SRPOL's System for the IWSLT 2020 End-to-End Speech Translation Task

Tomasz Potapczyk, Pawel Przybysz
Samsung R&D Institute, Poland
{t.potapczyk,p.przybysz}@samsung.com

Abstract

This paper describes the submission to IWSLT 2020(Ansari et al., 2020) End-to-End speech translation task by Samsung R&D Institute, Poland. We took part in the offline End-to-End English to German TED lectures translation task. We based our solution on our last year's submission(Potapczyk et al., 2019). We used a slightly altered *Transformer*(Vaswani et al., 2017) architecture with ResNet-like(He et al., 2016) convolutional layer preparing the audio input to Transformer encoder. To improve the model's quality of translation we introduced two regularization techniques and trained on machine translated *Librispeech*(Panayotov et al., 2015) corpus in addition to *iwslt-corpus*, *TEDLIUM2*(Rousseau et al., 2014) and *Must_C*(Di Gangi et al., 2019) corpora. Our best model scored almost 3 *BLEU* higher than last year's model. To segment 2020 test set we used exactly the same procedure as last year.

1 Introduction

This paper describes the submission to IWSLT 2020 End-to-End Speech Translation task by Samsung R&D Institute, Poland.

We propose a few improvements to our previous system. Introducing additional training data gave us 0.7 *BLEU* improvement. Spectrogram augmentation techniques increased quality by 0.2 *BLEU*. Encoder layer depth of 12 layers gives an increase of *BLEU* by 1 point and 1.8 points when combined with additional training data and two spectrogram augmentation techniques. Replacing simpler convolutions with ResNet-like convolutional layers gave around 0.5 *BLEU* improvement. Combining all of these and increasing embedding size to 512 resulted in almost 3 *BLEU* improvement compared to our last year's model.

Document structure is as follows. Firstly we describe data preparation and augmentation. Then we provide system specification and training procedure used in our experiments. We describe data segmentation algorithm used to segment test sets TED 2019 and TED 2020. We show results of our experiments. Finally we conclude our results.

2 Training Data

To train our system we used only IWSLT 2020 permissible audio corpora - *iwslt-corpus*, *TEDLIUM2* (Rousseau et al., 2014), *Must_C* corpus(Di Gangi et al., 2019) and machine translated *Librispeech* (Panayotov et al., 2015) corpus. We did not do any further data preparation in case of iwslt-corpus and *TEDLIUM2*, we used the same data as in 2019. In case of *Must_C* corpus, this year we ran a training with half of translations being synthetic. This improved the score by 0.35 *BLEU*. We used the same text translation models as last year to generate synthetic translations. Both models scored around 33.8 and 31.1 *BLEU* on *tst2010* and *tst2015* sets respectively.

2.1 Data filtration

We trained English ASR system that was used to filter iwslt-corpus and *TEDLIUM2* corpora. We removed cases where *WER* score exceeded 75% when comparing ASR output and English reference. We decided that *Must_C* corpus does not need filtration. Additionally, we filtered iwslt-corpus with regard to quality of translation using statistical dictionary-based methods. Size of the corpora before and after filtration is shown in Table 1. We did not filter Librispeech corpus.

2.2 Synthetic target data

TEDLIUM2 corpus did not provide any German translations, therefore we generated synthetic targets using two *Transformer Big* MT systems trained with different hyperparameters on WMT data - *Paracrawl*, *Europarl* and *OpenSubtitles*.

Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT), pages 89–94
July 9-10, 2020. ©2020 Association for Computational Linguistics
https://doi.org/10.18653/v1/P17

Corpora	Size	Filtered	Length
iwslt-corpus(ASR)	171121	158737	224h
+ trans. quality	158737	126817	188h
TEDLIUM2	92973	90715	197h
Must-C	229703	229703	400h
Librispeech	281241	281241	1000h

Table 1: Size (number of audio utterances) of the training corpora before and after filtration. *Iwslt-corpus* (ASR) is corpus filtered by ASR only. The last column is total audio length after filtration.

Training data for these systems has been prepared with our in-house data preparation pipeline. We also used synthetic translations as an alternative translation in *iwslt-corpus* when augmenting it. To diversify target data as much as possible, for each example created in augmentation process, we generated 4 translations, 2 per each MT model. Such a technique was described in (Jia et al., 2019). Number of training examples with synthetic data are shown in Table 6. Last year we did not examine the effectiveness of synthetic translations on our models. This year we altered our corpus and included synthetic translations of *Must_C corpus*. This corpus was augmented 3 times and in the case of 2 versions out of 4, synthetic translations were used.

Corpora	Ref.	MT-1	MT-2
iwslt-corpus	126817	2x158737	2x158737
TEDLIUM2	0	3x90715	3x90715
Must_C	229703	229703	229703
Librispeech	281241	281241	281241

Table 2: Size (number of text lines) of the training corpora with synthetic data. For each model two or three best beam results have been used.

2.3 Data Augmentation

We augmented the data by processing the audio files with three Sox's effects: *tempo*, *speed* and *echo*. We sampled the parameters with uniform distribution within ranges presented in Table 3.

For each file we repeated the process four times. *Librispeech* was augmented only once because it is the largest corpus and it is out of domain. As a result we had nearly five times larger audio corpus. The range of *speed* option is very small because we did not want our model to train on an unnaturally sounding samples. The rationale behind using *echo* option is the fact that many TED lectures have a significant echo.

Option	Min value	Max value
tempo	0.85	1.3
speed	0.95	1.05
echo delay	20	200
echo decay	5	20

Table 3: Sox parameters value ranges used in processing of audio data. Echo effect is parametrized by two values.

Final number of training audio examples is shown in Table 4.

Corpora	Orig.&Augm.	Length
iwslt-corpus	761765	1084h
TEDLIUM2	544290	1182h
Must_C	918812	1600h
Librispeech	562482	2000h
Total	2787349	5866h

Table 4: Size of the training audio corpora with data augmentation. Number of distinct audio and text pairs.

3 E2E Speech Translation System

In this section we will describe the architecture and training techniques of our end-to-end spoken language translation system. Some of these were used in our 2019 system.

3.1 ASR Transformer for SLT

As a baseline system we used our last year's Transformer architecture implemented in TensorFlow. The *Baseline* Transformer has hidden layer of size 384, convolutional (kernel size 9) feed forward layer of size 1536, 8-head self-attention, 6 encoder layers and 4 decoder layers. Audio data is turned into log mel spectrogram with frame size of 25 ms, frame step 10 ms and 80 filters. To log mel spectrograms we apply 2D 3x3 convolution twice with stride 2x2 and 256 filters and then 3x20 convolution to reduce the spectrogram to a 384 dimentional vector, exactly like in the case of ASR. Apart from baseline, we propose changes to convolutional layer and increase number of encoder layers and embedding size. Finally, our best system has ResNet-like convolutional layer, 12 encoder layers and embedding size of 512.

3.2 Dual learning: ASR and SLT tasks

This year we also introduced a second decoder with ASR task, making it a multitask setup similar to (Anastasopoulos and Chiang, 2018). A separate dictionary of size 32k was used for this task. In such a setup loss is calculated with two targets - one in English and one in German. Two decoders with different weights are simultaneously trained on these targets; convolutional layers and encoder are shared. An early experiment on non-augmented data showed almost 2 *BLEU* increase (15.23 vs 17.15 on tst2010) compared to the same model trained on a single task. All our trainings this year used dual learning.

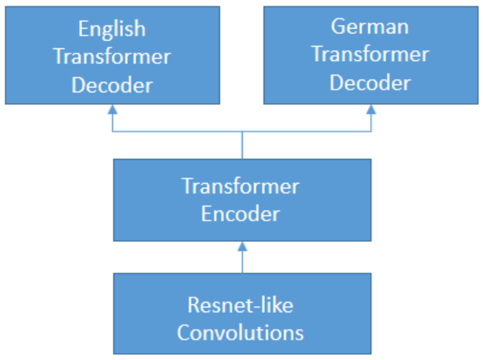

Figure 1: Dual learning architecture

3.3 Spectrogram augmentation

To augment data we implemented spectrogram masking technique described in (Park et al., 2019) This technique involves masking the spectrogram for a range of frequencies and periods of time. In our implementation we chose to introduce three such masks for frequency. The width of frequency range is selected randomly between values 5 and 10. This means that out of 80 filters 15 to 30 are masked. In time we chose one mask for every 300 time steps. Again, the length of such mask is random between 10 and 20 time steps. Apart from last year's data augmentation techniques used in 2019, we introduced additional regularization techniques - warp (Figure 2) and spectrogram noise. We implemented warping technique similar to Park et al.(Park et al., 2019). For each 10 time steps in spectrogram we delete one random time step and insert a step which is an average of two neighboring steps. The result is very similar to warp distortion - some parts of a spectrogram are shifted to the right and some are shifted to the left. We also

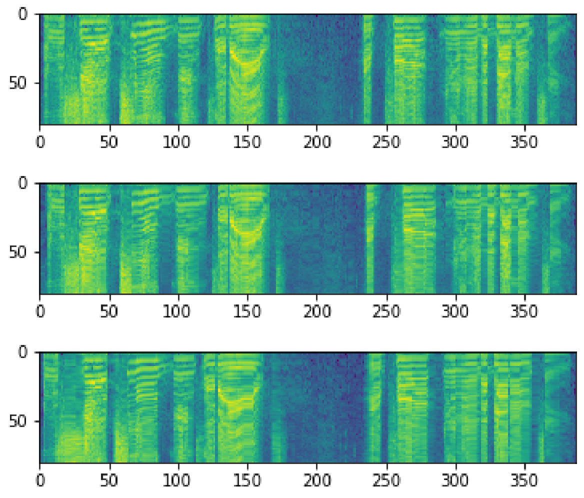

Figure 2: Original and warped spectrograms. The bottom figure is an exaggerated version of warping where much more insertions/deletions were used.

introduced a multiplicative noise on spectrogram with a value of +- 1%. Table 5 below shows the results of these techniques. We experimented with randomly varying step and window size of spectrograms during the training. Step size was varied between 8 and 12 ms and window size between 23 and 27 ms. This however gave mixed results and we did not include it in the final model. Figure 5 shows clear advantage of models with warp and spectrogram noise. The advantage of the model with varied step/window size is dubious.

Model	tst2010	tst2015	average
baseline[1]	26.5	21.87	24.15
warp	26.63	22.42	24.41
spec. noise	26.81	21.87	24.34
step/window	26.22	22.15	24.2

Table 5: Maximal BLEU scores on two tests sets. The last column is maximal average of two test sets for one checkpoint.

3.4 Synthetic Must_C and Librispeech data

Addition of synthetic *Must_C* translations improved *BLEU* score by over 0.35 *BLEU*. Addition of *Librispeech* corpus further improved *BLEU* by 0.35. Table 6 below shows the results.

[1]the same model as 2019 primary submission but trained for 1.2M steps which resulted with higher score than previously reported

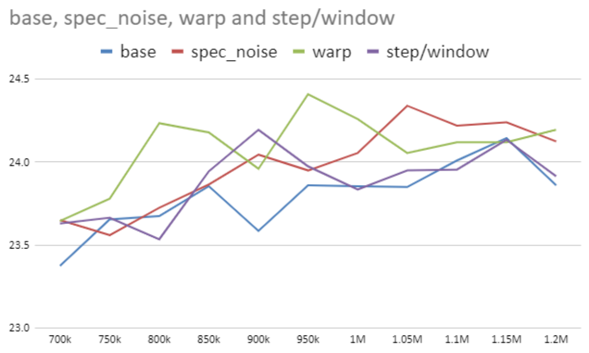

Figure 3: Comparison of warped and noised trainings to baseline. Y axis is average BLEU on tst2010 and tst2015. X axis is number of steps.

Model	tst2010	tst2015	average
base	26.5	21.87	24.15
synth_mc	27.08	22.04	24.5
+ libri	27.91	21.81	24.86

Table 6: Maximal BLEU scores on two test sets for *baseline*, additional *Must_C* synthetic targets and additional *Librispeech* data. The last column is maximal average of two tests set for one checkpoint.

3.5 12 layer encoder

Our experience with text translation suggests it is more efficient to increase number of layer in the encoder rather than decoder therefore we increased number of encoder layers to 12. Number of decoder layers stayed the same at 4 layers. This increased *BLEU* score by 1 point. Introducing warping, noise and Librispeech corpus increased *BLEU* by another 0.8 *BLEU*. Table 7 shows the results.

Model	tst2010	tst2015	average
base	26.5	21.87	24.15
12layer	27.48	22.92	25.14
12_wal	28.4	23.63	25.97

Table 7: Maximal BLEU scores on two test sets. Model *12_wal* is the model with 12 layers and trained on noised and warped data with *Librispeech* corpus. The last column is maximal average of two tests set for one checkpoint.

3.6 ResNet-like convolutional layers

Another improvement to our model is ResNet-like convolutional layers processing the spectrogram input. The idea is to make the convolutional layers deeper instead of using large number of channels. The spectrogram input is shrinked gradually in both axis using 2x2 pooling. As the spectrogram is shrinked channel, size is increasing. We start with a smaller channel size compared to the previous solution - 64 channels instead of 256 and end the convolutional processing with 256 channels. Figure 4 shows architectural diagram of our solution. Replacing previous architecture with ResNet improved the model by around 0.5 *BLEU*. Table 8 shows the results. Note that strictly maximal scores do not show the improvement well. Figure 5 shows a plot of the results which proofs stable advantage (around 0.5) of ResNet solution. We experimented with a version without the residual connections however decided not to include it in the final model.

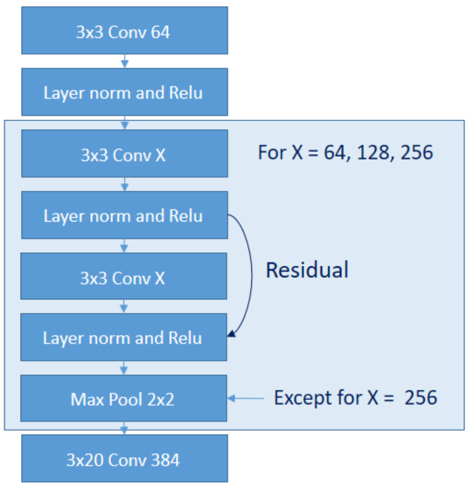

Figure 4: Convolutional layers architecture

Model	tst2010	tst2015	average
base	26.5	21.87	24.15
resnet	27.15	21.81	24.33
- residual	27.05	21.96	24.5

Table 8: Maximal BLEU scores on two test sets. The third row is same as ResNet model but without residual connections. The last column is maximal average of two tests set for one checkpoint.

3.7 Training process

We trained our models on 4 GTX 1080 Ti GPUs for about one and a half week, which resulted in 1.2M steps. *12layer* models were trained slightly longer - for 3 weeks because the training was slower. They were also trained on two NVIDIA Quadro 8000 GPUs because of its memory size. Batch size was 400000 timesteps for trainings on 1080 Ti and

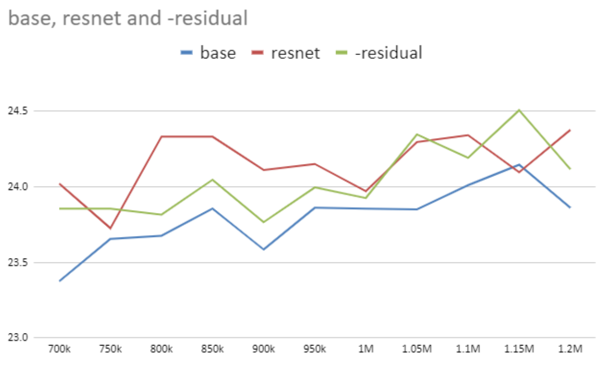

Figure 5: Comparison of Resnet trainings to baseline. Y axis is average BLEU on tst2010 and tst2015. X axis is number of steps.

3000000 timesteps on Quadro 8000. In the case of all trainings (except one) 10% dropout was applied. The final model with 12 layer and 512 embedding size used 20% dropout. Adam Multistep optimiser was used, effective batch size was increased 32 times.

3.8 Model averaging

For the final validation we averaged last 7 checkpoints of the training. Averaging checkpoints almost always resulted in higher *BLEU* scores. We experimented with continuation of training after averaging but it did not give any better results.

4 Final model

In this paper we presented improvements on simpler models. The final model uses a combination of all the promising techniques together with a larger embedding size. It would be unfeasible to perform all of these experiments on such a large model as its training took almost a month. However we are very satisfied with the end result which is 3 *BLEU* improvements compared to our baseline. The improvements from individual techniques cumulated very well.

Model	tst2010	tst2015	average
base	26.5	21.87	24.15
12_wal	28.4	23.63	25.97
+resnet[2]	28.26	23.77	26.02
+512 (final)	29.44	24.6	27.02

Table 9: Maximal BLEU scores on two test sets. The fourth row is our final submitted system. The last column is maximal average of two tests set for one checkpoint.

5 Segmentation

This year we used the same segmentation technique as last year. It relies on dividing the audio input densely using silence detection tool. These small fragments are then joined together up to a certain length depending on the length of the silence between them. Shorter distances between segments are joined earlier. This procedure is repeated until further joining results in segments longer than maximal length. Last year we determined such length should be 11s. However, for our current best model this distance turned out to be 15s. We used *tst2015* to optimize the process. We present the result in Table 10.

6 Evaluation

We have improved our score on *tst2019* by 4 *BLEU* compared to our last year's submission. It is importnat to notice the difference between given and our custom segmentation. Our method produces longer segments than the ones in the given segmentation. Our models seem to work much better on these longer segments giving around 3.9 *BLEU* higher scores.

tst2019	tst2019*	tst2020	tst2020*
20.1	23.96	21.49	25.3

Table 10: BLEU scores for our final model. * test sets use our custom segmentation

7 Conclusions

In this paper we have presented a significant improvement of translation quality of our end-to-end model. We have shown that despite limited parallel training data, end-to-end systems can compete with traditional pipeline systems. Using a longer segmentation, our model outscored the best IWSLT2019 pipeline system on *tst2019*(iws, 2019).

References

2019. *The IWSLT 2019 Evaluation Campaign*. Zenodo.

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. *arXiv preprint arXiv:1802.06655*.

[2]trained for shorter time than the other two

Ebrahim Ansari, Amittai Axelrod, Nguyen Bach, Ondrej Bojar, Roldano Cattoni, Fahim Dalvi, Nadir Durrani, Marcello Federico, Christian Federmann, Jiatao Gu, Fei Huang, Kevin Knight, Xutai Ma, Ajay Nagesh, Matteo Negri, Jan Niehues, Juan Pino, Elizabeth Salesky, Xing Shi, Sebastian Stüker, Marco Turchi, and Changhan Wang. 2020. Findings of the IWSLT 2020 Evaluation Campaign. In *Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT 2020)*, Seattle, USA.

Mattia A Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019. 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 1 (Long and Short Papers)*, pages 2012–2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Ye Jia, Melvin Johnson, Wolfgang Macherey, Ron J Weiss, Yuan Cao, Chung-Cheng Chiu, Naveen Ari, Stella Laurenzo, and Yonghui Wu. 2019. Leveraging weakly supervised data to improve end-to-end speech-to-text translation. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7180–7184. IEEE.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. 2015. Librispeech: An asr corpus based on public domain audio books. *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5206–5210.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *arXiv preprint arXiv:1904.08779*.

Tomasz Potapczyk, Pawel Przybysz, Marcin Chochowski, and Artur Szumaczuk. 2019. Samsung's system for the iwslt 2019 end-to-end speech translation task. In *Proceedings of the 16th International Conference on Spoken Language Translation (IWSLT 2019)*.

Anthony Rousseau, Paul Deléglise, and Yannick Esteve. 2014. Enhancing the ted-lium corpus with selected data for language modeling and more ted talks. In *LREC*, pages 3935–3939.

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 *Advances in neural information processing systems*, pages 5998–6008.