Identifying Protein-protein Interactions in Biomedical Literature using Recurrent Neural Networks with Long Short-Term Memory

Yu-Lun Hsieh
SNHCC, TIGP, Academia Sinica, and
National Cheng Chi University, Taiwan
morphe@iis.sinica.edu.tw

Yung-Chun Chang
Graduate Institute of Data Science,
Taipei Medical University, Taiwan
changyc@tmu.edu.tw

Nai-Wen Chang and **Wen-Lian Hsu**
IIS, Academia Sinica, Taiwan
{nwchang,hsu}@iis.sinica.edu.tw

Abstract

Accurate identification of protein-protein interaction (PPI) helps biomedical researchers to quickly capture crucial information in literatures. This work proposes a recurrent neural network (RNN) model to identify PPIs. Experiments on two largest public benchmark datasets, AIMed and BioInfer, demonstrate that RNN outperforms state-of-the-art methods with relative improvements of 10% and 18%, respectively. Cross-corpus evaluation also indicates that RNN is robust even when trained on data from different domains. These results suggest that RNN effectively captures semantic relationships among proteins without any feature engineering.

1 Introduction

In systematic biology, protein-protein interaction (PPI) is an important subject that aims at exploring the role of intermolecular interactions, which is crucial for reconstructing molecular networks in cells (Mori, 2004). A widely-used information source regarding PPI is PubMed, which contains over 27 million research papers and continues to grow at a rate of 1.5 million per year. Given the vast amount of papers published, collecting PPI information manually is time-consuming. Thus, a major research question in biomedical text-mining is to efficiently identify the sentences that contain PPIs. Although certain PPI may span across multiple sentences, existing work mostly focus on those PPIs existing within a single sentence (Tikk et al., 2010). For instance, given the sentence "STAT3 selectively interacts with Smad3 to antagonize TGF-β signaling," a model should correctly identify that proteins *STAT3*, *Smad3*, and *TGF-β* have interactions with one another. More specifically, there are $\binom{3}{2} = 3$ pairs of proteins in the sentence, and there are PPIs in all three pairs. Note that the exact type of interaction is not in the scope of this task.

Recent breakthrough in neural network (NN) led to increasing amount of work that apply NN on various text-mining tasks. Specifically, convolutional neural networks (CNN) (Lecun et al., 1998) have been most commonly adapted for PPI. Compared with traditional machine learning (ML) methods such as SVM (Cortes and Vapnik, 1995), CNN models do not require tedious feature engineering and domain knowledge. However, how to best incorporate linguistic and semantic information into the CNN model remains an active research topic, since previous CNN-based methods have not achieved state-of-the-art performance in PPI identification task (Peng and Lu, 2017).

This paper proposes a novel approach based on recurrent neural networks (RNNs) to capture the long-term relationships among words in order to identify PPIs. The proposed model is evaluated on two largest PPI corpora, *i.e.*, AIMed (Bunescu et al., 2005) and BioInfer (Pyysalo et al., 2007) using cross-validation (CV) and cross-corpus (CC) settings. Experimental results from CV show that RNN outperforms state-of-the-art methods by relative improvements of 10% and 18% on AIMed and BioInfer, respectively. Furthermore, RNN remains effective even when trained on a different domain in the CC setting.

The rest of this paper is organized as follows. Sec. 2 provides important previous work related to PPI and NN. Sec. 3 describes the architecture of the proposed model. Sec. 4 details the experimental procedure and Sec. 5 presents experimental results and findings. Finally, Sec. 6 concludes this paper and points to the directions for future work.

Proceedings of the The 8th International Joint Conference on Natural Language Processing, pages 240–245,
Taipei, Taiwan, November 27 – December 1, 2017 ©2017 AFNLP

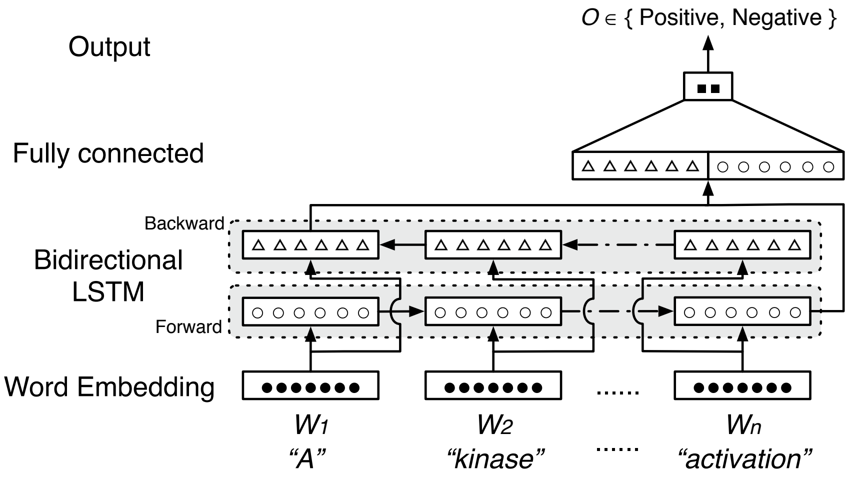

Figure 1: Overview of the proposed model. The first layer transforms words into corresponding embeddings and feeds them sequentially to the bi-directional RNN. The forward and backward output vectors are concatenated as the new "feature vector" and sent to the fully connected layer for final classification. For simplicity w/o losing details, dropout layers are omitted.

2 Related work

PPI identification can be cast as a binary classification problem where discriminative classifiers are trained with a set of positive and negative instances. Two major categories of approaches are proposed, *i.e.*, manual rule-based systems and ML approaches (Bunescu et al., 2005). The former approach is intuitive but time-consuming and requires intensive labor, while the latter are more common and primarily "kernel-based". Kernel-based methods usually take advantage of the syntactic or semantic structure of a sentence. For example, Qian and Zhou (2012) includes shortest dependency path (sdp) with tree-kernel classifier, and Chang et al. (2016) integrate knowledge base with a tree kernel to strengthen PPI identification. Other approaches include shortest path kernels (Bunescu and Mooney, 2005), graph kernel (Airola et al., 2008), composite kernel (Miwa et al., 2009), subsequence kernels (Kim et al., 2010), and tree kernels (Eom et al., 2006; Qian and Zhou, 2012). However, engineering features from different sources may not lead to optimal results.

Recent advances in NN research have been applied to PPI identification as well. Zhao et al. (2016) used an auto-encoder for feature extraction from words and a logistic regression for classification. Li et al. (2015) proposed a hybrid of kernel- and NN-based model and examined the strength of integrating NN-extracted features into kernels. They conclude that NNs can automatically extract discriminative features and aid kernels in PPI identification. Furthermore, Peng and Lu (2017) integrated dependency graph information into a CNN and improved performances on AIMed and BioInfer over kernel-based methods, with F-scores 63.5% and 65.3%, respectively. Hua and Quan (2016) used shortest dependency path feature to simplify the input and avoid bias from feature selection. Their method achieved 66.6% F-score on AIMed and 75.3% on BioInfer dataset. Alternatively, RNN with Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) have been shown to possess outstanding abilities when modeling sequential data with long-term dependency (Greff et al., 2017). Majority of previous work that use LSTM focused on machine translation (Sutskever et al., 2014), named-entity recognition (Lample et al., 2016), or classification of a sequence, *e.g.*, the sentiment of a piece of movie review (Tai et al., 2015). Recently, LSTMs have been utilized to perform relation extraction and classification on general texts (Miwa and Bansal, 2016).

3 Method

We propose a novel approach for identifying PPI using bi-directional RNN with LSTM. Figure 1 illustrates the overview of our model, which takes a sentence containing protein entities as input and outputs a probability distribution (Bernoulli distribution) corresponding to whether there exists a PPI or not. There are three types of layers: an embedding layer, a recurrent layer, and a fully connected layer, which are described as follows.

Embedding Layer transforms words into embeddings (Mikolov et al., 2013), which are dense, low-dimensional, and real-valued vectors. They capture syntactic and semantic information provided by its neighboring words. In this work, we examine the effect of pre-training embeddings by comparing randomly initialized and pre-trained ones from Chiu et al. (2016), which was trained on over 25 million PubMed records.

Recurrent Layer is constructed using LSTM cells, as illustrated in Fig. 2. An LSTM cell contains a "memory" cell and three "gates", *i.e.*, input, forget, and output. The input gate modulates the current input and previous output. The forget gate tunes the content from previous memory to the current. Finally, the output gate regulates the output from the memory.

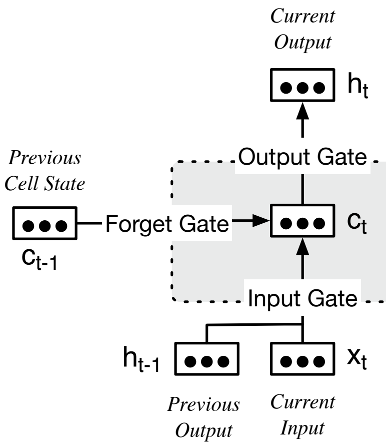

Figure 2: Simplified illustration of an LSTM cell. The input gate and forget gate jointly control the content of the memory c_t, and the output gate regulates output from c_t.

Specifically, let $\mathbf{x}_t$ be the input at time t, and $\mathbf{i}_t, \mathbf{f}_t, \mathbf{o}_t$ correspond to input, forget, and output gates, respectively. $\mathbf{c}_t$ denotes the memory cell and $\mathbf{h}_t$ is the output. The learnable parameters include $W_{i,f,o,c}$ and $U_{i,f,o,c}$. They are defined as:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(W_i \mathbf{x}_t + U_i \mathbf{h}_{t-1}) \\
\mathbf{f}_t &= \sigma(W_f \mathbf{x}_t + U_f \mathbf{h}_{t-1}) \\
\mathbf{o}_t &= \sigma(W_o \mathbf{x}_t + U_o \mathbf{h}_{t-1}) \\
\tilde{\mathbf{c}}_t &= \tanh(W_c \mathbf{x}_t + U_c \mathbf{h}_{t-1}) \\
\mathbf{c}_t &= \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \tilde{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \circ \tanh(\mathbf{c}_t)
\end{aligned}$$

where "$\circ$" denotes the element-wise product of vectors and σ represents the sigmoid function.

We use a bi-directional RNN to encode a sequence in forward and backward directions, which has been proven effective in sequence modeling tasks (Dyer et al., 2015). In essence, it uses two cells, one to encode the input sequence in its original order and the other in reverse. Subsequently, the two outputs are concatenated and fed to the **Fully Connected Layer**. It serves as a classifier where the output represents class probabilities.

4 Experiments

We evaluate the proposed method with two largest publicly available PPI corpora: AIMed and BioInfer. Distribution of the corpora is shown in Table 1. We adopt 10-fold cross-validation (CV) and cross-corpus (CC) testing scheme. The evaluation metrics are the precision, recall, and F1-score for both schemes. Compared methods include the shortest dependency path-directed constituent parse tree (SDP-CPT) method (Qian and Zhou, 2012), in which the tree representation generated from a syntactic parser is refined by using the shortest dependency path between two entity mentions derived from a dependency parser; A knowledge-based approach PIPE (Chang et al., 2016) that extracts linguistic interaction patterns and learned by a convolution tree kernel; A composite kernel approach (CK) (Miwa et al., 2009) which combines several different layers of information from a sentence with its syntactic structure by using several parsers; and a graph kernel method (GK) (Airola et al., 2008) that integrates parse structure sub-graph and a linear order sub-graph. We further compare with recent NN-based approaches: sdpCNN (Hua and Quan, 2016) which combines CNN with shortest dependency path features; McDepCNN (Peng and Lu, 2017) that uses positional embeddings along with word embeddings as the input, and a tree kernel using various word embeddings labeled as TK+WE (Li et al., 2015). We also evaluate the effect of pre-training of word embeddings by comparing ran-

Table 1: Statistics of AIMed and BioInfer.

Corpus	Total # of Sentences	# of Positive/Negative Protein Pairs
AIMed	1,955	1,000/4,834
BioInfer	1,100	2,534/7,132

Table 2: Results (in %) from 10-fold cross-validation on AIMed and BioInfer corpora. Bold font indicates the best performance in a column. Standard deviations are enclosed in parentheses.

Method	AIMed			BioInfer		
	Precision	**Recall**	**F-score**	**Precision**	**Recall**	**F-score**
GK	52.9	61.8	56.4	56.7	67.2	61.3
SDP-CPT	59.1	57.6	58.1	-	-	62.4
CK	55.0	68.8	60.8	65.7	71.1	68.1
PIPE	57.2	64.5	60.6	68.6	70.3	69.4
McDepCNN	67.3	60.1	63.5	62.7	68.2	65.3
sdpCNN	64.8	67.8	66.0	73.4	77.0	75.2
TK+WE	-	-	69.7	-	-	74.0
LSTM$_{\text{rand}}$	70.1 (6.5)	70.4 (6.4)	70.1 (5.5)	83.6 (2.4)	83.3 (2.7)	83.4 (2.3)
LSTM$_{\text{pre}}$	**78.8** (5.6)	**75.2** (5.0)	**76.9** (4.9)	**87.0** (2.3)	**87.4** (2.3)	**87.2** (1.9)

domly initialized and pre-trained embeddings, labeled as LSTM$_{\text{rand}}$ and LSTM$_{\text{pre}}$, respectively.

4.1 Experimental Setup

To ensure the generalization of the learned model, the protein names recognized in the text are replaced with “PROTEIN1”, “PROTEIN2”, or “PROTEIN”, where “PROTEIN1” and “PROTEIN2” are the pair of interest, and other non-participating proteins are marked as “PROTEIN”. An example is given as follows. The sentence “Thymocyte activation induces the association of phosphatidylinositol 3-kinase and pp120 with CD5” contains three proteins, namely, “phosphatidylinositol 3-kinase”, “pp120”, and “CD5”. In the three possible pairs of proteins, two of them are in interaction relations. Therefore, there are three variants of this sentence with proteins being replaced by the special labels in the data, and two of them are marked as “positive” while the other one as “negative”. During testing, all the variants will be scored. The maximum sentence length is set to 100, where longer sentences are truncated and shorter sentences padded with zeros. We use 200-dimension embeddings and 400-dimension LSTM cells. The dropout rate is set to 0.5. `RMSProp` optimizer (Tieleman and Hinton, 2012) with default learning rate settings are applied[1]. With a batch size of 16, training one epoch on one Titan X GPU takes approximately one minute. The throughput of the inference stage is around 130KB of text per second.

5 Results and Discussion

Ten-fold cross-validation results on AIMed and BioInfer are listed in Table 2. Kernel-based methods can achieve decent F-scores of 61% and 69%. All NN-based methods outperform kernel-based ones by up to 10% on AIMed and 5% on BioInfer. When using randomly initialized embeddings, RNN exhibits similar performance as other NN models. However, by taking advantage of pre-trained embeddings, RNN further advances F-scores by 7% and 13% on AIMed and BioInfer, respectively. In other words, pre-training contribute to relative improvements of 10% and 18%. These results demonstrate that, even though kernel-based methods all include syntactic or semantic structures and carefully crafted features, neural networks are capable of automatically capturing contextual information that is crucial for identifying PPIs. Moreover, we can see that the standard deviations of the performance by RNN on the larger corpus, *i.e.*, BioInfer, is much lower than that of the smaller corpus (5 vs. 2). Besides, the relative improvement of RNN over compared methods on BioInfer is greater as well (10% and 18%). This suggests that richer context information in a larger corpus may be difficult to be manually modeled via feature engineering or rule creation, but is a well-suited learning source for neural networks.

Table 3 shows the cross-corpus results, in which different methods are trained on one corpus and tested on the other. We observe that, although RNN performs slightly better than McDepCNN, CK and PIPE methods are much more robust when learning on different corpora. We postulate that knowledge may play an important role in this scenario, and effective incorporation of such informa-

[1] We implement the model using Keras with tensorflow (Abadi et al., 2015) backend. Code can be downloaded from `https://github.com/ylhsieh/ppi_lstm_rnn_keras`

tion into RNN can be a promising direction for future research.

Table 3: Cross-corpus results (in %) of two corpora. Bold font indicates the best performance in a column.

Method	Train AIMed, Test BioInfer	Train BioInfer, Test AIMed
GK	47.1	47.2
CK	53.1	49.6
PIPE	**58.2**	**52.1**
McDepCNN	48.0	49.9
Proposed	*49.3*	*50.7*

6 Conclusion

We propose an end-to-end RNN-based model to identify PPIs in biological literature. Cross-validation results demonstrate that it outperforms existing methods in the two largest corpora, BioInfer and AIMed. Potential directions for future work include integrating features that have been proven useful in identifying PPIs, and conduct extensive experiments under the cross-learning scheme. Also, we will explore networks with different architectures in order to further advance the current model.

Acknowledgments

We are grateful for the constructive comments from three anonymous reviewers. This work was supported by grant MOST105-2221-E-001-008-MY3 from the Ministry of Science and Technology, Taiwan.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org. http://tensorflow.org/.

Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Salakoski. 2008. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC bioinformatics* 9(11):S2.

R. C. Bunescu and R. J. Mooney. 2005. A shortest path dependency kernel for relation extraction. *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing* pages 724–731.

Razvan Bunescu, Ruifang Ge, Rohit J Kate, Edward M Marcotte, Raymond J Mooney, Arun K Ramani, and Yuk Wah Wong. 2005. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial intelligence in medicine* 33(2):139–155.

Y. C. Chang, C. H. Chu, Y. C. Su, C. C. Chen, and W. L. Hsu. 2016. PIPE: a protein-protein interaction passage extraction module for BioCreative challenge. *Database (Oxford)* 2016. https://doi.org/10.1093/database/baw101.

Billy Chiu, Gamal Crichton, Anna Korhonen, and Sampo Pyysalo. 2016. How to train good word embeddings for biomedical NLP. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*. Association for Computational Linguistics, Berlin, Germany, pages 166–174.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20(3):273–297.

Chris Dyer, Miguel Ballesteros, Wang Ling, Austin Matthews, and Noah A. Smith. 2015. Transition-based dependency parsing with stack long short-term memory. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, pages 334–343. http://www.aclweb.org/anthology/P15-1033.

J. H. Eom, S. Kim, S. H. Kim, and B. T. Zhang. 2006. A tree kernel-based method for protein-protein interaction mining from biomedical literature. *Knowledge Discovery in Life Science Literature, Proceedings* 3886:42–52.

Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. 2017. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems* PP(99):1–11.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. LSTM can solve hard long time lag problems. In *Advances in Neural Information Processing Systems*. pages 473–479.

Lei Hua and Chanqin Quan. 2016. A shortest dependency path based convolutional neural network for protein-protein relation extraction. *BioMed Research International* 2016.

Seonho Kim, Juntae Yoon, Jihoon Yang, and Seog Park. 2010. Walk-weighted subsequence kernels for protein-protein interaction extraction. *BMC bioinformatics* 11(1):107.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of NAACL-HLT*. pages 260–270.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. 1998. Gradient-based learning applied to document recognition. *Proceedings of the Ieee* 86(11):2278–2324. https://doi.org/Doi 10.1109/5.726791.

Lishuang Li, Rui Guo, Zhenchao Jiang, and Degen Huang. 2015. An approach to improve kernel-based protein-protein interaction extraction by learning from large-scale network data. *Methods* 83:44 – 50.

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

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 1105–1116.

Makoto Miwa, Rune Sætre, Yusuke Miyao, and Jun'ichi Tsujii. 2009. Protein–protein interaction extraction by leveraging multiple kernels and parsers. *International journal of medical informatics* 78(12):e39–e46.

Hirotada Mori. 2004. From the sequence to cell modeling: comprehensive functional genomics in escherichia coli. *BMB Reports* 37(1):83–92.

Yifan Peng and Zhiyong Lu. 2017. Deep learning for extracting protein-protein interactions from biomedical literature. In *Proceedings of the 2017 Workshop on Biomedical Natural Language Processing*. To appear.

Sampo Pyysalo, Filip Ginter, Juho Heimonen, Jari Björne, Jorma Boberg, Jouni Järvinen, and Tapio Salakoski. 2007. Bioinfer: a corpus for information extraction in the biomedical domain. *BMC bioinformatics* 8(1):50.

L. H. Qian and G. D. Zhou. 2012. Tree kernel-based protein–protein interaction extraction from biomedical literature. *Journal of Biomedical Informatics* 45(3):535 – 543.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the ACL and the 7th IJCNLP*. Association for Computational Linguistics, Beijing, China, pages 1556–1566.

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4(2).

Domonkos Tikk, Philippe Thomas, Peter Palaga, Jörg Hakenberg, and Ulf Leser. 2010. A comprehensive benchmark of kernel methods to extract protein–protein interactions from literature. *PLoS Comput Biol* 6(7):e1000837.

Zhehuan Zhao, Zhihao Yang, Hongfei Lin, Jian Wang, and Song Gao. 2016. A protein-protein interaction extraction approach based on deep neural network. *International Journal of Data Mining and Bioinformatics* 15(2):145–164.