A TIME-RESTRICTED SELF-ATTENTION LAYER FOR ASR

Daniel Povey[1,2]*, Hossein Hadian*[1]*, Pegah Ghahremani*[1]*, Ke Li*[1]*, Sanjeev Khudanpur*[1,2]

[1]Center for Language and Speech Processing, Johns Hopkins University, Baltimore, MD, USA,
[2]Human Language Technology Center of Excellence, Johns Hopkins University, Baltimore, MD, USA.

ABSTRACT

Self-attention – an attention mechanism where the input and output sequence lengths are the same – has recently been successfully applied to machine translation, caption generation, and phoneme recognition. In this paper we apply a restricted self-attention mechanism (with multiple heads) to speech recognition. By "restricted" we mean that the mechanism at a particular frame only sees input from a limited number of frames to the left and right. Restricting the context makes it easier to encode the position of the input – we use a 1-hot encoding of the frame offset. We try introducing attention layers into TDNN architectures, and replacing LSTM layers with attention layers in TDNN+LSTM architectures. We show experiments on a number of ASR setups. We observe improvements compared to the TDNN and TDNN+LSTM baselines. Attention layers are also faster than LSTM layers in test time, since they lack recurrence.

Index Terms— ASR, attention, lattice-free MMI, neural network, LSTM

1. INTRODUCTION

Attention-based models have recently become popular as they have been successfully applied to a variety of tasks such as machine translation [1], caption generation [2], and phoneme recognition [3]. However, they have been less successful when used for large vocabulary speech recognition [4, 5].

Attention-based models are usually used with encoder-decoder structures, where an encoder neural network maps the variable-length input sequence (e.g. a sequence of speech frames) into a fixed-size vector, and a decoder network - which takes this vector as input - generates the sequence of output labels using an attention mechanism that focuses on the relevant part of the input at each time step.

Recently, a new self-attention layer was proposed in [6], and was successfully used for neural machine translation. This layer has multiple heads which can jointly attend to different subspaces of the input representation and uses dot-product as the attention function.

In the work presented here, we adopt that self-attention layer in a time-restricted fashion, which is more suitable for speech recognition, where the input sequence consists of speech frames and can be considerably long. Previously time-restricted (or *local*) attention has been used by [7] in the context of machine translation. In that work, for each time step, the center of local attention is predicted using a weight matrix, but in our case, the input and output sequences have the same length, therefore we can safely assume that the center of attention is the current time. In addition, similar to [6], our attention mechanism is soft which means it can jointly attend to different points in time with different weights. This is different from hard attention in which the network attends entirely to a single point in time [2].

We use this time-restricted self-attention layer in our state-of-the-art lattice-free MMI (maximum mutual information) models [8] to further improve their performance in large-scale speech recognition tasks. The two common neural network structures that we use with LF-MMI are TDNN (time-delay neural network) and TDNN-LSTM which has interleaving TDNN and projected Long Short-Term Memory layers [9][10]. While the TDNN-LSTM setup achieves better results, the TDNN setup is significantly faster and has a reduced latency for online decoding. We use our time-restricted attention layer in both TDNN and TDNN-LSTM structures and show through experiments that it can improve the performance in both setups as well as speed up decoding in the TDNN-LSTM setup.

This is not "attention-based speech recognition"– we are not using attention to replace the left-to-right alignment of the HMM. It is simply an alternative to TDNN and LSTM layers in our model topology.

The rest of this paper is organized as follows. Section 2 describes the proposed self-attention layer in details. The experimental setup is explained in Section 3. Section 4 presents the experiments and results. Finally, the conclusions are presented in Section 5.

This work was partially supported by DARPA LORELEI award number HR0011-15-2-0024, NSF Grant No CRI-1513128 and IARPA MATERIAL award number FA8650-17-C-9115.

2. TIME-RESTRICTED SELF-ATTENTION LAYER

Our proposed time-restricted attention layer is comprised of 4 components as shown in Figure 1. All the trainable parameters are in the first component, which is an affine component. After the affine component, we have the attention nonlinearity component, and finally, a ReLU nonlinearity component followed by batch normalization [11] (without the trainable offset and scale). This structure is equivalent to the original multi-head attention in [6] but the nonlinearity and the trainable parameters are separated for efficient implementation. The actual attention mechanism takes place in the attention component, as shown in Figure 2.

Assuming the one-head case for simplicity, the attention component interprets its input $\boldsymbol{x_t}$ as being three things appended together: $\boldsymbol{q_t}$ and $\boldsymbol{k_t}$ and $\boldsymbol{v_t}$ which are the query, key and value respectively. The order in which we divide the input to key/query/value is not important as long as we do it consistently. The output $\boldsymbol{y_t}$ is a weighted sum (over time) of the values $\boldsymbol{v_t}$, where the weights are determined by dot products of the queries and the keys (normalized via softmax). This gives us:

$$\boldsymbol{y_t} = \sum_{\tau=t-L}^{t+R} c_t(\tau)\boldsymbol{v_t} \tag{1}$$

where $c_t(\tau) = \exp(\boldsymbol{q_t} \cdot \boldsymbol{k_\tau})/Z_t$ where Z_t ensures $\sum_\tau c_t(\tau) = 1$. In other words, the attention weight vector $\boldsymbol{c_t}$ is the softmax of a vector of key-query dot products.

2.1. Extension with positional encoding

In the formulation above, the model does not 'know' the relative position in time of the key and query. We introduce a positional encoding mechanism on the keys and values, as follows. Suppose $\mathbf{x}$ is an arbitrary vector, then let $\mathbf{extend}(\mathbf{x}, \tau, t)$ be $\mathbf{x}$ extended with a one-hot encoding of the relative position of τ versus t. The dimension of this expression is greater than the dimension of $\mathbf{x}$ by $L + 1 + R$ which is the number of possible relative positions, and it extends $\mathbf{x}$ by adding a vector which is all zeros except there is a one in position $\tau + L - t$ (assuming zero-based indexing). We change the above formulation by writing instead

$$\boldsymbol{y_t} = \sum_{\tau=t-L}^{t+R} c_t(\tau)\mathbf{extend}(\boldsymbol{v_t}, \tau, t) \tag{2}$$

$$c_t(\tau) = \frac{\exp(\boldsymbol{q_t} \cdot \mathbf{extend}(\boldsymbol{k_t}, \tau, t))}{Z_t} \tag{3}$$

so now the output dimension is greater by $L+1+R$ than the dimension of $\vec{v_t}$, and the dimension of the queries is greater by $L + 1 + R$ than the dimension of the keys.

2.2. Extension to multiple heads

Extension to more heads is straightforward because the heads operate independently. Supposing there are 10 heads, the input and output would be interpreted as 10 equal-sized blocks, with the computation described above happening within each block. This is illustrated in Figure 2b.

For left and right contexts that are not available, we just let the input to the attention component (i.e. the output of the preceding affine component) be zero.

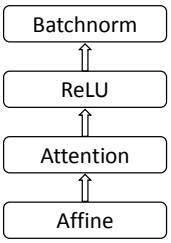

Fig. 1. The attention layer and the comprising components.

3. EXPERIMENTAL SETUP

For running the experiments, we use the Kaldi speech recognition toolkit [12]. This toolkit is open-source and the source codes related to this study are available online for reproducing the results. Experiments are done on four corpora: Wall street Journal [13], TED-LIUM [14], Switchboard [15] and AMI. For Switchboard, we report results on the full HUB5 '00 evaluation set and its "switchboard" and "CallHome" subsets, which are indicated in the results by "eval/fullset", "eval/swbd" and "eval/callhm" respectively. In the final results, we also report word error rates on the RT03 test set (LDC2007S10). The notation we use to show left/right context for the self-attention layer is $[-\mathbf{left}, \mathbf{right}]$. For example, context $[-15, 6]$ means the attention layer has a left context of 15, and a right context of 6.

4. EXPERIMENTS

In the following subsections, we try to investigate different aspects of the attention layer. In particular, we investigate the effect of the number of heads, key/value dimension and the size of the restricted time context. In these experiments, we only report results on Switchboard and/or TED-LIUM using the TDNN setup. In Section 4.7, however, we report results for all four databases in TDNN and TDNN-LSTM setups using a consistent configuration for the attention layer. In all the ASR experiments, we only report word error rates. Also, in all the tables, by "Baseline" we mean the case where no attention layer is used. We also report some preliminary experimental results for recurrent neural network language modeling.

4.1. Language modeling

As a preliminary experiment (to the main ASR experiments), we applied the proposed attention layer in the Kaldi based RNNLM setup [16]. Table 1 shows the perplexity improvements we obtained by replacing a TDNN layer with an attention layer, which has 20 heads, a key/value dimension of 40/80, and a left context of 15. The right context is 0 to avoid seeing the future.

Model	AMI	Switchboard	WSJ
3 LSTM layers	72.2	48.9	56.3
1 LSTM + 1 TDNN + 1 LSTM	70.8	47.5	55.1
1 LSTM + 1 self-attention + 1 LSTM	**68**	**46.5**	**53.5**

Table 1. Perplexities of different models on test data

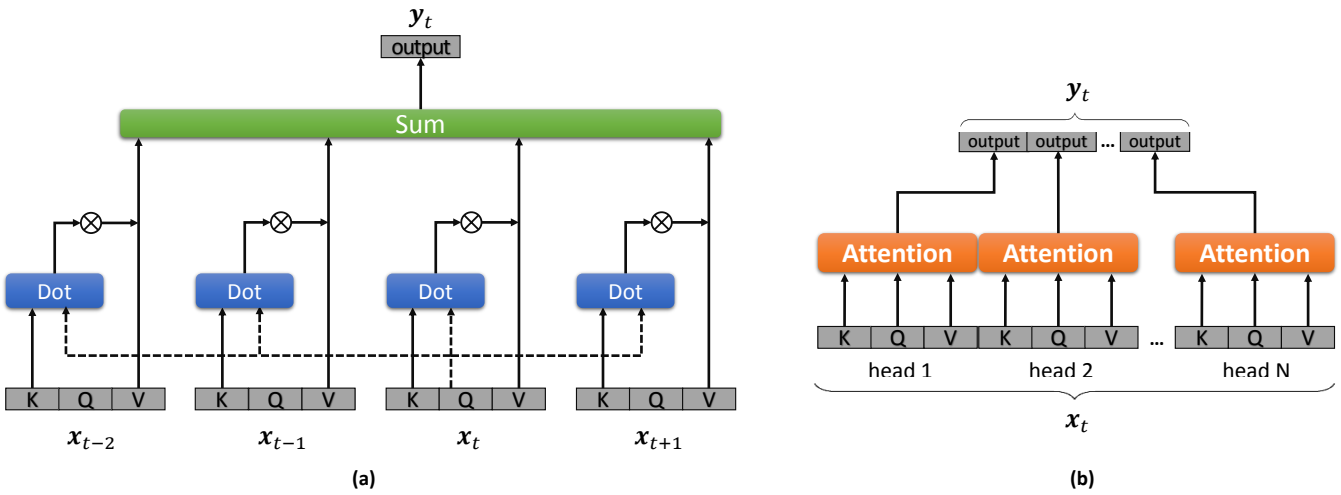

Fig. 2. (a) A single-head attention component. Left and right context sizes are 2 and 1 respectfully. For clarity, positional-encoding and the softmax (which is applied to the dot-products) are not shown. (b) A multi-head attention component using single-head attention blocks. K, Q, and V respectively mean key, query, and value.

Database	Test set	Baseline	L2	L4	L6	L7
Switchboard	eval/fullset	15.0	15.2	14.9	14.8	**14.6**
	eval/callhm	19.9	20.2	19.8	19.7	**19.5**
TED-LIUM	dev	8.6	8.4	8.4	**8.3**	8.4
	test	8.9	8.9	8.9	8.7	**8.5**

Table 2. Effect of location of the attention layer in the network. Li means layer i is attention.

Database	Test set	Baseline	Total context 13	19	25	31	37
Switchboard	eval/fullset	15.0	14.8	14.6	**14.5**	14.6	14.7
	eval/callhm	19.9	19.7	19.4	**19.3**	**19.3**	**19.3**
TED-LIUM	dev	8.6	8.4	**8.3**	8.4	8.6	8.4
	test	8.9	8.7	8.7	**8.6**	8.7	8.7

Table 3. Effect of symmetric context size.

4.2. Location of attention layer

In the first set of experiments, we simply replaced one hidden layer with an attention layer in our TDNN setup. The resulting word error rates appear in Table 2. The attention layer used has 15 heads, a context of $[-15, 6]$ and a key-dimension of 40. The value dimension is 80 for Switchboard and 60 for the smaller TED-LIUM task. It can be seen that the attention layer is more effective when used towards the end of the network. We also tried (not shown) using 2 or more attention layers, but it degraded the results. In the rest of the experiments, we always use a single attention layer near the end of the network.

4.3. Context

Table 3 shows results of using symmetric contexts with different total sizes. The results are not completely conclusive, but suggest that too wide or too narrow a context might degrade the results. Other experiments (not shown) showed that using different left vs. right context sizes did not make any significant difference. As a result, we use a context of $[-15, 6]$ in the rest of the experiments since it leads to a smaller latency for online decoding.

4.4. Number of heads

Table 4 shows word error rates for different number of heads in the self-attention layer, when the key/value dimensions are adjusted to yield the same total number of parameters in the network. The key to value ratio is 0.5 in all the experiments. We also include results for when the key/value dimensions are fixed to $30/60$.

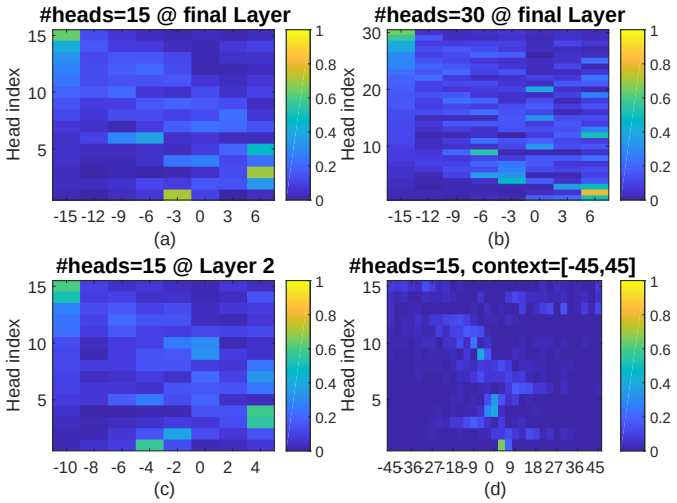

Fig. 3. Attention weight vector c_t for different attention configurations. The horizontal axis shows time.

4.5. Key vs. value dimension

Finally, we investigated the impact of the relative dimension of the value and the key. In this experiment, we change the ratio of key/value dimension, while keeping their sum fixed. As a result, the total number of parameters are the same in all cases. The results are

			Number of heads				
	Test set	Baseline	10	15	30	60	150
adjusted key/value dim	eval/fullset	15.0	14.7	14.6	14.7	**14.5**	14.7
	eval/callhm	19.9	19.6	19.3	19.4	**19.1**	19.5
	rt03	18.1	17.8	17.8	**17.7**	17.8	17.7
fixed key/value dim	eval/fullset	15.0	14.6	14.6	**14.5**	14.6	14.8
	eval/callhm	19.9	19.4	19.2	19.3	**19.1**	19.6
	rt03	18.1	17.5	17.7	17.5	**17.4**	17.6

Table 4. Effect of number of heads on Switchboard.

			key-dim/value-dim				
		Baseline	20/80	40/60	50/50	60/40	80/20
Switchboard	eval/callhm	19.9	19.4	**19.2**	19.6	19.4	19.5
	eval/fullset	15.0	**14.6**	**14.6**	14.8	14.8	14.8
TED-LIUM	dev	8.6	8.5	**8.3**	8.4	8.4	8.4
	test	8.9	**8.6**	**8.6**	**8.6**	8.8	8.8

Table 5. Impact of key/value dimension.

presented in Table 5. The results suggest that a key to value ratio of around 0.5 is slightly better than other ratios.

4.6. Self-attention analysis

Figure 3 shows the attention weight vector $\boldsymbol{c_t}$ averaged over a few mini-batches during training for different heads, where the heads are sorted w.r.t the 1^{st} weight (i.e. leftmost input in time). The plot is shown for different attention configurations all having context $[-15, 6]$ except 3d which has a wide context. By looking at 3{a,b}, we can see the weights for the first and last time-index are larger which shows higher attention to these time indexes. Figure 3c shows the attention is more uniform across the input time-indexes when used towards the beginning of the network (i.e. layer 2).

Figure 4 shows the aforementioned weight vector averaged over all heads when the context is $[-45, 45]$. It can be seen that frames further away in time are less important, which provides some justification for the choice to restrict the time context that the mechanism can see.

4.7. Final Results

Based on our findings in the previous subsections, we decided on a consistent configuration for the attention layer to test on various databases. This attention layer has 15 heads, a context of $[-15, 6]$, and a key-dimension of 40. The value dimension is adjusted according to the task size to a value in the range $[50, 80]$.

Results for using attention with TDNNs are presented in Table 6. Results for using attention with TDNN-LSTMs are shown in Table 7. In these experiments, we replaced the last LSTM layer with one attention layer. On average, the real-time factor for decoding the TDNN-LSTM models is 1.5 while it is 1.2 for TDNN-LSTM with attention.

		TDNN	TDNN+Attention
Switchboard	eval*	15/19.9/9.9	**14.6/19.4/9.7**
	rt03	18.1	**17.7**
TED-LIUM	dev	8.6	**8.3**
	test	8.9	**8.7**
WSJ	eval92	**2.5**	2.6
	dev93	**4.5**	4.6
AMI-SDM	eval	43.7	**43.3**
	dev	39.9	**39.5**
AMI-IHM	eval	21.4	**21.1**
	dev	21.4	**21.1**

* fullset/callhm/swbd

Table 6. Final TDNN results.

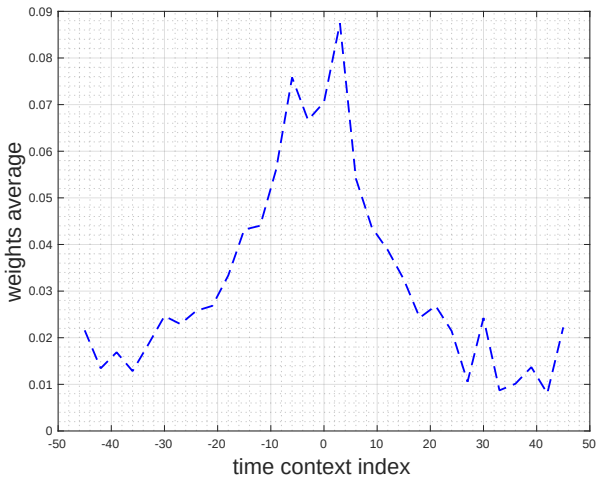

Fig. 4. Weight vector $\boldsymbol{c_t}$ averaged over all heads for an attention layer with 150 heads and context $[-45, 45]$.

		TDNN-LSTM	TDNN-LSTM+ Attention
Switchboard	eval*	14.3/19.3/9.4	**14.0/18.6/9.3**
	rt03	17.2	**16.5**
TED-LIUM	dev	8.3	**8.0**
	test	8.5	**8.2**
WSJ	eval92	**3.2**	3.3
	dev93	**5.7**	6.1
AMI-SDM	eval	42	**41.4**
	dev	38.6	**38.2**
AMI-IHM	eval	21.1	**20.5**
	dev	21.1	**20.9**

* fullset/callhm/swbd

Table 7. Final TDNN-LSTM results.

5. CONCLUSION

In this study, we introduced a time-restricted self-attention mechanism suitable for ASR and used it in our state-of-the-art LF-MMI models as a new layer, replacing a TDNN or LSTM layer. Through experiments on 4 different databases, we showed that using a single self-attention layer towards the end of the network can improve the WER by 0.2-0.6 in our TDNN and TDNN-LSTM setups (except on WSJ). In TDNN-LSTMs, it can also speed up decoding by 20%.

6. REFERENCES

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[2] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015, pp. 2048–2057.

[3] Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "End-to-end continuous speech recognition using attention-based recurrent nn: first results," *arXiv preprint arXiv:1412.1602*, 2014.

[4] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio, "Attention-based models for speech recognition," in *Advances in Neural Information Processing Systems*, 2015, pp. 577–585.

[5] Dzmitry Bahdanau, Jan Chorowski, Dmitriy Serdyuk, Philemon Brakel, and Yoshua Bengio, "End-to-end attention-based large vocabulary speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4945–4949.

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin, "Attention is all you need," *arXiv preprint arXiv:1706.03762*, 2017.

[7] Minh-Thang Luong, Hieu Pham, and Christopher D Manning, "Effective approaches to attention-based neural machine translation," *arXiv preprint arXiv:1508.04025*, 2015.

[8] Daniel Povey, Vijayaditya Peddinti, Daniel Galvez, Pegah Ghahremani, Vimal Manohar, Xingyu Na, Yiming Wang, and Sanjeev Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi.," in *INTERSPEECH*, 2016, pp. 2751–2755.

[9] Haşim Sak, Andrew Senior, and Françoise Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[10] Vijayaditya Peddinti, Yiming Wang, Daniel Povey, and Sanjeev Khudanpur, "Low latency modeling of temporal contexts," *IEEE Signal Processing Letters (submitted)*, 2017.

[11] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[12] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.

[13] Douglas B Paul and Janet M Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[14] Anthony Rousseau, Paul Deléglise, and Yannick Estève, "Enhancing the ted-lium corpus with selected data for language modeling and more ted talks.," in *LREC*, 2014, pp. 3935–3939.

[15] John J Godfrey, Edward C Holliman, and Jane McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*. IEEE, 1992, vol. 1, pp. 517–520.

[16] Hainan Xu, Ke Li, Yiming Wang, Jian Wang, Shiyin Kang, Xie Chen, Daniel Povey, and Sanjeev Khudanpur, "Kaldi rnnlm, an importance-sampling based neural network language modeling toolkit," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, submitted.