Self Attention Model for Natural Language Inference

Viet Nhat Nguyen
DISI, Univeristy of Trento
vietnhat.nguyen@studenti.unitn.it

Abstract

We studied two different classes of neural networks to address a Natural Language Inference task, which are Long Short Term Memory and Self-Attention. We compared the performances of our models when using different approaches and various of configurations and word embedding matrices. We found that models based solely on self-attention outperform model use RNNs only. Beside that, we experimented with adding POS tags to input as new features and proposed a simple combination of two approach to improve the accuracy with a trade-off of computational complexity.

1 Introduction

Natural Language Inference (NLI) is one of the central problems in Natural Language Understanding, which aims to determine whether a "hypothesis" is true (entailment), false (contradiction), or undetermined (neutral) given a "premise". This task, as learning the meaning of language, is vitally important since reasoning and inference are central to both human and artificial intelligence. One of the application is for controlling the consistency in chatbots and dialog systems.

The recent trend NLU approaches are based on fine-turning from a large model which was trained with multi-task learning manners. Probably the most important one is BERT (Devlin et al., 2018), a pre-training of Transformer architecture (Vaswani et al., 2017) which is based on attention mechanisms. Although BERT was described as marking the beginning of a new era in NLP, it is not the first work trying to dispensing with recurrence and convolutions entirely. In the best of our knowledge, (Parikh et al., 2016) is the first work implementing that idea. Specifically, they used Self-attention and Fully-connected Neural Networks for NLI problem, which yielded a lightweight model with competitive results. The aim of this project is not trying to archive high accuracy by using state-of-the-art models since they are expensive and resource consuming. Instead, we reproduce the method proposed in (Parikh et al., 2016) to demonstrate the power of attention mechanism in NLU comparing with an LSTM model. Beside that, we also add each POS tags to the corresponding words and propose a simple way to combine the two approaches above.

The remain of this report is organized as follow: we will start by investigating the properties of the dataset used in this project in section 2. In the section 3 after that we will present the approaches studied in this project which are LSTM model, model based on self attention, and a combination, respectively. Finally, experiment results and further discussion will be showed in section 4 and section 5.

2 Dataset

In this project, we study the Stanford Natural Language Inference (SNLI) corpus (Bowman et al., 2015). It is a collection of 570k human-written English sentence pairs manually labeled for balanced classification with the labels entailment, contradiction, and neutral. The dataset is already splitted into training set, validation set and test set with size 550k, 10k, 10k samples respectively. Each sample has the judgments of five mechanical turk workers and a consensus judgment which is a majority of votes. We discard the samples in which the annotations are not dominated by any category. Table 1 shows some examples drawn from traning set.

We analyze some characteristics of the dataset and present them in table 2. Although hypothesis sentences are typically shorter than premise sentences, there are many words in hypothesis that does not appear in premise. Beside that, sentences

Table 1: Some samples randomly drawn from training set

Premise	Annotations	Hypothesis
Two women are embracing while holding to go packages.	N E N N N Neutral	The sisters are hugging goodbye while holding tor just eating lunch.
A man selling donuts to a customer during a world exhibition event held in the city of Angeles	C C C C C Contradiction	A woman drinks her coffee in a small cafe.
A group of onlookers glance at a person doing a strange trick on her head.	E E E C E Entailment	People watch another person do a trick.
A small ice cream stand with two people standing near it.	C C C N C Contradiction	Two people selling ice cream from a car.

Table 2: Some characteristics of data examined from training set.

	Premise	Hypothesis
Max length	82	62
Average length	14.03	8.26
Number of distinct sentences	150737	479209
Vocab size	22918	36614

in premise set are also more divergent since the occurrence of them is mainly once.

3 Methods

First of all, we have to vectorize tokens in sentences, or apply word embedding specifically. In our model, following the work of (Parikh et al., 2016), we choose GLoVe (Pennington et al., 2014) which is performed on aggregated global word-word co-occurrence statistics, to present words in sentences. In particular, each embedding vector is normalized to have L_2 norm of 1. We randomly assign each out-of-vocabulary (OOV) word to one of 100 random vectors in which each was initialized to mean 0 and standard deviation 1.

We make a further step by trying to add information about POS tag of tokens in sentences to input representation. In particular, we convert each POS tag to one-hot encoding (there are 46 POS tags in dataset) then concatenate it with the word vectors described above. Fortunately, the sentences in dataset is provided with dependency grammar so we extract POS tags from them.

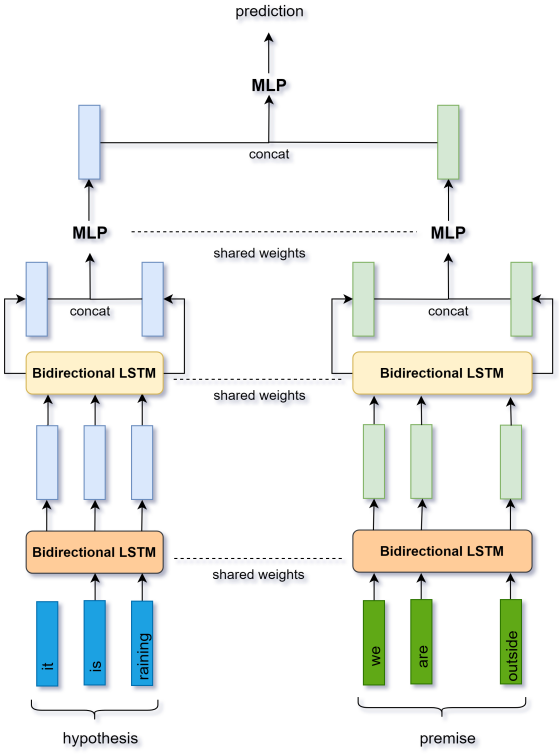

Figure 1: An overview of method.

3.1 Recurrent Neural Networks Models

Because Long Short Term Memory (LSTM) is widely used in NLP, we will applied this architecture to address NLI problem as a baseline model. Since our problem can be seen as a classification task, it is quite straightforward to use LSTM. Our baseline model is depicted in figure 1. It consists of 2 stacked layers in which each layer is a Bidirectional LSTM network. We set the hid-

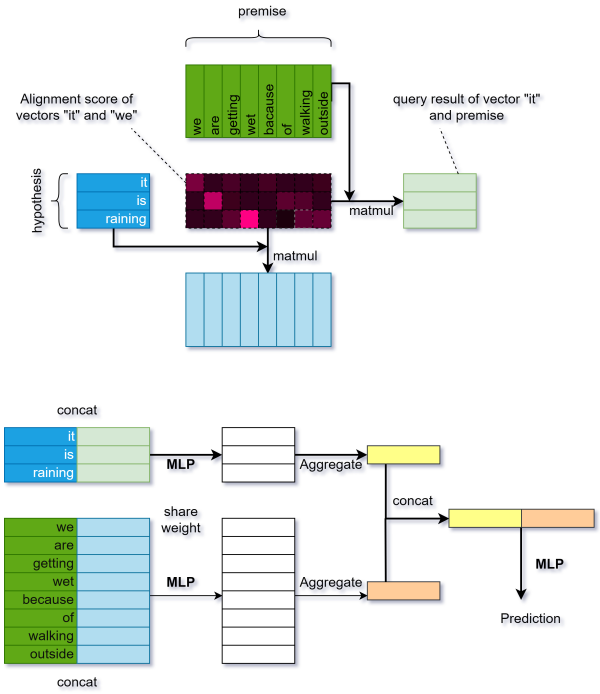

Figure 2: An overview of method.

den size for each each direction of bi-LSTM to be 200. We use the dropout regularization method (Srivastava et al., 2014) with dropout rate = 20%. Both premise sentence and hypothesis sentence is passed through this LSTM network which acts as a feature extractor to get 2 representation vectors respectively. These 2 vectors will be concatenated to 1 vector of 800 dimension and passed through a Multilayer perceptron which acts as a classifier to get the prediction.

3.2 A Decomposable Attention Model

Attention mechanism has been originally proposed and predominantly used in conjunction with LSTMs in machine translation (Bahdanau et al., 2015). Since then, it has become a trend in Natural Language Processing. The model presented in this section is purely based on word embedding, attention and feed-forward networks. Specifically, it includes 4 main steps: input representation, self attention, aggregation information from self attention step and classification. An overview of the method is illustrated in figure 2.

This model is composed of self attention operation and different multilayer perceptrons. In our implementation, we use an unique structure for multilayer perceptrons, denoted by $\mathcal{M}$. Explicitly, $\mathcal{M}$ consists of 2 fully-connected layers each has 200 hidden units with ReLU activation, dropout rate 0.2 which is also the rate used in LSTM network of baseline model.

3.2.1 Input Representation

Let us denote $\mathbf{a} = (a_1, a_2, ..., a_{l_a})$ and $\mathbf{b} = (b_1, b_2, ..., b_{l_b})$ be the two input sentences that have length l_a and l_b respectively. Similar to the baseline model, each a_i, b_j is a GLoVe embedding of a corresponding word in sentences. In this step, embedding vectors are projected to a vector having 200 dimensions independently through a linear function $\mathcal{L}$. Let $\bar{\mathbf{a}} = [\bar{a}_1, \bar{a}_2, ..., \bar{a}_{l_a}]$ and $\bar{\mathbf{b}} = [\bar{b}_1, \bar{b}_2, ..., \bar{b}_{l_b}]$ be the representations after projection, we have

$$\begin{aligned} \bar{a}_i &= \mathcal{L}(a_i) \\ \bar{b}_j &= \mathcal{L}(b_j) \end{aligned} \tag{1}$$

3.2.2 Self Attention

This module performs soft-align the elements of $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ and therefore is the crucial part of the model. We want to construct an alignment matrix $\mathbf{E}$ in which the element $e_{i,j}$ indicates the similarity between $\bar{a}_i$ and $\bar{b}_j$ through a metric which is a function $Score$. However, in order to avoid the quadratic complexity that would be associated with separately applying $Score$ function $l_a \times l_b$ times, we decompose the function $Score$ as the cosine distance after applying a multilayer perceptron $\mathcal{M}_1$ to each vector. By doing this, we reduce the number executions of $\mathcal{M}_1$ to $l_a + l_b$:

$$e_{i,j} = Score\left(\bar{a}_i, \bar{b}_i\right) = \mathcal{M}_1\left(\bar{a}_i\right)^T \mathcal{M}_1\left(\bar{b}_j\right) \tag{2}$$

Our target in this step is constructing $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ such as β_i is subphrase in $\bar{\mathbf{b}}$ that is (softly) aligned to $\bar{a}_i$ and vice versa for α_j. We can think β_i as a result of the query $\bar{a}_i$ agaist "database" $\bar{\mathbf{b}}$. We obtain $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by calculating weighted sum of $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$ with normalized weights derived from matrix $\mathbf{E}$ as in equation 3.

$$\begin{aligned} \alpha_j &= \sum_{i=1}^{l_a} \frac{\exp\left(e_{i,j}\right)}{\sum_{k=1}^{l_a} \exp(e_{i,k})} \bar{a}_i \\ \beta_i &= \sum_{j=1}^{l_b} \frac{\exp\left(e_{i,j}\right)}{\sum_{k=1}^{l_b} \exp(e_{k,j})} \bar{b}_j \end{aligned} \tag{3}$$

3.2.3 Aggregation

After having attention, we compare aligned phrases with the query. In other word, we combine

each $\bar{a}_i$ with the retrieval result β_i by a multilayer perceptron $\mathcal{M}_2$ defined previously:

$$u_i = \mathcal{M}_2\left([\bar{a}_i, \beta_i]\right)$$
$$v_j = \mathcal{M}_2\left(\left[\bar{b}_j, \alpha_i\right]\right) \quad (4)$$

where the brackets $[\cdot, \cdot]$ denote concatenation.

At this point we have two sets of comparison vectors $\mathbf{u}$ and $\mathbf{v}$. We can interpret u_i as the compatibility of word a_i against sentence $\mathbf{b}$ and vice versa. Now we aggregate over each set by taking summation and then feed the concatenation of two vectors into a multilayer perceptron $\mathcal{M}_3$ before passing it to a classifier.

$$\hat{x} = \mathcal{M}_3\left(\left[\sum_{i=1}^{l_a} u_i, \sum_{j=1}^{l_b} v_j\right]\right) \quad (5)$$

3.3 A hybrid method

Although self-attention model was proved to be efficient in this problem, it discards information about order of words in sentences. To take the advantage of this, we combine the two method above into one model. In particular, instead of computing the attention in the linear transformation of word embedding, we first feed two sentences into a LSTM then perform self-attention on top of output of LSTM component. We expect that this modification will take the strength from both then obtain better performance.

4 Experiment results

4.1 Setting

In our experiments we group samples of training set by lengths of premise and hypothesis, which means all samples in one batch have the same length of sentences. We implement our model in PyTorch using Adam optimizer(Kingma and Ba, 2015) with learning rate fixed at 2e-4. All the models is trained in 50 epochs. We examine the effects of changing word embedding to the accuracy by using three pre-trained word vectors models [1]. The first and second one were trained on Wikipedia dataset containing 6 bilion tokens, which have 50 and 300 dimensions. The third one having 300 dimensions was trained on a common crawled corpus involving 42 bilion. tokens. We will refer to these three embedding matrices as "50-6B", "300-6B" and "300-42B" respectively.

[1] https://nlp.stanford.edu/projects/glove/

Table 3: Test accuracy of different models

Methods	**Word Embedding**		
	50-6B	300-6B	300-42B
Baseline (LSTM)	0.7749	0.8011	0.8056
Attention	0.8459	0.8537	0.8559
Attention+LSTM (our)	**0.8490**	**0.8546**	**0.8575**

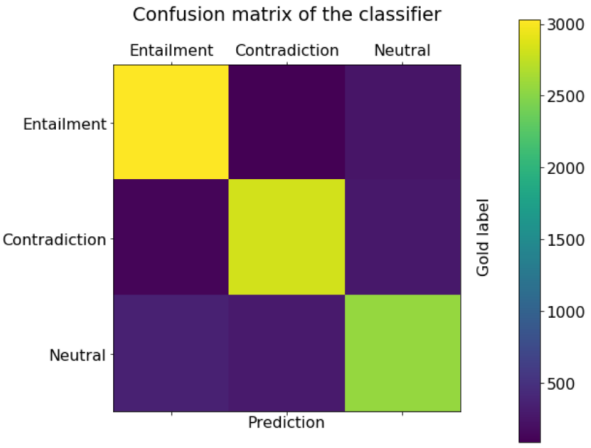

Figure 3: Confusion matrix of prediction

Table 3 shows the final test accuracy of different models with various word embedding matrices. We can see that the model solely based on self attention beats LSTM model with remarkable margins. Our model which is combine self-attention and LSTM is slightly better than the model with self-attention only. Nonetheless this combination expends high computational cost. Another interesting observation is about embedding matrices. Despite of being trained with the same corpus, the matrix having 300 dimensions gives better result than the one with 50 dimensions (the second column and the third column). On the other hand, the embedding matrix trained with corpus containing 42 billion tokens improves the accuracy comparing with the matrix having the same dimensions but was trained with smaller corpus.

Figure 3 visualizes how our model predicts over test test. It is quite successful at distinguishing a pair of premise and hypothesis is entailment or contradiction. It is reasonable that most of its mistakes are related to neutral samples.

Surperisingly, the new feature POS tags does not help much. The accuracies obtained by adding th new feature are roughly not change. We hypothesize that the POS tags are not relevant to the relationships between sentences or this information

Table 4: Test accuracy of different models

Method	Test accuracy	Train accuracy
LSTM-50	0.7820	0.8070
LSTM-100	0.8414	0.8790
LSTM-300	0.8399	0.9213
Attention-50	0.7966	0.7898
Attention-100	0.8414	0.8490
Attention-300	0.8506	0.8690

has been automatically inferred through tokens in sentences.

To understand the behaviours of LSTM model and self-attention model, we experiment with different size of hidden layers and present the obtained accuracies in table 4. We could see that the higher dimensions of hidden layers are, the more capacities the models have in order to fit the data. The difference between two models is that LSTM model tend to be overfitting than self-attention model, even we use the same regularization rate.

5 Discussion and conclusion

In this project we have implemented two approaches for Natural Language Inference problem, namely LSTM-based models and attention-based models. Interestingly, the models based on self attention only outperformed LSTM models. This shed some light on and affirmed the importance of attention in language understanding systems, as it notably has become a trend in recent years.

In addition, we also tried several of our ideas. Firstly we add the information about POS tags of words in sentences as a new feature. Unfortunately this modification was not advantageous. Secondly we combine the two basic approaches to attain a new type of model which improved the accuracy. However, this combination failed to maintain the advantage in computational expense of self-attention model. It is worth to attempt to combine these two approaches in a more effective way in further research.

Acknowledgments

This report is a part of final project in Language Understanding Systems course at the University of Trento, 2019.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. *CoRR*, abs/1508.05326.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *CoRR*, abs/1606.01933.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.