Relation extraction from TACRED dataset based on deep learning

Bowen Jing Yang Cui Tianpeng Huang
University of Manchester
Department of Computer Science
{bowen.jing,yang.cui,tianpeng.huang}@postgrad.manchester.ac.uk

Abstract

Relation extraction (RE) is essential for structuring unstructured textual information for better data usability and accessibility. This study compares deep learning models with and without transformers for text entity pair classification. Through numerous tests, this paper examines the performance of various LSTM, GCN, LSTM-GCN, BERT, RoBERTa, and Span-BERT models. These include model evaluations, few-shot learning scenarios, and analyses considering variations in sentence length. Our study on the TACRED dataset[1] shows that transformer-based models outperform non-transformer models with Micro F1 scores of 80-90%, compared to 55-67% for non-transformers. This distinction underscores the advanced ability of transformer models to grasp contextual nuances within text.

1 Introduction

Relation extraction(RE)[2], a crucial component of information extraction(IE)[3], garners significant attention for its extensive applications in question answering[4], biomedical knowledge discorery[5] and knowledge base population[1]. The RE process generally encompasses two stages: relation identification(RI) and relation classification(RC)[6]. We specifically concentrates on RC within the context of the TACRED dataset[1], targeting the task of sentence-level binary relation extraction. As a vital task within RE, RC aims to identify relationships between pairs of entities in textual data, thereby converting unstructured facts into structured triplet formats (entity1, relation, entity2). For instance, given the sentence "Tesla was founded by Elon Musk" and the entity pair ("Tesla", "Elon Musk"), an RE model is capable of identifying the ORG:FOUNDBY relationship. This classification step is essential for structuring data in a way that enhances the usability and accessibility of information extracted from text.

2 Literature Review

Deep learning without transformer Unlike rule-based and SVM methos, deep learning approaches for relationship extraction can automatically learn complex features from data, offering greater adaptability[7] and reducing the need for manual feature engineering and pre-defined domain-specific knowledge[8]. CNNs mark a significant shift towards learning hierarchical representations of text data. Pioneering work, such as that by Zeng et al. (2014), demonstrated the advantages of Convolutional Neural Networks (CNNs) in capturing lexical and sentence-level features associated with entities within sentences, thereby reducing the necessity for extensive feature engineering.[9]. Following these early efforts, there has been an emergence of studies focusing on CNN derivatives, such as the Multi-window CNN (Nguyen, 2015)[10] and the convolutional neural network that performs classification by ranking (CR-CNN) as introduced by Santos et al.(2015)[11]. These adaptations leverage the foundational principles of CNN architectures, thereby extending their applicability and enhancing performance within the domain of relation extraction.

Sequence models(such as LSTM) gains attention following the success of CNNs. Zhang etal.(2015)[12] highlighted the shortcomings of CNNs—- point out CNN models lack the ability to learn temporal features and their difficulties in capturing long-term dependencies. Zhou et al.(2016) advanced this field by integrating attention mechanisms within bidirectional LSTM[13]. Building on the success of sequence models, the Position-aware LSTM model introduced by Zhang et al. integrates position information into LSTM[1]. Furthermore, the SDP-LSTM model shifts focus from the entire sentence to the shortest denpendency path between entities[14].

Inspired by sequence models, Graph Convolutional Networks (GCNs) revolutionize relation extraction (RE) by efficiently encoding dependency-parsed sentences and enabling parallel computation. Originally implemented in the RE domain by Zhang et al.(2018)[15], the innovative C-GCN model combines a bidirectional LSTM for sequence contextualization with a GCN layer for dependency structure encoding. This model employs a distinctive pruning strategy, enhancing its effectiveness. Following this, Zhou et al.(2020) introduced an enhanced C-GCN model utilizing a distance-weighted adjacency matrix, which more accurately represents the dependency tree's structure, and integrates entity attention along with network depth to improve analytical capabilities[16]. Meanwhile, despite challenges in concentrating on SDP's critical nodes, Guo et al.(2019) have advanced the field by pioneering the use of dynamic matrices. This approach enables a more adaptable analysis of the dependency tree through a multi-head self-attention mechanism[17].

Deep learning with transformer The introduction of the transformer model by Vaswani et al. in 2017 marked a revolutionary shift in the domain of natural language processing (NLP)[18]. Its self-attention mechanism not only addresses the challenge of long-range dependencies inherent in sequence-based models such as RNNs and LSTMs but also boasts lower computational complexity for sequences shorter than the embedding size. Crucially, the computation of the attention layers can be performed in parallel, making the training of large language models more feasible and efficient.

In 2018, Devlin et al. introduced the Bidirectional Encoder Representations from Transformers (BERT)[19], a new language model pre-trained using a bidirectional masked language model and next sentence prediction tasks on a large corpus. This pre-training process significantly enhances BERT's ability to understand context. Soares et al. demonstrated that fine-tuning BERT leads to substantial improvements in performance over previous models in tasks such as relation extraction, highlighting the model's advanced understanding of language nuances[20].

Building on BERT's success, Liu et al. developed an optimised version known as RoBERTa (Robustly Optimized BERT Approach)[21]. This new approach incorporates dynamic masking, larger batch sizes, etc., enabling RoBERTa to outperform BERT in a variety of downstream tasks.

Joshi et al. introduced Span-BERT, a variation of BERT that shifts the focus from individual tokens to spans of text[22]. This modification allows Span-BERT to better performance in tasks requiring an understanding of text spans, such as extractive question answering and relation extraction.

3 Methodology

3.1 Data Process

Deep learning without transformer models In the prepossessing pipeline, we first normalize the raw tokens by converting them to lowercase to ensure uniformity. Next, we replace the entities names with their entity types, such as 'SUBJ-ORG' for orgnazitions. This helps in abstracting out specific information and allowing the model to focus on the relationship between entities. Furthermore, we map all tokens including the replaced entities to ID according to our predefined dictionary.

For integrating the GloVe embeddings[23], we normalize tokens to align with GloVe's format, for example, converting '(' to '-LRB-'. we construct our vocabulary by combining tokens from our dataset with those found in GloVe, applying a frequency threshold to filter out rare words. This amalgamated vocabulary includes both special tokens and entity mask tokens, the latter derived from the named entity recognition (NER) tags provided by the TACRED dataset, such as 'SUBJ-PERSON' for personal entities. In the end, we construct word embeddings by mapping out vocabulary to pre-trained Glove vectors, initialzing with random value where no match is found.

Deep learning with transformer models In the Transformers-based model, we combine text and two entities as input to the model. Usually, to introduce entities in the model, we replace two entities in the text with special placeholders, usually [MASK]. The purpose of this is to allow the model to be aware of these entities and process them without directly exposing the original entity name to the model.[24]

3.2 Experiment Setting

In this project, we have conducted several experiments to compare the distinct capabilities of each model in relation extraction on the TACRED database. Initially, we utilize the complete TACRED training dataset to determine the optimal hyperparameters for each model and their performance on the full dataset. Subsequently, we evaluate these models on test datasets of varying sentence lengths[1]. This approach allows us to assess the models' performance across different text lengths. Finally, we apply a few-shot learning approach, training the models on 20%, 40%, 60%, and 80% of the training dataset. This is to measure the models' performance with limited data, which may indicate how well each model adapts to different downstream tasks[25]. Before conducting the experiments, we perform hyperparameter optimization on all experimental models using the grid search method to identify the optimal combination of hyperparameters. This process enhances the models' performance and generalization capabilities.

3.3 Model Introduction

Deep learning without transformer models Based on our investigation of related works, we decide to adopt the combination stracture of att-BiLSTM-GCN for solving the problem without using transformer. Not only this approach use the temporal models by capturing the forward and backward contextual information of sentences, but also it utilizes the graph convolutional network(GCN) to capture the syntactic dependency tree of sentences, thereby understanding the relation between tokens and the overall structure of sentences.

The model input consists of tokens, POS, NER and dependency relations. Additionally, it processes the positions of two entities within the sentence. BiLSTM is employed to process those data to capture the sequence information. GCN aims to exploit the structured information in text, especially the dependencies between words. A multi-head attention mechanism is used to calculate the attention weights corresponding to different heads, and this information is used to enhance or weaken the feature representation of certain nodes. A max pooling strategy is used to compress sentence-level features into a fixed-size vector. In this end, the fully connection neural network with softmax function is used for relation classification.

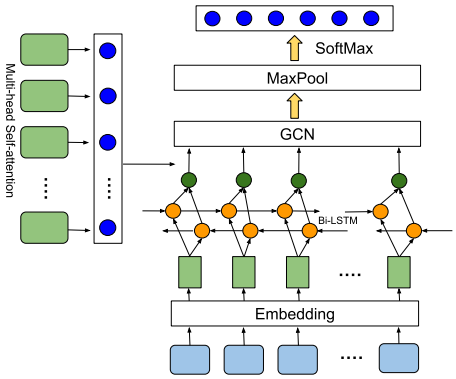

Figure 1: Att-BiLSTM-GCN Architecture

Deep learning with transformer models BERT uses transformers to understand word relationships in context. It was trained on a large corpus, allowing it to perform well across various NLP tasks. BERT operates in two phases: pre-training and fine-tuning. During pre-training, it uses tasks like Masked Language Modeling (MLM), where words are hidden and the model guesses them based on context, and Next Sentence Prediction (NSP), which helps it understand sentence relationships[19]. These tasks, especially NSP, are key to BERT's effectiveness in tasks like sentiment analysis and question answering.

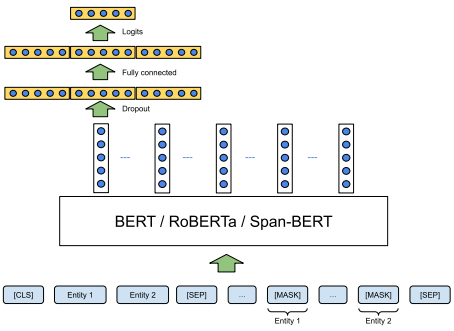

Figure 2: Attention-based model Architecture

In our methodology shown in Figure 2, we employ a transformer-based model architecture that use the power of pre-trained language models such as BERT, RoBERTa, and Span-BERT for encoding textual inputs. The initial step in our approach involves masking entities within sentences using the [MASK] token. Subsequently, we utilize the [SEP] token to separate the entity names from the sentence itself. These components, along with an attention mask, are then input into the transformer-based pre-trained language model to facilitate the encoding of tokens.

Following the encoding process, the model generates an output from the first hidden lay, which essentially represents the embedding for the fist token [CLS]. To further process the output, we introduce a dropout layer to mitigate the risk of overfitting. This is followed by a fully connected layer that derived embeddings to our target labels.

For the task of label prediction, we adopt the cross-entropy loss function. This is a in multi-class classification tasks, where it excels in quantifying the difference between the predicted probability distribution and the actual distribution of the labels. By minimizing this loss function, our model is trained to accurately predict the relationship labels, thereby enhancing its performance on the relation extraction task.

4 Result Analysis

4.1 Deep Learning without Transformer Method

Reference	P	R	F1
LR[1]	73.5	49.9	59.4
SDP-LSTM[14]	66.3	52.7	58.7
Tree-LSTM[26]	66.0	59.2	62.4
PA-LSTM[1]	65.7	**64.5**	65.1
GCN[15]	69.8	59.0	64.0
C-GCN[15]	69.9	63.3	66.4
AGGCN[17]	69.9	60.9	65.1
C-AGGCN[17]	73.1	64.2	**69.0**
Ours	P	R	F1
LSTM	64.37	61.30	61.19
LSTM-entity aware	66.51	61.73	62.73
PA-LSTM	67.42	62.50	64.87
GCN	68.35	58.92	63.26
att-GCN	66.50	60.66	63.448
BiLSTMGCN	**73.58**	60.90	66.66
att-BiLSTM-GCN	73.36	62.47	67.48

Table 1: (micro)F1 score of different deep learning without transformer model train with full training dataset of TACRED and evaluate with different test dataset. N.B some of our models I use the official code to reproduce the results

Figure 3: Deep learning without transformer model performance with few-shot ap- proach

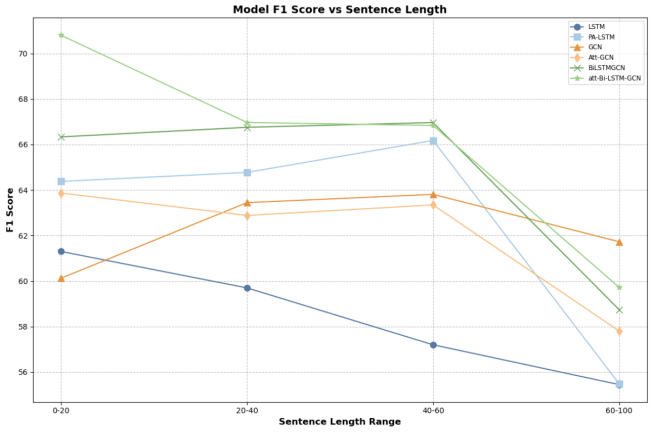

Figure 4: deep learning without transformer evaluations on difference sen- tence length

From the table, we can observe that Zhang et al.'s implementation of C-AGGCN achieved the highest F1 score. Temporal models like LSTM, with various enhancements, achieved differing results. However, models incorporating GCN generally outperformed conventional temporal models in terms of F1 scores. Moreover, from our few-shot experiments, we can see that as the number of training samples decreases, the performance of the models deteriorates. From experiments across different sentence lengths, it is evident that the performance of a unidirectional LSTM without any enhancements significantly worsens as sentence length increases. Models that incorporate temporal mechanisms (such as PA-LSTM, BiLSTM-GCN, Att-BiLSTM-GCN), which rely on temporal models to process temporal information, show varying degrees of performance decline with the increase in sentence length. Models that rely solely on GCN are also affected, but not as significantly as the temporal models.

4.2 Deep Learning with Transformer Method

Model	**TACRED**	**TACREV**
BERT	82.92	85.49
RoBERTa	**87.39**	**90.89**
Span-BERT	85.75	89.10
BERT(entity mask)	86.76	-
RoBERTa(entity mask)	**87.44**	-
Span-BERT(entity mask)	84.56	-

Table 2: (micro)F1 score of different transformer-based model train with full training dataset of TACRED and evaluate with different test dataset

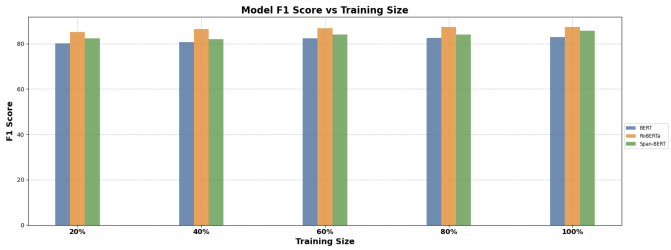

Figure 5: Transformer-based model performance with few-shot approach

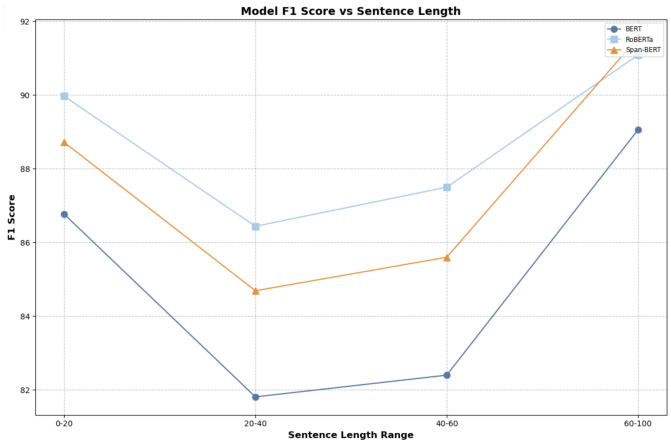

Figure 6: Transformer-based model evaluations on difference sentence length

In this study, we conducted two sets of experiments to evaluate the performance of a transformer-based model in relation extraction tasks. In the first experiment, we utilized a single [MASK] token to obscure each entity within a sentence. These masked sentences, along with the two entities, were then input into the model to predict the relationship between the entities. The second experiment involved using sentences without entity masking. Additionally, we assessed the model's performance on the TACREV (Revised TACRED) dataset, which is a more stringent and well-structured testing dataset compared to its predecessor[27].

Our experiments demonstrated that the transformer-based models performed exceptionally well, as evidenced by the results presented in Table 2. Notably, RoBERTa achieved an F1 score of 90.89 on the TACREV dataset, outperforming other models across all evaluation tests. Both RoBERTa and Span-BERT exhibited superior performance compared to the BERT model. This discrepancy can be attributed to Span-BERT's focus on span-level information, which is more relevant for entities typically appearing as spans within

text[22]. Furthermore, RoBERTa's enhanced pre-training and optimization processes contribute to its improved performance in word embedding and relation classification tasks[21]. Contrary to previous studies that suggested Span-BERT outperforms RoBERTa in relation extraction tasks[22], our findings did not support this. The deviation may be due to limited time and resources allocated for hyperparameter tuning, suggesting that the Span-BERT model might not have been optimized in our experiments. Moreover, our results generally surpassed those of previous BERT-based relation extraction approaches. This improvement is likely due to our novel structure and the incorporation of a MASK-Attention mechanism during training, which enhances the model's suitability for the task. Additionally, masking entity words prior to the tokenisation process likely simplified the task for the models.

In the few-shot learning scenario, as depicted in Figure 5, the three models exhibited similar performance levels, demonstrating effectiveness with only 20% of the training data. This suggests that the substantial size of the training dataset ensures that even a reduced subset is sufficient for the models to capture relevant features. Furthermore, performance improvements were observed as the size of the training set increased.

The third experiment, illustrated in Figure 6, investigated model performance across different sentence or sequence lengths. The results indicated that all three models performed well for sentence lengths of 0-20 and 60-100. Shorter sentences (0-20 length) likely presented less complexity, facilitating easier classification. For sentences within the 60-100 length range, the models benefited from fewer empty token inputs due to the maximum token limit of 128 set for our BERT model. Surprisingly, Span-BERT outperformed both RoBERTa and BERT in processing longer sentences (60-100 length), possibly due to its enhanced pre-training on longer sequences[22].

5 Discussion

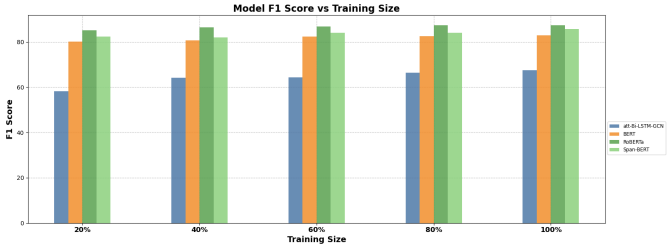

Figure 7: External comparison on Few-shot Approach

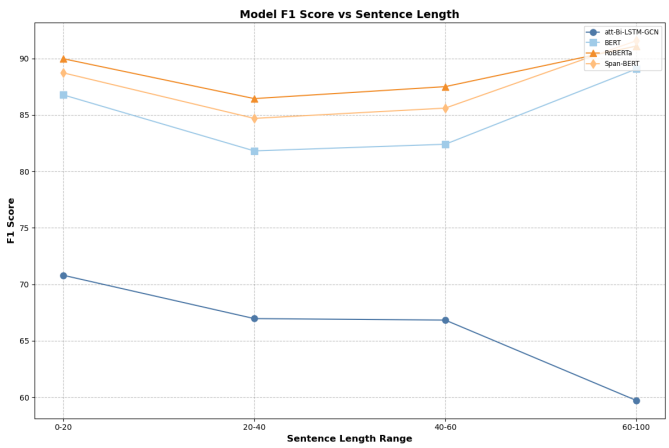

Figure 8: External comparison on Evaluations on Difference Sentence Length

Comparasion In this section, we conduct an external comparison between transformer-based models and non-transformer-based models. Figure 7 illustrates the performance of both an attention-based Bi-LSTM-GCN model and a transformer-based model in a few-shot learning scenario. The results unequivocally demonstrate that the transformer-based model outperforms the attention-based Bi-LSTM-GCN model in few-shot learning tasks. This advantage can be attributed to the pre-training phase of transformer models like BERT, which effectively captures word embeddings. Consequently, only minimal fine-tuning is required to adapt the transformer model to specific downstream tasks, resulting in significantly improved performance[28].

Figure 8 presents an external comparison across different sentence length ranges. Both the attention-based Bi-LSTM-GCN and the transformer-based models exhibit a decline in performance as sentence length increases from 0-20 to 20-40. However, a notable divergence occurs for sentence lengths ranging from 40-60 to 60-100. In this range, the performance of the transformer-based model increases, whereas that of the attention-based Bi-LSTM-GCN model decreases. This observation underscores the transformer architecture's superior capability in processing longer input sequences[29].

Additionally, Figure 9 in the appendix details the data distribution according to sentence length, revealing a lack of data within the 60-100 sentence length range. This further highlights the transformer-based pre-trained model's remarkable proficiency in few-shot learning scenarios, even with limited data availability.

Limitation There are several limitations about our experiments.

- our experiments may not fully capture the models' capabilities, as zero-shot learning for transformer-based models was not explored, and cross-dataset evaluations were omitted. These could have offered insights into generalization and adaptability.
- differences in hardware among team members caused inconsistencies in training conditions, affecting uniform evaluation of training processes.

6 Conclusion

We conducted a thorough research into RE approaches, comparing cutting-edge transformer-based models to standard deep learning methods using the TACRED dataset. Our research required us to analyse models such as LSTM, GCN, BERT, RoBERTa, and Span-BERT. After doing a thorough investigation, we discovered that transformer-based models, notably RoBERTa, outperform traditional models in terms of performance indicators. Transformer models performed significantly better in Micro F1 scores, ranging from 80% to 90%, compared to non-transformer models such as LSTM and GCN, which scored between 55% and 67%.

Furthermore, our analysis of few-shot learning scenarios shown how adaptive and efficient transformer models are, even in data-poor environments. This study broadens the applicability of these sophisticated models, demonstrating how they can be utilised to change RE activities in a variety of settings. Although this study establishes a standard for future research and considerably increases our understanding of the utility of RE models, it also recognises the need for additional research. It is especially recommended that these models be investigated for their cross-dataset generalizability and zero-shot learning capabilities. Our findings lay a solid foundation for present and future RE research, emphasising the critical need for more robust, generalised, and effective models capable of navigating the complexity of natural language with hitherto unheard-of precision and efficacy.

References

[1] Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D Manning. Position-aware attention and supervised data improve slot filling. In *Conference on Empirical Methods in Natural Language Processing*, 2017.

[2] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid O Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *arXiv preprint arXiv:1911.10422*, 2019.

[3] Claire Cardie. Empirical methods in information extraction. *AI magazine*, 18(4):65–65, 1997.

[4] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957*, 2016.

[5] Chris Quirk and Hoifung Poon. Distant supervision for relation extraction beyond the sentence boundary. *arXiv preprint arXiv:1609.04873*, 2016.

[6] Elisa Bassignana and Barbara Plank. What do you mean by relation extraction? a survey on datasets and study on scientific relation classification. *arXiv preprint arXiv:2204.13516*, 2022.

[7] Shantanu Kumar. A survey of deep learning methods for relation extraction. *arXiv preprint arXiv:1705.03645*, 2017.

[8] Sachin Pawar, Girish K Palshikar, and Pushpak Bhattacharyya. Relation extraction: A survey. *arXiv preprint arXiv:1712.05191*, 2017.

[9] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: technical papers*, pages 2335–2344, 2014.

[10] Thien Huu Nguyen and Ralph Grishman. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st workshop on vector space modeling for natural language processing*, pages 39–48, 2015.

[11] Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. Classifying relations by ranking with convolutional neural networks. *arXiv preprint arXiv:1504.06580*, 2015.

[12] Dongxu Zhang and Dong Wang. Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*, 2015.

[13] Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, pages 207–212, 2016.

[14] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 1785–1794, 2015.

[15] Yuhao Zhang, Peng Qi, and Christopher D Manning. Graph convolution over pruned dependency trees improves relation extraction. *arXiv preprint arXiv:1809.10185*, 2018.

[16] Li Zhou, Tingyu Wang, Hong Qu, Li Huang, and Yuguo Liu. A weighted gcn with logical adjacency matrix for relation extraction. In *ECAI 2020*, pages 2314–2321. IOS Press, 2020.

[17] Zhijiang Guo, Yan Zhang, and Wei Lu. Attention guided graph convolutional networks for relation extraction. *arXiv preprint arXiv:1906.07510*, 2019.

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[20] Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*, 2019.

[21] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[22] Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the association for computational linguistics*, 8:64–77, 2020.

[23] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[24] Yuanhe Tian, Yan Song, and Fei Xia. Improving relation extraction through syntax-induced pre-training with dependency masking. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1875–1886, 2022.

[25] Sam Brody, Sichao Wu, and Adrian Benton. Towards realistic few-shot relation extraction. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5338–5345, 2021.

[26] Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

[27] Christoph Alt, Aleksandra Gabryszak, and Leonhard Hennig. Tacred revisited: A thorough evaluation of the tacred relation extraction task. *arXiv preprint arXiv:2004.14855*, 2020.

[28] Bei Hui, Liang Liu, Jia Chen, Xue Zhou, and Yuhui Nian. Few-shot relation classification by context attention-based prototypical networks with bert. *EURASIP Journal on Wireless Communications and Networking*, 2020:1–17, 2020.

[29] Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in neural information processing systems*, 33:17283–17297, 2020.

[30] Mohammad Hossin and Md Nasir Sulaiman. A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1, 2015.

Appendix

a. Abbreviation List

Abbr	Description
BERT	Bidirectional Encoder Representations from Transformers
RoBERTa	Robustly Optimized BERT Pretraining Approach
Span-BERT	Span Bidirectional Encoder Representations from Transformers

Table 3: Models based on Transformer architecture Abbreviation

Abbr	Description
SVM	Support Vector Machine
CNN	Convolutional Neural Network
LLM	Language Learning Model
LSTM	Long Short-Term Memory
BiLSTM	Bidirectional Long Short-Term Memory
att-BiLSTM	Attention-Based Bidirectional Long Short-Term Memory
PA-LSTM	Position Aware Long Short-Term Memory
SDP-LSTM	Shortest Dependency Path Long Short-Term Memory
GCN	Graph Convolutional Network

Table 4: Models without Transformer architecture Abbreviation

Abbr	Description
RE	Relation Extraction
NER	Named Entity Recognition
RI	Relation Identification
RC	Reading Comprehension
TACRED	The TAC Relation Extraction Dataset

Table 5: NLP Tasks and Datasets Abbreviation

b. Evaluation Metrics

Confusion Matrix The confusion matrix is a performance measurement tool for classification models. It is particularly useful in understanding the performance of a model by visualizing the accuracy of predictions for each class.

True Positives (TP) are the cases where the model correctly predicts the positive class. False Positives (FP) are the cases where the model incorrectly predicts the positive class when it's actually negative. False Negatives (FN) are the cases where the model incorrectly predicts the negative class when it's actually positive. True Negatives (TN) are the cases where the model correctly predicts the negative class.

Precision Precision is the proportion of correctly predicted positive instances to all predicted positive samples. In the task of relation extraction, Precision indicates how many predicted relations by the model are correct, that is, among the entity pairs predicted as relations, how many truly represent the relationships in the text [30]:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall Recall is the proportion of correctly predicted positive instances to all true positive samples. In the task of relation extraction, Recall represents the model's ability to correctly identify all true relationships in the text. It measures the extent to which the model covers true relationships, i.e., how many true relationships the model can recognize [30]:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

Mirco F1 In this project, we chose Micro F1 as the comprehensive evaluation standard, and there are many reasons behind it. First, the TACRED dataset suffers from class imbalance, that is, the number of samples in some classes far exceeds that of other classes. Micro F1 effectively addresses this challenge because it considers performance across all categories and does not ignore certain categories because they have a low number of samples.[30]:

$$\text{Micro F1} = \frac{2 \times \text{Micro Precision} \times \text{Micro Recall}}{\text{Micro Precision} + \text{Micro Recall}}$$

c. Dataset Details

TACRED is a large-scale relation extraction dataset with 106,264 examples built over newswire and web text from the corpus used in the yearly TAC Knowledge Base Population (TAC KBP) challenges.

Data Split	# Ex.	Years
Train	75,050	2009–2012
Dev	25,764	2013
Test	18,660	2014

Table 6: Tacred Dataset

Dataset	# Rel.	# Ex.	% Neg.
SemEval-2010 Task 8	19	10,717	17.4%
ACE 2003–2004	24	16,771	N/A
TACRED	42	119,474	78.7%

Table 7: RE dataset comparison

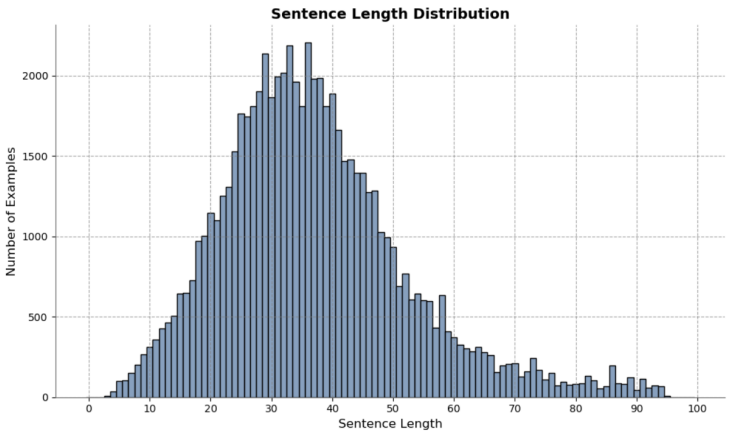

Figure 9: TACRED Sentence Length Distribution

Table 7 shows that the TACRED dataset stands out for its size and complexity in the field of relation extraction, with 42 different relation categories (compared to 19 for SemEval-2010 Task 8 and 24 for ACE 2003–2004) and up to 119,474 examples, the largest dataset in

the comparison. More significantly, its 78.7% negative example ratio provides an extremely challenging environment for training machine learning models to recognize and handle irrelevant instances. These features not only provide depth and breadth for model training, but also provide researchers with a rich resource to test and improve their algorithms.

Figure 9 shows the distribution of sentence lengths in the TACRED dataset. Sentence length is calculated as the number of words, distributed from 0 to 100. As can be seen from the figure, the sentence length distribution is close to a normal distribution, with the length of most sentences concentrated between 20 and 30 words. The peak of the distribution seems to be around 25 words. There are relatively few examples of sentence lengths less than 10 words or greater than 50 words, indicating that sentences in the dataset are mostly of medium length. Very few sentences are longer than 80 words. Such a distribution may indicate that the data set has been filtered to exclude those sentences that are too short or too long, thus ensuring moderate and consistent sentence length, which is crucial to ensure the quality of the relationship extraction task.

Relation	Train	Test
org:founded_by	124	68
no_relation	55112	12184
per:employee_of	1524	264
org:alternate_names	808	213
per:cities_of_residence	374	189
per:children	211	37
per:title	2443	500
per:siblings	165	55
per:religion	53	47
per:age	390	200
org:website	111	26
per:stateorprovinces_of_residence	331	81
org:member_of	122	18
org:top_members/employees	1890	346
per:countries_of_residence	445	148
org:city_of_headquarters	382	82
org:members	170	31
org:country_of_headquarters	468	108
per:spouse	258	66
org:stateorprovince_of_headquarters	229	51
org:number_of_employees/members	75	19
org:parents	286	62
org:subsidiaries	296	44
per:origin	325	132
org:political/religious_affiliation	105	10
per:other_family	179	60
per:stateorprovince_of_birth	38	8
org:dissolved	23	2
per:date_of_death	134	54
org:shareholders	76	13
per:alternate_names	104	11
per:parents	152	88
per:schools_attended	149	30
per:cause_of_death	117	52
per:city_of_death	81	28
per:stateorprovince_of_death	49	14
org:founded	91	37
per:country_of_birth	28	5
per:date_of_birth	63	9
per:city_of_birth	65	5
per:charges	72	103
per:country_of_death	6	9

Table 8: TACRED data set-label details

The TACRED dataset is an extensive relationship extraction resource that covers rich relationship types from personal information (such as employee relationships and city of residence) to organizational details (such as establishment information and headquarters location). It shows significant class distribution imbalance, especially the 'no_relation' label with up to 55,112 instances in the training set and 12,184 instances in the test set, far more than any other category. This imbalance reflects what is seen in real-world data and presents a challenge for relational classification models to identify non-relational instances.

The composition of the dataset demonstrates deep coverage of fine-grained relationships, such as categories such as 'per:title' and 'org:top_members/employees', with a large number of instances in both training and test sets, which is useful for training highly accurate and detailed Models are very useful. At the same time, some less common categories, such as 'per:religion' and 'org:dissolved', although the number of instances is smaller, their inclusion ensures the sensitivity of the model to a few categories.

The design of this dataset reflects the complexity of the real world, providing a clear division of training and test sets with varying numbers of instances for each relationship category, which is beneficial for evaluating the model's generalization ability in real-world applications. This construction of the dataset is designed to facilitate detailed and comprehensive learning of the model over a wide range of relationship types, as well as the adaptability and accuracy shown in handling rare and common relationship types.

The three bar graphs in Figures 10, 11, and 12 respectively represent the sample distribution of various relationship labels in the TACRED data set, which is divided into training, development (dev) and test (test) data sets.

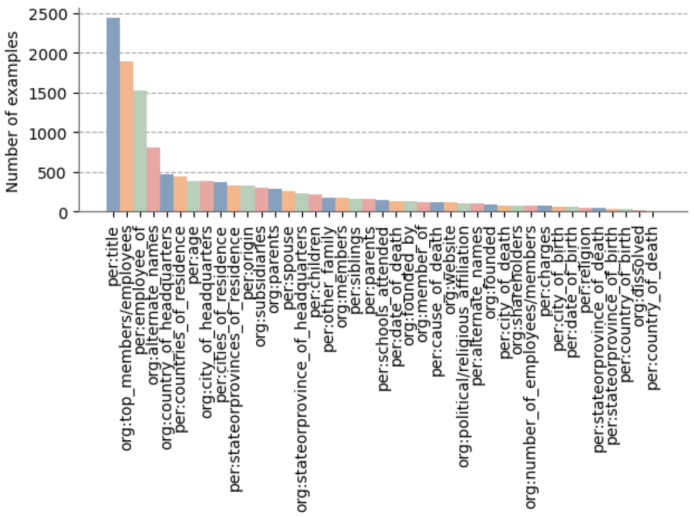

Figure 10: TACRED-train data label details.

The training data graph shows a highly imbalanced distribution of examples across different relation types. The 'per:title' relation has the highest number of examples, with a count significantly higher than other types, followed by relations like 'org:top_members/employees' and 'per:employee_of', which also have a high count. On the other end of the spectrum, relations like 'org:political/religious_affiliation', 'org:dissolved', and 'per:country_of_death' have the fewest examples, indicating that they are less common or perhaps more difficult to find in the data used for training.

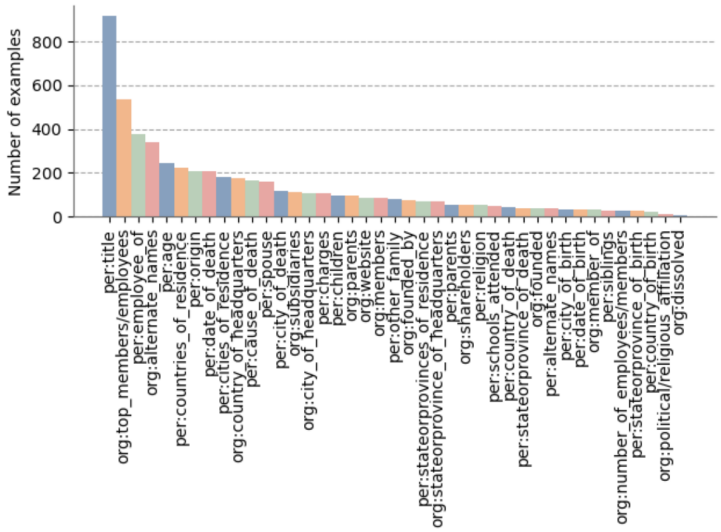

Figure 11: TACRED-dev data label details.

The development data set shows a similar pattern to the training set with 'per:title' having the highest count, which suggests consistency in the distribution of examples across these datasets. However, the overall numbers are lower because development datasets are typically smaller than training datasets. The distribution remains imbalanced, with the same minority classes having fewer examples as seen in the training set.

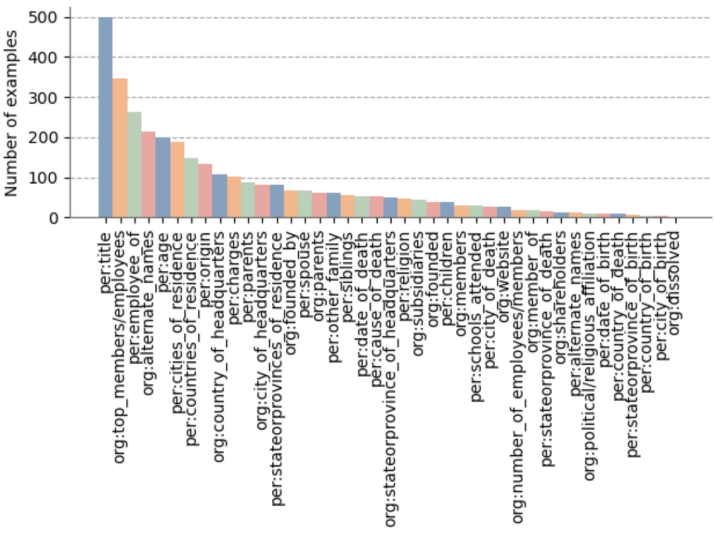

Figure 12: TACRED-test data label details.

The test data graph also shows an imbalanced distribution with 'per:title' having the most examples. This is consistent with the training and development sets, which indicates the testing set is structured to reflect the training data's distribution, ensuring that models are evaluated against a similar spread of relation types.

In all three datasets, the vast majority of relation types have a moderate to low number of examples, which is typical for specialized datasets like TACRED where certain types of relations are less common. This poses a challenge for models to perform well on these less-represented relation types, which can be critical for certain applications. The consistency across the training, development, and testing sets suggests that any machine learning models trained on this data will have to handle both common and rare relation types, simulating the variety and imbalances they will encounter in real-world scenarios.

d. Hyperparameter Optimisation

In deep learning projects, a model's performance is significantly influenced by hyperparameters (Feurer et al., 2019). By fine-tuning these hyperparameters, we can effectively improve the model's performance, tailoring it to the specific characteristics of a given dataset. To achieve this, we employ the Grid Search method, a powerful technique for exploring optimal hyperparameter combinations. Grid Search exhaustively evaluates various parameter values within predefined ranges, systematically assessing each combination to identify the most effective one. While this method incurs higher computational costs, it consistently yields superior hyperparameter configurations, ultimately enhancing the model's performance and its ability to generalize to unseen data.

Model	Learning Rate (lr)	Batch Size	Epochs
BERT	0.002	16	20
RoBERTa	0.002	16	15
Span-BERT	0.02	32	25

Table 9: Training Parameters for Models

e. Experiment Details and Setup

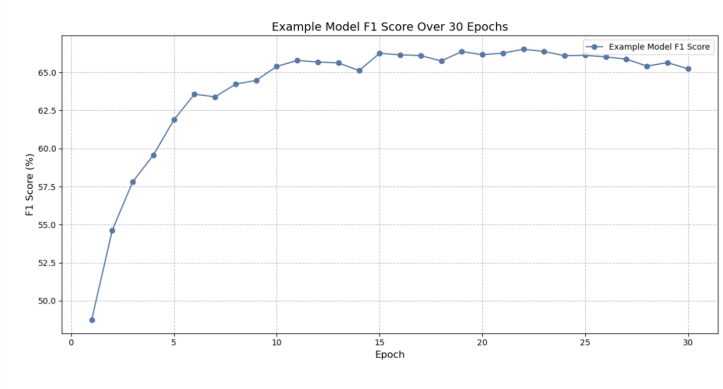

Figure 13: Training process1

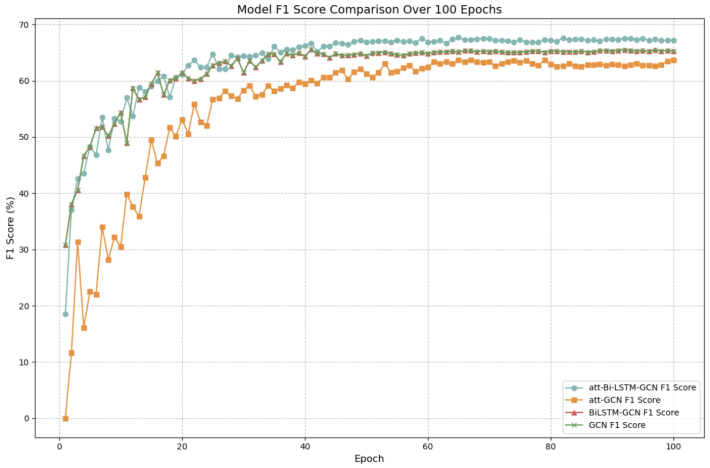

Figure 14: Training process2

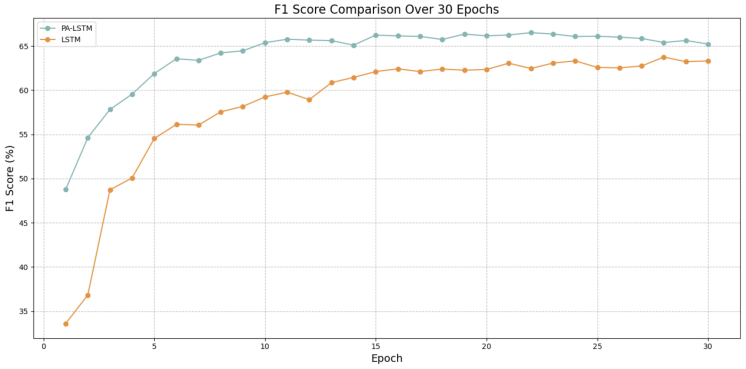

Figure 15: Training process3

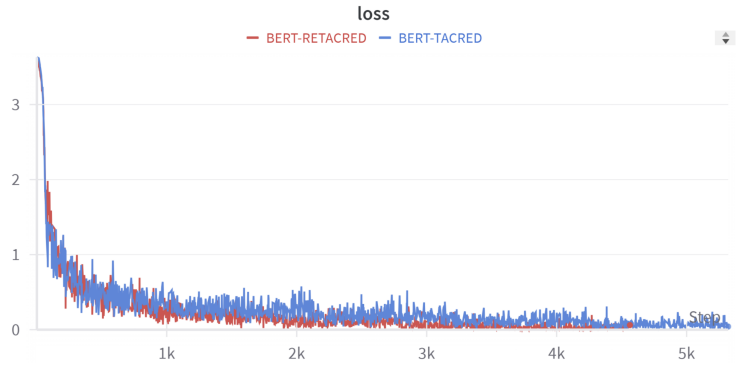

Figure 16: bert loss

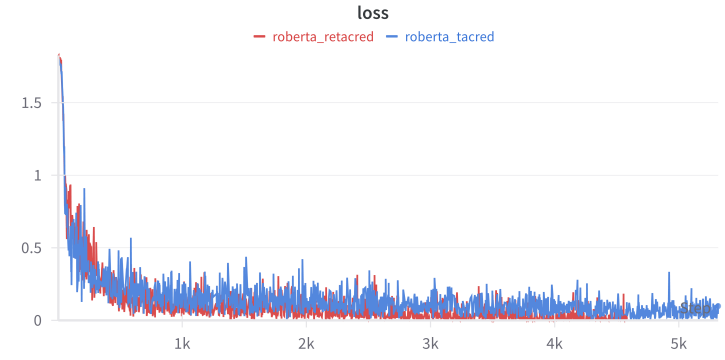

Figure 17: roberta loss

e. Math

A. Embedding Layer

For each word w_i in the sentence, the vector representation is obtained through the embedding matrix E_w:

$$e_{w_i} = E_w[w_i] \tag{1}$$

where E_w is the word embedding matrix, and w_i is the index of the word.

For each word's part-of-speech tag p_i, its vector representation is obtained through the embedding matrix E_p:

$$e_{p_i} = E_p[p_i] \tag{2}$$

where E_p is the POS tag embedding matrix, and p_i is the index of the POS tag.

For each word's named entity tag n_i, its vector representation is obtained through the embedding matrix E_n:

$$e_{n_i} = E_n[n_i] \tag{3}$$

where E_n is the named entity tag embedding matrix, and n_i is the index of the NER tag.

The final feature vector e_i for each word is the concatenation of these embeddings:

$$e_i = [e_{w_i}; e_{p_i}; e_{n_i}] \tag{4}$$

where e_i is the final embedding representation for the i-th word.

B. Bidirectional Long Short-Term Memory Network (BiLSTM)

The integrated features e_i from the embedding layer serve as inputs to the BiLSTM. The BiLSTM comprises two LSTMs: one processing the data from the start to the end of the sentence (forward pass) and the other from the end to the start (backward pass). This structure enables the network to capture information from both past and future contexts at each time step.

For the forward LSTM at each time step t, the following operations are performed:

1. The forget gate:

$$f_t^{(\text{forward})} = \sigma(W_f^{(\text{forward})} \cdot [h_{t-1}^{(\text{forward})}, e_t] + b_f^{(\text{forward})}) \tag{5}$$

2. The input gate:

$$i_t^{(\text{forward})} = \sigma(W_i^{(\text{forward})} \cdot [h_{t-1}^{(\text{forward})}, e_t] + b_i^{(\text{forward})}) \tag{6}$$

$$\tilde{C}_t^{(\text{forward})} = \tanh(W_C^{(\text{forward})} \cdot [h_{t-1}^{(\text{forward})}, e_t] + b_C^{(\text{forward})}) \tag{7}$$

3. The cell state update:

$$C_t^{(\text{forward})} = f_t^{(\text{forward})} * C_{t-1}^{(\text{forward})} + i_t^{(\text{forward})} * \tilde{C}_t^{(\text{forward})} \tag{8}$$

4. The output gate:

$$o_t^{(\text{forward})} = \sigma(W_o^{(\text{forward})} \cdot [h_{t-1}^{(\text{forward})}, e_t] + b_o^{(\text{forward})}) \tag{9}$$

$$h_t^{(\text{forward})} = o_t^{(\text{forward})} * \tanh(C_t^{(\text{forward})}) \tag{10}$$

Similarly, for the backward LSTM, at each time step t, we have:

1. The forget gate:

$$f_t^{(\text{backward})} = \sigma(W_f^{(\text{backward})} \cdot [h_{t+1}^{(\text{backward})}, e_t] + b_f^{(\text{backward})}) \tag{11}$$

2. The input gate:

$$i_t^{(\text{backward})} = \sigma(W_i^{(\text{backward})} \cdot [h_{t+1}^{(\text{backward})}, e_t] + b_i^{(\text{backward})}) \tag{12}$$

$$\tilde{C}_t^{(\text{backward})} = \tanh(W_C^{(\text{backward})} \cdot [h_{t+1}^{(\text{backward})}, e_t] + b_C^{(\text{backward})}) \tag{13}$$

3. The cell state update:

$$C_t^{(\text{backward})} = f_t^{(\text{backward})} * C_{t+1}^{(\text{backward})} + i_t^{(\text{backward})} * \tilde{C}_t^{(\text{backward})} \tag{14}$$

4. The output gate:

$$o_t^{(\text{backward})} = \sigma(W_o^{(\text{backward})} \cdot [h_{t+1}^{(\text{backward})}, e_t] + b_o^{(\text{backward})}) \tag{15}$$

$$h_t^{(\text{backward})} = o_t^{(\text{backward})} * \tanh(C_t^{(\text{backward})}) \tag{16}$$

The final feature representation h_t for each word in the sentence combines the information from both directions:

$$h_t = [h_t^{(\text{forward})}; h_t^{(\text{backward})}] \tag{17}$$

This concatenated feature captures contextual information from both the past and future relative to the current word, providing a comprehensive understanding of its role and meaning within the sentence.

C. Graph Convolutional Network (GCN)

Given the dependency structure of a sentence, the adjacency matrix A is constructed:

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2}\tilde{A}\tilde{D}^{-1/2}H^{(l)}W^{(l)}) \tag{18}$$

where $H^{(l)}$ and $H^{(l+1)}$ are the node features at layer l and $l+1$, respectively, $\tilde{A} = A + I$ (where I is the identity matrix), $\tilde{D}$ is the degree matrix of $\tilde{A}$, σ is the activation function, and $W^{(l)}$ is the weight matrix of layer l.

D. Attention Mechanism

The attention mechanism calculates the relative importance of each node to other nodes:

$$A = \text{softmax}\left(\frac{(HW^Q)(HW^K)^T}{\sqrt{d_k}}\right) \tag{19}$$

$$H' = AV \tag{20}$$

where W^Q, W^K, and W^V are the linear transformation matrices for queries, keys, and values, respectively, A is the attention weight matrix, and H' is the updated feature matrix with attention applied.

E. Pooling and Classification

Pool all node features to obtain a global representation:

$$z = \text{Pooling}(H') \tag{21}$$

The global representation z is then used for relation classification:

$$\text{Prob} = \text{softmax}(W_c z + b_c) \tag{22}$$

where W_c and b_c are the weight and bias of the classification layer, respectively, and Prob is the probability distribution over different relation types.