Knowledge-Based Systems 166 (2019) 18–29

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Feature assisted stacked attentive shortest dependency path based Bi-LSTM model for protein–protein interaction

Shweta Yadav *, Asif Ekbal *, Sriparna Saha *, Ankit Kumar, Pushpak Bhattacharyya

Department of Computer Science and Engineering, Indian Institute of Technology Patna (IIT Patna), Patna, India

ARTICLE INFO

Article history:
Received 9 August 2017
Received in revised form 10 November 2018
Accepted 15 November 2018
Available online 18 December 2018

Keywords:
Relation extraction
Protein-protein interaction
Bi-directional long short term memory(Bi-LSTM)
Stacked attention
Deep learning
Shortest dependency path
Support vector machine

ABSTRACT

Knowledge about protein–protein interactions is essential for understanding the biological processes such as metabolic pathways, DNA replication, and transcription etc. However, a majority of the existing Protein–Protein Interaction (PPI) systems are dependent primarily on the scientific literature, which is not yet accessible as a structured database. Thus, efficient information extraction systems are required for identifying PPI information from the large collection of biomedical texts.

In this paper, we present a novel method based on attentive deep recurrent neural network, which combines multiple levels of representations exploiting word sequences and dependency path related information to identify protein–protein interaction (PPI) information from the text. We use the stacked attentive bi-directional long short term memory (Bi-LSTM) as our recurrent neural network to solve the PPI identification problem. This model leverages joint modeling of proteins and relations in a single unified framework, which is named as the 'Attentive Shortest Dependency Path LSTM' (Att-sdpLSTM) model. Experimentation of the proposed technique was conducted on five popular benchmark PPI datasets, namely AiMed, BioInfer, HPRD50, IEPA, and LLL. The evaluation shows the F1-score values of 93.29%, 81.68%, 78.73%, 76.25%, & 83.92% on AiMed, BioInfer, HPRD50, IEPA, and LLL dataset, respectively. Comparisons with the existing systems show that our proposed approach attains state-of-the-art performance.

1. Introduction

The study of the Protein–Protein Interaction (PPI) is crucial in understanding the biological process, such as DNA replication, transcription, metabolic pathways and cellular organization. Owing to this fact, several databases have been manually curated to cache protein interaction information such as MINT [1], BIND [2], and SwissProt [3] in structured and standard formats. However, the rapid growth of biomedical literature has shown a significant gap between the availability of protein interaction article and its automatic curation. As such, a majority of the protein interaction information is still uncovered in the textual contents of biomedical literature. Moreover, the growth in biomedical literature is at an exponential pace. In the last 20 years, the overall size of MEDLINE has increased at a 4.2% compounded annual growth rate. There is 3.1% compounded annual growth rate in the number of new entries in MEDLINE database. MEDLINE currently has more than 6,000,000 publications, which are more than three millions than those published in the last 5 years alone [4]. Hence, owing to the exponential rise [5,6] and complexity of the biological information, the necessity for intelligent information extraction techniques to assist biologist in detecting, curating and maintaining database is becoming crucial. This has lead to a surge in the interest of Biomedical Natural Language Processing (BioNLP) community for automatic detection and extraction of PPI information.

Determining PPIs in the scientific text is the process of recognizing how two or more proteins in the given biomedical sentence are related. We exemplify interaction types between protein pairs in Table 1 where protein (*Bnrlp-Rho4p*) forms the interacting protein pair and the (*Bnrlp-Rho1p*) is non-interacted protein pairs.

Majority of the existing systems look upon this task as a binary classification problem by identifying whether any interaction occurs between a pair of proteins or not. One of the most explored techniques for PPI task includes kernel-based method [7,8]. The potentiality of the kernel-based method is due to the virtue of a large amount of carefully crafted features. However, extraction of these features relies on the other NLP tools such as ABNER [9], MedT-NER [10] or PowerBioNE [11] and machine learning (ML) tool (SVM-light with Tree-Kernels). Recently, with the widespread usages of neural network based techniques in clinical and biomedical domain natural language processing tasks [12–21], methods exploring latent features have emerged as strong alternative choices over the traditional machine learning based techniques. Some of the distinguished studies [22,23] for PPI extraction tasks utilize convolution neural networks (CNNs) which have shown

* Corresponding authors.
E-mail addresses: shweta.pcs14@iitp.ac.in (S. Yadav), asif@iitp.ac.in (A. Ekbal), sriparna@iitp.ac.in (S. Saha), pb@iitp.ac.in (P. Bhattacharyya).

https://doi.org/10.1016/j.knosys.2018.11.020
0950-7051/

Table 1
Exemplar description of protein protein interaction in a sentence.

Sentence	Protein entities	Interacted protein pair	Non-interacted protein pair
Bnrlp interacts with another Rho family member, Rho4p,but not with Rho1p.	Bnrlp, Rho4p, Rho1p	Bnrlp-Rho4p	Bnrlp-Rho1p
These data demonstrate that Stat3 but not Stat1 or Stat5 is directly recruited to the ligand-activated IL-10 receptor.	Stat3, Stat1, Stat5, IL-10	Stat3-IL-10	Stat3-Stat1, Stat3-Stat5, Stat3-IL-10, Stat1-Stat5, Stat1-IL-10, Stat5-IL-10

significant performance improvements over the existing state-of-art techniques. Some other popular neural network based models for relation extraction are [24,25] system. However, these systems are mostly applicable in identifying different relationships from newswire articles. Thus these approaches fail to produce a comparable performance on biomedical literature owing to the complexity of the biomedical text. Biomedical named entities do not have standard nomenclature. Moreover, the different protein entities often have similar names making it more difficult to capture the contextual information, and these arbitrariness increases the difficulty in capturing the semantic relationships between the entities (proteins).

Motivated by these observations, in this paper we propose an attentive shortest dependency path based Bi-directional LSTM architecture (Att-sdpLSTM) to identify PPI pairs from the text. The proposed method differs from the previous studies in three facets: firstly, utilizing the dependency relationships between the protein names, we generate the Shortest Dependency Path (SDP) of the sentences. This facilitates us to create more syntax-aware inferences about the capabilities of the proteins in a sentence in comparison to the technique developed based on classical kernel-based method. Second, we investigate the significance of Part-of-Speech (PoS) and position embedding features in improved learning of the Att-sdpLSTM. Finally, we exploit the stacked Bi-LSTM over attention by stacking multiple Bi-LSTMs layers on top of each other, and finally generating the weighted sum representation of hidden states using the attention mechanism. This approach potentially allows the hidden state at each level to operate at different timescale. In contrast to the systems proposed by [22] and [23], we employ attentive multi-layer Bi-LSTM models [26] instead of Convolutional Neural Network (CNN) [27]. In CNN, features are generated by performing pooling over the entire sentence based on continuous *n* grams, where *n* refers to the filter size. This puts constraints on longer sentences where long-term dependencies exist. Our method circumvents the shortcoming of CNN architecture by utilizing the Bi-LSTM layer, which can effectively encode the long-term dependencies using the recurrent connection. In general, Bi-LSTM can keep track of preceding and succeeding words. We also use the attention mechanism to generate the weighted representation of each word. As such, when we employ the LSTM, we obtain the features from the entire sentence possessing the whole information not just on *n*-grams as in state-of-the-art CNN based architecture [22,23]. The intuition behind Bi-LSTM network is that it combines the multiple levels of representations that are proven to be effective in deep networks with the flexible use of long range context that empowers RNNs (LSTM). Also, introducing attention mechanism in the context of relation classification helps in weighing of text segments (e.g., word or sentence) or some high-level feature representations obtained by learning a scoring function. This allows a model to pay more attention to the most influential segments of texts for a relationship category.

In contrary, the existing methods [24,28] generally consider a whole sentence as the input. The drawback of these existing techniques is that such representations fail to describe the relationships of two target entities which appear in the same sentence at a far distance (i.e. long distant). Considering these problems, in our proposed technique we exploited dependency parsing related feature to examine the sentence and capture the Shortest Dependency Path to generate SDP based word embedding. In order to further inject the explicit linguistic information and boost the performance of the attentive multi-layer LSTM architecture, we have included the PoS information of SDP based words to assist the LSTM based network. The position w.r.t protein and part-of-speech (PoS) are prominent features[1] in identifying the protein interaction information. PoS provides useful evidence that helps to detect important grammatical properties. Words assigned with same PoS possess similar syntactic behavior which provides an important clue to the system for inferencing the interaction between the protein pair.

The basic structure of a sentence can be obtained by determining the position of protein-word and the word occurring in its vicinity which provides pivotal clues to identify interactions in sentences. The extraction of SDP based word embeddings rather than full sentence embedding and its usage as an input to attentive Bi-LSTM network in an amalgamation with the other latent feature is the core contribution of our proposed work.

The key contributions of the proposed work are summarized below:

(1) An shortest dependency path based attentive Bi-LSTM model (Att-sdpLSTM) inspired from [29] is proposed for relation extraction in biomedical domain.
(2) Integration of different concepts (SDP, attention, stacking, & feature embedding) and application of the integrated system in solving the biomedical protein protein interaction task is a novel contribution.
(3) Latent features like Part-of-Speech (PoS) and position of token with respect to the proteins which are found to be effective are utilized in extracting protein–protein pairs in a deep learning framework.
(4) We have demonstrated that word embedding models learned on the PubMed, PMC and Wikipedia corpus are more powerful than the internal embedding models or the models trained on general corpus such as the news corpus.[2]
(5) Evaluations on five different benchmark corpora, namely AiMed, BioInfer, HPRD50, IEPA, and LLL establish the fact that our proposed approach is generic in nature. Please note that these five datasets were created by following different protein annotation guidelines.

2. Related works

1. **Pattern-based model:** Preliminary studies conducted by [30] and [31] explored pre-specified patterns and rules for the PPI task. However, the system lacks in identifying complex cases such as complex relationships expressed in various coordinating and relational clauses. For sentences containing complex relations between three or more entities, the approach usually yields erroneous results. For example, "*The gap1 mutant blocked stable association of Ste4p with the plasma membrane, and the ste18 mutant blocked stable*

[1] We used the word features and embeddings interchangeably for position and PoS input.

[2] https://code.google.com/archive/p/word2vec/.

Table 2
Feature Encoding for sentence "*Interaction between cell cycle regulator, Prot1, and Prot2 mediates repression of HIV-1 gene transcription.*" Here, the words occurring in the vicinity of SDP are used to generate features.

SDP words	PoS	PoS feature	PoS feature Encoding	Relative position from $Prot_1$	Relative position from $Prot_2$	Position feature-1	Position feature-2	Position feature encoding-1	Position feature encoding-2
Prot1	NN	10000000	[0.00171600 …0.0033500]	0	−6	0000000000	0000111111	[0.03141600 …0.9035500]	[0.1117600 …0.0223500]
Regulator	NN	00000000	[0.99121600 …0.0233500]	1	−5	0000000001	0000011111	[0.77171600 …0.4858500]	[0.83191600 …−0.1133500]
Between	IN	00100000	[0.25191600 …0.1739500]	2	−4	0000000011	0000001111	[0.33171600 …−0.8833500]	[0.58961600 …0.7189200]
Interaction	NN	10000000	[0.17171219 …0.7583350]	3	−3	0000000111	0000000111	[0.75171600 …0.5533500]	[0.99171600 …0.7633500]
and	CC	00100000	[0.17001600 …0.3030350]	4	−2	0000001111	0000000011	[0.78117600 …−0.033500]	[0.72171600 …0.1233500]
Repression	NN	10000000	[0.17858500 …0.8835300]	5	−1	0000011111	0000000001	[0.45897600 …−0.0522500]	[0.7800100 …0.3311500]
Prot2	NN	10000000	[0.98581600 …0.0263500]	6	0	0000111111	0000000000	[0.77451600 …0.8985500]	[0.1745100 …0.3323500]

association of Ste4p with both plasma membranes and internal membranes."
In [32] authors proposed a technique based on dynamic programming to automatically discover patterns. The system proposed in [33] also studied the performance of rule-based algorithms. They developed two models, first one made use of rapier rule-based system and the other one relied on longest common subsequences.

2. **Using dependency parsing:** Here we describe the works that take into account more syntax aware approach such as full and partial (shallow) parsing. In the partial parsing, sentence structure is divided partially and dependencies are generated locally within the phrase. While in full parsing, the whole sentence is considered to capture dependencies, [34] developed the system solely based on the shallow syntactic information. They further incorporated kernel functions to combine information from the entire sentence and the local contexts around the interacting entities. The work reported in [35] focused on extracting the SDP between the protein pairs by defining the cosine similarities and edit distance function via semi-supervised learning. Some of the other prominent works include the studies conducted by [36] and [37]. Other popular studies based on full parsing include the works as reported in [38–40].
3. **Kernel-based model:** Bunescu and Mooney [7] first proposed the idea of using kernel methods to extract PPI based on the SDP. Some of the effective kernel-based techniques for PPI task include graph kernel [41], bag-of-word (BoW) kernel [42], edit-distance kernel [35], all-path kernel [8] and tree kernel [43,44].
4. **Deep learning based model:** Recent studies show the applicability of deep learning models for the PPI task [22,23]. The work reported in [22] made use of Convolutional Neural Network (CNN) for developing the PPI based system. [23] proposed a CNN based model utilizing several handcrafted features exploiting lexical, syntactic and semantic level information in combination with word embeddings.

3. Method

In this study, we present a novel method to predict protein interaction pairs from the biomedical text. Our model leverages joint modeling of proteins and relations in a single model by exploiting attentive stacked Bi-LSTM technique. Dependency between entities captures the information relevant for identifying the relations. We begin by extracting SDP sentences, which capture the dependency information between the entities and exploiting latent features PoS and position embedding. Embeddings are generated corresponding to each feature which is passed as input to the stacked Bi-LSTM unit. The architecture of our proposed Att-sdpLSTM is shown in Fig. 2. We describe each phase in succeeding subsections.

3.1. Shortest Dependency Path (SDP)

The input to the sdpLSTM is the SDP between a protein pair. For this purpose, we exploit the dependency parse tree of the sentence. It describes the syntactic constituent structure of the sentence by annotating edges with dependency types, e.g. *subject*, *auxiliary*, *modifier* and captures the semantic *predicate-argument* relationships between the words. In general, [7] first proposed the idea of using dependency parse tree for relation extraction. They designed a kernel function exploring the shortest path between the entities to capture the relations. The main intuition behind this is based on the observation that shortest path reveals non-local dependencies within sentences which can help in capturing the relevant information from the sentence. The shortest path between the protein pair generally captures the essential information (aspects of sentence construction such as mood, modality and sometimes negation, which can significantly alter or even reverse the meaning of the sentence) to identify their relationship. The approach proposed in [45] was proved to be significantly better over the dependency tree kernel-based model. We follow this idea to use SDPs for extracting protein interacting pairs.

As illustrated in Fig. 2, the word 'bind' in SDP carries important information to predict the interaction between the protein pair. The dependency relation bounded here is by verb argument and as interaction verb carries essential evidence in PPI. For PPI task, capturing these dependency relations is important.

For the purpose of extracting dependency relations, we use Enju Parser[3] which is a syntactic parser for English and can effectively analyze syntactic and/or semantic structures of biomedical text and provide with predicate-argument information. We have generated a graph for every sentence that contains at least two protein entities where each word corresponds to the node of the graph and the edges between the nodes (dependency relation) are obtained by the parser. We utilize Breadth First Search (BFS) algorithm [46] to calculate the shortest distance between the protein pair. The words occurring between the SDP only takes part in the training instead of the whole words present in the sentences to generate SDP embedding.

[3] http://www.nactem.ac.uk/enju/.

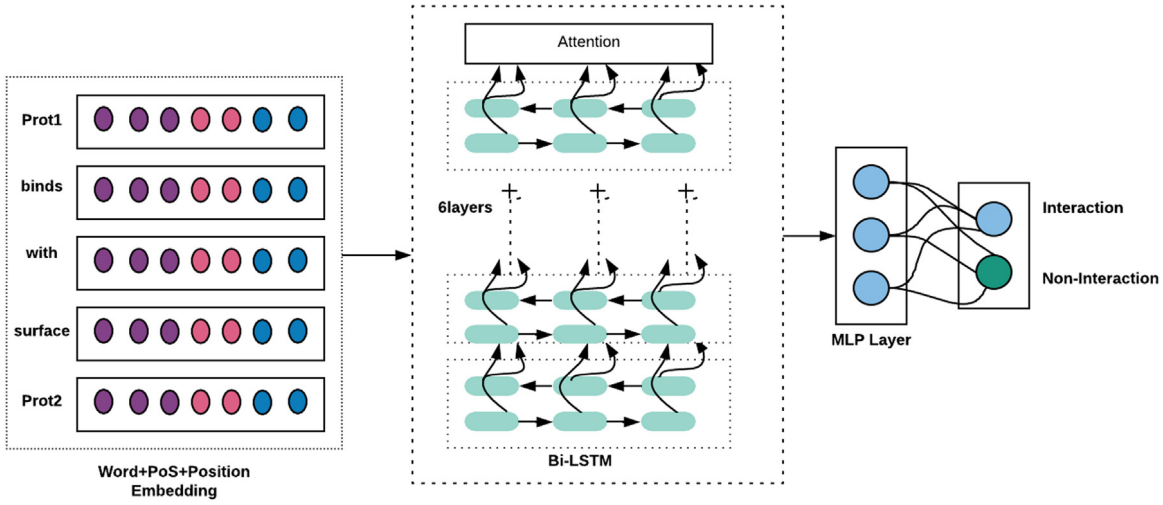

Fig. 1. Proposed model architecture for protein protein interaction. The input is the Shortest Dependency Path (SDP) between a pair of protein. The output of the model is the probability distribution over two class:'*interaction*' and '*non-interaction*'. (all the neurons representation are hypothetical).

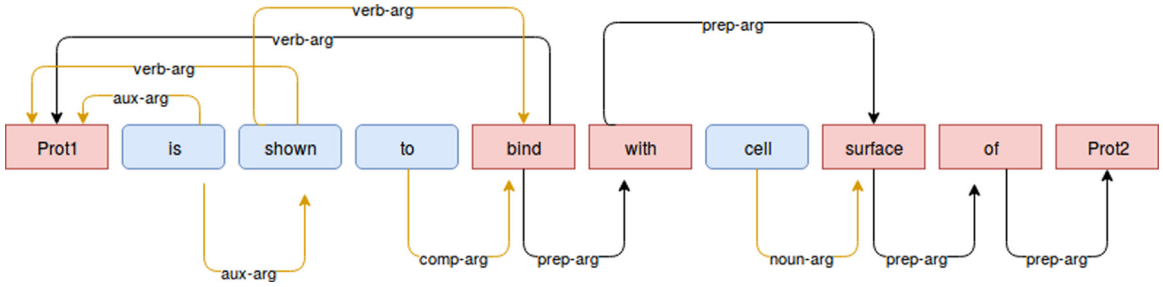

Fig. 2. The predicate argument of the example sentence "*Prot1 is shown to bind with cell surface of Prot2*". Here, the words represent the nodes and predicate argument relation is represented by edges. The red nodes form the SDP for the given sentence with the black arrow denoting the path to reach from '$Prot_1$' to '$Prot_2$'. The other words are represented in blue round-rectangular boxes that are not part of SDP. Thereby, the SDP for given sentence is "*Prot1 bind with surface of Prot2*".

Table 3
Binary representation of relative distance w.r.t the protein mention.

Distance	0	1	2	3	4	5	6	7	8	9	10	11-∞
Binary representation	0	0	0	0	0	0	0	0	0	0	1	1
	0	0	0	0	0	0	0	0	0	1	1	1
	0	0	0	0	0	0	0	0	1	1	1	1
	0	0	0	0	0	0	0	1	1	1	1	1
	0	0	0	0	0	0	1	1	1	1	1	1
	0	0	0	0	0	1	1	1	1	1	1	1
	0	0	0	0	1	1	1	1	1	1	1	1
	0	0	0	1	1	1	1	1	1	1	1	1
	0	0	1	1	1	1	1	1	1	1	1	1
	0	1	1	1	1	1	1	1	1	1	1	1

3.2. Latent feature encoding layer

Along with the SDP embedding, we design domain-independent features to assist our model in becoming more generic and adaptable. We explore PoS and position of each word as a feature. An exemplar illustration of latent feature encoding is provided in Table 2.

1. **PoS feature:** This represents the PoS for each word occurring in the vicinity of SDP. We use Genia Tagger[4] to extract PoS information of each token. Every PoS tag is encoded as a unique eight dimension one hot vector which is fed to a neural network (NN) based encoder. Auto-encoder [47] is employed to transfer the sparse PoS features to the dense real-valued feature vectors. This converts one-hot representation to dense feature representation of dimension 8. We use Adadelta optimizer [48] with loss function as a squared error to train our auto-encoder model.
Let P represents the one hot vector of a PoS tag corresponding to each word. The auto-encoder learns the transition functions φ and Ω such that reconstruction errors (squared errors) are minimized. The function φ and Ω are called the encoder and the decoder function, respectively. Mathematically, it can be written as follows:

$$\varphi, \Omega = \underset{\varphi,\Omega}{\operatorname{argmin}} \|P - P'\|^2 \quad (1)$$

where $\varphi : P \rightarrow Z$, $\Omega : Z \rightarrow P$.
2. **Position feature:** This feature helps us in identifying the significant interacting tokens between the two target protein entities. The position feature computes the relative distances of a word with respect to the protein mentions. We extract this feature on SDP of the target protein pairs. It is a two-dimensional tuple denoting distances of these tokens from the two target proteins. For e.g., consider the following sentence: 'Prot1 regulator between interaction and repression Prot2', the relative distances of the word 'interaction' with respect to Prot1 and Prot2 are -3 and 3, respectively. Relative distances are then mapped to 10-dimensional binary vectors. From Table 3, we can observe that more attention is given to the words near to the protein mentions, particularly to the words occurring in the vicinity of 10 surrounding words. Moreover, words whose relative distances exceed 10 are all treated equally.

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

Intuitively, the words which are nearer to the target words are more informative than the farther words. We perform experiments to determine the optimal dimension by varying the distance (from 5 to 12) of more informative words with respect to proteins as shown in Table 6. We notice that the system performs well when the maximum relative distance of the informative word is within the range of 10 w.r.t the protein term. As we follow the binary representation of distance, therefore, the position feature is represented using a feature vector of 10 dimensions.

Similar to PoS feature, every position feature is encoded as a 10-dimensional vector which is fed into an auto-encoder. Using the learned auto-encoder model, we convert the sparse position feature vector to a dense real valued feature vector of dimension 10.

3.3. Embedding layer

Word embedding persuades a real-valued latent semantic or syntactic vector for each word from a large unlabeled corpus by using continuous space language models [49]. In embedding layer we obtain real-valued vector corresponding to each word of the SDP. Let us assume that we have a SDP sentence $S_{sdp} = \{w_1, w_2 \dots w_N\}$ having size N, a pre-trained word embedding matrix $\mathbf{M} \in \mathbb{R}^{d \times |V|}$. A real-valued vector representation E_k^w for given a word w_k can be obtained as follows:

$$E_k = M \cdot j(w_k) \tag{2}$$

where $j(w_k)$ is the one hot vector representation of the word w_k. Thereafter, we augment the PoS and position embeddings (obtained from the previous layer) to the vector representation.

$$x_k = E_k^w \oplus E_k^{PoS} \oplus E_k^{position} \tag{3}$$

where E_k^{PoS} and $E_k^{position}$ are the PoS embedding and position embedding, respectively. The $\oplus$ denotes the concatenation operator. In our work, we use publicly available word embedding[5] (200 dimensions) pre-trained on a combination of PubMed and PMC articles to the text extracted from a recent English Wikipedia dump. The performance of the word embedding depends on various hyperparameter setting such as vector dimension, context window size, learning rate, sample size etc. Pysallo et al. [50] has released this pre-trained biomedical embedding after the deep analysis of various hyperparameter setting that obtains optimal embedding. Utilizing the pretrained word embedding not only helps in minimizing the time cost but also helpful in obtaining the best optimal parameter.

3.4. Stacked Bi-LSTM layer

The Stacked Bi-LSTM layer takes the input from embedding layer and provides a higher level abstract representation of each word in the sentence. Recurrent neural network (RNN) is a powerful technique to encode a sentence by capturing long term dependency. However, because of the long sequence it often suffers from vanishing or exploding gradient problems [26,51]. This problem can be overcome by gating and memory mechanism as introduced in LSTM [52]. LSTM provides a different way to compute the hidden states.

The feature word sequence is represented by a bidirectional LSTM-RNNs [26]. The LSTM unit at kth word consists of an *input* gate i_k, *forget* gate f_k, an *output* gate o_k, a memory cell c_k and hidden state h_k. The input to this unit is a n-dimensional input vector x_k, the previous hidden state h_{k1}, and the memory cell h_{k1}, and computes the new hidden states as follows:

$$\begin{aligned}
i_k &= \sigma(W_1^{(i)} x_k + W_2^{(i)} h_{k-1} + b^{(i)}) \\
f_k &= \sigma(W_1^{(f)} x_k + W_2^{(f)} h_{k-1} + b^{(f)}) \\
o_k &= \sigma(W_1^{(o)} x_k + W_2^{(o)} h_{k-1} + b^{(o)}) \\
u_k &= tanh(W_1^{(u)} x_k + W_2^{(u)} h_{k-1} + b^{(u)}) \\
c_k &= i_k \odot u_k + f_k \odot c_{k-1} \\
h_k &= o_k \odot tanh(c_k)
\end{aligned} \tag{4}$$

where $\sigma, \odot$ denote the sigmoid function and element-wise multiplication, respectively. The W_1, W_2 and b's are the weight-metrics and bias vectors, respectively. We can simplify Eq. (4) as follows:

$$h_k, c_k = LSTM(x_k, c_{k-1}, h_{k-1}) \tag{5}$$

Inspired by the success of stacked attentive LSTM in solving other NLP tasks [29,53–55], we use the stacked LSTM to encode the shortest dependency path sentence. The Stacked LSTM is an extension to LSTM model that has multiple hidden LSTM layers where each layer contains multiple memory cells. The purpose of using multiple LSTM layers is to learn more sophisticated conditional distributions from the data [56]. In this work, we employ vertical stacking strategy where the output of the previous layer of LSTM is fed to the input of the next layer of LSTM. Let the number of layers in stacked LSTM is L then the LSTM computes the hidden state and memory cell for each layer $l \in L$ as follows:

$$h_k^l, c_k^l = LSTM(x_k^l, c_k^{l-1}, h_k^{l-1}) \tag{6}$$

where, h_k^l and c_k^l are the hidden state representation and the memory cell at the lth layer, respectively. The inputs c_k^0 and h_k^0 to the first layer ($l = 1$) of LSTM are initialized randomly. The first layer of LSTM unit at kth word feature takes the input as the concatenation of word embedding, PoS embedding and position embeddings obtained from an auto-encoder: $x_k^1 = [E_k^w \oplus E_k^{PoS} \oplus E_k^{position}]$. The inputs $(x_k^{l+1}, c_k^l, h_k^l)$ to the $(l+1)$th LSTM layer is the (h_k^l, c_k^l, h_k^l), in other words the output hidden state h_k^l of the lth layer is the input to the $(l+1)$th layer and the hidden state and memory cell are initialized with the previous layer's hidden state and memory cell respectively. We compute the forward ($\overrightarrow{h_k}$) and backward ($\overleftarrow{h_k}$) hidden state for each word k in the sentence. The final hidden state at layer l is computed by augmenting both the hidden states: $z_k^l = [\overrightarrow{h_k^l} \oplus \overleftarrow{h_k^l}]$. The final SDP sentence representation is calculated by taking the hidden state of the last layer (L) of the LSTM as follows:

$$z_1, z_2, \dots, z_N = [\overrightarrow{h_1^L} \oplus \overleftarrow{h_1^L}], [\overrightarrow{h_2^L} \oplus \overleftarrow{h_2^L}], \dots, [\overrightarrow{h_N^L} \oplus \overleftarrow{h_N^L}] \tag{7}$$

3.5. Attention layer

We introduce another layer over the outputs of stacked LSTM. The attention layer uncovers the salient contexts from the SDP sentence and encodes those to form the context vector. Usually the contexts in our task are the clue words and the implicit information which play important roles in deciding the interaction or non-interaction between the protein pairs. The inputs to this layer are the hidden states as calculated in Eq. (7) and the output is the weighted sum based on the attention distribution. We first feed the hidden state z_k of the kth of the SDP sentence to one-layer perceptron to obtain the m_k as a hidden representation of z_k, then we compute the similarity with the context vector c. We obtain the normalized attention weights through softmax. Finally, the

[5] http://bio.nlplab.org/.

weighted SDP representation (R) is calculated by multiplying the attention weight to the stacked LSTM hidden representation z_k.

$$m_k = tanh(W_a z_k + b_a)$$
$$\alpha_k = \frac{e^{m_k^T * c}}{\sum_{i=1}^{N} e^{m_i^T * c}} \tag{8}$$
$$R = \sum_{k=1}^{N} \alpha_k * z_k$$

where, W_a and b_a are the weight matrix and bias vector, respectively. The context vector c is randomly initialized and jointly learned through training.

3.6. Multilayer Perceptron (MLP) layer

The output of attention layer R is fed into a fully connected layer with H number of hidden layers. More formally, given a sequence layer output R, number of hidden layers H, network calculates output as follows:

$$M = f(W_M * R + b_M) \tag{9}$$

where, $W_M \in \mathbb{R}^{H \times R}$ is the weight matrix between the output of sequence layer and hidden layer; $b_M \in \mathbb{R}^{H \times 1}$ is a bias term vector. Thereafter, the output M is transformed into $T \in \mathbb{R}^{C \times 1}$ by augmenting with a weight matrix $W_T \in \mathbb{R}^{C \times H}$, where C is the number of required labels. In our case the value of $C = 2$.

$$T = W_T * M \tag{10}$$

Finally, the transformed output T is fed into the softmax layer. The softmax layer provides the output probability of each label. Mathematically, it can be written as follows:

$$P(class = C|T) = \frac{e^{T_{class}}}{\sum_{k=1}^{C} e^{T_k}} \tag{11}$$

Fig. 1 represents the architecture of our Att-sdpLSTM model.

4. Results

4.1. Dataset

The proposed model is evaluated on the five popular benchmark corpora for PPI, namely AiMed, BioInfer,[6] HPRD50,[7] IEPA,[8] & LLL.[9] AiMed dataset is generated from 197 abstracts extracted from the Database of Interacting Protein (DIP). It contains 1955 sentences with the protein entities, manually tagged with the PPI interaction relations. This is recognized as the standard dataset for PPI extraction task.

The BioInfer corpus created by the Turku BioNLP group[10] consists of 836 sentences. In our work, we assume the protein interacted pair as the positive instance and non-interacted pair as the negative instance. To identify the negative instances which are not directly given in the dataset, we assume all the possible pairs of proteins that are possible in a given sentence and consider those protein-pairs to be negative instances whose relations are not given in the sentence. Thereby, we obtain 3109 negative instances and 939 positive instances for AiMed corpus. Similarly, in case of BioInfer corpus, we obtain 5951 negative instances over 1077 positive interactions. It can be observed that all dataset are imbalanced as they are strongly biased towards the negative examples.

HPRD50, referenced by the Human Protein Reference Database (HPRD) dataset is generated by randomly selecting a subset of 50 abstracts [57]. The annotation was done for direct physical interactions, regulatory relations, and for any modifications (e.g., phosphorylation). The dataset consists of a total 145 sentences, with 163 positive interaction pairs and 270 negative pairs.

IEPA, termed as the Interaction Extraction Performance Assessment (IEPA) consists of nearly three hundred abstracts [58]. These abstracts were retrieved from the MEDLINE utilizing the queries. Each query was the AND of two biochemical nouns.

LLL (Learning language in logic) is another PPI dataset, released as part of the LLL shared task challenge 2005 [59]. The aim of the task was to extract protein/gene interactions in the form of relations from biology abstracts of the Medline bibliography database, specifically concerning Bacillus subtilis transcription.

A detail statistics of these datasets are provided in Table 4.

Table 4
Dataset statistics for PPI extraction.

Datasets	Interacted pair	Non-interacted pair	Ratio
AiMed	939	3109	1:3.3
BioInfer	1077	5951	1:5.5
HPRD50	163	270	1:1.6
IEPA	335	482	1:1.4
LLL	164	166	1:1.0

4.2. Preprocessing

The protein entities are generalized with the protein IDs to make the model insensitive towards biases associated with the names of the proteins. This makes every protein unique and avoids the model to learn highly interacting protein pairs. We perform tokenization with the help of Genia Tagger.[11] The tokenized sentence is parsed with the Enju parser to obtain the dependency relations.

4.3. Network training and hyper-parameters

The objective of training the Bi-LSTM model is to minimize the binary cross entropy cost function. It can be written as follows:

$$\mathcal{L}(S, Y) = -\frac{1}{n}\sum_{i=1}^{n} y^{(i)} \ln a(s^{(i)}) + \left(1 - y^{(i)}\right) \ln\left(1 - a(s^{(i)})\right) \tag{12}$$

Here, $S = \{s^{(1)}, s^{(2)} \ldots s^{(n)}\}$ is the set of input SDP sentence in the training dataset, and $C = \{c^{(1)}, c^{(2)} \ldots c^{(n)}\}$ is the corresponding set of labels for those SDP sentences. The $a(s)$ denote the output of the MLP layer. The gradient-based optimizer is used to minimize our cost function described in Eq. (12). We have used Adam [60], an adaptive learning rate based optimizer, to update the parameters throughout training. To avoid over-fitting, the network dropout [61] mechanisms are used with a dropout rate of 0.3.

The hyper-parameter values were determined from the preliminary experiments by evaluating the model performance for 10-fold cross-validation. The proposed model described in Section 3 is implemented using Keras.[12] We have chosen Tensorflow as backend machine learning library. We tune our model for various hyper-parameters of the LSTM architecture including the number of LSTM units, dropout ratio, number of epochs and different

6 http://corpora.informatik.hu-berlin.de/.
7 https://goo.gl/M5tEJj.
8 https://goo.gl/JeboFE.
9 https://goo.gl/1DsDqL.
10 http://bionlp.utu.fi/.
11 http://www.nactem.ac.uk/GENIA/tagger/.
12 https://keras.io/.

Table 5
Optimal hyper-parameter setting on 10-fold cross validation for all datasets.

Hyper-parameters	Optimal value
Number of LSTM units	64
Dropout ratio	0.3
Activation function	Sigmoid
Optimization algorithm	Adam
# Epochs (AiMed & BioInfer)	115
# Epochs (HPRD50, IEPA & LLL)	50
Size of MLP layer output	30
No. of LSTM layers	6
Context vector size	75

Table 6
Analysis of context window on 10 fold cross validation data for position feature on *sdpLSTM* model.

Context window size	F-score (AiMed)	F-score (BioInfer)
[−5,5]	78.36	72.72
[−6,6]	78.54	73.18
[−7,7]	79.19	73.23
[−8,8]	81.16	74.29
[−9,9]	81.75	75.56
[−10,10]	82.89	75.93
[−11,11]	82.17	75.28
[−12,12]	81.41	74.88

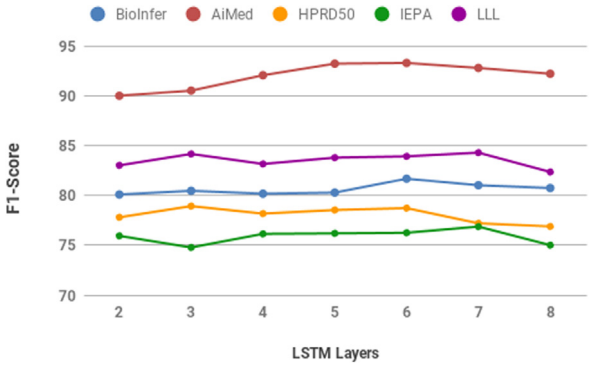

Fig. 3. Effect of stacking Bi-LSTM layers.

optimization algorithms etc. for all datasets. The optimum performance is achieved with 115 epochs for AiMed and BioInfer datasets as depicted in Fig. 4. We obtain the best results for all the PPI datasets on a set of optimized network hyper-parameters (c.f. Table 5) using 10-fold cross validation experiments.

4.4. Analysis of hyper-parameter settings

We setup all the experiments by varying the hyper-parameter values and analyze the behaviors of our model. For AiMed dataset, we observed that addition of LSTM units improves the model performance to a certain extent. Thereafter, it keeps on decreasing gradually. We define an optimal value 64 for the same, via cross-validation experiment. We started the experiment with single LSTM layer and keep on increasing till six layer of LSTMs. We observed that the performance start decreasing after sixth layer of LSTMs. The model performance against the varying number of LSTM layer can be visualize in Fig. 3. In case of other datasets, we also observe quite a similar trend in performance with the addition of LSTM units, size of context vector and stacking of LSTM layer.

As shown in Fig. 3, stacking helps in improving the performance of the system for the AiMed and BioInfer dataset. However, for the dataset HPRD50, IEPA, & LLL, there was not much impact on stacking multiple LSTM units. We observed that after stacking two layers of LSTM units, the performance of the system was almost constant.

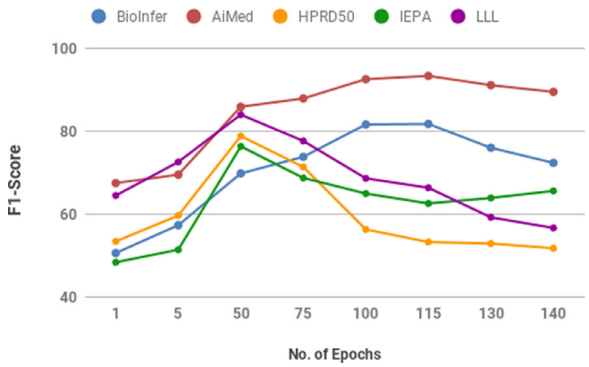

Fig. 4. Effect of varying epochs on the performance (F1-Score).

We also analyze the performance of our model on the number of epochs for which training was performed on all datasets. On AiMed dataset, the value of F1-score initially shows minor growth from epochs 1 to 5 and then shows regular growth with the increasing number of epochs from 5 to 115, and finally a dip on further increasing the number of epochs to 115 and 140. For BioInfer dataset there has been steady increase with the increase in the number of epochs. We achieve the optimum performance with the almost same number of epochs (115) for all datasets. The model behaviors with respect to the epochs are shown in Fig. 4. For the remaining dataset, the optimal results are achieved with 50 epochs (c.f. Fig. 4), this is because the HPRD50, IEPA and LLL datasets are small compared to AiMed and BioInfer datasets and model get over-fitted with higher number of epochs.

Similarly, we performed the cross-validation experiment with the varying size of context vector and found to be optimal on size 75 for all the datasets.

4.5. Evaluation on benchmark datasets

In the recent years, different kernel-based techniques and SVM based model were adopted as baselines against the deep learning CNN based model for the PPI task. It has been shown how deep learning based models perform superior compared to the feature based models [23,62]. As such, in order to make an effective comparison of our proposed approach, we design three strong baselines based on neural network architecture as follows:

1. **Baseline 1:** The first baseline model is constructed by training a multi-layer perceptron on the features obtained from the embedding layer as defined in Section 3.3. The sentence embedding S_M is generated by the concatenation of every PoS and position augmented word embeddings to SDP embedding.

$$S_M = x_1 \oplus x_2 \ldots \oplus x_n; \tag{13}$$

   Thereafter, S_M is fed into MLP layer described in Section 3.6.
2. **Baseline 2:** Our second baseline is based on the more advanced sentence encoding techniques, RNN. The SDP sentence encoding S_R can be generated as follows:

$$S_R = \sigma(\mathbf{U} * x_n + \mathbf{V} * h(n-1) + \mathbf{b}) \tag{14}$$

   where σ is a sigmoid function, $h(n-1)$ denotes the hidden representation of $(n-1)$th word in the SDP sentence. U, V, and b are the network parameters. Similar to Baseline 1, MLP layer is used to classify a SDP sentence into one of the two classes, *viz*: 'interacting pair' and 'non-interacting pair'.

Table 7
Comparative results of the proposed model (Att-sdpLSTM) with the baselines model.

Model	Approach	AiMed			BioInfer			HPRD50			IEPA			LLL		
		P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Baseline 1	MLP (SDP)	59.73	75.93	66.46	68.56	72.05	70.22	68.26	66.64	67.44	69.21	64.57	66.81	70.39	71.88	71.13
Baseline 2	RNN (SDP)	66.23	74.72	70.22	71.89	74.59	73.21	75.13	72.78	73.94	72.48	70.97	71.72	78.02	77.34	77.68
Baseline 3	sdpLSTM (SDP+Feature Embeddings)	91.10	82.20	86.45	72.40	83.10	77.35	79.19	76.14	77.64	76.17	74.95	75.56	83.10	82.81	82.95
Proposed model	Att-sdpLSTM (SDP+Feature Embedding+Attention + Stacking)	92.63	93.96	93.29	80.81	82.57	81.68	79.92	77.58	78.73	76.90	75.62	76.25	84.22	83.62	83.92

Table 8
Comparative results of the proposed model (Att-sdpLSTM) with state-of-the-art systems for AiMed and BioInfer dataset. Ref. [23]* and [63]* denote the re-implementation of the systems proposed in [23] and [63] with the authors reported experimental setups.

Model	Approach	AiMed			BioInfer		
		Precision	Recall	F1-score	Precision	Recall	F1-score
Proposed model	Att-sdpLSTM (SDP+Feature embedding+Attention+Stacking)	92.63	93.96	93.29	80.81	82.57	81.68
sdpLSTM + Negative sampling	sdpLSTM (SDP+Feature embeddings+Negative sampling)	89.61	80.39	84.75	70.06	82.61	75.82
[22]	sdpCNN (SDP+CNN)	64.80	67.80	66.00	73.40	77.00	75.20
[23]	DCNN (CNN+word/position embeddings+Semantic (WordNet) feature embeddings)	–	–	85.20	–	–	–
[23]*	DCNN	88.61	81.72	85.03	72.05	77.51	74.68
[64]	Single kernel+ Multiple Parser+SVM	59.10	57.60	58.10	63.61	61.24	62.40
[28]	McDepCNN (CNN+word+PoS+Chunk+NEs Multi-channel embedding)	67.30	60.10	63.50	62.70	68.20	65.30
[65]	Deep neutral network	51.50	63.40	56.10	53.90	72.90	61.60
[66]	All-path graph kernel	49.2	64.60	55.30	53.30	70.10	60.00
[63]	Multiple kernel+ Word Embedding+ SVM	–	–	69.70	–	–	74.00
[63]*	Multiple kernel+ Word Embedding+ SVM	67.18	69.35	68.25	72.33	74.94	73.61
[67]	Tuned tree kernels +SVM	72.80	62.10	67.00	74.50	70.90	72.60

3. **Baseline 3:** As a third baseline model, we utilized shortest dependency path based single Bi-LSTM model assisted by the latent features (PoS, position embedding). We call this baseline as *sdpLSTM*.

We perform 10-fold cross validation on all the datasets. With no official development data set available, cross validation seems to be the most reliable method of evaluating our proposed model. To evaluate the performance of our model, we use standard recall, precision, and F1-score. The detailed comparative analysis of our proposed model (Att-sdpLSTM) over these baselines and state-of-art systems are reported in Tables 7–9. The obtained results clearly show the effectiveness of our proposed Att-sdpLSTM based model over the other models exploring neural network architectures or conventional kernel or SDP based machine learning model. In our proposed model we obtain the significant F1-score improvements of 26.83, 23.07, and 6.84 points over the first three baselines for the AiMed dataset, respectively. On BioInfer dataset, our system shows the F1-Score improvements of 11.46, 8.47, and 4.33 points over these three baselines, respectively. For HPRD50 dataset, performance improvements of 11.29, 4.79, 1.09 points were observed by the proposed approach over the first three baselines, respectively. Similar improvements were also observed for the other two datasets. With IEPA, the proposed model outperforms the baselines by 9.44, 4.53, 0.69 points, respectively. On LLC dataset, the performance improvements of 12.79, 6.24, 0.97 F1-Score points were observed with the proposed approach.

5. Analysis

5.1. Comparative analysis with existing methods

In order to perform the comparative analysis with the existing approaches, we choose the recent approach exploiting neural network model for AiMed and BioInfer dataset. We explore other approaches utilizing SVM based kernel methods and word embedding feature as shown in Table 8. We observe that Att-sdpLSTM significantly performs better than all the state-of the-art techniques for AiMed and BioInfer dataset. From this, we can conclude that Att-sdpLSTM is more powerful in extracting protein interacted pairs over the CNN based architecture developed in [22] and [23]. We further make an interesting observation that incorporating the latent features embedded into the neural network based architecture improves the performance of the system.

Our proposed model attains an significant improvement of 8.09 F-score point (c.f. Table 8) over the model proposed in [23] for the AiMed dataset. The DCNN model [23] made use of a significant number (total 29) of domain dependent lexical, syntax and semantic level features. In contrast to this our model is more generic in the sense that we use only PoS and position features. We further re-implemented the DCNN system and evaluated it on AiMed and BioInfer datasets. Evaluation (c.f. Table 8) shows that our proposed model attains better performance for AiMed and BioInfer datasets. We also re-implemented the system reported in [63] to obtain the precision and recall values. We also conducted experiments by introducing negative sampling in Baseline 3 model for AiMed and BioInfer dataset. As shown in Table 8, the overall performance of the system has dropped by nearly 2% for all datasets compared to Baseline 3.

We also conducted comparative analysis for HPRD50, IEPA, and LLL dataset with the existing state-of-the-art system utilizing kernel based approach. Table 9 shows that our proposed model outperformed the state-of-art by 7.83, 1.15, and 1.72 F1-Score points on HPRD50, IEPA, and LLL dataset, respectively.

5.2. Effects of stacking Bi-LSTM with attention

We examined the impact of stacking multiple Bi-LSTM layers by varying the number of layers from 1 to 6. To investigate the role of stacking, we replaced basic LSTM model with the stacked Bi-LSTM model. We observed (c.f. Table 10) the performance improvement of 5.49 F1-Score points on AiMed dataset and 3.11 F1-Score points

Table 9
Comparative results of the proposed model (Att-sdpLSTM) with state-of-the-art systems for HPRD50, IEPA, and LLL dataset.

Model	Approach	HPRD50			IEPA			LLL		
		P	R	F1	P	R	F1	P	R	F1
Proposed model	Att-sdpLSTM (SDP+Feature Embedding+Attention+Stacking)	79.92	77.58	78.73	76.90	75.62	76.25	84.22	83.62	83.92
[66]	APG	68.2	69.8	67.8	66.6	82.6	73.1	71.3	91	78.1
[66]	APG(with SVM)	65.4	72.5	67.5	71.0	75.1	72.1	70.9	95.4	79.7
[68]	kBSPS	66.7	80.2	70.9	70.4	73.0	70.8	76.8	91.8	82.2
[41]	APG	64.3	65.8	63.4	69.6	82.7	75.1	72.5	82.2	76.8
[69]	Rich Feature Based	60.0	51.0	55.0	64.0	70.0	62.0	72.0	73.0	73.0
[70]	Hybrid	68.5	76.1	70.9	67.5	78.6	71.7	77.6	86.0	80.1
[41]	Co-occ	38.9	100.0	55.4	40.8	100.0	57.6	55.9	100.0	70.3
[71]	RelEx	76.0	64.0	69.0	74.0	61.0	67.0	82.0	72.0	77.0

Table 10
Effect of stacking and attention on proposed Att-sdpLSTM model.

Model	AiMed			BioInfer			HPRD50			IEPA			LLL		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
sdpLSTM	91.10	82.20	86.45	76.61	78.10	77.35	79.19	76.14	77.64	76.17	74.95	75.56	83.10	82.81	82.95
sdpLSTM + Stacking	92.89	91.02	91.94	79.29	81.67	80.46	79.53	76.73	78.10	76.29	75.16	75.72	83.41	82.94	83.17
Att-sdpLSTM (sdpLSTM + Stacking + Attention)	92.63	93.96	93.29	80.81	82.57	81.68	79.92	77.58	78.73	76.90	75.62	76.25	84.22	83.62	83.92

Table 11
Proposed model performance after removing PoS and position embeddings once at a time.

Model	AiMed			BioInfer			HPRD50			IEPA			LLL		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Att-sdpLSTM	92.63	93.96	93.29	80.81	82.57	81.68	79.92	77.58	78.73	76.90	75.62	76.25	84.22	83.62	83.92
Att-sdpLSTM - PoS Embeddings	94.61	92.44	93.51	79.95	82.04	80.98	78.52	76.82	77.66	75.88	74.91	75.39	83.13	82.50	82.81
Att-sdpLSTM - Position Embeddings	94.39	91.41	92.87	80.37	82.44	81.39	79.07	76.13	77.57	76.02	74.64	75.32	83.58	82.91	83.24
Att-sdpLSTM - PoS - Position Embeddings	95.61	89.17	92.27	79.16	81.95	80.53	78.32	75.89	77.09	75.72	74.49	75.10	82.97	82.28	82.62

improvement on the BioInfer dataset. For the other three datasets, we observed very modest improvement by introducing stacking. Performance improvements of 0.46, 0.16, and 0.22 points were observed for HPRD50, IEPA, and LLL dataset, respectively. The possible reason for not getting any significant improvement (unlike AiMed and BioInfer datasets) is the small dataset size. The model was easily overfitted and therefore no major impact was observed.

In order to understand the role of attention, we further incorporated the attention to sdpLSTM + stacking model. The obtained results show the effectiveness of attention mechanism on all the datasets. Incorporating attention boosts the performance of the stacked sdpLSTM model by 1.35, 1.22, 0.63, 0.53, and 0.75 F1-Score points on AiMed, BioInfer, HPRD50, IEPA, and LLL dataset, respectively. With the vanilla sdpLSTM model, performance improvements of 6.84, 4.33, 1.09, 0.69, and 0.97 F1-Score points were observed on AiMed, BioInfer, HPRD50, IEPA, and LLL datasets, respectively.

5.3. Effects of feature combination

In this section, we analyze the significance of each feature by performing feature ablation study (removing one feature at a time) as shown in Table 11. We begin by examining only SDP embedding. It can be observed that Att-sdpLSTM alone without using additional features shows a remarkable performance of 92.27, 80.53, 78.73, 76.25, and 83.92 F1-Score on AiMed, BioInfer, HPRD50, IEPA, and LLL datasets, respectively. This clearly shows the significance of SDP based embedding with attention in identifying protein interacted pairs. We observe that inclusion of position embedding slightly improves (0.42 F1 score) the performance on the AiMed dataset. However, there have been drops in F1-score by 0.22 points when PoS feature is added. This might be due to the data sparseness problem with the lack of training data. In case of BioInfer dataset, position embedding is comparatively less informative, but still boosts the F1-score by 0.29 F1-score points. The inclusion of PoS embeddings, however, shows an improvement of 0.70 F-score points. The reason is while adding a position to PoS feature, it helps as we have PoS tag information (which is NNP) of the closest potential entity. We analyze that, the improvements are not simply due to combining the features to SDP embedding. This suggests that these information sources are complementary to each other in some linguistic aspects. We closely investigate the outputs of the AiMed dataset produced in our system and make the summary with the following observations:

1. **PoS distribution:** Protein names are mainly noun phrases. For the AiMed dataset, we observed that the multi-word proteins were not properly tagged as the noun phrases. This encountered some errors which eventually propagated when introduced the PoS alone as a feature to the LSTM model.
2. **Presence of protein interacted words:** The presence of protein interacted words (inhibit, regulated, interaction etc.) provides an important clue to identify the interaction of proteins. When the system takes SDP as input, we observe that in some cases the PoS tagger is unable to tag the interacted words as verbs. This could be one of the reasons that the system performance is comparable when we use PoS information alone as a feature.

For the HPRD50 dataset, the exclusion of PoS drops the model performance by 1.07 F1 Score points. Position feature was also observed as a significant feature in assisting the proposed model. Removal of this feature leads to the decrease in F1-Score by 1.16 points. We observed similar phenomena for the IEPA and LLL datasets, where exclusion of PoS feature drops the F1 Score by 0.86, 1.11 point respectively. When the position feature is removed from the proposed model, it showed the F-score degradation by 0.93 and 0.68 points, respectively, for IEPA and LLL data sets. Interestingly,

Table 12
Statistical significance (Wilcoxon Signed-Rank with 95% confidence interval (C.I.)) tests for the baselines and proposed model. The p-values greater than the confidence level (0.05) are shown in italicized font.

Datasets	With Baseline 1 *p*-value (95% C.I.)	With Baseline 2 *p*-value (95% C.I.)	With Baseline 3 *p*-value (95% C.I.)
AiMed	2.453e−4	0.024	0.032
BioInfer	3.749e−3	0.014	0.039
HPRD50	4.258e−3	0.048	*0.129*
IEPA	3.477e−4	0.027	0.046
LLL	9.231e−4	0.047	*0.094*

combination of all the features improves the performance of the system by 1.02, 1.15, 1.64, 1.15 and 1.3 F1-score points on AiMed, BioInfer, HPRD50, IEPA, and LLL datasets, respectively. We observe that when the model is evaluated on the less number of epochs, performance improvement with the addition of features is 3%–4%. Increasing the epoch gradually vanishes the impact of additional features.

5.4. Statistical significance testing

We conduct the statistical significance tests to verify the improvements over the baselines. Specially, we used the Wilcoxon signed-ranks test [72]. The Wilcoxon signed-rank test is the non-parametric uni-variate test which is an alternative to the dependent t-test. Wilcoxon signed-rank test estimates the statistical significance for the null hypothesis that the two models (one of the baselines and the proposed model) are equally accurate. The p-values for the null hypothesis, corresponding to different baselines for our proposed model, are listed in Table 12. This test confirms that the performance of proposed model is statistically significant over Baseline 1 and Baseline 2 for all the PPI datasets. The dataset for which null hypothesis can be rejected (p-value < 0.05) are highlighted.

5.5. Error analysis

In this subsection, we analyze different sources of errors which lead to misclassification. We closely study the false positive and false negative instances and come up with following observations:

(1) When Enju dependency parser fails to capture dependencies, the error is propagated to BFS algorithm as such it does not return any valid SDP. For example, in the given sentence

"*The ProtId1 or ProtId2 family is targets of cytokines and other agents that induce HIV-1 gene expression*", the mentioned SDP outputs are **"ProtId1 and ProtId2"** and **"ProtId1 family ProtId2"**. It should be noted that this is a negative example and our SDP fails to capture the context. This hampers our accuracy significantly.

(2) **Presence of multiple protein entities:** Another form of misclassification is because of the presence of multiple protein instances in a sentence. Repetitive mention of protein is expected to act like a noise, which may cause neural models to lose relevant information from other words likely to be contextually important. For example:

"The nucleotide sequences of ProtId26 (ProtId29), ProtId28 (ProtId23), ProtId27 (ProtId31), ProtId22 (ProtId32), and ProtId30 (ProtId24) genes were partly determined for 19 wild strains of measles virus (MV) isolated over the past 10 years in Japan (nucleotide position ProtId33: 1301-1700, ProtId21: 1751-2190, ProtId25: 3571-4057, ProtId19: 6621-7210, ProtId20: 10381-11133) and also for a MV strain obtained from a patient with subacute sclerosing panencephalitis (SSPE) who had natural measles in 1980".

(3) **No mention of explicit protein:** The misclassification was observed where there is no mention of the explicit interaction bearing words. For example:

"Cotransfections with different combinations of these genes demonstrated that a subset of four of them, coding for the HSV ProtId242 complex (ProtId241, ProtId239, ProtId243 and the ProtId240, was already sufficient to mediate the helper effect".

(4) **Negative protein interacting word:** Interaction bearing words carry important information to identify protein interacted pairs such as bind, interact, inhibit. However, when interaction bearing words appear in negative context, system fails to properly classify those as non-interacted protein pairs. For example:

"in GSK-3 inhibitors suppressed Sema4D-induced growth", **inhibit** does not occur here in context of PPI.

6. Conclusion and future works

In this article, we have proposed an efficient model based on deep learning technique for PPI. The model makes use of SDP embeddings as low level input feature. In addition, it also exploits the latent PoS and position embedding features to complement the SDP embedding. The main contribution of the proposed methodology is the systematic integration of word embeddings learn from the biomedical literature and the use of SDP between protein pairs into the attentive stacked sdpLSTM architecture. Bio-medical word embedding was observed to capture semantic information more effectively than internal embedding. By employing SDP and Bi-LSTM, the proposed approach could make full use of structural information. Our comprehensive experimental results on five benchmark biomedical corpora, AiMed, BioInfer, HPRD50, IEPA,and LLL demonstrated that (i) the SDP based word embedding input is effective to describe protein–protein relationships in PPI task; (ii) the attentive Bi-LSTM architecture is useful to capture the long contextual and structure information; and (iii) high-quality pre-trained word embedding is important in the PPI task. The obtained results depict the superiority of Att-sdpLSTM over the complex state-of-art approaches leveraging CNN and several higher level features with the significant F1-score improvements of 8.09 and 6.48 points on AiMed and BioInfer dataset, respectively. Similarly, for the HPRD50, IEPA, LLL datasets, our proposed model outperformed the state-of-art by 7.83, 1.15, and 1.72 F1-Score points, respectively.

In future, we would like to validate our approach on other relation extraction tasks such as drug-drug interaction, chemical-protein interaction by overcoming the possible errors.

Acknowledgments

Asif Ekbal acknowledges the Young Faculty Research Fellowship (YFRF), supported by Visvesvaraya PhD scheme for Electronics and IT, Ministry of Electronics and Information Technology (MeitY), Government of India, being implemented by Digital India Corporation (formerly Media Lab Asia). The authors would like to gratefully acknowledge the partial support of Sushrut:ezDI Research Lab on Health Informatics, Department of Computer Science and Engineering, IIT Patna, India.

References

[1] A. Zanzoni, L. Montecchi-Palazzi, M. Quondam, G. Ausiello, M. Helmer-Citterich, G. Cesareni, MINT: a molecular interaction database, FEBS Lett. 513 (1) (2002) 135–140.
[2] G.D. Bader, D. Betel, C.W. Hogue, BIND: the biomolecular interaction network database, Nucleic Acids Res. 31 (1) (2003) 248–250.
[3] A. Bairoch, R. Apweiler, The swiss-prot protein sequence database and its supplement trembl in 2000, Nucleic Acids Res. 28 (1) (2000) 45–48.
[4] L. Hunter, K.B. Cohen, Biomedical language processing: what's beyond pubmed? Mol. Cell 21 (5) (2006) 589–594.
[5] Z. Lu, Pubmed and beyond: a survey of web tools for searching biomedical literature, Database 2011 (2011).
[6] R. Khare, R. Leaman, Z. Lu, Accessing biomedical literature in the current information landscape, Biomed. Literature Min. (2014) 11–31.

[7] R.C. Bunescu, R.J. Mooney, A shortest path dependency kernel for relation extraction, in: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2005, pp. 724–731.
[8] A. Airola, S. Pyysalo, J. Björne, T. Pahikkala, F. Ginter, T. Salakoski, A graph kernel for protein-protein interaction extraction, in: Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing, Association for Computational Linguistics, 2008, pp. 1–9.
[9] B. Settles, Abner: an open source tool for automatically tagging genes, proteins and other entity names in text, Bioinformatics 21 (14) (2005) 3191–3192.
[10] R. Sætre, K. Yoshida, A. Yakushiji, Y. Miyao, Y. Matsubayashi, T. Ohta, Akane system: protein-protein interaction pairs in biocreative2 challenge, ppi-ips subtask, in: Proceedings of the Second Biocreative Challenge Workshop, Vol. 209, Madrid, 2007, p. 212.
[11] G. Zhou, J. Zhang, J. Su, D. Shen, C. Tan, Recognizing names in biomedical texts: a machine learning approach, Bioinformatics 20 (7) (2004) 1178–1190.
[12] S. Yadav, A. Ekbal, S. Saha, P. Bhattacharyya, 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, Vol. 1, 2017, pp. 1159–1170.
[13] S. Yadav, A. Ekbal, S. Saha, P. Bhattacharyya, A. Sheth, 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), Vol. 2, 2018, pp. 271–277.
[14] S. Yadav, A. Ekbal, S. Saha, P. Bhattacharyya, Deep learning architecture for patient data de-identification in clinical records, in: Proceedings of the Clinical Natural Language Processing Workshop (ClinicalNLP), 2016, pp. 32–41.
[15] D. Ningthoujam, Shweta, A. Ekbal, P. Bhattacharyya, Relation extraction between the clinical entities based on the shortest dependency path based lstm, in: Proceedings of the 13th International Conference on Natural Language Processing, 2018.
[16] A. Ekbal, S. Saha, P. Bhattacharyya, et al., A deep learning architecture for protein-protein interaction article identification, in: Pattern Recognition (ICPR), 2016 23rd International Conference on, IEEE, 2016, pp. 3128–3133.
[17] A. Kumar, A. Ekbal, S. Saha, P. Bhattacharyya, et al., A recurrent neural network architecture for de-identifying clinical records, in: Proceedings of the 13th International Conference on Natural Language Processing, 2016, pp. 188–197.
[18] S. Yadav, A. Ekbal, S. Saha, Information theoretic-pso-based feature selection: an application in biomedical entity extraction, Knowl. Inf. Syst. (2018) 1–26.
[19] S. Yadav, A. Ekbal, S. Saha, P.S. Pathak, P. Bhattacharyya, Patient data de-identification: a conditional random-field-based supervised approach, in: Handbook of Research on Applied Cybernetics and Systems Science, IGI Global, 2017, pp. 234–253.
[20] S. Yadav, A. Ekbal, S. Saha, Feature selection for entity extraction from multiple biomedical corpora: a pso-based approach, Soft Comput. (2017) 1–24.
[21] F. Dernoncourt, J.Y. Lee, O. Uzuner, P. Szolovits, De-identification of patient notes with recurrent neural networks, J. Am. Med. Inf. Assoc. 24 (3) (2017) 596–606.
[22] L. Hua, C. Quan, A shortest dependency path based convolutional neural network for protein-protein relation extraction, BioMed. Res. Int. 2016 (2016).
[23] S.-P. Choi, Extraction of protein–protein interactions (ppis) from the literature by deep convolutional neural networks with various feature embeddings, J. Inf. Sci. (2016) 0165551516673485.
[24] M. Miwa, M. Bansal, End-to-end relation extraction using lstms on sequences and tree structures, arXiv preprint arXiv:1601.00770.
[25] Y. Liu, F. Wei, S. Li, H. Ji, M. Zhou, H. Wang, A dependency-based neural network for relation classification, arXiv preprint arXiv:1507.04646.
[26] A. Graves, Generating sequences with recurrent neural networks, CoRR abs/1308.0850, URL http://arxiv.org/abs/1308.0850.
[27] Y. LeCun, Y. Bengio, et al., Convolutional networks for images, speech, and time series, Handb. Brain Theory Neural Netw. 3361 (10) (1995) 1995.
[28] Y. Peng, Z. Lu, Deep learning for extracting protein-protein interactions from biomedical literature, arXiv preprint arXiv:1706.01556.
[29] Y. Wu, M. Schuster, Z. Chen, Q.V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, et al., Google's neural machine translation system: bridging the gap between human and machine translation, arXiv preprint arXiv:1609.08144.
[30] C. Blaschke, M.A. Andrade, C.A. Ouzounis, A. Valencia, Automatic extraction of biological information from scientific text: protein-protein interactions., in: Ismb, Vol. 7, 1999, pp. 60–67.
[31] T. Ono, H. Hishigaki, A. Tanigami, T. Takagi, Automated extraction of information on protein–protein interactions from the biological literature, Bioinformatics 17 (2) (2001) 155–161.
[32] M. Huang, X. Zhu, Y. Hao, D.G. Payan, K. Qu, M. Li, Discovering patterns to extract protein–protein interactions from full texts, Bioinformatics 20 (18) (2004) 3604–3612.
[33] R. Bunescu, R. Ge, R.J. Kate, E.M. Marcotte, R.J. Mooney, A.K. Ramani, Y.W. Wong, Comparative experiments on learning information extractors for proteins and their interactions, Artif. Intell. Med. 33 (2) (2005) 139–155.
[34] C. Giuliano, A. Lavelli, L. Romano, Exploiting shallow linguistic information for relation extraction from biomedical literature, in: EACL, Vol. 18, Citeseer, 2006, pp. 401–408.
[35] G. Erkan, A. Özgür, D.R. Radev, Semi-supervised classification for extracting protein interaction sentences using dependency parsing., in: EMNLP-CoNLL, Vol. 7, 2007, pp. 228–237.
[36] Y. Miyao, K. Sagae, R. Sætre, T. Matsuzaki, J. Tsujii, Evaluating contributions of natural language parsers to protein–protein interaction extraction, Bioinformatics 25 (3) (2009) 394–400.
[37] S. Garg, A. Galstyan, U. Hermjakob, D. Marcu, Extracting biomolecular interactions using semantic parsing of biomedical text, in: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, in: AAAI'16, AAAI Press, 2016, pp. 2718–2726, URL http://dl.acm.org/citation.cfm?id=3016100.3016282.
[38] J.M. Temkin, M.R. Gilder, Extraction of protein interaction information from unstructured text using a context-free grammar, Bioinformatics 19 (16) (2003) 2046–2053.
[39] N. Daraselia, A. Yuryev, S. Egorov, S. Novichkova, A. Nikitin, I. Mazo, Extracting human protein interactions from medline using a full-sentence parser, Bioinformatics 20 (5) (2004) 604–611.
[40] A. Yakushiji, Y. Miyao, Y. Tateisi, J. Tsujii, Biomedical information extraction with predicate-argument structure patterns, in: Proceedings of the first International Symposium on Semantic Mining in Biomedicine (SMBM), Hinxton, Cambridgeshire, UK, April, 2005.
[41] A. Airola, S. Pyysalo, J. Björne, T. Pahikkala, F. Ginter, T. Salakoski, All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning, BMC Bioinform. 9 (11) (2008) S2.
[42] R. Sætre, K. Sagae, J. Tsujii, Syntactic features for protein-protein interaction extraction., LBM (Short Papers) 319 (2007).
[43] C. Ma, Y. Zhang, M. Zhang, Tree kernel-based protein-protein interaction extraction considering both modal verb phrases and appositive dependency features, in: Proceedings of the 24th International Conference on World Wide Web, in: WWW '15 Companion, ACM, New York, NY, USA, 2015, pp. 655–660, http://dx.doi.org/10.1145/2740908.2741705, URL http://doi.acm.org/10.1145/2740908.2741705.
[44] M. Zhang, J. Zhang, J. Su, G. Zhou, A composite kernel to extract relations between entities with both flat and structured features, in: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, in: ACL-44, Association for Computational Linguistics, Stroudsburg, PA, USA, 2006, pp. 825–832, http://dx.doi.org/10.3115/1220175.1220279, URL http://dx.doi.org/10.3115/1220175.1220279.
[45] A. Culotta, J. Sorensen, Dependency tree kernels for relation extraction, in: Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics, Association for Computational Linguistics, 2004, p. 423.
[46] C.Y. Lee, An algorithm for path connections and its applications, IRE Trans. Electron. Comput. (3) (1961) 346–365.
[47] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.A. Manzagol, Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion, J. Mach. Learn. Res. 11 (Dec) (2010) 3371–3408.
[48] M.D. Zeiler, ADADELTA: an adaptive learning rate method, CoRR abs/1212.5701, URL http://arxiv.org/abs/1212.5701.
[49] B. Tang, H. Cao, X. Wang, Q. Chen, H. Xu, Evaluating word representation features in biomedical named entity recognition tasks, BioMed. Res. Int. 2014 (2014).
[50] S. Moen, T.S.S. Ananiadou, Distributional semantics resources for biomedical text processing, 2013.
[51] R. Pascanu, T. Mikolov, Y. Bengio, Understanding the exploding gradient problem, CoRR abs/1211.5063.
[52] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[53] A. Graves, A.-r. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: Acoustics, Speech and Signal Processing (icassp), 2013 ieee International Conference On, IEEE, 2013, pp. 6645–6649.
[54] C. Dyer, M. Ballesteros, W. Ling, A. Matthews, N.A. Smith, Transition-based dependency parsing with stack long short-term memory, arXiv preprint arXiv:1505.08075.
[55] A. Prakash, S.A. Hasan, K. Lee, V. Datla, A. Qadir, J. Liu, O. Farri, Neural paraphrase generation with stacked residual lstm networks, arXiv preprint arXiv:1610.03098.
[56] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv e-prints abs/1409.0473. URL https://arxiv.org/abs/1409.0473.
[57] K. Fundel, R. Küffner, R. Zimmer, RelExRelation extraction using dependency parse trees, Bioinformatics 23 (3) (2006) 365–371.
[58] J. Ding, D. Berleant, D. Nettleton, E. Wurtele, Mining medline: abstracts, sentences, or phrases? in: Biocomputing 2002, World Scientific, 2001, pp. 326–337.

[59] C. Nédellec, Learning language in logic-genic interaction extraction challenge, in: Proceedings of the 4th Learning Language in Logic Workshop (LLL05), Vol. 7, Citeseer, 2005, pp. 1–7.
[60] D.P. Kingma, J. Ba, Adam: A method for stochastic optimization, CoRR abs/1412.6980, URL http://arxiv.org/abs/1412.6980.
[61] N. Srivastava, G.E. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.
[62] Y. Kim, Convolutional neural networks for sentence classification, in: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, Doha, Qatar, 2014, pp. 1746–1751, URL http://www.aclweb.org/anthology/D14-1181.
[63] L. Li, R. Guo, Z. Jiang, D. Huang, An approach to improve kernel-based protein–protein interaction extraction by learning from large-scale network data, Methods 83 (2015) 44–50.
[64] L. Qian, G. Zhou, Tree kernel-based protein–protein interaction extraction from biomedical literature, J. Biomed. Inform. 45 (3) (2012) 535–543.
[65] Z. Zhao, Z. Yang, H. Lin, J. Wang, S. Gao, A protein-protein interaction extraction approach based on deep neural network, Int. J. Data Min. Bioinform. 15 (2) (2016) 145–164.
[66] D. Tikk, P. Thomas, P. Palaga, J. Hakenberg, U. Leser, A comprehensive benchmark of kernel methods to extract protein–protein interactions from literature, PLoS Comput. Biol. 6 (7) (2010) e1000837.
[67] S.-P. Choi, S.-H. Myaeng, Simplicity is better: revisiting single kernel ppi extraction, in: Proceedings of the 23rd International Conference on Computational Linguistics, Association for Computational Linguistics, 2010, pp. 206–214.
[68] P. Palaga, Extracting relations from biomedical texts using syntactic information, Mémoire de DEA, Vol. 138, Technische Universität Berlin, 2009.
[69] S. Van Landeghem, Y. Saeys, B. De Baets, Y. Van de Peer, Extracting protein-protein interactions from text using rich feature vectors and feature selection, in: 3rd International Symposium on Semantic Mining in Biomedicine (SMBM 2008), Turku Centre for Computer Sciences (TUCS), 2008, pp. 77–84.
[70] M. Miwa, R. Sætre, Y. Miyao, J. Tsujii, Protein–protein interaction extraction by leveraging multiple kernels and parsers, Int. J. Med. Inform. 78 (12) (2009) e39–e46.
[71] S. Pyysalo, A. Airola, J. Heimonen, J. Björne, F. Ginter, T. Salakoski, Comparative analysis of five protein-protein interaction corpora, in: BMC Bioinformatics, Vol. 9, BioMed Central, 2008, p. S6.
[72] F. Wilcoxon, Individual comparisons by ranking methods, Biometrics Bull. 1 (6) (1945) 80–83.