Relation extraction between the clinical entities based on the shortest dependency path based LSTM

Dhanachandra Ningthoujam[†], **Shweta Yadav**[*], **Pushpak Bhattacharyya**[*], **Asif Ekbal**[*]
[*]Indian Institute of Technology Patna, India
[†]ezDI, INC
[*]{shweta.pcs14,pb,asif}@iitp.ac.in, [†]dhanachandra.n@ezdi.com

Abstract

Owing to the exponential rise in the electronic medical records, information extraction in this domain is becoming an important area of research in recent years. Relation extraction between the medical concepts such as medical problem, treatment, and test etc. is also one of the most important tasks in this area. In this paper, we present an efficient relation extraction system based on the shortest dependency path (SDP) generated from the dependency parsed tree of the sentence. Instead of relying on many handcrafted features and the whole sequence of tokens present in a sentence, our system relies only on the SDP between the target entities. For every pair of entities, the system takes only the words in the SDP, their dependency labels, Part-of-Speech information and the types of the entities as the input. We develop a dependency parser for extracting dependency information. We perform our experiments on the benchmark i2b2 dataset for clinical relation extraction challenge 2010. Experimental results show that our system outperforms the existing systems.

1 Introduction

In recent years, the amount of clinical texts in electronic format has drastically increased. However, most of this information such as clinical named entities, the relationship between the clinical entities, medical summary etc. are still embedded in the form of unstructured text. Information are mostly available in an unstructured format. There is a necessity to investigate proper methods for making these information structured so that relevant information can be extracted at ease. Automatic extraction of this information is very essential for the other applications such as clinical decision-making, clinical trial screening, and pharmacovigilance. As a result, the information extraction in this domain is becoming an interesting area of research.

Relation extraction is one of the important tasks in Natural Language Processing (NLP) that aims to identify the semantic relationships between the entities. This also offers us an opportunity to solve other higher level NLP tasks, such as question answering, knowledge graph completion and information extraction. In the clinical domain, an entity can be a medical problem, a treatment, a medical test, body measurements etc. and the relationship between these entities can be diverse which are very much important to understand the clinical phenotypes.

Example 1.1 *A biopsy of this mass was consistent with hematoma .*

In the EMR sentence given in 1.1, the term “a biopsy” refers to a medical test. The terms “this mass” and the “hematoma” correspond to the entities related to medical problems. The aim of this task is to identify the underlying relationship that actually binds these together. In 2010, the Informatics for Integrating Biology and the Bedside (i2b2) organized fourth i2b2/VA challenge (Rink et al., 2011) which aims to extract 3 coarse clinical relations (identifying the possible relations between medical problems and treatments, between medical problems and tests, and between pairs of medical problems) and other 11 fine-grained relationships between the entities.

In recent past, this benchmark dataset has gained a lot of attention in biomedical NLP community, where several techniques have been proposed to solve this problem ranging from the semantic-

injected kernel model to the machine learning embedded models. In recent past, with the success of deep neural network in various biomedical tasks (Kumar et al., 2016; Yadav et al., 2018b, 2016; Ekbal et al., 2016; Yadav et al., 2017b), attention of the researchers have shifted towards building relation extraction systems using deep learning frameworks. Some of the prominent methods include the works presented in de Bruijn et al. (2011) which utilizes the kernel-based model to map features onto higher-dimensional space, Sahu et al. (2016) that explored the convolutional neural network (CNN) driven technique extracting the relations at the sentence level etc. Inspired by Luo et al. (2018), Yadav et al. (2018c), we focus on extracting the Shortest Dependency Path (SDP) between the entities, which helps in eliminating those words which are semantically not related. This enables our model to remove noise which hinders the performance of the system. Towards that, firstly we develop a dependency parser that determines the grammatical relationship between the words in the sentence. Details of the parser are provided in Section 3.1. Given the success of the neural network, in this paper, we utilize Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) network, which has theoretically proven to cover the long-term dependencies. The proposed model takes as an input the embedding of the SDP based words which are assisted by the other latent features such as dependency labels, Part-of-Speech (PoS) information and the entities types. We use the benchmark i2b2/VA challenge dataset to validate our proposed model and for the easy comparison with the existing state-of-the-art techniques. Evaluation shows that our proposed method obtains significant improvements of F-score for most of the relation types except PIP. In summary the key contributions of our proposed work are as follows:
1. We develop a robust parser specific to the clinical texts. Its performance is found to be very close to the other state-of-the-art parsers.
2. We propose a shortest dependency path based LSTM model that provides state-of-the-art performance for relation extraction.

2 Related Works

We consider the relation extraction as a multi-class classification task. Literature shows that most of the existing works in biomedical NLP are based on machine learning (Yadav et al., 2017a, 2018a, 2017b). A large body of works has been dedicated towards building robust relation extraction engines using traditional supervised machine learning models (Abacha et al., 2015; Singhal et al., 2016). A Support Vector Machine (SVM) was proposed in Rink et al. (2011) for relation extraction. The features used here was grouped into six classes: context features, similarity features, nested relation features, single concept features, Wikipedia features, and concept vicinity features. Minard et al. (2011) also used SVM for extracting the relations from the clinical reports. The features used in this task include the lists of medical abbreviations, features to capture the text writing style and semantic types of Unified Medical Language System (UMLS). Uzuner et al. (2010) developed a semantic relation classification system based on SVM. They used UMLS to define medical problems, tests, and treatments. In recent years, deep neural network based methods are being widely used because of its many-fold benefits. The advantage of using deep learning technique is its ability for extracting the features automatically. Sahu et al. (2016) relation extraction system employed Convolutional Neural Network(CNN) to learn features automatically and reduce the burden of manual feature extraction. To extract the relation between the entities in a sentence, it is not necessary to consider the whole sentence-rather only the shortest dependency path (SDP) between the entities as input to the system could achieve the robust state-of-the-art accuracy. Xu et al. (2015) designed a semantic relation extraction system based on SDP. This showed that using a simple negative sampling technique can improve the performance of the system.

Recurrent Neural Network (RNN) is used for sequence prediction problem like named entity recognition. But, when the input is very long then RNN suffers from vanishing gradient and exploding gradient (Hochreiter and Schmidhuber, 1997). To avoid these problems, LSTM (another variant of RNN) is widely used. Liu et al. (2017) used LSTM to identify entities from clinical text. The LSTM was used to extract the context information from the word representation sequence of the sentence.

Instead of relying on the full sentence, our proposed model considers only the SDP between the target entities as the input. It helps the model to eliminate those words which are not semantically important for deciding the relationship between the entities. Apart from the words in the SDP, we also consider other features like the POS tag information, the dependency label information, and the entity types. We use LSTM to extract the contextual information from the input features. The information generated by the LSTM is passed into a softmax layer to classify the type of relation.

3 Proposed Methodology

Our proposed method for relation extraction is based on LSTM network. At first we develop a dependency parser specific to the clinical text, extract shortest dependency path between the clinical entities using developed parser, and finally use these information for determining the relations between the entities.

We ignore the sentences that do not contain at least two entities. This was done as the relation can exist only between the two entities. These sentence are then passed to the dependency parser that we develop. In the succeeding section, we provide detailed description of each steps. Overall architectural diagram is depicted in Figure 1.

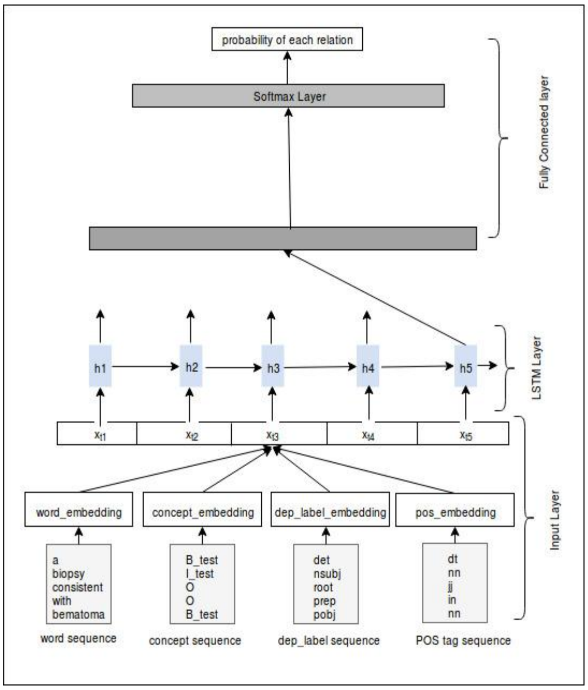

Figure 1: Our proposed architecture

3.1 Dependency Parser

We develop our own dependency parser that is trained on the clinical text. The data used for training the parser is available as the constituency treebank. We convert the constituency treebank into dependency treebank using the Stanford CoreNLP (De Marneffe et al., 2006; Manning et al., 2014). The output dependency trees are in the CoNLL-U format[1]. Our parser is based on the transition based algorithm developed by Nivre (2003). In a transition based system, we convert the gold tree into the labeled configurations. The configurations of the parser are fed into the neural network, and train a model to predict the next transition for the unseen data. The configuration consists of a buffer which stores the words in the sentence which have not been processed yet, a stack which stores the words which have been processed but not yet assigned the head and a set which consists of the dependency labels. From the configuration, the top four words from the stack and their modifiers and first four front words in the buffer, as well as their corresponding PoS tags, are used as the features for our model. There are variations of the transition based system. We follow the arc standard one for our parser. In this system, the initial configuration of the parser is given as, the stack contains a 'Root' word, the buffer contains all the words of the sentences and the dependency label set is empty. This system defines three transitions. Say, S be the stack, B is the buffer, A be the set of the relations and i and j are the words in the sentences. The transitions are defined as follows:

1. LEFT-ARC(L):
   $\{S|i, j; B; A\}$ $=>$ $\{S|j; B; AU\{jLi\}\}$; if the second top word in the stack is dependent on the top word of the stack then, a LEFT-ARC(L) operation is performed. Here, we pop the second top word from the stack and add the relation L between the top and second top element in the stack to the dependency relation set.

2. RIGHT-ARC(L):
   $\{S|i, j; B; A\}$ $=>$ $\{S|i; B; AU\{iLj\}\}$; if the top word is dependent on the second top word of the stack then, a RIGHT-ARC(L) operation is performed. Here, we pop the top word from the stack and add the relation L between the top and second top element in the stack to the dependency relation set.

3. SHIFT:

[1] http://universaldependencies.org/format.html

$\{S|i;B|j;A\} => \{S|i,j;B;A\}$; this operation simply sifts the first word from the buffer to the stack when there is no relation between the top two words in the stack.

To parse a new sentence, we use Genia tagger to get the PoS tags of the tokens[2]. Then, we initialize the parser configuration. The parser provides the next transition by considering this configuration as the input. It will, thus, generate a dependency tree. We evaluate the performance of the parser with the labeled dependency treebank. This labeled dependency treebank is converted from manually annotated constituency treebank. The treebank consists of around fifty-three thousand sentences of clinical text which is provided by an industry named ezDI [3]. The result shows that our parser provides the state-of-the-art accuracies with 93.15 UAS and 92.01 LAS. Where, UAS is the ratio of the count of correctly parsed head position to the total number of the token and LAS is the ratio of the count of correctly parsed head position as well as correctly predicted dependency label to the total number of the token.

3.2 Shortest Dependency Path-SDP

The Shortest Dependency Path (SDP) is the shortest path between the two medical concepts in the dependency tree. For a pair of concepts (c1, c2) in a sentence, we find the shortest path by considering c1 as source vertex and c2 as the target vertex. Though the dependency tree of a sentence is a directed graph, we convert it into an undirected graph when we compute the SDP. In the dependency tree given in the Figure 2 of the sentence given in Example 1.1, the texts shown in the bold are the entities present in the sentence. The relations present in the sentence are TeCP (a biopsy, this mass), TeCP (a biopsy, hematoma) and PIP (hematoma, this mass). Considering the two entities **biopsy** and **hematoma**, the SDP between the two entities is generated as *a biopsy consistent with hematoma*.

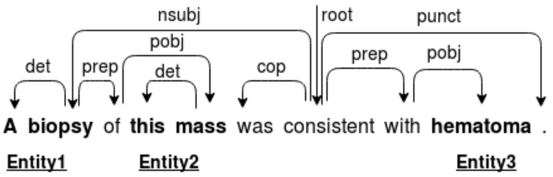

Figure 2: The dependency tree of the sentence given in Example 1.1

The SDP is marked with concept type as *B_Test I_Test O O B_Problem* [4]. We term this sequence as a concept sequence. Now, the sequence of words in the SDP along with the concept sequence, the dependency labels sequence, and the PoS tags sequence are fed into the neural network. We use LSTM to learn the characteristic of the inputs and their contextual information. The model then outputs the probability values corresponding to all these relation types.

3.3 Neural Network Architecture for Relation Extraction

Our proposed architecture which is based on Long Short-Term Memory (LSTM) is composed of an input layer that takes SDP based words, an LSTM layer, and a softmax layer. We depict the proposed architecture in Figure 1.

3.3.1 Input layer

The input layer has an embedding layer which maps each feature into the d-dimensional vectors. The vectors are then concatenated and passed into the LSTM layer. The feature vector of the t$^{\text{th}}$ input is defined as

$$x_{\text{t}} = E(w_{\text{t}}) \oplus E(e_{\text{t}}) \oplus E(d_{\text{t}}) \oplus E(p_{\text{t}})$$

where w_t, e_t, d_t and p_t are the t$^{\text{th}}$ word, concept tag, dependency label and PoS tag in the SDP, respectively. E maps the features into the vectors of dimension d and $\oplus$ is the concatenation operation.

3.3.2 LSTM Layer

This layer takes the output of the embedding layer and computes the characteristic and contextual information of the input. The LSTM networks are a special kind of RNN. Unlike, the traditional neural network, the RNN cell at time t takes the input x_t and previous hidden state $\text{h}_{\text{t-1}}$ to compute the current hidden state h_t. But, RNN is not able to store the information of the long past inputs (Graves, 2013). LSTMs are designed to overcome this issue. The key component in the LSTM is the cell state which carries information throughout the

[2] http://www.nactem.ac.uk/GENIA/tagger/

[3] https://www.ezdi.com/

[4] Here B, I and O denote the beginning, intermediate and outside of an entity

LSTM cells. An LSTM cell is composed of multiple components that control the information flow in the cell state.

The first component is responsible for deciding which information to throw out from the cell state $\mathrm{C}_{\text{t-1}}$. It is controlled by a sigmoid layer called forget gate. It takes the previous hidden state $\mathrm{h}_{\text{t-1}}$ and the current input x_{t} and outputs a value between 0 - 1. Where, 1 stands for completely keeping the information in the cell state and 0 stands for completely ignoring the information. The forget gate is defined by the following equation.

$$f_{\mathrm{t}} = \sigma(W_{\mathrm{f}} \cdot \left[h_{\text{t-1}}, x_{\mathrm{t}}\right] + b_{\mathrm{f}})$$

Where, W_{f} and b_{f} are the weight matrix and the bias vector of the forget gate. Now, the updated information in the cell state is

$$C_{\text{t-1}} = C_{\text{t-1}} * f_{\mathrm{t}}$$

The second component is responsible for deciding what new information to add in the cell state. It is controlled by a sigmoid layer called input gate (i_{t}) and tanh layer. The tanh layer computes a new candidate $\overline{C_{\mathrm{t}}}$ that has to be multiplied component wise with the output of the input gate.

$$i_{\mathrm{t}} = \sigma(W_{\mathrm{i}} \cdot \left[h_{\text{t-1}}, x_{\mathrm{t}}\right] + b_{\mathrm{i}})$$

$$\overline{C_{\mathrm{t}}} = tanh(W_{\mathrm{c}} \cdot \left[h_{\text{t-1}}, x_{\mathrm{t}}\right] + b_{\mathrm{c}})$$

Now, the new cell state value at the current LSTM cell is given by

$$C_{\mathrm{t}} = C_{\text{t-1}} + i_{\mathrm{i}} * \overline{C_{\mathrm{t}}}$$

The last component computes the hidden information of the LSTM cell. It consists of a sigmoid layer called the output gate(o_{t}) and a tanh layer.

$$o_{\mathrm{t}} = \sigma(W_{\mathrm{o}} \cdot \left[h_{\text{t-1}}, x_{\mathrm{t}}\right] + b_{\mathrm{o}})$$

$$h_{\mathrm{t}} = tanh(C_{\mathrm{t}}) * o_{\mathrm{t}}$$

Let H = (h_1, h_2, ... h_{n}) be the hidden states produced by the LSTM layer. Where n is the length of the SDP. Now, the output of the last hidden state is passed fully connected layer for classification.

3.3.3 Softmax layer

Softmax layer is used to classify the information that has been learned from the previous layers into relations. This layer converts the output of the LSTM layer into fixed length vector x of size same as the number of the relation classes n. Then, it computes the probability for all relation classes.

4 Datasets and Experimental Setup

4.1 Dataset

We use the i2b2-2010 relation extraction challenge dataset. The dataset is collected from three different hospitals viz, Partners Healthcare, Beth Israel Deaconess Medical Center, and the University of Pittsburgh Medical Center. It consists of discharge-summary and progress notes of the patients and is manually annotated by medical practitioners. We download the dataset from the i2b2 website. As mentioned in Sahu et al. (2016), we also get only 170 documents for training and 256 documents for testing. But, the actual dataset released during the challenge was 394 documents for training and 477 documents for testing. The training data consists of 5264 relations and the testing data consists of 9069 relations. The Table 1 shows the sample annotation format in i2b2 2010 dataset.

Text	75 He did have burst of atrial fibrillation and was started on a Amiodarone gtt .
Concept	c="burst of atrial fibrillation" 75:3 75:6||t= "problem" c="a amiodarone gtt" 75:11 75:13||t="treatment"
Relation	c="a amiodarone gtt" 75:11 75:13||r="TrAP"||c="burst of atrial fibrillation" 75:3 75:6

Table 1: Sample annotation format in i2b2-2010 dataset

In the first row, the number *75* indicates the line number of the sentence in the document. The dataset consists of 8 relations:

1. TrIP: Treatment improves a medical problem. Example:- *TrIP(po Amidarone||treatment, further episodes of AFIB||problem)* in the sentence, *He had no further episodes of AFIB while on po Amiodarone .*

2. TrWP: Treatment worsens a medical problem. Example:- *TrWP(increased nebulizer treatments||Treatment, upper respiratory like infection||problem)* in the sentence, *This is a 55-year-old female with multiple prior admissions for pneumonia , COPD , asthma exacerbation, over 3 weeks of upper respiratory like infection unremitting with increased nebulizer treatments at home .*

3. TrCP: Treatment causes a medical problem. Example:- *TrCP(Drugs||treatment, Known Allergies||problem)* in the sentence, *Patient*

recorded as having No Known Allergies to Drugs.

4. TrAP: Treatment is administered for a medical problem. Example:- *TrAP(CABG||treatment, MI||problem)* in the sentence, *Father with MI in 50 's and underwent CABG ..*

5. TrNAP: Treatment is not administered for a medical problem. Example:- *TrNAP(ointments||treatment, incisions||problem)* in the sentence, *No creams , lotions , powders , or ointments to incisions*.

6. TeRP: Medical test reveals a medical problem. Example:- *TeRP(CXR||test, left lower lobe atelectasis||problem)* in the sentence, *CXR 10-30 : Left lower lobe atelectasis has partially cleared .*

7. TeCP: Medical test is conducted to investigate a medical problem. Example:- *TeCP(cath||test, abnormal ett||problem)* in the sentence, *67 y/o male with worsening shortness of breath. Had abnormal ETT and referred for cath .*

8. PIP: Medical problem interacts with a medical problem. Example:- *PIP(wounds||problem, infection||problem)* in the sentence, *Monitor wounds for infection - redness , drainage , or increased pain*.

4.2 Experimental Setup

During preprocessing, we filter out all the sentences having entity lesser than two. Since the number of instances is very low in the training set, we perform five-fold cross-validation on the combined dataset of training and test. As mentioned in the above section, we use four features, the word sequence, the concept sequence, the dependency label sequence and the PoS tag sequence of the SDP. For word embedding we use pre-trained word vector trained using word2vec tool (Mikolov et al., 2013) on huge Pubmed articles. This word embedding is downloaded from `http://bio.nlplab.org/`. To generate embeddings for the remaining features i.e. POS tag, dependency label, and the entity type, we train our own vectors using gensim word2vec tool. We set the dimension of the vectors to 50 because of the small vocabulary size. We consider the instances of all the relation types including the NONE class. We train two models, one based on our developed parser and the other based on the state-of-the-art Stanford parser.

4.3 Hyperparameters

We use categorical cross-entropy as loss function for our neural network and the rmsprop as the optimizer. The number of neurons in the LSTM layer is 512. The next hidden layer consists of 256 neurons. We use Relu as an activation function in this layer. The output layer consists of 9 neurons which correspond to relation types. Since the problem is a classification problem, we apply softmax activation function in the output layer. We use dropout value as 0.3 to overcome the overfitting problem. The network is trained with 50 epochs.

5 Experimental Results

All the results shown below are the average of 5-fold experiments. To show the effectiveness of our proposed model, we build two competitive state-of-the-art baseline models. In the first baseline, we train the model with feed forward network and the second baseline model is similar to our proposed model but trained without dependency label and PoS tag features.
From the Table 2, we can see that our proposed model achieves an improvement around 2% F1-score over the first baseline and around 0.56 F1-score over the second baseline.

Model	Precision	recall	F1-score
Baseline model 1	89.97	90.86	90.27
Baseline model 2	92.19	92.43	92.24
Our proposed model	92.79	92.87	92.80

Table 2: Result comparison of our baseline models and the proposed model

5.1 Comparisons with state-of-the-art Systems

There are quite a few existing works which focused on experiments on the i2b2-2010 full dataset, but we found only Sahu et al. (2016) which performed the experiment on the i2b2-2010 partial dataset which is same as the dataset used in our experiment. We are not able to perform the experiments on the whole dataset because the whole dataset was only released during the i2b2-2010 challenge. It consists of 394 documents for training and 477 documents for testing (Rink et al., 2011). But when we download the dataset from the i2b2 website, we got only 170 documents for training and 256 documents for testing. Results shown in Table 3 demonstrates that our proposed model provides better performance for all the relation types except PIP.

Relation Type	Proposed model	Sahu et al. (2016) model
TeCP	**59.13**	50.56
TrCP	**62.13**	56.44
PIP	59.38	**64.92**
TrAP	**75.35**	69.23
TeRP	**83.86**	81.25

Table 3: Comparisons with Sahu et al. (2016)

In order to show how our proposed parser performs against the Stanford parser we develop two variations of our proposed relation extraction model, one by using the dependency relations extracted from our parser and the other by using the relations extracted from the Stanford parser. Performance as shown in Figure 3 shows that for most of the relation types, our parser based model outperforms the Stanford parser based model. Due to space constraint we can't include the result for NONE to the graph. For NONE, our parser based model attained an F-1 score of 96.50 compared to 96.58 F1-score of the Stanford parser.

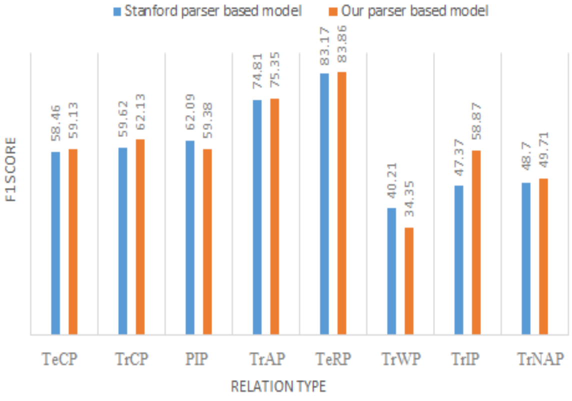

Figure 3: Comparison: Our parser based model vs. Stanford parser based model

Overall evaluation results of our parser based model and the Stanford parser based model are shown in Table 4.

Model	precision	recall	F1-score
Our parser based model	92.79	92.87	92.80
Stanford parser based model	92.88	92.92	92.86

Table 4: Overall evaluation: Our Parser based model vs. Stanford parser based model

5.2 Hypothesis Testing

We show in Table 5 the results of hypothesis testing.

Model	T-Value	P-Value
BL1	26.34559	< .00001
BL2	4.22267	.001453
Stanford parser	-0.41611	.344136

Table 5: Significance testing w.r.t our proposed model. Here, BL1 and BL2 denote the first and the second baseline, respectively

The first row, the second row, and the third row are the result of hypothesis testing of our proposed model against the baselines and the Stanford parser based model, respectively. From the hypothesis testing, it is evident that performance improvement in our proposed model is statistically significant over the baselines. However, we observe that the performance improvement of the Stanford parser based model is not significant over our parser based model. Instead the performance our parser is very close to the Stanford parser based model.

5.3 Error Analysis

We conduct detailed error analysis on our predicted outputs. Below we provide an example for error analysis of the predicted output.

Example 5.1 *# Neurologic - The patient was seen by the Neurology consult service and underwent MRI head which revealed lesions suspicious for metastases, possible hemorrhages, and findings consistent with hypoxic brain injury.*

In this example, the two parsers do not produce the identical outputs. The possible entity pairs present in the sentence are (lesions||Problem, metastases||Problem), (mri head ||Test, hypoxic brain injury||Problem), (mri head||Test, hemorrhages||Problem), (mri head||Test, metastases||Problem), (mri head||Test, lesions||Problem), (lesions||Problem, hypoxic brain injury||Problem), (lesions||problem, hemorage||Problem), (metastases||problem, hypoxic brain injury||Problem), (metastases||problem, hemorage||Problem), and (hemorage||Problem, hypoxic brain injury||Problem).

Actual relation	Our parser based model output	Stanford parser based model output
PIP	**NONE**	PIP
TeRP	TeRP	**NONE**
TeRP	TeRP	TeRP
TeRP	TeRP	TeRP
TeRP	TeRP	TeRP
NONE	NONE	**PIP**
NONE	NONE	NONE
NONE	NONE	NONE
NONE	NONE	NONE
NONE	NONE	NONE

Table 6: Actual relations present in the sentence given in Example 5.1 and corresponding outputs of our parser based model and the Stanford parser based model.

Table 6 shows all the correct relations present in the sentence given in Example 5.1 and the corresponding outputs of our parser based model and the Stanford parser based model. From Table 6, we can see that two parser models produce different outputs for three relations. The possible reasons for the incorrect outputs are noted below:

1. Actual relation is PIP(lesions||Problem, metastases||Problem). Our parser based model yields the output NONE whereas the Standford parser based model's output is PIP. And the SDP generated by both the parsers is “lesions suspicious for metastases”. For this relation, the output of the Stanford parser is correct even if the SDP generated by these two parsers are same.

2. The actual relation is TeRP(mri head||Test, hypoxic brain injury||Problem). Our parser based model's prediction is TeRP and the SDP is “mri head revealed lesions suspicious for metastases findings consistent with hypoxic brain injury”. The Stanford parser based model's prediction is NONE and the SDP is “mri head underwent seen findings consistent with hypoxic brain injury”. In this case, our parser based model provides the correct relation. It may be the reason that our parser based SDP provides a better sequence of words. In the SDP generated by the Stanford parser, the word “findings” depends on the word “seen” which is incorrect because the words “hemorage” and “findings” should depend on the word “metastases” with “conj” relation. But our parser provides correct dependency tree.

3. Actual relation is NONE(lesions||Problem, hypoxic brain injury||Problem) Our parser based model's output is NONE. Our parser based SDP is “lesions suspicious for metastases findings consistent with hypoxic brain injury”. The Stanford parser based model's output is PIP. The Stanford parser based SDP is “lesions revealed head underwent seen findings consistent with hypoxic brain injury”. Here also, our parser based model provides a correct relation type but the Stanford parser based model provides the wrong relation. It may be the reason that our parser based SDP provides a better sequence of words.

In the example 1.1, three entity pairs are present i.e. (a biopsy||test, this mass||problem), (a biopsy||test, hematoma||problem), and (hematoma||problem, this mass||problem). For this example, both the parser gives same dependency tree but the Stanford parser provides a better output. Both the models give a correct relation type for the first entity pairs i.e. TeCP and wrong relation type for the second entity pair i.e. TeRP instead of TeCP. But for the last entity pair, the Stanford parser based model provides correct relation type i.e PIP whereas our parser based model provides wrong relation type i.e. NONE.

6 Conclusion

In this paper we propose an effective model for relation extraction in clinical text. At first we develop a parser for clinical domain, and then use the dependency relations extracted from the parser as a feature to the deep learning model for relation extraction. We consider the SDP generated by the dependency parser a better feature representation for relation extraction. By comparing the results to the baseline models, we can conclude that LSTM based model can extract the contextual and the sequential information from the SDP and the dependency label and PoS tag information can enhance the performance of the model. Our experimental results also show that the overall performance of our parser based model is very close to the overall performance of the Standford parser based model. However, our detailed analysis reveals that our parser based model makes correct prediction for several instances for which Stanford parser based model makes wrong predictions.

References

Asma Ben Abacha, Md Faisal Mahbub Chowdhury, Aikaterini Karanasiou, Yassine Mrabet, Alberto Lavelli, and Pierre Zweigenbaum. 2015. Text mining for pharmacovigilance: Using machine learning for drug name recognition and drug–drug interaction extraction and classification. *Journal of biomedical informatics*, 58:122–132.

Berry de Bruijn, Colin Cherry, Svetlana Kiritchenko, Joel Martin, and Xiaodan Zhu. 2011. Machine-learned solutions for three stages of clinical information extraction: the state of the art at i2b2 2010. *Journal of the American Medical Informatics Association*, 18(5):557–562.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454. Genoa Italy.

Asif Ekbal, Sriparna Saha, Pushpak Bhattacharyya, et al. 2016. A deep learning architecture for protein-protein interaction article identification. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 3128–3133. IEEE.

Alex Graves. 2013. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Ankit Kumar, Asif Ekbal, Sriparna Saha, Pushpak Bhattacharyya, et al. 2016. A recurrent neural network architecture for de-identifying clinical records. In *Proceedings of the 13th International Conference on Natural Language Processing*, pages 188–197.

Zengjian Liu, Ming Yang, Xiaolong Wang, Qingcai Chen, Buzhou Tang, Zhe Wang, and Hua Xu. 2017. Entity recognition from clinical texts via recurrent neural network. *BMC medical informatics and decision making*, 17(2):67.

Yuan Luo, Yu Cheng, ÃŰzlem Uzuner, Peter Szolovits, and Justin Starren. 2018. Segment convolutional neural networks (seg-cnns) for classifying relations in clinical notes. *Journal of the American Medical Informatics Association*, 25(1):93–98.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

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

Anne-Lyse Minard, Anne-Laure Ligozat, and Brigitte Grau. 2011. Multi-class svm for relation extraction from clinical reports. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 604–609.

Joakim Nivre. 2003. An efficient algorithm for projective dependency parsing. In *Proceedings of the 8th International Workshop on Parsing Technologies (IWPT*. Citeseer.

Bryan Rink, Sanda Harabagiu, and Kirk Roberts. 2011. Automatic extraction of relations between medical concepts in clinical texts. *Journal of the American Medical Informatics Association*, 18(5):594–600.

Sunil Kumar Sahu, Ashish Anand, Krishnadev Oruganty, and Mahanandeeshwar Gattu. 2016. Relation extraction from clinical texts using domain invariant convolutional neural network. *arXiv preprint arXiv:1606.09370*.

Ayush Singhal, Michael Simmons, and Zhiyong Lu. 2016. Text mining for precision medicine: automating disease-mutation relationship extraction from biomedical literature. *Journal of the American Medical Informatics Association*, 23(4):766–772.

Ozlem Uzuner, Jonathan Mailoa, Russell Ryan, and Tawanda Sibanda. 2010. Semantic relations for problem-oriented medical records. *Artificial intelligence in medicine*, 50(2):63–73.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2015. Semantic relation classification via convolutional neural networks with simple negative sampling. *arXiv preprint arXiv:1506.07650*.

Shweta Yadav, Asif Ekbal, and Sriparna Saha. 2017a. Feature selection for entity extraction from multiple biomedical corpora: A pso-based approach. *Soft Computing*, pages 1–24.

Shweta Yadav, Asif Ekbal, and Sriparna Saha. 2018a. Information theoretic-pso-based feature selection: an application in biomedical entity extraction. *Knowledge and Information Systems*, pages 1–26.

Shweta Yadav, Asif Ekbal, Sriparna Saha, and Pushpak Bhattacharyya. 2016. Deep learning architecture for patient data de-identification in clinical records. In *Proceedings of the Clinical Natural Language Processing Workshop (ClinicalNLP)*, pages 32–41.

Shweta Yadav, Asif Ekbal, Sriparna Saha, and Pushpak Bhattacharyya. 2017b. Entity extraction in biomedical corpora: An approach to evaluate word embedding features with pso based feature selection. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1159–1170.

Shweta Yadav, Asif Ekbal, Sriparna Saha, Pushpak Bhattacharyya, and Amit Sheth. 2018b. Multi-task learning framework for mining crowd intelligence towards clinical treatment. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 271–277.

Shweta Yadav, Ankit Kumar, Asif Ekbal, Sriparna Saha, and Pushpak Bhattacharyya. 2018c. Feature assisted bi-directional lstm model for protein-protein interaction identification from biomedical texts. *arXiv preprint arXiv:1807.02162*.