Identifying Protein-Protein Interaction using Tree-Transformers and a Heterogeneous Graph Neural Network

Sudipta Singha Roy, Robert E. Mercer
The University of Western Ontario
London, Ontario, Canada
ssinghar@uwo.ca, mercer@csd.uwo.ca

Abstract

For a better understanding of the underlying biological mechanisms, it is crucial to identify the reciprocity between proteins. Often, extracting such interactions between proteins from biomedical articles faces challenges due to the complex sentence structure of the textual information sources. Most of the prominent previous works have applied additional hand-crafted features for the protein-protein interaction task. In this work, we have utilized two tree-structured attention-based neural network models along with a heterogeneous graph approach to perform this task. We suggest that the proposed model preserves the syntactic as well as the semantic information of the text. The experimental results demonstrate that even without using any additional feature extraction techniques, this model achieves significant performance boosts when applied on the five standard benchmark corpora compared to the previous works.

The exponential growth of scientific literature means that the majority of biological information is now in text form and can be found in the scientific literature. The MEDLINE database has seen an increase of more than 4% each year over the past two decennia, and currently holds more than 29 million records from various publications, which is 3 million more than in 2020 and more than 8 million over what it held in 2014 (Yadav et al. 2020). The massive amount of text found in biomedical research articles represents an invaluable resource of information for the field of automated biomedical information retrieval.

Given the exponential growth of biomedical data and the intricate nature of the textual representation of this data, it is crucial to utilize automated methods for information retrieval to assist biologists in finding relevant information, organizing databases, and offering decision support for medical professionals. Several studies have been conducted to extract the information present in these texts, including protein-protein interactions, chemical-disease relationships, clinical relations, drug-drug interactions, and more.

A cell's internal biological activities, including immune response, signal transduction, and cellular organization, are largely a result of interactions between various proteins (Sledzieski et al. 2021). Understanding molecular mechanisms of biological processes requires knowledge of protein-protein interactions (PPI) (Ahmed et al. 2019). These interactions have crucial relevance for biomedical fields, including the examination of drug targets (Gordon et al. 2020) and signal proteins (Altmann et al. 2020). Therefore, recognizing protein-protein interactions (PPIs) leads to a deeper comprehension of the functions, control, and communication between various proteins (Yao et al. 2019). The objective of recognizing PPIs is to extract the relationships between protein entities mentioned in a document (Krallinger et al. 2008).

A significant amount of information regarding PPIs is present in biomedical literature, but in an unstructured form. Manually extracting PPIs is a demanding task, both in terms of time and cost, due to the large number of published studies (Peng, Wei, and Lu 2016; Tang et al. 2022). As a result, automatically extracting PPIs from biomedical literature has become a crucial research area, garnering attention from many researchers. The information could be dispersed throughout the document, however, the current study is limited to detecting only the PPIs within individual sentences similar to many previous works (Pyysalo et al. 2008; Tikk et al. 2010; Ahmed et al. 2019). As an example of a sentence containing interactions between proteins (Howard et al. 2000):

> "At 89.3 nmol/L, maximal migration of CCR1 and CCR8 transfected cells was prompted by LEC and at 5.6 nmol/L, cell adhesion also occurred."

This sentence reflects two protein-protein interactions involving LEC and CCR1, as well as LEC and CCR8. But, importantly, no correlation is present between proteins CCR1 and CCR8.

In the early research phase, the commonly used methods for PPI extraction involved utilizing co-occurrence and pattern recognition techniques (Baumgartner et al. 2008; Yu et al. 2018). However, recent advancements in technology have led to the widespread adoption of machine learning techniques which have superior performance compared to these traditional methods. Early approaches involved constructing a feature set through feature engineering and kernel methods and then applying support vector machines or other classifiers for classification (Airola et al. 2008b; Murugesan, Abdulkadhar, and Natarajan 2017). In the last

few years, several research works (Zhao et al. 2016; Hua and Quan 2016; Choi 2016) have successfully applied deep learning techniques to PPI extraction, taking advantage of the widespread use of deep learning in NLP.

Most of the recent works utilize recurrent neural network (RNN) models for this task considering textual representations as sequences (Hsieh et al. 2017; Yadav et al. 2019). However, if the data is arranged in a structured format instead of being arranged in a sequence, these models are prone to miss the semantic compositions present within (Ahmed, Samee, and Mercer 2019a). This is due to the fact that they only take into account the word order and ignore the linguistic structure (Li et al. 2015). Contrarily, recursive neural networks, also known as tree-structured neural network models (Ahmed, Samee, and Mercer 2019b), process the sentences represented in a parsed tree form, thereby keeping both the syntax and semantics in a more effective way. Investigations have also taken place regarding graph-based methods for this task, where the models operate on either a fully connected graph composed of word nodes or on text segments of phrases (Fei et al. 2021). Our proposed model assembles these last two methods in a novel design.

While extracting relations between target proteins, we have considered three issues: firstly, how to retrieve the relation if the considered proteins are mentioned far apart in the text, secondly, how to deal with the phrasal structure of text in order to preserve the semantics so that the PPI extraction can attend to this information, and thirdly, what will happen if instead of using fixed word representations from pre-trained models, we update the word representations based on the considered sentence and then use these updated representations to impact the generated sentence representation for this task.

Uniting these considerations, we have investigated a model combining dependency and constituency tree transformers (Ahmed, Samee, and Mercer 2019b) and a heterogeneous graph attention network (Wang et al. 2020) for the PPI extraction task. The dependency tree-transformer captures the correlations between words at different parts of the sentence which allows the model to extract relations between the considered proteins even if they are positioned far apart in the sentence. For preserving the phrasal information we have used the constituency tree-transformer. And for word to sentence representation and sentence to word representation updates, we have utilized a heterogeneous graph neural network.[1] We provide a comprehensive analysis of the performance of these models on benchmark PPI datasets, which showcases the superiority of the proposed model over the previous prominent works.

Related Work

Several NLP techniques have emerged for determining links between proteins. At first, pattern-based methods were popular, where rules were established based on syntax and lexical features for finding relationships (Blaschke et al. 1999; Leeuwenberg et al. 2015). But, these models couldn't manage complex relationships expressed in relational and coordinating clauses correctly. Unlike simple pattern-based approaches, dependency-based methods are more focused on syntax and can be applied to a broader range of situations (Erkan, Özgür, and Radev 2007; Miyao et al. 2009).

Another common method for identifying correlations between proteins is the use of kernel-based techniques. These models acquire rich structural information through dependency structures and syntactic parse trees (Singha Roy and Mercer 2022). Airola et al. (2008a) suggested a method for identifying interactions between target proteins by examining information from linear and dependency subgraphs. Miwa et al. (2009) developed a system that incorporates a Support Vector Machine with weighted feature vectors derived from multiple corpora. Kim et al. (2010) matched e-walks and v-walks on the shortest dependency path to acquire non-contiguous syntactic structures by means of a walk-weighted sub-sequence kernel for this task. Zhang et al. introduced a neighbourhood hash graph kernel-based model to draw out PPIs. Chang et al. (2016) used a convolution tree kernel and PPI patterns to extract interlinkages between proteins. Murugesan et al. (2017) proposed the distributed smoothed tree kernel which has demonstrated substantial advancements when compared to other kernel methods for this task.

The recent surge in deep learning models has resulted in a plethora of experiments aimed at uncovering PPI relationships from biomedical literature (Quan et al. 2016; Hsieh et al. 2017; Zhang et al. 2018). Zhao et al. (2016) were the first to apply deep learning in the area of PPI relation extraction. Their approach involved training an autoencoder on unclassified training data to prepare the parameters for a multi-layer perceptron (MLP) model, which was then optimized through gradient descent to carry out PPI extraction. Peng and Lu (2017) involved the utilization of a double-channel CNN for this task. The first channel incorporated syntax-based features like syntactic dependencies, parts of speech, named entities, the distance of each word from the two proteins interacting, chunk parsing details, and the word itself. The second channel utilized a convolution process with respect to the parent word information for each word. The second channel provides a distributed representation of the sentence by applying convolution over each word's parent information. For PPI extraction, a three-channel CNN was implemented by Zhang et al. (2018). Convolution operations were carried out on the original words along with positional encoding, the shortest dependency path, and encoding features for dependency relations in each of the first, second, and third channels, respectively. Zhang et al. (2019) showed that using residual connections improves the performance of the CNN-based models when extracting PPIs from texts.

Since then, a series of studies have been carried out on the PPI task, utilizing Recurrent Neural Networks (RNNs), which have been seen to excel in processing sequential data. Hsieh et al. (2017), to generate a sentence vector representation, concatenated the left and right-most output vectors from a Bi-LSTM which was fed with the sentence, and then

[1]The code is available on https://github.com/sudipta90/PPI_Heterogenous.git

applied a softmax classifier for the classification task. Yadav et al. (2019) fed the shortest dependency information between unit pairs as input to a Bi-LSTM with structured attention. For their subsequent study, Yadav et al. (2020) implemented a self-attentive approach for performing two tasks simultaneously: extraction of protein-protein interactions and extraction of drug-drug interactions. Ahmed et al. (2019) applied structured attention over dependency tree-LSTMs for this task and showed the supremacy of the tree-structured neural networks over sequential models. Fei et al. (2021) introduced a span-graph neural architecture for extracting protein entity relations from biomedical texts. Their model jointly learns to identify the candidate entity spans and the correlaton between them. The entity graph is constructed by listing out probable entity span possibilities.

Proposed Model

In this portion of the paper, we delve into the specifics of our model. Our study of the protein-protein interaction extraction task utilizes two tree-structured neural networks: dependency and constituency tree-transformers (Ahmed, Samee, and Mercer 2019b); and a heterogeneous graph attention network (Wang et al. 2020). How each network functions is initially discussed. The discussion then moves on to cover the proposed model combining these modules.

Tree-Transformers

Two tree-based representations exist for representing a sentence: constituency trees and dependency trees. These forms of representation offer syntactic information about the sentence, capturing both the structure of phrases (constituency tree) and the dependencies between individual words (dependency tree). Ahmed, Samee, and Mercer (2019b) suggested two tree-transformer models: dependency and constituency tree-transformers utilizing these sources of syntactic structure information. The objective of these models is to traverse each sub-tree within a dependency or constituency tree structure, attentively, and derive a vector representation at its root.

Each node in a dependency tree holds a word. To traverse a sub-tree in this kind of tree, the dependency tree-transformer considers both the parent and child node representations. Conversely, in a constituency tree, only the leaf nodes hold words. The non-terminal node vectors are computed only after the sub-tree has been fully traversed. Ahmed, Samee, and Mercer (2019b) applied self-attention to the sentence's dependency and constituency tree representations, incorporating *query* ($\mathcal{Q}$), *key* ($\mathcal{K}$) and *value* ($\mathcal{V}$) matrices. These matrices are computed as follows (Vaswani et al. 2017):

$$\mathcal{K} = \omega_k \mathcal{M}_k \quad \text{s.t.} \quad \omega_k \in \mathbb{R}^{d \times d} \tag{1}$$

$$\mathcal{V} = \omega_v \mathcal{M}_v \quad \text{s.t.} \quad \omega_v \in \mathbb{R}^{d \times d} \tag{2}$$

$$\mathcal{Q} = \omega_q \mathcal{M}_q \quad \text{s.t.} \quad \omega_q \in \mathbb{R}^{d \times d} \tag{3}$$

In the dependency tree, the matrix $\mathcal{M}$ is formed by concatenating the word vectors of all child nodes for each corresponding parent node. On the other hand, for the constituency tree, $\mathcal{M}$ is the concatenation of the word vectors within a constituent. Using $\mathcal{Q}$, $\mathcal{K}$ and $\mathcal{V}$ matrices, the tree-transformer models compute the self attention matrix as follows:

$$\alpha = \text{softmax}(\frac{\mathcal{Q}\,\mathcal{K}^T}{\sqrt{d_k}})\mathcal{V} \tag{4}$$

where d_k refers to the dimension of $\mathcal{K}$. To implement the multi-branch attention $\mathcal{B}_i$ with *n* branches, *n* copies of *key*, *query*, and *value* matrices are generated using the appropriate weight matrices (ω_i). In the end, a scaled dot product attention (as per Eq. 4) is applied to each branch (Eq. 5).

$$\mathcal{B}_i = \alpha_{i\in[1,n]}(\textit{query}_i\ \omega_i^{\textit{query}}, \textit{key}_i\ \omega_i^{\textit{key}}, \textit{value}_i\ \omega_i^{\textit{value}}) \tag{5}$$

Afterwards, a residual connection is utilized on these tensors and a batch normalization layer is applied layer-wise, subsequently. Then, a scaling factor μ is employed to generate the branch representation as follows:

$$\tilde{\mathcal{B}}_i = \texttt{LayerNorm}(\mathcal{B}_i\omega_i^b + \mathcal{B}_i) \times \mu_i \tag{6}$$

Subsequently, a position-wise CNN (PCNN) is employed to every $\tilde{\mathcal{B}}_i$. This PCNN layer consists of two convolution operations on each position with a `ReLU` activation function in between. This PCNN layer works as Eq. 7:

$$\texttt{PCNN}(x) = \texttt{Conv}(\texttt{ReLU}(\texttt{Conv}(x) + b_1)) + b_2 \tag{7}$$

The final attentive representation of these semantic subspaces, generated from the PCNN layer, is obtained by performing a linear weighted summation (Eq. 8) where $\gamma \in \mathbb{R}^n$ is a model hyper-parameter.

$$\texttt{BranchAttn} = \sum_{i=1}^{n} \gamma_i \texttt{PCNN}(\tilde{\mathcal{B}}_i) \tag{8}$$

In the last step, a residual connection is established with `BranchAttn` and non-linearity (`tanh`) is applied. The parent node representation is achieved by performing element-wise summation (`ExS`). Eq. 9 represents the operation of this step.

$$\texttt{ParentNode} = \texttt{EWS}(\tanh((\chi_{attn} + \chi)\omega + b)) \tag{9}$$

In Eq. 9, χ and χ_{attn} symbolize the input and output features of the attention computation module.

Heterogeneous Graph Attention Network

The heterogeneous graph attention network (H-GAT) (Wang et al. 2020) was initially introduced for the textual summarization task to provide enriched cross-sentence relationships. In this work, we have utilized this approach to improve the sentence representation quality. At each iteration, this module is deployed once the constituency and dependency tree-transformers' forward passes are done. Via sentence-to-word and word-to-sentence update processes, this module provides enriched sentence vectors.

For this module the graph G has been structured as $G = \{V, E\}$. The set V represents the nodes in the graph, while E represents the edges between those nodes. For any sentence S containing n words (w_i), $V = \{w_1, w_2, ..., w_n, S\}$.

As this task finds protein-protein interactions in single sentences, the edges are established in such a way that the sentence node S is connected to every word node w_i. Once the graph G has been constructed, a Graph Attention Network (GAT) (Veličković et al. 2017) is employed to modify the feature values of the nodes. Let $h_i \in \mathbb{R}^{d_h}$ be the hidden states of the word and sentence nodes, where $i \in \{1 : (n+1)\}$ and d_h is the hidden state dimension. Then the GAT layer can be represented as:

$$\kappa_{i,j} = \texttt{LeakyReLU}(\omega_a[\omega_q h_i; \omega_k h_j]) \tag{10}$$

$$\alpha_{i,j} = \frac{exp(\kappa_{i,j})}{\sum_{l \in \mathcal{N}_i} exp(\kappa_{i,l})} \tag{11}$$

$$\mathcal{Z}_i = \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{i,j} \omega_v h_j) \tag{12}$$

where the ω_a, ω_q, ω_k, and ω_v weight-matrices are updated via backpropagation. The set of neighbouring nodes for any considered node is represented by $\mathcal{N}_i$. The attention score between h_i and h_j is represented by $\alpha_{i,j}$. The GAT incorporating multi-head attention, with $\mathcal{M}$ attention heads, can be defined as:

$$\mathcal{Z}^i = ||_{m=1}^{\mathcal{M}} \sigma(\sum_{j \in \mathcal{N}_i} \alpha_{i,j}^m \omega^m h_i) \tag{13}$$

In order to avoid the vanishing of gradients over time, a residual connection is also established. With the information u_i from this residual connection, the final hidden state representation is formulated as follows:

$$h_i = u_i + h_i \tag{14}$$

By means of the previously described GAT and a position-wise feed forward network (FFN) layer, which consists of two linear transformations (Wang et al. 2020), the word nodes are updated based on the information from the sentence node seen in Eqs. 15 and 16:

$$\mathcal{Z}_{s \to w}^1 = GAT(\mathcal{H}_w^0, \mathcal{H}_s^0, \mathcal{H}_s^0) \tag{15}$$

$$\mathcal{H}_w^1 = FFN(\mathcal{Z}_{s \to w}^1 + \mathcal{H}_s^0) \tag{16}$$

where $\mathcal{H}_w^0$ is the set of word nodes (the Bio-RoBERTa-based embeddings for words (Gururangan et al. 2020)) for the words present in the sentence. $\mathcal{H}_s^0$ is the average of the sentence representations from the dependency and constituency tree-transformers. In Eq. 15, $\mathcal{H}_w^0$ has been considered as the query matrix and $\mathcal{H}_s^0$ has been considered as both the key and value matrices following the work of Vaswani et al. (2017).

After updating the word nodes based on the sentence node, the next step involves updating the sentence node based on the just updated word nodes. These sentence-to-word and word-to-sentence node refinement processes continue at each iteration. For the t-th iteration, the process can be represented in the following manner:

$$\mathcal{Z}_{s \to w}^{t+1} = GAT(\mathcal{H}_w^t, \mathcal{H}_s^t, \mathcal{H}_s^t) \tag{17}$$

$$\mathcal{H}_w^{t+1} = FFN(\mathcal{Z}_{s \to w}^{t+1} + \mathcal{H}_w^t) \tag{18}$$

$$\mathcal{Z}_{w \to s}^{t+1} = GAT(\mathcal{H}_s^t, \mathcal{H}_w^{t+1}, \mathcal{H}_w^{t+1}) \tag{19}$$

$$\mathcal{H}_s^{t+1} = FFN(\mathcal{Z}_{w \to s}^{t+1} + \mathcal{H}_s^t) \tag{20}$$

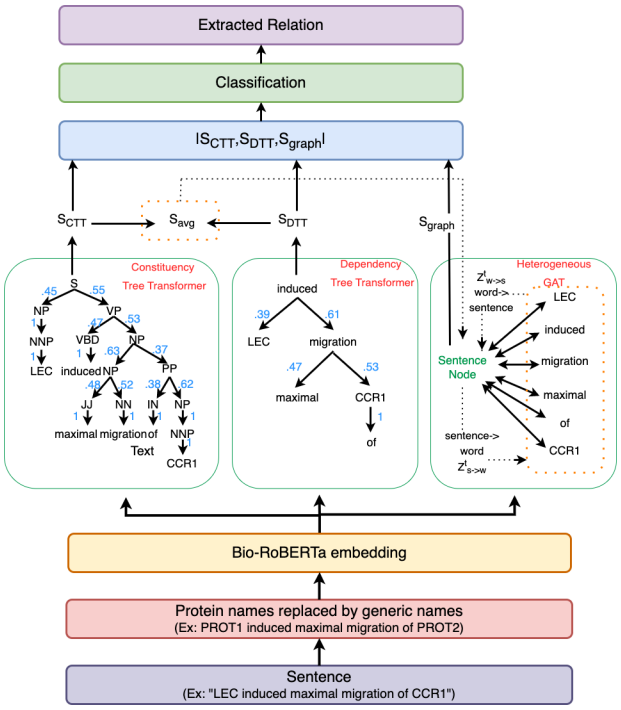

Figure 1: This diagram illustrates the approach for combining features from the dependency and constituency tree-transformers together with a heterogeneous graph attention network to create an integrated architecture for PPI prediction. The blue numbers in the Constituency and Dependency Tree Transformers indicate the attention value for the associated tree branches.

Model Architecture

Figure 1 sketches the overall architecture of the model. Each unit of the model is initially fed with the Bio-RoBERTa (Gururangan et al. 2020) word embeddings. Then the constituency and dependency tree transformers generate the sentence representations (S_{CTT} and S_{DTT}, respectively) in parallel. A point-wise average operation is applied to these two sentence vectors. This averaged sentence vector (S_{avg}) is then used for the sentence-to-word and word-to-sentence update steps in the heterogeneous graph attention network. This step provides another sentence representation (S_{graph}). In the following step, max-pooling is applied and followed by a multi-layer perceptron for the PPI extraction.

Experimental Details and Performance Analysis

In this section, we present the performance of the proposed model, evaluated using the F1-score. We have formulated the PPI extraction as a classification task. We conclude by comparing the efficacy of the proposed model to the leading sequential, tree-structured, and graph-based architectures that have been previously proposed for the PPI task. We first include a statistical overview of the five primary PPI corpora utilized in this task, as well as a discussion of

the pre-processing techniques employed on these corpora.

Corpus Description

First, to assess the performance of the examined model, we evaluate its performance on the five standard PPI benchmark corpora: BioInfer (Pyysalo et al. 2007), AIMed (Bunescu et al. 2005), HPRD50 (Fundel, Küffner, and Zimmer 2007), IEPA (Ding et al. 2001), and LLL (Nédellec 2005). In all of the experiments, the following transformed version of each corpus is employed, as specified by Ahmed et al. (2019) and Singha Roy and Mercer (2022). To provide a consistent classification task across all five corpora, all protein names are replaced with three distinct symbols: if a pair of proteins are to be considered as potentially interacting in a given sentence, they are substituted with the labels PROT1 and PROT2 and all other proteins mentioned in the sentence are substituted with PROT0. Thus, this approach has the model consider an interaction between two proteins, one at a time. To work with sentences containing more than two proteins, two proteins at a time are tagged with PROT1 and PROT2 and their interaction (positive or negative) is identified. Sequentially, all protein pairs are considered. So, for each sentence in the corpus containing η proteins, the modified corpus will feature ${}^{\eta}C_2$ variations. As an example, consider the following sentence: "At 89.3 nmol/L, maximal migration of CCR1 and CCR8 transfected cells was prompted by LEC and at 5.6 nmol/L, cell adhesion also occurred." To identify the possible relationship between LEC and CCR1, we replace their respective protein names with PROT1 and PROT2, while replacing CCR8 with PROT0. When the objective is to identify the possible interaction between LEC and CCR8, we replace their names with PROT1 and PROT2, and use PROT0 in place of CCR1. Similarly, when identifying the possible interaction between CCR1 and CCR8, they are replaced with PROT1 and PROT2 and LEC is replaced with PROT0. Interactions between protein pairs can be either positive or negative. For the above example, when the considered proteins are CCR1 and LEC or CCR8 and LEC, the nature of their interactions is positive in each case. However, when the considered protein pair is CCR1 and CCR8, the PPI is negative since no interaction is present between them. Thus the example sentence presents three possible interactions, resulting in three variants (${}^{3}C_2$) of the sentence in the modified corpus: two with positive interactions, and one with a negative interaction. Using generic names to represent protein names enhances the data by allowing for multiple samples of these generic names, as opposed to only a few samples for each individual protein name. Table 1 provides an overview of the demographic characteristics of the five modified corpora applying this above-mentioned approach. We have utilized the Stanford dependency and constituency parser to parse sentences in each corpus (Manning et al. 2014).

Experimental Setup

Next, turning to the details of the model, it uses an initial learning rate of 0.1. If the validation accuracy decreases from the previous iteration, the learning rate is reduced by 80% in each iteration. We set the batch size to 10.

Table 1: Statistics of the modified corpora

Corpus	Original Sentences	Positive Interactions	Negative Interactions
AIMED	1,995	1,000	4,834
BioInfer	1,100	2,534	7,132
IEPA	486	335	482
HPRD50	145	163	270
LLL	77	164	166

The tree-transformer models use six PCNN layers and six branches of attention layer, and employ 341-dimension and 300-dimension kernels in two CNN layers with dropout 0.1 in the second layer only. For the H-GAT unit, six attention heads are utilized. The model hyper-parameters are trained by the 'Adagrad' (Lydia and Francis 2019) optimizer. The final sentence representation of each individual unit as well as the model is a 512-dimensional vector. The model is fed with Bio-RoBERTa word embeddings. For the tree-transformers, these embeddings are not further updated. But, for the H-GAT unit, with the sentence-to-word update step, word embeddings are updated once each epoch. We have also experimented with PubMed-BERT (Gu et al. 2020), however, better performance is acquired when Bio-RoBERTa word embeddings are used. We have utilized StratifiedK-Fold from the scikit-learn package to perform 10-fold cross-validation in the model evaluation process. For each fold, the training was done on the training set and the test was done on a separate test set.

All of the experiments are performed on Linux Ubuntu 22.04 LTE with 16GB memory and Nvidia 1070Ti 8GB graphics memory. For implementing the model, we have used PyTorch 1.7.1. In this environment, the model took 8 hours each for training on the BioInfer and AIMed corpora.

Performance Analysis

Table 2 displays how our proposed model performs on the five benchmark corpora, along with the published results of several sequential, tree-structured, and graph-based models for comparison. For performance evaluation, we have used the F1-score. With the AIMED corpus, we have achieved 91.23% F1-score, which is a 2.96 percentage point (p.p.) performance boost compared to the current state of the art (Fei et al. 2021). The second dataset that has been used to evaluate the model is BioInfer. It has the highest number of annotated interactions compared to the other four datasets. In this dataset the sentences are notably longer and encompass a greater number of protein names mentioned within a single sentence. On this corpus, our model has achieved a F1-score of 96.97% which is 0.76 p.p. and 0.96 p.p. higher than Fei et al. (2021) and Singha Roy and Mercer (2022), respectively. The three remaining corpora (IEPA, HPRD50, and LLL) come with comparably smaller number of samples. Even for these corpora with very few samples our model has outperformed the current state of the art model (Fei et al. 2021) for the PPI task. Compared to Singha Roy and Mercer (2022), which is the best performing tree-structured model for the PPI extraction task, our

Table 2: Performance evaluation of the models by means of F1-score (in %). The sequential, tree-structured, and graph-based models are tagged with †, ‡, and ∗, accordingly. The performance metric of our model is presented in **bold**.

Methods	AIMed	BioInfer	IEPA	HPRD50	LLL	Avg.
Chang et al. (2016) †	60.6	69.4	71.4	71.5	80.6	70.7
Hsieh et al. (2017) †	76.9	87.2	76.31	80.51	78.3	79.84
Zhang et al. (2018) †	56.4	61.3	75.1	63.4	76.5	66.54
Yadav et al. (2020) †	77.33	76.33	-	-	-	76.83
Tai, Socher, and Manning (2015) ‡	80.6	88.1	76.4	82.0	84.8	82.38
Ahmed et al. (2019) ‡	81.6	89.1	78.5	81.3	84.2	82.94
Singha Roy and Mercer (2022) ‡	88.15	96.01	83.24	88.94	92.18	89.70
Fei et al. (2021) ∗	88.27	96.21	83.90	89.57	92.86	90.16
Proposed Model	**91.23**	**96.97**	**87.28**	**93.11**	**93.52**	**92.02**

Table 3: Cross-corpus experimental results by means of F1-score (in %). The training corpora are represented by the rows, while the testing data is presented in the columns. Rows marked with † are the results from Ahmed et al. (2019).

	AIMed	*BioInfer*	*IEPA*	*HPRD50*	*LLL*
AIMED †	-	47.0	38.6	41.5	34.6
BioInfer †	50.8	-	40.8	45.5	33.5
AIMED	-	55.1	42.7	46.2	39.5
BioInfer	56.7	-	44.0	50.3	40.8

model has gained 4.04 p.p., 4.17 p.p., and 1.34 p.p. higher F1-scores for the IEPA, HPRD50, and LLL corpora, accordingly. In comparison to the work of Fei et al. (2021), for these three corpora, in the order given above, the performance boosts for our model are 3.38 p.p., 3.54 p.p., and 0.66 p.p. On average, over these five corpora, our model has achieved 92.02% F1-score which is 1.86 p.p. higher than what is reported in Fei et al. (2021).

Further to these performance numbers, it is noteworthy that if we discard the H-GAT module, our proposed model is almost identical to the model presented in the work of Singha Roy and Mercer (2022). Comparing the F1-scores of these two models in Table 2, we can see that the sentence-to-word and word-to-sentence update processes are key to the improvement seen in the performance of our new model. This enhanced performance, we believe, is because when the H-GAT module is being employed, the sentence representations generated by the tree-transformers, and thus the newly generated word representations by the sentence-to-word update step, provide a more enriched semantics for the task which in turn help to produce (due to the word-to-sentence process) a better sentence representation for the following classifier. Our belief is further supported by noting that the max-pooling layer, having replaced a sentence feature concatenation layer in a previous version of our model, results in a 0.5-1.8 p.p. performance boost compared to the previous version (previous model's numbers not shown).

Cross-Corpus Performance Analysis

In addition, we have performed a cross-corpus assessment, motivated by Van Landeghem et al. (2008), which aims to address a critical inquiry regarding the effective extraction of protein-protein interactions in practical applications – "which corpus is most suitable for the training of a specific model in real-world scenarios?". Table 3 shows the results achieved by our model for the cross-corpus evaluation. The training data is represented by the rows, and the test data is represented by the columns. In this study, we utilized AIMed and BioInfer exclusively as the training datasets while disregarding the smaller ones. This is because training on small and simple corpora and testing on larger, more intricate datasets serves no practical purpose (Peng and Lu 2017; Ahmed et al. 2019). The results show a noticeable decline in performance across all corpora due to the lack of consistency between the distribution of the training and testing data. The acquired results support the basic principle of machine learning, which states that training and test sets should have identical distributions. Notably, our proposed model trained on BioInfer outperforms the same model trained on AIMed, likely due to the former's larger size. The results also show that, for our model, if it is used in real life scenarios, BioInfer should be the suggested corpus for model training. Furthermore, these transfer learning results show a performance boost compared with Ahmed et al. (2019).

Conclusion and Future Work

In this paper, we have proposed a supervised Protein-protein interaction extractor model which has the ability to obtain the word level dependencies, phrasal information, and better semantics by means of utilizing dependency and constituency tree-transformers and a heterogeneous graph neural network. Our model has shown significant performance improvements on all five benchmark corpora.

Despite the progress made in this work, there is still room for further improvement. The sentence-to-word and word-to-sentence node update step can be applied directly over the tree-transformers to see how they perform. Additional analysis of the results can be conducted by examining the AUC and ROC curves.

References

Ahmed, M.; Islam, J.; Samee, M. R.; and Mercer, R. E. 2019. Identifying protein-protein interaction using tree LSTM and structured attention. In *2019 IEEE 13th Int. Conf. on Semantic Computing (ICSC)*, 224–231.

Ahmed, M.; Samee, M. R.; and Mercer, R. E. 2019a. Improving tree-LSTM with tree attention. In *2019 IEEE 13th Int. Conf. on Semantic Computing (ICSC)*, 247–254.

Ahmed, M.; Samee, M. R.; and Mercer, R. E. 2019b. You only need attention to traverse trees. In *Proc. 57th Ann. Meet. of the Assoc. for Computational Linguistics*, 316–322.

Airola, A.; Pyysalo, S.; Björne, J.; Pahikkala, T.; Ginter, F.; and Salakoski, T. 2008a. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC Bioinformatics* 9(11):1–12.

Airola, A.; Pyysalo, S.; Björne, J.; Pahikkala, T.; Ginter, F.; and Salakoski, T. 2008b. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC bioinformatics* 9 Suppl 11:S2.

Altmann, M.; Altmann, S.; Rodriguez, P. A.; Weller, B.; Elorduy Vergara, L.; Palme, J.; Marín-de la Rosa, N.; Sauer, M.; Wenig, M.; Villaécija-Aguilar, J. A.; et al. 2020. Extensive signal integration by the phytohormone protein network. *Nature* 583(7815):271–276.

Baumgartner, W. A.; Lu, Z.; Johnson, H. L.; Caporaso, J. G.; Paquette, J.; Lindemann, A.; White, E. K.; Medvedeva, O.; Cohen, K. B.; and Hunter, L. 2008. Concept recognition for extracting protein interaction relations from biomedical text. *Genome biology* 9(2):1–15.

Blaschke, C.; Andrade, M. A.; Ouzounis, C. A.; and Valencia, A. 1999. Automatic extraction of biological information from scientific text: protein-protein interactions. In *Seventh Int. Conf. on Intell. Systems for Molecular Biology*, 60–67.

Bunescu, R.; Ge, R.; Kate, R.; Marcotte, E.; Mooney, R.; Ramani, A.; and Wong, Y. W. 2005. Comparative experiments on learning information extractors for proteins and their interactions. *Art. Intell. in Medicine* 33(2):139–155.

Chang, Y.-C.; Chu, C.-H.; Su, Y.-C.; Chen, C. C.; and Hsu, W.-L. 2016. Pipe: a protein–protein interaction passage extraction module for biocreative challenge. *Database* 2016.

Choi, S.-P. 2016. Extraction of protein-protein interactions (ppis) from the literature by deep convolutional neural networks with various feature embeddings. *Journal of Information Science* 44.

Ding, J.; Berleant, D.; Nettleton, D.; and Wurtele, E. 2001. Mining medline: abstracts, sentences, or phrases? In *Biocomputing 2002*. World Scientific. 326–337.

Erkan, G.; Özgür, A.; and Radev, D. 2007. Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In *Proc. of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 228–237.

Fei, H.; Zhang, Y.; Ren, Y.; and Ji, D. 2021. A span-graph neural model for overlapping entity relation extraction in biomedical texts. *Bioinformatics* 37(11):1581–1589.

Fundel, K.; Küffner, R.; and Zimmer, R. 2007. Relex—relation extraction using dependency parse trees. *Bioinformatics* 23(3):365–371.

Gordon, D. E.; Jang, G. M.; Bouhaddou, M.; Xu, J.; Obernier, K.; White, K. M.; O'Meara, M. J.; Rezelj, V. V.; Guo, J. Z.; Swaney, D. L.; et al. 2020. A sars-cov-2 protein interaction map reveals targets for drug repurposing. *Nature* 583(7816):459–468.

Gu, Y.; Tinn, R.; Cheng, H.; Lucas, M.; Usuyama, N.; Liu, X.; Naumann, T.; Gao, J.; and Poon, H. 2020. Domain-specific language model pretraining for biomedical natural language processing.

Gururangan, S.; Marasović, A.; Swayamdipta, S.; Lo, K.; Beltagy, I.; Downey, D.; and Smith, N. A. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 8342–8360.

Howard, O. Z.; Dong, H. F.; Shirakawa, A.-K.; and Oppenheim, J. J. 2000. LEC induces chemotaxis and adhesion by interacting with CCR1 and CCR8. *Blood, The Journal of the American Society of Hematology* 96(3):840–845.

Hsieh, Y.-L.; Chang, Y.-C.; Chang, N.-W.; and Hsu, W.-L. 2017. Identifying protein-protein interactions in biomedical literature using recurrent neural networks with long short-term memory. In *Proc. of the Eighth Int. Joint Conf. on Natural Language Proc. (Vol. 2: Short Papers)*, 240–245.

Hua, L., and Quan, C. 2016. A shortest dependency path based convolutional neural network for protein-protein relation extraction. *BioMed research international* 2016.

Kim, S.; Yoon, J.; Yang, J.; and Park, S. 2010. Walk-weighted subsequence kernels for protein-protein interaction extraction. *BMC Bioinformatics* 11(1):1–21.

Krallinger, M.; Leitner, F.; Rodriguez-Penagos, C.; and Valencia, A. 2008. Overview of the protein-protein interaction annotation extraction task of BioCreative II. *Genome Biology* 9(2):1–19.

Leeuwenberg, A.; Buzmakov, A.; Toussaint, Y.; and Napoli, A. 2015. Exploring pattern structures of syntactic trees for relation extraction. In *International Conference on Formal Concept Analysis*, 153–168.

Li, J.; Luong, M.-T.; Jurafsky, D.; and Hovy, E. 2015. When are tree structures necessary for deep learning of representations? *arXiv preprint arXiv:1503.00185*.

Lydia, A., and Francis, S. 2019. Adagrad—an optimizer for stochastic gradient descent. *Int. J. Inf. Comput. Sci* 6(5):566–568.

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J. R.; Bethard, S.; and McClosky, D. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, 55–60.

Miwa, M.; Sætre, R.; Miyao, Y.; and Tsujii, J. 2009. A rich feature vector for protein-protein interaction extraction from multiple corpora. In *Proc. of the 2009 Conf. on Empirical Methods in Natural Language Processing*, 121–130.

Miyao, Y.; Sagae, K.; Sætre, R.; Matsuzaki, T.; and Tsujii, J. 2009. Evaluating contributions of natural language parsers to protein–protein interaction extraction. *Bioinformatics* 25(3):394–400.

Murugesan, G.; Abdulkadhar, S.; and Natarajan, J. 2017. Distributed smoothed tree kernel for protein-protein interaction extraction from the biomedical literature. *PLOS ONE* 12:e0187379.

Nédellec, C. 2005. Learning language in logic-genic interaction extraction challenge. In *Proceedings of the Learning Language in Logic 2005 Workshop (LLL05)*, 31–37.

Peng, Y., and Lu, Z. 2017. Deep learning for extracting protein-protein interactions from biomedical literature. *arXiv preprint arXiv:1706.01556*.

Peng, Y.; Wei, C.-H.; and Lu, Z. 2016. Improving chemical disease relation extraction with rich features and weakly labeled data. *Journal of Cheminformatics* 8(1):1–12.

Pyysalo, S.; Ginter, F.; Heimonen, J.; Björne, J.; Boberg, J.; Järvinen, J.; and Salakoski, T. 2007. Bioinfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics* 8(1):1–24.

Pyysalo, S.; Airola, A.; Heimonen, J.; Björne, J.; Ginter, F.; and Salakoski, T. 2008. Comparative analysis of five protein-protein interaction corpora. *BMC Bioinformatics* 9(3):1–11.

Quan, C.; Hua, L.; Sun, X.; and Bai, W. 2016. Multichannel convolutional neural network for biological relation extraction. *BioMed Research International* 2016.

Singha Roy, S., and Mercer, R. E. 2022. Protein-protein interaction extraction using attention-based tree-structured neural network models. In *The International FLAIRS Conference Proceedings*, volume 35.

Sledzieski, S.; Singh, R.; Cowen, L.; and Berger, B. 2021. Sequence-based prediction of protein-protein interactions: a structure-aware interpretable deep learning model. *bioRxiv*.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*.

Tang, Z.; Guo, X.; Bai, Z.; Diao, L.; Lu, S.; and Li, L. 2022. A protein-protein interaction extraction approach based on large pre-trained language model and adversarial training. *KSII Transactions on Internet and Information Systems (TIIS)* 16(3):771–791.

Tikk, D.; Thomas, P.; Palaga, P.; Hakenberg, J.; and Leser, U. 2010. A comprehensive benchmark of kernel methods to extract protein–protein interactions from literature. *PLoS Computational Biology* 6(7):e1000837.

Van Landeghem, S.; Saeys, Y.; Van de Peer, Y.; and De Baets, B. 2008. Extracting protein-protein interactions from text using rich feature vectors and feature selection. In *3rd International symposium on Semantic Mining in Biomedicine (SMBM 2008)*, 77–84. Turku Centre for Computer Sciences (TUCS).

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in Neural Information Processing Systems* 30.

Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; and Bengio, Y. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903*.

Wang, D.; Liu, P.; Zheng, Y.; Qiu, X.; and Huang, X. 2020. Heterogeneous graph neural networks for extractive document summarization. *arXiv preprint arXiv:2004.12393*.

Yadav, S.; Ekbal, A.; Saha, S.; Kumar, A.; and Bhattacharyya, P. 2019. Feature assisted stacked attentive shortest dependency path based Bi-LSTM model for protein–protein interaction. *Knowledge-Based Systems* 166:18–29.

Yadav, S.; Ramesh, S.; Saha, S.; and Ekbal, A. 2020. Relation extraction from biomedical and clinical text: Unified multitask learning framework. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*.

Yao, Y.; Du, X.; Diao, Y.; and Zhu, H. 2019. An integration of deep learning with feature embedding for protein–protein interaction prediction. *PeerJ* 7:e7126.

Yu, K.; Lung, P.-Y.; Zhao, T.; Zhao, P.; Tseng, Y.-Y.; and Zhang, J. 2018. Automatic extraction of protein-protein interactions using grammatical relationship graph. *BMC medical informatics and decision making* 18:35–43.

Zhang, Y.; Lin, H.; Yang, Z.; Wang, J.; Zhang, S.; Sun, Y.; and Yang, L. 2018. A hybrid model based on neural networks for biomedical relation extraction. *Journal of Biomedical Informatics* 81:83–92.

Zhang, H.; Guan, R.; Zhou, F.; Liang, Y.; Zhan, Z.-H.; Huang, L.; and Feng, X. 2019. Deep residual convolutional neural network for protein-protein interaction extraction. *Ieee Access* 7:89354–89365.

Zhao, Z.; Yang, Z.; Lin, H.; Wang, J.; and Gao, S. 2016. A protein-protein interaction extraction approach based on deep neural network. *International Journal of Data Mining and Bioinformatics* 15(2):145–164.