Improvement attempts on (Blevins & Zettlemoyer, 2020) for the Word Sense Disambiguation task

The NLPLab (11010ISA562100) Term Project Report by Group 14th

馮少迪[1]
109062470
527969624@qq.com

王脩評
109062602
nrasa2009@gmail.com

葉文照
109065801
wind.yeh@gmail.com

***Abstract*—This report presents our study and experiments for exploring possible any improvements at the task of Word Sense Disambiguation (WSD). We intend to replicated and expand upon the results of Facebook Research's Bi-Encoders solution, a model which they design to disambiguate these words (Blevins & Zettlemoyer, 2020). Specifically, we propose the following attempts: replacement of BERT with DISTILBERT, loss function augmentation, optimizer replacement and building a Tri-Encoders model. The following GitHub repository contains all code used in this report, which forked from original (Blevins & Zettlemoyer, 2020) [2] : https://github.com/fsdfsd123/wsd-biencoders-main**

Keywords—biencoders, bert, WSD, WordNet

I. Introduction

Natural Language Processing (NLP) has a historical task from 1949 that is Word Sense Disambiguation. (WSD) (Bevilacqua et al., 2021) The task of selecting the correct sense for a word and algorithms take as input a word in context and a fixed inventory of potential word senses and outputs the correct word sense in context.(Jurafsky & Martin, n.d.)

In the recent years, there are 3 major approaches of WSD – 1) Knowledge-Based 2) Surpervised and 3) Unsurpervised approach. (Ranjan Pal & Saha, 2015) We will focus on Supervised approach based on the empirical, that study papers, explore their method by experiments, then fuse to our method for improvements.

The works what we wish to focus our project. Terra Blevins and Luke Zettlemoyer propose Gloss Informed Bi-encoders (a supervised model) for WSD; they represent the target words and senses in the same embedding space by using a context encoder to represent the target word and surrounding context, and a gloss encoder to represent the sense definitions. Those two encoders are jointly learned from the WSD objective alone and trained in an end-to-end fashion. This model made outperform prior work on the English all-words WSD task introduced in (Raganato et al., 2017). According to their paper, shows that these gains come almost entirely from better performance on the less frequent senses, with an 15.6 absolute improvement in F1 performance over the closest performing system (Blevins & Zettlemoyer, 2020)

Principally, we propose to replicate and expand on Blevins and Luke Zettlemoyer propose Gloss Informed Bi-encoders for WSD. Initially, after replication of results, we replacement of BERT with DISTILBERT for seeking the computation saving, since each training epoch may cost 120 minutes on BERT that cost too much. And, when looking down the training log we found the incremental performance of each epochs' model is unstable, so we try loss function augmentation, optimizer replacement. In the end, our study has built a Tri-Encoders model for getting better performance.

II. Methods and Experiments

About the basic architecture of Terra Blevins and Luke Zettlemoyer propose Gloss Informed Bi-encoders, by according to their paper's description: "The overall model architecture is shown in Figure 1. The bi-encoder model consists of two independent encoders: (1) a context encoder, which represents the target word (and its surrounding context) and (2) a gloss encoder, that embeds the definition text for each word sense. These encoders are trained to embed each token near the representation of its correct word sense. Each encoder is a deep transformer network initialized with BERT, in order to leverage the word sense information it captures from pretraining."(Blevins & Zettlemoyer, 2020). The context encoder, which is defined as Tc, takes as input a context sentence c containing a set of target words w to be disambiguated, s.t. c = c0, c1, ..., wi , ..., cn, where wi is the ith target word, the encoder then produces a sequence of representations r, where

$$r_{wi} = T_C(C)[i]$$

For the target word represented as multiple subword pieces, we take the average value of these subword pieces after passing through the encoder. It can be expressed as:

$$r_{wi} = \frac{1}{k-j}\sum_{l=j}^{k}(T_c(c)[i])$$

The gloss encoder, defined as Tg, takes in a gloss gs = g0, g1, ..., gm that defines the sense s as input. The gloss encoder represents s as

$$r_s = T_g(g_s)[0]$$

where we take the first representation output by the gloss encoder (corresponding to the input [CLS] token) as a global representation for s.

[1] Authors are arranged by student number.

[2] https://github.com/facebookresearch /wsd-biencoders

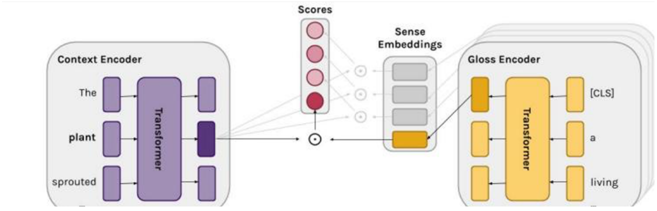

Fig. 1 Original (Blevins & Zettlemoyer, 2020) design architecture

A. Datasets

Training Datasset: We keep using the same dataset as original development, SemEval-2007 (SE07) (Pradhan et al., 2007) as development sets and evaluation sets. All sense glosses used in our system are retrieved from WordNet 3.0 (Miller, 1992). We train our model on SemCor, a large dataset manually annotated with senses from WordNet that contains 226,036 annotated examples covering 33,362 separate senses.

B. Trials

1) Pretrained language model replacement

After following the original papers' guidine to train the bi-encoder model by using BERT, we found each training epoch need around 2 hours that is costy. So, we try to adopt DISTILBERT pretrained model that each epoch cost 50 minutes arounding. Save time but few performance droping.

Traing 5 epochs and evaluating WSD model on semeval2007:

- f1 of BERT probe on semeval2007 test set = 71.9
- f1 of DISTILBERT probe on semeval2007 test set = 70.1

Although the DISTILBERT f1 score is lower than BERT at 5 epochs training but saving training time.

2) Normalize score $(sense \ \cdot target\ word)$

The original implementation scores each candidate sense $s \ \in \ S_w$ for a target word w by taking the dot product of r_w against every r_s for $s \ \in \ S_w$:

$$\phi(w, s_i) = r_w \cdot r_{si}$$

$$for\ i = 0, \dots, |S_w|$$

We tried normalizing score by pytorch torch.norm() function before sent to cross-entropy loss. Traing 5 epochs and evaluating WSD model on semeval2007:

- f1 of BERT probe on semeval2007 test set = 71.4
- f1 of DISTILBERT probe on semeval2007 test set = 70.3

This may not improve f1 score much but can make training process more stable. By comparing DISTILBERT training log,

original	Normalized dot product
Training probe... Dev f1 after 1 epochs = 62.6 Dev f1 after 2 epochs = 69.0 Dev f1 after 3 epochs = 70.1 Dev f1 after 4 epochs = 68.6 Dev f1 after 5 epochs = 67.9	Training probe... Dev f1 after 1 epochs = 61.3 Dev f1 after 2 epochs = 68.4 Dev f1 after 3 epochs = 69.7 Dev f1 after 4 epochs = 70.3 Dev f1 after 5 epochs = 70.1

3) optimizer replacement

Replace AdamW by AdaGrad as optimizer, although performance drop at beginning few epochs but when increase training epochs that may make positive.

Optimizer\Epoch	1	5	10
AdamW	0.51	0.65	0.77
AdaGrad	0.52	0.60	0.74

4) Tri-Encoder

Terra Blevins and Luke Zettlemoyer propose Gloss Informed Bi-encoders (a supervised model) for WSD. They use 2 separated encoders, one for Context encoder, the other is used for Gloss encoder. We propsed to use a new encoder, that is retrived by the doct product of (the dot product of "information:context-gloss" and gloss) and context.

On this basis, we propose to generate a new information: context-gloss data, make the dot product of "information:context-gloss" and gloss. After that, we do the dot product again for previous product result and context. Those will be used as input by a new encoder. So, we transform the Bi-Encoder to Tri-Encoder. The architecture of the model is showed as Figure2.

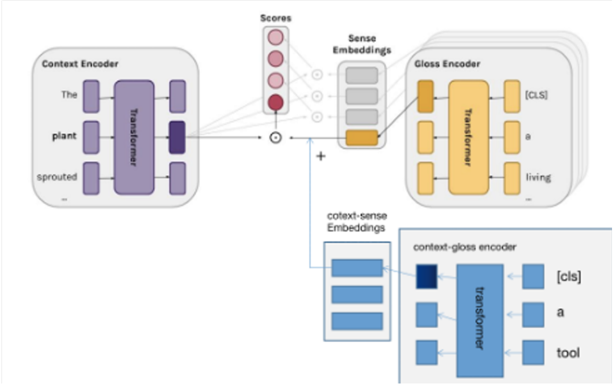

Fig.2 The Architecture of the Tri-Encoder model

Refer to the (Blevins & Zettlemoyer, 2020) paper for formulas about the Context Encoder and Gloss Encoder. Here will describe the context-gloss encoder: Go to replace the target word of the context with each possible sense。

Such as "I want to go to *hospital* to see a doctor.", the "hopital" is target word, so the possible sense will have:

1. an institution providing medical and surgical treatment and nursing care for sick or injured people.
2. a hospice, especially one run by the Knights Hospitaller.
3. a charitable institution for the education of the young.

After replacement, the content become:

1. I want to go to *an institution providing medical and surgical treatment and nursing care for sick or injured people* to see a doctor.
2. I want to go to *a hospice, especially one run by the Knights Hospitaller* to see a doctor.
3. I want to go to *a charitable institution for the education of the young* to see a doctor.

Those are called as the information of context-gloss. We define context-gloss encoder asF_q, $q = q_0, q_1 \dots q_i$, where q_i is the i^{th} context-gloss information. Where,

$$r_q = T_q(q_i)[0]$$

The 1st token is modified as ([CLS] token) for the representation of conntext-gloss。

At last, make score as:

$$score = r_s \cdot (r_s + r_q)$$

By this to get each sense's score, and the max score should be the label what prefered. Cross-entropy loss is adopted for loss function.

The experiment trained on signal word result[3] showed as Figure 3, After adding the context-gloss encoder, the Loss on the training set and the validation set will be relatively high at the beginning, but as the epoch increases, it will be lower than the model without the context-gloss encoder, and the model with the context-gloss encoder is added development accuracy will be slightly higher. Both train accuracy are similar. The experiment trained on all words are showed on Figure 4. We only trained 5 epoch cause the limit of time. The development accuracy on SE07 reached 75.48% after 5 epochs.

5) *Other unsuccessful attempts*

- In image recognition, average pooling is a method for synthesizing surrounding adjacent information. In the experiment, we averaged the embedding of each bert embedding and the adjacent word as the new embedding. But the fourth epoch ran out of memory and could not continue.
- In the process of running the original version of the program, the utilization rate of gpu is abnormally low, but there will be intermittent cpu burst. During the epoch, it is confirmed that the model and parameters have passed through cuda and GPU.
- The training time differs greatly between frozen and fine-tuned encoders.

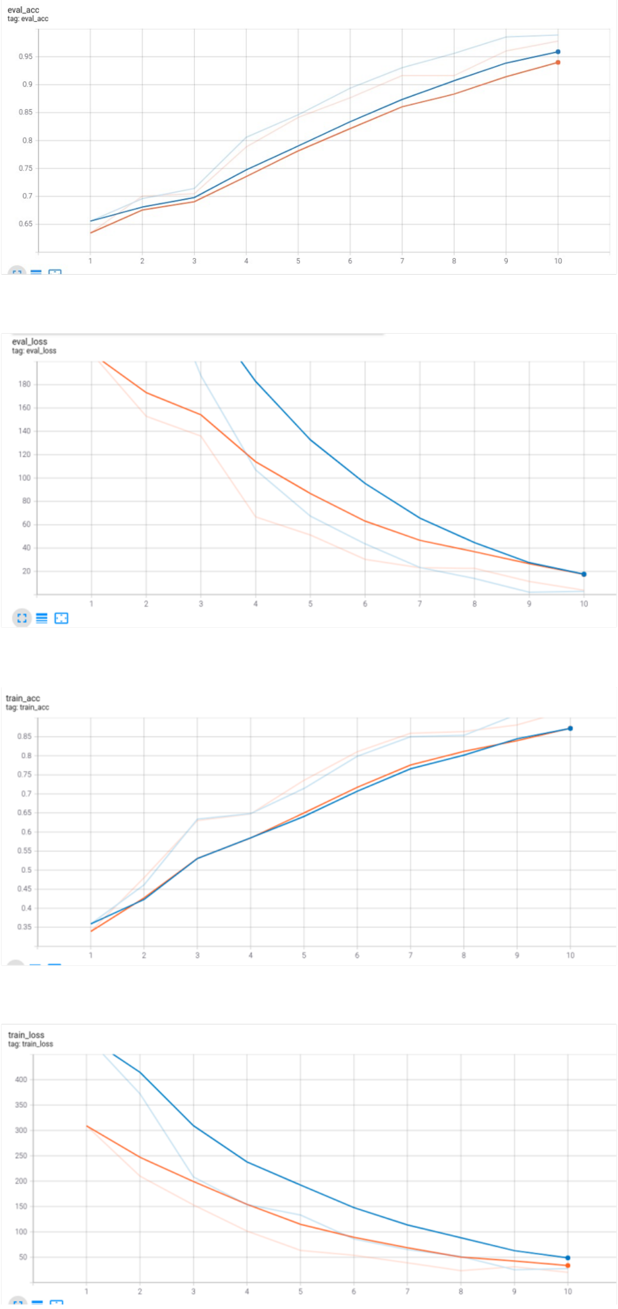

Fig. 3 The experiment result of Tri-Encoders design architecture on signal word data

[3] Detail of experiment result, please refer to the tensorboard: https://drive.google.com/drive/folders/1MIl9i0JoNGfNkYcFudedIc_ubvu6EifA?usp=sharing

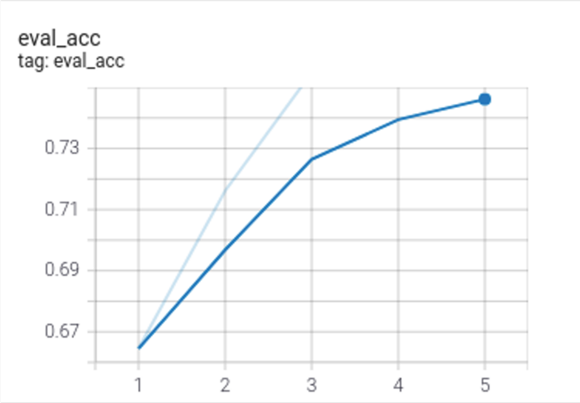

Fig.4 The experiment result of Tri-Encoders design architecture on all words data

III. CONCLUSION

Our results of such experiments may not exceed original Terra Blevins and Luke Zettlemoyer propose Gloss Informed Bi-encoders. Due to limited issue of development environment capability and time concern, we can’t afford to train as many epochs as need for reaching original implementation. But we do dig out some point can be addressed for enhance the WSD performance by enhance performance by efficiency training way and advanced architecture.

REFERENCES

Aliwy, A. H., & Taher, H. A. (2019). Word Sense Disambiguation: Survey Study. *Journal of Computer Science*, *15*(7), 1004–1011. https://doi.org/10.3844/jcssp.2019.1004.1011

Bevilacqua, M., Pasini, T., Raganato, A., & Navigli, R. (2021). Recent Trends in Word Sense Disambiguation: A Survey. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, 4330–4338. https://doi.org/10.24963/ijcai.2021/593

Blevins, T., & Zettlemoyer, L. (2020). Moving Down the Long Tail of Word Sense Disambiguation with Gloss Informed Bi-encoders. *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1006–1017. https://doi.org/10.18653/v1/2020.acl-main.95

Huang, L., Sun, C., Qiu, X., & Huang, X. (2020). GlossBERT: BERT for Word Sense Disambiguation with Gloss Knowledge. *ArXiv:1908.07245 [Cs]*. http://arxiv.org/abs/1908.07245

Jurafsky, D., & Martin, J. H. (n.d.). Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Computational Lexical Semantics 19.1 Word Sense Disambiguation: Overview. . . *Computational Lexical Semantics*, 51.

Miller, G. A. (1992). WordNet: A Lexical Database for English. *Commun. ACM*, *38*, 39–41.

Pradhan, S. S., Loper, E., Dligach, D., & Palmer, M. (2007). SemEval-2007 task 17: English lexical sample, SRL and all words. *Proceedings of the 4th International Workshop on Semantic Evaluations - SemEval ’07*, 87–92. https://doi.org/10.3115/1621474.1621490

Raganato, A., Camacho-Collados, J., & Navigli, R. (2017). Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 99–110. https://doi.org/10.18653/v1/E17-1010

Ranjan Pal, A., & Saha, D. (2015). Word Sense Disambiguation: A Survey. *International Journal of Control Theory and Computer Modeling*, *5*(3), 1–16. https://doi.org/10.5121/ijctcm.2015.5301