DM_NLP at SemEval-2018 Task 8: Neural Sequence Labeling with Linguistic Features

Chunping Ma, Huafei Zheng, Pengjun Xie, Chen Li, Linlin Li, Si Luo
Alibaba Group, China
{chunping.mcp, huafei.zhf, chengchen.xpj, puji.lc, linyan.lll, luo.si}@alibaba-inc.com

Abstract

This paper describes our submissions for SemEval-2018 Task 8: Semantic Extraction from CybersecUrity REports using NLP. The DM_NLP participated in two subtasks: SubTask 1 classifies if a sentence is useful for inferring malware actions and capabilities, and SubTask 2 predicts token labels (”Action”, ”Entity”, ”Modifier” and ”Others”) for a given malware-related sentence. Since we leverage results of Subtask 2 directly to infer the result of Subtask 1, the paper focus on the system solving Subtask 2. By taking Subtask 2 as a sequence labeling task, our system relies on a recurrent neural network named BiLSTM-CNN-CRF with rich linguistic features, such as POS tags, dependency parsing labels, chunking labels, NER labels, Brown clustering. Our system achieved the highest F1 score in both token level and phrase level.

1 Introduction

As a growing number of mobile devices and facilities are getting connected and digitized, malware attacks become increasingly rampant and dangerous. CybersecUrity attracts more public attention but few NLP research and efforts. A large number of malware-related texts is available online, such as malware reports and relevant blogs (DiMaggio, 2015). However due to the sheer volume and diversity of these texts, NLP researchers encounter problems to obtain valuable information, such as the specific actions taken by a certain malware and the capabilities described. Therefore, automatic screening malware-related contents and labeling every token of the contents become potential applications of NLP and have drawn growing research interests.

In order to create a database in CybersecUrity domain which helps researchers to parse malware-related texts, the organizers of SemEval 2018 Task 8 (Phandi et al., 2018) (Lim et al., 2017)proposed the follow tasks:

1. **SubTask1**: Classify if a sentence is relevant for inferring malware actions and capabilities.
2. **SubTask2**: Predict token labels for a given malware-related text.
3. **SubTask3**: Predict relation labels for a given malware-related text.
4. **SubTask4**: Predict attribute labels for a given malware-related text.

However, due to lack of time, we decided to address only SubTask 1 and SubTask 2. In this paper, we describe the system that we submitted for the SemEval 2018 shared task. Our system is based on RNN network and ranked first in both token level and phrase level.

Most existing high performance sequence labeling methods are linear statistical models, such as HMM (Hidden Markov Models) (Eddy, 1996) and CRF (Conditional Random Fields) (Lafferty et al., 2001). In the past few years, neural networks have been widely used to solve NLP problems. Specially, several RNN-based neural networks have been proposed to handle sequence labeling tasks including Chinese word segmentation (Yao and Huang, 2016), POS tagging (Huang et al., 2015), NER (Chiu and Nichols, 2015) (Lample et al., 2016), which achieved outstanding performance against traditional methods.

In this paper, we simple derive the result of SubTask1 from SubTask2 and regard SubTask 2 as the preorder. Namely, our system firstly outputs sequence labels of a given sentence, and then checks whether some target labels turn out, such as Action, Entity, Modifier. Sentences which have those target labels will be classify as malware-related one. Focusing on SubTask1, We propose a neural network architecture using a hybrid bidirectional LSTM and CNN architecture which takes character-level and word-level representations combined with rich linguistic features as input. Instead of decoding each label independently, we feed the output vectors of BiLSTM to a CRF layer. Experiments show the significant improvement of our system compared with baselines.

The remainder of this paper includes a detail description of our system in Section 2. Experiments and analysis of results are presented in Section 3. Finally, Section 4 draws a conclusion and Section 5 describes our future work.

2 System Overview

We treat Subtask 2 as a sequence labeling problem and design a neural network architecture with some hand-crafted features. Our system is mainly based on the BiLSTM-CNN-CRF model and apply model average strategy to avoid over-fitting problem.

2.1 Data Preprocessing

In order to exclude the noise from data provided by the organizers, we use a python program to correct spelling mistakes and unreadable characters. After that, in order to avoid data distribution problem, we mix the training set development set, and then shuffle and split them into five parts randomly. We take four parts as training set and the rest as development set.

2.2 Feature Extraction

Based upon many previous work on sequence labeling, our system incorporates 5 types of features: POS tags, dependency parsing, NER labels, Chunking labels and Brown clustering. All features are generated automatically. In detail, we use Stanford CoreNLP (Manning et al., 2014) [1] to annotate POS tags, dependency parsing, NER labels, and use Apache OpenNLP [2] to annotate Chunking labels. Brown clustering labels are generated by an open source implementation.

2.2.1 POS Tags

POS Tagging (part-of-speech Tagging), which attaches each word of a sentence a part of speech tag based on both its definition and its context, produces a generalization of words and is a fundamental procedure of other NLP tasks such as syntactic parsing and information extraction.

2.2.2 Dependency Labels

Dependency parsing describes the syntactic structure of a sentence in terms of the words (or lemmas) in a sentence and an associated set of directed binary grammatical relations that hold among the words. For a given word, our system takes the concatenation the dependency edge and its syntactic head as its dependency label, or 'ROOT' if the word has no syntactic head.

2.2.3 NER Labels

Named Entity Recognition (NER) labels sequences of words in a text which are the names of things, such as person and company names, or gene and protein names. We utilize NER labels as significant information to detect named token labels, such as Subject and Object.

2.2.4 Chunking Labels

Text chunking divides a text into phrases in such a way that syntactically related words become member of the same phrase. For instance, "technology organizations" is a noun phrase, our system annotates "technology" as "B-NP" and "organizations" as "I-NP".

2.2.5 Brown Clustering Labels

Similar words have similar distributions of words to their immediate left and right. Motivated by this intuition, Brown Clustering algorithm (Brown et al., 1992) gives an unsupervised class label to a word. Our system uses a C++ implementation[3] of the Brown clustering algorithm (Liang, 2005) and sets cluster number as 50. The Brown clusters was trained on a large corpus of APT reports[4] provided by the organizer.

2.3 Model Introduction

Similar to (Ma and Hovy, 2016), as shown in Figure 1, before feeding into the BiLSTM network, the model concatenates character-level representations obtained from CNN (LeCun et al., 1989), word-level representations and linguistic feature representations to acquire the final representation of the word. At the end, the model feeds the output

[1]https://stanfordnlp.github.io/CoreNLP/
[2]https://opennlp.apache.org/
[3]https://github.com/percyliang/brown-cluster
[4]https://www.atp.dk

vectors of BiLSTM into a CRF layer, using sentence level tag information to jointly decode sequence labels.

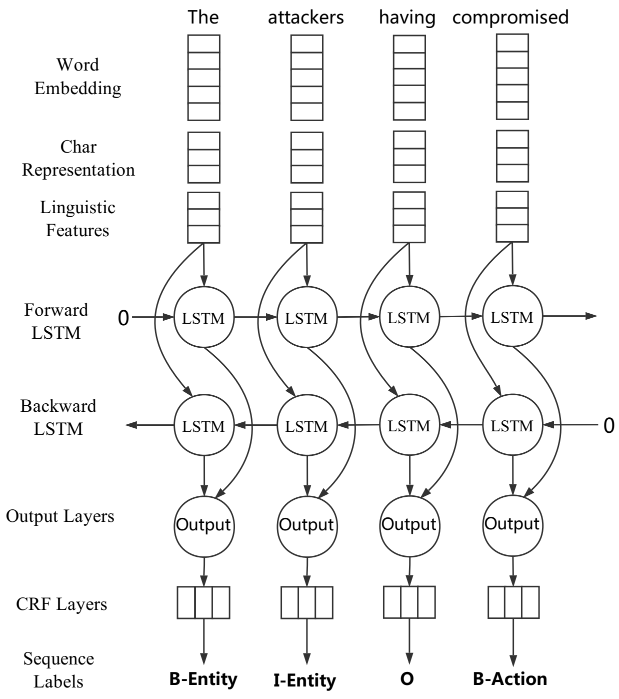

Figure 1: BiLSTM + CRF network architecture

Designing a neural network architecture with character representation as input is appealing for several reasons. First, words which have the same morphological properties(like the prefix or suffix of a word) often share the same grammatical function or meaning. Second, a character-level analysis can help to dual with the OOV (out-of-vocabulary) problem and the word starts with a capital letter may provide additional information. Previous studies (Santos and Zadrozny, 2014) (Chiu and Nichols, 2015) have shown that CNN is an effective approach to extract morphological information from characters of words, and consequently help to improve the performance of NER and POS tagging. As shown in Figure 2, we employ a max-pooling and a convolution layer to extract a new feature vector from character embeddings for each word. Then words are padded with a number of PADDING characters on both sides depending on the window size of the CNN.

2.3.1 BiLSTM for Word Representation

RNNs are well-studied solutions for a neural network to process variable length input and have

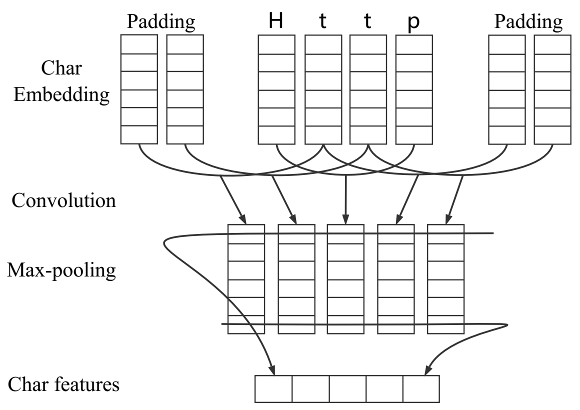

Figure 2: CNN feature extraction

long term memory. As a variant of RNNs, the long-short term memory (LSTM) unit with three multiplicative gates allows highly non-trivial long-distance dependencies to be easily learned. For sequence labeling tasks, we use a bidirectional LSTM network proposed in (Graves et al., 2013) in order to efficiently utilize both past features (via forward states) and future features (via backward states) for a specific time frame. Furthermore, pre-trained word embeddings learned from large unlabeled data are used.

2.3.2 Neural Network with Features

Before feeding into the BiLSTM network, we concatenate the char embedding, word embedding, POS embedding, NER embedding, Chunking labels embedding, Brown Clustering labels embedding as input.

2.3.3 CRF Layer

For sequence labeling task, such as POS tagging or NER, the output labels adjacent are often strongly related(e.g. I-ORG cannot follow B-PER or I-LOC in NER task of CoNLL2003). Therefore, we model BiLSTM networks jointly using a CRF layer to decode each label.

2.3.4 Model Average

Random initialization and shuffling order of training sentences introduce randomization into model training. During experiments, we found that model predictions vary considerably even when the same pre-trained data and parameters are used. In order to utilize the power of models ensembling and avoid over-fitting problem, we use a script provided by tensor2tensor to average values

Model	Phrase Level			Token Level		
	Precision	Recall	F1-score	Precision	Recall	F1-score
CRF	0.4867	0.2374	0.3191	0.5766	0.2739	0.3714
CRF with Linguistic Features	0.4971	0.2627	0.3438	0.5839	0.2911	0.3885
BiLSTM-CRF	0.5082	0.4305	0.4661	0.6273	0.4296	0.5100
BiLSTM-CRF with Linguistic Features	0.5265	0.4410	0.4799	0.6354	0.4437	0.5225
BiLSTM-CNN-CRF	0.5289	0.4461	0.4840	0.6370	0.4456	0.5243
BiLSTM-CNN-CRF with Linguistic Features	**0.5436**	**0.4623**	**0.4997**	**0.6428**	**0.4531**	**0.5315**

Table 1: Experiment results in phrase level and token level.

of variables in a list of checkpoint files generated by BiLSTM-CNN-CRF networks[5].

3 Experiments and Analysis

3.1 Experiment Settings

We found that in the original dataset provided by organizers, the average percentage of positive samples dropped from 23% in training set to less than 6.6% in development set, which suggests that a model may be strongly biased if trained on the training set and fine-tuned on the development set without randomization. Therefore we combined the training data and development data after spelling correction and removal of unreadable characters. In order to avoid over-fitting problem, we adopted 5-fold cross validation by randomly splitting the combined data into five folds. Each time we took four folds as training data and the rest as development data.

We downloaded GloVe (Pennington et al., 2014) data as the source of pre-trained word embeddings. For char and feature embeddings, we randomly initialized them with values drawn from the standard normal distribution.

The evaluation metrics were calculated by the CoNLL2000 Perl script at token level and phrase level. For each level, precision, recall and F1-score were calculated. Based on the highest F1-score we selected the best hyper-parameters (CNN output size, LSTM State size, learning rate, dropout, etc.) for single model in 5-fold cross validation. Besides, for the submission generated by the BiLSTM-CNN-CRF, we adopted model average strategy by averaging values of variables of 5 checkpoint files from 5 independent experiments sharing the same experiment settings.

3.2 Experiment Results

Our experiments focus on improving the performance in phrase level because of two motivations: phrase level is more meaningful than token level and the model superior in phrase level also outperforms the others in token level.

The comparison of models in Table 1 shows that neural networks models significantly outperform the traditional model based on CRF. Meanwhile models including character-level features in pre-trained word embeddings show better result. Last but not least, models with additional linguistic features improve the performance in both phrase level and token level significantly.

4 Conclusion

This paper describes our system for Subtask 1 and Subtask 2 of SemEval-2018 Task 8. For Subtask 2 we design a BiLSTM-CNN-CRF model combining several hand-crafted features, such as POS tagging, NER labels, Chunking labels, etc. For Subtask 1 we simply use the output labels generated by Subtask 2 to classify where a sentence is relevant to malware. It can be observed that rich linguistic features and pre-trained word embeddings for large unlabeled data benefit the task. The system is proven valid and effective to achieve the highest F1-score in Subtask 2.

5 Future Work

The fact is our system is not good enough to help semantic extraction from CybersecUrity reports. In the future, we will design a multi-task network to solve Subtask 1 and Subtask 2 simultaneously since they are highly related. Besides, more features, e.g., stemming and lemmatization, can be utilized in predicting token labels.

Acknowledgments

We thank the task organizers for providing malware corpus and organizing the shared task 8. Furthermore, we also appreciate the peer reviewers for their helpful comments on our first draft. We are especially grateful to Li Xiaoning for his insightful advice on the understanding of CybersecUrity reports.

[5]https://github.com/tensorflow/tensor2tensor

References

Peter F Brown, Peter V Desouza, Robert L Mercer, Vincent J Della Pietra, and Jenifer C Lai. 1992. Class-based n-gram models of natural language. *Computational linguistics*, 18(4):467–479.

Jason PC Chiu and Eric Nichols. 2015. Named entity recognition with bidirectional lstm-cnns. *arXiv preprint arXiv:1511.08308*.

Jon DiMaggio. 2015. The black vine cyberespionage group. *Retrieved January*, 26:2016.

Sean R Eddy. 1996. Hidden markov models. *Current opinion in structural biology*, 6(3):361–365.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. 2013. Speech recognition with deep recurrent neural networks. In *Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*, pages 6645–6649. IEEE.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551.

Percy Liang. 2005. *Semi-supervised learning for natural language*. Ph.D. thesis, Massachusetts Institute of Technology.

Swee Kiat Lim, Aldrian Obaja Muis, Wei Lu, and Chen Hui Ong. 2017. Malwaretextdb: A database for annotated malware articles. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1557–1567.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. *arXiv preprint arXiv:1603.01354*.

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

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Peter Phandi, Amila Silva, and Wei Lu. 2018. Semeval-2018 Task 8: Semantic Extraction from CybersecUrity REports using Natural Language Processing (SecureNLP). In *Proceedings of International Workshop on Semantic Evaluation (SemEval-2018)*, New Orleans, LA, USA.

Cicero D Santos and Bianca Zadrozny. 2014. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1818–1826.

Yushi Yao and Zheng Huang. 2016. Bi-directional lstm recurrent neural network for chinese word segmentation. In *International Conference on Neural Information Processing*, pages 345–353. Springer.