Induced Local Attention for Transformer Models in Speech Recognition

Tobias Watzel[0000−0002−3552−3325], Ludwig Kürzinger[0000−0001−5312−3870], Lujun Li[0000−0002−0641−3178], and Gerhard Rigoll[0000−0003−1096−1596]

Chair of Human-Machine Communication, Technical University of Munich
{tobias.watzel, ludwig.kuerzinger, lujun.li, rigoll}@tum.de

Abstract. The transformer models and their variations currently are considered the prime model architectures in speech recognition since they yield state-of-the-art results on several datasets. Their main strength lies in the self-attention mechanism, where the models receive the ability to calculate a score over the whole input sequence and focus on essential aspects of the sequence. However, the attention score has some flaws. It is heavily global-dependent since it takes the whole sequence into account and normalizes along the sequence length. Our work presents a novel approach for a dynamic fusion between the global and a local attention score based on a Gaussian mask. The small networks for learning the fusion process and the Gaussian masks require only few additional parameters and are simple to add to current transformer architectures. With our exhaustive evaluation, we determine the effect of localness in the encoder layers and examine the most effective fusion approach. The results on the dataset TEDLIUMv2 demonstrate a steady improvement on the dev and the test set for the base transformer model equipped with our proposed fusion procedure for local attention.

Keywords: Speech Recognition · Transformer · Local Attention · Attention Fusion

1 Introduction

Over the last years, sequence-to-sequence (Seq2Seq) models gain popularity as they are simple to train and require only little expert knowledge. The introduction of the transformer [18] proposed a novel way to eliminate the computational demanding long short-term memory (LSTM) layers by heavily relying on the self-attention (SA) mechanism. Despite the ordinary architecture, which mostly depends on feed-forward networks (FFNs), end-to-end speech recognition models based on the transformer were able to further reduce their word error rate (WER) on several different datasets. Nowadays, most state-of-the-art (SOTA) approaches rely on transformer model structure or its variations [2,10,13,17].

The SA is one of the critical elements in the transformer. This mechanism is based on an operation that predicts attention scores for the complete input sequence. These attention scores are gathered in global attention maps containing the relevance of each input element in the overall sequence and are normalized across the complete sequence. As a result, these attention maps describe a strong global dependency of the overall sequence. Therefore, the model is able to attend to all information in the input sequence and it can focus the importance of every element in the sequence by itself. However, this can be problematic. The SA mechanism performs the normalization by applying the softmax operation, whereby small values are getting smaller and large values are getting even larger. The valuable and important local context in the sequence is suppressed as only dominant values remain after the softmax operation.

A simple approach to support local context information is to restrict the global context. Diminishing the impact of the global context, i.e., create a local window, is already known. Luong et al. [5] proposed one of the first approaches in machine translation based on attention models [15]. The model predicts an aligned position token for each target word. Then, they utilized the predicted position as the mean of a Gaussian distribution to limit the computation context of the following context vector.

Later, the approach of adding a Gaussian window to focus more on the local context was transferred to the transformer [18] model in several works [12, 13, 20]. Shaw et al. [12] proposed to add a trainable parameter to the key vector. These parameters are the edges of a fully connected, directed graph, representing the relative positions in a predefined clipping range. Sperger et al. [13] utilized a Gaussian mask and added it to the attention maps. This mask acts as a bias onto the SA mechanism and does not work for the cross-attention between encoder and decoder. Instead of applying a single fixed window size (i.e., a fixed standard derivation parameter σ), they proposed a trainable σ for every head. This approach ensures that each head of the SA can determine its specific parameters to achieve local attention where it is necessary. In [20], the approach of adding a Gaussian mask was further extended. They proposed a flexible way to adjust the window size (the standard derivation σ) and the position (the mean μ) of the mask. Therefore, the limitation for only utilizing the mask in the SA was eliminated, and they demonstrated a solution for masking the cross-attention in the transformer. However, similar to [13], their main focus relied on determining the window size. Multiple approaches for predicting this window size were compared, from fixed window sizes to layer-dependent window sizes. Even though this solution was capable of inserting the mask to the cross-attention, most of the improvement was still achieved when a Gaussian mask was added to the encoder's SA.

Recently, Nguyen et al. [6] closed this gap by proposing a fully differentiable window, which is also applicable to the SA in the encoder, and the decoder, and in the cross-attention between encoder and decoder. They investigated different ways of adding the local window to the SA mechanism and where the local mask had the most significant impact on the performance of the overall transformer model. Their study demonstrated that in the case of machine translation, the best model is returned by utilizing an additive window in the encoder's SA, an additive segment-based window in the cross-attention, and a multiplicative window in the decoder's SA.

Nearly all these approaches are proposed in the domain of machine translation, where they return consistent improvements in their translation score. Even though [13] shows a way to utilize localness in automatic speech recognition (ASR), their best model applies an LSTM for modeling the positional encoding. To the best of our knowledge, direct integration of localness into the global score of a transformer model for ASR has not be done. Our contributions are the following:

- We transfer the idea of local attention to the domain of ASR.
- We demonstrate that solely the encoder's SA can already benefit from localness.
- We propose a novel approach to fuse the local and global attention scores.

2 Proposed Method

2.1 Transformer Network

The transformer network relies on the SA mechanism to calculate a score of importance for each input element. As this SA takes into account the complete sequence, it can be considered as an attention with strong global context. The network itself is built up by stacked encoder and stacked decoder networks, connected by a cross-attention mechanism. In this work, we only examine the influence of localness on the stacked encoder. The standard SA is defined as a scoring between the query sequence $\boldsymbol{Q} = (\boldsymbol{q}_1, \cdots, \boldsymbol{q}_i, \cdots, \boldsymbol{q}_I)$ and the key sequence $\boldsymbol{K} = (\boldsymbol{k}_1, \cdots, \boldsymbol{k}_i, \cdots, \boldsymbol{k}_I)$ of length I:

$$\mathrm{Score}(\boldsymbol{Q}, \boldsymbol{K}) = \frac{(\boldsymbol{Q}\boldsymbol{W}^Q)(\boldsymbol{K}\boldsymbol{W}^K)^T}{\sqrt{d}}. \tag{1}$$

Since the scoring values are not normalized, a softmax operation is applied, followed by the value sequence $\boldsymbol{V} = (\boldsymbol{v}_1, \cdots, \boldsymbol{v}_i, \cdots, \boldsymbol{v}_I)$ to return the final SA:

$$\mathrm{SelfAttention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Softmax}(\mathrm{Score}(\boldsymbol{Q}, \boldsymbol{K}))\boldsymbol{V}\boldsymbol{W}^V. \tag{2}$$

Here, $\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i \in \mathbb{R}^d$ with the vector dimension d are combined into the matrices $\boldsymbol{Q}$, $\boldsymbol{K}$, $\boldsymbol{V} \in \mathbb{R}^{I \times d}$, respectively and connected to the corresponding trainable weight matrices $\boldsymbol{W}^Q$, $\boldsymbol{W}^K$,$\boldsymbol{W}^V \in \mathbb{R}^{d \times d}$. Note that in case of the SA, $\boldsymbol{Q} = \boldsymbol{K} = \boldsymbol{V}$ and correspond to the output of the previous layer.

For a single attention head, the model would be restricted to certain positions learned during training. We obtain a more flexible model by splitting the single SA to a multi-head attention (MHA) approach [18]:

$$\mathrm{MultiHeadAttention}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = \mathrm{Concat}(\boldsymbol{h}_1, \cdots, \boldsymbol{h}_n, \cdots, \boldsymbol{h}_N)\boldsymbol{W}^{\mathrm{O}}, \tag{3}$$

where we concatenate the output of each head $\boldsymbol{h}_n$ and transform the concatenation into the output space by $\boldsymbol{W}^{\mathrm{O}} \in \mathbb{R}^{d \times d}$. Each head $\boldsymbol{h}_n$ corresponds to a SA:

$$\begin{aligned} \boldsymbol{h}_n &= \mathrm{SelfAttention}(\boldsymbol{Q}_n, \boldsymbol{K}_n, \boldsymbol{V}_n) \\ &= \mathrm{Softmax}\left(\frac{(\boldsymbol{Q}_n \boldsymbol{W}_n^Q)(\boldsymbol{K}_n \boldsymbol{W}_n^K)^T}{\sqrt{d_{\mathrm{mha}}}}\right)(\boldsymbol{V}_n \boldsymbol{W}_n^V), \end{aligned} \tag{4}$$

where $d_{\mathrm{mha}} = \frac{d}{N}$, $\boldsymbol{Q}_n, \boldsymbol{K}_n, \boldsymbol{V}_n \in \mathbb{R}^{I \times d_{\mathrm{mha}}}$ and $\boldsymbol{W}_n^Q, \boldsymbol{W}_n^K, \boldsymbol{W}_n^V \in \mathbb{R}^{d_{\mathrm{mha}} \times d_{\mathrm{mha}}}$ are trainable parameters. The number N of total heads $\boldsymbol{h}_n$ can be chosen freely.

2.2 Local Attention via Flexible Gaussian Window

In the following, we demonstrate the local attention only for the SA since the formulas would be heavily cluttered with indices in the case of MHA. The approach can seamlessly be transferred to the MHA with its parameters.

Local attention can be achieved by defining a Gaussian mask $\boldsymbol{G}$:

$$\boldsymbol{G}_{i,j} = -\frac{(j - P_i)^2}{2\sigma_i^2}, \tag{5}$$

where $\boldsymbol{G} \in \mathbb{R}^{I \times I}$ with $\boldsymbol{G}_{i,j} \in [0, -\infty)$. The mask is adjustable in its position P_i and its window size $\sigma_i = \frac{D_i}{2}$. The parameter P_i and D_i are learned by a FFNs and restricted to the current input length I, which is defined below. The mask provides the model with the ability to determine localness by itself if necessary. For example, if the model is lowering the standard deviation σ_i via D_i, it is able to focus on relevant parts of the sequence and ignore the irrelevant ones. On the other hand, if it is crucial to have global sequence information, the model can widen its focus by increasing σ_i.

2.3 Trainable Parameters of the Gaussian Mask

In order to predict the central position p_i and the window size z_i, we follow the approach in [20]:

$$\begin{pmatrix} P_i \\ D_i \end{pmatrix} = I \cdot \mathrm{Sigmoid}\begin{pmatrix} p_i \\ z_i \end{pmatrix}. \tag{6}$$

The values p_i and z_i are learned by FFNs and are integrated into the SA procedure. The predict values define a well-fitting Gaussian mask, which induces local attention in the SA.

Central Position Prediction The score in Equation 1 is calculated between the key sequence $\boldsymbol{K}$ and the query sequence $\boldsymbol{Q}$. In our case, we add the local attention to the encoder's SA. Therefore, we make our prediction for the central position p_i dependent on $\boldsymbol{K}$ or $\boldsymbol{Q}$. Similar to [20], we also utilize the query sequence $\boldsymbol{Q}$ to transform it into a hidden positional state p_i:

$$p_i = \boldsymbol{u}_{\mathrm{p}}^T \tanh(\boldsymbol{W}_{\mathrm{p}} \boldsymbol{q}_i), \tag{7}$$

where $\boldsymbol{u}_{\mathrm{p}} \in \mathbb{R}^d$ and $\boldsymbol{W}_{\mathrm{p}} \in \mathbb{R}^{d \times d}$ are trainable linear transformations.

Window Size Prediction There are several ways to set or learn a specific window size z_i [20]. Besides setting a fixed window size, it is possible to make the prediction dependent on the mean of the key sequence $\boldsymbol{K}$. In this way, we condense all the information of $\boldsymbol{K}$ into a layer-dependent value z. However, we want to give the model as much flexibility as possible. Thus, we select the approach where we depend on all the predictions of the query sequence $\boldsymbol{Q}$:

$$z_i = \boldsymbol{u}_{\mathrm{d}}^T \tanh(\boldsymbol{W}_{\mathrm{p}} \boldsymbol{q}_i), \tag{8}$$

where $\boldsymbol{u}_{\mathrm{d}} \in \mathbb{R}^d$ denotes a trainable linear transformation. The advantage of reusing the transformation $\boldsymbol{W}_{\mathrm{p}} \boldsymbol{q}_i$ is that we receive enough flexibility to learn the corresponding parameters p_i and z_i with only a few additional parameters [20].

2.4 Global and Local Attention Score Fusion

There are multiple ways, how to integrate the Gaussian mask $\boldsymbol{G}$ into the SA mechanism. First, we revisit the fusion approach from [20]. We propose two refinements for the fusion process to enhance the integration of the local attention.

Bias Attention Fusion The simplest method is to add the local score $\boldsymbol{G}$ to the global scoring in Equation 1, where the mask acts like a bias [20]:

$$\mathrm{Score}(\boldsymbol{Q}, \boldsymbol{K}) = \frac{(\boldsymbol{Q}\boldsymbol{W}^Q)(\boldsymbol{K}\boldsymbol{W}^K)^T}{\sqrt{d}} + \boldsymbol{G}. \tag{9}$$

We believe that it is challenging for the model to create a local mask utilizing the standard weight matrices $\boldsymbol{W}^Q, \boldsymbol{W}^K, \boldsymbol{W}^V \in \mathbb{R}^{d \times d}$, transforming the queries, keys, and values into their own space containing global information.

Improved Attention Fusion Inspired by [6], we add weight matrices $\boldsymbol{W}_{\mathrm{local}}^Q$, $\boldsymbol{W}_{\mathrm{local}}^K$, $\boldsymbol{W}_{\mathrm{local}}^V \in \mathbb{R}^{I \times d}$ for the local attention. However, in [6], they utilize a differential window instead of a Gaussian mask. For that reason, we transfer the idea of their fusion process to our approach:

$$S_{\mathrm{global}} = (\boldsymbol{Q}\boldsymbol{W}^Q)(\boldsymbol{K}\boldsymbol{W}^K)^T \tag{10}$$

$$S_{\text{local}} = (\boldsymbol{Q}\boldsymbol{W}^{Q}_{\text{local}})(\boldsymbol{K}\boldsymbol{W}^{K}_{\text{local}})^{T} \odot \boldsymbol{G} = S'_{\text{local}} \odot \boldsymbol{G}, \tag{11}$$

where $\odot$ denotes an element-wise multiplication and set the final scoring in Equation 9 with the attention score:

$$\text{Score}(\boldsymbol{Q}, \boldsymbol{K}) = \frac{S_{\text{global}} + S_{\text{local}}}{\sqrt{d}}. \tag{12}$$

The local weight matrices share the same dimensionality as the global weight matrices. Now, the model is more flexible in generating local attention masks since the dependency of the global weight matrices is removed.

Adjustable Attention Fusion Although the SA is now split into two independent attention branches, the additional term in Equation 12 still weights the global score S_{global} and the local score S_{local} equally, which could not be optimal, e.g., if S_{global} is more relevant for a precise prediction. To cope with this issue, we insert a weighting parameter α to Equation 12:

$$\text{Score}(\boldsymbol{Q}, \boldsymbol{K}) = \frac{\alpha\, S_{\text{global}} + (1-\alpha)\, S_{\text{local}}}{\sqrt{d}}, \tag{13}$$

The parameter α is learned by FFNs:

$$\alpha = \text{Sigmoid}(\boldsymbol{u}_{\alpha}^{T} \tanh(\boldsymbol{W}_{\alpha}\overline{\boldsymbol{k}})), \tag{14}$$

where $\boldsymbol{u}_{\alpha} \in \mathbb{R}^{d}$, $\boldsymbol{W}_{\alpha} \in \mathbb{R}^{d\times d}$ and $\overline{\boldsymbol{k}} \in \mathbb{R}^{d}$ is the mean key over the key sequence $\boldsymbol{K}$.

3 Experiments

3.1 Training Setup

In order to test our approach for local attention, we evaluate our model on the dataset TEDLIUMv2 [9]. The dataset combines more than 200 h of training data which is already transcribed. The overall data is divided into train, test, and dev set with a lexicon of 150 k words. Furthermore, we perform different augmentation techniques. Before the actual training, we enhance the training data by applying speed perturbation [4], where the original signal is resampled with three different speed factors: 0.9, 1.0, and 1.1. Then, we extract 80-dimensional log Mel filterbanks as feature vectors, followed by 3-dimensional pitch features vectors with Kaldi [8], and concatenated the resulting vectors to the final 83-dimensional feature vector. Moreover, we generate byte pair encoding (BPE) units [11] of size 500 by utilizing the transcript and set these units as our target values. During training, we apply another augmentation technique SpecAugment [7], which warps the created features and blocks certain frequency channels or time steps via masking.

The transformer model is implemented in the ESPnet toolkit [19], where the pre-processed dataset is fed to the front-end of the transformer model. This front-end network sub-samples the input feature sequence utilizing two conv2D layers with ReLU activation functions. Each convolutional layer has $d_{\text{att}} = 256$ channels and employs a 3×3 kernel with a stride of length two. A linear layer with $d_{\text{att}} = 256$ dimensions serves as the output of the front-end, to which the position encoding from [18] is added.

The transformer contains an encoder and a decoder branch. The stacked encoder branch is built up by 12 layers with 2048 units, respectively. The decoder branch has only six layers and shares an equal amount of units as the encoder branch. We set the dimension of the SA mechanism to $d_{\text{att}} = 256$, which is applied to all encoder and decoder layers. We also utilize the benefit of the MHA and set the number of heads $N = 4$.

All our experiments are based on the identical training setup, whereas we vary between the different fusion approaches for adding localness. The local attention is only applied in the SA of the encoder layers. We train all our models for 50 epochs and set the batch size to 128. The transformer models are optimized by the Adam optimizer [3]. To avoid an early local minimum, we perform a warm-up phase [18], in which the learning rate is slowly increased until it is steadily decreased in the regular training setup. The warm-up phase includes 25 000 steps. For regularization purposes, we follow the approach in [18], where we apply the standard and residual dropout [14] with a rate of 0.1 in each encoder and decoder layer and smooth the target labels by 0.1 [16].

The resulting transformer model is trained by the Kullback–Leibler (KL) loss, which is guided by the auxiliary loss of the connectionist temporal classification (CTC) [1] network on top of the encoder branch. The CTC loss is weighted by 0.3.

During decoding, we combine the transformer and the corresponding CTC outputs. The predictions of the CTC network are weighted with 0.3. We apply a standard beam search with a beam size of 20 and omit the language model.

3.2 Ablation Study

Since the improvement mentioned in [6, 20] is located in the domain of machine translation and language modeling, it is not clear if the same application holds for the local attention in ASR. Therefore, we perform a short ablation study where to apply localness and to identify the most effective way to fuse the global score from Equation 10 and local score from Equation 11.

Effective Fusion We examine different approaches to effectively fuse the local and global attention score in the encoder. The standard transformer model without local attention acts as the baseline for our study. The comparison of the different fusion setups share the procedure described in Section 3.1 and results are exhibited in Table 1.

Table 1: Ablation study of how to fuse the local attention. The results are in WER and evaluated on TEDLIUMv2 [9].

	TEDLIUMv2	
Model	dev	test
Baseline w/o localness	10.0	9.2
+ Bias Attention Fusion [20]	10.1	9.0
+ Improved Attention Fusion	10.3	8.8
+ Adjustable Attention Fusion	9.8	9.1

Table 2: Ablation study of the layer location for integrating the local attention from Equation 13. The results are in WER and evaluated on TEDLIUMv2 [9].

	TEDLIUMv2	
Model	dev	test
Baseline w/o localness	10.0	9.2
Layer 1-3	10.1	8.8
Layer 1-6	10.2	8.9
Layer 1-9	10.0	8.8
Layer 1-12	9.8	9.1

Our baseline model achieves a WER of 10.0% on the dev and 9.2% on the test set, which is close to the SOTA results reported in the ESPnet repository[1]. The only difference is that we trained our model for only 50 epochs and decoded it with a beam size of 20.

In the first *Bias Attention Fusion* setup, we integrate the Gaussian mask to the transformer model, similar as proposed in [20]. During our experiments, we faced the problem that the proposed mask returned only minor WER reductions compared to the baseline model. Although we reduce the WER on the test set from 9.2% to 9.0%, we do not observe a similar performance gain on the dev set, where the WER increases from 10.0% to 10.1%. Since the improvements are not consistent, we think it is challenging to learn a favorable Gaussian mask if there is no local branch available which is entirely focusing on inducing localness.

For this reason, we extend the latter approach to the *Improved Attention Fusion* setup, where we separate the local and global attention scores. The extension further reduces the WER on the test set from 9.2% to 8.8%, though we notice an increase of the WER in the dev set from 10.0% to 10.3%. It seems that the model benefits from a separate local attention branch, however, without consistent WER reductions. A reason for these divergent results could be the final fusion between both scores, which is still equally weighted.

In the *Adjustable Attention Fusion* setup, we equip the model with the ability to weigh the global and local scores by itself. Therefore, we utilize the mean key $\overline{\boldsymbol{k}}$ of the key sequence $\boldsymbol{K}$ to predict an α value, which defines a fusion weight between the local and global attention score. We obtain a highly flexible model which returns consistent WER reductions. The final model reduces the WER for the dev set from 10.0% to 9.8% and for the test set from 9.2% to 9.1%.

[1] `https://github.com/espnet/espnet/blob/master/egs/tedlium2/asr1/RESULTS.md` (commit c881192)

Table 3: Final results in WER trained for 100 epochs between the current SOTA result and our best approach. Evaluation was done on TEDLIUMv2 [9] and was decoded with a beam size of 40.

TEDLIUMv2 [9]			
Model	#Param	dev	test
Baseline ESPnet	28 M	10.1	8.9
+ Adjustable Attention Fusion	29 M	10.0	8.7

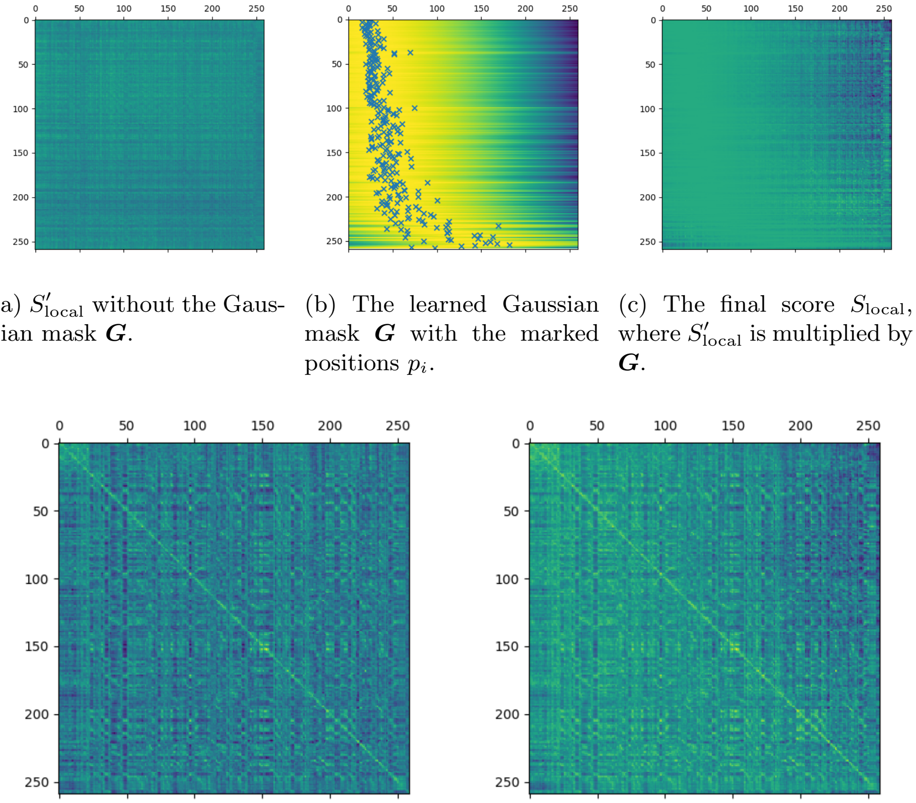
Location of Localness Recent approaches as [6,20] already demonstrated that the location of the local attention in the encoder's SA of the model is relevant and improves the transformer performance. They argue that the improvement results in the fact that the lower layers of the model process more low-level features, which contain more local information. As we do not know if it also holds for ASR, we define four setups, where we integrate localness in the encoder with our *Improved Attention Fusion* approach continuously. We begin with the *Layer 1-3* setup, where we apply the local attention from Equation 13 in the first three SA encoder layers. In the following three setups, we always add our local attention approach for the next three SA encoder layers until the complete encoder is equipped with it.

Our results in Table 2 reveal only a minor impact to the layer location of the local attention. For the *Layer 1-3* setup, we observe an improvement on the test set from 9.2% to 8.8% and a minor increase on the dev set from 10.0% to 10.1%. We obtain a similar result for localness in the first six layers, where the model achieves a decline in the WER on the test to 8.9%, however, a slight increase to 10.2% on the dev set. In the *Layer 1-9* setup, we are able to equalize with the baseline setup, where the model returns a WER of 10.0% on the dev set and reduces the WER on the test set from 9.2% to 8.8%.

The most consistent improvements are returned for the local attention employed in all SA layers of the encoder. For this setup, we are able to reduce the WER on the dev set from 10.0% to 9.8% and gain a slight decline on the test set from 9.2% to 9.1%. All in all, we do not observe similar findings as in [6,20]. One reason might be the length of the input feature sequence. Although the input sequence is sub-sampled to reduce its length, it is still several times longer than the output sequence. For machine translation, this is not the case since the input sentence and the output sentence share a high length overlap.

3.3 Final Results

For the final results, we trained our approach with a similar training setup as the current SOTA results reported in the ESPnet repository(c.f. footnote above). The baseline model and the extension *Adjustable Attention Fusion* are optimized for 100 epochs and decoded with a beam size of 40. The extension requires a minor increase from 28 M to 29 M total model parameters.



(a) S'_{local} without the Gaussian mask $\boldsymbol{G}$.

(b) The learned Gaussian mask $\boldsymbol{G}$ with the marked positions p_i.

(c) The final score S_{local}, where S'_{local} is multiplied by $\boldsymbol{G}$.

(d) The global score S_{global} without inducing local attention.

(e) The final attention score out of S_{global} and S_{local}, where $\alpha = 0.423$.

Fig. 1: The procedure of the *Adjustable Attention Fusion* with the final attention score. The global score S_{global} and the local S_{local} are determined and weighted by the parameter α. Certain parts of the final attention score in Figure 1e are getting boosted by S_{local}.

Our results in Table 3 demonstrate that our approach is competitive with current SOTA transformer model, hence, localness is also beneficial for transformer models in the domain of ASR. We are able to slightly reduce the WER on the dev set from 10.1% to 10.0% and on the dev set from 8.9% to 8.7%.

Furthermore, we depict in Figure 1 the qualitative results of our approach. In the upper row, we can observe the process of generating the local score S_{local}. First, in Figure 1a the model branch for the local attention determines the local score S'_{local} without the Gaussian mask $\boldsymbol{G}$. Then, in Figure 1b two FFNs predict the position p_i and window size z_i for each entry in $\boldsymbol{G}$. As the input sequence length increases, the position p_i of the Gaussian mask slowly transits to the end of the output sequence where the more relevant information of the score is. After the multiplication of S'_{local} and $\boldsymbol{G}$, we observe in Figure 1c that certain values of the local score are raised since the color shading is brighter, whereas other parts are lowered noticeable by the darker shading.

The fusion process of the S_{local} and S_{global} is shown in the lower row of Figure 1. In Figure 1d, the standard score S_{global} is plotted without applying any local attention. If the fusion from Equation 13 is applied, we obtain the final score in Figure 1e. There, it is observable that some values of the final score are assigned with higher importance since at the positions that plot is much brighter. As a result, we are able to demonstrate that our approach is visible in the quantitative as well as the qualitative results.

4 Conclusion

Our work presented a novel approach to induce localness into the global score of the transformer network's attention mechanism. Thereby, the local attention score is achieved by employing a Gaussian mask, where it is essential to fuse the global and local scores efficiently. Our novel fusion approach provides an excellent way to do so, with only a minor increase of the total model parameters. In our future work, we plan to integrate the local attention mechanism to the SA of the decoder network and the cross-attention between the encoder and decoder network.

References

1. Graves, A., Fernández, S., Gomez, F., Schmidhuber, J.: Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In: Proceedings of the 23rd international conference on Machine learning. pp. 369–376. ACM (2006)
2. Gulati, A., Qin, J., Chiu, C.C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., et al.: Conformer: Convolution-augmented transformer for speech recognition. arXiv preprint arXiv:2005.08100 (2020)
3. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
4. Ko, T., Peddinti, V., Povey, D., Khudanpur, S.: Audio augmentation for speech recognition. In: Sixteenth Annual Conference of the International Speech Communication Association (2015)
5. Luong, M.T., Pham, H., Manning, C.D.: Effective Approaches to Attention-Based Neural Machine Translation. arXiv preprint arXiv:1508.04025 (2015)
6. Nguyen, T.T., Nguyen, X.P., Joty, S., Li, X.: Differentiable window for dynamic local attention. arXiv preprint arXiv:2006.13561 (2020)
7. Park, D.S., Chan, W., Zhang, Y., Chiu, C.C., Zoph, B., Cubuk, E.D., Le, Q.V.: Specaugment: A simple data augmentation method for automatic speech recognition. arXiv preprint arXiv:1904.08779 (2019)
8. Povey, D., Ghoshal, A., Boulianne, G., Burget, L., Glembek, O., Goel, N., Hannemann, M., Motlicek, P., Qian, Y., Schwarz, P., et al.: The Kaldi speech recognition toolkit. In: IEEE 2011 workshop on automatic speech recognition and understanding. No. CONF, IEEE Signal Processing Society (2011)

9. Rousseau, A., Deléglise, P., Esteve, Y.: Enhancing the TED-LIUM corpus with selected data for language modeling and more TED talks. In: LREC. pp. 3935–3939 (2014)
10. Salazar, J., Kirchhoff, K., Huang, Z.: Self-attention networks for connectionist temporal classification in speech recognition. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 7115–7119. IEEE (2019)
11. Sennrich, R., Haddow, B., Birch, A.: Neural machine translation of rare words with subword units. arXiv preprint arXiv:1508.07909 (2015)
12. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. arXiv preprint arXiv:1803.02155 (2018)
13. Sperber, M., Niehues, J., Neubig, G., Stüker, S., Waibel, A.: Self-attentional acoustic models. arXiv preprint arXiv:1803.09519 (2018)
14. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research **15**(1), 1929–1958 (2014)
15. Sutskever, I., Vinyals, O., Le, Q.: Sequence to sequence learning with neural networks. Advances in NIPS (2014)
16. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
17. Tian, Z., Yi, J., Tao, J., Bai, Y., Wen, Z.: Self-attention transducers for end-to-end speech recognition. Proc. Interspeech 2019 pp. 4395–4399 (2019)
18. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
19. Watanabe, S., Hori, T., Karita, S., Hayashi, T., Nishitoba, J., Unno, Y., Soplin, N.E.Y., Heymann, J., Wiesner, M., Chen, N., et al.: Espnet: End-to-end speech processing toolkit. arXiv preprint arXiv:1804.00015 (2018)
20. Yang, B., Tu, Z., Wong, D.F., Meng, F., Chao, L.S., Zhang, T.: Modeling localness for self-attention networks. arXiv preprint arXiv:1810.10182 (2018)