Multi-Scale Self-Attention for Text Classification

Qipeng Guo,[‡*] **Xipeng Qiu,**[‡†] **Pengfei Liu,**[‡] **Xiangyang Xue,**[‡] **Zheng Zhang**[§]
[‡]Shanghai Key Laboratory of Intelligent Information Processing, Fudan University
[‡]School of Computer Science, Fudan University
[§]AWS Shanghai AI Lab
[§]New York University Shanghai
{qpguo16, xpqiu, pfliu14, xyxue}@fudan.edu.cn, zz@nyu.edu

Abstract

In this paper, we introduce the prior knowledge, multi-scale structure, into self-attention modules. We propose a Multi-Scale Transformer which uses multi-scale multi-head self-attention to capture features from different scales. Based on the linguistic perspective and the analysis of pre-trained Transformer (BERT) on a huge corpus, we further design a strategy to control the scale distribution for each layer. Results of three different kinds of tasks (21 datasets) show our Multi-Scale Transformer outperforms the standard Transformer consistently and significantly on small and moderate size datasets.

Introduction

Self-Attention mechanism is widely used in text classification tasks, and models based on self-attention mechanism like Transformer (Vaswani et al. 2017), BERT (Devlin et al. 2018) achieves many exciting results on natural language processing (NLP) tasks, such as machine translation, language modeling (Dai et al. 2019), and text classification. Recently, Radford et al. (2018) points out the weakness of self-attention modules, especially the poor performance on small and moderate size datasets. Although the pre-training on huge corpus could help this, we believe the fundamental reason is that self-attention module lacks suitable inductive bias, so the learning process heavily depends on the training data. It is hard to learn a model with good generalization ability from scratch on a limited training set without a good inductive bias.

Multi-Scale structures are widely used in computer vision (CV), NLP, and signal processing domains. It can help the model to capture patterns at different scales and extract robust features. Specific to NLP domain, a common way to implement multi-scale is the hierarchical structure, such as convolutional neural networks (CNN) (Kalchbrenner, Grefenstette, and Blunsom 2014), multi-scale recurrent neural networks (RNN) (Chung, Ahn, and Bengio 2016) tree-structured neural networks (Socher et al. 2013; Tai, Socher, and Manning 2015) and hierarchical attention (Yang et al. 2016). The principle behind these models is the characteristic of language: the high-level feature is the composition of low-level terms. With hierarchical structures, these models capture the local compositions at lower layers and non-local composition at high layers. This division of labor makes the model less data-hungry.

However, for self-attention modules, there is no restriction of composition bias. The dependencies between words are purely data-driven without any prior, leading to easily overfit on small or moderate size datasets.

In this paper, we propose a multi-scale multi-head self-attention (MSMSA), in which each attention head has a variable scale. The scale of a head restricts the working area of self-attention. Intuitively, a large scale makes the feature involving more contextual information and being more smooth. A small scale insists on the local bias and encourages the features to be outstanding and sharp. Based on MSMSA, we further propose multi-scale Transformer, consisting of multiple MSMSA layers. Different from the multi-scale in hierarchical structures, each layer of multi-scale Transformer consists of several attention heads with multiple scales, which brings an ability to capture the multi-scale features in a single layer.

Contributions of this paper are:

- We introduce the multi-scale structure into self-attention framework, the proposed model Multi-Scale Transformer can extract features from different scales.
- Inspired by the hierarchical structure of language, we further develop a simple strategy to control the scale distribution for different layers. Based on the empirical result on real tasks and the analysis from BERT, we suggest using more small-scale attention heads in shallow layers and a balanced choice for different scales in deep layers.
- The building block of Multi-Scale Transformer, multi-scale multi-head self-attention provides a flexible way to introduce scale bias (local or global), and it is a replacement of the multi-head self-attention and position-wise feed-forward networks.
- Results on three tasks (21 datasets) show our Multi-Scale Transformer outperforms the standard Transformer con-

[*]Work done during internship at AWS Shanghai AI Lab.
[†]Corresponding author: Xipeng Qiu (xpqiu@fudan.edu.cn).

sistently and significantly on small and moderate size datasets.

Background

Self-attention and its extend architecture, Transformer, achieved many good results on NLP tasks. Instead of utilizing CNN or RNN unit to model the interaction between different words, Transformer achieves pair-wised interaction by attention mechanism.

Mulit-Head Self-attention The main component of the Transformer is the multi-head dot-product attention, which could be formalized as follows. Given a sequence of vectors $\mathbf{H} \in \mathbb{R}^{N \times D}$, where N is the length of the sequence and the D is the dimension of the vector. When doing multi-head self-attention, the module projects the $\mathbf{H}$ into three matrices: the query $\mathbf{Q}$, the key $\mathbf{K}$ and the value $\mathbf{V}$. These three matrices would be further decomposed into N' sub-spaces which corresponds to the N' heads and each head has D' units.

$$\mathrm{MSA}(\mathbf{H}) = [\mathrm{head}_1, \cdots, \mathrm{head}_{N'}]\mathbf{W}^O, \tag{1}$$

$$\mathrm{head}_i = \mathrm{softmax}(\frac{\mathbf{Q}_i \mathbf{K}_i^T}{\sqrt{D'}})\mathbf{V}_i, \tag{2}$$

$$\mathbf{Q} = \mathbf{H}\mathbf{W}^Q, \mathbf{K} = \mathbf{H}\mathbf{W}^K, \mathbf{V} = \mathbf{H}\mathbf{W}^V, \tag{3}$$

where $\mathrm{MSA}(\cdot)$ represents the Multi-head Self-Attention, and $\mathbf{W}^Q$, $\mathbf{W}^K$, $\mathbf{W}^V$, $\mathbf{W}^O$ are learnable parameters.

Transformer Each layer in the Transformer consists of a multi-head self-attention and a FFN layer (also called Position-wise Feed-Forward Networks in Vaswani et al. (2017)). The hidden state of $l+1$ layer, $\mathbf{H}_{l+1}$ could be calculated as followings.

$$\mathbf{Z}_l = \mathrm{norm}(\mathbf{H}_l + \mathrm{MSA}(\mathbf{H}_l)), \tag{4}$$

$$\mathbf{H}_{l+1} = \mathrm{norm}(\mathbf{Z}_l + \mathrm{FFN}(\mathbf{Z}_l)), \tag{5}$$

where $\mathrm{norm}(\cdot)$ means the layer normalization (Ba, Kiros, and Hinton 2016). In addition, the Transformer augments the input features by adding a positional embedding since the self-attention could not capture the positional information by itself.

Despite its effectiveness on machine translation and language modeling, Transformer usually fails on the task with moderate size datasets due to its shortage of inductive bias.

Model

In the multi-head self-attention, each head captures the pairwise interactions between words in different feature space. Each head has the same scale of sentence length.

Scale-Aware Self-Attention To introduce the concept of multi-scale into the self-attention mechanism, we use a simple way to enhance the regular self-attention, named Scale-Aware Self-Attention (SASA) which is equal to the restricted self-attention which is proposed in Vaswani et al. (2017) but using a dynamic size of the window.

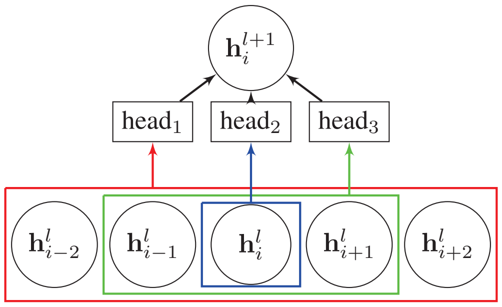


Figure 1: A diagram of Multi-Scale Multi-Head Self-Attention, we can see three heads which correspond to three different scales in the figure. The blue, green, red box illustrate the scale of $\omega = 1$, $\omega = 3$, $\omega = 5$, respectively.

Given a sequence of vectors $\mathbf{H} \in \mathbb{R}^{N \times D}$ with length N. SASA has a parameter ω which is either a constant number or a ratio according to the sequence length to control its working scope. An attention head can be computed as

$$\mathrm{head}(\mathbf{H}, \omega)_{i,j} = \mathrm{SM}(\frac{\mathbf{Q}_{ij}\mathrm{C}_{ij}(\mathbf{K}, \omega)^T}{\sqrt{D}})\mathrm{C}_{ij}(\mathbf{V}, \omega), \tag{6}$$

$$\mathrm{C}_{ij}(\mathbf{x}, \omega) = [\mathbf{x}_{i,j-\omega}, ..., \mathbf{x}_{i,j+\omega}], \tag{7}$$

$$\mathbf{Q} = \mathbf{H}\mathbf{W}^Q, \mathbf{K} = \mathbf{H}\mathbf{W}^K, \mathbf{V} = \mathbf{H}\mathbf{W}^V, \tag{8}$$

where i indicates the i-th head and j means j-th position. The SM represents "Softmax" function, C is a function to extract context for a given position. $\mathbf{W}^Q$, $\mathbf{W}^K$, $\mathbf{W}^V$ are learnable parameters. The scale of a head could be a variable like $\frac{N}{2}$ or a fixed number like 3.

Multi-Scale Multi-Head Self-Attention With SASA, we can implement a Multi-Scale Multi-Head Self-Attention (MSMSA) with multiple scale-aware heads. Each head works on different scales. For N' heads self-attention, MSMSA with scales $\Omega = [\omega_1, \cdots, \omega_{N'}]$ is

$$\mathrm{MSMSA}(\mathbf{H}, \Omega) = [\mathrm{head}_1(\mathbf{H}, \omega_1); \cdots; \mathrm{head}_{N'}(\mathbf{H}, \omega_{N'})]\mathbf{W}^O, \tag{9}$$

where $\mathbf{W}^O$ is a parameter matrix.

Compared to the vanilla multi-head self-attention, variable Ω controls the attended area and makes different heads have different duties.

Multi-Scale Transformer With the MSMSA, we could construct the multi-scale Transformer (MS-Transformer. In short, MS-Trans). Besides using MSMSA to replace the vanilla multi-head self-attention, we also remove the FFN (see Eq. (5)) because it could be view as a self-attention with scale $\omega = 1$ plus a non-linear activation function. Since MSMSA introduces the locality to the model, the MSMSA with a small scale can be an alternative to the positional embedding. Therefore, the multi-scale Transformer could be formalized as followings.

$$\mathbf{H}_{l+1} = \mathrm{norm}(\mathbf{H}_l + \mathrm{ReLU}(\mathrm{MSMSA}(\mathbf{H}_l), \Omega_l)) \tag{10}$$

where l is the layer index.

In this work, we limit the choice of scale size among constant numbers $\{1, 3, \cdots\}$ or variables depend on the sequence length $\{\frac{N}{16}, \frac{N}{8}, \cdots\}$. And we force the scale size to be an odd number.

Compared to the hierarchical multi-scale models, multi-scale Transformer allows the attention heads in one layer have various scales of vision, it is a "soft" version of viewing different layers as different scales.

Classification node We also find adding a special node at the beginning of each sequence and connecting it directly to the final sentence representation can improve the performance. This technique was introduced in BERT (Devlin et al. 2018), refer as "[CLS]". And it is also similar to the "relay node" in Guo et al. (2019). Different with them, we combine the "[CLS]" node and the feature from applying max-pooling over all the nodes in the final layer to represent the sentence. There is no difference between the "[CLS]" node and other tokens in the input sequence except it can directly contribute to the final representation.

Looking for Effective Attention Scales

Multi-scale Transformer is a flexible module in which each layer can have multi-scale attention heads. Therefore, an important factor is how to design the scale distribution for each layer.

Hierarchical Multi-Scale or Flexible Multi-Scale

A simple way to implement multi-scale feature extraction is following the hierarchical way, which stacks several layers with small-scale heads. The lower layers capture the local features and the higher capture the non-local features.

To verify the ability of hierarchical multi-scale, we design a very simple simulation task named mirrored summation. Given a sequence $\mathbf{A} = \{a_1, ..., a_N\}, \quad a \in \mathbb{R}^d$ and drawn from $\mathbf{U}(0, 1)$. The target is $\sum_{i=1}^{K} a_i \odot a_{N-i+1}$, where K is a fixed integer less than the sequence length N and $\odot$ means the Hadamard production. The minimum dependency path length is $N - K$, we use this task to test the ability of models for capturing long-range dependencies. Both train and test set are random generated and they have 200k samples each. We can assume the size of training set is enough to train these models thoroughly.

We use three different settings of MS-Trans.

1. MS-Trans-Hier-S: MS-Transformer with two layers, and each layer has 10 heads with a small scale $\omega = 3$.
2. MS-Trans-deepHier-S: MS-Transformer with six layers, and each layer has 10 heads with a small scale $\omega = 3$.
3. MS-Trans-Flex: MS-Transformer with two layers, and each layer has 10 heads with flexible multi-scales $\omega = 3, \frac{N}{16}, \frac{N}{8}, \frac{N}{4}, \frac{N}{2}$. Each scale has two heads.

As shown in Fig-2, Transformer achieves the best result, and MS-Trans-Flex follows with it. Although Transformer has the highest potential to capture long-range dependencies, it requires large training samples. Meanwhile,

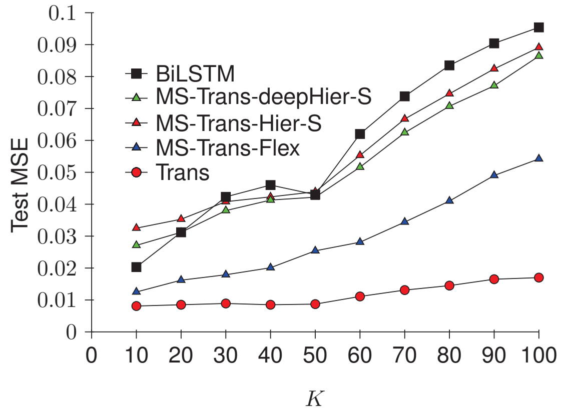


Figure 2: Mirrored Summation Task. The curve of MSE versus valid number K with the sequence length $n = 100$ and the dimension of input vectors $d = 50$.

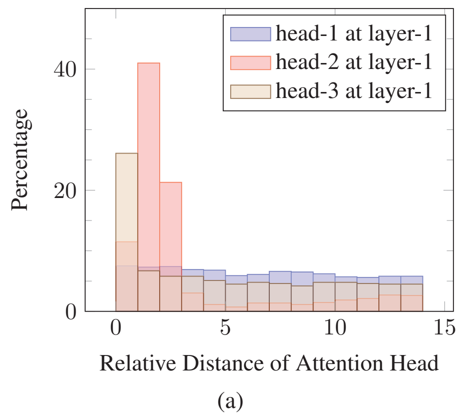
our model balance the data-hungry problem and the ability for capturing long-range dependencies. Based on the comparison of MS-Trans-Hier-S and MS-Trans-deepHier-S, we can find the improvement of additional layers is relatively small. According to the synthetic experiment and the performance on real tasks (see. Sec-), we think the large-scale heads are necessary for the lower layers and stacked small-scale heads are hard to capture long-range dependencies. In this case, a good prior should contain both small-scale and large-scale.

Scale Distributions of Different Layers

Since multi-scale is necessary for each layer, the second question is **how to design the proportion of different scales for each layer?** We think each layer may have its preference for the scale distribution. From the linguistic perspective, an intuitive assumption may be: the higher layer has a higher probability for the large-scale heads and the shallow layer has a higher probability for the small-scale heads.

To look for the empirical evidence, we probe several typical cases and analysis the corresponding behavior in a data-driven model, BERT (Bidirectional Encoder Representations from Transformers) (Devlin et al. 2018).

Analogy Analysis from BERT BERT is a pre-trained Transformer on large scale data, which has shown its power on many NLP tasks and it has good generalization ability. Since BERT is based on Transformer, it is not guided by prior knowledge, their knowledge is learned from data. We probe the behavior of BERT to see whether it fits the linguistic assumption. There are two aspects we want to study, the first is the working scales of attention heads of each layer in BERT. The second is the difference between the scale distributions of different layers, especially the preference of local and global relations. To probe these behaviors, we first run the BERT over many natural sentences and pick the highest activation of the attention map as the attention edge. Then



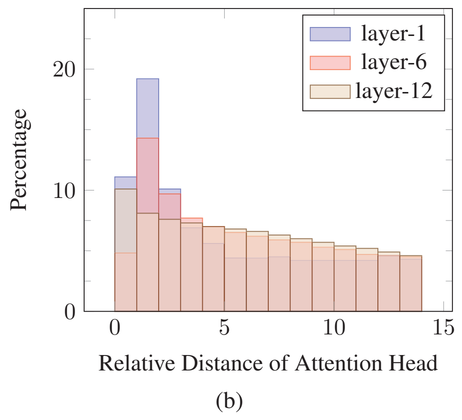


Figure 3: The visualization of BERT. (a) The attention distance distributions of three heads in the first layer. The red head only cares about the local pattern and the blue head equally looks at different distances. (b) The attention distance distribution of different layers. The shallow layer prefers the small scale size and tends to large scale size slowly when the layer gets deeper. Even in the final layer, local patterns still occupy a large percentage. We truncate the distance at $\frac{N}{2}$ for better visualization. The full figure can be found in the Appendix.

we record the relative distances of these attention edges. In this work, we obtain the data from running BERT on CoNLL03 dataset (see Tab-1).

We first draw the Fig-3a for observing the behavior of heads in the same layer. We pick three heads, and their difference is significant. As shown in the figure, "head-2" focus on a certain distance with a small scale, and "head-1" cover all the distances. There is a clear division of labor of these heads, and one layer can have both local and global vision via combining features from different heads.

The second figure Fig-3b shows the trend of distance preference when the depth of layer increased. We can find the model move from local vision to global vision slowly and shallow layers have a strong interest in local patterns.

The visualization of BERT fits the design of Multi-Scale Transformer, the first observation corresponds to the design of multi-scale multi-head self-attention which ask different heads focus on different scales, and the second observation provides a good reference of the trend of scale distribution across layers. Using such knowledge can largely reduce the requirement of training data for Transformer-like models.

Control Factor of Scale Distributions for Different Layers

From the intuitive linguistic perspective and empirical evidence, we design a control factor of scale distributions for different layers of multi-scale Transformer.

Let L denote the number of layers in multi-scale Transformer, $|\Omega|$ denote the number of candidate scale sizes, and n_k^l denote the number of heads for l-th layer and k-th scale size.

The head number n_k^l is computed by

$$z_k^l = \begin{cases} 0 & l = L \text{ or } k = |\Omega| \\ z_{k+1}^l + \frac{\alpha}{l} & k \in \{0, \cdots, |\Omega| - 1\} \end{cases} \tag{11}$$

$$n^l = \text{softmax}(z^l) \cdot N' \tag{12}$$

In the above equations, we introduce a hyper-parameter α to control the change of preference of scale sizes for each layer. For example, $\alpha = 0$ means all the layers use the same strategy of scale size, $\alpha > 0$ means the preference of smaller scale increased with the decline of layer depth, and $\alpha < 0$ indicates the preference of larger scale increased with the decline of layer depth. As the conclusion of analyzing BERT, we believe the deep layer has a balanced vision over both local and global patterns, so the top layer should be set to looking all the scale size uniformly. More specific, when $\alpha = 0.5$ and $N' = 10$, three layers have $n^{l=1} = \{5, 2, 2, 1, 0\}, n^{l=2} = \{4, 2, 2, 1, 1\}, n^{l=3} = \{2, 2, 2, 2, 2\}$, it represents the first layer has 5 head with scale size of 1, 2 head with scale size of 3, 2 head with scale size of $\frac{N}{16}$ and 1 head with scale size of $\frac{N}{8}$.

Experiments

We evaluate our model on 17 text classification datasets, 3 sequence labeling datasets and 1 natural language inference dataset. All the statistics can be found in Tab-1. Besides, we use GloVe (Pennington, Socher, and Manning 2014) to initialize the word embedding and JMT (Hashimoto et al. 2017) for character-level features. The optimizer is Adam (Kingma and Ba 2014) and the learning rate and dropout ratio are listed in the Appendix.

To focus on the comparison between different model designs, we don't list results of BERT-like models because the data augmentation and pre-training is an orthogonal direction.

Text Classification

Text Classification experiments are conducted on Stanford Sentiment Treebank(SST) dataset (Socher et al. 2013) and MTL-16 (Liu, Qiu, and Huang 2017) consists of 16 small datasets in different domains. Besides the base model we introduced before, we use a two-layer MLP(Multi-Layer Per-

Table 1: An overall of datasets and its hyper-parameters, "H DIM, α, head DIM" indicates the dimension of hidden states, the hyper-parameter for controlling the scale distribution, the dimension of each head, respectively. The candidate scales are $1, 3, \frac{N}{16}, \frac{N}{8}, \frac{N}{4}$ for SST,MTL-16,SNLI datasets. And we use $1, 3, 5, 7, 9$ for sequence labeling tasks. MTL-16† consists of 16 datasets, each of them has 1400/200/400 samples in train/dev/test.

Dataset		Train	Dev.	Test	$\|V\|$	H DIM	α	head DIM
SST (Socher et al. 2013)		8k	1k	2k	20k	300	0.5	30
MTL-16 † (Liu, Qiu, and Huang 2017)	Apparel Baby Books Camera DVD Electronics Health IMDB Kitchen Magazines MR Music Software Sports Toys Video	1400	200	400	8k$\sim$28k	300	0.5	30
PTB POS (Marcus, Santorini, and Marcinkiewicz 1993)		38k	5k	5k	44k	300	1.0	30
CoNLL03 (Sang and Meulder 2003)		15k	3k	3k	25k	300	1.0	30
CoNLL2012 NER (Pradhan et al. 2012)		94k	14k	8k	63k	300	1.0	30
SNLI (Bowman et al. 2015)		550k	10k	10k	34k	600	0.5	64

Table 2: Test Accuracy on SST dataset.

Model	Acc
BiLSTM (Li et al. 2015)	49.8
Tree-LSTM (Tai, Socher, and Manning 2015)	51.0
CNN-Tensor (Lei, Barzilay, and Jaakkola 2015)	51.2
Emb + self-att (Shen et al. 2018a)	48.9
BiLSTM + self-att (Yoon, Lee, and Lee 2018)	50.4
CNN + self-att (Yoon, Lee, and Lee 2018)	50.6
Dynamic self-att (Yoon, Lee, and Lee 2018)	50.6
DiSAN (Shen et al. 2018a)	51.7
Transformer	50.4
Multi-Scale Transformer	**51.9**

ceptron) with softmax function as the classifier. It receives the feature from applying max-pooling over the top layer plus the classification node.

Tab-2 and 3 give the results on SST and MTL-16. Multi-Scale Transformer achieves 1.5 and 3.56 points against Transformer on these two datasets, respectively. Meanwhile, Multi-Scale Transformer also beats many existing models including CNNs and RNNs.

Since the sentence average length of MTL-16 dataset is relatively large, we also report the efficiency result in Fig-4. We implement the MS-Trans with Pytorch[1] and DGL(Wang et al. 2019). Multi-Scale Transformer achieves 6.5 times acceleration against Transformer on MTL-16 dataset on average (average sentence length equals 109 tokens). The maximum of acceleration reaches 10 times (average 201 tokens) and Multi-Scale Transformer can achieve 1.8 times acceleration on very short sentences (average 22 tokens).

Sequence Labelling

Besides tasks which use the model as a sentence encoder, we are also interested in the effectiveness of our model on sequence labeling tasks. We choose the Part-of-Speech (POS) tagging and the Named Entity Recognition (NER) task to verify our model. We use three datasets as our benchmark: Penn Treebank (PTB) POS tagging dataset (Marcus, Santorini, and Marcinkiewicz 1993), CoNLL2003 NER dataset (Sang and Meulder 2003), CoNLL2012 NER dataset (Pradhan et al. 2012).

Results in Tab-4 shows Multi-Scale Transformer beats the vanilla Transformer on these three sequence labeling datasets, which consists of other results reported above. It shows Multi-Scale Transformer can extract useful features for each position as well.

[1]https://pytorch.org

Table 3: Test Accuracy over MTL-16 datasets. "SLSTM" refer to the sentence-state LSTM (Zhang, Liu, and Song 2018).

Dataset	Acc (%)			
	MS-Trans	Transformer	BiLSTM	SLSTM
Apparel	87.25	82.25	86.05	85.75
Baby	85.50	84.50	84.51	86.25
Books	85.25	81.50	82.12	83.44
Camera	89.00	86.00	87.05	90.02
DVD	86.25	77.75	83.71	85.52
Electronics	86.50	81.50	82.51	83.25
Health	87.50	83.50	85.52	86.50
IMDB	84.25	82.50	86.02	87.15
Kitchen	85.50	83.00	82.22	84.54
Magazines	91.50	89.50	92.52	93.75
MR	79.25	77.25	75.73	76.20
Music	82.00	79.00	78.74	82.04
Software	88.50	85.25	86.73	87.75
Sports	85.75	84.75	84.04	85.75
Toys	87.50	82.00	85.72	85.25
Video	90.00	84.25	84.73	86.75
Average	**86.34**	82.78	84.01	85.38

Table 4: Results on sequence labeling tasks. We list "Advanced Techniques" except pre-trained embeddings (GloVe, Word2Vec, JMT) in columns. The "Char" indicates character-level features, it also includes the Capitalization Features, Lexicon Features, etc. The "CRF" means an additional Conditional Random Field layer.

Model	Adv Tech		POS	NER	
			PTB	CoNLL2003	CoNLL2012
	char	CRF	Acc	F1	F1
Ling et al. (2015)	✓	✓	97.78	-	-
Huang, Xu, and Yu (2015)	✓	✓	97.55	90.10	-
Chiu and Nichols (2016a)	✓	✓	-	90.69	86.35
Ma and Hovy (2016)	✓	✓	97.55	91.06	-
Chiu and Nichols (2016b)	✓	✓	-	91.62	86.28
Zhang, Liu, and Song (2018)	✓	✓	97.55	91.57	-
Akhundov, Trautmann, and Groh (2018)	✓	✓	97.43	91.11	87.84
Transformer			96.31	86.48	83.57
Transformer + Char	✓		97.04	88.26	85.14
Multi-Scale Transformer			97.24	89.38	85.26
Multi-Scale Transformer + Char	✓		97.54	91.33	86.77
Multi-Scale Transformer + Char + CRF	✓	✓	**97.66**	**91.59**	**87.80**

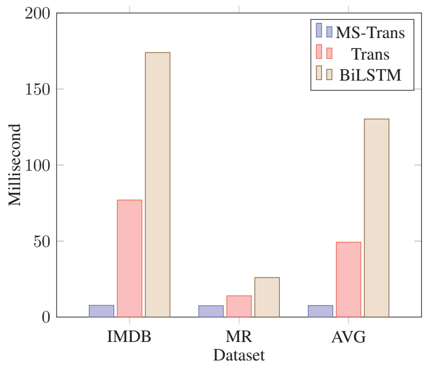


Figure 4: Test time per batch (batch size is 128) on the dataset which has the longest length (IMDB), the dataset which has the shortest length (MR), and the average over 16 datasets in MTL-16.

Table 5: Test Accuracy on SNLI dataset for sentence vector-based models.

Model	Acc
BiLSTM (Liu et al. 2016)	83.3
BiLSTM + self-att (Liu et al. 2016)	84.2
4096D BiLSTM-max (Conneau et al. 2017)	84.5
300D DiSAN (Shen et al. 2018a)	85.6
Residual encoders (Nie and Bansal 2017)	86.0
Gumbel TreeLSTM (Choi, Yoo, and Lee 2018)	86.0
Reinforced self-att (Shen et al. 2018b)	86.3
2400D Multiple DSA (Yoon, Lee, and Lee 2018)	87.4
Transformer	82.2
Multi-Scale Transformer	**85.9**

Natural Language Inference

Natural Language Inference (NLI) is a classification which ask the model to judge the relationship of two sentences from three candidates, "entailment", "contradiction", and "neutral". We use a widely-used benchmark Stanford Natural Language Inference (SNLI) (Bowman et al. 2015) dataset to probe the ability of our model for encoding sentence, we compare our model with sentence vector-based models. Different with the classifier in text classification task, we follow the previous work (Bowman et al. 2016) to use a two-layer MLP classifier which takes concat$(\mathbf{r}_1, \mathbf{r}_2, \|\mathbf{r}_1 - \mathbf{r}_2\|, \mathbf{r}_1 - \mathbf{r}_2)$ as inputs, where $\mathbf{r}_1, \mathbf{r}_2$ are representations of two sentences and equals the feature used in text classification task.

As shown in Tab-5, Multi-Scale Transformer outperforms Transformer and most classical models, and the result is comparable with the state-of-the-art. The reported number of Transformer is obtained with heuristic hyper-parameter selection, we use a three-layer Transformer with heavy dropout and weight decay. And there is still a large margin compared to Multi-Scale Transformer. This comparison also indicates the moderate size training data (SNLI has 550k training samples) cannot replace the usefulness of prior knowledge.

Analysis

Influence of Scale Distributions As we introduced in Eq. (11), we control the scale distributions over layers by a hyper-parameter α. In this section, we give a comparison of using different α, where the positive value means local bias increased with the decline of layer depth and the negative value means global bias increased with the decline of layer depth.

As shown in the upper part of Tab-6, local bias in shallow layers is a key factor to achieve good performance, and an appropriate positive α achieves the best result. In contrast,

Table 6: Analysis of different scale distributions on SNLI test set. The upper part shows the influence of hyper-parameter α which change the distribution of scales across layers. The five candidates of scale size are $1, 3, \frac{N}{16}, \frac{N}{8}, \frac{N}{4}$, respectively. The lower part lists the performance of single-scale models which use a fixed scale for the whole model.

multi-scale	α	N'	L	Acc
A	1.0	5	3	85.5
B	0.5	5	3	**85.9**
C	0.0	5	3	84.9
D	-0.5	5	3	84.7
E	-1.0	5	3	84.3
single-scale	ω		L	Acc
F	3		3	84.3
G	$N/16$		3	83.9
H	$N/8$		3	81.7
I	$N/4$		3	80.7

all the negative values harm the performance, that means too much global bias in shallow layers may lead the model to a wrong direction. The observation of this experiment fits our intuition, the high-level feature is the composition of low-level terms.

Multi-Scale vs. Single-Scale As we claimed before, Multi-Scale Transformer can capture knowledge at different scales at each layer. Therefore, a simple question needs to be evaluated is whether the multi-scale model outperforms the single-scale model or not. To answer this question, we compare Multi-Scale Transformer with several single-scale models. Model F,G,H,I have the same number of layers and attention heads with Multi-scale Transformer, but their scales are fixed.

Result in the lower part of Tab-6 reveal the value of Multi-Scale Transformer, it achieves 1.6 points improvement against the best single-scale model. And this result also supports that local bias is an important inductive bias for NLP task.

Related Works

Typical multi-scale models The multi-scale structure has been used in many NLP models, it could be implemented in many different ways. Such as stacked layers (Kalchbrenner, Grefenstette, and Blunsom 2014; Kim 2014), tree-structure (Socher et al. 2013; Tai, Socher, and Manning 2015; Zhu, Sobhani, and Guo 2015), hierarchical timescale (El Hihi and Bengio 1995; Chung, Ahn, and Bengio 2016), layer-wise gating (Chung et al. 2015). Since these models are built on modules like RNNs and CNNs, which embodies the intrinsic local bias by design, the common spirit of introducing multi-scale is to enable long-range communications. In contrast, Transformer allows long-range communications, so we want the multi-scale brings local bias.

Transformers with additional inductive bias This work is not the first attempt of introducing inductive bias into Transformer.

Shaw, Uszkoreit, and Vaswani (2018) suggest Transformer should care about the relative distance between tokens rather than the absolute position in the sequence. The information of relative distance could be obtained by looking multi-scale of the same position, so our model could be aware of the relative distance if using enough scales.

Li et al. (2018) propose a regularization of enhancing the diversity of attention heads. Our multi-scale multi-head self-attention can make a good division of labor of heads via restricting them in different scales.

Yang et al. (2018a) and Yang et al. (2018b) also introduce the local bias into Transformer.

Different from the above models, we focus on importing the notion of multi-scale to self-attention. Meanwhile, their models use the single-scale structure. Our experimental results have shown the effectiveness of the multi-scale mechanism.

Conclusion

In this work, we present Multi-Scale Self-Attention and Multi-Scale Transformer which combines the prior knowledge of multi-scale and the self-attention mechanism. As a result, it has the ability to extract rich and robust features from different scales. We compare our model with the vanilla Transformer on three real tasks (21 datasets). The result suggests our proposal outperforms the vanilla Transformer consistently and achieves comparable results with state-of-the-art models.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (No. 2018YFC0831103), National Natural Science Foundation of China (No. 61672162), Shanghai Municipal Science and Technology Major Project (No. 2018SHZDZX01) and ZJLab.

References

Akhundov, A.; Trautmann, D.; and Groh, G. 2018. Sequence labeling: A practical approach. *CoRR* abs/1808.03926.

Ba, L. J.; Kiros, R.; and Hinton, G. E. 2016. Layer normalization. *CoRR* abs/1607.06450.

Bowman, S. R.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, 632–642. The Association for Computational Linguistics.

Bowman, S. R.; Gauthier, J.; Rastogi, A.; Gupta, R.; Manning, C. D.; and Potts, C. 2016. A fast unified model for parsing and sentence understanding. In *ACL (1)*. The Association for Computer Linguistics.

Chiu, J., and Nichols, E. 2016a. Sequential labeling with bidirectional lstm-cnns. In *Proc. International Conf. of Japanese Association for NLP*, 937–940.

Chiu, J. P. C., and Nichols, E. 2016b. Named entity recognition with bidirectional lstm-cnns. *TACL* 4:357–370.

Choi, J.; Yoo, K. M.; and Lee, S. 2018. Learning to compose task-specific tree structures. In *AAAI*, 5094–5101. AAAI Press.

Chung, J.; Ahn, S.; and Bengio, Y. 2016. Hierarchical multiscale recurrent neural networks. *arXiv preprint arXiv:1609.01704*.

Chung, J.; Gülçehre, Ç.; Cho, K.; and Bengio, Y. 2015. Gated feedback recurrent neural networks. In *ICML*, volume 37 of *JMLR Workshop and Conference Proceedings*, 2067–2075. JMLR.org.

Conneau, A.; Kiela, D.; Schwenk, H.; Barrault, L.; and Bordes, A. 2017. Supervised learning of universal sentence representations from natural language inference data. In *EMNLP*, 670–680. Association for Computational Linguistics.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J. G.; Le, Q. V.; and Salakhutdinov, R. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *CoRR* abs/1901.02860.

Devlin, J.; Chang, M.; Lee, K.; and Toutanova, K. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR* abs/1810.04805.

El Hihi, S., and Bengio, Y. 1995. Hierarchical recurrent neural networks for long-term dependencies. In *NIPS*, 493–499. Citeseer.

Guo, Q.; Qiu, X.; Liu, P.; Shao, Y.; Xue, X.; and Zhang, Z. 2019. Star-transformer. In *NAACL-HLT (1)*, 1315–1325. Association for Computational Linguistics.

Hashimoto, K.; Xiong, C.; Tsuruoka, Y.; and Socher, R. 2017. A joint many-task model: Growing a neural network for multiple NLP tasks. In *EMNLP*, 1923–1933. Association for Computational Linguistics.

Huang, Z.; Xu, W.; and Yu, K. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR* abs/1508.01991.

Kalchbrenner, N.; Grefenstette, E.; and Blunsom, P. 2014. A convolutional neural network for modelling sentences. In *Proceedings of ACL*.

Kim, Y. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on EMNLP*, 1746–1751.

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

Lei, T.; Barzilay, R.; and Jaakkola, T. S. 2015. Molding cnns for text: non-linear, non-consecutive convolutions. In *EMNLP*, 1565–1575. The Association for Computational Linguistics.

Li, J.; Luong, T.; Jurafsky, D.; and Hovy, E. H. 2015. When are tree structures necessary for deep learning of representations? In *EMNLP*, 2304–2314. The Association for Computational Linguistics.

Li, J.; Tu, Z.; Yang, B.; Lyu, M. R.; and Zhang, T. 2018. Multi-head attention with disagreement regularization. In *EMNLP*, 2897–2903. Association for Computational Linguistics.

Ling, W.; Dyer, C.; Black, A. W.; Trancoso, I.; Fermandez, R.; Amir, S.; Marujo, L.; and Luís, T. 2015. Finding function in form: Compositional character models for open vocabulary word representation. In *EMNLP*, 1520–1530. The Association for Computational Linguistics.

Liu, Y.; Sun, C.; Lin, L.; and Wang, X. 2016. Learning natural language inference using bidirectional LSTM model and inner-attention. *CoRR* abs/1605.09090.

Liu, P.; Qiu, X.; and Huang, X. 2017. Adversarial multi-task learning for text classification. In *ACL (1)*, 1–10. Association for Computational Linguistics.

Ma, X., and Hovy, E. H. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *ACL (1)*. The Association for Computer Linguistics.

Marcus, M. P.; Santorini, B.; and Marcinkiewicz, M. A. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics* 19(2):313–330.

Nie, Y., and Bansal, M. 2017. Shortcut-stacked sentence encoders for multi-domain inference. In *RepEval@EMNLP*, 41–45. Association for Computational Linguistics.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Pradhan, S.; Moschitti, A.; Xue, N.; Uryupina, O.; and Zhang, Y. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *EMNLP-CoNLL Shared Task*, 1–40. ACL.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training.

Sang, E. F. T. K., and Meulder, F. D. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *CoNLL*, 142–147. ACL.

Shaw, P.; Uszkoreit, J.; and Vaswani, A. 2018. Self-attention with relative position representations. In *NAACL-HLT (2)*, 464–468. Association for Computational Linguistics.

Shen, T.; Zhou, T.; Long, G.; Jiang, J.; Pan, S.; and Zhang, C. 2018a. Disan: Directional self-attention network for rnn/cnn-free language understanding. In *AAAI*, 5446–5455. AAAI Press.

Shen, T.; Zhou, T.; Long, G.; Jiang, J.; Wang, S.; and Zhang, C. 2018b. Reinforced self-attention network: a hybrid of hard and soft attention for sequence modeling. In *IJCAI*, 4345–4352. ijcai.org.

Socher, R.; Perelygin, A.; Wu, J.; Chuang, J.; Manning, C. D.; Ng, A. Y.; and Potts, C. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, 1631–1642. ACL.

Tai, K. S.; Socher, R.; and Manning, C. D. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *ACL (1)*, 1556–1566. The Association for Computer Linguistics.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *NIPS*, 6000–6010.

Wang, M.; Yu, L.; Zheng, D.; Gan, Q.; Gai, Y.; Ye, Z.; Li, M.; Zhou, J.; Huang, Q.; Ma, C.; Huang, Z.; Guo, Q.; Zhang, H.; Lin, H.; Zhao, J.; Li, J.; Smola, A. J.; and Zhang, Z. 2019. Deep graph library: Towards efficient and scalable deep learning on graphs. *CoRR* abs/1909.01315.

Yang, Z.; Yang, D.; Dyer, C.; He, X.; Smola, A.; and Hovy, E. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of NAACL*, 1480–1489.

Yang, B.; Tu, Z.; Wong, D. F.; Meng, F.; Chao, L. S.; and Zhang, T. 2018a. Modeling localness for self-attention networks. In *EMNLP*, 4449–4458. Association for Computational Linguistics.

Yang, B.; Wang, L.; Wong, D. F.; Chao, L. S.; and Tu, Z. 2018b. Convolutional self-attention network. *CoRR* abs/1810.13320.

Yoon, D.; Lee, D.; and Lee, S. 2018. Dynamic self-attention : Computing attention over words dynamically for sentence embedding. *CoRR* abs/1808.07383.

Zhang, Y.; Liu, Q.; and Song, L. 2018. Sentence-state LSTM for text representation. In *ACL (1)*, 317–327. Association for Computational Linguistics.

Zhu, X.-D.; Sobhani, P.; and Guo, H. 2015. Long short-term memory over recursive structures. In *ICML*, 1604–1612.