CTC-based Compression for Direct Speech Translation

Marco Gaido[†, *], **Mauro Cettolo**[†], **Matteo Negri**[†], **Marco Turchi**[†]
[†]Fondazione Bruno Kessler
[*]University of Trento
{mgaido,cettolo,negri,turchi}@fbk.eu

Abstract

Previous studies demonstrated that a dynamic phone-informed compression of the input audio is beneficial for speech translation (ST). However, they required a dedicated model for phone recognition and did not test this solution for direct ST, in which a single model translates the input audio into the target language without intermediate representations. In this work, we propose the first method able to perform a dynamic compression of the input in direct ST models. In particular, we exploit the Connectionist Temporal Classification (CTC) to compress the input sequence according to its phonetic characteristics. Our experiments demonstrate that our solution brings a 1.3-1.5 BLEU improvement over a strong baseline on two language pairs (English-Italian and English-German), contextually reducing the memory footprint by more than 10%.

1 Introduction

Speech translation (ST) is the process that converts utterances in one language into text in another language. Traditional approaches to ST consist of separate modules, each dedicated to an easier subtask, which are eventually integrated in a so-called *cascade* architecture (Stentiford and Steer, 1988; Waibel et al., 1991). Usually, its main components are an automatic speech recognition (ASR) model - which generates the transcripts from the audio - and a machine translation (MT) model - which translates the transcripts into the target language. A newer approach is *direct* ST, in which a single model performs the whole task without intermediate representations (Bérard et al., 2016; Weiss et al., 2017). The main advantages of direct ST systems are: *i)* the access to information not present in the text (e.g. prosody, vocal characteristics of the speaker) during the translation phase, *ii)* a reduced latency, *iii)* a simpler and easier to manage architecture (only one model has to be maintained), which *iv)* avoids error propagation across components.

In both paradigms (cascade and direct), the audio is commonly represented as a sequence of vectors obtained with a Mel filter bank. These vectors are collected with a high frequency, typically one every 10 ms. The resulting sequences are much longer than the corresponding textual ones (usually by a factor of ~10). The sequence length is problematic both for RNN (Elman, 1990) and Transformer (Vaswani et al., 2017) architectures. Indeed, RNNs fail to represent long-range dependencies (Bengio et al., 1993) and the Transformer has a quadratic memory complexity in the input sequence length, which makes training on long sequences prohibitive due to its memory footprint. For this reason, architectures proposed for direct ST/ASR reduce the input length either with convolutional layers (Bérard et al., 2018; Di Gangi et al., 2019) or by stacking and downsampling consecutive samples (Sak et al., 2015). However, these fixed-length reductions of the input sequence assume that samples carry the same amount of information. This does not necessarily hold true, as phonetic features vary at a different speed in time and frequency in the audio signals.

Consequently, researchers have studied how to reduce the input length according to dynamic criteria based on the audio content. Salesky et al. (2019) demonstrated that a phoneme-based compression of the input frames yields significant gains compared to fixed length reduction. Phone-based and linguistically-informed compression also proved to be useful in the context of visually grounded speech (Havard et al., 2020). However, Salesky and Black (2020) questioned the approach, claiming that the addition of phone features without segmentation and compression of the input is more effective.

None of these works is a direct ST solution, as they all require a separate model for phone recog-

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics, pages 690–696
April 19 - 23, 2021. ©2021 Association for Computational Linguistics

nition and intermediate representations. So, they: *i)* are affected by *error propagation* (Salesky and Black 2020 show in fact that lower quality in phone recognition significantly degrades final ST performance), *ii)* have higher latency and *iii)* a more complex architecture. A direct model with phone-based multi-task training was introduced by Jia et al. (2019) for speech-to-speech translation, but they neither compared with a training using transcripts nor investigated dynamic compression.

In this paper, we explore the usage of phones and dynamic content-based input compression for direct ST (and ASR). Our goal is an input reduction that, limiting the amount of redundant/useless information, yields better performance and lower memory consumption at the same time. To this aim, we propose to exploit the Connectionist Temporal Classification (CTC) (Graves et al., 2006) to add phones prediction in a multi-task training and compress the sequence accordingly. To disentangle the contribution of the introduction of phone recognition and the compression based on it, we compare against similar trainings leveraging transcripts instead of phones. Our results show that phone-based multi-task training with sequence compression improves over a strong baseline by up to 1.5 BLEU points on two language pairs (English-German and English-Italian), with a memory footprint reduction of at least 10%.

2 CTC-based Sequence Compression

The CTC algorithm is usually employed for training a model to predict an output sequence of variable length that is shorter than the input one. This is the case of speech/phone recognition, as the input is a long sequence of audio samples, while the output is the sequence of uttered symbols (e.g. phones, sub-words), which is significantly shorter. In particular, for each time step, the CTC produces a probability distribution over the possible target labels augmented with a dedicated `<blank>` symbol representing the absence of a target value. These distributions are then exploited to compute the probabilities of different sequences, in which consecutive equal predictions are collapsed and `<blank>` symbols are removed. Finally, the resulting sequences are compared with the target sequence.

Adding an auxiliary CTC loss to the training of direct ST and acoustic ASR models has been shown to improve performance (Kim et al., 2017; Bahar et al., 2019). In these works, the CTC loss is computed against the transcripts on the encoder output to favour model convergence. Generally, the CTC loss can be added to the output of any encoder layer, as in Figure 1 where the hyper-parameter N_{CTC} indicates the number of the layer at which the CTC is computed. Formally, the final loss function is:

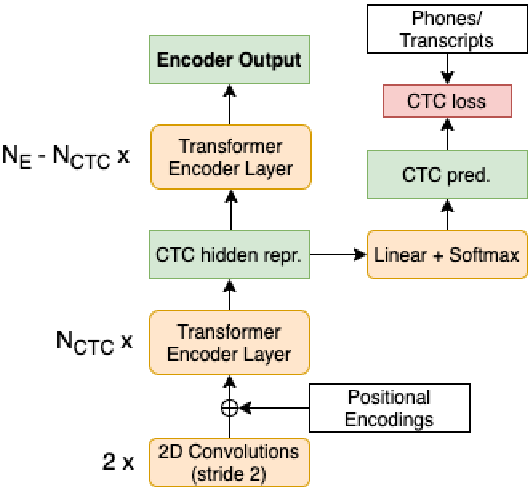

Figure 1: Encoder architecture with CTC loss.

$$\lambda = CTC(E_{\text{N}_{\text{CTC}}}) + CE(D_{\text{N}_{\text{D}}}) \qquad (1)$$

where E_{x} is the output of the x-th encoder layer, $D_{\text{N}_{\text{D}}}$ is the decoder output, CTC is the CTC function, and CE is the label smoothed cross entropy. If N_{CTC} is equal to the number of encoder layers (N_{E}), the CTC input is the encoder output. We consider this solution as our baseline and we also test it with phones as target.

As shown in Figure 1, we use as model a Transformer, whose encoder layers are preceded by two 2D convolutional layers that reduce the input size by a factor of 4. Therefore, the CTC produces a prediction every 4 input time frames. The sequence length reduction is necessary both because it makes possible the training (otherwise out of memory errors would occur) and to have a fair comparison with modern state-of-the-art models. A logarithmic distance penalty (Di Gangi et al., 2019) is added to all the Transformer encoder layers.

Our proposed architecture is represented in Figure 2. The difference with the baseline is the introduction of an additional block (*Collapse same predictions*) that exploits the CTC predictions to compress the input elements (vectors). Hence, in this case the CTC does not only help model convergence, but it also defines variable-length segments representing the same content. So, dense audio portions can be given more importance, while re-

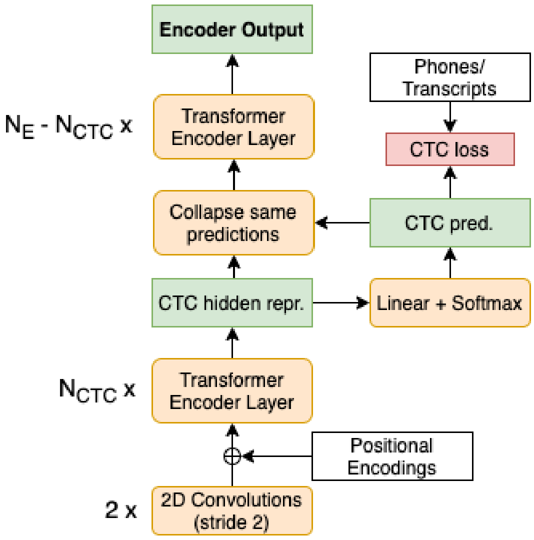

Figure 2: Encoder architecture with CTC compression.

dundant/uninformative vectors can be compressed. This allows the following encoder layers and the decoder to attend to useful information without being "distracted" by noisy elements. The architecture is a direct ST solution as there is a single model whose parameters are optimized together without intermediate representations. At inference time, the only input is the audio and the model produces the translation into the target language (contextually generating the transcripts/phones with the CTC).

We compare three techniques to compress the consecutive vectors with the same CTC prediction:

- **Average.** The vectors to be collapsed together are averaged. As there is only a linear layer between the CTC inputs and its predictions, the vectors in each group are likely to be similar, so the compression should not remove much information.

- **Weighted.** The vectors are averaged but the weight of each vector depends on the confidence (i.e. the predicted probability) of the CTC prediction. This solution is meant to give less importance to vectors whose phone/transcript is not certain.

- **Softmax.** In this case, the weight of each vector is obtained by computing the `softmax` of the CTC predicted probabilities. The idea is to propagate information (nearly) only through a single input vector (the more confident one) for each group.

3 Data

We experiment with MuST-C (Cattoni et al., 2021), a multilingual ST corpus built from TED talks. We focus on the English-Italian (465 hours) and English-German (408 hours) sections. For each set (train, validation, test), it contains the audio files, the transcripts, the translations and a YAML file with the start time and duration of the segments.

In addition, we extract the phones using Gentle.[1] Besides aligning the transcripts with the audio, Gentle returns the start and end time for each recognized word, together with the corresponding phones. For the words not recognized in the audio, Gentle does not provide the phones, so we lookup their phonetic transcription on the VoxForge[2] dictionary. For each sample in the corpus, we rely on the YAML file and the alignments generated by Gentle to get all the words (and phones) belonging to it. The phones have a suffix indicating the position in a word (at the end, at the beginning, in the middle or standalone). We also generated a version without the suffix (we refer to it as `PH W/O POS` in the rest of the paper). The resulting dictionaries contain respectively 144 and 48 symbols.

4 Experimental Settings

Our Transformer layers have 8 attention heads, 512 features for the attention and 2,048 hidden units in FFN. We set a 0.2 dropout and include *SpecAugment* (Park et al., 2019) in our trainings. We optimize label smoothed cross entropy (Szegedy et al., 2016) with 0.1 smoothing factor using Adam (Kingma and Ba, 2015) (betas *(0.9, 0.98)*). The learning rate increases linearly from 3e-4 to 5e-3 for 4,000 updates, then decays with the inverse square root. As we train on 8 GPUs with mini-batches of 8 sentences and we update the model every 8 steps, the resulting batch size is 512. The audio is pre-processed performing speaker normalization and extracting 40-channel Mel filter-bank features per frame. The text is tokenized into subwords with 1,000 BPE merge rules (Sennrich et al., 2016).

As having more encoder layers than decoder layers has been shown to be beneficial (Potapczyk and Przybysz, 2020; Gaido et al., 2020), we use 8 Transformer encoder layers and 6 decoder layers for ASR and 11 encoder and 4 decoder layers for ST unless stated otherwise. We train until the

[1] https://lowerquality.com/gentle/
[2] http://www.voxforge.org/home

model does not improve on the validation set for 5 epochs and we average the last 5 checkpoints. Trainings were performed on K80 GPUs and lasted ~48 hours (~50 minutes per epoch). Our implementation[3] is based on Fairseq (Ott et al., 2019).

We evaluate performance with WER for ASR and with BLEU (Papineni et al., 2002)[4] and SacreBLEU (Post, 2018)[5] for ST.

	WER (↓)	**RAM (MB)**
Baseline - 8L EN	16.0	6929 (1.00)
8L PH	**15.6**	6661 (0.96)
2L PH AVG	21.2	3375 (0.49)
4L PH AVG	17.5	4542 (0.66)
8L PH AVG	16.3	6286 (0.91)
8L PH W/O POS. AVG	16.4	6565 (0.95)
8L EN AVG	16.3	6068 (0.88)

Table 1: Results on ASR using the CTC loss with transcripts and phones as target. AVG indicates that sequence is compressed averaging the vectors.

5 Results

5.1 ASR

We first tested whether ASR benefits from the usage of phones and sequence compression. Table 1 shows that having phones instead of English transcripts (Baseline - 8L EN) as target of the CTC loss (8L PH) without compression is beneficial. When compressing the sequence, there is little difference according to the target used (8L PH AVG, 8L PH W/O POS. AVG, 8L EN AVG). However, the compression causes a 0.3-0.5 WER performance degradation and a 12-5% saving of RAM. Moving the compression to previous layers (4L PH AVG, 2L PH AVG) further decreases the output quality and the RAM usage. We can conclude that compressing the input sequence harms ASR performance, but might be useful if RAM usage is critical and should be traded off with performance.

5.2 Direct ST

In early experiments, we pre-trained the first 8 layers of the ST encoder with that of the ASR model, adding three *adapter* layers (Bahar et al., 2019). We realized that ASR pre-training was not useful (probably because ASR and ST data are the same), so we report results without pre-training.

As we want to ensure that our results are not biased by a poor baseline, we compare with (Di Gangi et al., 2020), which uses the same framework and similar settings.[6] As shown in Table 2, our strong baseline (8L EN) outperforms (Di Gangi et al., 2020) by 2 BLEU on en-it and 1.3 BLEU on en-de.

As in ASR, replacing the transcripts with phones as target for the CTC loss (8L PH) further improves respectively by 0.5 and 1.2 BLEU. We first explore the introduction of the compression at different layers. Adding it to the 8th layer (8L PH AVG) enhances the translation quality by 0.6 (en-it) and 0.2 (en-de) BLEU, with the improvement on en-it being statistically significant over the version without CTC compression. Moving it to previous layers (4L PH AVG, 2L PH AVG) causes performance drops, suggesting that many layers are needed to extract useful phonetic information.

Then, we compare the different compression policies: AVG outperforms (or matches) WEIGHTED and SOFTMAX on both languages. Indeed, the small weight these two methods assign to some vectors likely causes an information loss and prevents a proper gradient propagation for the corresponding input elements.

Finally, we experiment with different CTC targets, but both the phones without the position suffix (8L PH W/O POS. AVG) and the transcripts (8L EN AVG) lead to lower scores.

The different results between ASR and ST can be explained by the nature of the two tasks: extracting content knowledge is critical for ST but not for ASR, in which a compression can hide details that are not relevant to extrapolate meaning, but needed to generate precise transcripts. The RAM savings are higher than in ASR as there are 3 more layers. On the 8th layer, they range from 11% to 23% for en-it, 16% to 22% for en-de. By moving the compression to previous layers, we can trade performance for RAM requirements, saving up to 50% of the memory.

We also tested whether we can use the saved RAM to add more layers and improve the translation quality. We added 3 encoder and 2 decoder layers: this (8L PH AVG (14+6L)) results in

[3] Available at https://github.com/mgaido91/FBK-fairseq-ST/tree/eacl2021.

[4] To be comparable with previous works.

[5] The version signature is: BLEU+c.mixed+#.1+s.exp+tok.13a+v.1.4.3.

[6] We acknowledge that better results have been published in a contemporaneous paper by Inaguma et al. (2020). Besides the contemporaneity issue, our results are not comparable with theirs, as they use: *i)* a different architecture built on ESPnet-ST (a newer framework that, alone, outperforms Fairseq), *ii)* higher dimensional input features (83 vs 40 dimensions), *iii)* data augmentation, and *iv)* pre-training techniques.

	en-it			en-de		
	BLEU (↑)	**SacreBLEU (↑)**	**RAM (MB)**	**BLEU (↑)**	**SacreBLEU (↑)**	**RAM (MB)**
(Di Gangi et al., 2020)	20.1	-	-	19.1	-	-
Baseline - 8L EN	22.1	21.8	9624 (1.00)	20.4	20.5	9166 (1.00)
8L PH	22.6*	22.3*	9567 (0.99)	21.6*	21.6*	9190 (1.00)
2L PH AVG	20.2	20.0	5804 (0.60)	17.8	17.8	4484 (0.49)
4L PH AVG	21.6	21.3	6193 (0.64)	20.1	20.2	5186 (0.57)
8L PH AVG	*23.2*†	*22.8*†	8554 (0.89)	*21.8**	*21.9**	7348 (0.80)
8L PH WEIGHTED	22.7*	22.5*	7636 (0.79)	21.7*	21.8*	7380 (0.81)
8L PH SOFTMAX	22.6*	22.3*	7892 (0.82)	*21.8**	*21.9**	7436 (0.81)
8L PH W/O POS. AVG	22.2	22.0	7451 (0.77)	21.5*	21.6*	7274 (0.79)
8L EN AVG	22.2	21.9	8287 (0.86)	20.6	20.7	7143 (0.78)
8L PH AVG (14+6L)	**23.4**†	**23.2**†	8658 (0.90)	**21.9**†	**22.0**†	7719 (0.84)

Table 2: Results using the CTC loss with transcripts and phones as target. `AVG`, `WEIGHTED` and `SOFTMAX` indicate the compression method. If none is specified, no compression is performed. The symbol "*" indicates improvements that are statistically significant with respect to the baseline. "†" indicates statistically significant gains with respect to `8L PH`. Statistical significance is computed according to (Koehn, 2004) with $\alpha = 0.05$. Scores in *italic* indicate the best models among those with equal number of layers.

small gains (0.2 on en-it and 0.1 on en-de), but the additional memory required is also small (the RAM usage is still 10-16% lower than the baseline). The improvements are statistically significant with respect to the models without compression (`8L PH`) on both language pairs. When training on more data, the benefit of having deeper networks might be higher, though, and this solution allows to increase the number of layers without a prohibitive memory footprint. We leave this investigation for future works, as experiments on larger training corpora are out of the scope of this paper.

6 Conclusions

As researchers' focus is shifting from cascade to direct solutions due to the advantages of the latter, we proposed a technique of dynamic sequence-length reduction for direct ST. We showed that averaging the vectors corresponding to the same phone prediction according to the CTC improves the translation quality and reduces the memory footprint, allowing for training deeper models. Our best model outperforms a strong baseline, which uses transcripts in a multi-task training, by 1.3 (en-it) and 1.5 (en-de) BLEU, reducing memory usage by 10-16%.

Acknowledgments

This work is part of the "End-to-end Spoken Language Translation in Rich Data Conditions" project,[7] which is financially supported by an Amazon AWS ML Grant. The authors also wish to thank Mattia Antonino Di Gangi for the insightful discussions on this work.

References

Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019. A Comparative Study on End-to-end Speech to Text Translation. In *Proceedings of International Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 792–799, Sentosa, Singapore.

Yoshua Bengio, Paolo Frasconi, and Patrice Y. Simard. 1993. The problem of learning long-term dependencies in recurrent networks. In *IEEE International Conference on Neural Networks*, pages 1183–1188 vol.3.

Alexandre Bérard, Laurent Besacier, Ali Can Kocabiyikoglu, and Olivier Pietquin. 2018. End-to-End Automatic Speech Translation of Audiobooks. In *Proceedings of ICASSP 2018 - IEEE International Conference on Acoustics, Speech and Signal Processing*, Calgary, Alberta, Canada.

Alexandre Bérard, Olivier Pietquin, Christophe Servan, and Laurent Besacier. 2016. 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.

Roldano Cattoni, Mattia A. Di Gangi, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2021. MuST-C: A multilingual corpus for end-to-end speech translation. *Computer Speech & Language*, 66:101155.

Mattia A. Di Gangi, Marco Gaido, Matteo Negri, and Marco Turchi. 2020. On Target Segmentation for Direct Speech Translation. In *Proceedings of the 14th Conference of the Association for Machine Translation in the Americas (AMTA 2020)*, pages 137–150,

[7] `https://ict.fbk.eu/units-hlt-mt-e2eslt/`

Virtual. Association for Machine Translation in the Americas.

Mattia A. Di Gangi, Matteo Negri, Roldano Cattoni, Roberto Dessi, and Marco Turchi. 2019. Enhancing Transformer for End-to-end Speech-to-Text Translation. In *Proceedings of Machine Translation Summit XVII Volume 1: Research Track*, pages 21–31, Dublin, Ireland. European Association for Machine Translation.

Jeffrey L. Elman. 1990. Finding structure in time. *COGNITIVE SCIENCE*, 14(2):179–211.

Marco Gaido, Mattia A. Di Gangi, Matteo Negri, and Marco Turchi. 2020. End-to-End Speech-Translation with Knowledge Distillation: FBK@IWSLT2020. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 80–88, Online. Association for Computational Linguistics.

Alex Graves, Santiago Fernández, Faustino J. Gomez, and Jürgen Schmidhuber. 2006. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *Proceedings of the 23rd international conference on Machine learning (ICML)*, pages 369–376, Pittsburgh, Pennsylvania.

William N. Havard, Jean-Pierre Chevrot, and Laurent Besacier. 2020. Catplayinginthesnow: Impact of Prior Segmentation on a Model of Visually Grounded Speech.

Hirofumi Inaguma, Shun Kiyono, Kevin Duh, Shigeki Karita, Nelson Yalta, Tomoki Hayashi, and Shinji Watanabe. 2020. ESPnet-ST: All-in-One Speech Translation Toolkit. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 302–311, Online. Association for Computational Linguistics.

Ye Jia, Ron J. Weiss, Fadi Biadsy, Wolfgang Macherey, Melvin Johnson, Zhifeng Chen, and Yonghui Wu. 2019. Direct Speech-to-Speech Translation with a Sequence-to-Sequence Model. In *Proceedings of Interspeech 2019*, pages 1123–1127, Graz, Austria.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4835–4839, New Orleans, Louisiana.

Diederik Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of 3rd International Conference on Learning Representations (ICLR)*, San Diego, California.

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain. Association for Computational Linguistics.

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 the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania. Association for Computational Linguistics.

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. In *Proceedings of Interspeech 2019*, pages 2613–2617, Graz, Austria.

Matt Post. 2018. A Call for Clarity in Reporting BLEU Scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Tomasz Potapczyk and Pawel Przybysz. 2020. SRPOL's system for the IWSLT 2020 end-to-end speech translation task. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 89–94, Online. Association for Computational Linguistics.

Haşim Sak, Andrew Senior, Kanishka Rao, and Françoise Beaufays. 2015. Fast and Accurate Recurrent Neural Network Acoustic Models for Speech Recognition. In *Proceedings of Interspeech 2015*, Dresden, Germany.

Elizabeth Salesky and Alan W. Black. 2020. Phone Features Improve Speech Translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2388–2397, Online. Association for Computational Linguistics.

Elizabeth Salesky, Matthias Sperber, and Alan W. Black. 2019. Exploring Phoneme-Level Speech Representations for End-to-End Speech Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1841, Florence, Italy. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational*

Linguistics (Volume 1: Long Papers), pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Frederick W. M. Stentiford and Martin G. Steer. 1988. Machine Translation of Speech. *British Telecom Technology Journal*, 6(2):116–122.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2818–2826, Las Vegas, Nevada, United States.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Proceedings of Advances in Neural Information Processing Systems 30 (NIPS)*, pages 5998–6008, Long Beach, California.

Alex Waibel, Ajay N. Jain, Arthur E. McNair, Hiroaki Saito, Alexander G. Hauptmann, and Joe Tebelskis. 1991. JANUS: A Speech-to-Speech Translation System Using Connectionist and Symbolic Processing Strategies. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing, ICASSP 1991*, pages 793–796, Toronto, Canada.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-Sequence Models Can Directly Translate Foreign Speech. In *Proceedings of Interspeech 2017*, pages 2625–2629, Stockholm, Sweden.