INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

Knowledge Distillation on Joint Task End-to-End Speech Translation

Khandokar Md. Nayem[§] , *Ran Xue*[†] , *Ching-Yun Chang*[†] , *Akshaya Vishnu Kudlu Shanbhogue*[†]

[§]Department of Computer Science, Indiana University, Bloomington, USA
[†]Amazon, Alexa AI, Cambridge, USA

knayem@iu.edu, {ranxue, cychang, ashanbho}@amazon.com

Abstract

An End-to-End Speech Translation (E2E-ST) model takes input audio in one language and directly produces output text in another language. The model requires to learn both speech-to-text modality conversion and translation tasks, which demands a large architecture for effective learning of this joint task. Yet, to the best of our knowledge, we are the first to optimize compression of E2E-ST models. In this work, we explore knowledge distillation for a cross-modality joint-task E2E-ST system from 3 dimensions: 1) student architecture and weight initialization scheme, 2) importance of loss terms associated with different tasks and data modalities, 3) knowledge distillation training scheme customized for the multi-task/module model. Comparing with the full size model, our compressed model's encoder and decoder size are 50% smaller, while it retains 90% and $> 95\%$ performance on speech translation task and machine translation task respectively on MUST-C en→de testset.

Index Terms: speech translation, knowledge distillation, joint task, deep learning

1. Introduction

Speech Translation (ST) allows for the conversion of spoken audio into written text in various languages. Typically, this is accomplished by using a cascaded system of Automatic Speech Recognition (ASR) and Text-to-Text Translation models. Recent End-to-End (E2E) models, like the FAIR Speech Translation (FAIR-ST) System [1], can perform ST task using a single model and produce similar results as the traditional cascaded approach [2, 3]. Compared to cascaded models, E2E-ST models offer faster inference, reduced compounding errors, and lower overall system complexity [4, 5].

Recently, transformer-based E2E-ST models have gained widespread adoption [1, 6]. These models are responsible for two tasks: converting speech input to text representation and then performing the translation. A typical E2E-ST model requires a large number of parameters. For example, the Alexa-ST model [7] consists of a speech encoder, a text encoder connected through an adapter, and a unified decoder for the translation task, which results in a model of 1050.1 million parameters and a footprint of 2.1 GB with float16. This poses a limitation when E2E-ST models need to be deployed on-device for real life application [8]. Therefore, a compressed ST model is often desirable. A single model architecture can facilitate compression in one shot. However, there is a lack of research in optimizing the compression for cross-modality joint task models.

One common approach to reduce model size is through Knowledge Distillation (KD) [9] which is a technique to transfer knowledge from an expert teacher model to a compressed student model in a supervised manner, with additional conditions applied on the transferable knowledge, such as response-based, feature-based, and relation-based [10]. In the machine translation (MT) domain [11, 12], KD is a well-established approach used for training small models retaining expert model performance. In the E2E-ST task, [13] and [6] independently propose a similar structure E2E-ST model with an MT teacher model, but they attach an additional feature extraction module without compressing the model. To the best of our knowledge, there is no work yet on compressing an E2E-ST model to reduce the model size. This may be due to the challenge that smaller models can only learn knowledge from an expert teacher model to a certain extent [14], and the speech-to-text modality mapping is a major performance bottleneck in E2E-ST task [15].

In this work, we aim to compress a large E2E-ST model into a smaller model with 50% less parameters while performing similar to the teacher model. We consider Alexa-ST model as our baseline model, and apply different KD loss terms during training our smaller model, so that it can effectively transfer knowledge from the teacher model. We focus on, 1) finding smaller model architecture and weight initialization, 2) identifying effective KD loss terms, and 3) exploring progressive training scheme.

2. Model

Alexa-ST model [7] is an E2E-ST system which takes both speech and text as input for speech translation model training. This model shares the same architecture as FAIR-ST [1] with improved loss function. We consider the original Alexa-ST model as the baseline model, from which we build our compressed models.

2.1. Baseline model

Our base model employs an encoder-decoder architecture, featuring two encoder units: a *text* encoder and a *speech* encoder, and one decoder unit. The text encoder is a 12-layer transformer, pre-trained with the mBART encoder [1]. The speech encoder is a 24-layer transformer, where the first 12 layers and the speech feature extractor are pre-trained using the Wav2Vec 2.0 model [16]. The remaining 12 layers of the speech encoder share the same weights as the text encoder. An adaptor [12] made up of three 1-D convolution layers with a stride of two is inserted between the speech encoder and text encoder to compress the speech encoder output by a factor of eight. The decoder is pre-trained using the mBART [17] decoder and is shared by both speech and text modalities. For pre-trained models, only the LayerNorm and Attention (LNA) [12] parameters are fine-tuned for data efficiency and cross-lingual knowledge learning. Figure 1 shows the overview of the model structure where encoder has 24-transformer layers and decoder has 12-transformer layers.

§ Work done during an internship at Amazon Alexa AI.

10.21437/Interspeech.2023-891

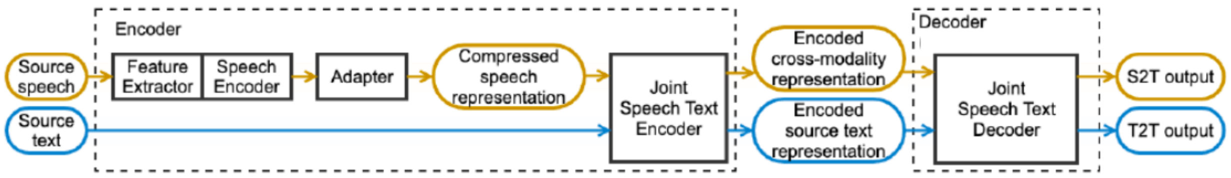
Encoder
Source speech
Source text
Feature Extractor
Speech Encoder
Adapter
Compressed speech representation
Joint Speech Text Encoder
Encoded cross-modality representation
Encoded source text representation
Decoder
Joint Speech Text Decoder
S2T output
T2T output

Figure 1: *Overview of our proposed model. The flow of speech and text modalities are represented by yellow and blue lines, respectively. Number of encoder and decoder layers differs between baseline and compressed models.*

2.2. Compressed model

We simplify the model structure by reducing the number of layers in the speech encoder, text encoder, and decoder. In Figure 1, the compressed model has 12 transformer layers in the encoder and 6 transformer layers in the decoder when using the 6-6-6 configuration. In the 8-8-2 configuration, the encoder has 16 layers and the decoder has 2 layers. Note that in this work we focus on using a general model compression scheme which excludes the vocabulary trimming, namely reducing the embedded parameters of mBART, since the vocabulary size depends on the translation languages and domains. Table 1 shows the details of model parameter size in our baseline and compressed models. Average number of model parameters used by our proposed 50% compressed models is ∼356 million (M), which corresponds to 57.8% of the total parameters in the baseline.

As proposed by [18], we initialize the compressed model by copying weights from the baseline model's layers that are maximally spaced. When we initialize a 6-layer compressed encoder/decoder from a 12-layer baseline encoder/decoder, we choose $[0-11]_{12} \rightarrow \{0,2,4,7,9,11\}_6$ layers from the baseline model. In the 8-8-2 structure, we select $\{0,2,3,5,6,8,9,11\}_8$ layers from each encoder and $\{0,11\}_2$ layers from the decoder. We apply this model compression scheme on both the encoder (speech encoder and text encoder) and decoder, which differs from the approach taken in [18] and [19] where only the text encoder and decoder undergo compression.

3. Methodology

We train our model to perform both speech-to-text (S2T) and text-to-text (T2T) translation. During training, we simultaneously optimize these functions. We also use the knowledge distillation (KD) framework [9, 11], to compress a large model into a smaller one that maintains similar performance. With this KD approach, we explore different training schemes where we incrementally train different modules of the smaller model.

3.1. Training objective function

Speech translation involves two data modalities, i.e. speech (*sph*) and text (*txt*). To bridge the gap between these modalities, we adapt the training objective function from [7, 20] as:

$$\mathcal{L} = \overbrace{(\mathcal{L}_{sph} + \mathcal{L}_{t\_guide}) + \mathcal{L}_{sph\_kd}}^{total\ speech\ loss} + \overbrace{\mathcal{L}_{txt} + \mathcal{L}_{txt\_kd}}^{total\ text\ loss} + \mathcal{L}_{cross\_attn} \quad (1)$$

Table 1: *Model size and number of parameters in the models.*

models→	Baseline	Compressed	
SphEnc-TxtEnc-Dec	12-12-12	8-8-2	6-6-6
total params (M)↓	616.33	347.6	364.4
total params(%)↓	100	56.4	59.12
embedded params (M)	93.58	93.58	93.58
params excluding embedded (M)↓	522.76	254.02	270.82
params excluding embedded(%)↓	100	48.59	51.81

where the *total speech loss* is calculated as a weighted sum of the S2T loss and the speech KD loss (referred to as $\mathcal{L}_{sph\_kd}$). The S2T loss consists of two components: $\mathcal{L}_{sph}$, the loss between the hypothesis of speech and ground truth, and $\mathcal{L}_{t\_guide}$, the loss between the hypothesis of speech and text [20]. In our proposed KD model, the text hypothesis generated by the teacher model is used to compute $\mathcal{L}_{t\_guide}$. The *total text loss* is calculated in a similar manner using the T2T loss ($\mathcal{L}_{txt}$) and the text KD loss ($\mathcal{L}_{txt\_kd}$). Details on KD losses are discussed in section 3.2. Additionally, $\mathcal{L}_{cross\_attn}$ is the cross-attention regularization from speech and text data modalities [20]. The S2T and T2T losses are calculated using the dynamic dual skew divergence (DDSD) [21] loss function, as previous studies have shown better performance with this function compared to cross-entropy loss in translation tasks. We use the following equation to calculate DDSD loss between speech and text hypothesis:

$$D(T,S) = \beta D_d(S||T) + (1-\beta) D_d(T||S) \quad (2)$$

$$D_d(T||S) = D_{kl}\big(T||(\alpha T + (1-\alpha)S)\big) \quad (3)$$

$$D_d(S||T) = D_{kl}\big(S||(\alpha S + (1-\alpha)T)\big) \quad (4)$$

Here D_{kl} is Kullback-Leibler (KL) divergence function, T and S are text and speech hypotheses respectively, and α and β are hyperparameters.

3.2. Knowledge distillation

In the KD framework [9], a teacher-student setting is used where the student learns from both the ground-truth labels and the soft labels provided by the teacher. The probability mass associated with each class in the soft labels allows the student to learn more information about the label similarities for a given sample. This knowledge is referred to as response-based knowledge [10]. Later studies [11, 22] have shown that directly matching intermediate layer outputs is more effective compared to using only response-based KD, and this is known as feature-based knowledge. Additionally, knowledge distillation on attention heads of BERT model [23] has been proven to be effective, and this is referred to as relation-based knowledge [11]. Here, our baseline model (section 2.1) is considered the teacher model, and the compressed model (section 2.2) is considered the student model. We calculate these three types of KD losses (response-based, feature-based, and relation-based) on both speech and text modalities and represent them as $\mathcal{L}_{sph\_kd}$ and $\mathcal{L}_{txt\_kd}$, respectively. We calculate the KD loss terms as follow:

$$\mathcal{L}_{X\_kd} = \mathcal{L}^{X}_{kd\_res} + \mathcal{L}^{X}_{kd\_feat} + \mathcal{L}^{X}_{kd\_rel} \quad (5)$$

$$\mathcal{L}^{X}_{kd\_res} = \sum_{l \in \{L_{enc}, L_{dec}\}} D\big(P(X^B_l), P(X^M_l)\big) \quad (6)$$

$$\mathcal{L}^{X}_{kd\_feat} = \sum_{l \in L} \big(1 - \cos\big(lay(X^B_l), lay(X^M_l)\big)\big) \quad (7)$$

$$\mathcal{L}^{X}_{kd\_rel} = \sum_{l \in L} \big(1 - \cos\big(attn(X^B_l), attn(X^M_l)\big)\big) \quad (8)$$

Table 2: *Different training progression schemes. Active and inactive loss terms are denoted using* $\surd$ *and* $\times$*, respectively.*

↓Scheme	↓Stage	KD loss			Speech & Guide loss	Text loss
		Sph encoder	Txt encoder	Decoder		
Module	Sph encoder	√	×	×	√	√
	Sph encoder + Txt encoder	√	√	×	√	√
	Sph encoder + Txt encoder + Decoder	√	√	√	√	√
Task	Sph encoder	√	×	×	×	×
	Sph encoder + Txt encoder	√	√	×	×	√
	Sph encoder + Txt encoder + Decoder	√	√	√	√	√
All	Same in all 3 stages	√	√	√	√	√

Here $\mathcal{L}_{kd\_res}$, $\mathcal{L}_{kd\_feat}$, and $\mathcal{L}_{kd\_rel}$ represent response-based, feature-based, and relation-based KD loss terms, respectively, for a given data modality. We represent each layer of the model as l and the layers of the model are $L = \{l_0, l_1, \cdots\}$. X denotes the modality (i.e. *sph* and *txt*) and X_l denotes layer-wise output. The symbols B and M represent the baseline (teacher) and compressed (student) models with KD, respectively. To calculate $\mathcal{L}_{kd\_res}$ (equation (6)), we consider the class probability $P()$ from the last layer of the encoder (L_{enc}) and decoder (L_{dec}) modules for both speech and text, then apply the DDSD loss function $D()$. For the feature-based and relation-based KD losses, we apply the cosine loss function to each transformer layer l. In equation (7) and (8), we represent layer output and self-attention head using $lay()$ and $attn()$, respectively. During the training of the student model, the weights of the teacher model are fixed and set to evaluation mode, i.e. 0 dropout on the teacher.

3.3. Training scheme

In translation tasks, the complexity of linguistic learning increases from input to output, as reported in [24]. Typically, all internal layers are distilled simultaneously, but [11] found that training layers incrementally with KD loss performs well in machine translation tasks. Following this insight, we investigate incremental training schemes where student models are trained in three stages: 1) Sph encoder, 2) Sph encoder+Txt encoder, and 3) Sph encoder+Txt encoder+Decoder. During these stages, we consider three schemes, depending on which loss terms are active or inactive. In the *Module* scheme, we incrementally activate KD module losses and use S2T and T2T loss terms at all times. In the *Task* scheme, we incrementally activate KD module losses and only use S2T and T2T losses when the necessary modules for inference are being trained. Since we initialize the decoder with the pre-trained T2T model (mBART) weight, the S2T hypothesis of the model in stage 1 and 2 may lead to incorrect weight updates. Thus, the S2T loss (*Speech & Guide loss*) is not used in stages 1 and 2. Also, in stage 1, the text encoder is not trained, but it shares the last 12 layers with the speech encoder. Therefore, any T2T inference during this partial encoder training stage may have a negative impact on model weight, and as a result, the T2T loss (*Text loss*) is not used in stage 1. Finally, in the *All* scheme, we use all loss terms in all stages. Table 2 shows the individual loss term activation in the different training schemes. Note that we train each module of the student model one after another, whereas [11] trains one layer after another.

4. Experiments

In this section, we describe the datasets used, followed by a detailed description of model training settings used in experiments.

4.1. Datasets

We train our models using MuST-C V2 [25], CoVoST v2 [26] and Europarl-ST V1.1 train-clean dataset [27]. The combined corpus contains audio-text pairs for E2E speech translation (ST), including aligned source and target audio transcriptions. We only consider en→de speech translation task. MuST-C covers 14 target languages of different families with English source audio, including 408hrs of en→de speech. CoVoST provides translations of 21 languages-to-en and en-to-15 languages on Common Voice speech corpus [28] which includes 430hrs of en→de speech with professionally collected transcriptions. Europarl-ST supports translation between 6 European languages, including 83hrs of en→de speech which are constructed using debates in the European Parliament. We discard audio clips shorter than 50ms and longer than the 30s. We hold off 1% of the training data as the validation set. For evaluation, there are two test sets, *Common* and *HE*, provided in the MuST-C V2 dataset. Since both test sets have similar data quality and the Common set is 4.12 times larger than the HE set, we choose the MuST-C V2 Common test set to report model performance in this work.

4.2. Model Settings

We use the sequence modeling toolkit fairseq[1] to train our models. To train our baseline teacher model, we adopt the base model hyperparameter settings from [7] and set the loss weights of $\mathcal{L}_{sph}$, $\mathcal{L}_{t\_guide}$, $\mathcal{L}_{txt}$, and $\mathcal{L}_{cross\_attn}$ as 0.2, 0.8, 0.2, and 0.02, respectively. In DDSD loss, α is set to 0.01 and β is set to 0.5. The model is initialized by pre-trained Wav2vec 2.0 [16] and mBART [17] model weights. We apply the Adam optimizer [29] and inverse square root scheduler. We set the warm-up phase to $5k$ steps and the training batch size to a maximum of 3 for both the base and proposed models. The model parameters are updated in every 4 batch and trained until convergence when the loss on dev set increases for 3 consecutive validation steps. With initial learning rate 2.5×10^{-4}, the teacher model is trained for $\sim$ 24hrs.

Small sized student models use same configuration as the teacher model; however, they are trained for longer time, i.e. around 40hrs for 50% compressed models. Student models with knowledge distillation loss terms use KD loss weights $\mathcal{L}_{X\_kd}$ of 0.5 and initial learning rate 5×10^{-4}. After the student model converges on KD task, we continue to fine-tune the student model without KD losses following the teacher model configuration, i.e. weight of $\mathcal{L}_{X\_kd} = 0$. Each model is trained on 8 NVIDIA V100 GPUs on AWS P3 instances. Average training time for non-KD and KD compressed models are $\sim$ 40hrs and $\sim$ 87.3hrs, respectively.

5. Results

We use the Alexa-ST model as our baseline model, referred as B, which has a higher number of transformer layers. We propose student ST models, which have only half the transformer layers of B, optimized using KD loss terms, represented as M. A separate baseline C for compressed model is trained without any KD losses. We evaluate BLEU scores for both speech input and text input, and denote them as *speech BLEU*(BLEU [sph]) and *text BLEU* (BLEU [txt]). Note that text BLEU is just for analyzing the model's learning ability on MT task and is not a metric for assessing speech translation.

[1]`https://github.com/pytorch/fairseq`

Table 3: *Performance of compressed models using different weight initialization schemes & structure. Best shown in* ***bold***.

$\downarrow$ Models	BLEU$\uparrow$ [sph]	Degradation (%)$\downarrow$ [sph]	BLEU $\uparrow$ [txt]	Degradation (%) $\downarrow$ [txt]
B	29.75	-	32.94	-
C : 6-6-6 (fine-tuned)	23.1	18.6	**31.18**	**5.34**
C : 6-6-6 (pre-trained)	**23.34**	**17.76**	30.78	6.56
C : 8-8-2 (pre-trained)	22.97	19.06	28.6	13.18

5.1. Importance of model structure & weight initialization

We firstly experiment with different ways to initialize the compressed model. We initialize the compressed model C (50% compression) with pre-trained Wav2Vec2.0 and mBART model weights, which is the same method used to initialize the baseline model B and is referred as C:pre-trained. Additionally, we initialize the C model with the fine-tuned weights from the B model and refer to this model as C:fine-tuned. In this experiment, we consider the 6-6-6 model structure for the compressed models. Table 3 shows the performance of these models compared to the large teacher model. We find that the model initialized with pre-trained parameters outperforms the model initialized using teacher weights in speech BLEU by a large margin.
We then investigate two different model architectures, 6-6-6 and 8-8-2 configurations as described in section 2.2, both of which achieve 50% of the model compression rate. The 8-8-2 configuration is inspired by [30] which shows shallow decoders combined with deep encoders can be beneficial in non-autoregressive MT tasks. We initialize these models with pre-trained model weights in light of the previous results. In both speech and text BLEU (Table 3), the 6-6-6 model performs better than the other, indicating the importance of a comparatively deeper decoder module. In these experiments, training parameters are similar to baseline training, i.e. no KD approach is applied.

5.2. Importance of KD loss terms

We conduct experiments to evaluate the effectiveness of individual KD loss and overall performance gain in a compressed model. As shown in ablation study part of Table 4 , we individually apply KD loss terms to the smaller student model. The model $M : \mathcal{L}_{kd\_res}(enc)$ performs the best compared to the other M models and the non-KD compressed model C. The $M : \mathcal{L}_{kd\_feat}$ model shows the next best performance, followed by the $M : \mathcal{L}_{kd\_res}(dec)$ model. We observe that using only relation-based KD does not perform better than the non-KD C model. From our experimental results, the most to least effective KD loss terms are, 1) encoder response-based loss $\mathcal{L}_{kd\_res}(enc)$, 2) decoder response-based loss $\mathcal{L}_{kd\_res}(dec)$, and feature-based loss $\mathcal{L}_{kd\_feat}$, 3) relation-based loss $\mathcal{L}_{kd\_rel}$.

Table 4: *Performance of compressed models using KD losses. Best shown in* ***bold***.

$\downarrow$ Models	BLEU$\uparrow$ [sph]	Degradation (%)$\downarrow$ [sph]	BLEU $\uparrow$ [txt]	Degradation (%) $\downarrow$ [txt]
Baselines				
B	28.38	-	32.94	-
C : 6-6-6 (pre-trained)	23.34	17.76	30.78	6.56
Ablation study on individual KD loss terms				
M	23.24	18.11	31.03	5.8
$M : \mathcal{L}_{kd\_res}(dec)$	23.26	18.04	31.22	5.22
$M : \mathcal{L}_{kd\_res}(enc)$	**23.78**	**16.21**	**31.31**	**4.95**
$M : \mathcal{L}_{kd\_feat}$	23.69	16.53	31.05	5.74
$M : \mathcal{L}_{kd\_rel}$	21.66	23.68	29.98	8.99
Performance of student KD models				
$M : \mathcal{L}_{kd\_res}$	**24.72**	**12.9**	**32.15**	**2.4**
$M : \mathcal{L}_{kd\_res}(enc) + \mathcal{L}_{kd\_feat}$	24.31	14.34	31.34	4.86
$M : \mathcal{L}_{kd\_res}(enc) + +\mathcal{L}_{kd\_avg(feat)}$	24.14	14.94	31.6	4.07
$M : \mathcal{L}_{kd\_res} + \mathcal{L}_{kd\_feat}$	24.38	14.09	31.58	4.13
Fine-tune the best-performing $M : \mathcal{L}_{kd\_res}$ with S2T and T2T loss				
$M^* : \mathcal{L}_{kd\_res}$	**25.33**	**10.75**	**32.07**	**2.64**

Table 5: *Performance of the student models using $\mathcal{L}_{kd\_res}$ loss and trained with different training scheme. Best shown in* ***bold***.

$\downarrow$ Models	BLEU$\uparrow$ [sph]	Degradation (%)$\downarrow$ [sph]	BLEU $\uparrow$ [txt]	Degradation (%) $\downarrow$ [txt]
Baselines				
$M : \mathcal{L}_{kd\_res}$	24.72	12.9	32.15	2.4
$M^* : \mathcal{L}_{kd\_res}$	25.33	10.75	32.07	2.64
Performance of using different training schemes				
$M : module$	24.47	13.78	31.77	3.55
$M : task$	**25.11**	**11.52**	31.69	3.79
$M : all$	24.4	14.02	**31.93**	**3.07**
Fine-tune with S2T and T2T loss				
$M^* : module$	24.78	12.68	31.64	3.95
$M^* : task$	**25.24**	**11.06**	31.67	3.86
$M^* : all$	25.16	11.35	**31.72**	**3.7**

Based on the findings of the ablation study, we experiment with multiple KD loss terms. Table 4 shows the performance of our proposed 50% compressed student KD-ST models. Response-based KD (both encoder and decoder KD loss) gives the best performance in both speech and text BLEU scores. Additionally, we observe that adding both response-based and feature-based KD results in a degradation in performance compared with using response-based KD loss only, which may be because the model has to optimize too many constraints at once. We fine-tune the best-performing $M : \mathcal{L}_{kd\_res}$ with S2T and T2T loss, resulting in additional performance gain in the model $M^* : \mathcal{L}_{kd\_res}$ with a 0.61 gain in speech BLEU.

5.3. Training scheme of compressed model

We investigate three progressive training schemes to improve student model training. We use the best-performing $M : \mathcal{L}_{kd\_res}$ model configuration for this experiment. We train the model for 10 epochs in the first two stages, and then train it until convergence. After full convergence, we fine-tune the models without any KD loss terms, namely only performing the S2T and T2T tasks. Table 5 shows the performance among different training schemes. We observe that the model $M : task$ outperforms all the other training schemes and results in speech BLEU improvement over the highest performing non-incremental training method $M : \mathcal{L}_{kd\_res}$ before fine-tuning. This indicates that the *task* incremental training method effectively transfers knowledge from the teacher model. However, after fine-tuning, the fine-tuned $M^* : task$ model did not outperform the fine-tuned $M^* : \mathcal{L}_{kd\_res}$ model. This may be due to the incremental training method placing more emphasis on teacher-student knowledge transfer than on achieving optimal S2T performance.

6. Conclusion

In this study, we conduct a detailed study on compressing joint E2E speech translation models using knowledge distillation techniques, which is among the pioneers in achieving competitive results. Despite not reducing the size of vocabulary or embeddings, we observe only a 10.75% of the BLEU degradation with 50% model compression rates, on English to German translation. The performance of our models can be further improved by optimal hyperparameter tuning, which was not the focus of this work though. We also experiment with progressive module training schemes for effective knowledge transfer. In the future, we plan to extend this work to multiple languages, and generalize the method to other E2E architectures in addition to our transformer-based model. Furthermore, we will also investigate incremental training schemes on a layer basis.

7. References

[1] Y. Tang, H. Gong, X. Li, C. Wang, J. Pino, H. Schwenk, and N. Goyal, "Fst: the fair speech translation system for the iwslt21 multilingual shared task," *arXiv preprint arXiv:2107.06959*, 2021.

[2] A. Anastasopoulos, O. Bojar, J. Bremerman, R. Cattoni, M. Elbayad, M. Federico, X. Ma, S. Nakamura, M. Negri, J. Niehues, J. Pino, E. Salesky, S. Stüker, K. Sudoh, M. Turchi, A. Waibel, C. Wang, and M. Wiesner, "FINDINGS OF THE IWSLT 2021 EVALUATION CAMPAIGN," in *Proceedings of the International Conference on Spoken Language Translation (IWSLT 2021)*. Bangkok, Thailand (online): Association for Computational Linguistics, Aug. 2021. [Online]. Available: https://aclanthology.org/2021.iwslt-1.1

[3] E. Ansari, A. Axelrod, N. Bach, O. Bojar, R. Cattoni, F. Dalvi, N. Durrani, M. Federico, C. Federmann, J. Gu *et al.*, "Findings of the iwslt 2020 evaluation campaign," in *Proceedings of the International Conference on Spoken Language Translation*, 2020, pp. 1–34.

[4] S. Deena, R. W. Ng, P. Madhyastha, L. Specia, and T. Hain, "Exploring the use of acoustic embeddings in neural machine translation," in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2017, pp. 450–457.

[5] H. Inaguma, K. Duh, T. Kawahara, and S. Watanabe, "Multilingual end-to-end speech translation," in *IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2019, pp. 570–577.

[6] M. Gaido, M. A. Di Gangi, M. Negri, and M. Turchi, "End-to-end speech-translation with knowledge distillation: Fbk@ iwslt2020," *arXiv preprint arXiv:2006.02965*, 2020.

[7] A. Shanbhogue, R. Xue, C. Y. Chang, and S. Campbell, "Amazon alexa ai's system for iwslt 2022 offline speech translation shared task," in *Proceedings of the International Conference on Spoken Language Translation (IWSLT 2022)*, 2022, pp. 169–176.

[8] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[9] G. Hinton, O. Vinyals, J. Dean *et al.*, "Distilling the knowledge in a neural network," *arXiv preprint arXiv:1503.02531*, vol. 2, no. 7, 2015.

[10] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *International Journal of Computer Vision*, vol. 129, pp. 1789–1819, 2021.

[11] G. Aguilar, Y. Ling, Y. Zhang, B. Yao, X. Fan, and C. Guo, "Knowledge distillation from internal representations," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7350–7357.

[12] X. Li, C. Wang, Y. Tang, C. Tran, Y. Tang, J. Pino, A. Baevski, A. Conneau, and M. Auli, "Multilingual speech translation with efficient finetuning of pretrained models," *arXiv preprint arXiv:2010.12829*, 2020.

[13] Y. Liu, H. Xiong, Z. He, J. Zhang, H. Wu, H. Wang, and C. Zong, "End-to-end speech translation with knowledge distillation," *arXiv preprint arXiv:1904.08075*, 2019.

[14] S. I. Mirzadeh, M. Farajtabar, A. Li, N. Levine, A. Matsukawa, and H. Ghasemzadeh, "Improved knowledge distillation via teacher assistant," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, no. 04, 2020, pp. 5191–5198.

[15] M. A. Di Gangi, R. Cattoni, L. Bentivogli, M. Negri, and M. Turchi, "Must-c: a multilingual speech translation corpus," in *the conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2019, pp. 2012–2017.

[16] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in Neural Information Processing Systems*, vol. 33, pp. 12 449–12 460, 2020.

[17] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer, "Multilingual denoising pre-training for neural machine translation," *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 726–742, 2020.

[18] S. Shleifer and A. M. Rush, "Pre-trained summarization distillation," *arXiv preprint arXiv:2010.13002*, 2020.

[19] Z. Li, Z. Wang, M. Tan, R. Nallapati, P. Bhatia, A. Arnold, B. Xiang, and D. Roth, "Dq-bart: Efficient sequence-to-sequence model via joint distillation and quantization," *arXiv preprint arXiv:2203.11239*, 2022.

[20] Y. Tang, J. Pino, X. Li, C. Wang, and D. Genzel, "Improving speech translation by understanding and learning from the auxiliary text translation task," *arXiv preprint arXiv:2107.05782*, 2021.

[21] Z. Li, H. Zhao, Y. Wu, F. Xiao, and S. Jiang, "Controllable dual skew divergence loss for neural machine translation," *arXiv preprint arXiv:1908.08399*, 2019.

[22] A. Romero, N. Ballas, S. E. Kahou, A. Chassang, C. Gatta, and Y. Bengio, "Fitnets: Hints for thin deep nets," *arXiv preprint arXiv:1412.6550*, 2014.

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

[24] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," *arXiv preprint arXiv:1906.04341*, 2019.

[25] R. Cattoni, M. A. Di Gangi, L. Bentivogli, M. Negri, and M. Turchi, "Must-c: A multilingual corpus for end-to-end speech translation," *Computer Speech & Language*, vol. 66, p. 101155, 2021.

[26] C. Wang, A. Wu, and J. Pino, "Covost 2 and massively multilingual speech-to-text translation," *arXiv preprint arXiv:2007.10310*, 2020.

[27] J. Iranzo-Sánchez, J. A. Silvestre-Cerda, J. Jorge, N. Roselló, A. Giménez, A. Sanchis, J. Civera, and A. Juan, "Europarl-st: A multilingual corpus for speech translation of parliamentary debates," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 8229–8233.

[28] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] J. Kasai, N. Pappas, H. Peng, J. Cross, and N. A. Smith, "Deep encoder, shallow decoder: Reevaluating non-autoregressive machine translation," *arXiv preprint arXiv:2006.10369*, 2020.