

DA-Transformer: Distance-aware Transformer

Chuhan Wu[†] Fangzhao Wu[‡] Yongfeng Huang[†]

[†]Department of Electronic Engineering & BNRist, Tsinghua University, Beijing 100084, China

[‡]Microsoft Research Asia, Beijing 100080, China

{wuchuhan15, wufangzhao}@gmail.com

yfhuang@tsinghua.edu.cn

Abstract

Transformer has achieved great success in the NLP field by composing various advanced models like BERT and GPT. However, Transformer and its existing variants may not be optimal in capturing token distances because the position or distance embeddings used by these methods usually cannot keep the precise information of real distances, which may not be beneficial for modeling the orders and relations of contexts. In this paper, we propose DA-Transformer, which is a distance-aware Transformer that can exploit the real distance. We propose to incorporate the real distances between tokens to re-scale the raw self-attention weights, which are computed by the relevance between attention query and key. Concretely, in different self-attention heads the relative distance between each pair of tokens is weighted by different learnable parameters, which control the different preferences on long- or short-term information of these heads. Since the raw weighted real distances may not be optimal for adjusting self-attention weights, we propose a learnable sigmoid function to map them into re-scaled coefficients that have proper ranges. We first clip the raw self-attention weights via the ReLU function to keep non-negativity and introduce sparsity, and then multiply them with the re-scaled coefficients to encode real distance information into self-attention. Extensive experiments on five benchmark datasets show that DA-Transformer can effectively improve the performance of many tasks and outperform the vanilla Transformer and its several variants.

1 Introduction

Transformer has achieved huge success in the NLP field in recent years (Vaswani et al., 2017). It serves as the basic architecture of various state-of-the-art models like BERT (Devlin et al., 2019) and GPT (Radford et al., 2019), and boosts

the performance of many tasks like text generation (Koncel-Kedziorski et al., 2019), machine translation (Vaswani et al., 2017), and reading comprehension (Xu et al., 2019). Thus, the improvement on the Transformer architecture would be beneficial for many NLP-related fields.

A core component of Transformer is multi-head self-attention, which is responsible for modeling the relations between contexts (Yang et al., 2019; Guo et al., 2019). However, self-attention is position-agnostic since it does not distinguish the orders of inputs. Thus, in the vanilla Transformer, position encoding is applied to the input to help Transformer capture position information. However, in contrast to recurrent and convolutional neural networks, it is difficult for vanilla Transformers to be aware of the token distances (Shaw et al., 2018), which are usually important cues for context modeling. Thus, several works explored to incorporate token distance information into Transformer. For example, Shaw et al. (2018) proposed to combine the embeddings of relative positions with attention key and value in the self-attention network. They restricted the maximum relative distance to only keep the precise relative position information within a certain distance. Yan et al. (2019) proposed a variant of self-attention network for named entity recognition, which incorporates sinusoidal embeddings of relative position to compute attention weights in a direction- and distance-aware way. However, the distance or relative position embeddings used by these methods usually cannot keep the precise information of the real distance, which may not be beneficial for the Transformer to capture word orders and the context relations.

In this paper, we propose a distance-aware Transformer (DA-Transformer), which can explicitly exploit real token distance information to enhance context modeling by leveraging the relative distances between different tokens to re-scale the raw

attention weights before softmax normalization. More specifically, since global and local context modeling usually has different distance preferences, we propose to learn a different parameter in different attention heads to weight the token distances, which control the preferences of attention heads on long or short distances. In addition, since the weighted distances may not have been restricted to a proper range, we propose a learnable sigmoid function to map the weighted distances into re-scaled coefficients. They are further multiplied with the raw attention weights that are clipped by the ReLU function for keeping the non-negativity and introducing sparsity. We conduct extensive experiments on five benchmark datasets for different tasks, and the results demonstrate that our approach can effectively enhance the performance of Transformer and outperform its several variants with distance modeling.

The main contributions of this paper include:

- We propose a distance-aware Transformer that uses the real token distances to keep the precise distance information for adjusting the attention weights.
- We propose to use different parameters to weight real distances in different attention heads to control their preferences on short-term or long-term information.
- We propose a learnable sigmoid function to map the weighted distances into re-scaled coefficients with proper ranges for better adjusting the attention weights.
- We conduct extensive experiments on five benchmark datasets and the results validate the effectiveness of our proposed method.

2 Related Work

2.1 Transformer

To make this paper self-contained, we first briefly introduce the architecture of Transformer, which was initially introduced to the machine translation task (Vaswani et al., 2017). It has become an important basic neural architecture of various state-of-the-art NLP models like BERT (Devlin et al., 2019) and GPT (Radford et al., 2019). The core component of Transformer is multi-head self-attention. It has h attention heads, where the parameters in each head are independent. For the i -th attention head,

it takes a matrix \mathbf{H} as the input. It first uses three independent parameter matrices $\mathbf{W}_Q^{(i)}$, $\mathbf{W}_K^{(i)}$, and $\mathbf{W}_V^{(i)}$ to respectively transform the input matrix H into the input query $\mathbf{Q}^{(i)}$, key $\mathbf{K}^{(i)}$ and value $\mathbf{V}^{(i)}$, which is formulated as follows:

$$\mathbf{Q}^{(i)}, \mathbf{K}^{(i)}, \mathbf{V}^{(i)} = \mathbf{H}\mathbf{W}_Q^{(i)}, \mathbf{H}\mathbf{W}_K^{(i)}, \mathbf{H}\mathbf{W}_V^{(i)}. \quad (1)$$

Then, it uses a scaled dot-product attention head to process its query, key and value, which is formulated as follows:

$$\text{Attention}(\mathbf{Q}^{(i)}, \mathbf{K}^{(i)}, \mathbf{V}^{(i)}) = \text{softmax}\left(\frac{\mathbf{Q}^{(i)}\mathbf{K}^{(i)\top}}{\sqrt{d}}\right)\mathbf{V}^{(i)}, \quad (2)$$

where d is the dimension of the vectors in the query and key. The outputs of the h attention heads are concatenated together and the final output is a linear projection of the concatenated representations, which is formulated as follows:

$$\text{Multihead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)\mathbf{W}_O, \quad (3)$$

where $\text{head}_i = \text{Attention}(\mathbf{Q}^{(i)}, \mathbf{K}^{(i)}, \mathbf{V}^{(i)})$,

where \mathbf{W}_O is an output projection matrix. In the standard Transformer, a position-wise feed-forward neural network is further applied to the output of multi-head self-attention network. Its function is formulated as follows:

$$\text{FFN}(\mathbf{x}) = \max(0, \mathbf{x}\mathbf{W}_1 + \mathbf{b}_1)\mathbf{W}_2 + \mathbf{b}_2, \quad (4)$$

where \mathbf{W}_1 , \mathbf{W}_2 , \mathbf{b}_1 , \mathbf{b}_2 are kernel and bias parameters. Transformer also employs layer normalization and residual connection techniques after the multi-head self-attention and feed-forward neural networks, which are also kept in our method.

Since self-attention network does not distinguish the order and position of input tokens, Transformer adds the sinusoidal embeddings of positions to the input embeddings to capture position information. However, position embeddings may not be optimal for distance modeling in Transformer because distances cannot be precisely recovered from the dot-product between two position embeddings.

2.2 Distance-aware Transformer

Instead of directly using the sinusoidal position embedding (Vaswani et al., 2017) or the absolute position embedding (Devlin et al., 2019), several variants of the Transformer explore to use the relative positions to better model the distance between contexts (Shaw et al., 2018; Wang et al., 2019; Dai et al., 2019; Yan et al., 2019). For example, Shaw et al. (2018) proposed to add the embeddings of

relative positions to the attention key and value to capture the relative distance between two tokens. They only kept the precise distance within a certain range by using a threshold to clip the maximum distance to help generalize to long sequences. Dai et al. (2019) proposed Transformer-XL, which uses another form of relative positional encodings that integrate content-dependent positional scores and a global positional score into the attention weights. Yan et al. (2019) proposed direction-aware sinusoidal relative position embeddings and used them in a similar way with Transformer-XL. In addition, they proposed to use the un-scaled attention to better fit the NER task. However, relative position embeddings may not be optimal for modeling distance information because they usually cannot keep the precise information of real token distances. Different from these methods, we propose to directly re-scale the attention weights based on the mapped relative distances instead of using sinusoidal position embeddings, which can explicitly encode real distance information to achieve more accurate distance modeling.

3 DA-Transformer

In this section, we introduce our proposed distance-aware Transformer (DA-Transformer) approach, which can effectively exploit read token distance information to enhance context modeling. It uses a learnable parameter to weight the real distances between tokens in each attention head, and uses a learnable sigmoid function to map the weighted distances into re-scaled coefficients with proper ranges, which are further used to adjust the raw attention weights before softmax normalization. The details of DA-Transformer are introduced in the following sections.

3.1 Head-wise Distance Weighting

Similar with the standard Transformer, the input of our model is also a matrix that contains the representation of each token, which is denoted as $\mathbf{H} = [\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_N]$, where N is the length of the sequence. We denote the relative distance between the i -th and j -th positions as $R_{i,j}$, which is computed by $R_{i,j} = |i - j|$. We can then obtain the relative distance matrix $\mathbf{R} \in \mathbb{R}^{N \times N}$ that describes the relative distance between each pair of positions. In each attention head, we use a learnable parameter w_i to weight the relative distance by $\mathbf{R}^{(i)} = w_i \mathbf{R}$, which will be further used to

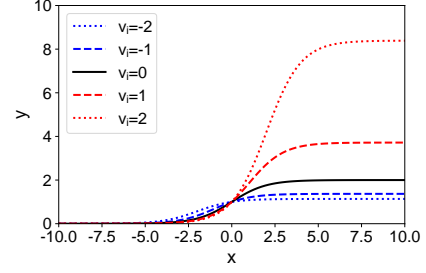


Figure 1: The curves of our learnable sigmoid function under different v_i .

adjust the self-attention weights. In our method, we stipulate that a more positive $\mathbf{R}^{(i)}$ will amplify the attention weights more strongly while a more negative $\mathbf{R}^{(i)}$ will diminish them more intensively. Thus, a positive w_i means that this attention head prefers to capture long-distance information, while a negative w_i means that it focuses more on local contexts. By learning different values of w_i , different attention heads may have different preferences on capturing either short-term or long-term contextual information with different intensity.

3.2 Weighted Distance Mapping

Since the raw weighted distances may not in the proper range for adjusting the attention weights, we need to map them into the re-scaled coefficients via a function $\hat{\mathbf{R}}^{(i)} = f(\mathbf{R}^{(i)})$ that is suitable for adjusting the self-attention weights. However, it is not a trivial task to design the function $f(\cdot)$ because it needs to satisfy the following requirements: (1) $f(0) = 1$. We stipulate that zero distances do not influence the self-attention weights. (2) The value of $f(\mathbf{R}^{(i)})$ should be zero when $\mathbf{R}^{(i)} \rightarrow -\infty$. This requirement is to guarantee that if an attention head prefers to capture local information ($w_i < 0$), the long-distance information should be surpassed.¹ (3) The value of $f(\mathbf{R}^{(i)})$ should be limited when $\mathbf{R}^{(i)} \rightarrow +\infty$. This requirement is to ensure that the model is able to process long sequences without over-emphasize distant contexts. (4) The scale of $f(\cdot)$ needs to be tunable. This aims to help the model better adjust the intensity of distance information. (5) The function $f(\cdot)$ needs to be monotone. To satisfy the four requirements above, we propose a learnable sigmoid function to map the weighted relative distances $\mathbf{R}^{(i)}$, which is formu-

¹Although the raw negative attention weights may be raised to 0 by $f(\cdot)$, the model can still surpass these attention weights after softmax by increasing the scale of other attention weights.

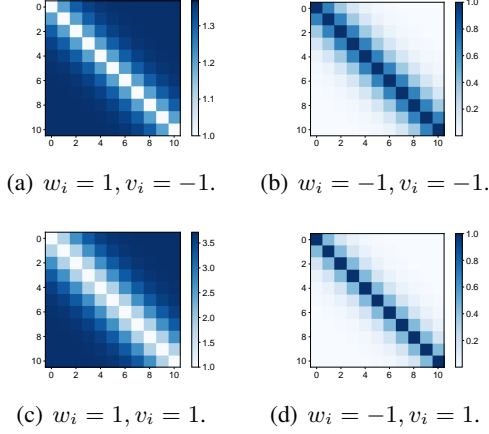


Figure 2: The re-scaled coefficient matrices under different values of w_i and v_i .

lated as follows:

$$f(\mathbf{R}^{(i)}; v_i) = \frac{1 + \exp(v_i)}{1 + \exp(v_i - \mathbf{R}^{(i)})}, \quad (5)$$

where v_i is a learnable parameter in this head that controls the upperbound and ascending steepness of this function. The curves of our learnable sigmoid function under several different values of v_i are plotted in Fig. 1. We can see that the proposed function satisfies all the requirements above. In addition, from this figure we find that if v_i is larger, the upperbound of the curve is higher, which means that distance information is more intensive. When $v_i = 0$, it is in fact identical to the standard sigmoid function except for the scaling factor of 2. By mapping the weighted distances $\mathbf{R}^{(i)}$ via the function $f(\cdot)$, we can obtain the final re-scaled coefficients $\hat{\mathbf{R}}^{(i)}$ in a learnable way. Several illustrative examples of the re-scaled coefficients under $w_i = \pm 1$ and $v_i = \pm 1$ are respectively shown in Figs. 2(a)-2(d). We can see that if w_i is positive, long-distance contexts are preferred while short-term contexts are surpassed. The situation is reversed if w_i turns to negative. In addition, the coefficients in Fig. 2(c) have larger dynamic ranges than the coefficients in Fig. 2(a), indicating that long-distance information is more dominant in Fig. 2(c). Moreover, the coefficients in Fig. 2(d) are “sharper” than those in Fig. 2(b), which indicates that the model tends to capture shorter distances.

3.3 Attention Adjustment

Then, we use the re-scaled coefficients to adjust the raw attention weights that are computed by the dot-product between the query and key, i.e., $\frac{\mathbf{Q}^{(i)}\mathbf{K}^{(i)\top}}{\sqrt{d}}$.

Different from existing methods that add the query-key dot-product with position or distance representations, in our approach we propose to multiply the re-scaled coefficients with the query-key dot-product. This is because for the tokens whose relations are very weak, if their re-scaled coefficients are large, their final attention weights will be over-amplified if we simply add the re-scaled coefficients to their raw attention weights. This is not optimal for modeling contextual information because the attention weights of irrelevant contexts cannot be fully surpassed. However, there are also some problems if we directly multiply the re-scaled coefficients $\hat{\mathbf{R}}^{(i)}$ and the raw attention weights $\frac{\mathbf{Q}^{(i)}\mathbf{K}^{(i)\top}}{\sqrt{d}}$. This is because the sign of attention weights $\frac{\mathbf{Q}^{(i)}\mathbf{K}^{(i)\top}}{\sqrt{d}}$ is indefinite and the multiplied results cannot accurately reflect the influence of distance information. Thus, we propose to add a ReLU (Glorot et al., 2011) activation function to the raw attention weights to keep non-negativity. In this way, the final output $\mathbf{O}^{(i)}$ of an attention head can be formulated as follows:

$$\mathbf{O}^{(i)} = \text{softmax}\left(\frac{\text{ReLU}(\mathbf{Q}^{(i)}\mathbf{K}^{(i)\top}) * \hat{\mathbf{R}}^{(i)}}{\sqrt{d}}\right)\mathbf{V}^{(i)}, \quad (6)$$

where $*$ represents element-wise product. The ReLU function can also introduce sparsity to the self-attention because only the positive attention weights can be amplified by the re-scaled coefficients, which makes the attention weights in our method sharper. We concatenate the output from the h independent attention heads, and project it into a unified output. In addition, we keep the same layer normalization and residual connection strategy as the standard Transformer.

3.4 Discussions on Computational Complexity

Compared with the standard Transformer, the major additional time cost is brought by computing the re-scaled coefficients $\hat{\mathbf{R}}^{(i)}$ and using them to adjust the attention weights. The time complexity of the two operations in each head is $O(N^2)$. Compared with the time complexity of standard Transformer, i.e., $O(N^2 \times d)$, the additional time consumption of our method is very light. The increase of parameters is also minimal because we only introduce $2h$ additional parameters, which are usually ignorable compared with the projection matrices like $\mathbf{W}_Q^{(i)}$. Thus, our approach inherits the efficiency of the Transformer architecture.

4 Experiments

4.1 Datasets and Experimental Settings

Our experiments are conducted on five benchmark datasets for different tasks. Four of them are benchmark NLP datasets. The first one is AG’s News² (denoted as *AG*), which is a news topic classification dataset. The second one is Amazon Electronics (He and McAuley, 2016) (denoted as *Amazon*), which is a dataset for review rating prediction. The third one is Stanford Sentiment Treebank (Socher et al., 2013) (denoted as *SST*). We use the binary classification version of this dataset. The fourth one is Stanford Natural Language Inference (Bowman et al., 2015) (*SNLI*) dataset, which is a widely used natural language inference dataset. The detailed statistics of these datasets are summarized in Table 1. In addition, we also conduct experiments on a benchmark news recommendation dataset named *MIND* (Wu et al., 2020), aiming to validate the effectiveness of our approach in both text and user modeling. It contains the news impression logs of 1 million users from Microsoft News³ from October 12 to November 22, 2019. The training set contains the logs in the first five weeks except those on the last day which are used for validation. The rest logs are used for test. The key statistics of this dataset are summarized in Table 2.

Dataset	# Train	# Dev.	# Test	# Classes
AG	108k	12k	7.6k	4
Amazon	40k	5k	5k	5
SST	8k	1k	2k	2
SNLI	55k	10k	10k	2

Table 1: Statistics of *AG*, *Amazon*, *SST* and *SNLI* datasets.

# Users	1,000,000	Avg. title len.	11.52
# News	161,013	# Click samples	5,597,979
# Impressions	500,000	# Non-click samples	136,162,621

Table 2: Statistics of the *MIND* dataset.

In our experiments, we use the 300-dimensional Glove (Pennington et al., 2014) embeddings for word embedding initialization.⁴ The number of attention head is 16, and the output dimension of each attention is 16. On the *AG*, *SST* and *SNLI* datasets,

²<https://www.di.unipi.it/en/>

³<https://www.msn.com/en-us>

⁴We do not use contextualized embeddings generated by language models like BERT because we mainly focus on validating the effectiveness of our Transformer architecture.

we directly apply Transformer-based methods to the sentences. On the *Amazon* dataset, since reviews are usually long documents, we use Transformers in a hierarchical way by learning sentence representations from words via a word-level Transformer first and then learning document representations from sentences via a sentence-level Transformer. On the *MIND* dataset, following (Wu et al., 2019) we also use a hierarchical model architecture that first learns representations of historical clicked news and candidate news from their titles with a word-level Transformer, then learns user representations from the representations of clicked news with a news-level Transformer, and final matches user and candidate news representations to compute click scores. We use Adam (Kingma and Ba, 2015) as the optimization algorithm. Each experiment is repeated 5 times independently. On the *AG*, *Amazon*, *SST* and *SNLI* datasets, accuracy and macro-Fscore are used as the performance metric. On the *MIND* dataset, following (Wu et al., 2019) we use the average AUC, MRR, nDCG@5 and nDCG@10 scores of all sessions as the metrics.

4.2 Performance Evaluation

We compare our proposed *DA-Transformer* method with several baseline methods, including: (1) *Transformer* (Vaswani et al., 2017), the vanilla Transformer architecture, where sinusoidal positional embeddings are used. (2) *Transformer-RPR* (Shaw et al., 2018), a variant of Transformer with relative position representations. (3) *Transformer-XL* (Dai et al., 2019), a variant of Transformer that consists of a segment-level recurrence mechanism and a sinusoidal relative position encoding scheme. (4) *TENER* (Yan et al., 2019), a variant of Transformer that uses direction- and distance-aware position encoding for named entity recognition. The results of our approach and these methods on the five datasets are respectively shown in Tables 3 and 4.⁵ From the results, we have several observations.

First, compared with the vanilla Transformer, the compared methods that consider distance information consistently achieve better performance. It shows that distance information is very important in context modeling. Second, among the methods with distance information, the performance of *Transformer-RPR* is lower than the others. This may be because *Transformer-RPR* does not keep the precise long-distance information. Third, by

⁵Detailed ablation studies are presented in supplements.

Methods	AG		Amazon		SST		SNLI	
	Accuracy	Macro-F	Accuracy	Macro-F	Accuracy	Macro-F	Accuracy	Macro-F
Transformer	93.01	93.00	65.15	42.14	89.67	89.59	81.45	81.42
Transformer-RPR	93.14	93.13	65.29	42.40	89.94	89.90	82.20	82.18
Transformer-XL	<u>93.35</u>	<u>93.34</u>	<u>65.50</u>	<u>42.88</u>	90.06	90.02	<u>83.19</u>	<u>83.15</u>
TENER	93.28	93.27	65.47	42.69	<u>90.15</u>	<u>90.10</u>	82.35	82.31
*DA-Transformer	93.72	93.70	66.38	44.29	90.49	90.43	84.18	84.16

Table 3: Results on the four benchmark NLP datasets. *Improvement over the underlined second best results is significant at $p < 0.05$.

Methods	AUC	MRR	nDCG@5	nDCG@10
Transformer	67.76	33.05	35.94	41.63
Transformer-RPR	67.81	33.10	35.98	41.65
Transformer-XL	<u>67.92</u>	<u>33.15</u>	<u>36.04</u>	<u>41.70</u>
TENER	67.70	33.01	35.89	41.58
DA-Transformer	68.32	33.36	36.34	42.07

Table 4: Results on the *MIND* dataset. *Improvement over the underlined second best results is significant at $p < 0.05$.

comparing *Transformer-XL* and *TENER*, we find that the performance of *TENER* is better on the SST dataset, while *Transformer-XL* is better on other datasets. This is probably because *TENER* is more suitable for modeling local contexts and the sentences in the SST dataset are usually short, while *Transformer-XL* may be more appropriate for modeling long sequences. Fourth, our method consistently achieves better performance on the five datasets, and its improvement over the second best method is statistically significant. This is because our method can explicitly encode real distance information rather than using positional encoding, making the modeling of distance more accurate.

4.3 Influence of Different Mapping Functions

Next, we study the influence of using different mapping functions $f(\cdot)$ for computing the re-scaled coefficients. We compare the performance of our method w.r.t. several different $f(\cdot)$, including: (1) $f(x) = \min(x, T)$ (clip), using a threshold T to clip the weighted distance; (2) $f(x) = k_i x + b_i$ (linear), using a linear transformation to the weighted distance; (3) $f(x) = \exp(x)$ (exponent), using an exponent function to map the weighted distance; (4) $f(x) = \frac{1}{1+\exp(-x)}$ (sigmoid), using the sigmoid function to activate the weighted distance; and (5) $f(x; v_i) = \frac{1+\exp(v_i)}{1+\exp(v_i-x)}$, our learnable sigmoid function. Due to space limitation, we only present the results on the AG, Amazon and MIND datasets in Fig. 3. From these results, we find that clip is not optimal for mapping the weighted dis-

tance. This is because it cannot keep the precise distance information beyond a certain range. In addition, simply using the linear transformation is also insufficient. This may be because our attention adjustment method requires $f(\cdot)$ to be positive, but linear transformation cannot guarantee. Besides, we find that the sigmoid function and our proposed function are better than the exponential function. This may be because long sequences will lead to the problem of exponent explosion, which is harmful to context modeling. Moreover, our proposed learnable sigmoid function is better than the standard sigmoid function. It shows that adjusting the activation function in a learnable way can better map the raw distances into re-scaled coefficients.

4.4 Influence of Different Attention Adjusting Methods

Then, we explore the influence of different methods for adjusting the raw attention weights. We consider four different kinds of methods, including: (1) adding the re-scaled coefficients to the attention weights normalized by softmax (late add); (2) multiplying the re-scaled coefficients with the attention weights normalized by softmax (late multiply); (3) adding the re-scaled coefficients to the raw attention weights before normalization (early add), which is widely used in existing methods like *Transformer-XL*; (4) multiplying the re-scaled coefficients with the raw attention weights activated by ReLU, which is the method used in our approach (early multiply). The results on the AG, Amazon and MIND datasets are shown in Fig. 4. According to these results, we find that early adjustment is better than late adjustment. This may be because the late adjustment methods will change the total amount of attention, which may not be optimal. In addition, we find that multiplying is better than adding for both early and late adjustment. This may be because adding large re-scaled coefficients may over-amplify some attention weights. For ex-

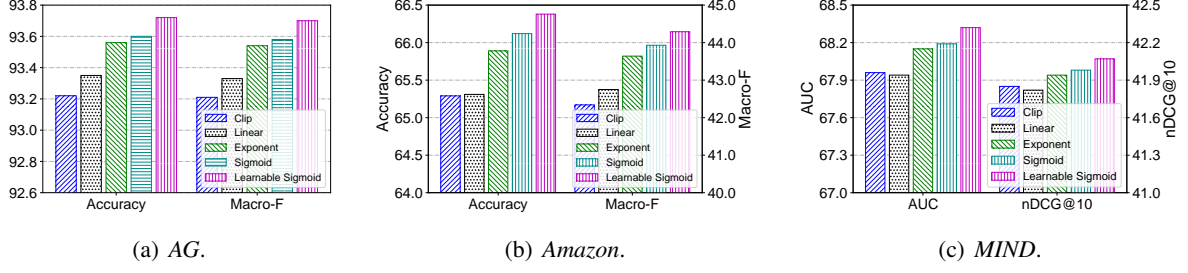


Figure 3: Influence of using different mapping functions.

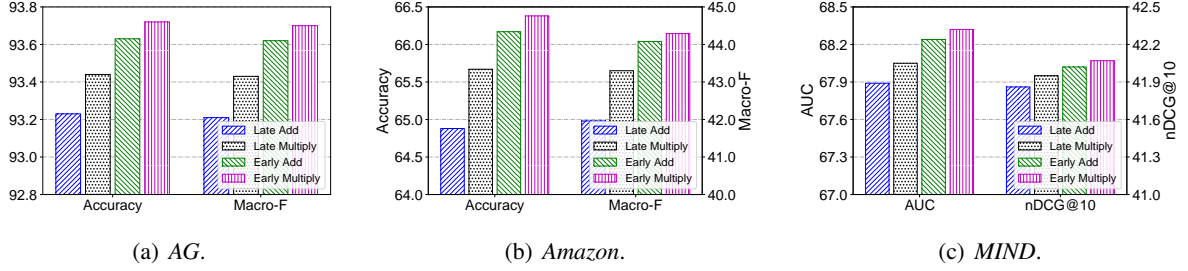
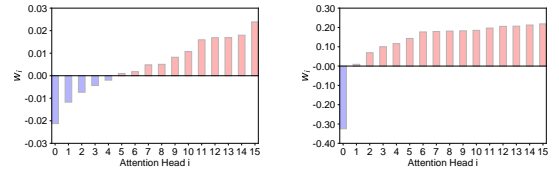
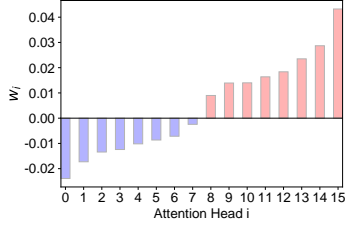


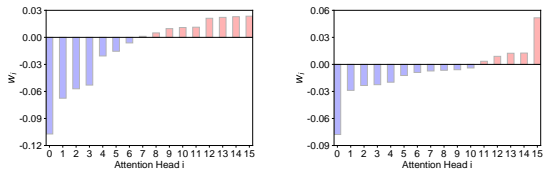
Figure 4: Influence of using different attention adjusting methods.



(a) Word-level Transformer. (b) News-level Transformer.

Figure 5: The distance weights learned by different attention heads on the AG dataset.

Figure 7: The distance weights learned by different attention heads on the MIND dataset.



(a) Word-level Transformer. (b) Sentence-level Transformer.

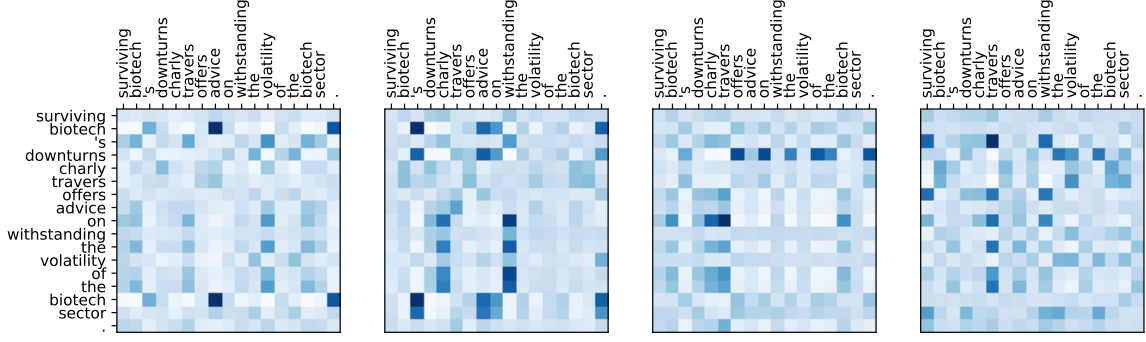
Figure 6: The distance weights learned by different attention heads on the Amazon dataset.

ample, if a raw attention weight is relatively small, it is not suitable to add large re-scaled coefficients to it because the corresponding contexts may not have close relations. In contrast, multiplying the re-scaled coefficients will not over-amplify the low attention weights. Moreover, in our early multiply method we further propose to use the ReLU function to introduce sparsity to make the Trans-

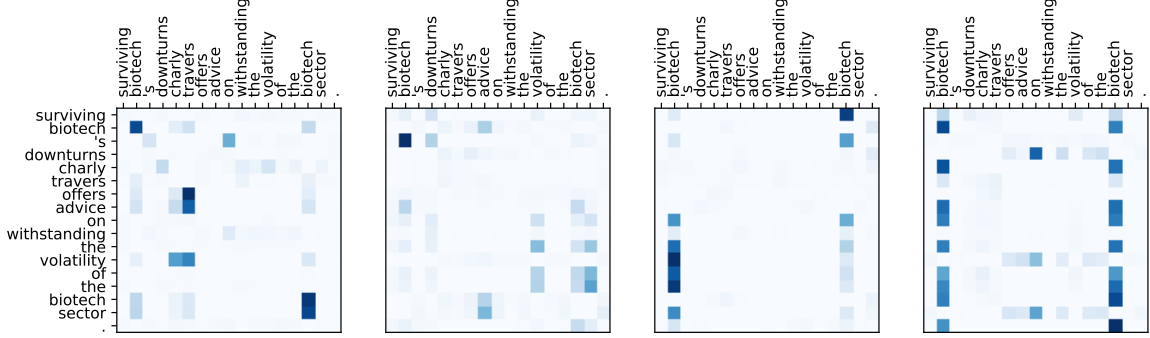
former more “focused”. Thus, our method is better than the existing early add method in adjusting the attention weights.

4.5 Model Interpretation

Finally, we interpret our proposed method by visualizing its key parameters and the attention weights. we first visualize the parameters w_i in our method, which control the preferences of attention heads on long-term or short-term information. The visualization results on the AG, Amazon and MIND datasets are respectively shown in Figs. 5, 6 and 7. From Fig. 5, we find it is very interesting that half of the parameters w_i are positive and the rest of them are negative. It indicates that half of the attention heads mainly aim to capture local contexts, while the rest ones are responsible for modeling long-distance contexts. It may be because both short-term and long-term contexts are useful for understanding



(a) Vanilla Transformer.



(b) DA-Transformer. The first two heatmaps are produced by heads with $w_i < 0$ and others are produced by heads with $w_i > 0$.

Figure 8: The self-attention weights learned by the vanilla Transformer and our proposed DA-Transformer method.

news topics. From Figs. 6(a), we find that nearly half of the w_i in the word-level Transformer are negative, but their absolute values are much larger than the positive ones. In addition, as shown in Fig. 6(b), the values of w_i are dominated by negative ones. These results show that both local and global contexts are useful for the review sentiment analysis task, but local contexts need to be more emphasized. From Fig. 7(a), we find long-term information is somewhat more important than local information in modeling news texts for news recommendation. However, From Fig. 7(b) we find an interesting phenomenon that only one head has a strong negative w_i while the values of w_i in all the rest heads are positive. It means that only one attention head tends to capture short-term user interests while all the other heads prefer to capture long-term user interests. This is intuitive because users usually tend not to intensively click very similar news and their long-term interests may have more decisive influence on their news clicks. These results show that our method can flexibly adjust its preference on short-term or long-term information according to the task characteristics by learning different values of w_i .

We then visualize the attention weights produced by the vanilla Transformer and the distance-aware attention weights in our DA-Transformer method. The attention weights of a sentence in the AG dataset computed by four different attention heads are respectively shown in Figs. 8(a) and 8(b). From Fig. 8(a), we find it is difficult to interpret the self-attention weights because they are too “soft”. In addition, it is difficult for us to understand the differences between the information captured by different attention heads. Different from the vanilla Transformer, from Fig. 8(b) we find that the attention weights obtained by our method are more sparse, indicating that the attention mechanism in our method is more focused. In addition, it is easier for us to interpret the results by observing the attention heatmap. For example, the first two heatmaps in Fig. 8(b) are produced by the two attention heads with preferences on short-term contexts. We can see that they mainly capture the relations among local contexts, such as the relations between “biotech” and “sector”. Differently, in the latter two heatmaps obtained by the two attention heads that prefer long-term contexts, we can observe that the model tends to capture the relations between

a word (e.g., “biotech”) with the global contexts. These results show that different attention heads in our method are responsible for capturing different kinds of information, and their differences can be directly observed from the self-attention weights. Thus, our method can be better interpreted than vanilla Transformers.

5 Conclusion and Future Work

In this paper, we propose a distance-aware Transformer (DA-Transformer), which can leverage the real distance between contexts to adjust the self-attention weights for better context modeling. We propose to first use different learnable parameters in different attention heads to weight the real relative distance between tokens. Then, we propose a learnable sigmoid function to map the weighted distances into re-scaled coefficients with proper ranges. They are further multiplied with the raw attention weights that are activated by the ReLU function to keep non-negativity and produce sharper attention. Extensive experiments on five benchmark datasets show that our approach can effectively improve the performance of Transformer by introducing real distance information to facilitate context modeling.

In our future work, we plan to explore how to incorporate the direction of distance into our model to make it direction-aware. In addition, we plan to incorporate our method into pre-trained language models like BERT to see whether our method can enhance their performance.

References

- Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, pages 632–642.
- Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, pages 2978–2988.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186.
- Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Deep sparse rectifier neural networks. In *AISTATS*, pages 315–323.
- Qipeng Guo, Xipeng Qiu, Pengfei Liu, Yunfan Shao, Xiangyang Xue, and Zheng Zhang. 2019. Star-transformer. In *NAACL-HLT*, pages 1315–1325.
- Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW*, pages 507–517.
- Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.
- Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *NAACL-HLT*, pages 2284–2293.
- Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, pages 1532–1543.
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *Technical report, OpenAI*.
- Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. 2018. Self-attention with relative position representations. In *NAACL-HLT*, pages 464–468.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*, pages 5998–6008.
- Xing Wang, Zhaopeng Tu, Longyue Wang, and Shuming Shi. 2019. Self-attention with structural position representations. In *EMNLP-IJCNLP*, pages 1403–1409.
- Chuhan Wu, Fangzhao Wu, Suyu Ge, Tao Qi, Yongfeng Huang, and Xing Xie. 2019. Neural news recommendation with multi-head self-attention. In *EMNLP-IJCNLP*, pages 6390–6395.
- Fangzhao Wu, Ying Qiao, Jiun-Hung Chen, Chuhan Wu, Tao Qi, Jianxun Lian, Danyang Liu, Xing Xie, Jianfeng Gao, Winnie Wu, et al. 2020. Mind: A large-scale dataset for news recommendation. In *ACL*, pages 3597–3606.
- Hu Xu, Bing Liu, Lei Shu, and S Yu Philip. 2019. Bert post-training for review reading comprehension and aspect-based sentiment analysis. In *NAACL-HLT*, pages 2324–2335.
- Hang Yan, Bocao Deng, Xiaonan Li, and Xipeng Qiu. 2019. Tener: Adapting transformer encoder for name entity recognition. *arXiv preprint arXiv:1911.04474*.
- Baosong Yang, Jian Li, Derek F Wong, Lidia S Chao, Xing Wang, and Zhaopeng Tu. 2019. Context-aware self-attention networks. In *AAAI*, volume 33, pages 387–394.