Improving Speech Translation by Understanding and Learning from the Auxiliary Text Translation Task

Yun Tang, Juan Pino, Xian Li, Changhan Wang, Dmitriy Genzel
Facebook AI
{yuntang,juancarabina,xianl,changhan,dgenzel}@fb.com

Abstract

Pretraining and multitask learning are widely used to improve the speech to text translation performance. In this study, we are interested in training a speech to text translation model along with an auxiliary text to text translation task. We conduct a detailed analysis to understand the impact of the auxiliary task on the primary task within the multitask learning framework. Our analysis confirms that multitask learning tends to generate similar decoder representations from different modalities and preserve more information from the pretrained text translation modules. We observe minimal negative transfer effect between the two tasks and sharing more parameters is helpful to transfer knowledge from the text task to the speech task. The analysis also reveals that the modality representation difference at the top decoder layers is still not negligible, and those layers are critical for the translation quality. Inspired by these findings, we propose three methods to improve translation quality. First, a parameter sharing and initialization strategy is proposed to enhance information sharing between the tasks. Second, a novel attention-based regularization is proposed for the encoders and pulls the representations from different modalities closer. Third, an online knowledge distillation is proposed to enhance the knowledge transfer from the text to the speech task. Our experiments show that the proposed approach improves translation performance by more than 2 BLEU over a strong baseline and achieves state-of-the-art results on the MuST-C English-German, English-French and English-Spanish language pairs.

1 Introduction

End-to-end methods have achieved significant progress in speech to text translation (ST) and even surpassed the traditional pipeline-based methods in some applications (Niehues et al., 2019; Salesky and Black, 2020). However, the success of end-to-end methods relies on large amounts of training data, which is quite expensive to obtain and relatively small in practice. Building ST systems from pretrained models with multitask learning (MTL) is widely used to overcome the limited training data issue (Weiss et al., 2017; Anastasopoulos and Chiang, 2018; Bahar et al., 2019; Indurthi et al., 2020; Wang et al., 2020b; Li et al., 2020). Nevertheless, little prior work has been devoted to understanding the interactions between different tasks. Standley et al. (2020) conduct an empirical study on computer vision tasks for MTL. They find many "assumptions" for MTL may not be held for specific applications. For example, "similar" tasks do not necessarily train better together.

In this study, we focus on training the ST model along with an auxiliary text to text machine translation (MT) task. We are interested in the task interactions with different modalities and in improving the primary ST task with the help from the auxiliary MT task. The model is initialized with pretrained modules from automatic speech recognition (ASR) and MT. Two types of analysis are conducted on the fine-tuned multitask learned models. The first focuses on the model variation by comparing fine-tuned models with pretrained models for different tasks. The second aims to measure internal representation differences due to different modalities. The analysis leads to three main findings. First, the analysis confirms that MTL tends to generate similar model representations for different input modalities and preserves more information from the pretrained MT modules. Second, we do not observe significant negative transfer effect from the MT task to the corresponding ST task. Sharing more parameters is helpful to transfer knowledge to the primary ST task. Finally, the top layers in the ST decoder are more critical to the translation

performance and they are also more sensitive to the modality difference. The model representations from different modalities demonstrate larger difference for the top layers in our analysis.

Inspired by these findings, we propose three techniques to enhance the performance of the primary ST task. First, we propose to maximize parameter sharing between the ST and MT tasks, i.e. the entire decoder and the top encoder layers. Those shared parameters are initialized with the corresponding MT models. Second, a cross-attentive regularization is introduced for the encoders. It minimizes the $L2$ distance between two reconstructed encoder output sequences and encourages the encoder outputs from different modalities to be closer to each other. Finally, an online knowledge distillation learning is introduced for MTL in order to enhance knowledge transfer from the MT to the ST task.

Our contributions are summarized as follows:

1. A detailed analysis is conducted on the interaction between the primary ST task and the auxiliary MT task.
2. A parameter sharing and initialization strategy are proposed to encourage information sharing between tasks.
3. Cross-attentive regularization and online knowledge distillation are proposed to reduce the model representation difference between different modalities and enhance the knowledge transfer from the MT task to the ST task.
4. Our system achieves state of the art results on the MuST-C English-German (EN-DE), English-French (EN-FR) and English-Spanish (EN-ES) language pairs, with 2 or more BLEU gains over strong baselines.

2 Related Work

Multitask learning aims to improve generalization by leveraging domain-specific information contained in the training signals of related tasks (Vandenhende et al., 2020). Compared with single task, MTL has many advantages, such as the potential to improve performance by sharing complementary information or acting as a regularizer. Many previous works focus on learning a good model for all tasks. Chen et al. (2018) study the gradients from different tasks and conduct task dependent gradient normalization to encourage different tasks to learn at similar speed. Maninis et al. (2019); Liu et al. (2019a); Pfeiffer et al. (2020) introduce task-dependent components to enhance individual task performance.

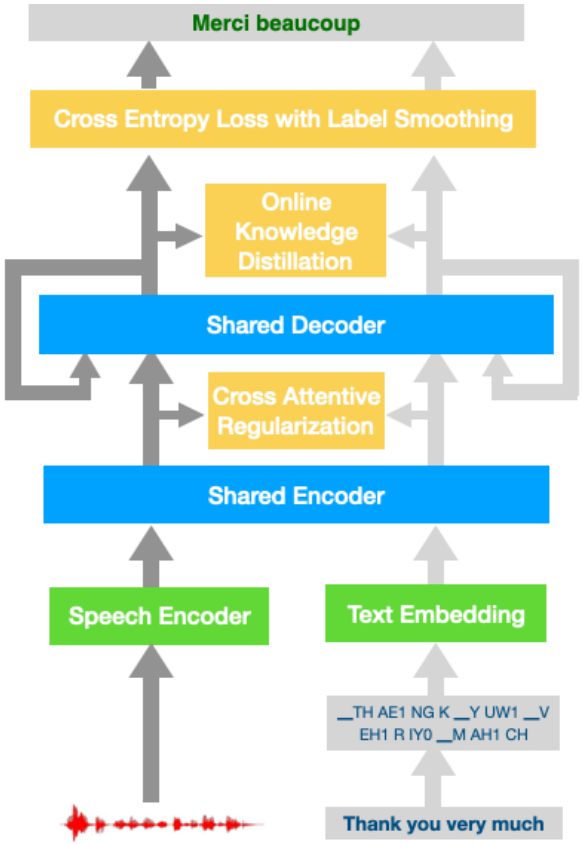


Figure 1: Joint Training framework. The speech to text translation task is depicted as dark gray line, text to text translation task is illustrated as light gray line. The parameters in blue modules are shared between two tasks.

Weiss et al. (2017) explore different multitask training strategies for ST, and they find the one-to-many strategy, in which an encoder is shared between the ST and ASR tasks, is more effective. Anastasopoulos and Chiang (2018) further extend it to a triangle structure by concatenating ASR and ST models. Bahar et al. (2019) compare different multitask strategies for the ST task, and they confirm many-to-one strategy, in which MT and ST are trained together and the decoder is shared between two tasks, is effective if extra bitext data is used. In this work, we carefully study the relation between co-trained tasks in the many-to-one strategy, and the analysis results guide us to propose three techniques to learn more from the auxiliary MT task and enhance the ST performance further.

Model analysis Chatterji et al. (2020) propose criticality analysis to measure the importance of different modules from the trained model. Parameters

in the selected module or layer are partially rolled back to the initial values, and the module criticality or importance is measured by the performance drop after modification. Larger performance drops indicate a more critical module. Inspired by their work, we extend it to the analysis on the jointly trained models with different pretrained modules and schemes. Raghu et al. (2017); Morcos et al. (2018) propose to employ canonical correlation to measure the similarity between different models given the same input. We extend their work to study a model with inputs from different modalities.

3 Methods

The proposed ST system is co-trained with the MT task as depicted in Figure 1. The modules in the primary ST task are connected with dark gray lines and the auxiliary MT task is illustrated with light gray lines. The parameters in the blue modules are shared between the two tasks. During inference with speech input, only modules related to the ST task are used.

The model has two encoders, a text encoder and a speech encoder, to take text and speech input respectively. The decoder is shared between the two tasks. To encourage knowledge sharing between the two tasks, the top encoder layers are also shared. The parameters of the shared modules are initialized with a pretrained MT model. A novel cross-attentive regularization is proposed to reduce the distance between encoder outputs from different input modalities. We also introduce a novel online knowledge distillation method where the output from the auxiliary MT task is used to guide the ST model training. The cross-attentive regularization and online knowledge distillation are illustrated as orange modules in Figure 1 and the details are presented in the following two subsections.

3.1 Cross-Attentive Regularization

The cross-attentive regularization (CAR) is proposed to increase the similarity between the text encoder outputs and their corresponding speech encoder outputs. Hence, the performance of the more difficult ST task can be improved by learning from the relatively easier MT task. Encoder output sequences from different modalities can not be compared directly since they have different lengths. In CAR, the two reconstructed sequences are calculated from the text output sequence via self-attention or the speech output sequence via cross attention over the text output sequence. The two reconstructed sequences have the same length and the distance is simply measured as the $L2$ distance between the two sequences.

Formally, we denote a speech to text translation training sample as a triplet $o = (\mathbf{X^s}, \mathbf{x^t}, \mathbf{y})$. $\mathbf{X^s} \in \mathcal{R}^{d_s \times N}$, $\mathbf{x^t} \in \mathcal{R}^M$, and $\mathbf{y} \in \mathcal{R}^K$ are the speech feature input, text token input and target text output respectively. N, M and K are the corresponding sequence lengths. Assume $\mathbf{H^s} = (\mathbf{h}_1^s, \mathbf{h}_2^s, \cdots, \mathbf{h}_N^s)$ and $\mathbf{H^t} = (\mathbf{h}_1^t, \mathbf{h}_2^t, \cdots, \mathbf{h}_M^t)$, $\mathbf{h}_n^s, \mathbf{h}_m^t \in \mathcal{R}^{d_h}$ are outputs from the speech encoder and text encoder respectively, where d_h is the dimension of the output states. A similarity matrix $\mathbf{S} \in \mathcal{R}^{N \times M}$ is defined as the cosine distance between the tensors in the two sequences:

$$s_{i,j} = \frac{(\mathbf{h}_i^s)' \cdot \mathbf{h}_j^t}{||\mathbf{h}_i^s||_2 ||\mathbf{h}_j^t||_2} \tag{1}$$

where $s_{i,j}$ is the ith row and jth column component in $\mathbf{S}$. The text encoder outputs $\mathbf{H}^t$ are reconstructed through the speech encoder outputs $\mathbf{H}^s$ and similarity matrix $\mathbf{S}$ as below.

$$\mathbf{H}^{s \to t} = \mathbf{H}^s \cdot \mathrm{softmax}(\mathbf{S}) \tag{2}$$

$\mathbf{H}^{t \to t}$, the reconstruction of $\mathbf{H}^t$ from itself, can be computed similarly via self-attention. CAR is defined as the $L2$ distance between the two reconstruction encoder outputs:

$$\mathcal{L}_{CAR}(\theta_s) = \frac{1}{M} \left\| \mathbf{H}^{s \to t} - sg[\mathbf{H}^{t \to t}] \right\|_2 \tag{3}$$

where $sg[\cdot]$ is the stop-gradient operator and θ_s are the ST model parameters. By optimizing the model with CAR, the speech encoder is encouraged to learn from more accurate text encoder and generates similar encoder outputs after reconstruction. CAR is inspired by the attention mechanism between the encoder and decoder where the decoder states are reconstructed through encoder output states via the attention mechanism.

3.2 Online Knowledge Distillation

Knowledge distillation (KD) is widely used for model compression (Hinton et al., 2015; Kim and Rush, 2016) where a smaller student network is trained to mimic the original teacher network by minimizing the loss between the student and teacher outputs. The ST task is considerably more difficult than the MT task since the speech input is noisier and more ambiguous than the text input.

The accuracy of the MT model is usually much higher than the corresponding ST model. Knowledge distillation from a well trained MT model to a ST model has been proved to be an effective way to improve the ST performance (Liu et al., 2019b; Gaido et al., 2020). In this work, we extend knowledge distillation to the MTL framework where both ST and MT are fine-tuned simultaneously with shared parameters.

Concretely, we assume an MTL model learns from a data set $\mathcal{D}$ with target vocabulary size $|V|$. The training criterion is to minimize negative log likelihood (NLL) for each example $o = (\mathbf{X^s}, \mathbf{x^t}, \mathbf{y}) \in \mathcal{D}$ from the training data:

$$\mathcal{L}_{NLL}(\theta_s) = -\sum_{o}^{\mathcal{D}} \sum_{k=1}^{K} \sum_{v=1}^{|V|} \delta(y_k = v) \log p(y_k = v | y_{<k}, \mathbf{X}^s, \theta_s) \quad (4)$$

where $\delta(\cdot)$ is the indicator function and p the distribution from the ST model (parameterized by θ_s).

Assume the probability distribution for y_k given text input $\mathbf{x}^t$ and MT model θ_t is $q(y_k = v | y_{<k}, \mathbf{x}^t, \theta_t)$, the knowledge distillation loss is defined as minimizing the cross-entropy with the MT's probability distribution

$$\mathcal{L}_{KD}(\theta_s) = -\sum_{o}^{\mathcal{D}} \sum_{k=1}^{K} \sum_{v=1}^{|V|} q(y_k = v | y_{<k}, \mathbf{x}^t, \theta_t) \log p(y_k = v | y_{<k}, \mathbf{X}^s, \theta_s) \quad (5)$$

The overall loss is the combination of cross-attentive regularization, knowledge distillation loss, negative log likelihood loss for both ST and MT, as follows:

$$\mathcal{L}(\theta_s, \theta_t) = \alpha \mathcal{L}_{NLL}(\theta_s) + (1 - \alpha) \mathcal{L}_{KD}(\theta_s) + \lambda \mathcal{L}_{CAR}(\theta_s) + \mathcal{L}_{NLL}(\theta_t) \quad (6)$$

where α and λ are predefined hyper-parameters.

4 Experimental Setup

Experiments are conducted on three MuST-C (Gangi et al., 2019a) language pairs: EN-DE, EN-ES and EN-FR. The models are developed and analyzed on the dev set and the final results are reported on the tst-COMMON set. We use WMT parallel data from different years, 2013 for Spanish, 2014 for German, and 2016 for French, as extra text training corpus for MTL. Case-sensitive detokenized BLEU is reported by SACREBLEU with default options (Post, 2018).

We use the "T-Md" configuration from (Wang et al., 2020a) in all experiments. The speech encoder has 12 transformer layers while the decoder is with 6 transformer layers. For the MTL model, the text encoder has 6 transformer layers. The transformer layer has an input embedding size of 512 and middle layer dimension 2048. We share parameters of all 6 text encoder transformer layers with the top 6 transformer layers in the speech encoder, hence both encoders use the same modules to generate the encoder outputs.

The Adam optimizer (Kingma and Ba, 2014) with a learning rate 0.002 is employed in the experiments. Label smoothing and dropout rate are both set to 0.1. We choose $\alpha = 0.8$ and $\lambda = 0.02$ in Equation 6 through grid search ($[0.1, 1.0]$ for α and $[0.01, 0.05]$ for λ).

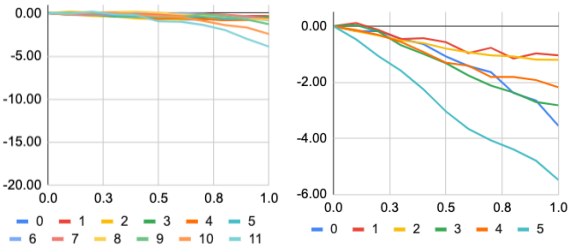
Input speech is represented as 80D log mel-filterbank coefficients computed every 10ms with a 25ms window. Global channel mean and variance normalization is applied. The SpecAugment (Park et al., 2019) data augmentation with the LB policy is applied in all experiments. The input text tokens are converted into their corresponding pronunciation form as phoneme sequences (Tang et al., 2021; Renduchintala et al., 2018). The grapheme to phoneme conversion is done through the "g2p_en" python package (Lee and Kim, 2018). The leading phoneme in a word is appended with an extra "_" to mark word boundaries. In total, the vocabulary size for the input phonemes is 134. The target vocabulary consists of 10k "unigram" subword units learned by SentencePiece (Kudo and Richardson, 2018) with full character coverage of all training text data.

All ST or jointly trained models are initialized with pretrained ASR and MT modules. The ASR model is trained on the same English speech training data from MuST-C with the "T-Md" configuration too. The pretrained MT models are trained for each language pair with the aforementioned WMT data. The MT encoder and decoder configurations are the same as the text encoder and decoder in the MTL model mentioned above.

The models are fine-tuned to 100 epochs using 8 V100 GPUs for approximate one day. The batch size is 10,000 frames for speech to text translation samples and 10,000 tokens for parallel text samples per GPU. The model parameters are updated every 4 batches. Speech training samples and text input samples are used to update the model alternatively.

Model Configuration	Encoder		
	Speech	Text	Shared
ST	ASR	None	None
JT	ASR	MT	None
JT-S-ASR	ASR	MT	ASR
JT-S-MT	ASR	MT	MT

Table 1: Model initialization schemes



(a) ST Enc. (b) ST Dec.

Figure 2: Criticality analysis for the "ST" model.

The models are trained with FAIRSEQ (Ott et al., 2019; Wang et al., 2020a). The last 10 checkpoints are averaged for inference with beam size 5. [1].

5 MTL Analysis

5.1 Model Variation

We extend Chatterji et al. (2020)'s work to analyze a MTL model. We initialize models with different pretrained modules and fine-tune them for ST and MT tasks within the MTL framework. The pretrained modules come from ASR and MT tasks.

Criticality analysis is conducted on the ST model after the MTL fine-tuning step. The parameters in the selected modules are interpolated with corresponding parameters in the pretrained modules. MUST-C EN-DE dev set is used for BLEU computation. With different interpolation ratios, we obtain different BLEU scores. The BLEU difference comes from two sources. The first one comes from the selected module itself. If the module is important and sensitive, very small perturbation could result in a nontrivial BLEU difference as (Chatterji et al., 2020). Another source of difference is that if the selected module changes significantly to adapt to the ST task, rewinding the parameters back to the initial task may lead to a substantial decrease in BLEU. We attempt to quantify the extent of the degradation from the second source, which can be indicative of the model variation from the pretrained task to the ST task. This is accomplished by comparing the BLEU differences for the same module but using different initialization and training schemes.

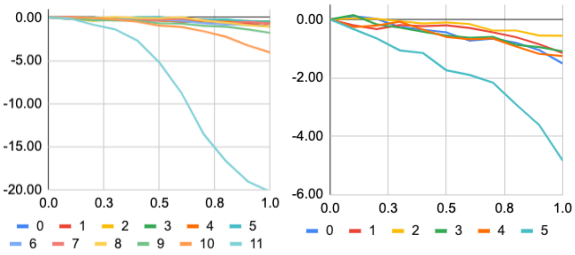
Table 1 lists models initialized with different pretrained modules. "ST" designates a ST model trained with the single ST task, "JT" corresponds to a ST model trained with the primary ST task and auxiliary MT task together. "JT-S-ASR" and "JT-S-MT" are another two jointly trained models but with the top encoder layers shared as described in section 4. The difference between the two models is how we initialized the shared encoder layers, either from the pretrained ASR model for "JT-S-ASR" or from the pretrained MT model for "JT-S-MT".

ST Figure 2 shows the analysis for the "ST" model. The x-axis is the interpolation ratio and "1.0" means the pretrained parameters are used. The y-axis is the relative change in BLEU compared with the well-trained ST model. **It is clear that higher layers are more critical to the performance**. Around 5 BLEU decrease is observed on the top encoder layer (11) and top decoder layer (5) during the criticality tests. The following analysis will compare with Figure 2 and we can separate the aforementioned second source from the first one.

JT Figure 3 presents the analysis for the "JT" model. The jointly trained model shows smaller degradation compared with "ST" for the decoder layers. **This indicates that training the ST and MT tasks together helps to preserve more information from the original MT decoder and partially remedies the catastrophic forgetting (McCloskey and Cohen, 1989) during the fine-tuning phase.** On the other hand, after rolling parameters back to the initial ASR model, the jointly trained model shows a larger degradation for the encoder layers. This means that the speech encoder in the jointly trained model has deviated far away from the speech encoder in the initial ASR task. We conclude that the shared decoder is subject to more constraints since it is optimized toward both MT and ST tasks while the speech encoder has to undergo larger changes in order to align with the text encoder, although there is no parameter sharing between two encoders.

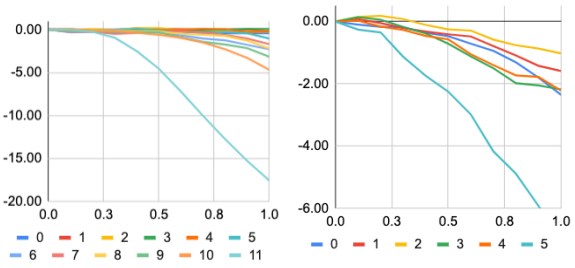
JT-S-ASR and JT-S-MT Results for models with

[1]The source code will be released at https://github.com/pytorch/fairseq/tree/master/examples/speech_text_joint_to_text



(a) JT Enc. (b) JT Dec.

Figure 3: Criticality analysis for the "JT" model.

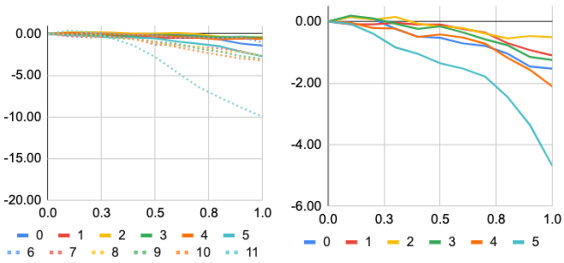


(a) JT-S-ASR Enc. (b) JT-S-ASR Dec.

Figure 4: Criticality analysis for the "JT-S-ASR" model. The shared encoder layers are initialized with the layers from the ASR encoder.

the top encoder layers shared are presented in Figure 4 and 5. In "JT-S-MT", the top 6 shared encoder layers are initialized with the pretrained MT encoder. We illustrate their BLEU difference trajectories with dotted lines in Figure 5 (a) so they can be easily distinguished from other layers initialized from the ASR encoder.

The BLEU difference for the top encoder layer is down from 20.2 to 17.6 when the parameters are replaced with the ones in the pretrained ASR encoder. It is further reduced to 10.0 if the shared layers are initialized with MT encoder layers. The BLEU differences in the decoder layers are mixed. The performance of "JT-S-ASR" degrades quickly in the criticality test for the top decoder layer, while "JT-S-MT performs similarly in the test as "JT" decoder. We argue that the top layers in the fine-tuned ST encoder might be closer to the MT encoder than the ASR encoder. **It preserves more information from the MT task by sharing more parameters between two tasks and initializing them with pretrained MT modules**. This is a desirable property since we want to transfer more knowledge from the text corpus to the ST task.



(a) JT-S-MT Enc. (b) JT-S-MT Dec.

Figure 5: Criticality analysis for the "JT-S-MT" model. The shared encoder layers are initialized with the layers from the MT encoder.

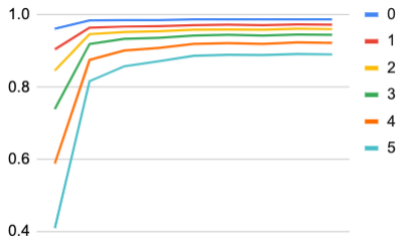


Figure 6: Comparison of decoder layers correlation coefficients between text and speech input ("JT-S-MT").

5.2 Modality Variation

The jointly trained model takes input from two modalities, i.e. text or speech, and we are interested in the model internal representation difference for paired inputs. Given text target $\mathbf{y}$, we extract the decoder hidden state representations for the corresponding text input $\mathbf{x}_t$ and speech input $\mathbf{X}_s$. The decoder representation difference solely comes from different input modalities. The difference is quantified by the correlation coefficient over all samples evaluated between two input modalities:

$$r^{s,t}(l,d) = \frac{\sigma_{st}(l,d)}{\sigma_s(l,d)\sigma_t(l,d)} \quad (7)$$

where $\sigma_z(l,d), z \in [s,t]$ is the standard deviations of decoder hidden states at layer l for component d in all samples, and $\sigma_{st}(l,d)$ is the corresponding covariance. The layer-wise correlation coefficient is the average of all components:

$$r^{s,t}(l) = \frac{1}{D}\sum_{d} r^{s,t}(l,d) \quad (8)$$

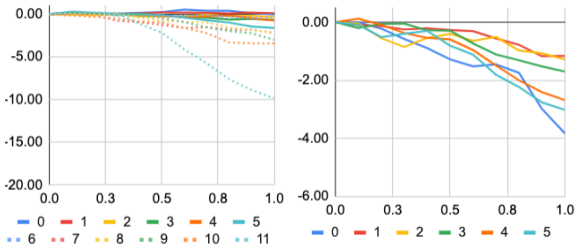
Figure 6 depicts the correlation coefficient between speech input and text input for each decoder layer in the model "JT-S-MT". The x-axis is the number of training epochs and the y-axis represents the correlation coefficient for each layer. There

Data corpus	#pars(m)	DE	ES	FR
Gangi et al. (2019b)	30	17.7	20.9	26.5
Inaguma et al. (2020)	-	22.9	28.0	32.7
Pino et al. (2020)	435	25.2	-	34.5
ST	76	21.5	28.1	33.8
JT	76	24.1	29.0	35.1
JT Proposed	76	26.8	31.0	37.4

Table 2: BLEU on three language pairs in the MuST-C tst-COMMON datasets.

	EN-DE	EN-ES	EN-FR
JT	24.1	29.0	35.1
JT-S-ASR	24.4	29.4	35.4
JT-S-MT	24.7	29.7	35.3
+ CAR	25.0	30.4	36.2
+ CAR + KD	26.8	31.0	37.4

Table 3: Ablation study.



(a) JT Proposed Enc. (b) JT Proposed Dec.

Figure 7: Criticality analysis for "JT Proposed".

are two observations. First, the correlation coefficients become larger and close to "1.0" as training converges. Second, the higher the layer, the smaller the correlation coefficient. We hypothesize that the inputs to the lower layers are dominated by the decoder text embeddings, which are the same for both modalities, and the inputs to the higher layers would contain more information from the encoder outputs, which result in the decoder internal representation differences. **The analysis shows a well trained MTL decoder has similar representations for paired text and speech input. However, the top decoder layers still have nontrivial representation differences due to different modalities**.

6 Experimental Results

6.1 Main Results

The main ST results are presented in Table 2. The first three rows are results from the literature. "ST" and "JT" are models initialized as Table 1 and studied in section 5. The last row ("JT Proposed") presents results from the proposed system, in which the top encoder layers and decoder are shared, and the models are optimized following Equation 6. The second column ("pars(m)") lists the number of parameters used during inference. From Table 2, our "ST" baseline is comparable to the previously reported results except (Pino et al., 2020), who use a much larger model and additional weakly supervised speech training data. As expected, the vanilla joint training baseline ("JT") outperforms the "ST" baseline with the help of extra bitext training data. Finally, the proposed joint training model ("JT Proposed") achieves 2.0∼2.7 BLEU gains over the strong joint training baseline ("JT").

6.2 Ablation

Table 3 breaks down the performance gains into individual components/changes. Sharing encoder layers improves the quality for all three language pairs ("JT" v.s. "JT-S-ASR"). Initializing the shared encoder layers with pretrained MT modules leads to BLEU increase for two of the three evaluated translation pairs ("JT-S-ASR" v.s. "JT-S-MT"). For EN-FR, the degradation is minimal (-0.1 BLEU). Overall, sharing top encoder layers can increase BLEU by 0.2∼0.7 ("JT-S-MT" v.s. "JT"). CAR further improves the translation by another 0.3∼0.9 BLEU. The best results are achieved by applying the shared top encoder layers, CAR and online KD together. They are about 2.9+ BLEU better than the single task based system ("ST") and achieve 2+ BLEU increase on top of the strong vanilla joint training system("JT").

Figure 7 demonstrates the model variation for the proposed system on the MuST-C EN-DE dev set. Compared with Figure 5, the decoder shows less degradation during the criticality test and it shows CAR and online KD help to preserve more information from the MT task. Figure 8 shows the corresponding correlation coefficients between paired text and speech input from the top decoder

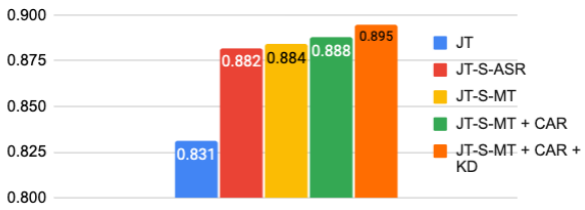


Figure 8: Correlation coefficient for the top decoder layers (epoch 100).

Model	BLEU
JT-S-MT	24.7
JT-S-MT + Adapter	24.7
JT-S-MT + Dedicated Attention	24.2

Table 4: BLEU score for models with task dependent components

	EN-DE	EN-ES	EN-FR
ST (JT Proposed)	26.8	31.0	37.4
MT (Gangi et al., 2019a)	28.1	34.2	42.2
MT	25.4	27.7	33.5
MT (Tuned)	29.6	34.3	41.4
MT (JT)	28.9	33.9	41.6
MT (JT Proposed)	30.5	34.7	42.3

Table 5: Comparison between ST and MT.

layer from different model configurations. It also confirms that the proposed methods, i.e., shared top encoder layers, CAR and online KD, all reduce the modality difference substantially.

6.3 Task Dependent Components

In MLT, many works (Maninis et al., 2019; Liu et al., 2019a; Zhang et al., 2020; Pfeiffer et al., 2020) employ task-dependent components to alleviate the negative transfer effect. In Table 4, we compare the "JT-S-MT" model with two variants using different task-dependent components. The first one ("JT-S-MT + Adapter") (Bapna et al., 2019) adds an extra adapter module on the top of the speech encoder. Hence, the speech encoder outputs, which are generated from shared encoder layers, are further processed to reduce the difference between speech input and text input. The adapter module consists of a linear layer and layer normalization layer. The second variant ("JT-S-MT + Dedicated Attention") (Blackwood et al., 2018) introduces dedicated decoder modules for different tasks. Attention layers between encoder and decoder, and the layer normalization modules are not shared between the ST and MT tasks. It gives the decoder more flexibility to handle information from different modalities.

The results show the extra adapter layer doesn't bring gain while the task dependent attention module actually makes the performance worse. It indicates that the negative transfer effect is not significant in this study and adding extra task-dependent components might not be necessary.

6.4 Impact on the MT Task

As shown in Table 2, training ST models with an auxiliary MT task improves the translation quality substantially. It may be interesting to examine the impact on the auxiliary task itself. We evaluate the MT model jointly trained with the ST task. Results are shown in Table 5. "ST (JT Proposed)" in the first row corresponds to the best results obtained for the ST task. The detailed experimental setup is described in Appendix A. For reference, we also include the MT evaluation results from MuST-C (Gangi et al., 2019a) in the second row. All MT models (in the last 4 rows) take phoneme sequences as input instead of SentencePiece.

"MT" (row 3) shows the results from pretrained MT models on WMT. In the "MT (Tuned)" row, the MT models pretrained on WMT are fine-tuned on the MuST-C datasets. The large improvements clearly show a domain mismatch between WMT and MuST-C. The MT models trained with WMT data are improved after fine-tuning, and they are comparable with the ones reported in (Gangi et al., 2019a), though the input token is in pronunciation form, which is more ambiguous than the corresponding SentencePiece unit.

"MT (JT)" and "MT (JT Proposed)" are results from the co-trained MT models in "JT" and "JT Proposed" respectively. After fine-tuning using both MuST-C (speech and text) and WMT (text only) training data, the auxiliary MT models perform better than the corresponding ST models. The proposed techniques further improve the co-trained MT models by 0.7$\sim$1.6 BLEU. While this is a surprising result, we note that the dedicated MT models may be improved with better hyperparameter tuning. In conclusion, the results show the proposed methods are effective to unify two tasks into one model with minimal negative transfer effect.

7 Conclusions

In this study, we focus on understanding the interactions between the ST and MT tasks under the MTL framework, and on boosting the performance of the primary ST model with the auxiliary MT task. Two types of analysis on model variation and modality variation, are conducted on the MTL models. The analysis demonstrates MTL helps to preserve information from the MT task and generates similar model representations for different modalities. We observe a minimal negative transfer effect between the two tasks. Sharing more parameters can further boost the information transfer from

the MT task to the ST model. The analysis also reveals that the model representation difference due to modality difference is nontrivial, especially for the top decoder layers, which are critical for the translation performance. Inspired by the findings, we propose three techniques to increase knowledge transfer from the MT task to the ST task. These techniques include parameter sharing and initialization strategy to improve the information sharing between tasks, CAR and online KD to encourage the ST system to learn more from the auxiliary MT task and then generate similar model representations from different modalities. Our results show that the proposed methods improve translation performance and achieve state-of–the-art results on three MuST-C language pairs.

References

Antonios Anastasopoulos and David Chiang. 2018. Tied multitask learning for neural speech translation. In *NAACL-HLT*.

Parnia Bahar, Tobias Bieschke, and Hermann Ney. 2019. A comparative study on end-to-end speech to text translation. In *ASRU*.

Ankur Bapna, N. Arivazhagan, and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *EMNLP/IJCNLP*.

G. Blackwood, Miguel Ballesteros, and T. Ward. 2018. Multilingual neural machine translation with task-specific attention. In *COLING*.

Niladri S. Chatterji, Behnam Neyshabur, and H. Sedghi. 2020. The intriguing role of module criticality in the generalization of deep networks. In *ICLR*.

Z. Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*.

Marco Gaido, Mattia Antonino Di Gangi, Matteo Negri, and Marco Turchi. 2020. End-to-end speech-translation with knowledge distillation: Fbk@iwslt2020.

Mattia Antonino Di Gangi, Roldano Cattoni, Luisa Bentivogli, Matteo Negri, and Marco Turchi. 2019a. MuST-C: a multilingual speech translation corpus. In *NAACL-HLT*.

Mattia Antonino Di Gangi, Matteo Negri, and Marco Turchi. 2019b. One-to-many multilingual end-to-end speech translation. In *ASRU*.

Geoffrey E. Hinton, Oriol Vinyals, and J. Dean. 2015. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531.

H. Inaguma, S. Kiyono, K. Duh, S. Karita, N. Soplin, T. Hayashi, and S. Watanabe. 2020. Espnet-st: All-in-one speech translation toolkit. In *ACL*.

Sathish Reddy Indurthi, HouJeung Han, Nikhil Kumar Lakumarapu, Beom seok Lee, Insoo Chung, Sang-Ha Kim, and Chanwoo Kim. 2020. End-end speech-to-text translation with modality agnostic meta-learning. In *ICASSP*.

Yoon Kim and Alexander M. Rush. 2016. Sequence-level knowledge distillation. In *EMNLP*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *ICLR*.

T. Kudo and J. Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *EMNLP*.

Y. Lee and T. Kim. 2018. Learning pronunciation from a foreign language in speech synthesis networks. *ArXiv*.

Xian Li, Changhan Wang, Yun Tang, Chau Tran, Yuqing Tang, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. 2020. Multilingual speech translation with efficient finetuning of pretrained models. *arXiv: Computation and Language*.

Shikun Liu, Edward Johns, and A. Davison. 2019a. End-to-end multi-task learning with attention. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1871–1880.

Yuchen Liu, Hao Xiong, Zhongjun He, Jiajun Zhang, Hua Wu, Haifeng Wang, and Chengqing Zong. 2019b. End-to-end speech translation with knowledge distillation. In *Interspeech*.

K. Maninis, Ilija Radosavovic, and I. Kokkinos. 2019. Attentive single-tasking of multiple tasks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1851–1860.

M. McCloskey and N. J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165.

Ari S. Morcos, M. Raghu, and S. Bengio. 2018. Insights on representational similarity in neural networks with canonical correlation. In *NeurIPS*.

Jan Niehues, R. Cattoni, Sebastian Stüker, Matteo Negri, Marco Turchi, Elizabeth Salesky, Ramon Sanabria, Loïc Barrault, Lucia Specia, and Marcello Federico. 2019. The IWSLT 2019 evaluation campaign.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, S. Gross, Nathan Ng, David Grangier, and M. Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL*.

D. Park, W. Chan, Y. Zhang, C. Chiu, B. Zoph, E. Cubuk, and Q. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *Interspeech*.

Jonas Pfeiffer, Ivan Vulic, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. In *EMNLP*.

J. Pino, Q. Xu, X. Ma, M. Dousti, and Y. Tang. 2020. Self-training for end-to-end speech translation. In *Interspeech*.

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, Brussels, Belgium. Association for Computational Linguistics.

M. Raghu, J. Gilmer, J. Yosinski, and Jascha Sohl-Dickstein. 2017. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. In *NIPS*.

A. Renduchintala, S. Ding, M. Wiesner, and S. Watanabe. 2018. Multi-modal data augmentation for end-to-end asr. In *INTERSPEECH*.

Elizabeth Salesky and Alan W Black. 2020. Phone features improve speech translation. In *ACL*.

T. Standley, A. Zamir, Dawn Chen, L. Guibas, Jitendra Malik, and S. Savarese. 2020. Which tasks should be learned together in multi-task learning? In *ICML*.

Yun Tang, Juan Pino, Changhan Wang, Xutai Ma, and Dmitriy Genzel. 2021. A general multi-task learning framework to leverage text data for speech to text tasks. In *ICASSP*.

Simon Vandenhende, S. Georgoulis, Wouter Van Gansbeke, M. Proesmans, Dengxin Dai, and L. Gool. 2020. Multi-task learning for dense prediction tasks: A survey. *arXiv: Computer Vision and Pattern Recognition*.

C. Wang, Y. Tang, X. Ma, A. Wu, D. Okhonko, and J. Pino. 2020a. fairseq s2t: Fast speech-to-text modeling with fairseq. In *AACL (demo)*.

Chengyi Wang, Yu Wu, Shujie Liu, Zhenglu Yang, and Ming Zhou. 2020b. Bridging the gap between pretraining and fine-tuning for end-to-end speech translation. In *AAAI*.

Ron J. Weiss, Jan Chorowski, Navdeep Jaitly, Yonghui Wu, and Zhifeng Chen. 2017. Sequence-to-sequence models can directly translate foreign speech. In *INTERSPEECH*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *ACL*.

A Appendix

The detailed experimental setup for "MT" and "MT(Tuned)" in Table 5 are described as below.

We trained MT models for each language pair in "EN-DE", "EN-ES", and "EN-FR". The training data is from WMT from different years, 2013 for "EN-ES", 2014 for "EN-DE" and 2016 for "EN-FR". We use "transformer_wmt_en_de" architecture from Fairseq. The models are with embedding size 512 and feed-forward layer dimension 2048. Both encoder and decoder are with 6 transformer layers. The input is phoneme sequence and output is SentencePiece sequence. The vocabularies are shared with the corresponding speech to text translation models. The models are optimized with Adam with learning rate equal to 0.001. Beside experiments in Table 5, the trained MT models are used to initialize the jointly trained models.

We further fine-tuned the "MT" models trained from WMT data to MuST-C data sets using source transcripts and target translation labels. No speech data is used. Similar to the "MT" models, Adam optimizer with learning rate equal to 0.001 is used. The models are fine-tuned on the corresponding MuST-C data sets for 15 epochs and the checkpoints from the last 5 epochs are averaged for evaluation.