trained, so that the model can effectively use the constituent label information of the preceding and following words to predict the label information of the current word. Although the effect of LSTM itself has been sufficient to meet certain requirements, using a CRF network can predict tag-tag relations in a sentence accurately. For instance, using BIO standard labeling, the prediction of LSTM may still predict *I*, the intermediate state, to the beginning of the sentence, which actually does not exist in the situation. This problem can be solved by sentence-level CRF prediction. The BiLSTM-CRF model is shown on Fig. 5.3

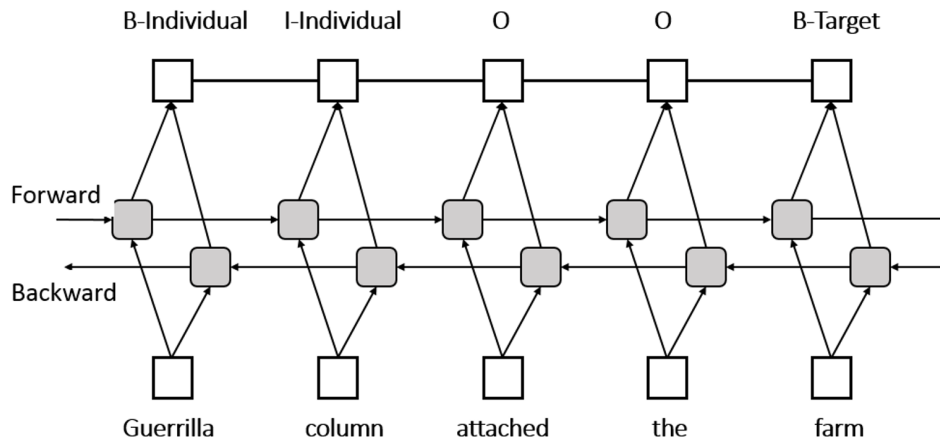


Figure 5.3: BiLSTM-CRF Model

There are three layers for BiLSTM-crf models. The models three layers are shown on Fig. 5.4

Embedding Layer At the embedding layer, each token x_i is in the input se-

quence as a concatenation of its word embedding and contextual token representation. Also, 100-dimensional GloVe pre-trained word vectors are used for word embedding, using BERT-base to generate contextual representations and then connect the two. Equation is in (5.1)

$$\begin{aligned}
 xe_i &= E(x_i) \\
 xb_1, xb_2, \dots, xb_m &= BERT(x_1, x_2, \dots, x_m) \\
 x_i &= \text{concat}(xe_i, xb_m)
 \end{aligned} \tag{5.1}$$

BiLSTM Layer A three-layer BiLSTM encoder is used in equation (5.1) to enable models capture features in sequence tags.

$$p_1, p_2, \dots, p_m = BiLSTM(x_1, x_2, \dots, x_m) \tag{5.2}$$

CRF Layer Passing p_i matrix through a linear layer, then getting P of size $m \times$ label space size, where $P_{i,j}$ is the score of the i th label j tokens in the sequence. For label sequence $y = y_1, \dots, y_m$, the score for sequence label pairs is in equation (5.3):

$$\text{score}(X, y) = \sum_{i=0}^m A_{y_i, y_{i+1}} + \sum_{i=0}^m P_{i, y_i} \tag{5.3}$$

5.2.3 Multi-Granularity Reader

A multi-granularity reader leverages the different granularities of text information for improved understanding and prediction in models. The typical approach is to create multiple readers or encoders focusing on different levels of granularity in the text which including word-level, sentence-level, paragraph-level and document level readers. The structure of multi-granularity reader is on Fig. 5.5.

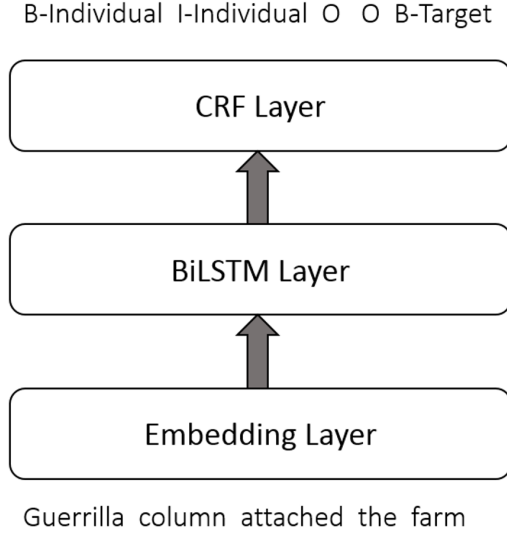


Figure 5.4: Three Layers of BiLSTM-CRF Models

5.2.3.1 Embedding Layer

As the above BiLSTM-CRF model mentioned, the embedding layer combines GloVe and contextualized embeddings from BERT. GloVe embeddings are based on word co-occurrence statistics and thus capture general semantic and syntactic properties of words, yet are context-insensitive. On the other hand, BERT embeddings are context-sensitive and are capable of capturing the nuances of word usage based on the specific context in a sentence. The sentence-level and paragraph-level word embedding representations are as below:

$$\begin{aligned}
 sentence_i &= \tilde{x}_1^1, \tilde{x}_2^1, \dots, \tilde{x}_{l_1}^1 \\
 paragraph_i &= \hat{x}_1^1, \hat{x}_2^1, \dots, \hat{x}_{l_k}^k
 \end{aligned}
 \tag{5.4}$$

5.2.3.2 BiLSTM Layer

Sentence-level BiLSTM processes each sentence of a document as a separate sequence. This method can capture the contextual information within each sentence, such as the dependencies between words and the overall semantic meaning

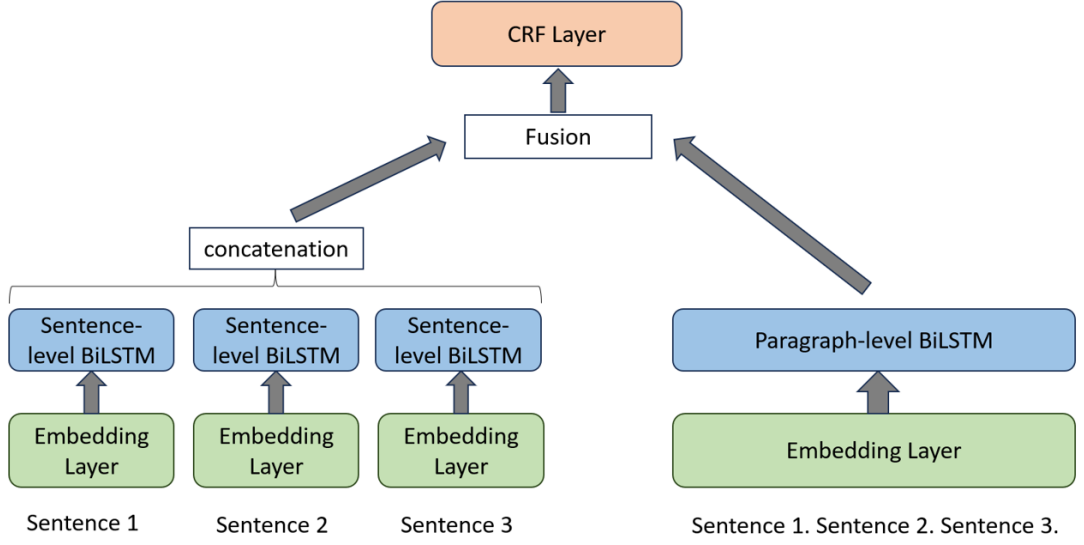


Figure 5.5: Multi-Granularity Reader [30]

of the sentence.

$$\begin{aligned}
 \tilde{p}_1^1, \tilde{p}_2^1, \dots, \tilde{p}_{l_1}^1 &= BiLSTM_{sentence}(\tilde{x}_1^1, \tilde{x}_2^1, \dots, \tilde{x}_{l_1}^1) \\
 \tilde{p}_1^2, \tilde{p}_2^2, \dots, \tilde{p}_{l_2}^2 &= BiLSTM_{sentence}(\tilde{x}_1^2, \tilde{x}_2^2, \dots, \tilde{x}_{l_2}^2) \\
 &\dots \\
 \tilde{p}_1^k, \tilde{p}_2^k, \dots, \tilde{p}_{l_k}^k &= BiLSTM_{sentence}(\tilde{x}_1^k, \tilde{x}_2^k, \dots, \tilde{x}_{l_k}^k)
 \end{aligned} \tag{5.5}$$

And paragraph-level BiLSTM, on the other hand, processes each paragraph as a sequence. By considering more text at a time, this model can capture more context and better understand the relationships between sentences.

$$\hat{p}_1^1, \dots, \hat{p}_{l_1}^1, \dots, \hat{p}_1^k, \dots, \hat{p}_{l_k}^k = BiLSTM_{paragraph} = \hat{x}_1^1, \dots, \hat{x}_{l_1}^1, \dots, \hat{x}_1^k, \dots, \hat{x}_{l_k}^k \tag{5.6}$$

5.2.3.3 Fusion Layer

Fusion involves combining the contextual representations obtained from last layers which are Sentence-Level BiLSTM and Paragraph-Level BiLSTM outputs. This can be achieved in various ways including concatenation, element-wise addition, or weighted fusion operations. Weighted fusion uses attention mechanisms

to weigh the importance of different components of these representations before they are combined. This step uses model learnable weights to decide how much information should be propagated to next layers. The equation is as follow:

$$\begin{aligned} g_i^j &= \text{sigmoid}(W_1 \tilde{p}_i^j + W_2 \hat{p}_i^j + b) \\ p_i^j &= g_i^j \cdot \tilde{p}_i^j + (1 - g_i^j) \cdot \hat{p}_i^j \end{aligned} \tag{5.7}$$

5.2.4 Position Encoding

Positional information can be added to models using various methods, each with its own strengths and weaknesses. The common approaches are as follows. The first approach is known as Absolute Position Encoding. This method involves adding a unique identifier to each position in the sequence, usually in the form of a vector. These vectors can be learned, which is certain implementations of Transformer models, or fixed. One common form of fixed absolute position encoding is Sinusoidal Encoding [119] which generates positional encodings using a combination of sine and cosine functions, allowing the model to extrapolate to sequence lengths beyond those seen in training.

Another method is Relative Position Encoding which, in contrast to absolute position encoding, focuses on the relative positions between tokens in a sequence, rather than their absolute positions. This approach can be more robust to variations in sequence length and could potentially capture more meaningful relationships between tokens, particularly in tasks where the relative order is more important than the absolute order [107].

5.3 Proposed Method

Adding positional information to a Multi-Granularity Contextualized Encoding method involves encoding the position of words, sentences and paragraphs within the document and incorporating these positional encodings into the model for the training, evaluation and testing.

5.3.1 Motivation and Analysis

Document-Level Event Role Filler Extraction is a complex task in natural language processing (NLP) that involves identifying and categorizing entities or phrases that play specific roles within events described across entire documents. Traditional models, including pre-trained language models like BERT, while powerful, face significant challenges in this domain. They often struggle with capturing the nuanced semantic relationships that span across large text segments and dealing with the noisy or incomplete data typical in real-world documents [43]. Additionally, the inherently hierarchical structure of documents, from characters to sentences and beyond, presents a further layer of complexity that requires an innovative approach to effectively model and understand [50].

To address these challenges, this research introduces novel position encoding strategies within a multi-granularity contextualized model. By integrating both sinusoidal and relative position encodings, the model significantly enhances its ability to capture the contextual interdependencies and the intricate hierarchical structures within texts. This approach allows for a more nuanced understanding of the spatial and sequential relationships between textual elements, crucial for accurately extracting event role fillers [119].

Sinusoidal position encoding, inspired by the Transformer architecture [27], provides a continuous representation that can effectively model the sequence of words or tokens within a document. This method is particularly beneficial for modelling long-range dependencies, a common challenge in document-level tasks.

Relative position encoding, on the other hand, focuses on the relative distances between tokens, offering a dynamic way to understand the context and relationships between words regardless of their absolute position in the text. This method is crucial for tasks like event extraction, where the relevance of a token to an event may depend more on its relation to other key tokens than its specific location within the document.

5.3.2 Sinusoidal Position Encoding

The first step is to compute positional encodings that absolute and relative positional encoding methods are chosen from. Transformer [119] model raises an absolute positioning called the sinusoidal position method which incorporates sine and cosine functions with varying frequencies to encode the position of each token in the input sequence. The equations are as below:

$$\begin{aligned}\text{Pos_enc}(pos, 2i) &= \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right) \\ \text{Pos_enc}(pos, 2i + 1) &= \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)\end{aligned}\tag{5.8}$$

where pos represents the position of the token, ranging from 0 to the length of the input sequence minus 1. $2i$ and $2i + 1$ denote the even and odd dimensions respectively in the positional encoding matrix. d_{model} represents the dimensionality of the model, which is typically the same as the embedding dimension used for the input tokens.

The sinusoidal position encoding equations follow a pattern where the frequencies of the sine and cosine functions decrease exponentially as the dimension index i increases. The scaling factor 10000 ensures that the frequencies vary appropriately across different dimensions. The choice of this factor is based on heuristics and empirical observations to achieve a reasonable range of frequencies.

5.3.3 Relative Position Encoding

Sinusoidal position encodings are absolute, which means the encoding is unique to the position of a token in a sequence and does not change relative to other tokens. Instead of giving each token a static position encoding, relative position encoding provides a dynamic encoding based on the token's position relative to other tokens in the sequence. This means the encoding changes depending on where the token is in the sequence and how it is positioned relative to other

tokens.

$$\begin{aligned}
 \text{rel_pos_emb}_{ij} &= E(j - i + \text{max_len}) \\
 \text{rel_pos_emb_final}_s &= \text{repeat} \left(\sum_j \text{rel_pos_emb}_{ij}, \text{batch_size} \right) \\
 \text{output}_s &= x_s + \text{repeat} \left(\sum_j \text{rel_pos_emb}_{ij}, \text{batch_size} \right) \\
 E(p) &= W_p
 \end{aligned} \tag{5.9}$$

where i and j represent indices within a sequence, and E represents the embedding function defined by the trainable weights of the relative position encoding module. Max_len is a predefined constant to handle negative positions. The operation $j - i + max_len$ ensures that all relative positions are non-negative. *Repeat* represents the operation of summing the embeddings along the sequence length dimension and repeating the operation for each sequence s in the batch. This results in a tensor of the same size as the input tensor, $(batch_size, seq_len, d_model)$. Then, adding the relative position embeddings to the original sequence embeddings. x_s represents the original sequence embeddings, and the result is a tensor where relative position information is incorporated into the original sequence embeddings.

5.3.4 Word embedding

Word embeddings are a powerful tool for converting discrete words into continuous vector representations that capture their semantic meanings. However, by themselves, word embeddings don't carry any information about the order of words in a sequence, which is crucial for many natural language processing tasks. We propose positional word embedding in three parts including GloVe, Contextualized embeddings and position encoding.

5.3.4.1 GloVe Embedding

GloVe is a powerful word representation technique that provides dense vector representations for words that capture different aspects of their meanings based on their co-occurrence patterns in a given corpus [96]. Typically, pre-trained GloVe embeddings are available in dimensions such as 50, 100, 200, or 300. For the model utilized in this thesis, pre-trained embeddings in 100 dimensions are applied.

$$\text{GloVe}_i = G(i) \quad (5.10)$$

5.3.4.2 Contextualized Embeddings

GloVe generates a single static embedding for each word regardless of the context. BERT [27] provides dynamic context-dependent embeddings which means that the BERT representations for a word can change depending on the sentence in which the word is used. This characteristic presents as a significant advantage when dealing with homonymous words, that is words which are spelled the same but have different meanings, and polysemous words that with multiple meanings. To compare with the previous study, average of all twelve layers representations are selected to freeze the weight . The equation is as follows:

$$\text{BERT}_i = \text{BERT}(i) \quad (5.11)$$

5.3.4.3 Embedding Combination

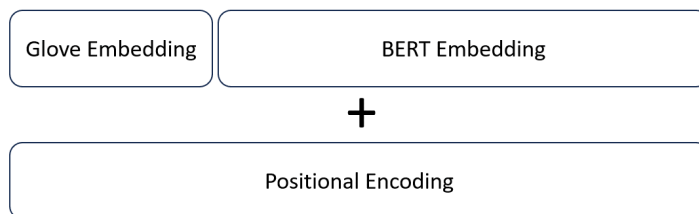


Figure 5.6: Word Embedding is GloVe and BERT Concatenation plus Positional Encoding

To combine GloVe, BERT, and positional encoding into a single word embedding, GloVe and BERT embeddings are concatenated first. Then positional encoding is added to the last concatenation result. The equation is as follow:

$$E_i = \text{concat}(G(i) + BERT(i)) + \text{pos_embedding}(i) \quad (5.12)$$

5.4 Experiment

We test the performance of the positional information in multi-granularity contextualized encoding model regarding event role filler extraction using MUC-4 [84] which is benchmarked to event extraction applications. Three aspects, namely are relative encoding, static encoding and traditional multi-granularity model, are compared in experiment.

5.4.1 Environment and Parameters

The results from a previously reported multi-granularity model [30] were quite encouraging. The novel multi-granularity reader, which dynamically blends both sentence and paragraph level context, performed better than both the traditional non-end-to-end systems and the base k-sentence readers. Thus, the configuration of the system in the experiment was set the same as the multi-granularity model. The model was implemented on a machine with the following specifications: 6G graphical memory card GPU and 32 RAM computer. The system runs on a Windows 10 operating system. And all programming was done in Python 3.9. The deep learning portion of the system was built using PyTorch framework. We then proceeded to use the fine tuned sentence-level and paragraph-level reader as the multi-granularity modules. For word embedding, concatenation of 100 dimensional GloVe and 768 dimensional BERT pre-trained embeddings are utilized. This leverages the semantic richness of GloVe and the contextual understanding provided by BERT. In addition to this, positional encoding information is added

to the 868 dimensional embeddings for a word.

5.4.2 Dataset

The dataset used in this experiment is known as MUC-4 and arose from a series of conferences over several years entitled Message Understanding Conference (MUC). These conferences were held to discuss and evaluate the performance of different types of natural language processing systems. The MUC-4 dataset provides a basis for extracting structured information from unstructured text data. Specifically, the MUC-4 dataset is used for tasks related to event extraction and revolves around a terrorism domain and involves a collection of texts that need to be processed to identify key information, such as the perpetrator of an attack, the type of attack, the location, the time, and the result.

The dataset itself consists of a set of news articles in English that describe various terrorist activities. There are 1,700 documents which are divided into 1,300 training set, 300 development set and 300 test set to train, evaluate and test the model's performance.

5.4.3 Justification of Datasets Used

Complex Event Structures: The datasets may have been selected due to their complex event structures, which include various roles like perpetrators, targets, and victims, challenging the models to accurately identify and classify nuanced relationships and entities within text.

Real-world Relevance: The datasets probably have high real-world relevance, containing events that are significant for security, social, or economic analysis. This relevance ensures that the improvements in model performance have practical implications for information retrieval, surveillance, or decision-making.

Benchmarking Purposes: These datasets might serve as benchmarks in the field of event extraction, allowing for a direct comparison with previous works. Using well-established datasets ensures that the results are meaningful and comparable

across different studies.

5.4.4 Evaluation Methods

Precision, Recall, and F1 score are common evaluation metrics used in information retrieval and deep learning tasks to measure the quality of a model's predictions [58]. A brief description of each of these metrics is as follows.

Precision measures how many of the items that a model identified were actually relevant. It is calculated as the number of true positives, which are items correctly identified as relevant, divided by the sum of true positives and false positives, which are items incorrectly identified as relevant. A higher precision score indicates that when the model identifies something as relevant which is likely to be correct. Precision is crucial when it's important for the extracted event role fillers to be as accurate as possible. A high precision indicates that the model returns more relevant results than irrelevant ones. This can be particularly important if false positives (incorrectly identified fillers) could have negative consequences.

Recall measures how many of the truly relevant items were identified by the model. It is calculated as the number of true positives divided by the sum of true positives and false negatives which are relevant items that the model failed to identify. A higher recall score means the model is good at catching all the relevant cases. Recall is important when it's crucial to extract as many relevant event role fillers as possible. A high recall indicates that the model is good at capturing all potentially relevant fillers. This may be key if missing a relevant role filler could have detrimental effects.

The third metric discussed here, known as F1 score, is the harmonic mean of precision and recall, and it provides a single measure that balances the two. Because it considers both precision and recall, the F1 score can be a better measure than either metric alone, particularly in situations where the distribution of positives and negatives is uneven. An F1 score reaches its best value at 1 and worst at 0.

Head Noun Match and Exact Match are two major ways to measure the performance of event role filler extraction tasks. Head Noun Match means a correct extraction is counted if the head noun of the extracted phrase matches with the head noun of the target phrase. For example, if the target is 'the new table' and the extracted phrase is 'new table', then it is considered a correct extraction, because the head noun 'table' matches in both phrases. Exact Match means a correct extraction is counted only if the extracted phrase exactly matches with the target phrase. Using the same example, 'the new table' would only be considered a correct extraction if the extracted phrase was also exactly 'the new table'. This is a stricter measure than the head noun match, as it requires the complete phrase to be correct, not just the head noun.

5.4.5 Techniques Used in Comparison

Multi-Granularity (MG): This approach is likely chosen for its ability to capture information at different levels of detail, from individual words to entire phrases or sentences. This is particularly useful for event extraction, where the significance of an entity or action can depend on its context within a larger narrative structure.

Sinusoidal Position Multi-Granularity (Sin Pos MG): The inclusion of sinusoidal position embeddings addresses the limitation of static embeddings by incorporating information about the relative or absolute position of words in the text. This technique is particularly relevant for event extraction, where the order of words can change the meaning of an event or its participants' roles.

Relative Position Multi-Granularity (Rel Pos MG): This method extends the idea of positional information by focusing on the relative positions of words to each other, which is crucial for understanding complex event structures where the relationship between entities and actions is not merely linear but hierarchical or networked.

These techniques are chosen because they represent innovative approaches to

addressing the challenges of event extraction, such as the need to understand context, capture relationships between entities, and deal with the variability of natural language. By comparing these methods, the study aims to highlight the importance of positional and granularity considerations in improving the performance of event extraction systems.

5.4.6 Experiment Result

The metrics used for evaluation are Precision, Recall, and F1 score, with the results reported separately for Head Noun Match and Exact Match. The MACRO average accuracy results are in table 5.1 Adding Sinusoidal Position Encoding to

	Head Noun Match			Exact Match		
	Precision	Recall	F1	Precision	Recall	F1
Multi-Granularity (MG)	56.44	62.77	59.44	52.03	56.81	54.32
Sinusoidal Position MG	54.18	67.45	60.09	49.46	62.96	55.40
Relative Position MG	61.26	60.06	60.65	56.49	56.81	56.74

Table 5.1: Performance Comparison between Multi-Granularity and Positional Multi-Granularity Methods.

the MG model slightly decreases precision but significantly improves recall, indicating that the model is able to capture more relevant instances at the expense of incorrectly identifying some irrelevant instances. The F1 score, which balances precision and recall, shows a slight improvement for Head Noun Match, but a small decrease for Exact Match. This suggests that Sinusoidal Position Encoding may be more beneficial when Head Noun Match are acceptable.

The Relative Position MG encoding shows a significant improvement in precision over both the base MG and Sinusoidal Position MG models, with a small decrease in recall for the Head Noun Match and almost no change in recall for Exact Match. This implies that the model is better at accurately identifying relevant instances, but may miss some instances. The F1 scores for both the Head Noun Match and Exact Match are higher than for the other two models, suggesting that the Relative Position MG model provides the best overall performance

among the three.

In summary, the use of position encoding methods, specifically Relative Position Encoding, has shown to improve the performance of the base MG model. These findings suggest that incorporating positional information can enhance the extraction accuracy in tasks such as event role filler extraction. However, the optimal choice of position encoding may depend on the specific task and the importance placed on precision versus recall.

Each event role categories results are in table 5.2. The F1 score, which pro-

	PerpInd			PerpOrg			Target			Victim			Weapon		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
MG	53.08	52.23	52.65	50.99	67.88	58.23	60.38	64.10	62.18	49.34	62.05	54.97	68.42	67.57	67.99
Sin Pos MG	58.20	50.68	54.18	57.55	59.52	58.52	60.31	75.86	67.20	44.85	75.79	56.35	50.00	75.41	60.13
Rel Pos MG	64.77	43.24	51.86	50.37	65.48	56.94	64.66	58.62	61.49	53.28	67.37	59.50	73.21	65.57	69.18

Table 5.2: Event Role Performance Comparison for Multi-Granularity (MG), the Sinusoidal Position Multi-Granularity (Sin Pos MG) and the Relative Position Multi-Granularity (Rel Pos MG).

vides a balanced measure of a model’s precision and recall, is particularly crucial in the context of information extraction tasks like MUC-4 Event Role Filler Extraction. A higher F1 score indicates a more balanced and accurate model, which correctly identifies more true positives while minimizing both false positives and false negatives.

In the Table 5.2, it is clear that the Relative Position Multi-Granularity (Rel Pos MG) model typically performs well across multiple event role categories - Perpetrator Individual (PerpInd), Perpetrator Organization (PerpOrg), Victim, and Weapon - consistently achieving the highest or competitive F1 scores.

Although for the Perpetrator Individual (PerpInd) category, the Multi-Granularity (MG) model exhibits the highest F1 score, the Relative Position MG model surpasses both the MG and the Sinusoidal Position MG models in the Perpetrator Organization (PerpOrg) and Victim categories in terms of F1 score. This suggests that relative position information can significantly enhance the model’s performance, particularly in correctly identifying organizations and individuals involved

in the event.

Interestingly, in the Target category, the Sinusoidal Position MG model achieves the highest F1 score, indicating that in this specific category, considering absolute position information can be beneficial. However, for the Weapon category, the Relative Position MG model returns to providing the highest F1 score, emphasizing the importance of relative positioning in such scenarios.

In conclusion, despite some exceptions, the Relative Position MG model tends to have overall best performance, achieving higher or highly competitive F1 scores across different event role categories. This emphasizes that considering relative position information can substantially enhance the extraction of event role fillers which indicates that the Relative Position MG model is an effective approach for tasks like MUC-4 Event Role Filler Extraction.

Each event role results analysis suggests that different models excel in different aspects. Depending on the specific needs of a task-whether it is more important to minimize false positives (higher precision), minimize false negatives (higher recall), or balance the two (higher F1)-one may choose to use the MG, Sinusoidal Position MG, or Relative Position MG model. However, the Relative Position MG model shows the most consistent performance across different roles, indicating that considering the relative positions of tokens could be beneficial for the task of event role filler extraction.

5.5 Analysis

The following provides a discussion of the benefits of Sinusoidal and Relative Position encodings that are relevant to the experiments within this chapter.

- **Sinusoidal Position Encoding** for each position is a combination of sine and cosine functions of different frequencies. Specifically, each dimension of the positional encoding corresponds to a sine or cosine of a different frequency, which allows the model to potentially learn to attend to relative

positions based on these frequencies. The use of a combination of sine and cosine functions creates a unique and continuous mapping for each position, which enables the model to easily learn relative positional relationships. Additionally, because the sine and cosine functions are periodic, this encoding method can create meaningful relationships between distant positions.

- **Relative Position Encoding** focuses on the relationships between positions. This can be thought of as creating a 'difference vector' between every pair of positions, which directly models the concept of 'distance' between words in a sequence. This method is particularly powerful because it transforms the problem with inference of absolute positions that can be arbitrarily far apart, and difficult for the model to generalize to reason about relative distances which are more bounded and easier for the model to understand. Relative position encoding emphasizes the distance or difference between positions. For instance, instead of considering a word as being the 10th word in a sentence (absolute position), one could consider it as being 2 words after the 8th word, 1 word before the 11th word, and so on. This approach is especially useful in natural language understanding tasks because the semantic relationship between words often depends more on their relative positions than their absolute positions. For example, in the sentence 'The cat chased the mouse,' the relationship between 'cat' and 'chased' is the same regardless of whether 'cat' is the second word in the sentence or the tenth word in a longer sentence.

Traditional multi-granularity encoding models utilize BERT embeddings that includes position information, which are added to the word embeddings to provide the model with some notion of the position of the words in a sequence. However, the BERT model's position embeddings are learned, and the model is limited to a fixed sequence length that is determined at training time.

Adding Sinusoidal Position and Relative Position encodings can offer advantages over this learned position encoding for several reasons:

- **Flexibility:** Unlike the learned position embeddings in BERT, sinusoidal and relative position encoding can handle sequences of any length. This can be beneficial in tasks where the input sequences can vary greatly in length.
- **Different positional perspectives:** The position of a word in a sentence can influence its meaning, and different tasks may require different kinds of positional information. The learned position embeddings in BERT provide a 'global' position perspective, but they may not be as effective at capturing 'local' positional relationships between nearby words. Sinusoidal and relative position encodings can provide additional or alternative positional perspectives that can be beneficial in certain tasks.
- **Expanded Representational Capacity:** Relative position encoding has a larger and more flexible representational capacity. If n is the sequence length, the absolute position encoding has n unique embeddings, whereas the relative position encoding has n^2 unique embeddings, one for each pairwise combination of positions in the sequence. This allows RPE to capture a broader range of positional relationships, which may be why it often outperforms absolute position encoding methods on certain tasks.

5.6 Conclusion

In this study, the addition of relative position encoding in a multi-granularity model for Document-Level Event Role Filler Extraction has been investigated. The main findings suggest that incorporating relative positional information in the model significantly improves performance over using absolute position encodings or no explicit position encoding. The improvements appear to have arisen from the expanded representational capacity of relative position encoding that is applied here, which can capture a broader range of positional relationships.

In comparison to the standard Multi-Granularity approach and Sinusoidal Position MG, the Relative Position MG consistently showed a superior or comparable

performance across various event roles, particularly in the F1 score which combines precision and recall. This indicates that the model is more balanced and effective at identifying role fillers in both common and edge-case scenarios.

While the model with relative position encoding demonstrated strong performance, there are still areas for improvement. For example, this model did not consistently outperform in every category, indicating there may be other factors influencing the performance.

For future work, it will be beneficial to explore how different types of position encoding approaches could be combined or adapted to further improve model performance. More complex or task-specific position encoding strategies could yield additional improvements.

Furthermore, it would be interesting to investigate the impact of positional encoding in other natural language processing tasks, particularly ones that rely heavily on the relative positions of elements in a sequence. Such tasks could potentially benefit even more from the expanded representational capacity of relative position encoding.

Moreover, there could be potential to enhance the model’s performance by incorporating other forms of contextual information, such as semantic relationships between elements in a sequence or larger-scale document structure. Arguably, this can be especially beneficial for tasks dealing with longer texts or more complex event structures.

Chapter 6

Conclusion

This thesis has made foundational contributions to the field of text classification and information extraction through innovative deep learning algorithms, which can be utilized to improve NLP applications such as information compliance checking systems. The research in this thesis has focused on enhancing the efficiency and robustness of NLP techniques powered by deep learning methods. Novel methodologies and approaches in the fields of text classification, information extraction, and document-level event role filler extraction have been applied.

6.1 Overview of The Previous Methods

The first method (described in Chapter 3) is centred around text classification, where the goal in those studies was to improve the accuracy of compliance information checking using GCNs. Traditional neural networks struggle with capturing the complex relations within a document, which are crucial for accurate information compliance checking. To overcome this limitation, the proposed approach involved a document-relational GCNs that incorporated cumulative TF-IDF document-document relations as features. This was to allow for the creation of complex and rich relation-based adjacent matrix graphs, leading to superior accuracy in text classification for compliance documents.

The second method focused on information extraction, specifically in the context of heterogeneous linguistic graphs. Pre-trained language models such as BERT are powerful in capturing complex linguistic information, but they face challenges in handling semantic relationships and noisy data. To address this, the novel GraphNorm method was introduced as an approach to fine information extraction from such models. GraphNorm employs dynamic normalization layers for each GNN module in the HLG model, which improves accuracy and reduces memory consumption during the training process. In experiments described in Chapter 4, by regulating the influence of outliers and noisy data points, the application of GraphNorm enhanced the training model’s resilience to graph modifications, and resulted in more effective in handling real-world scenarios.

The third method revolved around document-level event role filler extraction, which requires capturing contextual interdependencies and internal hierarchical structure in text. To achieve this, novel position encoding strategies were proposed within a multi-granularity contextualized model. By integrating sinusoidal and relative position encodings at different levels of granularity, a richer and more contextualized representation of words were obtained. The approach, described in Chapter 5, combined GloVe word embeddings with context-aware BERT embeddings, and this effectively captured sequential and contextual aspects of text in a dataset, MUC-4, so as to improve the accuracy of event extraction tasks.

Overall, these three approaches described in this thesis contributes to the advancement of information compliance checking by enhancing the efficiency and accuracy of NLP techniques. By incorporating innovative approaches in text classification, information extraction, and document-level event role filler extraction, this research opens up new possibilities for improving productivity, accuracy, and automation in information checking tasks. The following sections describe how the methods introduced in this thesis provide valuable insights, and how techniques that can be applied in other domains and industries, leading to further advancements in machine learning and AI.

6.2 Significance of the thesis

The thesis makes significant contributions to the field of information compliance checking through the development and refinement of deep learning algorithms, as discussed in the three major points as below.

6.2.1 Text Classification on GCNs

Novel Methodology: This study has introduced a novel methodology in the context of GCNs, termed document-relational GCNs. It departs from traditional methods in that it created adjacent matrix graphs in GCNs at the word-document and word-word levels, and added a new dimension of document-document relations into the learning.

Utilization of Cumulative TF-IDF. This thesis has leveraged cumulative TF-IDF to generate document-document relations, and the results showed that this approach added complexity and richness to the adjacency matrix graph used for text classification. This addition resulted in better results in text classification tasks.

Performance Evaluation on Multiple Datasets. The research on this topic within the thesis evaluated the performance of the new model using five popular benchmark databases. This comprehensive evaluation strengthened the validity of the model and offered a robust comparison against current methods.

Investigation of Optimal Model Parameters. This thesis explored the optimal parameters for the document-relational GCNs model and discovered that by increasing the number of hidden layers, specifically between 1200 and 1500, the test accuracy of the model was improved.

Improvement in Accuracy. The proposed model demonstrated improved accuracy over the original text GCNs and the synonym augmented GCNs in four out of five datasets. This indicated that the inclusion of document-document relations as a feature could indeed enhance the text classification performance.

6.2.2 Text Classification on HLG

GraphNorm in HLG: This work introduced GraphNorm, an innovative normalization method to improve the performance of HLG models. Application of this method was aimed to enhance model accuracy during the training process.

Dynamic Normalization Layers: GraphNorm added dynamic normalization layers to each GNN module in the HLG model. This approach retained reliable node information from the graph and mitigated noise, and increased the model’s resilience to graph modifications.

Generalization Potential: The research showed that GraphNorm has the potential to generalize across a variety of datasets, and achieved superior accuracy in 4 out of 5 benchmark datasets, and this altogether suggests that it could be beneficial for a wide range of NLP applications.

6.2.3 Information Extraction

The introduction of novel position encoding strategies within a multi-granularity contextualized model significantly advanced the extraction of event role fillers in information compliance checking. By integrating sinusoidal and relative position encodings at different levels of granularity, the method captures the contextual interdependencies and internal hierarchical structure in text more effectively. This contribution results in a richer, more contextualized representation of words, crucial for accurate event extraction. The research expands the understanding of position encoding techniques and their application in document-level event analysis, with implications for improving information extraction in various domains. Collectively, these three contributions described in this thesis push the boundaries of deep learning in the context of information compliance checking. The research has introduced innovative methodologies that improved the efficiency, accuracy, and automation of labour-intensive tasks in information checking fields. By addressing the challenges associated with text classification, information extraction, and document-level event analysis, the thesis offers valuable insights and tech-

niques that can be applied in other industries and domains. The contributions made in this research have the potential to revolutionize information checking workflows and pave the way for future breakthroughs in machine learning and artificial intelligence.

6.3 Future Researches

The thesis opens up several avenues for future research in the field of information compliance checking and deep learning. Some potential areas of exploration are discussed below:

Document Relational Investigation: Filtering of useful or important document relation features, the development of sophisticated algorithms for calculating document-document relations, and the simplification of the graph could further improve the model. This foresight helps direct further research in this area.

Enhanced Graph Neural Networks: Further advancements can be made in the design and implementation of GCNs for information compliance checking. Future research can focus on exploring more sophisticated graph structures and techniques for feature engineering. This includes investigating different graph architectures, exploring attention mechanisms in GCNs, and leveraging other graph-based deep learning models, such as graph attention networks or graph transformers. Additionally, the integration of domain-specific knowledge and expert rules into GCNs can be explored to further improve their performance in compliance checking tasks.

Multi-Modal Learning: Compliance information often contain multiple types of information, including textual descriptions, graphical elements, and spatial relationships. Future research can investigate the integration of multi-modal learning techniques to leverage these different modalities for more comprehensive and accurate compliance checking. This can involve combining text-based deep learning models with image-based models, such as CNN, and exploring methods to

fuse information from multiple modalities effectively.

Active Learning and Data Augmentation: information compliance checking typically requires a large amount of labelled data for training deep learning models. Future research can explore methods for active learning, where the model actively selects the most informative data samples to be labelled, reducing the annotation effort. Additionally, data augmentation techniques specifically tailored for information can be developed to generate synthetic data and increase the diversity of the training set. This can help improve the robustness and generalization of the models in real-world scenarios.

Explainability and Interpretability: Deep learning models are often considered as black boxes, making it challenging to understand their decision-making processes. Future research can focus on developing methods for interpreting and explaining the decisions made by compliance documents checking models. This can involve techniques such as attention visualization, feature attribution, and rule extraction, which can provide insights into how the models analyze and interpret documents to make compliance judgments.

With regards to the GraphNorm-HLG experiments carried out in this thesis (Chapter 4), future work could focus on evaluating GraphNorm-HLG using larger, more complex datasets than the five benchmark datasets used. Also, datasets specific to different domains, such as medical, financial and legal texts, could also be used so as to assess GraphNorm-HLG performs in a variety of other real-world situations.

Another point of future research is to optimize GraphNorm for Memory Usage and Speed . While GraphNorm improves accuracy and reduces memory consumption, there is likely further optimization that can make it more efficient in terms of computational time and memory usage. Future work could also investigate how GraphNorm influences various aspects of NLP tasks, such as semantic understanding, entity recognition, and sentiment analysis, among others.

Studies that focussed on positional encoding in this thesis involved the application

of two strategies, sinusoidal and relative positional encodings, for the first time in the domain of Document-Level Event Role Filler Extraction. This innovative application of encoding methods enhances the understanding of the contextual interdependencies and the hierarchical structure in a given text. Also, a combination of GloVe word embeddings, BERT embeddings and the new positional encodings were applied, and this resulted in richer, more contextualized representation of words from datasets. This work shows that enhanced representation is particularly crucial for tasks like event extraction. Furthermore, the proposed method, which harnessed the strengths of BiLSTM-CRF models and the new positional encoding strategies, yielded superior performance on event role filler extraction tasks. This superiority was demonstrated through experimental results using the MUC-4 dataset, where the model outperformed existing benchmarks. Our relative position encoding approach showed high performance across multiple event role categories. This is indicative of the broad applicability and generalization of the model to various tasks and scenarios. Our analysis of the results provides valuable insights into which position encoding strategy works best for specific tasks and event role categories. These insights offer guidelines for future task-specific model adaptation.

By addressing these future research directions, the field of information compliance checking can continue to benefit from the advancements in deep learning, leading to more accurate, efficient, and automated approaches for ensuring compliance checking domains.

Bibliography

- [1] A. A. Abro, et al. (2023). ‘Natural language processing challenges and issues: A literature review’. *Gazi University Journal of Science* pp. 1–1.
- [2] T. Akhmetshin (2023). *Graph-based neural networks for generation of synthetically accessible molecular structures*. Ph.D. thesis, Université de Strasbourg; Kazanskij gosudarstvennyj universitet im. VI
- [3] M. S. R. Amin & L. Amador (2014). ‘The multi-criteria based pavement management system for regional road network of Atlantic Provinces of Canada’. *International Journal of Pavements* **13**(1-2-3).
- [4] J. L. Ba, et al. (2016). ‘Layer normalization’. *arXiv preprint arXiv:1607.06450* .
- [5] C. Baziotis, et al. (2017). ‘Datastories at semeval-2017 task 4: Deep LSTM with attention for message-level and topic-based sentiment analysis’. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pp. 747–754.
- [6] M. Belsky, et al. (2016). ‘Semantic enrichment for building information modeling’. *Computer-Aided Civil and Infrastructure Engineering* **31**(4):261–274.
- [7] I. Beltagy, et al. (2019). ‘SciBERT: A pretrained language model for scientific text’. *arXiv preprint arXiv:1903.10676* .
- [8] A. Benamira, et al. (2019). ‘Semi-supervised learning and graph neural networks for fake news detection’. In *2019 IEEE/ACM International Conference*

- on *Advances in Social Networks Analysis and Mining (ASONAM)*, pp. 568–569. IEEE.
- [9] F. M. Bianchi, et al. (2020). ‘Hierarchical representation learning in graph neural networks with node decimation pooling’. *IEEE Transactions on Neural Networks and Learning Systems* .
- [10] P. Bojanowski, et al. (2017). ‘Enriching word vectors with subword information’. *Transactions of the association for computational linguistics* **5**:135–146.
- [11] L. Bozarth & C. Budak (2020). ‘Toward a better performance evaluation framework for fake news classification’. In *Proceedings of the International AAAI Conference on Web and Social Media*, vol. 14, pp. 60–71.
- [12] M. M. Bronstein, et al. (2017). ‘Geometric deep learning: going beyond euclidean data’. *IEEE Signal Processing Magazine* **34**(4):18–42.
- [13] K. Cai, et al. (2021). ‘Feedback convolutional network for intelligent data fusion based on near-infrared collaborative IoT technology’. *IEEE Transactions on Industrial Informatics* **18**(2):1200–1209.
- [14] E. Cambria & B. White (2014). ‘Jumping NLP curves: A review of natural language processing research’. *IEEE Computational Intelligence Magazine* **9**(2):48–57.
- [15] W. B. Cavnar, et al. (1994). ‘N-gram-based text categorization’. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, vol. 161175. Las Vegas, NV.
- [16] I. Chalkidis, et al. (2019). ‘Large-scale multi-label text classification on EU legislation’. *arXiv preprint arXiv:1906.02192* .
- [17] Y.-N. Chang & C.-Y. Lee (2018). ‘Semantic ambiguity effects on traditional Chinese character naming: A corpus-based approach’. *Behavior Research Methods* **50**:2292–2304.

- [18] T. Chen, et al. (2017). ‘Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN’. *Expert Systems with Applications* **72**:221–230.
- [19] W.-L. Chiang, et al. (2019). ‘Cluster-gcn: An efficient algorithm for training deep and large graph convolutional networks’. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 257–266.
- [20] F. Colas & P. Brazdil (2006). ‘Comparison of SVM and some older classification algorithms in text classification tasks’. In *IFIP International Conference on Artificial Intelligence in Theory and Practice*, pp. 169–178. Springer.
- [21] A. Conneau, et al. (2018). ‘XNLI: Evaluating cross-lingual sentence representations’. *arXiv preprint arXiv:1809.05053* .
- [22] C. Coulombe (2018). ‘Text data augmentation made simple by leveraging nlp cloud apis’. *arXiv preprint arXiv:1812.04718* .
- [23] P. Covington, et al. (2016). ‘Deep neural networks for youtube recommendations’. In *Proceedings of the 10th ACM conference on recommender systems*, pp. 191–198.
- [24] A. Creswell, et al. (2018). ‘Generative adversarial networks: An overview’. *IEEE Signal Processing Magazine* **35**(1):53–65.
- [25] K. Da (2014). ‘A method for stochastic optimization’. *arXiv preprint arXiv:1412.6980* .
- [26] T. Däubler, et al. (2012). ‘Natural sentences as valid units for coded political texts’. *British Journal of Political Science* **42**(4):937–951.
- [27] J. Devlin, et al. (2018). ‘Bert: Pre-training of deep bidirectional transformers for language understanding’. *arXiv preprint arXiv:1810.04805* .

- [28] J. Ding, et al. (2019). ‘Meta-path based text feature enrichment using knowledge graph’. In *2019 IEEE Fourth International Conference on Data Science in Cyberspace (DSC)*, pp. 649–655. IEEE.
- [29] P.-T. Doutre, et al. (2017). ‘Comparison of some approaches to define a CAD model from topological optimization in design for additive manufacturing’. In *Advances on Mechanics, Design Engineering and Manufacturing: Proceedings of the International Joint Conference on Mechanics, Design Engineering & Advanced Manufacturing (JCM 2016), 14-16 September, 2016, Catania, Italy*, pp. 233–240. Springer.
- [30] X. Du & C. Cardie (2020). ‘Document-level event role filler extraction using multi-granularity contextualized encoding’. *arXiv preprint arXiv:2005.06579* .
- [31] S. Ghosh & C. Shah (2018). ‘Towards automatic fake news classification’. *Proceedings of the Association for Information Science and Technology* **55**(1):805–807.
- [32] S. Goguelin, et al. (2016). ‘A bottom-up design framework for CAD tools to support design for additive manufacturing’. In *Sustainable Design and Manufacturing 2016*, pp. 411–421. Springer.
- [33] W. Hamilton, et al. (2017). ‘Inductive representation learning on large graphs’. *Advances in neural information processing systems* **30**.
- [34] J. B. Hamrick, et al. (2018). ‘Relational inductive bias for physical construction in humans and machines’. *arXiv preprint arXiv:1806.01203* .
- [35] Z. S. Harris (1954). ‘Distributional structure’. *Word* **10**(2-3):146–162.
- [36] H. He & J. D. Choi (2021). ‘The stem cell hypothesis: Dilemma behind multi-task learning with transformer encoders’. *arXiv preprint arXiv:2109.06939* .

- [37] Y. He, et al. (2020). ‘Multi-Branch Deep Residual Learning for Clustering and Beamforming in User-Centric Network’. *IEEE Communications Letters* **24**(10):2221–2225.
- [38] W. Hersh, et al. (1994). ‘OHSUMED: an interactive retrieval evaluation and new large test collection for research’. In *SIGIR 94*, pp. 192–201. Springer.
- [39] J. Hirschberg & C. D. Manning (2015). ‘Advances in natural language processing’. *Science* **349**(6245):261–266.
- [40] S. Hochreiter & J. Schmidhuber (1997). ‘Long short-term memory’. *Neural Computation* **9**(8):1735–1780.
- [41] M. J. Hofmann, et al. (2018). ‘Simple co-occurrence statistics reproducibly predict association ratings’. *Cognitive Science* **42**(7):2287–2312.
- [42] C. C. Horgan, et al. (2021). ‘High-throughput molecular imaging via deep-learning-enabled raman spectroscopy’. *Analytical Chemistry* **93**(48):15850–15860.
- [43] S. Hou, et al. (2017). ‘Hindroid: An intelligent android malware detection system based on structured heterogeneous information network’. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1507–1515.
- [44] X. Hou, et al. (2021). ‘Graph ensemble learning over multiple dependency trees for aspect-level sentiment classification’. *arXiv preprint arXiv:2103.11794* .
- [45] Y. Huang, et al. (2018). ‘Learning semantic concepts and order for image and sentence matching’. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6163–6171.
- [46] Z. Huang, et al. (2015). ‘Bidirectional LSTM-CRF models for sequence tagging’. *arXiv preprint arXiv:1508.01991* .

- [47] S. Ioffe & C. Szegedy (2015). ‘Batch normalization: Accelerating deep network training by reducing internal covariate shift’. In *International Conference on Machine Learning*, pp. 448–456. pmlr.
- [48] K. Jamroz, et al. (2014). ‘Tools for road infrastructure safety management—Polish experiences’. *Transportation Research Procedia* **3**:730–739.
- [49] H. Jelodar, et al. (2020). ‘Deep sentiment classification and topic discovery on novel coronavirus or covid-19 online discussions: NLP using LSTM recurrent neural network approach’. *arXiv preprint arXiv:2004.11695* .
- [50] R. Jia, et al. (2019). ‘Document-Level N -ary Relation Extraction with Multiscale Representation Learning’. *arXiv preprint arXiv:1904.02347* .
- [51] K. Jing & J. Xu (2019). ‘A survey on neural network language models’. *arXiv preprint arXiv:1906.03591* .
- [52] M. Kanakaraj & R. M. R. Guddeti (2015). ‘NLP based sentiment analysis on Twitter data using ensemble classifiers’. In *2015 3rd International Conference on Signal Processing, Communication and Networking (ICSCN)*, pp. 1–5. IEEE.
- [53] S.-W. Kim & J.-M. Gil (2019). ‘Research paper classification systems based on TF-IDF and LDA schemes’. *Human-centric Computing and Information Sciences* **9**:1–21.
- [54] Y. Kim (2014). ‘Convolutional neural networks for sentence classification’. *arXiv preprint arXiv:1408.5882* .
- [55] T. N. Kipf & M. Welling (2016). ‘Semi-supervised classification with graph convolutional networks’. *arXiv preprint arXiv:1609.02907* .
- [56] R. Klabunde (2002). ‘Daniel Jurafsky/James H. Martin, speech and language processing’. *Zeitschrift für Sprachwissenschaft* **21**(1):134–135.

- [57] S. Kobayashi (2018). ‘Contextual augmentation: Data augmentation by words with paradigmatic relations’. *arXiv preprint arXiv:1805.06201* .
- [58] S. B. Kotsiantis, et al. (2006). ‘Machine learning: a review of classification and combining techniques’. *Artificial Intelligence Review* **26**:159–190.
- [59] S. Krish (2011). ‘A practical generative design method’. *Computer-Aided Design* **43**(1):88–100.
- [60] A. Krizhevsky, et al. (2017). ‘Imagenet classification with deep convolutional neural networks’. *Communications of the ACM* **60**(6):84–90.
- [61] G. Lample, et al. (2016). ‘Neural architectures for named entity recognition’. *arXiv preprint arXiv:1603.01360* .
- [62] Q. Le & T. Mikolov (2014). ‘Distributed representations of sentences and documents’. In *International conference on machine learning*, pp. 1188–1196. PMLR.
- [63] Y. LeCun, et al. (2015). ‘Deep learning’. *nature* **521**(7553):436–444.
- [64] J. Lee, et al. (2020). ‘BioBERT: a pre-trained biomedical language representation model for biomedical text mining’. *Bioinformatics* **36**(4):1234–1240.
- [65] K. Lee, et al. (2011). ‘Twitter trending topic classification’. In *2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 251–258. IEEE.
- [66] C. Li, et al. (2018a). ‘News text classification based on improved Bi-LSTM-CNN’. In *2018 9th International Conference on Information Technology in Medicine and Education (ITME)*, pp. 890–893. IEEE.
- [67] G. Li, et al. (2023). ‘Position-Aware Relational Transformer for Knowledge Graph Embedding’. *IEEE Transactions on Neural Networks and Learning Systems* .

- [68] Q. Li, et al. (2018b). ‘Deeper insights into graph convolutional networks for semi-supervised learning’. *arXiv preprint arXiv:1801.07606* .
- [69] Y. Li, et al. (2022). ‘Enhancing chinese pre-trained language model via heterogeneous linguistics graph’. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1986–1996.
- [70] Y. Li, et al. (2019). ‘Enhancing pre-trained chinese character representation with word-aligned attention’. *arXiv preprint arXiv:1911.02821* .
- [71] Z. Li & M. Sun (2009). ‘Punctuation as implicit annotations for Chinese word segmentation’. *Computational Linguistics* **35**(4):505–512.
- [72] H. Liu, et al. (2019a). ‘MVDLite: A Light-weight Representation of Model View Definition with Fast Validation for BIM Applications’. *arXiv preprint arXiv:1909.06997* .
- [73] H. Liu, et al. (2021a). ‘Perception consistency ultrasound image super-resolution via self-supervised CycleGAN’. *Neural Computing and Applications* pp. 1–11.
- [74] W. Liu, et al. (2021b). ‘Item relationship graph neural networks for e-commerce’. *IEEE Transactions on Neural Networks and Learning Systems* **33**(9):4785–4799.
- [75] X. Liu, et al. (2018). ‘Lcqmc: A large-scale chinese question matching corpus’. In *Proceedings of the 27th International Conference on Computational Linguistics*, pp. 1952–1962.
- [76] Y. Liu, et al. (2019b). ‘Roberta: A robustly optimized bert pretraining approach’. *arXiv preprint arXiv:1907.11692* .
- [77] P. Lops, et al. (2011). ‘Content-based recommender systems: State of the art and trends’. *Recommender Systems Handbook* pp. 73–105.

- [78] F. Luan, et al. (2017). ‘Deep photo style transfer’. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4990–4998.
- [79] A. K.-F. Lui, et al. (2021). ‘Modelling of Destinations for Data-driven Pedestrian Trajectory Prediction in Public Buildings’. In *2021 IEEE International Conference on Big Data (Big Data)*, pp. 1709–1717. IEEE.
- [80] A. K.-F. Lui, et al. (2022). ‘Modelling of Pedestrian Movements near an Amenity in Walkways of Public Buildings’. In *2022 8th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 394–400. IEEE.
- [81] X. Ma & E. Hovy (2016). ‘End-to-end sequence labeling via bi-directional lstm-cnns-crf’. *arXiv preprint arXiv:1603.01354* .
- [82] F. Majeed, et al. (2019). ‘Social media news classification in healthcare communication’. *Journal of Medical Imaging and Health Informatics* **9**(6):1215–1223.
- [83] C. D. Manning, et al. (2014). ‘The Stanford CoreNLP natural language processing toolkit’. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pp. 55–60.
- [84] V. McLean (1992). ‘Fourth message understanding conference (MUC-4)’. In *Proceedings of Fourth Message Understanding Conference (MUC-4)*.
- [85] Y. Meng, et al. (2019). ‘Glyce: Glyph-vectors for chinese character representations’. *Advances in Neural Information Processing Systems* **32**.
- [86] T. Mikolov, et al. (2013a). ‘Efficient estimation of word representations in vector space’. *arXiv preprint arXiv:1301.3781* .
- [87] T. Mikolov et al. (2012). ‘Statistical language models based on neural networks’. *Presentation at Google, Mountain View, 2nd April* **80**(26).
- [88] T. Mikolov, et al. (2010). ‘Recurrent neural network based language model.’. In *Interspeech*, pp. 1045–1048. Makuhari.

- [89] T. Mikolov, et al. (2013b). ‘Exploiting similarities among languages for machine translation’. *arXiv preprint arXiv:1309.4168* .
- [90] T. Mikolov, et al. (2013c). ‘Distributed representations of words and phrases and their compositionality’. *Advances in Neural Information Processing Systems* **26**.
- [91] J. L. Milchak, et al. (2012). ‘Implementation of a peer review process to improve documentation consistency of care process indicators in the EMR in a primary care setting’. *Journal of Managed Care Pharmacy* **18**(1):46–53.
- [92] D. Nadeau & S. Sekine (2007). ‘A survey of named entity recognition and classification’. *Linguisticae Investigationes* **30**(1):3–26.
- [93] N. O. Nawari (2019). ‘A generalized adaptive framework (GAF) for automating code compliance checking’. *Buildings* **9**(4):86.
- [94] M. Niepert, et al. (2016). ‘Learning convolutional neural networks for graphs’. In *International Conference on Machine Learning*, pp. 2014–2023. PMLR.
- [95] C. Parsing (2009). ‘Speech and language processing’. *Power Point Slides* .
- [96] J. Pennington, et al. (2014). ‘Glove: Global vectors for word representation’. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543.
- [97] D. Quercia, et al. (2012). ‘Tweetlda: supervised topic classification and link prediction in twitter’. In *Proceedings of the 4th Annual ACM Web Science Conference*, pp. 247–250.
- [98] A. Radford, et al. (2019). ‘Language models are unsupervised multitask learners’. *OpenAI Blog* **1**(8):9.

- [99] C. Raffel, et al. (2020). ‘Exploring the limits of transfer learning with a unified text-to-text transformer’. *The Journal of Machine Learning Research* **21**(1):5485–5551.
- [100] P. Rajpurkar, et al. (2016). ‘Squad: 100,000+ questions for machine comprehension of text’. *arXiv preprint arXiv:1606.05250* .
- [101] J. Ramos et al. (2003). ‘Using tf-idf to determine word relevance in document queries’. In *Proceedings of the First Instructional Conference on Machine Learning*, vol. 242, pp. 29–48. Citeseer.
- [102] A. Rehmer & A. Kroll (2020). ‘On the vanishing and exploding gradient problem in Gated Recurrent Units’. *IFAC-PapersOnLine* **53**(2):1243–1248.
- [103] Q. Ren, et al. (2024). ‘Automatic quality compliance checking in concrete dam construction: Integrating rule syntax parsing and semantic distance’. *Advanced Engineering Informatics* **60**:102409.
- [104] A. Saksida, et al. (2017). ‘Co-occurrence statistics as a language-dependent cue for speech segmentation’. *Developmental Science* **20**(3):e12390.
- [105] D. M. Salama & N. M. El-Gohary (2016). ‘Semantic text classification for supporting automated compliance checking in construction’. *Journal of Computing in Civil Engineering* **30**(1):04014106.
- [106] T. B. Shahi & A. K. Pant (2018). ‘Nepali news classification using naive bayes, support vector machines and neural networks’. In *2018 International Conference on Communication Information and Computing Technology (IC-CICT)*, pp. 1–5. IEEE.
- [107] P. Shaw, et al. (2018). ‘Self-attention with relative position representations’. *arXiv preprint arXiv:1803.02155* .
- [108] J. Straková, et al. (2019). ‘Neural architectures for nested NER through linearization’. *arXiv preprint arXiv:1908.06926* .

- [109] G. Sun, et al. (2020). ‘Representative task self-selection for flexible clustered lifelong learning’. *IEEE Transactions on Neural Networks and Learning Systems* .
- [110] M. Sun, et al. (2016). ‘Thuctc: an efficient chinese text classifier’. *GitHub Repository* .
- [111] Y. Sun, et al. (2019). ‘Ernie: Enhanced representation through knowledge integration’. *arXiv preprint arXiv:1904.09223* .
- [112] I. Sutskever, et al. (2014). ‘Sequence to sequence learning with neural networks’. *Advances in neural information processing systems* **27**.
- [113] H. Tang, et al. (2020a). ‘Hin: Hierarchical inference network for document-level relation extraction’. In *Advances in Knowledge Discovery and Data Mining: 24th Pacific-Asia Conference, PAKDD 2020, Singapore, May 11–14, 2020, Proceedings, Part I 24*, pp. 197–209. Springer.
- [114] H. Tang, et al. (2020b). ‘An integration model based on graph convolutional network for text classification’. *IEEE Access* **8**:148865–148876.
- [115] L. Q. Trieu, et al. (2017). ‘News classification from social media using twitter-based doc2vec model and automatic query expansion’. In *Proceedings of the Eighth International Symposium on Information and Communication Technology*, pp. 460–467.
- [116] H. Tseng, et al. (2005). ‘A conditional random field word segmenter for sighan bakeoff 2005’. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- [117] D. Ulyanov, et al. (2016). ‘Instance normalization: The missing ingredient for fast stylization’. *arXiv preprint arXiv:1607.08022* .
- [118] M. A. Valenzuela-Escarcega, et al. (2015). ‘Description of the Odin event extraction framework and rule language’. *arXiv preprint arXiv:1509.07513* .

- [119] A. Vaswani, et al. (2017). ‘Attention is all you need’. *Advances in Neural Information Processing Systems* **30**.
- [120] P. Vincent, et al. (2010). ‘Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.’. *Journal of Machine Learning Research* **11**(12).
- [121] Q. Wan, et al. (2023). ‘Joint document-level event extraction via token-token bidirectional event completed graph’. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10481–10492.
- [122] C. Wang, et al. (2016). ‘Text classification with heterogeneous information network kernels’. In *Thirtieth AAAI Conference on Artificial Intelligence*.
- [123] P. Wang, et al. (2015). ‘A unified tagging solution: Bidirectional lstm recurrent neural network with word embedding’. *arXiv preprint arXiv:1511.00215* .
- [124] S. I. Wang & C. D. Manning (2012). ‘Baselines and bigrams: Simple, good sentiment and topic classification’. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pp. 90–94.
- [125] X. Wang, et al. (2019). ‘Heterogeneous graph attention network’. In *The world wide web conference*, pp. 2022–2032.
- [126] X. Wei, et al. (2020). ‘Recurrent graph neural networks for text classification’. In *2020 IEEE 11th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 91–97. IEEE.
- [127] T. Wu, et al. (2023). ‘A brief overview of ChatGPT: The history, status quo and potential future development’. *IEEE/CAA Journal of Automatica Sinica* **10**(5):1122–1136.

- [128] Z. Wu, et al. (2020a). ‘On Scalability of Association-rule-based recommendation: A unified distributed-computing framework’. *ACM Transactions on the Web (TWEB)* **14**(3):1–21.
- [129] Z. Wu, et al. (2020b). ‘A comprehensive survey on graph neural networks’. *IEEE Transactions on Neural Networks and Learning Systems* **32**(1):4–24.
- [130] Z. Wu, et al. (2017). ‘Efficiently Translating Complex SQL Query to MapReduce Jobflow on Cloud’. *IEEE Transactions on Cloud Computing* **8**(2):508–517.
- [131] H. Yang, et al. (2021). ‘Document-level event extraction via parallel prediction networks’. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 6298–6308.
- [132] M. Yang, et al. (2023). ‘Semi-automatic representation of design code based on knowledge graph for automated compliance checking’. *Computers in Industry* **150**:103945.
- [133] Z. Yang, et al. (2016). ‘Hierarchical attention networks for document classification’. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1480–1489.
- [134] L. Yao, et al. (2019). ‘Graph convolutional networks for text classification’. In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 7370–7377.
- [135] D. Yu, et al. (2014). ‘Mixed pooling for convolutional neural networks’. In *Rough Sets and Knowledge Technology: 9th International Conference, RSKT 2014, Shanghai, China, October 24-26, 2014, Proceedings 9*, pp. 364–375. Springer.

- [136] J. Zeng, et al. (2018). ‘Topic memory networks for short text classification’. *arXiv preprint arXiv:1809.03664* .
- [137] C. Zhang, et al. (2017). ‘Collaborative user network embedding for social recommender systems’. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pp. 381–389. SIAM.
- [138] J. Zhang & N. M. El-Gohary (2015). ‘Automated information transformation for automated regulatory compliance checking in construction’. *Journal of Computing in Civil Engineering* **29**(4):B4015001.
- [139] X. Zhang, et al. (2015). ‘Character-level convolutional networks for text classification’. *Advances in Neural Information Processing Systems* **28**.
- [140] Y. Zhang, et al. (2021). ‘Semantic role labeling as dependency parsing: Exploring latent tree structures inside arguments’. *arXiv preprint arXiv:2110.06865* .
- [141] Z. Zhang, et al. (2024). ‘Found in the Middle: How Language Models Use Long Contexts Better via Plug-and-Play Positional Encoding’. *arXiv preprint arXiv:2403.04797* .
- [142] Z. Zhao, et al. (2017). ‘Ngram2vec: Learning improved word representations from ngram co-occurrence statistics’. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 244–253.
- [143] H. Zhong, et al. (2018). ‘Legal judgment prediction via topological learning’. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pp. 3540–3549.
- [144] J. Zhou, et al. (2018). ‘Graph neural networks: A review of methods and applications’. *arXiv preprint arXiv:1812.08434* .

- [145] L. Zhou & D. Zhang (2003). ‘NLPIR: A theoretical framework for applying natural language processing to information retrieval’. *Journal of the American Society for Information Science and Technology* **54**(2):115–123.
- [146] W. Zhou, et al. (2021). ‘GMNet: Graded-Feature Multilabel-Learning Network for RGB-Thermal Urban Scene Semantic Segmentation’. *IEEE Transactions on Image Processing*.

Every reasonable effort has been made to acknowledge the owners of copyright material. I would be pleased to hear from any copyright owner who has been omitted or incorrectly acknowledged.