INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Adapting Transformer to End-to-end Spoken Language Translation

Mattia A. Di Gangi[1,2], *Matteo Negri*[2], *Marco Turchi*[2]

[1]University of Trento, Italy
[2]Fondazione Bruno Kessler, Trento, Italy
{digangi,negri,turchi}@fbk.eu

Abstract

Neural end-to-end architectures for sequence-to-sequence learning represent the state of the art in machine translation (MT) and speech recognition (ASR). Their use is also promising for end-to-end spoken language translation (SLT), which combines the main challenges of ASR and MT. Exploiting existing neural architectures, however, requires task-specific adaptations. A network that has obtained state-of-the-art results in MT with reduced training time is Transformer. However, its direct application to speech input is hindered by two limitations of the self-attention network on which it is based: quadratic memory complexity and no explicit modeling of short-range dependencies between input features. High memory complexity poses constraints to the size of models trainable with a GPU, while the inadequate modeling of local dependencies harms final translation quality. This paper presents an adaptation of Transformer to end-to-end SLT that consists in: *i)* downsampling the input with convolutional neural networks to make the training process feasible on GPUs, *ii)* modeling the bidimensional nature of a spectrogram, and *iii)* adding a distance penalty to the attention, so to bias it towards local context. SLT experiments on 8 language directions show that, with our adaptation, Transformer outperforms a strong RNN-based baseline with a significant reduction in training time.

1. Introduction

By enabling to tackle sequence-to-sequence problems with a single end-to-end model, neural approaches based on the encoder-decoder architecture [1] with attention [2] achieved state-of-the-art results in MT [3–6] and obtained increasingly good performance in ASR [7–11]. End-to-end techniques are appealing, compared to the classic cascade approaches, as they *i)* prevent the error propagation that typically affects cascade approaches [12], *ii)* reduce inference latency, and *iii)* usually result in conceptually simpler models. These advantages led to the proposal of end-to-end solutions also for speech-to-text translation (or spoken language translation, SLT) [13–15]), which combines the main challenges of its two parent tasks. Indeed, SLT models have to perform a more complex mapping between input and output compared to ASR, as it has to deal with word reordering and ambiguous word meaning, in absence of an explicit transcript. For this reason, we are interested in adapting powerful MT architectures to effectively handle audio signal input. Transformer [5] is currently the most popular encoder-decoder architecture in MT. Its main component is the self-attention network (SAN), which has been used to obtain state-of-the-art results also in other NLP tasks [16, 17]. SANs have the same capabilities of modeling long-range dependencies as long short-term memories (LSTMs [18]) [19], and their computation can be parallelized like for convolutional neural networks (CNNs). Given these advantages, also recent works on ASR proposed SANs for both acoustic modeling [20, 21] and end-to-end ASR [11, 22, 23]. However, the application of SANs to SLT is not easy, and the best results so far have been obtained by industrial players with very large models based on LSTMs, which are costly and slow to train [24, 25], by exploiting large amounts of synthetic data [26]. In fact, the application of SANs to speech input has to face additional problems compared to handling textual data, in particular: *i*) the GPU memory complexity of SANs, which is quadratic in the sequence length, *ii*) the bidimensional dependencies along the time and frequency dimensions in a spectrogram [27], and *iii*) the absence of an explicit bias towards the local context (i.e. short-range dependencies between the input features). The first problem is usually tackled with downsampling methods [7, 14, 28] or, in [29], by performing training with TPUs on one small dataset. The second problem has been successfully approached with 2D adaptation strategies of the Transformer encoder [11]. The third problem has been addressed by explicitly modeling short-range dependencies for acoustic models either using hard masking [21], or penalizing the self-attention weighting based on the distance between input elements [20].

In this paper, we study for the first time, to the best of our knowledge, the above-mentioned problems in the context of applying Transformer to end-to-end SLT. In doing so, we show that: *i)* sequence compression with CNNs and downsampling enables SLT training on GPUs, *ii)* modeling 2D dependencies produces better results, and *iii)* biasing the encoder self-attention with a distance penalty improves translation quality. We perform experiments on two relatively small datasets, Augmented Librispeech [30] for English→French and IWSLT 2018 for English→German. Additionally, we validate the results on the 8 language directions covered by MuST-C [31], a large multilingual SLT corpus. Our adaptations of Transformer result in a model that significantly outperforms a strong end-to-end baseline both in translation quality and training speed.

2. Background

Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n)$ be a sequence of n audio time frames, with $\mathbf{x}_i \in R^d$, and $\mathbf{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m)$ be a sequence of m characters representing words in a target language. End-to-end SLT can be defined as a mapping $T : \mathbf{X} \rightarrow \mathbf{Y}$. The function is learned by minimizing the cross-entropy between the probabilities $\tilde{\mathbf{y}}$ estimated by our model with parameters θ and the true target distribution:

$$L(\theta) = \sum_{i=0}^{m} P(\tilde{\mathbf{y}}_i = \mathbf{y}_i | \mathbf{X}, \mathbf{y}_{<i}; \theta) \quad (1)$$

Sequence-to-sequence models can learn the mapping of two sequences $(\mathbf{X}, \mathbf{Y})$ with different sequence lengths. They usually consists of three conceptual blocks: an *encoder* that maps the input $\mathbf{X}$ into a memory bank $\mathbf{H}$, a *decoder* that predicts $\mathbf{y}_{i+1}$

http://dx.doi.org/10.21437/Interspeech.2019-3045

given $\mathbf{y}_{1\ldots i}$, and one or multiple *attentions* that provide the decoder with a *context vector* that summarizes the part of the source $\mathbf{H}$ relevant to the prediction for every decoder time step. In the following, we briefly describe two alternative encoder-decoder architectures for SLT: the LSTM-based model proposed in [28], which we use as a baseline for our experiments, and the Transformer.

2.1. Convolutional and Recurrent model

Beràrd et al. [28] proposed an attentive encoder-decoder model for SLT that takes in input sequences of audio features and outputs the target-language sequences at character level. The encoder processes the input with two consecutive fully-connected layers that expand the size of the representation, followed by two 2D convolutional layers with stride $(2, 2)$ to reduce the sequence length by a factor of 4. The output of the convolutions is then processed by three stacked LSTMs. The decoder consists of a two-layered deep transition [32] LSTMs with one attention layer in between. The final output of the decoder is a function of the concatenation of the LSTM output, the context vector computed through the attention model and the previous-character embedding. Henceforth, we will refer to this model as **CNN+LSTM**.

2.2. Transformer

Transformer [5] is an encoder-decoder architecture entirely based on attention networks. Given three sequences of vectors $\mathbf{Q}, \in R^{d\times T}, \mathbf{K}, \mathbf{V} \in R^{d\times Z}$, the attention computes a context vector c_i for each query time step $i \in \{0, 1, \ldots, T\}$ ($\mathbf{Q}_i$) as a weighted average of the values $\mathbf{V}$. The weights are computed as the normalized similarity score between each query value $\mathbf{Q}_i$ and all the key values $\mathbf{K}$:

$$d_i = \text{softmax}(\mathbf{Q}\mathbf{K}^T/\sqrt{d_{\text{model}}}) \cdot \mathbf{V} \tag{2}$$

where $\sqrt{d_{\text{model}}}$ is a constant scaling factor based on the layer size d_{model}. The core component of Transformer is the multi-head attention (MHA), a network that, given two input sequences $\mathbf{a}, \mathbf{b}$ computes attention between $\mathbf{a}$ and $\mathbf{b}$ in multiple, parallel branches, called heads. MHA is used to model dependencies both between encoder and decoder ($\mathbf{K}, \mathbf{V} = \mathbf{a}$ and $\mathbf{Q} = \mathbf{b}$), and within the two networks (self-attention, $\mathbf{K}, \mathbf{V}, \mathbf{Q} = \mathbf{a}$). As it is shown in Equation 2, MHA is fully content-based and, as such, it is position invariant. Positional information within a sequence is conveyed by means of a fixed positional encoding based on trigonometric functions:

$$\text{ENC}(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right) \tag{3}$$

$$\text{ENC}(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i+1}{d_{\text{model}}}}}\right) \tag{4}$$

Such encoding is used to generate a vector for each position pos, whose components are sinusoids and cosinusoids of different periods in order to represent short and long distances. The resulting vectors are summed to the input embedding to provide positional information to the attention network without modifying equation 2. Another relevant property of the MHA is the possibility to compute it in parallel for all the time steps in both $\mathbf{Q}$ and $\mathbf{K}$, as well as for all the multiple heads. This, however, comes at the cost of a quadratic memory complexity.

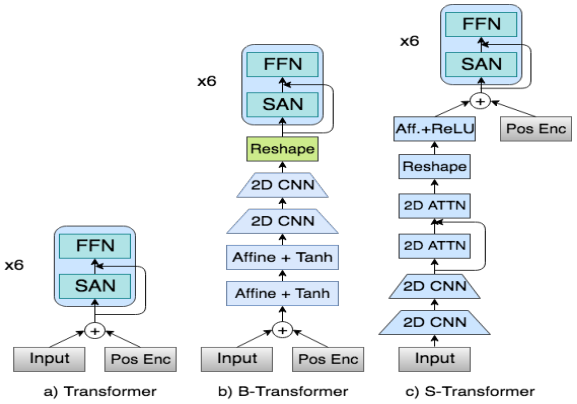

Figure 1: *Encoder of a) Transformer; b) our B-Transformer; and c) S-Transformer. Components in grey are non-learnable.*

3. SLT Transformer

The application of Transformer on speech data requires solutions to computational and modeling problems. The computational problem is Transformer's memory complexity for which we propose to reduce the input length. From a modeling point of view, Transformer cannot model temporal short-range dependencies [20, 21], or 2D dependencies over time and frequency [27] in a spectrogram, which is tackled by aggregating the input features and adding a penalty in the encoder self attention

3.1. Encoding with 2D CNNs

In this section, we propose two variants of Transformer. B-Transformer replaces the LSTM layers in CNN+LSTM with Transformer's encoder. S-Transformer is an improvement of B-Transformer that adds to the encoder the capability of modelling 2D dependencies in the input data. In all the variants, the adaptations regard only the layers preceding the Transformer encoder. The following Transformer encoder and decoder stacks are left unchanged.

B-Transformer (Figure 1a). Our baseline model uses the same encoder as CNN+LSTM [28] but replaces the LSTM layers with Transformer encoder layers. The replacement of LSTMs makes the encoder position invariant, and thus the sequential order is conveyed by summing the positional encoding directly to the input features.[1]

S-Transformer (Figure 1c). Our improvements over the baseline consist in: *i)* summing the positional encoding right before the self-attentional layers and not to the input signal; and *ii)* following the idea of modeling 2D joint dependencies in the input signal by applying a stack of 2D components to the input [11]. The first two CNNs capture local 2D-invariant features [33] in the input, while the following two 2D self-attention layers (Figure 2) model long-range context [11]. The 2D self-attention computes the three tensors $\mathbf{K}, \mathbf{Q}, \mathbf{V}$ with three parallel 2D CNNs of its input with c output channels. Each of the c channels is used as an attention head in an MHA network. $\mathbf{K}, \mathbf{Q}$ and $\mathbf{V}$ are used to compute the attention over the temporal dimension as in Equation 2. Then, the three matrices are transposed and another MHA is computed over the frequency dimension. Finally, the $2c$ channels from the two MHAs are

[1] Due to its high GPU memory occupation, we could not train a baseline Transformer (comparable in size to the other models used for experiments) without input compression.

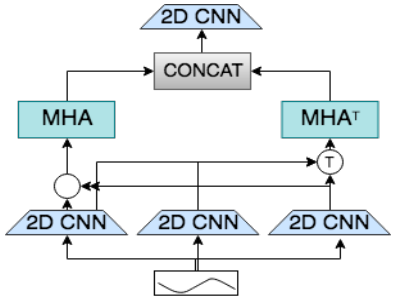

Figure 2: *Schematic representation of 2D self-attention.*

concatenated and processed by an additional $2D$ CNN with n output channels. The 2D attentions enrich the encoder representation by modelling 2D dependencies that cannot be captured by CNNs.

3.2. Distance Penalty

We propose to further bias the encoder towards short-range dependencies by introducing a distance penalty mechanism in the encoder self-attention. This mechanism penalizes long distances without imposing hard constraints on the possible distances. Equation 2 is modified as follows:

$$\mathbf{c} = \text{softmax}(\mathbf{Q}\mathbf{K}^{\mathbf{T}}/\sqrt{d_{\text{model}}} - \pi(\mathbf{D}))\mathbf{V} \quad (5)$$

where $\mathbf{D}$ is a matrix containing, in each cell $d_{i,j}$, the position distance $|i-j|$, and π is a distance penalty function. [20] introduced a Gaussian penalty that computes Gaussian-shaped penalty distributions with a distinct learnable variance σ for each head in the MHA as follows:

$$\pi_G(d) = \frac{(d)^2}{2\sigma^2} \quad (6)$$

This function gives to the network the flexibility to shrink or extend the attention span in each attention head, allowing it to model different dependency ranges. The downside of this approach is that the initial value of the variance is an additional hyperparameter that highly affects the performance. In order to eliminate this additional hyperparameter, we propose to use a logarithmic function as a distance penalty:

$$\pi_{\log}(d) = \begin{cases} 0, & \text{if d = 0} \\ \log_e(d), & \text{else} \end{cases} \quad (7)$$

The logarithm biases the network towards the local context but the penalty grows slowly with distance, and thus does not impede the modeling of global dependencies.

4. Experiments

Datasets. We run our experiments on three SLT datasets. The first one is built from the training data of IWSLT En→De 2018 [34] and the test data from IWSLT 2014 [35].[2] The second dataset is the Augmented Librispeech corpus [30] that is produced using English audiobooks of novels translated into French. The last dataset is MuST-C [31], a multilingual corpus that we recently built from English TED talks and covers more language directions (En→De/Es/Fr/It/Nl/Pt/Ro/Ru). The datasets statistics are listed in Table 1.

[2]We could not use the IWSLT 2018 test data, because the gold standard has not been released.

Table 1: *Data statistics for IWSLT, Librispeech and MuST-C. Train, Valid and Test are numbers of sentence pairs.*

Corpus	Hours	Train	Valid	Test
IWSLT (En-De)	273	171K	1000	1000
Librispeech (En-Fr)	236	95K	1071	2048
MuST-C				
En-De	408	234K	1423	2641
En-Es	504	270K	1316	2502
En-Fr	492	280K	1412	2632
En-It	465	258K	1309	2574
En-Nl	442	253K	1423	2615
En-Pt	385	210K	1367	2502
En-Ro	432	240K	1370	2556
En-Ru	489	270K	1317	2513

Experimental setup. For a fair comparison of the different architectures, we first reproduce the setting for the recurrent baseline (CNN+LSTM, §2.1) as in [28]. Then, we adjust the Transformer to have a number of parameters similar to the recurrent one (∼9.5M). The CNNs have a 3×3 kernel and 16 output filters. The LSTMs in the baseline have a hidden size of 512, with 3 layers in the encoder and 2 in the decoder. The Transformer models have 6 layers in both encoder and decoder, with a layer size of 256 and a hidden size of 768. S-Transformer BIG uses same CNN sizes but a layer size of 512 and an hidden size of 1024. B-Transformer serves as a baseline to evaluate the impact of the proposed adaptations. We train our S-Transformer models with and without distance penalty, either Gaussian or logarithmic. We test all these configurations on the IWSLT and Librispeech corpora. Then, considered the results, we do not use B-Transformer in the experiments with MuST-C. Following [28, 36], we first train a model with the ASR part of each corpus and then we use it to initialize the weights of the SLT encoder. All the experiments are run on a single GPU Nvidia 1080 Ti.

Data processing and evaluation. 40-dimensional Mel filterbanks were extracted from the audio signals of each dataset using window size of 25 ms and step size of 10 ms. The frame energy feature was additionally extracted from the LibriSpeech audio, similar to [28]. All texts were tokenized and split into characters. Performance is evaluated with BLEU [37] at token level after aggregating the output characters into words.

5. Results and Discussion

Baseline. Our first goal is to evaluate the results of a baseline Transformer given the GPU memory constraints. To this aim, we compare CNN+LSTM with B-Transformer on Librispeech and IWSLT. As can be seen in Table 2, B-Transformer shows a considerable reduction in training time per epoch over the CNN+LSTM (3 to 4 times less), but at the cost of lower translation quality (-4 BLEU points on Librispeech and -2.1 BLEU points on IWSLT). Compressing the input representation makes the training of Transformer feasible for SLT, whereas we could not run a training of Transformer without adaptation with a comparable number of parameters, but it is not sufficient to make it competitive with a strong LSTM bseline.

Modeling 2D dependencies. The next step is to evaluate the enhancements in modeling 2D input proposed in S-Transformer. Its results are +3.5 BLEU points better than B-Transformer in Librispeech, and +2.7 on IWSLT, while hav-

Table 2: *Results on the Librispeech and IWSLT 2014 test set. Differences wrt the baseline (CNN+LSTM) are statistically significant (randomization test, p=0.05).*

Librispeech	BLEU ↑	Time (s)	Time/Ep.
CNN+LSTM	13.2	248K	$\sim$ 2.8K
B-Transformer	9.0	101K	$\sim$ 0.69K
S-Transformer	12.5	76K	$\sim$ 0.79K
- Gauss penalty	13.8	88K	$\sim$ 0.86K
- log penalty	13.5	76K	$\sim$ 0.86K
IWSLT	BLEU ↑	Time (s)	Time/Ep.
CNN+LSTM	9.2	112K	$\sim$ 2.9K
B-Transformer	7.1	67K	$\sim$ 1.0K
S-Transformer	9.8	89K	$\sim$ 1.1K
- Gauss penalty	10.8	90K	$\sim$ 1.2K
- log penalty	10.6	81K	$\sim$ 1.2K

Table 3: *Results on 8 language pairs covered by MuST-C. LSTM is the CNN+LSTM model. Results in columns 3-6 are computed with S-Transformer with logarithmic (log) or Gaussian (Gauss) distance penalty. Improvements over CNN+LSTM are statistically significant (randomization test, p=0.05 [38]).*

TGT	LSTM	log	Gauss	BIG+log	BIG+Gauss
De	12.9	**14.5**	14.4	**17.3**	16.2
Es	17.9	18.4	**18.6**	**20.8**	20.1
Fr	22.3	23.1	**24.0**	**26.9**	24.7
It	15.0	15.0	**15.6**	**16.8**	16.2
Nl	**18.2**	**18.1**	17.2	**18.8**	18.1
Pt	17.1	18.6	**19.7**	**20.1**	19.3
Ro	13.4	14.7	**15.0**	**16.5**	16.1
Ru	7.2	8.8	**9.1**	**10.5**	8.5

ing a similar parameter count. The convergence time is 25% shorter than B-Transformer on Librispeech, but 33% longer on IWSLT. Compared to CNN+LSTM, S-Transformer is significantly faster and its BLEU scores are -0.7 on Librispeech and +0.6 on IWSLT.

Distance penalty. In Table 2 we show the results obtained using the distance penalties introduced in §3.2 to model short-range dependencies in the Transformer encoder. Distance penalties produce performance improvements for S-Transformer that range from 0.8 to 1.3 BLEU points, with the Gaussian penalty (initial variance = 5.0) being $0.2 \sim 0.3$ BLEU points better than the logarithmic one. S-Transformer with Gaussian penalty obtains the best results in both corpora, with improvements of $+0.6$ and $+1.6$ BLEU points over CNN+LSTM on, respectively, Librispeech and IWSLT. These results show that biasing the self-attention with a distance penalty significantly improves translation quality and allows Transformer to outperform the strong CNN+LSTM model.

MuST-C. In Table 3, we report the results on 8 language directions of MuST-C using the baseline CNN+LSTM and S-Transformer with both distance penalties. S-Transformer outperforms CNN+LSTM on 6 language directions with gains from $+0.5$ to $+1.6$ BLEU points with log penalty and from $+0.7$ to $+2.6$ with Gaussian penalty. Gaussian penalty generally achieves results only slightly better than the logarithmic one, except for the top improvements of $+0.9$ and $+1.1$ respectively on En→Fr and En→Pt. To explain this difference, it is useful to recall that the parameters of the encoders of SLT models (including their Gaussian variances) are initialized from a model pre-trained on English ASR. In particular, for MuST-C we use the same model trained on the larger dataset. The inherited variance from this model may affect differently the different target languages. The only exceptions are En→It, where we observe only a small improvement with Gaussian penalty, and En→Nl where S-Transformer with Gaussian penalty underperforms CNN+LSTM. Motivating these results requires deeper investigation. Experiments with a larger model (S-Transformer BIG) show further improvements from a minimum of 0.7 points for En→Nl to a maximum of 3.8 points for En→Fr with log penalty, while the poor results with Gaussian penalty confirm that it is less stable than the log. The number of training iterations is also reduced to less than half of the base systems.

Table 4: *Results on the three tasks of ASR, MT and ST with cascade models. ASR performance is computed with WER in percentage (the lower the better), while MT and ST are computed with BLEU (the higher the better).*

TGT	ASR (WER)	MT (BLEU)	Cascade (BLEU)
De	27.0	25.3	18.5
Es	26.6	29.9	22.5
Fr	25.8	35.5	27.9
It	26.4	25.8	18.9
Nl	26.6	30.3	22.2
Pt	28.0	31.1	21.5
Ro	27.6	22.6	16.8
Ru	27.0	14.0	11.1

Cascade. Finally, we evaluate the data efficiency of our approach by comparing it with cascade systems trained on the same data. Our cascade model combines an ASR system built on our CNN+LSTM models (it outperformed S-Transformer in this task), and a Transformer-based MT system trained on bpe-segmented words. The input of MT is lowercased and without punctuation. The results of the cascade models are reported in Table 4. In most cases, the cascade outperforms the end-to-end model by only 1-2 BLEU points, with the only evident exception of En-Pt where the difference is 3.4 points. The ASR and MT systems are trained only with MuST-C data, whereas they could be stronger by adding more data for the task. Nonetheless, these results show that our S-Transformer can approach a stronger method when trained on the same data.

To conclude, our experiments show that our task-specific adaptations: *i)* enable training the Transformer for SLT; *ii)* make it able to outperform a strong LSTM-based baseline; *iii)* take significant advantage from a logarithmic distance penalty, which is preferable to the Gaussian one as it does not require additional hyperparameter tuning still resulting in competitive performance.

6. Conclusion

We proposed an adaptation of Transformer for SLT, targeting effective and efficient encoding of long audio sequences featuring local, short-range time/frequency dependencies. Our solution is based on: *i)* an input compression technique based on a 2D-oriented processing applied before the Transformer stack, and *ii)* a distance penalty to bias the self-attention towards local context. Results on three corpora covering 8 language directions indicate the effectiveness of our approach both in performance and computation time, showing significant improvements in particular when more data are available.

7. References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," in *Proceedings of NIPS 2014*.

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," in *Proceedings of ICLR 2015*, 2015.

[3] L. Bentivogli, A. Bisazza, M. Cettolo, and M. Federico, "Neural versus Phrase-Based Machine Translation Quality: a Case Study," in *Proceedings of EMNLP 2016)*, 2016.

[4] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional Sequence to Sequence Learning," in *Proceedings of ICML 2017*, Sydney, Australia, August 2017.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is All You Need," in *Proceedings of NIPS 2017*, 2017.

[6] M. X. Chen, O. Firat, A. Bapna, M. Johnson, W. Macherey, G. Foster, L. Jones, N. Parmar, M. Schuster, Z. Chen *et al.*, "The best of both worlds: Combining recent advances in neural machine translation," *arXiv preprint arXiv:1804.09849*, 2018.

[7] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[8] C.-C. Chiu, T. Sainath, Y. Wu, R. Prabhavalkar, P. Nguyen, Z. Chen, A. Kannan, R. J. Weiss, K. Rao, K. Gonina, N. Jaitly, B. Li, J. Chorowski, and M. Bacchiani, "State-of-the-art Speech Recognition with Sequence-to-sequence Models," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4774–4778.

[9] Y. Zhang, W. Chan, and N. Jaitly, "Very deep convolutional networks for end-to-end speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4845–4849.

[10] A. Zeyer, K. Irie, R. Schlüter, and H. Ney, "Improved Training of End-to-end Attention Models for Speech Recognition," in *Proceedings of Interspeech 2018*, 2018, pp. 7–11.

[11] L. Dong, S. Xu, and B. Xu, "Speech-Transformer: A No-Recurrence Sequence-to-Sequence Model for Speech Recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018.

[12] N. Ruiz, M. A. Di Gangi, N. Bertoldi, and M. Federico, "Assessing the tolerance of neural machine translation systems against speech recognition errors," *Proc. Interspeech 2017*, pp. 2635–2639, 2017.

[13] A. Bérard, O. Pietquin, L. Besacier, and C. Servan, "Listen and Translate: A Proof of Concept for End-to-End Speech-to-Text Translation," in *NIPS Workshop on end-to-end learning for speech and audio processing*, Barcelona, Spain, December 2016.

[14] R. J. Weiss, J. Chorowski, N. Jaitly, Y. Wu, and Z. Chen, "Sequence-to-Sequence Models Can Directly Translate Foreign Speech," in *Proceedings of Interspeech 2017*, Stockholm, Sweden, August 2017.

[15] A. Anastasopoulos and D. Chiang, "Tied Multitask Learning for Neural Speech Translation," in *Proceedings of NAACL 2018*, 2018.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[17] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 2016, pp. 551–561.

[18] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural computation*, vol. 9, no. 8, 1997.

[19] G. Tang, M. Müller, A. Rios, and R. Sennrich, "Why self-attention? a targeted evaluation of neural machine translation architectures," *arXiv preprint arXiv:1808.08946*, 2018.

[20] M. Sperber, J. Niehues, G. Neubig, S. Stüker, and A. Waibel, "Self-Attentional Acoustic Models," *Proceedings of Interspeech 2018*, pp. 3723–3727, 2018.

[21] D. Povey, H. Hadian, P. Ghahremani, K. Li, and S. Khudanpur, "A time-restricted self-attention layer for asr," in *Proceedings of ICASSP 2018*. IEEE, 2018, pp. 5874–5878.

[22] S. Zhou, L. Dong, S. Xu, and B. Xu, "Syllable-based sequence-to-sequence speech recognition with the transformer in mandarin chinese," *Proc. Interspeech 2018*, pp. 791–795, 2018.

[23] S. Zhou, S. Xu, and B. Xu, "Multilingual end-to-end speech recognition with a single transformer on low-resource languages," *arXiv preprint arXiv:1806.05059*, 2018.

[24] T. Lei, Y. Zhang, and Y. Artzi, "Training RNNs as Fast as CNNs," *arXiv preprint arXiv:1709.02755*, 2017.

[25] M. A. Di Gangi and M. Federico, "Deep Neural Machine Translation with Weakly-Recurrent Units," in *Proc. of EAMT*, 2018.

[26] Y. Jia, M. Johnson, W. Macherey, R.-J. Weiss, Y. Cao, C.-C. Chiu, S.-L. Ari, and Y. Wu, "Leveraging Weakly Supervised Data to Improve End-to-End Speech-to-Text Translation," *ArXiv e-prints arXiv:1811.02050*, 2018.

[27] J. Li, A. Mohamed, G. Zweig, and Y. Gong, "Exploring Multidimensional LSTMs for Large Vocabulary ASR," in *Proceedings of ICASSP 2016*. IEEE, 2016, pp. 4940–4944.

[28] A. Bérard, L. Besacier, A. C. Kocabiyikoglu, and O. Pietquin, "End-to-End Automatic Speech Translation of Audiobooks," in *Proceedings of ICASSP 2018*, Calgary, Alberta, Canada, April 2018.

[29] L.-C. Vila, C. Escolano, J. A. Fonollosa, and M.-R. Costa-Jussà, "End-to-End Speech Translation with the Transformer," *Proceedings of IberSPEECH 2018*, pp. 60–63, 2018.

[30] A. C. Kocabiyikoglu, L. Besacier, and O. Kraif, "Augmenting Librispeech with French Translations: A Multimodal Corpus for Direct Speech Translation Evaluation," in *Proceedings of LREC 2018*, Miyazaki, Japan, May 2018.

[31] M. A. Di Gangi, R. Cattoni, L. Bentivogli, M. Negri, and M. Turchi, "MuST-C: a Multilingual Speech Translation Corpus," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, Minneapolis, MN, USA, June 2019.

[32] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," in *Proceedings of ICLR 2014*, 2014.

[33] D. Amodei, S. Ananthanarayanan, R. Anubhai, J. Bai, E. Battenberg, C. Case, J. Casper, B. Catanzaro, Q. Cheng, G. Chen *et al.*, "Deep speech 2: End-to-end speech recognition in english and mandarin," in *International conference on machine learning*, 2016, pp. 173–182.

[34] J. Niehues, R. Cattoni, S. Stüker, M. Cettolo, M. Turchi, and M. Federico, "The IWSLT 2018 Evaluation Campaign," in *Proceedings of IWSLT 2018*, Bruges, Belgium, October 2018.

[35] M. Cettolo, J. Niehues, S. Stüker, L. Bentivogli, and M. Federico, "Report on the 11th IWSLT Evaluation Campaign, IWSLT 2014," in *Proceedings of IWSLT 2014*, 2014.

[36] S. Bansal, H. Kamper, K. Livescu, A. Lopez, and S. Goldwater, "Pre-training on High-resource Speech Recognition Improves Low-resource Speech-to-text Translation," *Proceedings of NAACL 2019*, 2018.

[37] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," in *Proceedings of ACL 2002*, 2002.

[38] J. H. Clark, C. Dyer, A. Lavie, and N. A. Smith, "Better hypothesis testing for statistical machine translation: Controlling for optimizer instability," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, 2011, pp. 176–181.