Deep Network Architectures for Relation Extraction in Biomedical Texts

A B. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Manoj Ghuhan A
(140101082)

under the guidance of

Dr. Ashish Anand

to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this thesis entitled "**Deep Network Architectures for Relation Extraction in Biomedical Texts**" is a bonafide work of **Manoj Ghuhan A (Roll No. 140101082)**, carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Dr. Ashish Anand**

Assistant Professor,

Department of Computer Science & Engineering,

Indian Institute of Technology Guwahati, Assam.

Apr, 2020

Guwahati.

Acknowledgements

I would like to express my gratitude to Dr. Ashish Anand for giving an opportunity to work under him for my BTP and also for guiding me despite his busy schedule.

Contents

List of Figures

Chapter 1

Introduction

Relation extraction(RE) is an important component of information extraction. RE is the task of detecting and identifying the relationship between the entities present in a sentence or a piece of text. For example consider the given sentence [S1], here the entities *Harshini* and *Darshini* are related by the *siblings* relation.

[S1] : Bhuvana has two daughters **Harshini** and **Darshini**.

RE is important for many applications such as knowledge base creation, facilitating precise sentence interpretation, building modern question answering systems ([CLJ07], [MFE03]). In particular, RE plays a vital role in the biomedical domain. A huge amount of clinical and biomedical unstructured texts are available in diverse sources such as research articles, discharge summaries, medical reports, case studies. These unstructured texts represent useful information which when extracted can be helpful for many applications such as identifying gene-disease relationships, protein-protein Interaction, Drug-Drug Interaction, medical Knowledge base creation ([YSY11]). Presently available state of the art entity recognition systems for proteins, drugs, disease, genes, test, and treatments have achieved sufficient levels of accuracy. So the performance of an RE system will depend on how accurately the relationships between the entities are classified.

Many of the existing methods for RE needs explicit feature engineering which becomes tedious and time consuming given the diversity of the biomedical data sources. Recently Deep network architectures have gained popularity due to their capacity to capture important features without requiring extensive manual feature engineering. In this project, we propose a model that uses LSTMs and GCNs (explained in chapter 3) in sequence for the task of relation classification. Our proposed model uses only word embedding and syntactic dependency graph as input features.

1.1 Organization of The Report

This chapter described the task of RE in NLP and also discussed its importance in the biomedical domain. The remainder of the report is organized as follows: In Chapter 2, we provide the literature review of the existing methods for RE. In Chapter 3, we give an introduction to GCNs and discuss our proposed architecture. In chapter 4, we describe the datasets used and also discuss the experimental details. In chapter 5, we examine the performance of our model and finally conclude in chapter 6.

Chapter 2

Review of Prior Works

Over the years researchers have employed various methods for building efficient relation extractors. Some of the methods are:

- Co-occurrence based methods
- Rule based methods
- Bootstrapping
- Distantly supervised method
- Feature based methods
- Kernel methods

Co-occurrence based methods are one of the simplest methods. In this method, it is assumed that a relationship exists between two entities if they appear together in many sentences ([RBM06], [QSY11]). Rule based methods use hand built patterns for extracting relations. These patterns are generated by examining the patterns in the relation instances carefully ([JTC00], [GLM03]).

Bootstrapping methods uses an initial set of seeds to learn patterns in large unstructured text iteratively ([Xu08]). The initial seeds can be few high precision patterns for each relation class.

Feature based methods first perform feature extraction on the sentences with named entities. It uses various syntactic and semantic features such as the Constituent path through the parse tree, Base syntactic chunk path, Typed dependency path etc. Feature vectors are constructed from the extracted features and are trained using a classifier ([Hon05], [BRR11]). Kernel based methods uses kernel functions for exploiting syntactic and semantic information and are an extension to feature based methods ([QZ12], [DZR03]). Performance of these methods relies hugely on the extracted features. Though Feature and kernel based methods have obtained state of the art results they suffer from various disadvantages.

- Feature extraction requires heavy linguistic and domain-specific knowledge.
- It also depends on the performance of other NLP systems for dependency parsing, POS tagging, chunking etc.
- Often these methods end up in high dimensional feature vectors. An enormous amount of training data is needed in such cases for obtaining good accuracy.
- Also, Feature extraction is tailored according to the characteristics of the information source. As seen earlier the biomedical texts are present in wide variety of sources. Thus automatic feature extraction becomes very difficult and time consuming.

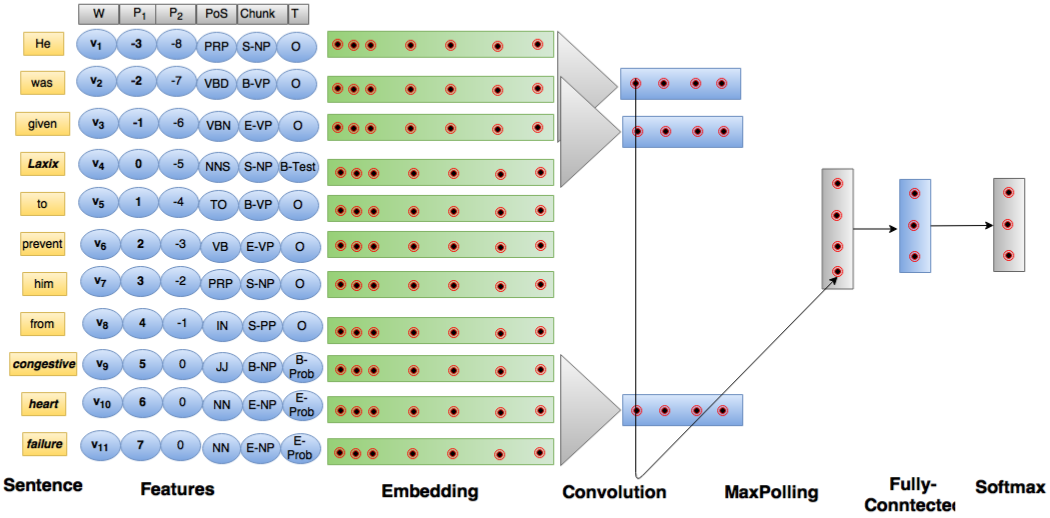
Deep learning models have gained much popularity lately due to their ability to efficiently learn the features without the need for extensive manual feature engineering. Deep learning architectures have been employed successfully in numerous fields such as natural language processing, computer vision, image processing etc. In the following sections, we will review three convolutional and recurrent neural network models for relation classification in biomedical texts.

2.1 CNN model

The architecture of the model proposed by [SKSG16] is shown in 2.1. The main advantage of this model is that it uses less number of features than the previous state of the art methods and also achieves better results compared to them.

The model has 6 layers. The features used are: the word itself, POS tag, chunk tag, distance of the word from the first entity, distance of the word from the second entity and the entity type. Each feature value is mapped to a feature vector in the embedding layer. The feature vector for the entire word is obtained by concatenating the vector representation of all the feature values. Pre-trained word vectors were used as feature vectors for word embedding. The rest of the feature matrix were randomly initialized.

Convolution using filters of different sizes are applied on the obtained feature vectors to extract local features from different parts of the sentence. Max pooling is applied to get fixed size global features for the sentence. This global feature vector is fed into a fully connected feed forward layer. Softmax classifier is used as the output layer.The output is a probability vector corresponding to all the relation classes. This model obtained an F1 score of 71.16% on the i2b2 dataset.



	W	P_1	P_2	PoS	Chunk	T
He	v_1	-3	-8	PRP	S-NP	O
was	v_2	-2	-7	VBD	B-VP	O
given	v_3	-1	-6	VBN	E-VP	O
Laxix	v_4	0	-5	NNS	S-NP	B-Test
to	v_5	1	-4	TO	B-VP	O
prevent	v_6	2	-3	VB	E-VP	O
him	v_7	3	-2	PRP	S-NP	O
from	v_8	4	-1	IN	S-PP	O
congestive	v_9	5	0	JJ	B-NP	B-Prob
heart	v_{10}	6	0	NN	E-NP	E-Prob
failure	v_{11}	7	0	NN	E-NP	E-Prob

Fig. 2.1: CNN model [SKSG16]

2.2 LSTM

Figure 2.2 shows one of the architectures proposed by [SA17]. CNN learns local context from the sentences well. But when the sentence is long or the important features are separated by a larger distance, CNN may fail to capture the context from the sentence. In order to overcome this issue of long term dependency, [SA17] uses Long Short Term Memory networks(LSTMs). They are designed specifically for avoiding the long term dependency problem.

This model uses only 3 features namely the word itself, the distance of the word from the first entity and the distance of the word from the second entity to represent the words in the sentence. As before the embedding layer maps the feature values to its vector representations. Pre-trained word vectors were used for word embedding and the rest of the feature matrix were randomly initialized.

Bi-LSTM layer is applied so that both the forward and the backward context can be encoded to the words feature vector. The LSTM layer is followed by a pooling layer, fully connected layer and softmax output layer. [SA17] has proposed 3 models namely B-LSTM, AB-LSTM, joint AB-LSTM. B-LSTM uses max pooling for obtaining the global feature vector. AB-LSTM uses attentive pooling for obtaining the global feature vector. AB-LSTM is an ensemble of the other two models. Training and testing were done on SemEval-2013 DDI extraction dataset. The proposed model outperformed the CNN model.

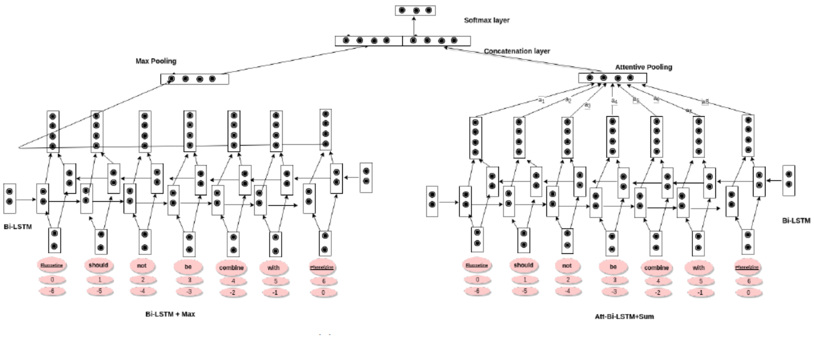


Fig. 2.2: Joint AB-LSTM model [SA17]

2.3 CRNN

Figure 2.3 shows the architecture proposed by [DRA17]. The proposed architecture uses a combination of both LSTM and CNN for obtaining the global feature vector. CNN learns the local context better and LSTM is capable of capturing the long term dependencies. Robust representation of the sentence is obtained by using both short and long term dependencies.

The model uses only the word embedding as the input feature. It is followed by a Bi-LSTM layer. In [JZ15] it is showed that regional embedding conveys higher level concepts than the word embedding as all words may not be useful for representing the sentence. Regional embedding of the sentence is obtained by applying max pooling over short phrases. A CNN layer is applied over the regional embedding. It is followed by a pooling layer, fully connected layer and a softmax output layer.

Training and testing were done on both SemEval 2013 DDI extraction dataset and i2b2 2010 relation extraction dataset. This model outperformed the previous models.

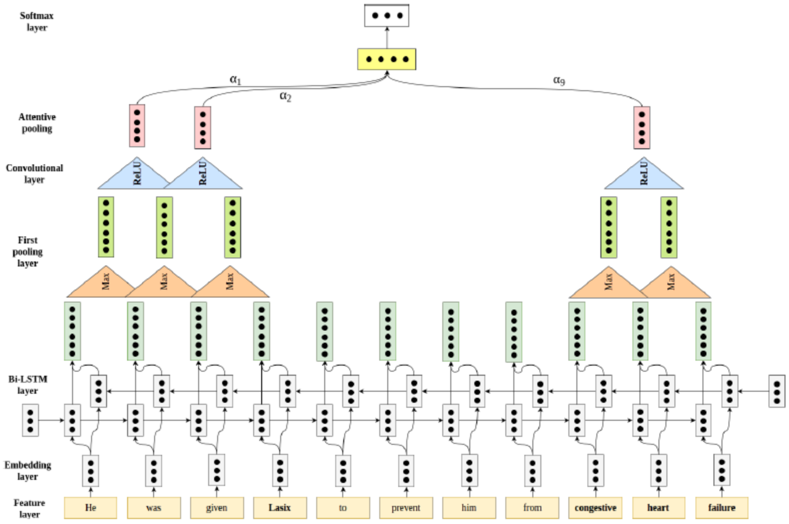


Fig. 2.3: CRNN-Att Model [DRA17]

2.4 Limitations

Though the models mentioned above performed well in general, they face difficulties in classifying relations in longer sentences or when entities are separated by larger distances. We believe that by incorporating the syntactic information of the sentences for relation classification, we can overcome the above limitations. The syntactic relations between words can be represented by using dependency graph. In the dependency graph (see fig.2.4) the arrows point from head to their dependents and the labels indicate the grammatical function of the dependent. In next chapter, we describe a version of GCNs as proposed by [KW17] for modeling the syntactic dependency structure into the neural networks.

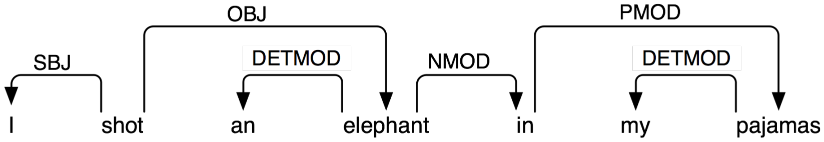


Fig. 2.4: Dependency graph of an example sentence

Chapter 3

Proposed Architecture

3.1 Graph Convolution Networks

This section describes the Graph Convolution Networks (GCNs) proposed by [KW17]. GCNs are neural networks designed to work on arbitrarily structured data. [KW17] proposed GCN for the task of semi-supervised classification of nodes in an undirected graph. The goal of GCN is to produce node level output features by encoding information about the neighbors. A single layer of GCN can encode information only about its immediate neighbors. Information about higher degree neighborhoods can be incorporated by stacking multiple GCN layers together. The inputs required for the GCN are:

- A undirected graph $G = (\nu, \epsilon)$, where ν and ϵ are the node set and edge set of the graph respectively.
- An input feature representation $x_i \in R^m$ for every node i in ν.

The information about the neighbors of a node is computed as

$$h_v = ReLU(\sum_{u \in N(v)} (Wx_u + b)),$$

where $W \in \mathbb{R}^{m \times m}$ and $b \in \mathbb{R}^{m}$ are the weight and the bias parameters. It is assumed that every node has a self loop so that its input representation influences its encoded feature. Higher degree neighbourhoods is obtaining by stacking GCN layers:

$$h_v^{(k+1)} = ReLU(\sum_{u \in N(v)} (W^{(k)} h_u^{(k)} + b^{(k)})),$$

where k denotes the layer number and $h_u^{(1)} = x_u$.

3.2 Syntax-Aware GCNs

GCNs described in the previous section can only operate on undirected graphs. As syntactic dependency graphs are directed and labeled, the above mentioned version of GCNs can't be directly applied. [MT17] proposed a generalization of GCNs for operating on directed and labeled graphs. In the following section, we briefly discuss the modifications introduced by [MT17].

3.2.1 Incorporating direction and labels

[MT17] argued that there is no reason to assume that information flows only along the dependency arcs (eg. from shot to elephant). In order to allow for information to flow in the opposite direction (ie. from elephant to shot) opposite edges (from dependents to heads) are introduced in the graph. Syntactic functions are encoded only in the bias vector. The modified computation for encoding the neighbors information is given as

$$h_v = ReLU(\sum_{u \in N(v)} (W_{dir(v,u)} x_u + b_{L(v,u)})),$$

here $dir(v,u)$ is the direction of the egde (v,u). It can be either outgoing (syntactic arc), incoming (opposite edge) or self loop. $L(v,u)$ represents the syntactic function of the label corresponding to the edge (v,u). Having different weight matrices allows information to

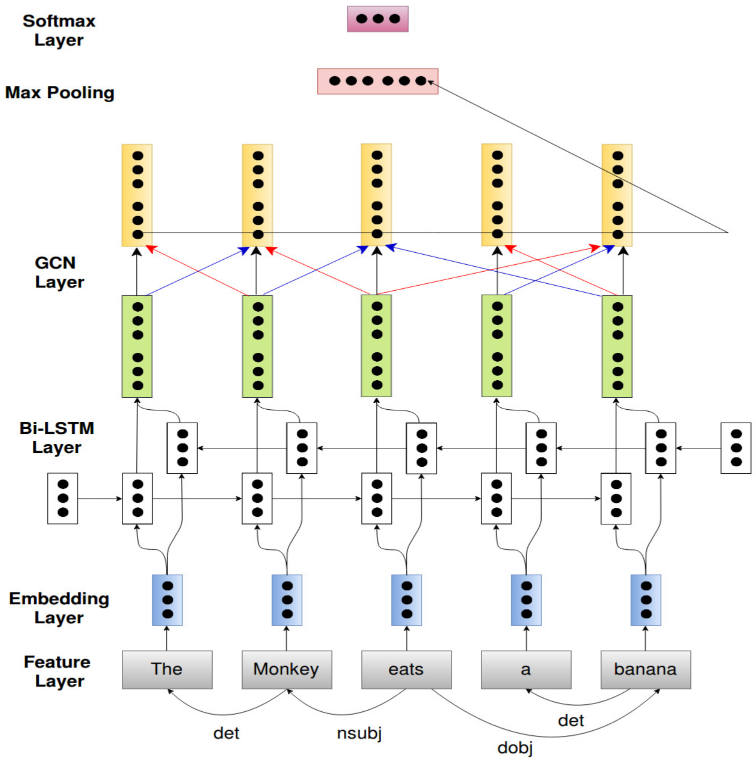


Fig. 3.1: Single layer GCN model. In the GCN layer, black arrows are self loops, red arrows are the syntactic arcs, blue arrows are the opposite edges

flow differently along the edges depending on their direction. We investigate the effect of the opposite edges and different weight matrices in section 5.1.

3.3 Proposed Model

3.3.1 Embedding layer

Each word in the sentence is represented by its word embedding. Pre-trained word vectors obtained using GloVe method ([JPM13]) are used for word embedding. Words that are not present in the embedding matrix are initialized randomly.

3.3.2 Bi-LSTM layer

Bi-LSTM layer as used in [Gra13] is applied to obtain the sequential information present in the sentence. Bi-LSTM layer is used for obtaining sequential information in both forward and backward directions. Let h_f^t and h_b^t be the outputs at time t of both the LSTM layers respectively. The combined output is obtaining by concatenating both these vectors.

$$h^t = h_f^t : h_b^t, \quad h^t \in \mathbb{R}^{n_o}$$

3.3.3 GCN layer

Each word in the sentence is considered as a node and the edge set includes all the dependency arcs, the opposite edges and the self loops. An external syntactic parser is used for determining the syntactic dependency. The output of the Bi-LSTM layer for each word h_v is the input feature representation of each nodes. The GCN version as described above is used. Let g^i be the feature vector for the i^{th} word in the sentence after incorporating the neighbors information.

3.3.4 Pooling layer

Max pooling over time is applied for obtaining the global feature vector. Max pooling extracts the most important feature from the entire sentence ([CW08]).

$$z = \max_{1 \leq i \leq |\nu|} [g^i],$$

where $z \in \mathbb{R}^{n_o}$ is the dimension wise maximum of the $g^{i\prime}s$.

3.3.5 Fully connected and softmax

The global feature vector obtained from the max pooling layer is fed into a fully connected feed forward layer consisting of r units, where r is the total number of relation classes. It is followed by a softmax layer. The output is a probability distribution vector over all the relation classes.

$$p(r_i|x) = Softmax(W_i^f z + b_i^f)$$

where W^f and b^f are the weight and bias parameters of the fully connected layer respectively.

Chapter 4

Experiments

4.1 Datasets

we have used 2 datasets namely SemEval-2013 DDI extraction dataset and i2b2-2010 clinical relation extraction dataset for examining the performance of the models.

4.1.1 SemEval-2013 DDI extraction dataset

This dataset was released as a part of SemEval-2013 task. The corpus contains manually annotated documents taken from two sources, DrugBank database and MedLine abstracts. The interaction between drugs is classified into one of the following four classes namely advice (suggestion or advice regarding the use of 2 drugs is specified), effect (effect of DDI is described), mechanism (pharmacokinetic interactions) and int (drug interaction exists without any information). For sentences containing more than two drug names, separate instances for each pair of drug names are included and the interaction between them is annotated. The statistics of the dataset is given in table 4.1.

4.1.2 i2b2-2010 relation extraction dataset

This dataset was released as a part of i2b2-2010 shared task challenge. It contains manually annotated sentences from discharge summaries. The dataset has 8 relation types

Class	Train Size	Test Size
Mechanism	1264	302
Effect	1620	360
Advice	820	221
Int	140	96
None	12651	3046
Total	**3844**	**979**

Table 4.1: Statistics of the DDI dataset

namely, treatment caused medical problems (TrCP), treatment administered medical problem (TrAP), treatment worsen medical problem (TrWP), treatment improve or cure medical problem (TrIP), treatment was not administered because of medical problem (TrNAP), test reveal medical problem (TeRP), test conducted to investigate medical problem (TeCP), and medical problem indicates medical problem (PIP). Though the original dataset had 394 documents for training and 477 documents for testing, we were able to download only 170 training documents and 256 testing documents. We combined all the documents and split it into an 80:20 ratio for the training set and the testing set respectively. Again for all sentences containing more than two entities, separate instances corresponding each pair of entities is included. Initial experiments indicated that there were not enough training samples present for all classes. So we removed all the instances belonging to 3 relation classes: TrNAP (173 instances), TrIP (202 instances), TrWP (132 instances) and). The statistics of the dataset is given in Table 4.2.

Class	Train Size	Test Size
TeCP	408	102
TrCP	434	109
TrAP	2107	527
PIP	1775	444
TeRP	2453	614
None	42658	10665
Total	**7177**	**1796**

Table 4.2: Statistics of the i2b2 dataset

4.2 Preprocessing

In the DDI dataset, the entities(drug names) are substituted with the tokens DRUG_A and DRUB_B respectively. Remaining drug names are substituted with the token DRUG_N. Similarly in the i2b2 relation extraction dataset, the entities are replaced with their entity types. For example, consider the sentence:" *MRI* would be more sensitive for the detection of *acute infarction*." It is replaced with " *TEST_A* would be more sensitive for the detection of *PROBLEM_B*". Further, all the digits are replaced with the token DG. Negative instances were filtered from the dataset as earlier studies ([SA17],[LWL16]) have reported positive impact of negative instance filtering.

4.3 Implementation Details

Pre-trained word vectors used in the embedding layer are obtained using the GloVe method ([JPM13]) on PubMed corpus. Word vectors of size 100 are used and are updated during the training period. We used dependency parser from Stanford CoreNLP ([CDMM14]) for generating the dependency graph of the sentences. Both L2 regularization and dropout techniques ([NSS14]) are used for regularization. In dropout technique, nodes are randomly dropped to avoid co-adaptation of hidden units. Dropout is applied on the output of the pooling layer. Loss function is optimized using Adam technique ([KB14]). We used 20% of the training set as the validation set. For baseline models, hyperparameter values as suggested in their respective papers are used. We implemented our model in python language using Tensorflow package ([MA16]). The performance of the models is discussed in next chapter.

4.4 Baseline Methods

We use 3 baseline methods that were previously used for the task of relation classification. A multifilter CNN with max-pooling, an LSTM model with max pooling, and the CRNN model with max pooling are used as the baseline methods. All three models are described in detail in chapter 2. The first two models are single layer neural networks while the final one is a two layer network consisting of a CNN layer followed by an LSTM layer.

Chapter 5

Results and Discussions

5.1 Effect of direction and opposite edges

Model	**i2b2-2010**			**DDI extraction**		
	Precision	**Recall**	**F1 score**	**Precision**	**Recall**	**F1 score**
SD(k=1)	0.797	0.750	0.767	**0.801**	0.550	0.647
UD(k=1)	0.798	0.727	0.760	0.748	0.592	0.655
GCN(k=1)	**0.801**	0.**772**	**0.783**	0.737	**0.625**	**0.664**

Table 5.1: Effect of direction and opposite edges

In the dependency graph of the sentence, we added opposite edges (from dependents to heads) to allow the flow of information in the opposite direction. Further, we used 3 different weight matrices corresponding to 3 different edge directions (incoming, outgoing, self-loop). In order to investigate the effect of direction and opposite edges, we examine the performance of 2 modified versions of our model on i2b2 and DDI datasets. In the first version of the model (SD) opposite edges are not added to the dependency graph. In the second version (UD) same weight matrix is used for all types of edges.The performance of the models is given in Table 5.1. The results indicate that both the opposite edges and having different weight matrices for different edge types play a vital role in improving the performance of the model.

5.2 Effect of number of GCN layers

Model	i2b2-2010			DDI extraction		
	Precision	Recall	F1 score	Precision	Recall	F1 score
LSTM(k=0)	0.791	0.748	0768	**0.772**	0.585	0.658
GCN(k=1)	**0.801**	0.772	0.783	0.737	0.625	0.664
GCN(k=2)	0.786	**0.799**	**0.791**	0.732	**0.640**	**0.679**

Table 5.2: Effect of no of GCN layers

In order to understand the importance of GCN layers, we compared the performance of our model by varying the no of GCN layers. The performance of the models with 0, 1 and 2 GCN layers on i2b2-2010 dataset and DDI extraction dataset are given in Table 5.2. Each layer of GCN expands the syntactic neighborhood of the words in the sentence. It can be seen that stacking more GCN layers on top of the LSTM layer improves the performance of the model on both the datasets. This shows that GCN layers are effective. Further, we can also conclude that GCNs and LSTMs have complementary modeling power.

5.3 Comparison with baseline methods

Model	i2b2-2010			DDI extraction		
	Precision	Recall	F1 score	Precision	Recall	F1 score
CNN	0.708	0.719	0.708	0.688	0.549	0.602
LSTM	0.791	0.748	0768	0.772	0.585	0.658
CRNN	**0.803**	0.748	0.772	0.753	0.598	0.657
GCN(k=1)	0.801	0.772	0.783	**0.737**	0.625	0.664
GCN(k=2)	0.786	**0.799**	**0.791**	0.732	**0.640**	**0.679**

Table 5.3: Comparison of our model with baseline models on i2b2-2010 and DDI extraction datasets

Table 5.3 shows the performance of our model on the i2b2 and DDI extraction datasets, as compared to other baseline models. As can be seen, our model outperforms the baseline models on both the datasets.

5.4 Effect of sentence length

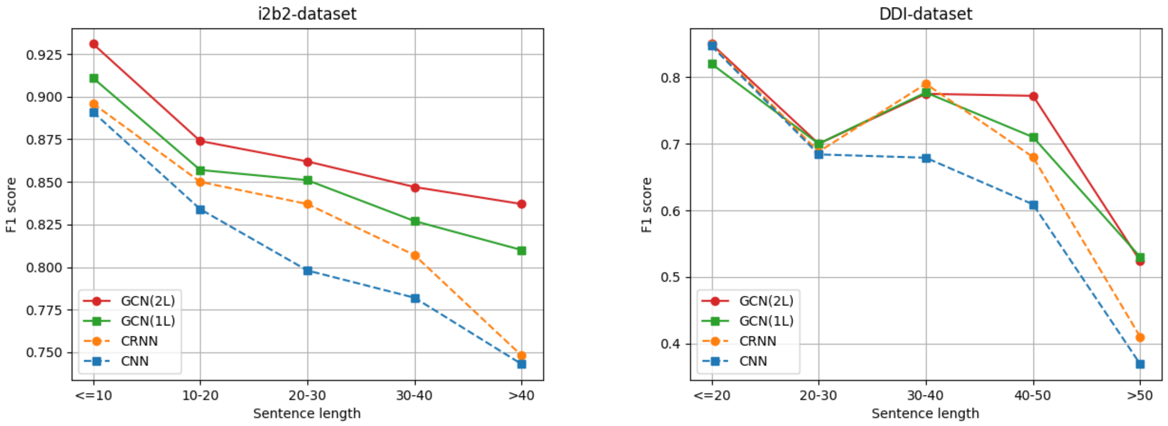


Fig. 5.1: Comparison of the models on the basis of sentence length on i2b2 and DDI datasets

We conjecture that our model should be able to classify relations on longer sentences better when compared to other models because longer sentences are likely to contain long distance syntactic dependencies which are directly encoded in GCNs. To confirm our hypothesis, we partitioned the sentences into various buckets based on their length and computed F1-score for each of them. The results are given in Fig.5.1. It is evident from the results that our model significantly outperforms other models as the sentence length increases, thus confirming our hypothesis.

Chapter 6

Conclusion

In this work, we proposed a neural network model consisting of LSTM and GCNs in sequence for the task of relation classification in biomedical texts. GCNs are able to encode the syntactic information of the sentences at word level efficiently. We evaluated the performance of our model on two datasets namely SemEval-2013 DDI extraction dataset and i2b2-2010 dataset and our model outperformed the baseline models on both the datasets. We also observed that our model performed significantly better on longer sentences when compared to other models.

References

[BRR11] Sanda Harabagiu Bryan Rink and Kirk Roberts. Automatic extraction of relations between medical concepts in clinical texts. *Journal of the American Medical Informatics Association*, pages 594–600, 2011.

[CDMM14] John Bauer Jenny Rose Finkel Steven Bethard Christopher D Manning, Mihai Surdeanu and David McClosky. The stanford corenlp natural language processing toolkit. *In ACL*, 2014.

[CLJ07] Yi-Gyu Hwang Changki Lee and Myung-Gil Jang. Fine-grained named entity recognition and relation extraction for question answering. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 799–800, 2007.

[CW08] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. *Proceedings of the 25th international conference on Machine learning*, pages 160–167, 2008.

[DRA17] Sunil Kumar Sahu Desh Raj and Ashish Anand. Learning local and global contexts using a convolutional recurrent network model for relation classification in biomedical text. *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 311–321, 2017.

[DZR03] Chinatsu Aone Dmitry Zelenko and Anthony Richardella. Kernel methods for relation extraction. *The Journal of Machine Learning Research*, pages 1083–1106, 2003.

[GLM03] Hsinchun Chen Gondy Leroy and Jesse D Martinez. A shallow parser based on closed-class words to capture relations in biomedical text. *Journal of biomedical Informatics*, page 145158, 2003.

[Gra13] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[Hon05] Gumwon Hong. Relation extraction using support vector machine. *Natural Language ProcessingIJCNLP*, pages 366–377, 2005.

[JPM13] Richard Socher Jeffrey Pennington and Christopher D Manning. Glove: Global vectors for word representation. *EMNLP*, pages 1532–1543, 2013.

[JTC00] Christos Ouzounis Stephen Pulman James Thomas, David Milward and Mark Carroll. Automatic extraction of protein interactions from scientific. *Pacific symposium on biocomputing*, pages 538–549, 2000.

[JZ15] Rie Johnson and Tong Zhang. Semi-supervised convolutional neural networks for text categorization via region embedding. *Advances in neural information processing systems*, pages 919–927, 2015.

[KB14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KW17] Thomas N. Kip and Max Welling. Semi-supervised classification with graph convolution networks. *arXiv:1609.02907v4*, 2017.

[LWL16] Gerard de Melo Linlin Wang, Zhu Cao and Zhiyuan Liu. Relation classification via multi-level attention cnns. *In ACL*, 2016.

[MA16] Paul Barham Eugene Brevdo Zhifeng Chen Craig Citro Greg S Corrado Andy Davis Jeffrey Dean Matthieu Devin et al. Martn Abadi, Ashish Agarwal. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467L*, 2016.

[MFE03] Eduard Hovy Michael Fleischman and Abdessamad Echihabi. Offline strategies for online question answering: Answering questions before they are asked. *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 1–7, 2003.

[MT17] Diego Marcheggiani and Ivan Titov. Encoding sentences with graph convolutional networks for semantic role labeling. *arXiv:1703.04826v4*, 2017.

[NSS14] Alex Krizhevsky Ilya Sutskever Nitish Srivastava, Geoffrey E Hinton and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

[QSY11] Yousuke Watanabe Qiang Song and Haruo Yokota. Relationship extraction methods based on co-occurrence in web pages and files. *Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services, iiWAS 11*, pages 82–89, 2011.

[QZ12] Longhua Qian and Guodong Zhou. Tree kernel-based proteinprotein interaction extraction from biomedical literature. *Journal of Biomedical Informatics*, pages 535–543, 2012.

[RBM06] Arun Ramani Razvan Bunescu, Raymond Mooney and Edward Marcotte. Integrating cooccurrence statistics with information extraction for robust retrieval of protein interactions from medline. *Proceedings of the Workshop on Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis*, pages 49–56, 2006.

[SA17] Sunil Kumar Sahu and Ashish Anand. Drug-drug interaction extraction from biomedical text using long short term memory network. *arXiv:1701.08303v2*, 2017.

[SKSG16] Krishnadev Oruganty Sunil Kumar Sahu, Ashish Anand and Mahanandeeshwar Gattu. Relation extraction from clinical texts using domain invariant convolutional neural network. *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 206–215, 2016.

[Xu08] Fei-Yu Xu. Bootstrapping relation extraction from semantic seeds. *Ph.D. thesis, Saarland University*, 2008.

[YSY11] Hongfei Lin Yue Shang, Yanpeng Li and Zhihao Yang. Enhancing biomedical text summarization using semantic relation extraction. *PLoS ONE*, 2011.