Comparison of Attention-based Methods in Natural Language Inference

Qianyu Cheng Luyu Jin Yiran Xu Xinsheng Zhang
Center For Data Science, New York University
{qc510, lj1035, yx1350, xz1757}@nyu.edu

Abstract

We reproduce four published methods for natural language inference (NLI). All the models are implemented with [1]PyTorch and evaluated on the Stanford Natural Language Inference ([2]SNLI) (Bowman et al., 2015) dataset. We compare their performance under different embedding techniques with best combination of tuned hyperparameters. Our results suggest that the Decomposable Attention Model outperforms the other three in terms of accuracy.

1 Introduction

One central problem in natural language understanding is Natural language Inference, also called Recognizing Textual Entailment (RTE), which explores the relationship between a premise and a hypothesis. There are three types of relationships: Entailment, Neutral, and Contradiction. Here are a few examples, in which "P" stands for "premise", "H" for "hypothesis", "L" for "label":

1. P: *Two older men in coats are standing outside.*
   H:*They are outside wearing coats.*
   L: *Entailment*

2. P: *A person on a horse jumps over a broken-down airplane.*
   H: *A person is training his horse for a competition.*
   L: *Neutral*

3. P: *Children are smiling and waving at camera.*
   H: *The kids are frowning.*
   L: *Contradiction*

NLI has a broad range of applications in other language understanding tasks, such as question answering, paraphrase identification, and document summarization. If a model can achieve relatively good performance on NLI, then one would believe that this model effectively represents sentence meanings.

There are two main classes of neural networks used to do NLI tasks. One is sentence encoder-based models, which encodes the premise and the hypothesis as two separate vectors fed through Multilayer Perceptron (MLP) classifiers. Another class is attention and memory models, in which encoding of the premise considers the alignment between the premise and the hypothesis.

Many researchers have proposed variants of attention and memory models. Rocktäschel et al. shows that the neural attention mechanism helps better summarizing the premise used to match the hypothesis (Rocktäschel et al., 2015). Wang et al. improve the performance of the model by using each of the attention-weighted representations of the premise for matching (Wang and Jiang, 2015). In order to reduce the expensive computation, Parikh et al. present a more effective model to decompose problem into paralleled sub-problems with attention (Parikh et al., 2016).

To compare and test those attention-based models, we built Continuous-Bag-of-Words (CBOW) (Mikolov et al., 2013) combined with MLP and Gated Recurrent Unit (GRU(Chung et al., 2014)) as our baseline models. Then we implemented

[1]GitHub: https://github.com/nyuxz/ds1011_final_project
[2]https://nlp.stanford.edu/projects/snli/

four attention-based neural networks such as GRUs with attention, GRUs with word-by-word attention, matching GRUs with word-by-word Attention, and Decomposable Attention Model.

We propose that the Decomposable Attention Model with 300 dimensional GloVe embedding performs the best among all the models and experiments we implemented.

2 Related Work and Background

Natural Language Inference tasks have been widely studied by many previous works.

Some early works of NLIs use feature-based models. Bowman et al. (2015) use a simple lexicalized classifier and attempt several variants of input features. However, the results are not as good as those of neutral network based mechanisms.

Recently, neural network based methods start to draw researchers' interests. Bowman et al. (2015) use two LSTMs to encode the premise and hypothesis separately, and achieve more robust performance to learn long-term dependencies.

The attention-based methods is proposed by Rocktäschel et al. (2015) to improve LSTM-based recurrent neural network. After neural attention mechanism is successfully applied in RNNs, it has been used to convolutional neural networks for NLI tasks (Yin et al., 2015). With the same attention mechanism, Wang et al. propose a better matching for attention-weighted representation, by which LSTMs can remember more important matches between the premise and hypothesis (Wang and Jiang, 2015).

In order to better solve the long-dependency issue, a Decomposable Attention mechanism has been proposed (Parikh et al., 2016). They decompose the task into sub-tasks and solve them separately.

At the same time, Bowman et al. build a tree structure model over a sequence based on shift-reduce parsing (Bowman et al., 2016). This architecture has better performance on semantic compositionality and solves ambiguities. To tackle the limitation that lower-level interactive features between two sentences cannot be captured precisely, a bilateral multi-perspective matching is proposed to achieve better performance on NLI tasks (Wang et al., 2017).

3 Data

We used Stanford Natural Language Inference (SNLI) dataset to train and evaluate our work. This dataset contains 570K sentence pairs; all pairs are labeled as entailment, contradiction, or neutral. The distributions for the three classes are balanced. Compared with previous datasets for NLI tasks, SNLI is superior in terms of corpus size and quality.

This dataset is created by asking humans to write a sentence that is definitely correct / maybe correct / definitely incorrect against a given sentence (a caption for a photo). The given sentence is called "premise" while the paired sentence created based on the premise is called "hypothesis". Each pair of premise and hypothesis has gone through five judgments in terms of "contradiction", "neutral", and "entailment" before given a gold label according to consensus judgment.

Since this year (2017), Multi-Genre Natural Language Inference (MultiNLI) corpus is being used as an alternative to SNLI corpus. MultiNLI has more diverse topics than SNLI and hence it is more challenging. However, due to the time limit, we have not trained our models on MultiNLI for this project.

4 Methods

4.1 Baseline

We implemented two baseline models to compare with more advanced models. One is CBOW with MLP, which is one simple and typical approach to NLI while being reasonably competitive. The other is plain GRU. In later subsections, we added various attention mechanisms to the GRU to demonstrate the advantages of attention.

4.1.1 CBOW with MLP

CBOW is a classic model of word embedding which predicts a center word from its surrounding context (Mikolov et al., 2013). In our case, we first got the mean of embedded word vectors for premise and hypothesis, and concatenated two means together. Then the concatenated vectors are fed into two layers of MLP with a softmax function to perform classification.

4.1.2 GRU

GRU are known for its capability of effectively storing long-distance dependency information. We independently encoded the premise and hypothesis using GRU, and then concatenated the vectors and fed into an MLP classifier.

4.2 Attention-based GRU

Attention mechanism has been widely used in natural language processing, where the model is able to focus on some certain regions of sentences. We gained the idea from Rocktäschel et al. to implement both attention and word-by-word attention.

For all the following GRU based models, the premise and the hypothesis are processed by two different GRUs. The memory state of the second GRU (i.e. GRU over the hypothesis) is initialized with the last cell state of the first GRU (i.e. GRU over the premise). We will now focus on describing the differences among different models.

4.2.1 GRU with attention

We used $\mathbf{Y}$ to denote the matrix of output vectors [$\mathbf{h}_1$,...,$\mathbf{h}_L$] from the first GRU and $\mathbf{h}_N$ to denote the last output vector. Then we produced:

$$\mathbf{M} = \tanh(\mathbf{W}^y\mathbf{Y} + \mathbf{W}^h\mathbf{h}_N \otimes \mathbf{e}_L), \quad (1)$$

$$\alpha = \text{softmax}(\mathbf{w}^T\mathbf{M}), \quad (2)$$

$$\mathbf{r} = \mathbf{Y}\alpha^T, \quad (3)$$

where $\mathbf{e}_L$ is a vector of 1s, α is a vector of attention weights, and $\mathbf{r}$ is a weighted representation of the premise.

The final classification is:

$$\mathbf{h}^* = \tanh(\mathbf{W}^p\mathbf{r} + \mathbf{W}^x\mathbf{h}_N) \quad (4)$$

4.2.2 GRU with word-by-word (wbw) Attention

One improvement of the attention mechanism proposed by Rocktaschel et al. is word-by-word attention model. The main idea is to introduce a series of attention weights for each hidden state of the premise and soft-align each weight of every word vector in hypothesis.

We defined $\mathbf{a}_k$ as the attention-weighted vector for a specific word x_k^t in the hypothesis:

$$\mathbf{a}_k = \sum_{j=1}^{M} \alpha_{kj}\mathbf{h}_j^s, \quad (5)$$

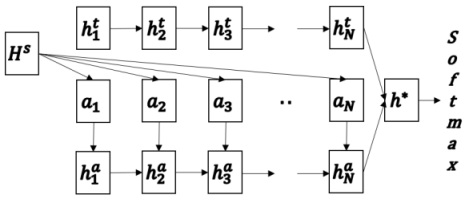

Figure 1: GRU with word-by-word attention (Wang and Jiang, 2015)

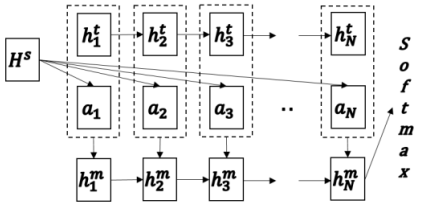

Figure 2: *m*GRU with word-by-word attention (Wang and Jiang, 2015)

where α_{kj} is an attention weight that measures how much $\mathbf{x}_k^t$ in the hypothesis is aligned with $\mathbf{x}_j^s$ in the premise and $\mathbf{h}_j^s$ is the hidden state corresponding to $\mathbf{x}_j^s$.

We generated α_{kj} as follows:

$$\alpha_{kj} = \frac{\exp(e_{kj})}{\sum_{j'}\exp(e_{kj'})}, \quad (6)$$

$$e_{kj} = \mathbf{w}^e \cdot \tanh\left(\mathbf{W}^s\mathbf{h}_j^s + \mathbf{W}^t\mathbf{h}_k^t + \mathbf{W}^a\mathbf{h}_{k-1}^a\right). \quad (7)$$

Then we built an RNN model over all attention-weighted premise by defining the following hidden states:

$$\mathbf{h}_k^a = \mathbf{a}_k + \tanh \mathbf{V}^a\mathbf{h}_{k-1}^a \quad (8)$$

Finally, we used $\mathbf{h}_N^a$ and $\mathbf{h}_N^t$ to make final predictions.

4.3 *m*GRU

Wang et al. (2015) proposed a new matching representation for sequential pairs of premise and hypothesis to improve LSTM with word-by-word attention. We used GRUs in our case. This approach takes in $\mathbf{a}_k$ and $\mathbf{h}_k^t$ as input:

$$\mathbf{m}_k = \begin{bmatrix}\mathbf{a}_k \\ \mathbf{h}_k^t\end{bmatrix} \quad (9)$$

First, we independently processed the premise and the hypothesis with two GRUs. In contrast to regular GRU with word-by-word attention, we did not feed the last cell state of the premise to the GRU of the hypothesis. Then, we still used $\mathbf{a}_k$ as Eqn(5), but with different e_{kj} as follows:

$$e_{kj} = \mathbf{w}^e \tanh \mathbf{W}^s \mathbf{h}_j^s + \mathbf{W}^t \mathbf{h}_j^t + \mathbf{W}^m \mathbf{h}_{k-1}^m \quad (10)$$

4.4 Decomposable Attention

This approach relies on alignment and is fully computationally decomposable with respect to the input text (Parikh et al., 2016). The core model consists of the following three components:

Attend. First, we soft-aligned the elements of $\bar{a}$ and $\bar{b}$ and computed unnormalized attention weights:

$$e_{ij} := F(\bar{a}_i, \bar{b}_i) := F(\bar{a}_i)^T F(\bar{b}_j), \quad (11)$$

where F is a feed-forward network.
The attention weights are then normalized as:

$$\beta_i = \sum_{j=1}^{\ell_b} \frac{\exp(e_{ij})}{\sum_{k=1}^{\ell_b} \exp(e_{ik})} \bar{b}_j,$$

$$\alpha_j = \sum_{i=1}^{\ell_a} \frac{\exp(e_{ij})}{\sum_{k=1}^{\ell_a} \exp(e_{kj})} \bar{a}_i. \quad (12)$$

Compare. Next, we separately compared the aligned phrases $\{(\bar{a}_i, \beta_i)\}_{i=1}^{\ell_a}$ and $\{(\bar{b}_j, \alpha_j)\}_{j=1}^{\ell_b}$ using a feed-forward network G:

$$\mathbf{v}_{1,i} := G([\bar{a}_i, \beta_i]) \quad \forall i \in [1, \ldots, \ell_a],$$
$$\mathbf{v}_{2,j} := G([\bar{b}_j, \alpha_j]) \quad \forall i \in [1, \ldots, \ell_b], \quad (13)$$

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

Aggregate. Finally, we aggregated the sets $\{\mathbf{v}_{1,i}\}_{i=1}^{\ell_a}$ and $\{\mathbf{v}_{2,j}\}_{j=1}^{\ell_b}$ by summation:

$$\mathbf{v}_1 = \sum_{i=1}^{\ell_a} \mathbf{v}_{1,i},$$

$$\mathbf{v}_2 = \sum_{j=1}^{\ell_b} \mathbf{v}_{2,j}. \quad (14)$$

Then we concatenate $\mathbf{v}_1$ and $\mathbf{v}_2$ and pass it through a feed-forward network H:

$$\hat{y} = H([\mathbf{v}_1, \mathbf{v}_2]), \quad (15)$$

where $\hat{y}$ represents the predicted scores for each class.

5 Experiments and Results

In this section, we compare our models with different attention-based methods: 1) with attention only based on the last context vector of the hypothesis, 2) with word-by-word attention based on all context vectors, 3) with matching GRU, and 4) with Decomposable Attention. Besides, we focus on our best-performing model and compared three different embedding settings with this model.

5.1 Best Hyperparameter Settings

We present the best combinations of hyperparameters in details below and results of these settings are shown in Table 1.

CBOW+MLP: Adam optimizer (Kingma and Ba, 2014), embedding dimension (500), hidden dimension (500), batch size (64), and learning rate (0.0001).

GRU: Adam optimizer, hidden dimension (159), batch size (32), and learning rate (0.0003).

GRU with attention: Adam optimizer, learning rate (0.001), batch size (32), hidden dimension (300), dropout (0.1), and ℓ_2 regularization strength (0.0001).

GRU with word-by-word attention: Adam optimizer, learning rate (0.0003), batch size (32), and hidden dimension (300).

***m*GRU:** Adam optimizer, learning rate (0.001, with decay ratio of 0.95 for every 30 epochs), batch size (30), and hidden dimension (150).

Decomposable Attention: Adagrad optimizer (Duchi et al., 2011), word embedding (300d GloVe), network size (2-layers, each with 200 neurons), batch size (32), dropout (0.2), and learning rate (0.05).

Following we only provide details of how we preprocessed data and dealt with embeddings about our best-performing model: Decomposable Attention Model. First, we prepended each sentence with a "NULL" token. During training, we padded each sentence up to the maximum length of the batch and semi-sorted the training data based on lengths of sentences.

After using 300 dimensional GloVe embeddings to represent words, we normalized each embedding vector to make its ℓ_2-norm equal 1 and projected this vector down to 200 dimensions. Besides, we hashed

Method	Batch Size	Hid Dim	Learning Rate	Dev (%)	Test (%)
CBOW + MLP	64	300	0.0001	74.9	73.7
GRU	32	159	0.0003	77.0	76.4
GRU + Attention	32	300	0.0003	78.4	77.8
GRU + wbwAttention	32	150	0.001	79.5	78.7
*m*GRU	30	150	0.001	83.2	82.6
Decomposable Attention	32	200	0.05	**85.6**	**84.8**

Table 1: Experiment results of different methods under best settings of hyperparameters.

out-of-vocabulary (OOV) words to one of 100 random embeddings with each initialized with mean of 0 and standard deviation of 1, whereas we initialized other parameter weights with mean of 0 and standard deviation of 0.01.

5.2 Results and Error Analysis

5.2.1 Methods Comparison

Table 1 compares the performance of various models we evaluated with the same 300 dimensional GloVe embedding setting. We can see that the Decomposable Attention model achieves the best test accuracy of 84.8%, followed by *m*GRU of 82.6% and GRU with word-by-word attention of 78.7%.

It makes sense that GRU with attention mechanism outperforms plain GRU model, since attention-based models produce context vectors which can summarize the premise information and is helpful for attending to the hypothesis.

As for the GRU with word-by-word attention model, it can detect whether the hypothesis is simply the paraphrase of the premise. For example, the hypothesis might simply be the reordering of the premise or use synonyms. Therefore, it is reasonable that the GRU with word-by-word attention model outperforms the GRU with attention model because word-by-word attention places more attention weights on more informative words instead of using attention only based on the last layer context vector in the hypothesis.

Within expectation, *m*GRU with word-by-word attention performs better than regular GRU with word-by-word attention since *m*GRU down-weights less important matches, like matches on stopwords. In addition, it remembers more important matching results like some particular content words. For example, when we sequentially process the hypothesis, if a word in the hypothesis mismatches another word of the corresponding premise, then it is very likely that there exists a contradiction due to this mismatch.

In contrast to all previous approaches, the Decomposable Attention model does not take into account of word orders and depends only on attention. Although this model has fewer parameters, it is superior to other models. This further proves the effectiveness of the attention mechanism.

5.2.2 Embedding Comparison

Embedding	Dev (%)	Test (%)
300D GloVe	85.6	84.8
300D fastText	80.7	79.7
100D character n-grams	84.7	83.8

Table 2: Experiment results of Decomposable Attention model with different pretrained embeddings: GloVe, fastText, character n-grams.

To understand how different pretrained embeddings will affect model performance in NLI tasks. we trained the Decomposable Attention Model with different embeddings.

As shown in Table 2, this model performs relatively poor with fastText, whereas perform almost equally well with GloVe and character-level embedding. One reason of this discrepancy in performance might be that each embedding is trained with a different cost function and on a different corpus, which implies that different underlying structures are captured during the training process.

According to Wang et. al (2015), GloVe cover more words in the SNLI corpus than fastText. As a result, the model encounters more OOV words with fastText. So another reason of the discrepancy might be resulted from OOV words.

6 Summary and Future Work

We implemented different attention-based neural networks and compared their performance. The Decomposable Attention Model achieves both the best development accuracy and the best test accuracy. With the best model, we examined three types of pretrained embeddings: GloVe, fastText and character n-grams with different embedding dimensions. Among all of our experiments, we found that 300 dimensional GloVe performs the best.

In future work, we would like to evaluate these models on MultiNlI dataset. We also plan to try tree-RNNs with attention.

Collaboration Statement

Qianyu Cheng: Implemented GRU with attention and GRU with word-by-word attention. Tested out performances and tunes parameters of models.
Luyu Jin: Implemented Decomposable Attention Model. Ran experiments on different embedding settings.
Yiran Xu: Implemented two baseline models. Ran experiments for various models (Baseline, Decomposable Attention and GRU models).
Xinsheng Zhang: Implemented *m*GRU and ran different experiments to find best hyperparameters of *m*GRU as well as words embedding setting.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. 2016. A fast unified model for parsing and sentence understanding. *CoRR*, abs/1603.06021.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12:2121–2159, July.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*.

Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Tomás Kociský, and Phil Blunsom. 2015. Reasoning about entailment with neural attention. *CoRR*, abs/1509.06664.

Shuohang Wang and Jing Jiang. 2015. Learning natural language inference with LSTM. *CoRR*, abs/1512.08849.

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *CoRR*, abs/1702.03814.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2015. ABCNN: attention-based convolutional neural network for modeling sentence pairs. *CoRR*, abs/1512.05193.