Enhancing Sentence Embedding with Generalized Pooling

Qian Chen
University of Science and Technology of China
cq1231@mail.ustc.edu.cn

Zhen-Hua Ling
University of Science and Technology of China
zhling@ustc.edu.cn

Xiaodan Zhu
ECE, Queen's University
xiaodan.zhu@queensu.ca

Abstract

Pooling is an essential component of a wide variety of sentence representation and embedding models. This paper explores generalized pooling methods to enhance sentence embedding. We propose vector-based multi-head attention that includes the widely used max pooling, mean pooling, and scalar self-attention as special cases. The model benefits from properly designed penalization terms to reduce redundancy in multi-head attention. We evaluate the proposed model on three different tasks: natural language inference (NLI), author profiling, and sentiment classification. The experiments show that the proposed model achieves significant improvement over strong sentence-encoding-based methods, resulting in state-of-the-art performances on four datasets. The proposed approach can be easily implemented for more problems than we discuss in this paper.

1 Introduction

Distributed representation learned with neural networks has shown to be effective in modeling natural language at different granularities. Learning representation for words (Bengio et al., 2000; Mikolov et al., 2013; Pennington et al., 2014), for example, has achieved notable success. Much remains to be done to model larger spans of text such as sentences or documents. The approaches to computing sentence embedding generally fall into two categories. The first consists of learning sentence embedding with unsupervised learning, e.g., auto-encoder-based models (Socher et al., 2011), Paragraph Vector (Le and Mikolov, 2014), SkipThought vectors (Kiros et al., 2015), FastSent (Hill et al., 2016), among others. The second category consists of models trained with supervised learning, such as convolution neural networks (CNN) (Kim, 2014; Kalchbrenner et al., 2014), recurrent neural networks (RNN) (Conneau et al., 2017; Bowman et al., 2015), and tree-structure recursive networks (Socher et al., 2013; Zhu et al., 2015; Tai et al., 2015), just to name a few.

Pooling is an essential component of a wide variety of sentence representation and embedding models. For example, in recurrent-neural-network-based models, pooling is often used to aggregate hidden states at different time steps (i.e., words in a sentence) to obtain sentence embedding. Convolutional neural networks (CNN) also often uses max or mean pooling to obtain a fixed-size sentence embedding.

In this paper we explore generalized pooling methods to enhance sentence embedding. Specifically, by extending scalar self-attention models such as those proposed in Lin et al. (2017), we propose vector-based multi-head attention, which includes the widely used max pooling, mean pooling, and scalar self-attention itself as special cases. On one hand, the proposed method allows for extracting different aspects of the sentence into multiple vector representations through the multi-head mechanism. On the other, it allows the models to focus on one of many possible interpretations of the words encoded in the context vector through the vector-based attention mechanism. In the proposed model we design penalization terms to reduce redundancy in multi-head attention.

We evaluate the proposed model on three different tasks: natural language inference, author profiling, and sentiment classification. The experiments show that the proposed model achieves significant improvement over strong sentence-encoding-based methods, resulting in state-of-the-art performances on

Proceedings of the 27th International Conference on Computational Linguistics, pages 1815–1826
Santa Fe, New Mexico, USA, August 20-26, 2018.

four datasets. The proposed approach can be easily implemented for more problems than we discuss in this paper.

2 Related Work

There exist in the literature much previous work for sentence embedding with supervised learning, which mostly use RNN and CNN as building blocks. For example, Bowman et al. (2015) used BiLSTMs as sentence embedding for natural language inference task. Kim (2014) used CNN with max pooling for sentence classification. More complicated neural networks were also proposed for sentence embedding. For example, Socher et al. (2013) introduced Recursive Neural Tensor Network (RNTN) over parse trees to compute sentence embedding for sentiment analysis. Zhu et al. (2015) and Tai et al. (2015) proposed tree-LSTM. Yu and Munkhdalai (2017a) proposed a memory augmented neural networks, called Neural Semantic Encoder (NSE), as sentence embedding for natural language understanding tasks.

Some recent research began to explore inner/self-sentence attention mechanism for sentence embedding, which can be classified into two categories: self-attention network and self-attention pooling. Cheng et al. (2016) proposed an intra-sentence level attention mechanism on the base of LSTM, called LSTMN. For each step in LSTMN, it calculated the attention between a certain word and its previous words. Vaswani et al. (2017) proposed a self-attention network for the neural machine translation task. The self-attention network uses multi-head scaled dot-product attention to represent each word by weighted summation of all word in the sentence. Shen et al. (2017) proposed DiSAN, which is composed of a directional self-attention with temporal order encoded. Shen et al. (2018) proposed reinforced self-attention network (ReSAN), which integrate both soft and hard attention into one context fusion with reinforced learning.

Self-attention pooling has also been studied in previous work. Liu et al. (2016) proposed inner-sentence attention based pooling methods for sentence embedding. They calculate scalar attention between the LSTM states and the mean pooling using multi-layer perceptron (MLP) to obtain the vector representation for a sentence. Lin et al. (2017) proposed a scalar structure/multi-head self-attention method for sentence embedding. The multi-head self-attention is calculated by a MLP with only LSTM states as input. There are two main differences from our proposed method; i.e., (1) they used scalar attention instead of vectorial attention, (2) we propose different penalization terms which is suitable for vector-based multi-head self-attention, while their penalization term on attention matrix is only designed for scalar multi-head self-attention. Choi et al. (2018) proposed a fine-grained attention mechanism for neural machine translation, which also extend scalar attention to vectorial attention. Shen et al. (2017) proposes multi-dimensional/vectorial self-attention pooling on the top of self-attention network instead of BiLSTM. However, both of them didn't consider multi-head self-attention.

3 The Model

In this section we describe the proposed models that enhance sentence embedding with generalized pooling approaches. The pooling layer is built on a state-of-the-art sequence encoder layer. Below, we first discuss the sequence encoder, which, when enhanced with the proposed generalized pooling, achieves state-of-the-art performance on three different tasks on four datasets.

3.1 Sequence Encoder

The sequence encoder in our model takes into T word tokens of a sentence $\boldsymbol{S} = (w_1, w_2, \ldots, w_T)$. Each word w_i is from the vocabulary $\boldsymbol{V}$. For each word we concatenate pre-trained word embedding and embedding learned from characters. The character composition model feeds all characters of the word into a convolution neural network (CNN) with max pooling (Kim, 2014). The detailed experiment setup will be discussed in Section 4. The sentence $\boldsymbol{S}$ is represented as a word embedding sequence: $\boldsymbol{X} = (\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_T) \in \mathbb{R}^{T \times d_e}$, where d_e is the dimension of word embedding which concatenates embedding obtained from character composition and pretrained word embedding.

To represent words and their context in sentences, the sentences are fed into stacked bidirectional LSTMs (BiLSTMs). Shortcut connections are applied, which concatenate word embeddings and input

hidden states at each layer in the stacked BiLSTM except for the first (bottom) layer. The formulae are as follows:

$$\overrightarrow{\boldsymbol{h}}_t^l = \text{LSTM}([\boldsymbol{e}_t; \overrightarrow{\boldsymbol{h}}_t^{l-1}], \overrightarrow{\boldsymbol{h}}_{t-1}^l)\,, \tag{1}$$

$$\overleftarrow{\boldsymbol{h}}_t^l = \text{LSTM}([\boldsymbol{e}_t; \overleftarrow{\boldsymbol{h}}_t^{l-1}], \overleftarrow{\boldsymbol{h}}_{t+1}^l)\,, \tag{2}$$

$$\boldsymbol{h}_t^l = [\overrightarrow{\boldsymbol{h}}_t^l; \overleftarrow{\boldsymbol{h}}_t^l]\,. \tag{3}$$

where hidden states $\boldsymbol{h}_t^l$ in layer l concatenate two directional hidden states of LSTM at time t. Then the sequence is represented as the hidden states in the top layer L: $\mathbf{H}^L = (\boldsymbol{h}_1^L, \boldsymbol{h}_2^L, \ldots, \boldsymbol{h}_T^L) \in \mathbb{R}^{T\times 2d}$. For simplicity, we ignore the superscript L in the remainder of the paper.

3.2 Generalized Pooling

3.2.1 Vector-based Multi-head Attention

To transform a variable length sentence into a fixed size vector representation, we propose a generalized pooling method. We achieve that by using a weighted summation of the T LSTM hidden vectors, and the weights are vectors rather than scalars, which can control every element in all hidden vectors:

$$\mathbf{A} = \text{softmax}(\mathbf{W}_2\text{ReLU}(\mathbf{W}_1\mathbf{H}^{\mathrm{T}} + \boldsymbol{b}_1) + \boldsymbol{b}_2)^{\mathrm{T}}\,, \tag{4}$$

where $\mathbf{W}_1 \in \mathbb{R}^{d_a\times 2d}$ and $\mathbf{W}_2 \in \mathbb{R}^{2d\times d_a}$ are weight matrices; $\boldsymbol{b}_1 \in \mathbb{R}^{d_a}$ and $\boldsymbol{b}_2 \in \mathbb{R}^{2d}$ are bias, where d_a is the dimension of attention network and d is the dimension of LSTMs. $\mathbf{H} \in \mathbb{R}^{T\times 2d}$ and $\mathbf{A} \in \mathbb{R}^{T\times 2d}$ are the hidden vectors at the top layer and weight matrices, respectively. The softmax ensures that $(\mathbf{A}_1, \mathbf{A}_2, \ldots, \mathbf{A}_T)$ are non-negative and sum up to 1 for every element in vectors. Then we sum up the LSTM hidden states $\mathbf{H}$ according to the weight vectors provided by $\mathbf{A}$ to get a vector representation $\boldsymbol{v}$ of the input sentence.

However, the vector representation usually focuses on a specific component of the sentence, like a special set of related words or phrases. We extend pooling method to a multi-head way:

$$\mathbf{A}^i = \text{softmax}(\mathbf{W}_2^i\text{ReLU}(\mathbf{W}_1^i\mathbf{H}^{\mathrm{T}} + \boldsymbol{b}_1^i) + \boldsymbol{b}_2^i)^{\mathrm{T}}\,, \forall i \in 1, \ldots, I\,, \tag{5}$$

$$\boldsymbol{v}^i = \sum_{t=1}^{T} \boldsymbol{a}_t^i \odot \boldsymbol{h}_t^i\,, \forall i \in 1, \ldots, I\,, \tag{6}$$

where $\boldsymbol{a}_t^i$ indicates the vectorial attention from $\mathbf{A}^i$ for the t-th token in i-th head and $\odot$ is the element-wise product (also called the Hadamard product). Thus the final representation is a concatenated vector $\boldsymbol{v} = [\boldsymbol{v}^1; \boldsymbol{v}^2; \ldots; \boldsymbol{v}^I]$, where each $\boldsymbol{v}^i$ captures different aspects of the sentence. For example, some heads of vectors may represent the predicate of sentence and other heads of vectors represent argument of the sentence, which enhances representation of sentences obtained in single-head attention.

3.2.2 Penalization Terms

To reduce the redundancy of multi-head attention, we design penalization terms for vector-based multi-head attention in order to encourage the diversity of summation weight across different heads of attention. We propose three types of penalization terms.

Penalization Term on Parameter Matrices The first penalization term is applied to parameter matrix $\mathbf{W}_1^i$ in Equation 5, as shown in the following formula:

$$P = \mu \sum_{i=1}^{I} \sum_{j=i+1}^{I} \max(\lambda - \|\mathbf{W}_1^i - \mathbf{W}_1^j\|_{\mathrm{F}}^2, 0)\,. \tag{7}$$

Intuitively, we encourage different heads to have different parameters. We maximum the *Frobenius* norm of the differences between two parameter matrices, resulting in encouraging the diversity of different heads. It has no further bonus when the *Frobenius* norm of the difference of two matrices exceeds the a threshold λ. Similar to adding an L2 regularization term on neural networks, the penalization term P will be added to the original loss with a weight of μ. Hyper-parameters λ and μ need to be tuned on a development set. We can also add constrains on $\mathbf{W}_2^i$ in a similar way, but we did not observe further improvement in our experiments.

Penalization Term on Attention Matrices The second penalization term is added on attention matrices. Instead of using $\|\mathbf{A}\mathbf{A}^{\mathrm{T}} - \mathbf{I}\|_{\mathrm{F}}^2$ to encourage the diversity for scalar attention matrix as in Lin et al. (2017), we propose the following formula to encourage the diversity for vectorial attention matrices. The penalization term on attention matrices is

$$P = \mu \sum_{i=1}^{I} \sum_{j=i+1}^{I} \max(\lambda - \|\mathbf{A}^i - \mathbf{A}^j\|_{\mathrm{F}}^2, 0)\,, \tag{8}$$

where λ and μ are hyper-parameters which need to be tuned based on a development set. Intuitively, we try to encourage the diversity of any two different $\mathbf{A}^i$ under the threshold λ.

Penalization Term on Sentence Embeddings In addition, we propose to add a penalization term on multi-head sentence embedding $\boldsymbol{v}^i$ directly as follows:

$$P = \mu \sum_{i=1}^{I} \sum_{j=i+1}^{I} \max(\lambda - \|\boldsymbol{v}^i - \boldsymbol{v}^j\|_2^2, 0)\,, \tag{9}$$

where λ and μ are hyper-parameters. Here we try to maximize the l^2-norm of any two different heads of sentence embeddings under the threshold λ.

3.3 Top-layer Classifiers

The output of pooling is fed to a top-layer classifier to solve different problems. In this paper we evaluate our sentence embedding models on three different tasks: natural language inference (NLI), author profiling, and sentiment classification, on four datasets. The evaluation covers two typical types of problems. The author profiling and sentiment tasks classify individual sentences into different categories and the two NLI tasks classify sentence pairs.

For the NLI tasks, to enhance the relationship between sentence pairs, we concatenate the embeddings of two sentences with their absolute difference and element-wise product (Mou et al., 2016) as the input to the multilayer perceptron (MLP) classifier:

$$\boldsymbol{v} = [\boldsymbol{v}_a; \boldsymbol{v}_b; |\boldsymbol{v}_a - \boldsymbol{v}_b|; \boldsymbol{v}_a \odot \boldsymbol{v}_b]\,, \tag{10}$$

where $\odot$ is the element-wise product. The MLP has two hidden layers with *ReLU* activation with shortcut connections and a *softmax* output layer. The entire model is trained end-to-end through minimizing the cross-entropy loss. Note that for the two classification tasks on individual sentences (i.e., the author profiling and sentiment classification task), we use the same MLP classifiers described above for sentence pair classification. But instead of concatenating two sentences, we directly feed a sentence embedding into MLP.

4 Experimental Setup

4.1 Data

SNLI The SNLI (Bowman et al., 2015) is a large dataset for natural language inference. The task detects three relationships between a premise and a hypothesis sentence: the premise entails the hypothesis

(*entailment*), they contradict each other (*contradiction*), or they have a neutral relation (*neutral*). We use the same data split as in Bowman et al. (2015), i.e., 549.367 samples for training, 9,842 samples for development and 9,824 samples for testing.

MultiNLI MultiNLI (Williams et al., 2017) is another natural language inference dataset. The data are collected from a broader range of genres such as fiction, letters, telephone speech, and 9/11 reports. Half of these 10 genres are used in training while the rest are not, resulting in-domain and cross-domain development and test sets used to test NLI systems. We use the same data split as in Williams et al. (2017), i.e., 392,702 samples for training, 9,815/9,832 samples for in-domain/cross-domain development, and 9,796/9,847 samples for in-domain/cross-domain testing. Note that, we do not use SNLI as an additional training/development set in our experiments.

Age Dataset To compare our models with that of Lin et al. (2017), we use the same Age dataset in our experiment here, which is an Author Profiling dataset. The dataset are extracted from the Author Profiling dataset[1], which consists of tweets from English Twitter. The task is to predict the age range of authors of input tweets. The age range are split into 5 classes: 18-24, 25-34, 35-49, 50-64, 65+. We use the same data split as in Lin et al. (2017), i.e., 68,485 samples for training, 4,000 for development, and 4,000 for testing.

Yelp Dataset The Yelp dataset[2] is a sentiment analysis task, which takes reviews as input and predicts the level of sentiment in terms of the number of stars, from 1 to 5 stars, where 5-star means the most positive. We use the same data split as in Lin et al. (2017), i.e., 500,000 samples for training, 2,000 for development, and 2,000 for testing.

4.2 Training Details

We implement our algorithm with Theano (Theano Development Team, 2016) framework. We use the development set (in-domain development set for MultiNLI) to select models for testing. To help replicate our results, we publish our code[3], which is developed from our codebase for multiple tasks (Chen et al., 2018; Chen et al., 2017a; Chen et al., 2016; Zhang et al., 2017). Specifically, we use Adam (Kingma and Ba, 2014) for optimization. The initial learning rate is 4e-4 for SNLI and MultiNLI, 2e-3 for Age dataset, 1e-3 for Yelp dataset. For SNLI and MultiNLI dataset, stacked BiLSTMs have 3 layers. For Age and Yelp dataset, stacked BiLSTMs have 1 layer. The hidden states of BiLSTMs for each direction and MLP are 300 dimension, except for SNLI whose dimensions are 600. We clip the norm of gradients to make it smaller than 10 for SNLI and MultiNLI, and 0.5 for Age and Yelp dataset. The character embedding has 15 dimensions, and 1D-CNN filters lengths are 1, 3 and 5, respectively. Each filter has 100 feature maps, resulting in 300 dimensions for character-composition embedding. We initialize word-level embedding with pre-trained *GloVe-840B-300D* embeddings (Pennington et al., 2014) and initialize out-of-vocabulary words randomly with a Gaussian distribution. The word-level embedding is fixed during training for SNLI and MultiNLI dataset, but updated during training for Age and Yelp dataset, which is determined by the performance on development sets. The mini-batch size is 128 for SNLI and 32 for the rest. We use 5 heads generalized pooling for all tasks. And d_a is 600 for SNLI and 300 for the other datasets. For the penalization term, we choose $\lambda = 1$; the penalization weight μ is selected from [1,1e-1,1e-2,1e-3,1e-4] based on performances on the development sets.

5 Experimental Results

5.1 Overall Performance

For the NLI tasks, there are many ways to add cross-sentence (Rocktäschel et al., 2015; Parikh et al., 2016; Chen et al., 2017a) level attention. To ensure the comparison is fair, we only compare methods that use sentence-encoding-based models; i.e., cross-sentence attention is not allowed. Note that this

[1] http://pan.webis.de/clef16/pan16-web/author-profiling.html
[2] https://www.yelp.com/dataset/challenge
[3] https://github.com/lukecq1231/generalized-pooling

Model	Test
100D LSTM (Bowman et al., 2015)	77.6
300D LSTM (Bowman et al., 2016)	80.6
1024D GRU (Vendrov et al., 2015)	81.4
300D Tree CNN (Mou et al., 2016)	82.1
600D SPINN-PI (Bowman et al., 2016)	83.3
600D BiLSTM (Liu et al., 2016)	83.3
300D NTI-SLSTM-LSTM (Yu and Munkhdalai, 2017b)	83.4
600D BiLSTM intra-attention (Liu et al., 2016)	84.2
600D BiLSTM self-attention (Lin et al., 2017)	84.4
4096D BiLSTM max pooling (Conneau et al., 2017)	84.5
300D NSE (Yu and Munkhdalai, 2017a)	84.6
600D BiLSTM gated-pooling (Chen et al., 2017b)	85.5
300D DiSAN (Shen et al., 2017)	85.6
300D Gumbel TreeLSTM (Choi et al., 2017)	85.6
600D Residual stacked BiLSTM (Nie and Bansal, 2017)	85.7
300D CAFE (Tay et al., 2018)	85.9
600D Gumbel TreeLSTM (Choi et al., 2017)	86.0
1200D Residual stacked BiLSTM (Nie and Bansal, 2017)	86.0
300D ReSAN (Shen et al., 2018)	**86.3**
1200D BiLSTM max pooling	85.3
1200D BiLSTM mean pooling	85.0
1200D BiLSTM last pooling	84.9
1200D BiLSTM **generalized pooling**	**86.6**

Table 1: Accuracies of the models on the SNLI dataset.

follows the setup in the RepEval-2017 Shared Task. Table 1 shows the results of different models for NLI, consisting of results of previous work on sentence-encoding-based models, plus the performance of our baselines and that of the model proposed in this paper. We have three additional baseline models: the first uses max pooling on top of BiLSTM, which achieves an accuracy of 85.3%; the second uses mean pooling on top of BiLSTM, which achieves an accuracy of 85.0%; the third uses last pooling, i.e., concatenating the last hidden states of forward and backward LSTMs, which achieves an accuracy of 84.9%. Instead of using heuristic pooling methods, the proposed sentence-encoding-based model with generalized pooling achieves a new state-of-the-art accuracy of 86.6% on the SNLI dataset; the improvement over the baseline with max pooling is statistically significant under the one-tailed paired t-test at the 99.999% significance level. The previous state-of-the-art model ReSAN (Shen et al., 2018) used a hybrid of hard and soft attention model with reinforced learning achieved an accuracy of 86.3%.

Table 2 shows the results of different models on the MultiNLI dataset. The first group is the results of previous sentence-encoding-based models. The proposed model with generalized pooling achieves an accuracy of 73.8% on the in-domain test set and 74.0% on the cross-domain test set; both improve over the baselines using max pooling, mean pooling and last pooling. In addition, the results on cross-domain test set yield a new state of the art at an accuracy of 74.0%, which is better than 73.6% of shortcut-stacked BiLSTM (Nie and Bansal, 2017).

Table 3 shows the results of different models for the Yelp and the Age dataset. The BiLSTM with self-attention proposed by Lin et al. (2017) achieves better result than CNN and BiLSTM with max pooling. One of our baseline models using max pooling on BiLSTM achieves accuracies of 65.00% and 82.30% on the Yelp and the Age dataset respectively, which is already better than the self-attention model proposed by Lin et al. (2017). We also show that the results of baseline with mean pooling and last pooling, in which mean pooling achieves the best result on the Yelp dataset among three baseline models and max pooling achieves the best on the Age dataset among three baselines. Our proposed generalized pooling method obtains further improvement on these already strong baselines, achieving 66.55% on the Yelp dataset and 82.63% on the Age dataset (statistically significant $p < 0.00001$ against best baselines),

Model	In	Cross
CBOW (Williams et al., 2017)	64.8	64.5
BiLSTM (Williams et al., 2017)	66.9	66.9
BiLSTM gated-pooling (Chen et al., 2017b)	73.5	**73.6**
Shortcut stacked BiLSTM (Nie and Bansal, 2017)	**74.6**	**73.6**
BiLSTM max pooling	73.6	73.0
BiLSTM mean pooling	71.5	71.6
BiLSTM last pooling	71.6	71.9
BiLSTM **generalized pooling**	**73.8**	**74.0**

Table 2: Accuracies of the models on the MultiNLI dataset.

Model	Yelp	Age
BiLSTM max pooling (Lin et al., 2017)	61.99	77.30
CNN max pooling (Lin et al., 2017)	62.05	78.15
BiLSTM self-attention (Lin et al., 2017)	**64.21**	**80.45**
BiLSTM max pooling	65.00	82.30
BiLSTM mean pooling	65.30	81.78
BiLSTM last pooling	64.95	81.10
BiLSTM **generalized pooling**	**66.55**	**82.63**

Table 3: Accuracies of the models on the Yelp and Age dataset.

which are also new state of the art performances on these two datasets.

5.2 Detailed Analysis

Effect of Multiple Vectors/Scalars To compare the difference between vector-based attention and scalar attention, we draw the learning curves of different models using different heads on the SNLI development dataset without penalization terms as in Figure 1. The green lines indicate scalar self-attention pooling added on top of the BiLSTMs, same as in Lin et al. (2017), and the blue lines indicate vector-based attention used in our generalized pooling methods. It is obvious that the vector-based attention achieves improvement over scalar attention. Different line styles are used to indicate self-attention using different numbers of multi-head, ranging from 1, 3, 5, 7 to 9. For vector-based attention, the 9-head model achieves the best accuracy of 86.8% on the development set. For scalar attention, the 7-head model achieves the best accuracy of 86.4% on the development set.

Effect of Penalization Terms To analyze the effect of penalization terms, we show the results with/without penalization terms on the four datasets in Table 4. Without using any penalization terms, the proposed generalized pooling achieves an accuracy of 86.4% on the SNLI dataset, which is already slightly better than previous models (compared to accuracy 86.3% in Shen et al. (2018)). When we use penalization on parameter matrices, the proposed model achieves a further improvement with an accuracy of 86.6%. In addition, we also observe a significant improvement on MultiNLI, Yelp and Age dataset after using the penalization terms. For the MultiNLI dataset, the proposed model with penalization on parameter matrices achieves an accuracy of 73.8% and 74.0% on the in-domain and the cross-domain test set, respectively, which outperform the accuracy of 73.7% and 73.4% of the model without penalization, respectively. For the Yelp dataset, the proposed model with penalization on parameter matrices achieves the best results among the three penalization methods, which also improve the accuracy of 65.25% to 66.55% compared to the models without penalization. For the Age dataset, the proposed model with penalization on attention matrices achieves the best accuracy of 82.63%, compared to the 82.18% accuracy of the model without penalization. In general, the penalization on parameter matrices achieves the most effective improvement among most of these tasks, except for the Age dataset.

To verify whether the penalization term P discourages the redundancy in the sentence embedding, we

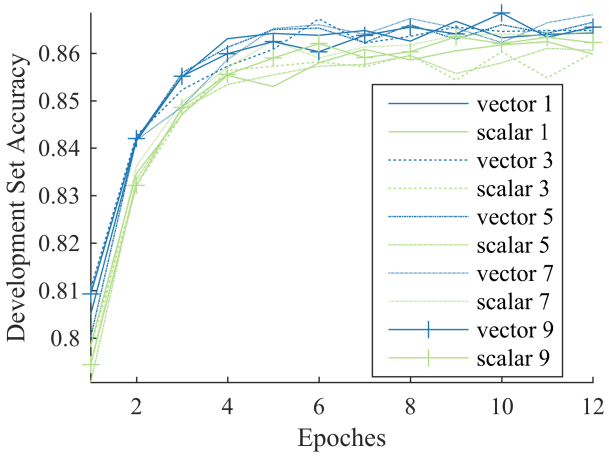


Figure 1: The effect of the number of heads and vectors/scalars in sentence embedding. The vertical axis indicates the development-set accuracy and the horizontal axis indicates training epochs. Numbers in the legend are the number of heads.

Model	**SNLI**	**MultiNLI**		**Yelp**	**Age**
		IN	**Cross**		
w/ Penalization on Parameter Matrices	**86.6**	**73.8**	**74.0**	**66.55**	82.45
w/ Penalization on Attention Matrices	86.2	73.6	73.8	66.15	**82.63**
w/ Penalization on Sentence Embeddings	86.1	73.5	73.6	65.75	82.15
w/o Penalization	86.4	73.7	73.4	65.25	82.18

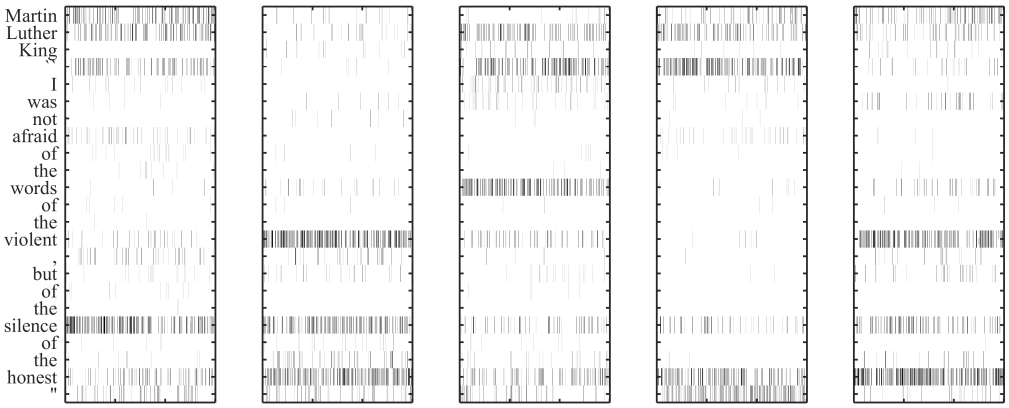
Table 4: Performance with/without the penalization term. The penalization weight is selected from [1,1e-1,1e-2,1e-3,1e-4] on the development sets.

visualize the vectorial multi-head attention according. We compare two models with the same hyper-parameters except that one is with penalization on attention matrices and the other without penalization. We pick a sentence from the development set of the Age data: *Martin Luther King "I was not afraid of the words of the violent, but of the silence of the honest"* , with the gold label being the category of 65+. We plot all 5 heads of attention matrices as in Figure 2. From the figure we can tell that the model trained without the penalization term has much more redundancy between different heads of attention (Figure 3b), resulting in putting significant focus on the word "Martin" in the 1st, 3rd and 5th head, and on the word "violent" in the 2nd and 4th head. However in Figure 3a, the model with penalization shows much more variation between different heads.

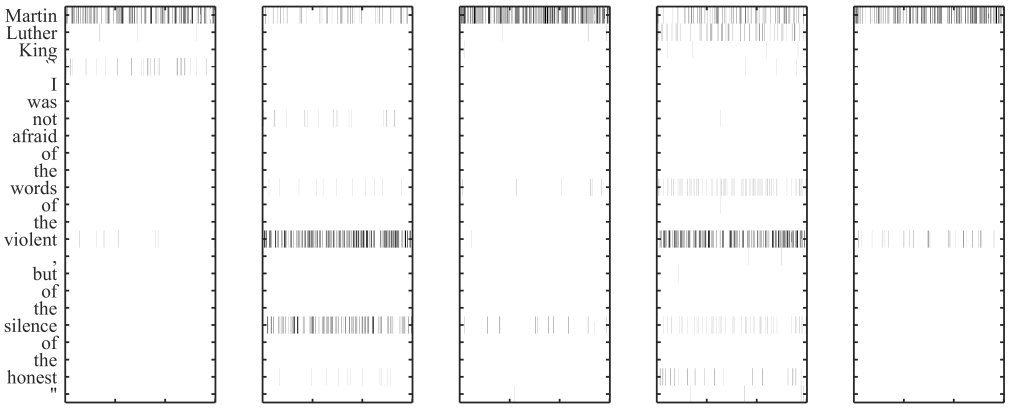
6 Conclusions

In this paper, we propose a generalized pooling method for sentence embedding through vector-based multi-head attention, which includes the widely used max pooling, mean pooling, and scalar self-attention as its special cases. Specifically the proposed model aims to use vectors to enrich the expressiveness of attention mechanism and leverage proper penalty terms to reduce redundancy in multi-head attention. We evaluate the proposed approach on three different tasks: natural language inference, author profiling, and sentiment classification. The experiments show that the proposed model achieves significant improvement over strong sentence-encoding-based methods, resulting in state-of-the-art performances on four datasets. The proposed approach can be easily implemented for more problems than we discuss in this paper.

Our future work includes exploring more effective MLP to use the structures of multi-head vectors, inspired by the idea from Lin et al. (2017). Leveraging structure information from syntactic and semantic parses is another direction interesting to us.



(a) Age dataset with penalization



(b) Age dataset without penalization

Figure 2: Visualization of vectorial multi-head attention. The vertical and horizontal axes indicate the source word tokens and the 600 dimensions of the attention $\mathbf{A}^i$ for different heads.

Acknowledgments

This work was partially funded by the National Natural Science Foundation of China (Grant No. U1636201) and the Key Science and Technology Project of Anhui Province (Grant No. 17030901005).

References

Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. 2000. A neural probabilistic language model. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*, pages 932–938.

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 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642.

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. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, and Hui Jiang. 2016. Distraction-based neural networks for modeling document. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 2754–2760.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017a. Enhanced LSTM for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1657–1668.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Recurrent neural network-based sentence encoder with gated attention for natural language inference. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP, RepEval@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 36–40.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Diana Inkpen, and Si Wei. 2018. Neural natural language inference models enhanced with external knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018*.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 551–561.

Jihun Choi, Kang Min Yoo, and Sang-goo Lee. 2017. Unsupervised learning of task-specific tree structures with tree-lstms. *CoRR*, abs/1707.02786.

Heeyoul Choi, Kyunghyun Cho, and Yoshua Bengio. 2018. Fine-grained attention mechanism for neural machine translation. *Neurocomputing*.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 670–680.

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*, pages 1367–1377.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pages 655–665.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1746–1751.

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

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3294–3302.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pages 1188–1196.

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *CoRR*, abs/1703.03130.

Yang Liu, Chengjie Sun, Lei Lin, and Xiaolong Wang. 2016. Learning natural language inference using bidirectional LSTM model and inner-attention. *CoRR*, abs/1605.09090.

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

Lili Mou, Rui Men, Ge Li, Yan Xu, Lu Zhang, Rui Yan, and Zhi Jin. 2016. Natural language inference by tree-based convolution and heuristic matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*.

Yixin Nie and Mohit Bansal. 2017. Shortcut-stacked sentence encoders for multi-domain inference. In *Proceedings of the 2nd Workshop on Evaluating Vector Space Representations for NLP, RepEval@EMNLP 2017, Copenhagen, Denmark, September 8, 2017*, pages 41–45.

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2249–2255.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

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

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Shirui Pan, and Chengqi Zhang. 2017. Disan: Directional self-attention network for rnn/cnn-free language understanding. *CoRR*, abs/1709.04696.

Tao Shen, Tianyi Zhou, Guodong Long, Jing Jiang, Sen Wang, and Chengqi Zhang. 2018. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. *CoRR*, abs/1801.10296.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011. Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 151–161.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642. Association for Computational Linguistics.

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 Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1556–1566.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018. A compare-propagate architecture with alignment factorization for natural language inference. *CoRR*, abs/1801.00102.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv e-prints*, abs/1605.02688, May.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6000–6010.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2015. Order-embeddings of images and language. *CoRR*, abs/1511.06361.

Adina Williams, Nikita Nangia, and Samuel R. Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *CoRR*, abs/1704.05426.

Hong Yu and Tsendsuren Munkhdalai. 2017a. Neural semantic encoders. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 397–407.

Hong Yu and Tsendsuren Munkhdalai. 2017b. Neural tree indexers for text understanding. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers*, pages 11–21.

Junbei Zhang, Xiaodan Zhu, Qian Chen, Lirong Dai, Si Wei, and Hui Jiang. 2017. Exploring question understanding and adaptation in neural-network-based question answering. *CoRR*, abs/arXiv:1703.04617v2.

Xiaodan Zhu, Parinaz Sobhani, and Hongyu Guo. 2015. Long short-term memory over recursive structures. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 1604–1612.