Start-Before-End and End-to-End: Neural Speech Translation by AppTek and RWTH Aachen University

Parnia Bahar*, **Patrick Wilken**, **Tamer Alkhouli**, **Andreas Guta**, **Pavel Golik**, **Evgeny Matusov**, **Christian Herold***
Applications Technology (AppTek), Aachen, Germany
{pbahar,pwilken,talkhouli,aguta,pgolik,ematusov,cherold}@apptek.com
*Also RWTH Aachen University, Germany

Abstract

AppTek and RWTH Aachen University team together to participate in the offline and simultaneous speech translation tracks of IWSLT 2020. For the offline task, we create both cascaded and end-to-end speech translation systems, paying attention to careful data selection and weighting. In the cascaded approach, we combine high-quality hybrid automatic speech recognition (ASR) with the Transformer-based neural machine translation (NMT). Our end-to-end direct speech translation systems benefit from pretraining of adapted encoder and decoder components, as well as synthetic data and fine-tuning and thus are able to compete with cascaded systems in terms of MT quality. For simultaneous translation, we utilize a novel architecture that makes dynamic decisions, learned from parallel data, to determine when to continue feeding on input or generate output words. Experiments with speech and text input show that even at low latency this architecture leads to superior translation results.

1 Introduction

When developing English→German speech translation systems for the IWSLT 2020 evaluation, we had the following goals:

- To obtain the best possible translation quality with the baseline cascaded approach. This includes data filtering, weighting, and domain adaptation for the MT component, hybrid ASR (Section 2.1) with a strong recurrent language model (LM) for the ASR component, and a preprocessing scheme that converts the written English source text into spoken forms with hand-crafted rules for numbers, dates, abbreviations, etc. (Section 2.2).
- Starting from the best cascaded system for text and speech input in terms of data composition, to design and implement an architecture that obtains the best possible translation quality for simultaneous speech translation at different levels of latency, learning a flexible read/output strategy from the underlying linguistic qualities of aligned parallel data. Our simultaneous translation approach is described in Section 3.
- For the end-to-end direct speech translation, to benefit as much as possible from the model components of the cascaded approach, including pre-training encoder/decoder parts, an adapter component, and using synthetic data at different levels (see Section 4), and try to obtain translation quality that reaches the level of our best cascaded approach.

Traditionally, RWTH/AppTek can train strong attention-based LSTM models, which still compete on-par with Transformer-based architectures on some language pairs and translation tasks. Therefore, we train both LSTM and Transformer *base* and *big* models (Vaswani et al., 2017). For the simultaneous translation task, we choose LSTM models for their simpler architecture that allows for an easier modification of the encoder and decoder process to partial input and prediction of chunk boundaries, as will be discussed in Section 3. For the offline translation tasks, our final submissions are ensembles of different encoder-decoder architectures, as well as ensembles of cascaded and end-to-end direct speech translation systems.

2 Cascaded Speech Translation

2.1 Automatic Speech Recognition

Our ASR systems are based on hybrid LSTM/HMM model (Bourlard and Wellekens, 1989; Hochreiter and Schmidhuber, 1997) and attention models (Bahdanau et al., 2015).

2.1.1 Hybrid LSTM/HMM model

The acoustic model has been trained on a total of approx. 2300 hours of transcribed speech including

Proceedings of the 17th International Conference on Spoken Language Translation (IWSLT), pages 44–54

https://doi.org/10.18653/v1/P17

EuroParl, How2, MuST-C, TED-LIUM (excluding the black-listed talks), LibriSpeech, Mozilla Common Voice, and IWSLT TED corpora.

As described in (Matusov et al., 2018), we apply an automatic re-alignment process to improve the quality of the TED talk segmentations. We use the TED-LIUM pronunciation lexicon. The acoustic model takes 80-dim. MFCC features as input and estimates state posterior probabilities for 5K tied triphone states. It consists of 4 bi-directional (BiLSTM) layers with 512 units for each direction. Frame-level alignment and state tying are obtained from a bootstrap model based on a Gaussian mixture acoustic model. We train the network for 10 epochs using the Adam update rule (Kingma and Ba, 2015) with Nesterov momentum and reducing the learning rate using the Newbob scheme.

The baseline language model is a simple 4-gram count model trained with Kneser-Ney smoothing on all allowed English text data (approx. 2.8B running words). The vocabulary consists of the same 152k words from the training lexicon and the out-of-vocabulary rate is far below 1%.

In addition, we train a neural LM with noise contrastive estimation (NCE) loss (Gutmann and Hyvärinen, 2010). The model estimates the distribution over the full vocabulary given the unconstrained history starting from the sentence begin. It learns 128-dim. word embeddings that are processed by two LSTM layers with 2048 units each. The output of the second LSTM layer is projected by a linear bottleneck layer onto 512 dimensions. We use the frequency sorted log-uniform distribution to sample 1024 negative examples for NCE loss calculation. This training approach results in a self-normalized model (Gerstenberger et al., 2020), which allows for an efficient, single-pass decoding with the neural LM (Beck et al., 2019).

The streaming recognizer implements a version of chunked processing (Chen and Huo, 2016; Zeyer et al., 2016), which allows to use the same BiLSTM-based acoustic model in both offline and online speech translation applications.

2.1.2 Attention Model

Following the work of LSTM-based attention ASR models (Zeyer et al., 2019), we apply a 6-layer BiLSTM encoder of 1024 nodes with interleaved max-pooling resulting in a total time reduction factor of 6 and a 1-layer LSTM decoder with a size of 1024 equipped with a single-head additive attention. We use a variant of SpectAugment (Park et al., 2019) for data augmentation. A layer-wise pre-training strategy similar to (Zeyer et al., 2018b) is applied during training for a more stable and faster initial convergence. We start with a small encoder (small in depth and width, i.e. number of layers and hidden dimensions) and then grow it over time. It means, we add layer by layer till the 6th layer, and increase the dimension till 1024 nodes. With each pre-training epoch, we grow the network in terms of both the number of layers and the number of hidden dimensions. Moreover, connectionist temporal classification (CTC) (Graves et al., 2006) as an additional loss is used on top of the speech encoder during training.

The models are trained using the Adam optimizer, dropout probability of 0.1 and label smoothing. We employ a learning rate scheduling scheme with a decay factor in the range of 0.8 to 0.9 based on perplexity on the development set. We apply byte-pair-encoding (BPE) (Sennrich et al., 2016b) with 5k merge operations with a dropout of 0.1. The beam size of 12 is used during the search without an extra language model. To enable the pre-training of the components, the same architecture is used in the speech encoder side of our direct speech translation models.

2.2 Written-to-Spoken Text Conversion

The large majority of MT parallel data comes from text sources and thus includes punctuation marks, digits, and special symbols. We apply additional preprocessing to the English side of the data to make it look like speech transcripts produced by the ASR system. We lowercase the text, remove all punctuation marks, expand common abbreviations, especially for measurement units, and convert numbers, dates, and other entities expressed with digits into their spoken form. For the cases of multiple readings of a given number (e.g. "one oh one" and "one hundred and one"), we select one randomly, so that the system can learn to convert alternative readings in English to the same number expressed with digits in German. Because of this preprocessing, our MT systems learn to insert punctuation marks, restore word case, and convert spoken number and entity forms to digits as part of the translation process. The same preprocessing is applied to the English monolingual data that is used in language model training of the ASR system.

2.3 Data Filtering and Domain Adaptation

For NMT training, we utilize the parallel data allowed for the IWSLT 2020 evaluation. We divide it into three parts: in-domain, clean, and out-of-

domain. We consider data from the TED and MuST-C corpora as in-domain and use it for subsequent fine-tuning experiments, as well as the "ground truth" for filtering the out-of-domain data based on sentence embedding similarity with the in-domain data. As "clean" we consider the News-Commentary, Europarl, and WikiTitles corpora and use their full versions in training.

To reduce the size of the training data, we apply a filtering approach based on sentence similarity. We train monolingual GloVe word embeddings (Pennington et al., 2014) both on the source and the target side of the data. Following Arora et al. (2017) we use a weighted average over the word embeddings of a sentence to generate a fixed-size sentence embedding. To obtain a sentence *pair* embedding, we concatenate the source and target sentence embedding of each bilingual sentence pair. Afterwards we employ k-Means clustering from the scikit-learn toolkit (Pedregosa et al., 2011) in the sentence pair embedding space.

After obtaining a set of clusters, we use the in-domain data to determine which clusters should be used for training. This is done by selecting all clusters which contain a non-negligible portion of the in-domain data using a fixed threshold n. We apply this technique to the noisy and out-of-domain corpora, namely ParaCrawl, CommonCrawl, rapid and OpenSubtitles. With the tuned threshold $n = 5.0\%$ we achieve a data reduction of around 45% (from 42.5M to 23.3M lines) and an improvement in the system performance of 1.6 % BLEU on the development set (from 30.7% to 32.3% BLEU).

A similar approach is applied to the German monolingual data allowed by the IWSLT 2020 evaluation that we incorporate into the MT training using back-translation (Sennrich et al., 2016a). First, from the billions of words of allowed text data we extract only sentence portions of at least four words which are enclosed in quotes. Especially in the news texts, these often represent quoted speech and thus may be more suitable to be used in training of speech NMT systems. Then, we apply the monolingual variant of the sentence embedding similarity approach described above to select 7.9M sentences. To create the synthetic parallel data, we translate these sentences into English with a De-En NMT Transformer base model that is trained on the in-domain and clean parallel data.

2.4 Neural Machine Translation

We employ the *base* and *big* Transformer model with multi-head attention. The base Transformer model consists of a self-attentive encoder and decoder, each of which is composed of 6 stacked layers. Every layer consists of two sub-layers: a 8-head self-attention layer followed by a rectified linear unit (ReLU). We apply layer normalization (Ba et al., 2016) before and dropout (Srivastava et al., 2014) and residual connections (He et al., 2016) after each sub-layer. All projection and multi-head attention layers consist of 512 nodes followed by a feed-forward layer equipped with 2048 nodes.

In comparison, the architecture of the big Transformer model incorporates 16-head self-attention sub-layers. Furthermore, all projection and attention layers consist of 1024 nodes and each feed-forward layer consists of 4096 nodes.

All models are trained on a single GPU and increased the effective batch size by accumulating gradient updates before applying them with a factor of 2 and 8 for the base and big Transformer respectively. All models are trained using Adam optimizer with an initial learning rate of 0.0003 and 1M lines per checkpoint. We apply a learning rate scheduling based on the perplexity on the validation set for a few consecutive evaluation checkpoints. Label smoothing (Pereyra et al., 2017) and dropout rates of 0.1 are used. The source and target sentences are segmented into subwords using SentencePiece (SP) (Kudo and Richardson, 2018) with a vocabulary size of 20K and 30K respectively.

3 Simultaneous Translation

In simultaneous translation a stream of source words is translated into a stream of target words without relying on the context of a full sentence. In this process, the system has to make decisions on when to read further input and when to produce partial translations. Hence, there is an inherent compromise between latency and MT quality.

3.1 Alignment-based Chunking

We develop a novel model architecture, based on offline LSTM models which are similar to Bahdanau et al. (2015). The approach is described in full detail in Wilken et al. (2020). Our model consists of a multi-layer BiLSTM encoder, a unidirectional decoder and an attention mechanism. We expand the forward encoder with an additional binary output trained to predict chunk boundaries in the incoming source word stream. These chunk boundaries mark positions where enough context for translation is present to trigger a translation. We generate training examples for such chunks based on sta-

tistical word alignment, created using the Eflomal Toolkit (Östling and Tiedemann, 2016). The chunk sequence of a sentence pair is defined such that it is monotonic[1], no word in the chunk is aligned to a word outside the chunk, and chunks are of minimal size. By this, reordering happens only within the chunks, thus in terms of word alignment the source side of a chunk provides enough information to continue the partial translation monotonically.

We shift the extracted source boundaries by D positions to the right such that the first words after the actual boundary provide context for the boundary detection component. Furthermore, we improve the chunk extraction described above by removing a chunk boundary if the target word following it is important as context for translation of the last word in the candidate chunk. Details are given in (Wilken et al., 2020). The words in the chunks are converted to SP subword sequences prior to the training of the simultaneous NMT system.

3.2 Streaming ASR

For the speech-to-text condition we use the cascaded approach, integrating the streaming version of the ASR system described in Section 2.1 into the decoder. We send 1-second chunks of the incoming audio into the ASR system. We have to alter the ASR system to output the common prefix of all hypothesized transcriptions in the beam, such that words in the output are guaranteed to not change due to further evidence. For each 1-second chunk we check whether new words were generated by the ASR. If so, we pass them to the encoder of the MT system. From that point on, translation happens as described in the next section.

3.3 Online MT Decoding

For each word in the input stream, we first apply subword splitting. Then we feed the subwords into the forward encoder one by one, producing the encoding of that subword and a boundary decision. If a boundary is predicted, all source words of the current chunk are fed into the backwards encoder. After that, the decoder produces the translation attending to the forward and backwards encodings of all words of the sentence read so far. Here, we perform the beam search with a beam size of 12. For length normalization, we divide the scores by $I^{0.9}$, I being the target length. To know when to stop decoding of a chunk, we predict the target chunk boundaries via a binary translation factor (Wilken and Matusov, 2019). A hypothesis in the beam is considered final as soon as a boundary is predicted. The states of the forward encoder and the decoder are kept across chunks. The backward encoder is initialized for each chunk. In both encoder and decoder we feed an embedding of the boundary decision into the next recurrent step, analogous to label feedback of the target word.

[1]Given a pair of subsequent chunks, the first word of the second chunk immediately follows the last word of the first chunk on the source and target side.

Training data\Running words	EN	DE
DST	7.5M	8.1M
ASR[1]	32.9M	-
MT[2]	309.8M	289.9M
SYNTH_SPEECH[3]	4.2M	5.0M
SYNTH_TRANS[4]	32.9M	37.3M
BT[5]	125.2M	117.3M

Table 1: Data size. [1]Contains the ASR portion of DST data; [2]contains the MT supervised data of DST data; [3]additional synthetic DST data by synthesizing bilingual MT data (using TTS model); [4]additional synthetic DST data by translation ASR transcriptions; [5]back-translation of German monolingual data.

4 End-to-End Direct Speech Translation

The direct speech translation models have been trained using direct speech translation (DST) training data including MuST-C, IWSLT TED, and EuroParl corpora, i.e. a total of approx. 420 hours of transcribed and translated speech (see Table 1). We remove all sequences longer than 75 tokens and all utterances longer than 6000 frames.

The end-to-end models are based on encoder-decoder architectures. The LSTM-based *speech encoder* uses 6 stacked BiLSTM layers with interleaved max-pooling layers in between to reduce the utterance length with a factor of 6. We apply layer-wise encoder pre-training w.r.t. both the number of layers and dimensions. The CTC loss is used on top of speech encoder except in pre-training. All other parameters are similar to ASR training; thus, we also apply SpectAugment in all of our DST experiments similar to (Bahar et al., 2019b).

The *text decoder* is based on the decoder of MT models, as illustrated in Figure 1, using either the LSTM or the Transformer topology. In LSTM setups, the decoder is equipped with a 1-layer unidirectional LSTM with cell size 1024 and single-head additive attention. All tokens are mapped into a 512-dimensional embedding space. Both base and big Transformer decoders are based on the architecture explained in Section 2.4.

To solve the data sparseness problems of DST

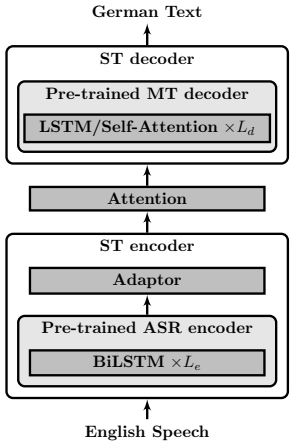

Figure 1: Overview of the DST model with pretraining and an adaptor. Shallow grey blocks correspond to pre-trained components, and dark grey blocks are fine-tuned on the DST task.

training, we explore various strategies to augment the data by leveraging weakly supervised data, i.e. ASR and MT training data. Our high-quality Transformer *big* model has been employed to generate synthetic DST training data by automatically translating the correct transcripts of ASR training data (Jia et al., 2019). SYNTH_TRANS refers to machine-translated ASR training data. As listed in Table 1, we translate the whole ASR training data (32.9M words) resulting in 37.3M German tokens and combine it with the original DST data, weighting each set equally. Similarly, we create synthetic DST training data by generating speech from the source side of an MT parallel corpus (Jia et al., 2019). We refer to it as SYNTH_SPEECH, and its statistics can be found in Table 1. Our text-to-speech synthesis (TTS) model is trained on ASR LibriSpeech dataset as described in (Rossenbach et al., 2020). Using the TTS model, we synthesize 800k random samples (total of 5M words as listed in Table 1) from the OpenSubtitles corpus pre-filtered as described in Section 2.3. Again, the generated data is uniformly mixed with the original DST data.

To further leverage the weakly supervised data, we apply pre-training of both the encoder and decoder with an adaptor layer in between. Initialization of model components using pre-trained ASR and MT models is a common transfer learning strategy to reduce dependency on scarce DST training data. We pre-train the encoder using our ASR model explained in Section 2.1, and the decoder using our MT model, either the LSTM attention or Transformer, as described in Section 2.4. After initialization with pre-trained components, we fine-tune on the DST training data. As proposed in (Bahar et al., 2019a), in order to familiarize the pre-trained text decoder with the output of the pre-trained speech encoder, we insert an additional adaptor layer which is a BiLSTM layer between the encoder and decoder. We train the adaptor component jointly without freezing the parameters in the fine-tuning stage. An abstract overview is shown in Figure 1.

5 Experimental Results

In this section we report results for offline cascaded and direct speech translation, as well as for simultaneous NMT under various training data conditions.

Acoustic training of the baseline model and the HMM decoding have been performed with the RWTH ASR toolkit (Wiesler et al., 2014). All neural models have been built with RETURNN (Doetsch et al., 2017; Zeyer et al., 2018a) using Sisyphus framework (Peter et al., 2018).

The number of running words of all training corpora is presented in Table 1. The data used for training the NMT models is referred to as MT and contains the in-domain, clean, and filtered bilingual data as defined in Section 2.3. On the other hand, BT denotes the parallel data obtained through back-translating the filtered monolingual data (see also Section 2.3). When the concatenation of MT and BT is used for training, we over-sample the in-domain and clean part of MT 5 times. We remove transcriber comments and emulate the ASR output using the preprocessing described in Section 2.2.

As heldout tuning sets, we use the concatenation of the TED dev2010, tst2014, and MuST-C dev corpora. As heldout test data, we use TED tst2015, MuST-C tst-HE and MuST-C tst-COMMON.

We report case-sensitive BLEU (Papineni et al., 2002) and TER (Snover et al., 2006) scores. For simultaneous NMT, also the average lagging (AL) metric (Ma et al., 2019) is reported. To measure AL, we have integrated our online decoder into the server-client implementation of IWSLT 2020 within the fairseq framework (Ott et al., 2019).

5.1 ASR Quality

For training of the ASR component used in the cascaded approach, we first pool the data from all available corpora, removing utterances that can not be aligned using a baseline model trained on the IWSLT TED corpus, resulting in 2300h of aligned audio. The performance of the model trained on this data is shown in the first line of Table 2. To understand the contribution of the various data

#	Model		TED	MuST-C	MuST-C
	AM	LM	tst2015	tst-HE	tst-COM
Hybrid HMM					
1	LSTM	4-gram	8.7	10.5	13.1
2	LSTM	4-gram	11.1	9.4	11.5
3	LSTM	LSTM	9.6	7.5	9.9
Attention					
4	LSTM	None	6.9	7.7	10.6

Table 2: ASR word error rate results in [%].

sources, we train a model for each corpus. Based on the accuracy on the dev set we decide to exclude EuroParl and How2 data sets, as they appear to be the worst match for the target domain. The second line shows that fine-tuning on the "matched" subset (about 85% of the total training data) does not lead to a consistent reduction of WER. Still, we decide to proceed with this acoustic model, based on the experience with the single corpus experiments. Finally, switching to the neural LM (see Section 2.1.1) considerably improves the accuracy on the test sets shown in line 3. This final system is used in the cascaded translation approach. The attention ASR model described in Section 2.1.2 has been trained using 2300h meaning 32.9M words. As shown in Table 2, the performance of the LSTM model (line 4) is competitive to the hybrid HMM model. We use LSTM speech encoder for all of our direct ST modeling in pre-training.

5.2 ASR Output for MT Fine-Tuning

For cascaded speech translation, both offline and simultaneous, we apply fine-tuning on the DST corpora. with correct source transcripts. In addition, we augment this data with the MuST-C and TED tst2010 through tst2013 sets, the source side of which is generated using the hybrid HMM (see Table 2 line 2). All fine-tuning systems employ an initial learning rate of 0.0008. The simultaneous systems and the offline Transformer base model trained on the MT+BT data (see Table 1) are fine-tuned using 100k lines per checkpoint, whereas the other offline models use 1M lines per checkpoint.

5.3 Offline Speech Translation

The results for the offline speech translation systems are presented in Table 3. The first line shows the results obtained when translating the ground truth source text of the test sets with a Transformer base model trained on the MT data, thus eliminating potential speech recognition errors. The preprocessing on the source side emulates the ASR output by applying lower-casing, removing punctuation marks and removing transcriber comments.

Line 2 through 8 present the results of translating the output of the hybrid HMM ASR system (see Table 2 line 3). In comparison to the first line, we see a significant loss of up to 3.5% BLEU when translating the ASR output (line 2). Fine-tuning this model as described in Section 5.2 leads to a performance gain of up to 1.9% BLEU (line 3).

Furthermore, we train models on the MT+BT data (line 4 to 8). Although the Transformer base model in line 4 outperforms the corresponding model in line 2, applying fine-tuning (line 5) does not yield better performance than the fine-tuned model in line 3, which can be traced back to the over-sampled clean data. The big Transformer models in line 6 and 7 outperform the base models in lines 4 and 5, respectively.

Overall, the fine-tuned big model (line 7) performs better on tst2015 and tst-HE, whereas the fine-tuned base model trained without oversampling and back-translated data (line 3) performs better on tst-COMMON. Our final submission (line 8) consists of the ensemble of the fine-tuned models in line 5 and 7 and yields the best performance on average. The results obtained translating the output of the attention ASR system (see Table 2 line 4) using the ensemble of the two models (line 5 and 7) are listed in line 9.

5.4 Direct Speech Translation

The fourth block of Table 3 shows the results of direct speech translation where we do not rely on intermediate transcriptions. In the first set of experiments, our DST models are based on the LSTM attention architecture where both encoder and decoder are composed of LSTM units (line 10 to 12). The LSTM attention model outperforms the Transformer model. Again, pre-training the entire network (plus a BiLSTM layer as an adaptor in between) yields improvements of 2.9% BLEU and 4.3% TER on average across all test sets indicating that pre-training is an effective strategy to leverage the supervised ASR and MT training data in practice.

Augmenting ASR data with automatic translations (SYNTH_TRANS) shows slightly worse results (line 12), which might be due to domain mismatch. In line with our pure MT and ASR experiments, we combine our strong speech LSTM encoder with our powerful text decoder, i.e. big Transformer (lines 13 to 16). As shown, this combination provides additional gain over vanilla pre-trained models. These lines differ in terms of training data

#	System	TED tst2015 BLEU	TED tst2015 TER	MuST-C tst-HE BLEU	MuST-C tst-HE TER	MuST-C tst-COMMON BLEU	MuST-C tst-COMMON TER	Training data composition
Pure text MT								
1	Transformer base	**31.2**	**52.3**	**28.5**	**55.8**	**31.3**	**50.1**	MT
Cascaded hybrid ASR → MT								
2	Transformer base	29.0	56.6	26.3	58.9	27.8	54.7	MT+ASR
3	+ fine-tune	30.2	55.7	28.1	57.2	**29.7**	**53.1**	MT+ASR
4	Transformer base	29.8	56.1	27.2	57.8	28.3	54.9	(MT+BT)+ASR
5	+ fine-tune	30.1	55.7	28.2	56.7	28.8	55.7	(MT+BT)+ASR
6	Transformer big	30.5	55.2	27.9	56.7	28.7	54.6	(MT+BT)+ASR
7	+ fine-tune	**30.9**	**55.2**	**28.6**	**56.3**	28.8	55.5	(MT+BT)+ASR
8	Ensemble (5, 7)	30.9	55.2	28.7	56.4	29.7	54.5	(MT+BT)+ASR
Cascaded attention ASR → MT								
9	Ensemble (5, 7)	30.3	54.2	28.3	56.9	28.8	55.3	(MT+BT)+ASR
End2end Direct DST								
10	LSTM-attention	23.6	64.1	22.1	63.3	24.3	59.1	DST
11	+ pretraining	26.0	59.1	24.7	60.1	27.9	54.3	DST+ASR+MT
12	+ pretraining	25.0	61.0	24.3	60.3	26.7	55.7	(DST + SYNTH_TRANS)+ASR+MT
13	+ big Transformer decoder	26.4	58.2	24.6	59.3	**29.1**	**53.8**	DST+DST+MT
14	+ big Transformer decoder	26.1	58.6	**25.1**	**58.8**	28.7	53.8	DST+ASR+MT
15	+ big Transformer decoder	25.9	59.3	24.1	63.5	27.0	55.9	(DST + SYNTH_SPEECH)+ASR+MT
16	+ big Transformer decoder	**27.0**	**58.3**	25.1	61.3	27.3	55.8	(DST + SYNTH_TRANS)+ASR+MT
17	+ fine-tune	26.8	58.6	25.1	62.3	27.9	55.3	(DST + SYNTH_TRANS)+ASR+MT
18	Ensemble (13, 17)	27.2	57.9	25.5	60.7	29.4	53.3	
19	Ensemble (13, 15, 16, 17)	28.0	57.3	26.5	58.1	29.6	53.4	

Table 3: Offline speech translation results measured in BLEU [%] and TER [%].

which is used either for pre-training or for fine-tuning. For instance, in line 13, we use the ASR model trained on the DST (in-domain) data for pre-training the encoder whereas line 14 corresponds to the ASR model trained on all (in-domain and out-of-domain) ASR data. In lines 15 and 16, we use additional augmented data. In general, ASR data augmented with synthetic translations can help the model, while synthesized speech for the MT data is less effective and still performs worse than the model using DST data only (see lines 14, 15). Another aspect to consider is that the additional synthetic data we generate might be out-of-domain. Therefore, we fine-tune on top of generated data to mitigate the domain gap (line 17). This approach improves the results on the tst-COMMON set. In the end, to benefit from all data variations, we do an ensemble of models that outperforms all single ones.

With data augmentation, pre-training, fine-tuning, and careful architecture selection, a combination of LSTM encoder and big Transformer decoder, we obtain comparable results and even on par on tst-COMMON set and close the gap between the cascaded and the direct models.

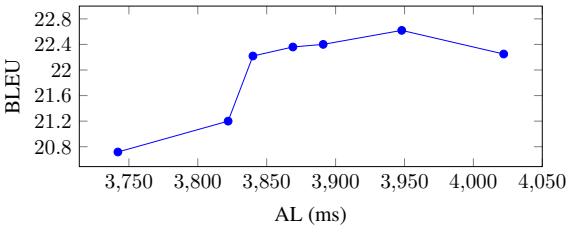

Figure 2: BLEU vs. Average Lagging latency for a unidirectional 6-encoder 2-decoder system, generated by varying the maximum chunk size using the values $C \in \{5, 6, 7, 8, 9, 10, 20\}$. The results are computed for the tst-HE dataset.

5.5 Simultaneous Speech Translation

We present results for simultaneous speech and text translation models fine-tuned on a concatenation of the MuST-C and TED training data. In the case of speech translation models, the fine-tuning is done as described in Section 5.2. All simultaneous models use 30k SP units for the source and target side.

Table 4 displays the results for the simultaneous speech translation task. In the upper part, we provide the results for the offline Transformer base system trained on the same data for reference. The results are shown for the reference transcript, and the streaming ASR output. In the middle of the table, we list multiple simultaneous NMT systems with varying settings. We enforce a maximum source-

System	TED tst2015 BLEU	TED tst2015 TER	MuST-C tst-HE BLEU	MuST-C tst-HE TER	MuST-C tst-HE AL	MuST-C tst-COMMON BLEU	MuST-C tst-COMMON TER
Offline baseline, Transformer (MT+BT training data)							
using reference transcript	32.7	50.9	30.1	54.3		32.6	48.9
using streaming ASR	28.6	56.3	26.0	59.2		26.4	57.3
Simultaneous NMT (AL $\leq$ 4s)							
6enc, 2dec, C=10, D=2	24.3	60.8	22.6	63.1	3.95s	22.4	60.2
6enc, 4dec, C=10, D=2	25.1	59.5	22.2	63.1	3.94s	22.3	60.0
6enc, 2dec, C=6, D=3	23.3	62.1	21.9	64.7	3.98s	22.3	61.3
2x4enc, 1dec, C=6, D=3	23.8	60.9	22.3	63.1	3.99s	22.3	61.0
6enc, 2dec, C=20, D=4	24.9	61.2	23.0	63.0	4.45s	22.1	62.0

Table 4: Experimental results (in %) for simultaneous NMT of speech, IWSLT 2020 English→German. C refers to the enforced maximum chunk size, D indicates the boundary decision delay.

System	Avg. AL	TED tst2015 BLEU	TED tst2015 TER	MuST-C tst-HE BLEU	MuST-C tst-HE TER	MuST-C tst-COMMON BLEU	MuST-C tst-COMMON TER
Simultaneous NMT							
6enc, 2dec, D=2	4.55	30.5	52.5	29.0	54.6	30.3	50.4
6enc, 2dec, D=3	5.21	30.5	52.6	28.9	55.4	29.8	51.0
6enc, 2dec, D=4	5.99	30.3	52.7	29.1	54.0	30.5	50.3
2x4enc, 1dec, D=3	5.33	29.9	53.4	29.0	54.9	30.7	50.4

Table 5: Experimental results (in %) for simultaneous NMT of text input, IWSLT 2020 English→German, D indicates the boundary decision delay.

side chunk size C and vary the source boundary delay D to achieve a latency below 4 seconds on tst-HE. We compare a unidirectional architecture of 6 LSTM encoder layers and 2 or 4 LSTM decoder layers to a bidirectional model. The model has two stacks of 4 forward and 4 backward LSTM encoder layers, concatenated at the top-most layer. The model uses 1 LSTM decoder layer. We observe that training with a lower delay D=2 and relaxing the maximum chunk size (C=10) produces better results than training with a larger delay (D=3), and using a smaller (C=6). The lower row shows results for a system with C=20, achieving a latency of 4.45 seconds. We note that the model makes dynamic decisions to decide on the source chunk boundaries that directly influence the latency.

Table 5 shows the results for simultaneous text translation. We compare unidirectional and bidirectional models with different latencies. All models use a fixed maximum chunk size of C=20. The models are trained with different delay values. We observe that even with a delay D=2 the model is able to learn reasonable chunk boundaries that achieve lower latency than higher-delay models and also maintain a comparable performance.

Figure 2 illustrates the performance on tst-HE against average lagging (AL) latency. The latency is varied by changing the maximum chunk size C. The model used is a unidirectional 6-encoder 2-decoder model trained with delay D=2. We observe little improvement when increasing the maximum chunk size beyond C=7. At C=7, AL is equal to 3.84s with a performance of 22.2% BLEU, comparable to 22.3% BLEU obtained when setting C=20 (corresponding to AL=4.02s). This is likely due to the learned chunking that is able to set the boundaries without the need for external intervention by capping the chunk size. On the other hand, reducing the maximum chunk size to 5 and 6 tokens reduces latency, but also reduces translation context and therefore hurts performance.

6 Final Results

Compared to last year's submission, the results of both cascade and direct offline speech translation models have improved. The cascade system shows an improvement of 2.0% BLEU compared to the 2018 submission. The MT quality of the direct model almost reached the one of the cascade model, obtaining a huge improvement of 12.4% BLEU. The performance on the tst2019 and tst2020 test sets is shown in Table 6, as evaluated by the IWSLT 2020 server. Our primary cascade and direct systems correspond to the lines 8 and 19 of Table 3 respectively. The contrastive systems which are single models correspond to the lines 7 and 17 of the table. We see that the provided reference segmentation negatively affects the MT quality. In contrast, the segmentation obtained by our hybrid ASR model yields segments which apparently are more sentence-like, include less noise and thus can be

better translated. On the condition with automatic segmentation, the difference between our cascade and direct models ranges from 1.8 to 2.3 BLEU points. This holds both for our primary ensemble submission and the contrastive single systems, which have lower BLEU scores by 1% or less as compared to the ensembles. More results can be found in (Ansari et al., 2020).

System	TED tst2019		TED tst2020	
	BLEU	TER	BLEU	TER
reference segmentation				
cascade (primary)	21.0	67.2	22.5	65.2
direct (primary)	19.2	71.2	20.5	70.1
automatic segmentation				
cascade (primary, ensemble)	23.4	63.5	25.1	61.4
direct (primary, ensemble)	21.6	66.2	23.3	64.8
cascade (contrastive, single)	23.2	63.6	24.6	61.9
direct (contrastive, single)	20.9	67.2	22.3	66.5

Table 6: AppTek/RWTH IWSLT 2020 submission for offline speech translation, BLEU and TER scores in %.

7 Conclusions

In this paper, we summarize the results of the joint participation of AppTek and RWTH Aachen University in the IWSLT 2020 evaluation. For the first time, we present simultaneous translation results on real speech from our hybrid streaming ASR system. With a latency of 4 seconds they are only 4 BLEU points behind our strong cascaded offline NMT baseline. This baseline still exhibits the best results in the offline speech translation task, but our direct single end-to-end system, with careful architecture selection, pre-training, and data augmentation, is almost able to compete with our best cascaded system, obtaining a BLEU score of 29.1 vs 29.7% on MuST-C tst-COMMON set. On the TED tst2015 set, the ensemble of our direct end-to-end systems yields a BLEU score of 28.0%, exactly reaching AppTek's cascaded system results at IWSLT 2018, obtained one and a half years ago. At that time, our first DST prototype scored only 17.1% BLEU on the same test set. This shows the fast and tremendous progress of our direct speech translation research.

References

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.

Sanjeev Arora, Yingyu Liang, and Tengyu Ma. 2017. A simple but tough-to-beat baseline for sentence embeddings. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*. Version 1.

Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019a. A comparative study on end-to-end speech to text translation. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 792–799, Sentosa, Singapore.

Parnia Bahar, Albert Zeyer, Ralf Schlüter, and Hermann Ney. 2019b. On using specaugment for end-to-end speech translation. In *International Workshop on Spoken Language Translation*, Hong Kong, China.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Eugen Beck, Wei Zhou, Ralf Schlüter, and Hermann Ney. 2019. LSTM language models for LVCSR in first-pass decoding and lattice-rescoring. Https://arxiv.org/abs/1907.01030.

Hervé Bourlard and Christian J. Wellekens. 1989. Links between Markov models and multilayer perceptrons. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems I*, pages 502–510. Morgan Kaufmann, San Mateo, CA, USA.

Kai Chen and Qiang Huo. 2016. Training deep bidirectional LSTM acoustic model for LVCSR by a context-sensitive-chunk BPTT approach. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(7):1185–1193.

Patrick Doetsch, Albert Zeyer, Paul Voigtlaender, Ilia Kulikov, Ralf Schlüter, and Hermann Ney. 2017. Returnn: the rwth extensible training framework for universal recurrent neural networks. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 5345–5349, New Orleans, LA, USA.

Alexander Gerstenberger, Kazuki Irie, Pavel Golik, Eugen Beck, and Hermann Ney. 2020. Domain robust, fast, and compact neural language models. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 7954–7958, Barcelona, Spain.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29*, pages 369–376.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9, pages 297–304, Chia Laguna Resort, Sardinia, Italy.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

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 *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 7180–7184. IEEE.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, San Diego, CA, USA.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Mingbo Ma, Liang Huang, Hao Xiong, Renjie Zheng, Kaibo Liu, Baigong Zheng, Chuanqiang Zhang, Zhongjun He, Hairong Liu, Xing Li, Hua Wu, and Haifeng Wang. 2019. STACL: Simultaneous translation with implicit anticipation and controllable latency using prefix-to-prefix framework. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3025–3036, Florence, Italy. Association for Computational Linguistics.

Evgeny Matusov, Patrick Wilken, Parnia Bahar, Julian Schamper, Pavel Golik, Albert Zeyer, Joan Albert Silvestre-Cerda, Adria Martinez-Villaronga, Hendrik Pesch, and Jan-Thorsten Peter. 2018. Neural speech translation at apptek. In *Proceedings of the 15th International Workshop on Spoken Language Translation*, pages 104–111, Bruges, Belgium.

Robert Östling and Jörg Tiedemann. 2016. Efficient word alignment with Markov Chain Monte Carlo. *Prague Bulletin of Mathematical Linguistics*, 106:125–146.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA.

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*.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake VanderPlas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. 2011. Scikit-learn: Machine learning in python. *J. Mach. Learn. Res.*, 12:2825–2830.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. *CoRR*, abs/1701.06548.

Jan-Thorsten Peter, Eugen Beck, and Hermann Ney. 2018. Sisyphus, a workflow manager designed for machine translation and automatic speech recognition. In *Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium.

Nick Rossenbach, Albert Zeyer, Ralf Schlüter, and Hermann Ney. 2020. Generating synthetic audio data for attention-based speech recognition systems. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016a. Improving neural machine translation models with monolingual data. pages 86–96.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas*, pages 223–231, Cambridge, Massachusetts, USA.

Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

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.

Simon Wiesler, Alexander Richard, Pavel Golik, Ralf Schlüter, and Hermann Ney. 2014. RASR/NN: The RWTH neural network toolkit for speech recognition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 3313–3317, Florence, Italy.

Patrick Wilken, Tamer Alkhouli, Evgeny Matusov, and Pavel Golik. 2020. Neural simultaneous speech translation using alignment-based chunking. In *International Workshop on Spoken Language Translation*.

Patrick Wilken and Evgeny Matusov. 2019. Novel applications of factored neural machine translation. *arXiv preprint arXiv:1910.03912*.

Albert Zeyer, Tamer Alkhouli, and Hermann Ney. 2018a. RETURNN as a generic flexible neural toolkit with application to translation and speech recognition. In *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, System Demonstrations*, pages 128–133.

Albert Zeyer, Parnia Bahar, Kazuki Irie, Ralf Schlüter, and Hermann Ney. 2019. A comparison of transformer and lstm encoder decoder models for asr. In *IEEE Automatic Speech Recognition and Understanding Workshop*, pages 8–15, Sentosa, Singapore.

Albert Zeyer, Kazuki Irie, Ralf Schlüter, and Hermann Ney. 2018b. Improved training of end-to-end attention models for speech recognition. In *19th Annual Conf. Interspeech, Hyderabad, India, 2-6 Sep.*, pages 7–11.

Albert Zeyer, Ralf Schlüter, and Hermann Ney. 2016. Towards online-recognition with deep bidirectional LSTM acoustic models. In *Interspeech*, pages 3424–3428, San Francisco, CA, USA.