Relation Extraction using Neural Network

Rajat Nagpal
M.Tech SE
rajatnagpal

Ankur Debnath
M.Tech EE
ankurdebnath

Danish Shaikh
M.Tech CN
shaikhm

Abstract

In this project, we have analyzed the task of Relation Extraction by using different neural networks like PCNN, Bi-GRU and GCN. Initially, these models have been used separately and then later, we have ensemble them for the same task. In our experiments, we found that PCNN being a window-based method effectively captures representation of each word in a local context whereas Bi-GRU captures the global representation of the words in the context of the entire sentence. We have found that GCN is much more efficient than LSTMs and Bi-GRU. We have experimented with novel ways to combine these models to explore effectiveness of these models.

1 Introduction

Relationship extraction is the task of extracting semantic relationships from a raw unstructured text. Extracted relationships usually occur between two or more entities of a certain type (e.g. Person, Organisation, Location) and fall into a number of semantic categories (e.g. married to, employed by, lives in).

For example, in the sentence ”The US patent system is in **crisis** that has been caused by the **flood** of patent applications being filed at the USPTO every year.” The entities **[crisis]** and **[flood]** are **cause-effect** relationship.

Supervised methods based on neural networks have been found successful for relation extraction tasks. These neural network methods require large scale labeled data. There is a method called Distant Supervision which automatically creates training data on its own.

Given an entity pair (e; e”) from a knowledge base such as Freebase, assuming that the predefined semantic relation on the KB is r, we simply label all sentences containing the two entities by label r. However, it has a major shortcoming that the distant supervision assumption is too strong and may cause the wrong label problem. It is possible that these two entities may simply share the same topic or maybe representing some other relation than that present in knowledge-base.

The task of relation extraction is divided into two distinct parts. First, is to extract candidate sentences and entity pairs from a plain text and between elements of entity pair, we want to identify the relations and the second task is the relation classification task. In this project, Supervised learning methods are used. In this project we have worked with three neural models, PCNN and Bi-GRU and GCN. Taking a step further, we have proposed a novel interpolation of these three models and with that interpretation we proposed better hybrid models by combining these models.

For this project, SemEval-2010 Task 8 benchmark dataset has been used for experimentation. On this dataset, the present state of the art has F-1 score of **85.9.**

2 Related Work

Recently, the neural network models have dominated the work of Relation Extraction because of higher performances. [9] Zeng et al., 2014 used a convolutional deep neural network (DNN) to extract lexical and sentence level features. This work also proposed the inclusion of position features for the task of relation extraction. These two levels of features are concatenated to form the final extracted feature vector. Softmax classier is used to predict the relationship between two marked entity. In an extension to this, [2] Yatian Shen, Xuanjing Huang, 2016 proposed a attention based convolutional neural network architecture for this task that makes full use of word embedding, part-of-speech tag embedding and position embedding information. Further to incorporate distant supervision, [8] Zeng et al., 2015 combined the multi-instance learning with piecewise convolutional neural networks to learn more relevant features. [4] Lin et al., 2016 employed CNN with sentence-level attention over multiple instances to encode the semantics of sentences. [6] Miwa and Bansal, 2016 used a syntax-tree-based long short-term memory networks (LSTMs) on the sentence sequences. Further, [3] Jat et al., 2017 proposed a weighted ensemble model of a BiGRU-based word attention model and EA, an entity-centric attention model. The current state of art model for relation extraction is proposed in [10] Zhengqiu He et al., 2018 centered on the ideas of using tree-GRU based syntax aware embeddings.The idea of GCN is proposed in [7] **RESIDE** paper by Talukdar et al.

3 Proposed Approach

In this Project, we have experimented with the three models viz., PCNN, Bi-GRU and GCN which have been proposed in literature that are shown to be effective for relationship extraction task. We have experimented with novel combinations of these models.

As components, we have used Piecewise Convolutional Neural Networks, Bi-Gru Word Attention Model and Graph Convolution Networks.

All our models takes a sentence as input and learns useful representations of the sentence building on the initial representation that we feed into it. The initial representation is composed of two components: Word vector representations, position features as described below. This is inline with most recent research in the field.

3.1 Input Representations

The neural network cannot be directly applied on raw word tokens, so we need some vector representation of words. Words are transformed into a low dimension space using pre-trained word embedding lookup. Moreover, [9] have shown position features of each word in the sentence with respect to the entity locations to be effective for the task. Hence, position features are also added using position embedding and the final embedding for a word is concatenation of both the embeddings.

3.1.1 Word Embeddings

The word embeddings helps to map words to a distributed k-dimension representation. These embeddings captures syntactic and semantic information of tokens and using these embeddings and updating them while training has become common process for various classification tasks. We have intitalized our word embeddings with pretrained word embedding and these are treated as model parameters and tuned during model training. The embeddings are pretrained using Word2Vec [5]. The pretraining task aims at capturing the similarity between words based on the assumption that similar words occur in similar neighbourhood

3.1.2 Position Embeddings

A Position feature is defined as the combination of the relative distances from the current word to entity1 and entity2. Two position embedding matrices similar to [9] are randomly initialized and the relative distances are transformed into vectors by looking in these two matrices. The aim of these features is to incorporate in our model the information that which word in the sentence are actually our target nouns/entities between which we want to find the relation using the context given by the sentence.

3.2 PCNN

A key challenge in the task of Relation Extraction is that of variable length of the sequences. Under such conditions, Convolution neural network along with pooling layers gives effective solutions to learn sentence representation. Conventionally, a max pool layer over all the tokens of the words is used for the task of sentence classification. But, this is not very effective in the context of relation extraction due to presence of entities have much more structure to it which needs to be effectively captured. Our framework needs to give special consideration to these words. Based on the above motivation [8] proposed a piece wise max pooling layer following the output of the convolution layer. The intuition behind using a piece wise CNN is to capture both internal and external context. The internal contexts consists the tokens between the two entities and the external contexts consists of the tokens around the entities. Assume a sentence consists of n words and each word is a d-dimensional vector. The representation of every word becomes w_i after convolution layer which is of dimension N × 1 .

N = Number of filters

The sentence is segmented into three parts : $[s1 = w_1......w_{e1-1}]$,$[s1 = w_e1......w_{e2}]$,$[s1 = w_{e2+1}......w_n]$

Now, the piecewise max-pooled sentence is represented as:

$$[s_p = max(s1), max(s2), max(s3)]$$

s_p will be of dimension $N \times 1$

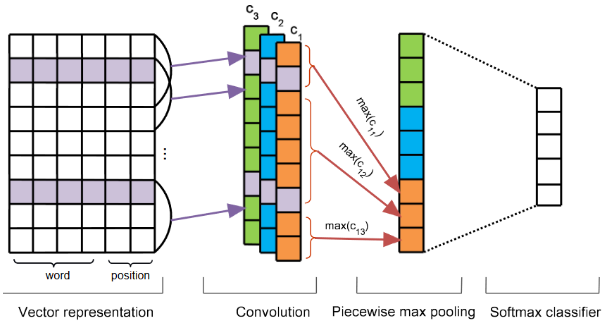

Figure 1: PCNN

3.3 Bi-GRU

A GRU has two gates, a reset gate R, and an update gate Z. Intuitively, the reset gate determines how to combine the new input with the previous memory, and the update gate defines how much of the previous memory to keep around.

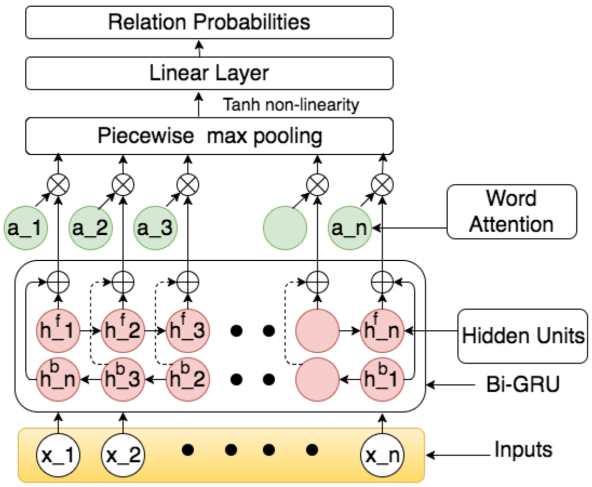

Figure 2: Bi-GRU

The representation output of the word is concatenation of both forward h_i^f and backward state h_i^b vector of the Bi-GRU, each of length $\frac{g}{2}$. The u_i is degree of relevance defined as:

$$w_i = [h_i^f h_i^b]; u_i = w_i \times A \times r$$

$$a_i = softmax(u_i); w_i^{'} = a_i \times w_i$$

We will apply piecewise average pooling same as that done in PCNN.

3.4 Graph Convolutional Network

Given a graph with n nodes, we can represent the graph structure with an n × n adjacency matrix **A** where $A_{ij} = 1$ if there is an edge going from node i to node j. In an L-layer GCN, if we denote by $h_i^{(l1)}$ the input vector and $h_i^{(l)}$the output vector of node i at the l-th layer, a graph convolution operation can be written as

$$h_i^{(l)} = \sigma(\sum_{j=1}^{N} A_{ij}^{'} W^{(l)} h_j^{(l-1)}/d_i + b^{(l)})$$

where $A^{'} = A + I$, $W^{(l)}$ is a linear transformation, d_i is a normalizing constant, $b^{(l)}$ a bias term, and σ a nonlinear function (e.g., ReLU). Intuitively, during each graph convolution, each node gathers and summarizes information from its neighboring nodes in the graph.

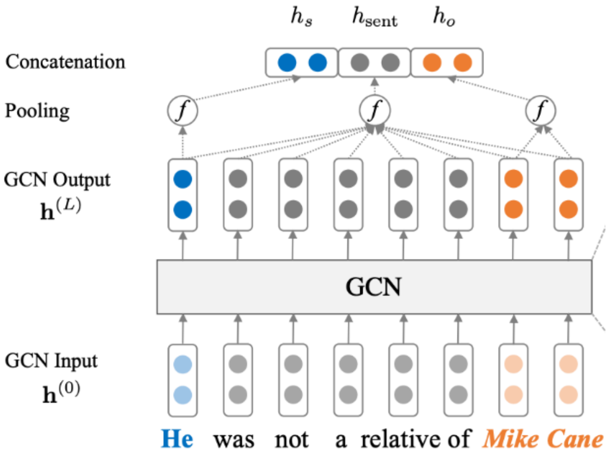

Figure 3: GCN

Note that the GCN model presented above uses the same parameters for all edges in the dependency graph. We also experimented with: (1) using different transformation matrices W for top-down, bottom-up, and self-loop edges and (2) adding dependency relation-specific parameters for edge-wise gating. We found that modeling directions does not lead to improvement and adding edgewise gating further hurts performance. We hypothesize that this is because the presented GCN model is already able to capture dependency edge patterns that are informative for classifying relations, and modeling edge directions and types does not offer additional discriminative power to the network before it leads to overfitting. For example, the relations entailed by $''A's$ son, B'' and

”$B's$ son, A” can be readily distinguished with”$'s$” attached to different entities, even when edge directionality is not considered. We therefore treat the dependency graph as undirected, i.e. $\forall$ i, j $A_{ij} = A_{ji}$.

We have tried to implement GCN layer to the concatenated position embeddings and word embeddings vector. In [11], pooling layer is used at the output.

3.4.1 Contextualized GCN

The network architecture introduced so far learns effective representations for relation extraction, but it also leaves a few issues inadequately addressed. First, the input word vectors do not contain contextual information about word order or disambiguation. Second, the GCN highly depends on a correct parse tree to extract crucial information from the sentence (especially when pruning is performed), while existing parsing algorithms produce imperfect trees in many cases.

To resolve these issues, we further apply a Contextualized GCN (C-GCN) model, where the input word vectors are first fed into a Bi-GRU network to generate contextualized representations, which are then used as $h^{(0)}$ in the original model. This Bi-GRU contextualization layer is trained jointly with the rest of the network.

For instance, in the sentence ” *She was diagnosed with cancer last year, and succumbed this June* ” , the dependency path **She←diagnosed→cancer** is not sufficient to establish that cancer is the cause of death for the subject unless the conjunction dependency to ***succumbed*** is also present.

3.5 Novel Combination Models

As discussed above, GCN layer can be applied after getting vector embeddings. We have tried different combinations of the models discussed above. We have observed that PCNN being a window based method, captures syntactic properties whereas Bi-GRU captures long term dependencies, thus, capturing semantic features. We have experimented with novel combination of these models for the relationship extraction task.

3.5.1 Sequential Models

In these models, we have combined two layers sequentially. The reason being that the first model might not learn some useful features so we feed the output of the first model to the next model. The benefit of sequential modelling is that the next model only focuses on those features which are missed by the previous model.

We have applied PCCN layer before Bi-GRU, Bi-GRU before CNN layer and Bi-GRU followed be GCN layer and then we apply softmax layer to the output.

A comparative analysis of all these models is done in the upcoming sections.

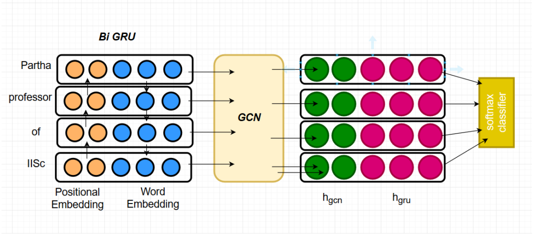

Figure 4: BiGRU with GCN

3.5.2 Ensemble Models

Here, the three component models discussed above are trained parallel to each other and then we take ensemble of all these model outputs to get the desired output. In this method, weights associated to these models are learned in the training process.

$$O_e = W_g \times O_g + W_b \times O_b + W_p \times O_p$$

$$Output = softmax(O_e)$$

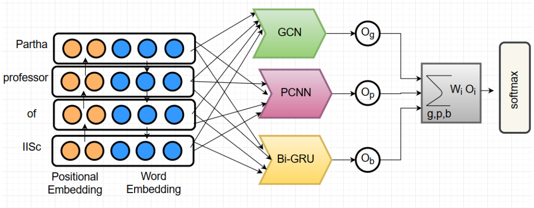

Figure 5: Ensemble Model

After getting the ensemble of these models, a softmax layer is applied to get the desired relationship of the entity pair.

3.5.3 Local Global Word Representation Model

In this Model, we parallely train Bi-GRU and CNN to get the word embeddings. As discussed earlier, CNN give the syntactic representation whereas Bi-GRU give semantic word representation. The idea is to combine both the model embeddings to get the desired word embedding.

The representations returned by both the models are concatenated at word level and then they are sent to a word attention layer for learning attention weights for the words. After the attention layer, the sentence is segmented into three segments based on entities and the segments are pooled and then concatenated as being performed in PCNN. The last stage is Softmax Classifier.

4 Experimental Results

4.1 Dataset

Our experiments in relation extraction are based on the SemEval-2010 Task 8 dataset [1]. The dataset is freely available and contains 10,717 annotated examples, including 8,000 training instances and 2,717 test instances. There are 9 relationships (with two directions) and an undirected other class. The following are examples of the included relationships: Cause-Effect, Component-Whole and Entity-Origin.

4.2 Hyper-parameters

For all our models we have initialized word embeddings with pre-trained word vectors N_w of length 50. The dimension of position embeddings N_1 and N_2 corresponding to each entity is set at 5 and they are initialized from random normal distribution. This is a standard practice which is followed in all the relevant works and hence these parameters are directly taken without any validation. Now, final embedding d for each word in a sentence will be of length 60.

$$N_w = 50, N_1 = 5, N_2 = 5$$

$$N_p = N_1 + N_2$$

$$N = N_w + N_p$$

In **PCNN**, Number of filters used are 230, context window size is 3 and is followed by piece wise max-pooling. To avoid overfitting dropout layer with probability of 0.5 is being used on max-pooling output. Final representation of sentence is of dimension 690.

In **Bi-GRU**, Number of hidden units in each GRU cell is 100 and GRU cell dropout probability is kept at 0.5. Output of BiGRU layer is of $\Re^{n\times200}$, where n is the length of sentence. Attention layer is applied to this output and piecewise average pooling is done to obtain final representation of sentence of dimension 600.

In **GCN**, batch size is set to 50 and trained for 100 epochs with GCN dropout probability of 0.5. Default hidden layer size and other parameters are used and no validation was done to tune the hyper parameters of the model.

Remaining combination models are built using the parameters of above mentioned individual models. Also, we have used L2-regualrization to update word embeddings, position embeddings, convolution layer filter weights, attention layer weights and weights which connect sentence level representation to input of softmax. This regularization constant has been kept as 0.05.

4.3 Word Attention

The aim of word attention is to let the model identify which words in a sentence are more relevant to our classification and accordingly give more importance to them.

The figures below shows how effectively our model was able to learn the salient features in the sentence by giving attention to a particular class of words which defines the relationship between the two entities.

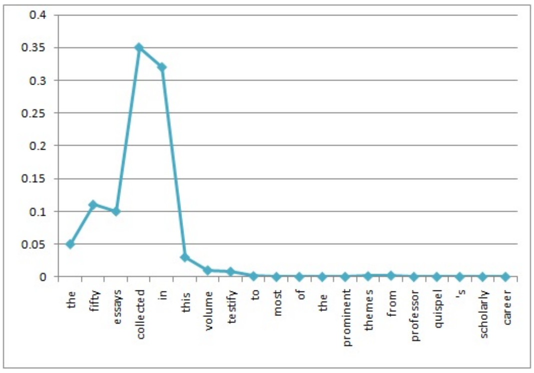

Figure 6

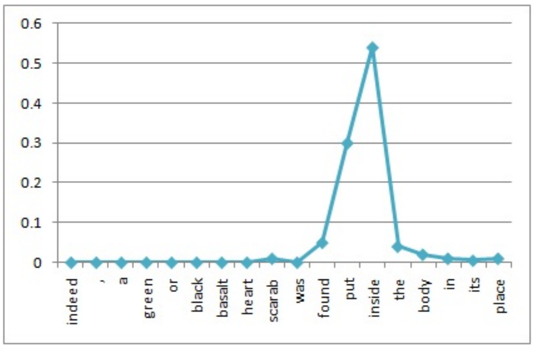

Figure 7

In Figure 6, the sentence contains a relation of Member-Collection between the entity pairs and essays and volume and our model is rightly giving the most attention to the word collected. While in Figure 7, the sentence contains a relation of Content-Container between the entity pairs scarab and body and our model is rightly giving the most attention to the phrase put inside.

4.4 Precision-Recall Curve

Precision Recall curve is used as performance metric in our experiment. It gives the trade off between precision and recall. Precision Recall curve is plotted as two class classification in this case. Below are precision recall curves for different models :

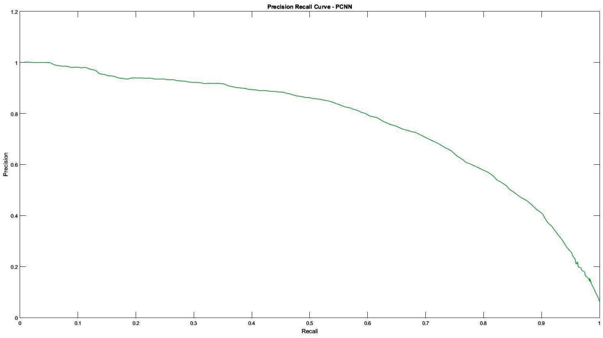

Figure 8: PCNN

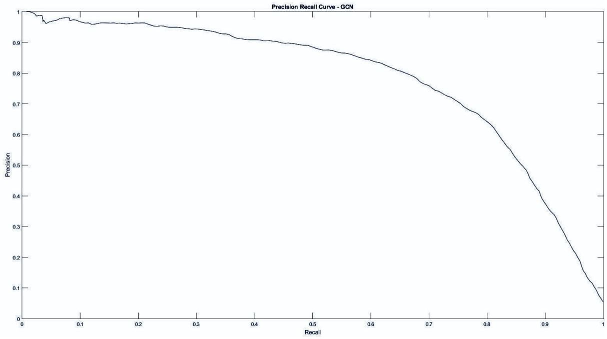

Figure 9: GCN

A high area under precision recall curve denotes both high recall and high precision which shows that the classifier is returning accurate results(high precision) as well as returning a majority of all positive results(high recall).

4.5 F1 Score

F1-score is weighted harmonic mean of Precision and Recall which is used as metric to evaluate the performance of models built. F1-Score:

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

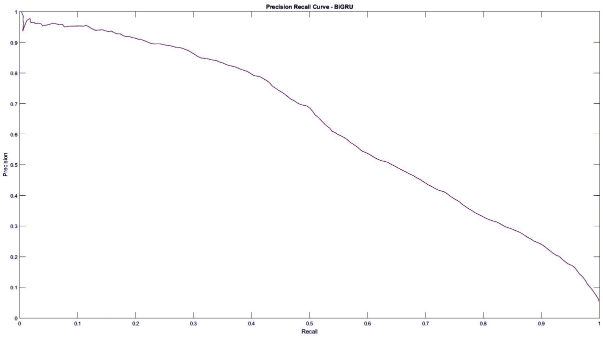

Figure 10: Bi-GRU

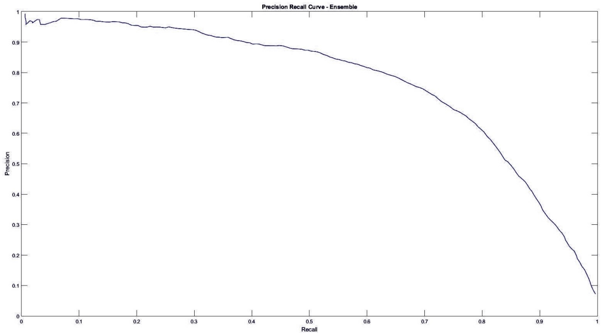

Figure 11: Ensemble Model

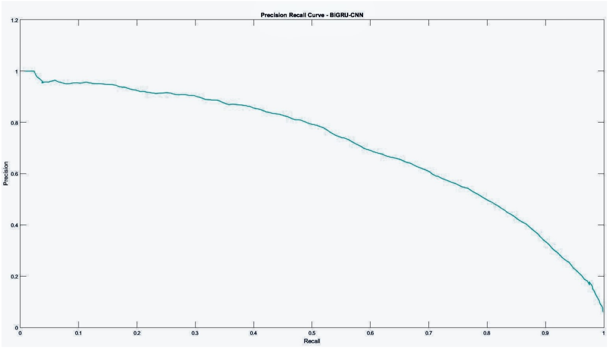

Figure 12: Sequential Model: CNN followed by Bi-GRU

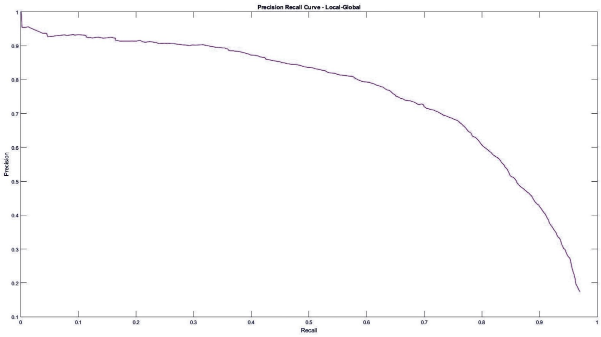

Figure 13: Local Global Word Representation Model

It can be inferred from the above figure that the Local Global model and Ensemble model gives

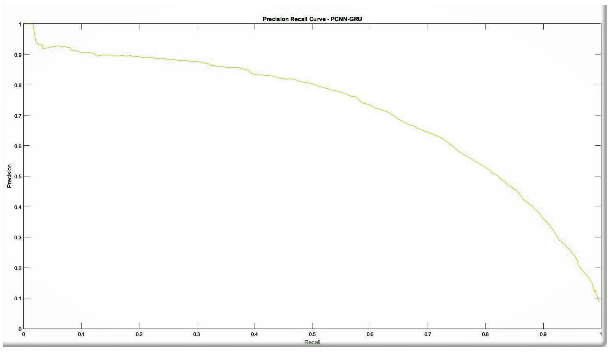

Figure 14: Sequential Model: BiGRU followed by PCNN

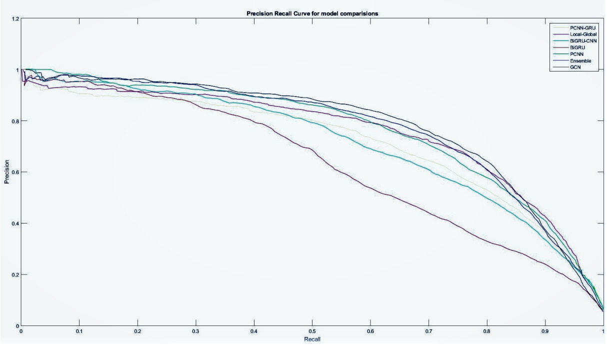

Figure 15: PR Curve for all the Models

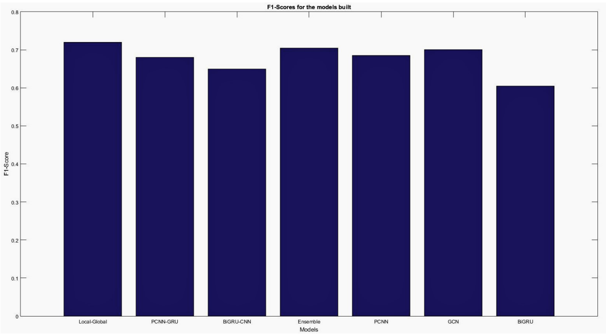

Figure 16: F1 Scores

the best performance among all, if only F1-score measure is considered.

4.6 Gradation Table

Due to lack of training data, neural network models suffer from overfitting as can be seen in relation extraction, there is limited amount of data in supervised settings. Our models also faced the same problems. To avoid these problems, we used regularization, pre trained word embeddings and Xavier initialization of weights of the model. We have used position embeddings and word embeddings and we have noticed that concatenating both, position as well as word embeddings yield better results

Gradation Table

Gradation	F1 Score
Word Embeddings	0.47
word+pos embeddings	0.52
word+pos embeddings +regularization	0.63
pretrained word vectors+pos embeddings+regularization	0.71

Table 1: F1 score incorporating various techniques

Table 1 represents the F1-score performance with such techniques being incorporated into our model one after other.

A key conclusion that can be drawn from the above experiment is that a model is only as effective as its tuning. Hyper-parameter tuning and proper regularization sometimes give better boost to the performance than going for more advanced models.

5 Conclusion

In this project, we worked on the task of relation extraction. The speciality of relation extraction task as compared to other sentence classification tasks is the presence of entities. This salient feature of relation extraction task encourages the ideas of positional embeddings and piecewise pooling. As baseline models, we have used PCNN, Bi GRU and GCN. Observing the strengths of all these models, we have tried novel combinations of these models. We believe that different combinations of these models can be applied due to their distinguishable properties.

We came across several challenges during the course of the project. Key challenges include working with variable length input, avoiding overfitting in a high dimensional neural model while training with a small dataset. Capturing the salient structural properties in a sentence which can aid our performance in the sentence specific manner.

This project is our first major exposure to apply deep learning techniques to an ongoing research problem and we had several useful key takeaways from this experience. First, we developed an understanding and comfort in using deep learning tools like tensorflow. Second, we gained practical incites on good practices to follow while applying deep learning models.

Acknowledgments

We would like to express gratitude to our course instructor, Dr. Partha Pratim Talukdar for giving us this opportunity to work on this project. The project gave us insight to develop a better understanding of all the theoretical concepts taught in class and a great research experience in a growing field of Natural Language Understanding. We strongly believe this exposure would be highly beneficial for our future research endeavours.

References

[1] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 94–99. Association for Computational Linguistics, 2009.

[2] Xuanjing Huang et al. Attention-based convolutional neural network for semantic relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2526–2536, 2016.

[3] Sharmistha Jat, Siddhesh Khandelwal, and Partha Talukdar. Improving distantly supervised relation extraction using word and entity based attention. *arXiv preprint arXiv:1804.06987*, 2018.

[4] Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2124–2133, 2016.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[6] Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. *arXiv preprint arXiv:1601.00770*, 2016.

[7] Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. RESIDE: Improving distantly-supervised neural relation extraction using side information. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1257–1266, Brussels, Belgium, October-November 2018. Association for Computational Linguistics.

[8] Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, 2015.

[9] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. Relation classification via convolutional deep neural network. 2014.

[10] Min Zhang, Jie Zhang, and Jian Su. Exploring syntactic features for relation extraction using a convolution tree kernel. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 288–295. Association for Computational Linguistics, 2006.

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