Pre-training for Speech Translation: CTC Meets Optimal Transport

Phuong-Hang Le [1] **Hongyu Gong** [2] **Changhan Wang** [2] **Juan Pino** [2] **Benjamin Lecouteux** [1] **Didier Schwab** [1]

Abstract

The gap between speech and text modalities is a major challenge in speech-to-text translation (ST). Different methods have been proposed to reduce this gap, but most of them require architectural changes in ST training. In this work, we propose to mitigate this issue at the pre-training stage, requiring no change in the ST model. First, we show that the connectionist temporal classification (CTC) loss can reduce the modality gap by design. We provide a quantitative comparison with the more common cross-entropy loss, showing that pre-training with CTC consistently achieves better final ST accuracy. Nevertheless, CTC is only a partial solution and thus, in our second contribution, we propose a novel pre-training method combining CTC and optimal transport to further reduce this gap. Our method pre-trains a Siamese-like model composed of two encoders, one for acoustic inputs and the other for textual inputs, such that they produce representations that are close to each other in the Wasserstein space. Extensive experiments on the standard CoVoST-2 and MuST-C datasets show that our pre-training method applied to the *vanilla* encoder-decoder Transformer achieves state-of-the-art performance under the no-external-data setting, and performs on par with recent strong multi-task learning systems trained with external data. Finally, our method can also be applied on top of these multi-task systems, leading to further improvements for these models.

1. Introduction

Speech-to-text translation (ST) is a challenging task that often requires training two auxiliary tasks for better performance: automatic speech recognition (ASR) and machine translation (MT). This can be achieved through either *pre-training* (Bérard et al., 2018; Bansal et al., 2019; Wang et al., 2020e;b; Inaguma et al., 2020; Le et al., 2021) or *multi-task learning* (joint-training) approaches (Anastasopoulos & Chiang, 2018; Sperber et al., 2019; Le et al., 2020; Chuang et al., 2020; Wang et al., 2020d; Tang et al., 2021a;b; Ye et al., 2022). In particular, ASR and MT pre-trainings have become arguably the most standard practice in ST development. Indeed, they have been used to obtain strong baselines in popular libraries such as ESPnet-ST (Inaguma et al., 2020) and FAIRSEQ-S2T (Wang et al., 2020b), or in standard benchmarks such as MuST-C (Di Gangi et al., 2019) and CoVoST (Wang et al., 2020a;c). Furthermore, they have also been adopted in most of the submissions to recent IWSLT evaluation campaigns (Anastasopoulos et al., 2021; 2022), as well as in strong multi-task learning systems (Tang et al., 2021a;b; 2022). Despite such ubiquity, however, this approach presents a major limitation, namely the so-called *modality gap*. Indeed, the ASR encoder is pre-trained with speech inputs, whereas the MT decoder is pre-trained with text inputs, thus plugging them together (for ST fine-tuning) will naturally result in a mismatch. This explains why simply using a pre-trained MT decoder in addition to a pre-trained ASR encoder only brings modest gains (Alinejad & Sarkar, 2020) or sometimes even worsens the performance (Bahar et al., 2019).

In this work, we make two major contributions for mitigating the modality gap without requiring any changes in the ST model. First, we show that CTC (Graves et al., 2006) is a viable solution to this problem. We advocate through theoretical analysis and extensive experiments the use of CTC for ASR pre-training over the standard cross-entropy (CE) loss. Indeed, pre-training with *vanilla* CTC (i.e., without additional components such as an external language model) is not only faster but also yields better final ST results than pre-training with CE. At first glance, this is rather counter-intuitive, given that vanilla CTC generally produces inferior ASR performance than CE (as already noticed by previous work (Bahdanau et al., 2016; Kim et al., 2017) and again confirmed by our experiments); but as we will point out, this success of CTC may be attributed to the ability of its trained encoder to align speech input to text output *without the need for a decoder*. Even so, however, CTC can only partially alleviate the modality gap. Therefore, in our second contri-

[1]Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, 38000 Grenoble, France [2]Meta AI. Correspondence to: Phuong-Hang Le <hang.le@univ-grenoble-alpes.fr>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

bution, we propose a novel pre-training method combining CTC and optimal transport (OT) (Peyré et al., 2019) to further reduce this gap. It consists in training a Siamese-like model composed of two encoders, one for acoustic inputs and the other for textual inputs, such that they produce representations (for the same sentence) close to each other in the Wasserstein space, i.e., a metric space equipped with the Wasserstein distance from OT. Furthermore, we introduce *positional encoding* to the given OT formulation in order to take into account the *monotonicity* of the inputs, which brings an extra boost in performance. The proposed method can work with or without MT pre-training, and in the former case, it can use the pre-trained MT decoder in a more effective way. Extensive experiments on the standard CoVoST-2 (Wang et al., 2020c) and MuST-C (Di Gangi et al., 2019) datasets show that our pre-training method applied to the *vanilla* encoder-decoder Transformer achieves state-of-the-art performance under the no-external-data setting. Moreover, simply increasing the model size and using our method, still without using any additional data, leads to performance that is competitive with recent strong multi-task learning systems trained with external data. Finally, our pre-training method can also be applied on top of these multi-task systems, leading to further improvements. Our code is available at `github.com/formiel/fairseq`.

2. Related Work

Speech-to-text translation The classical approach to ST consists in using *cascaded* systems composed of an ASR module followed by an MT one (Ney, 1999). This approach, however, presents important limitations such as having high latency and being susceptible to error propagation (Anastasopoulos & Chiang, 2018). Recently, much effort has been put into exploring *end-to-end* ST models (Duong et al., 2016; Berard et al., 2016; Weiss et al., 2017; Di Gangi et al., 2019; Inaguma et al., 2019; Le et al., 2020). While the first works in this direction only obtained modest results (Berard et al., 2016; Weiss et al., 2017), the most recent ones have largely closed the gap with cascaded models, or even surpassed them (Bentivogli et al., 2021; Xu et al., 2021; Ye et al., 2021; Tang et al., 2021b; Ye et al., 2022). Our work falls into the end-to-end paradigm as a generic pre-training approach that can be applied on top of existing methods.

Reducing pre-training modality gap Various methods have been studied to bridge the gap between pre-training and fine-tuning such as using cascaded encoders (Liu et al., 2020; Wang et al., 2020d; Xu et al., 2021), adaptor modules (Bahar et al., 2019; Li et al., 2020; Xu et al., 2021), or adversarial regularizer (Alinejad & Sarkar, 2020). Multi-task learning (Tang et al., 2021a;b; 2022) can also be seen as a solution to this problem. However, most of these methods do not tackle the issue at the pre-training stage but adapt the ST architecture to suit the pre-trained weights. Our method, instead, can mitigate the modality gap at the pre-training stage without requiring any changes in the ST model.

CTC for ST pre-training CTC has been used for pre-training in previous work, either alone (Wang et al., 2020d) or in combination with cross-entropy (Zhang et al., 2020; Wang et al., 2020e; Xu et al., 2021; Inaguma et al., 2021) or other losses (Bapna et al., 2022). However, no detailed quantitative comparison with cross-entropy was given and thus it was not clear how much CTC actually contributed to the obtained improvements in these works. Our work can be seen as complementary to them in terms of quantitative analysis of CTC pre-training. We believe that this is an important contribution as the effectiveness of CTC pre-training has gone relatively unnoticed by the community. For example, cross-entropy has still been used by default in the most recent multi-task learning systems (Tang et al., 2021a;b; 2022). We show that simply replacing it with CTC leads to improvements for these methods.

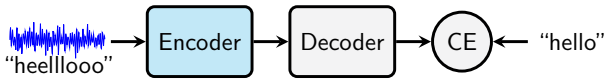
Aligning speech and text Learning to align speech and text features has been considered previously for ST, e.g., in supervised pre-training (similar to our setting) using an adversarial loss (Alinejad & Sarkar, 2020), in self-supervised pre-training (Bapna et al., 2021; 2022; Chen et al., 2022; Ao et al., 2022a;b), and in multi-task learning using Euclidean distance (Liu et al., 2020; Dong et al., 2021; Tang et al., 2021b), cosine distance (Chuang et al., 2020), Kullback–Leibler divergence (Tang et al., 2022), and contrastive loss (Han et al., 2021; Ye et al., 2022; Ouyang et al., 2022). We should point out that our architecture is conceptually simpler than those proposed in these works. Moreover, our framework can also work with OT replaced by any (differentiable) distance function, including those aforementioned. A major strength of OT, in addition to its strong theoretical properties and practical performance, is that it can *natively* handle sequences of different lengths, whereas Euclidean distance and Kullback–Leibler divergence require some length-matching mechanism such as average pooling (Liu et al., 2020; Dong et al., 2021) or attention (Tang et al., 2021b). Finally, we should note that speech-text alignment has been used before in other speech applications beyond ST, such as ASR (Juang, 1984; Berndt & Clifford, 1994), text-to-speech (Haubold & Kender, 2007), or speaker identification (Yuan et al., 2008; Muda et al., 2010), with many of them employing dynamic time warping (DTW) (Sakoe & Chiba, 1978). A more complete survey on this topic can be found in Liang et al. (2022, Section 4).

OT for speech translation OT has widely been used in MT (Alqahtani et al., 2021; Alvarez-Melis et al., 2019; Alvarez-Melis & Jaakkola, 2018; Grave et al., 2019; Chen et al., 2019) and also in speech processing, e.g., for speech enhancement (Lin et al., 2021). OT has not been used for speech translation, to the best of our knowledge.

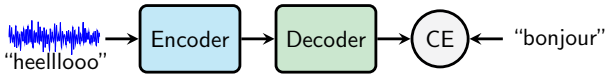
3. The Modality Gap in ST Pre-training

We review the most standard pre-training recipe for ST (namely ASR and MT pre-training, Figure 1) and discuss the induced gap between pre-training and fine-tuning. In a few words, we pre-train ASR and MT using the standard encoder-decoder architecture with cross-entropy loss, then the ASR encoder and MT decoder are used to initialize the corresponding ST components. Note that MT pre-training is optional as it does not necessarily improve the performance, as discussed in Section 1. Here we assume that the ST model is also the vanilla encoder-decoder. In more sophisticated models such as multi-task learning (Tang et al., 2021b; Ye et al., 2022), all pre-trained components are typically used.

(a) **MT pre-training (optional)** Standard encoder-decoder with cross-entropy (CE) loss. The encoder is often discarded for ST.



(b) **ASR pre-training** Standard encoder-decoder with CE loss. The decoder is often discarded for ST fine-tuning.



(c) **ST fine-tuning** Using ASR encoder and MT decoder (if available) for initialization.

Figure 1. Standard pre-training recipe for ST.

In an attention-based encoder-decoder model, the decoder typically learns to *align* the output with the input (Bahdanau et al., 2015). In the above recipe, the two components of ST are pre-trained to be aligned with other components that are later discarded, causing a *loss of alignment information*. This explains the modality discrepancy between pre-training and fine-tuning. In the next section, we provide an explanation why CTC can partially solve this problem.

4. ASR Pre-training with CTC

In this section, we revisit CTC (Graves et al., 2006) for ASR pre-training and explain why it can be seen as a partial solution to the modality gap issue.

4.1. Review of CTC

Given an input sequence of (typically pre-processed and downsampled) audio features $\mathbf{X} \triangleq (\mathbf{x}_1, \dots, \mathbf{x}_S)$ (in some language, e.g., English), the task of ASR consists in predicting its transcription (in the same language), represented by an output sequence $\mathbf{y} \triangleq (y_1, \dots, y_T)$, where $y_t \in \mathcal{V} \triangleq \{1, 2, \dots, V\}$, a vocabulary of size V. Suppose that $\mathbf{X}$ is processed by an encoder to obtain

$$\mathbf{H} \triangleq (\mathbf{h}_1, \dots, \mathbf{h}_S) \triangleq \textsc{encode}(\mathbf{X}; \boldsymbol{\theta}_{\text{enc}}), \tag{1}$$

where $\mathbf{h}_t \in \mathbb{R}^d$ is the hidden feature vector at time step $t \in \{1, \dots, S\}$, and $\boldsymbol{\theta}_{\text{enc}}$ is the parameters of the encoder. The idea of CTC is to predict a token $\hat{a}_t \in \mathcal{V}$ for each time step t based on $\mathbf{h}_t$:

$$\begin{aligned} p(a_t \,|\, \mathbf{X}) &= \text{softmax}(\mathbf{W}\mathbf{h}_t + \mathbf{b})[a_t] \; \forall a_t \in \mathcal{V}, \\ \hat{a}_t &= \operatorname*{argmax}_{a_t \in \mathcal{V}} p(a_t \,|\, \mathbf{X}), \end{aligned} \tag{2}$$

where $\mathbf{W} \in \mathbb{R}^{V \times d}, \mathbf{b} \in \mathbb{R}^V$ are the weights and biases of the final linear layer, and $\mathbf{v}[i]$ denotes the ith element of a vector $\mathbf{v}$. The vector $\hat{\mathbf{a}} \triangleq (\hat{a}_1, \dots, \hat{a}_S)$ is called an *alignment*. Clearly, this produces a sequence of tokens of the same length as the input, which is not desirable. CTC solves this by applying a collapsing function, $\mathbf{y} = \textsc{collapse}(\hat{\mathbf{a}})$, that removes consecutive repetitions. It is assumed that the vocabulary contains a special *blank* token ("␣", which the model can predict), and collapsing only happens in-between blanks and not across them:

$$\begin{aligned} \textsc{collapse}(\texttt{heellllloooo}) &= \texttt{helo} \\ \textsc{collapse}(\texttt{he␣ll␣l␣oo␣␣}) &= \texttt{hello}. \end{aligned}$$

The issue is that different $\hat{\mathbf{a}}$ may collapse to the same $\mathbf{y}$, and thus the most likely assignment (given by (2)) may not correspond to the most probable final output. To obtain the latter, CTC actually computes the highest sum over the probability of all its possible alignments using the Viterbi algorithm (Viterbi, 1967). We refer the reader to Graves et al. (2006) for further details.

4.2. CTC can reduce modality gap in pre-training

CTC has been used for pre-training in the ST literature, either alone (Wang et al., 2020d) or with cross-entropy (CE) (Zhang et al., 2020; Wang et al., 2020e; Xu et al., 2021; Inaguma et al., 2021). This is illustrated in Figure 2 (compare with Figure 1b).

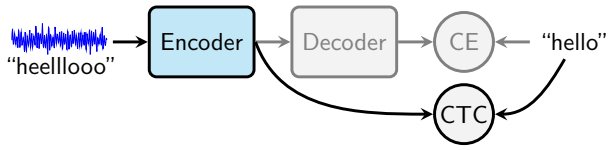


Figure 2. Two flavors of ASR pre-training with CTC: with and without CE. In the latter, the CE branch (in gray) is not present.

We will see later in the experiments that CTC completely outperforms CE in terms of ST accuracy, while being on par with CE+CTC, which indicates its importance for pre-training. There is a simple explanation for this. On one

hand, recall from our discussion in Section 3 that the discrepancy between pre-training and ST fine-tuning is caused by the loss of alignment information after discarding the ASR decoder and the MT encoder. On the other hand, recall from Section 4.1 that **the ASR encoder trained with CTC already learns to align speech input to text output** without a decoder, which means that *no alignment information will be lost* when the encoder is used for ST fine-tuning, as the encoder already contains this information. That is, even though the model trained with CE is stronger (than CTC) in general (Bahdanau et al., 2016; Kim et al., 2017), the contribution of the CE decoder to the overall performance could be so important that removing it would make the CE encoder (alone) weaker than the CTC encoder. This is indeed confirmed by our experiments (Appendix C.1).

Finally, CTC obviously cannot recover the loss of alignment information when discarding the MT encoder, and thus it should be considered only as a partial solution. In the next section, we present a method for filling this gap.

5. Optimal Transport for Pre-training

This section presents our core contribution for reducing the modality gap in ST. The idea is to train the speech encoder to generate representations that are close to those produced by a text encoder. The challenge here is that, given the same sentence, its speech features typically have a *much longer sequence length* than its text features, which makes it difficult to "compare" them. The Wasserstein distance from optimal transport (OT) turns out to be a suitable solution.

5.1. Review of discrete optimal transport

We first give a brief review of OT (Peyré et al., 2019).

Let α be a discrete probability distribution represented by positive masses $a_1, \dots, a_m$ (where $a_1 + \dots + a_m = 1$) at locations $\mathbf{u}_1, \dots, \mathbf{u}_m \in \mathbb{R}^d$, respectively (i.e., a_i is the quantity of mass at $\mathbf{u}_i$). Suppose we want to form a new distribution β with masses $b_1, \dots, b_n$ (where $b_1 + \dots + b_n = 1$) at new locations $\mathbf{v}_1, \dots, \mathbf{v}_n \in \mathbb{R}^d$ by transporting all the masses from α to β. Suppose that the *cost* of transporting a unit of mass from $\mathbf{u}_i$ to $\mathbf{v}_j$ is given by $c(\mathbf{u}_i, \mathbf{v}_j)$, where $c : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ is some function. Let $Z_{ij} \geq 0$ be the quantity of mass to be transported from $\mathbf{u}_i$ to $\mathbf{v}_j$, which induces a cost of $Z_{ij} c(\mathbf{u}_i, \mathbf{v}_j)$. The total transportation cost is thus $\sum_{i=1}^m \sum_{j=1}^n Z_{ij} c(\mathbf{u}_i, \mathbf{v}_j)$. The total quantity of mass that β receives from $\mathbf{u}_i$ is $\sum_{j=1}^n Z_{ij}$, which must equal the mass stored at $\mathbf{u}_i$, thus $\sum_{j=1}^n Z_{ij} = a_i$. Similarly, the total quantity of mass that $\mathbf{v}_j$ receives from α is $\sum_{i=1}^m Z_{ij}$, and thus $\sum_{i=1}^m Z_{ij} = b_j$. OT consists in finding the *transportation plan* $\mathbf{Z}^*$ that has the minimum cost:

$$\min_{\mathbf{Z}} \langle \mathbf{C}, \mathbf{Z} \rangle \text{ s.t. } \mathbf{Z}\mathbf{1}_n = \mathbf{a}, \mathbf{Z}^\top \mathbf{1}_m = \mathbf{b}, \mathbf{Z} \geq \mathbf{0}, \quad (3)$$

where $\mathbf{1}_n$ denotes the n-dimensional vector of ones, $\mathbf{a} = (a_1, \dots, a_m)$, $\mathbf{b} = (b_1, \dots, b_n)$, $\mathbf{Z}$ and $\mathbf{C}$ denote the $m \times n$ matrices whose elements are Z_{ij} and $C_{ij} = c(\mathbf{u}_i, \mathbf{v}_j)$.

Wasserstein loss Let $\mathbf{Z}^*$ denote the optimal solution to (3). If we define $W(\alpha, \beta) = \langle \mathbf{C}, \mathbf{Z}^* \rangle$ (i.e., the minimum transportation cost), then W can be seen as a distance measure between α and β, and is called the *Wasserstein distance*. In theory, one can use W as a loss function (called the *exact* Wasserstein loss (Frogner et al., 2015)) because it is differentiable almost everywhere. However, evaluating this loss requires solving (3), which is expensive in practice. It is typically better to work with an upper-bound approximation of W, defined as (subject to the same constraints as in (3))

$$W_\lambda(\alpha, \beta) = \min_{\mathbf{Z}} \{ \langle \mathbf{C}, \mathbf{Z} \rangle - \lambda H(\mathbf{Z}) \}, \quad (4)$$

where H is the entropy function and $\lambda > 0$ is a regularization weight. The function W_λ is not only fully differentiable but also very efficient to evaluate using the so-called Sinkhorn algorithm (Sinkhorn & Knopp, 1967). We refer the reader to Cuturi (2013) and Frogner et al. (2015) for details. From now on, "Wasserstein distance" (or loss) refers to this regularized variant W_λ.

5.2. Learning to align speech and text features

In this section, we present our proposed model for learning to align speech and text features. Recall that our goal was to mitigate the modality gap issue arising in ST, which involves the discrepancy between acoustic and textual representations. To this end, we propose an architecture composed of two encoders, one for speech inputs and the other for text inputs. Then, given an input pair of an audio sequence and its transcript, we feed them to the corresponding encoders and train the model to produce features that are close to each other in terms of Wasserstein distance. In addition, as ASR data (audio together with transcripts) are assumed to be available for this model, we make further use of them to enrich the learned speech representations by jointly training with a CTC loss. This is summarized in Figure 3.

Wasserstein distance between speech and text We have mentioned the Wasserstein distance between two sets of features, without having formally defined it (note that the previous section only presents the Wasserstein distance between two probability distributions). Let $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_m)$ and $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_n)$ be the output speech and text features, respectively, where $\mathbf{u}_i \in \mathbb{R}^d$ and $\mathbf{v}_j \in \mathbb{R}^d$ for all i, j. Here we have assumed that the hidden dimensions of the two encoders are the same and equal to d, but the sequence lengths m and n can be different (typically m is much larger than n). Define two distributions α and β whose masses are uniformly distributed at locations $(\mathbf{u}_1, \dots, \mathbf{u}_m)$ and $(\mathbf{v}_1, \dots, \mathbf{v}_n)$, respectively; here uniform distribution means

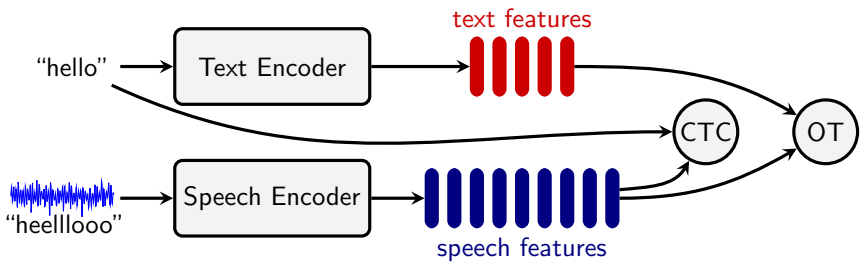


Figure 3. Our proposed Siamese-like architecture for learning to align speech and text representations. An input pair of audio sequence and its transcript are fed to the corresponding speech and text encoders, then their outputs are compared using optimal transport (OT). The speech features are enhanced by jointly training with CTC.

that $a_i = \frac{1}{m}$ and $b_j = \frac{1}{n}$ for all i, j, where $\mathbf{a}$ and $\mathbf{b}$ are defined as in the previous section. Define the cost of transporting a unit of mass from $\mathbf{u}_i$ to $\mathbf{v}_j$ as $c(\mathbf{u}_i, \mathbf{v}_j) = \|\mathbf{u}_i - \mathbf{v}_j\|_p$ for some $p \geq 1$ (typically $p = 2$). Then, the Wasserstein distance $W_\lambda(\alpha, \beta)$ (see (4)) can be seen as the discrepancy between $\mathbf{U}$ and $\mathbf{V}$. Without ambiguity, we refer to this quantity as the Wassertein distance between these two sequences. The obtained optimal transportation plan can then be seen as an alignment map between the two sequences.

Positional encoding for OT An important property of the above Wasserstein distance between two sets of features is that it does not take into account the sequence orders. Indeed, if we replace one of the sequences by any of its permutations, then the Wasserstein distance remains the same. While this property can be useful in many situations, taking into account the sequence orders could be beneficial in our case due to the nature of the task: the inputs to our encoders are *monotonically* aligned. That is, the first audio frames should represent the same part of the sentence as the first text tokens do, and vice-versa (note that this is true for ASR but not for MT or ST). We propose to integrate this prior information into the OT model as follows. The idea is to modify the cost matrix $\mathbf{C}$ such that transporting from one location to another will have a high cost if their positions in the corresponding sequences are very different. For example, transporting from $\mathbf{u}_1$ to $\mathbf{v}_1$ (or from $\mathbf{u}_m$ to $\mathbf{v}_n$) should induce a low cost while transporting from $\mathbf{u}_1$ to $\mathbf{v}_n$ should induce a high cost. As the input sequence lengths can be very different, we normalize them to unit length so that the first element of the sequence has position 0 and the last has position 1. That is, the position vectors $(1, 2, \ldots, m)$ and $(1, 2, \ldots, n)$ will be normalized to, respectively, $(s_1, s_2, \ldots, s_m)$ and $(t_1, t_2, \ldots, t_n)$, where

$$s_i = \frac{i-1}{m-1} \quad \text{and} \quad t_j = \frac{j-1}{n-1}. \tag{5}$$

Then, a simple way of including the positional constraint into the cost matrix could be to define, e.g., $C_{ij} = c(\mathbf{u}_i, \mathbf{v}_j) + \gamma\,|s_i - t_j|$ (where $\gamma > 0$ is a weight scalar that can be empirically tuned), which would penalize any mismatch in terms of positions.[1] However, this may cause inefficiency in practice. Indeed, in our case c is an ℓ_p-norm for some $p \geq 1$ (according to the previous paragraph), for which the Wasserstein distance can be evaluated very efficiently without computing and storing the matrix $\mathbf{C}$ explicitly (Feydy et al., 2019); the above modification of $\mathbf{C}$ could make C_{ij} no longer an ℓ_p-norm (unless $p = 1$), thus losing this efficiency. Therefore, given that c is an ℓ_p-norm, we propose to modify the cost matrix as follows:

$$C_{ij} = \left(\|\mathbf{u}_i - \mathbf{v}_j\|_p^p + \gamma^p\,|s_i - t_j|^p\right)^{1/p}. \tag{6}$$

Put $\mathbf{u}'_i = [\mathbf{u}_i; \gamma s_i]$ and $\mathbf{v}'_j = [\mathbf{v}_j; \gamma t_j]$, then the above is just $\|\mathbf{u}'_i - \mathbf{v}'_j\|_p$. That is, we have constructed a new feature vector at each sequence element by appending its normalized position to its existing feature vector, so that performing OT on these new features is equivalent to performing it on the original features using the new cost matrix defined by (6).

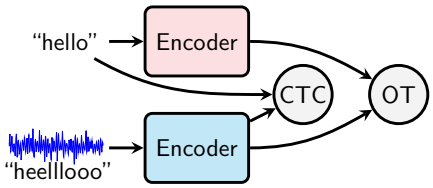
Beyond optimal transport Clearly, the proposed method can also work with OT replaced by any differentiable loss function, including (soft) DTW (Cuturi & Blondel, 2017), adversarial loss (Ganin et al., 2016), Euclidean distance, or KL divergence (the latter two cannot work directly on sequences of different lengths but require some length-matching mechanism such as average pooling (Liu et al., 2020; Dong et al., 2021) or attention (Tang et al., 2021b); see also Section 2 and Appendix A for detailed descriptions of these methods). We will see in later experiments that OT turns out to have the best practical performance (in addition to its strong theoretical properties, such as being a metric).

5.3. Proposed recipes for speech translation

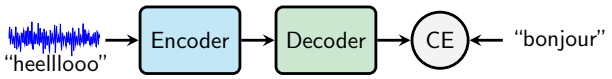
Equipped with the above method for learning to align acoustic and textual representations, we are now ready to present

[1] It is worth pointing out that, even though the alignment of the speech and text features is supposed to be monotonic, it is not necessarily *linear*. For example, the speech features at (normalized) position 0.5 need not correspond to the text features at position 0.5. Thus, our mismatch cost can be viewed as favoring a linear approximation of the true alignment (which is unknown).

(a) **MT pre-training** Standard and same as Figure 1a.



(b) **ASR pre-training** Speech encoder and text encoder learn to produce similar representations in the Wasserstein space. The text encoder is initialized with the pre-trained MT encoder.



(c) **ST fine-tuning** Both pre-trained ASR encoder and MT decoder are used for initialization.

Figure 4. Our proposed pre-training recipe for ST.

our proposed training recipe for ST. As shown in Figure 4, our recipe differs from the standard one (Figure 1) only at the ASR pre-training stage. We follow the above method (Section 5.2) and train a Siamese-like model using CTC and OT. Here the text encoder is initialized with the pre-trained MT encoder, which helps the speech encoder learn to align with the text decoder right at the pre-training stage. Note that freezing the text encoder during ASR pre-training also produced good results in our experiments, though not as good as training both encoders. In addition, a simplified variant of our recipe can also be obtained by omitting MT pre-training. In that case, the text encoder in ASR pre-training is initialized randomly. As shown later in Section 6, this recipe also outperforms ASR pre-training with CTC and cross-entropy, indicating the effectiveness of OT.

6. Experiments

In this section, we provide empirical evidence for the effectiveness and versatility of our proposed method. Section 6.1 outlines the experimental setup, while Section 6.2 conducts ablation studies to justify our various design choices. Comparative results of the presented pre-training methods are discussed in Section 6.3, and comparison to state-of-the-art methods is presented in Section 6.4. Finally, Section 6.5 demonstrates that our pre-training method can also be applied to multi-task learning, showing its generality.

6.1. Experimental setup

Datasets We evaluate the pre-training methods presented in this paper on the standard **MuST-C** (Di Gangi et al., 2019) and **CoVoST-2** (Wang et al., 2020c) datasets. MuST-C is a large-scale one-to-many ST dataset built from audio recordings of TED Talks, covering pairs from English (En) to 8 European languages: Dutch (Nl), French (Fr), German (De), Italian (It), Portuguese (Pt), Romanian (Ro), Russian (Ru), and Spanish (Es). Each direction includes a triplet of speech, transcription, and translation. CoVoST-2 is a large and diversified multilingual ST corpus based on the Common Voice project (Ardila et al., 2020). It covers translations from 21 source languages into English and from English into 15 target languages, many of which involves non-European and very low-resource languages. We refer the reader to the original papers for further details. It is important to note that all our analyses are conducted on the *dev* splits of these datasets to prevent overfitting their *test* sets. Then, only the best-performing models will be selected for comparison with existing methods on the *test* sets.

Training settings While we focus on *multilingual* translation (i.e., a single model for all language pairs), our ablation analysis is conducted under the *bilingual* setting with only two language pairs (one model for each), due to the high training cost. For pre-training, we consider two settings: *with* and *without* MT pre-training. In the latter, some components are initialized randomly (see Figures 1 and 4).

Implementation details Our implementation is based on the FAIRSEQ S2T toolkit (Wang et al., 2020b). We follow closely previous work (Wang et al., 2020b;c) for setup, including models, data processing, training and evaluation settings (see Appendix B for details). In particular, we use a model whose speech encoder, speech decoder, and text decoder have respectively 12, 6, and 6 layers. For the analysis, we use a *medium* architecture with hidden dimension $d = 512$. In the final experiments where we aim to reach state-of-the-art performance for comparison with existing methods, we also use the *large* variant where $d = 1024$.

Loss functions As CTC plays a crucial role in the final ST performance, when training it jointly with another loss (CE or OT), we keep it unchanged and scale the latter, i.e., the final loss will be either CTC $+ \alpha$CE or CTC $+ \alpha$OT. We performed a grid search for α among $[0.1, 0.2, 0.5, 1.0]$ on MuST-C En-De and found that $\alpha = 0.1$ works well for both CE and OT (the other values yielded similar performance). Therefore, we set $\alpha = 0.1$ in all experiments. Note that previous work (Wang et al., 2020e; Zhang et al., 2020) used 0.7CE $+ 0.3$CTC, but we found that this combination performs worse than our choice (see Table 10 in Appendix C.2), which again indicates the importance of CTC. For the Wasserstein loss, we use the efficient GPU implementation by Feydy et al. (2019) with the default parameters (regularization weight $\lambda = 1$ and ℓ_2 cost function). Tuning these could further boost the performance, but we use the default for simplicity. For the weight γ of OT positional encoding (see (6)), after doing a quick grid search

among $[0.1, 0.5, 1.0, 5.0, 10.0]$ on a small training subset of MuST-C En-De, we found that $\gamma = 1.0$ works reasonably well and thus we use this value in all experiments.

6.2. Ablation analysis

We first conduct ablation studies to support some of our claims and to validate the rationale behind various design choices for our method. The following results are obtained under the *bilingual* setting using two language pairs (En-De and En-Fr) of the MuST-C *dev* sets, and *with-MT* setting. More ablation results can be found in Appendix C.

ASR vs. ST performance The results shown in Table 1 strongly support our claim that CTC obtains better final ST accuracy than CE (and not far off from CTC+CE) despite its inferior ASR performance (recall our claim that CTC can reduce modality gap). Comparing CTC and CTC+OT, we see that OT could only improve WER very slightly (0.1-0.2 points), whereas the gain in the final ST performance is more important (0.4-0.7 points). This indicates that OT has a bigger impact on the final ST performance than merely improving WER during ASR pre-training.

Table 1. BLEU and WER (in parentheses) on MuST-C *dev* sets.

Method	En→De	En→Fr	Training time*
CE	23.04 (13.1)	28.89 (12.6)	7.10h
CTC	24.08 (17.7)	29.91 (17.3)	6.27h
CTC+CE	24.28 (**12.9**)	30.21 (**11.9**)	7.78h
CTC+OT	**24.74** (17.6)	**30.31** (17.1)	7.62h

*100 epochs on En-Fr, batch size 40K tokens, 8 V100 GPUs

Table 2. BLEU and WER (in parentheses) on MuST-C *dev* sets for different distance functions in Siamese pre-training.

Metric	Length-match	En→De	En→Fr
(CTC alone)	-	24.08 (17.7)	29.91 (17.3)
Euclidean	average	24.41 (18.1)	29.86 (17.3)
	attention	24.30 (18.6)	29.81 (19.2)
	interpolation	23.94 (19.3)	29.77 (17.8)
KL-diverg.	attention	24.56 (18.3)	30.10 (**17.1**)
	interpolation	24.26 (18.1)	29.96 (17.4)
Adversarial	-	23.73 (20.6)	29.98 (19.6)
Wasserstein	-	**24.74** (**17.6**)	**30.31** (**17.1**)

OT vs. other distances As discussed in Section 5.2, our Siamese pre-training can use different distance functions than OT (Appendix A), including Euclidean distance, KL divergence, Soft-DTW (Cuturi & Blondel, 2017), or adversarial loss (Ganin et al., 2016; Lample et al., 2018; Alinejad & Sarkar, 2020). The first two require a length-matching operation to be applied to the sequences, for which we experiment with average pooling (Liu et al., 2020; Dong et al., 2021), cross-attention (Tang et al., 2021b), and linear interpolation (ours). Soft-DTW is excluded due to its prohibitive memory footprint. For all methods we perform the same hyper-parameter search (Section 6.1). Table 2 shows that Euclidean distance and adversarial loss do not always improve over CTC,[2] while KL divergence and OT consistently improve over that baseline, with OT being the best.

Different variants of Siamese pre-training As discussed in Section 5.3, our proposed method can have different variants: weight sharing between the encoders,[3] and using (or not) positional encoding for OT (Section 5.2). From the results in Table 3, we observe that, without MT pre-training, sharing the encoders tends to give slightly better results, while it is the opposite with MT pre-training (but with a larger margin). Using OT positional encoding gives better results in most cases. Therefore, we disable weight sharing and use positional encoding in the remaining experiments.

Table 3. BLEU for different variants of Siamese-PT.

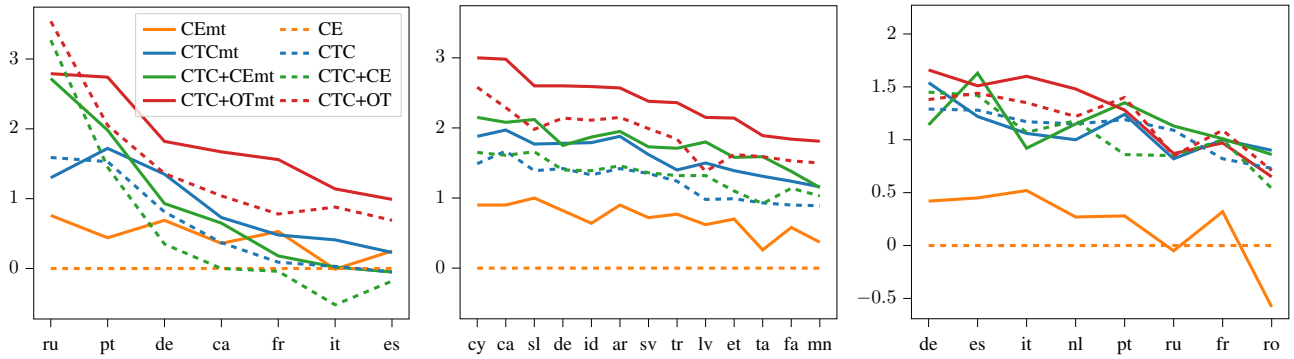
	Shared	Positional	En-De	En-Fr
w/o MT	-	-	23.69	29.69
	✓	-	**24.12**	29.73
	-	✓	23.91	**29.83**
	✓	✓	23.89	**29.86**
with MT	-	-	24.29	30.04
	✓	-	23.99	29.68
	-	✓	**24.74**	**30.31**
	✓	✓	24.29	29.64

6.3. Comparative results on the dev sets

In this section, we compare the performance of the presented pre-training methods (CE, CTC, CTC+CE, and CTC+OT). A preliminary comparison has appeared in the previous section (Table 1), but under a restricted setting (*bilingual* and *with-MT*). Here we provide more extensive results where we also consider the *multilingual* (our focus) and *without-MT* settings. As the number of configurations is still high at this point, we keep using the *dev* sets instead of the *test* sets for performance evaluation. Figure 5 presents the results in terms of relative BLEU with respect to the baseline CE, and Table 4 shows absolute BLEU. For the very low-resource languages in CoVoST-2 many-to-one (X→En), the results are very poor as no external data is used (0.13–4.56 points, see Table 13 in Appendix D), which makes the variance too high to draw comparisons. Therefore, we exclude

[2]Our results of adversarial loss differs from Alinejad & Sarkar (2020) but in line with Lample et al. (2018). See `github.com/facebookresearch/UnsupervisedMT/issues/40`.

[3]The speech encoder has 6 layers more than the text one, so only the last 6 are shared.



(a) CoVoST-2 X*→En. *(b)* CoVoST-2 En→X. *(c)* MuST-C.

Figure 5. Results on the *dev* sets, relative to CE. The postfix "mt" in the legends indicates that *MT pre-training* was performed (see Figures 1a and 4a). Best viewed in color. Detailed results can be found in Tables 12–14 in Appendix D.

these languages from the presentation, indicated by an asterisk: X*→En (see Table 13 for complete results). We observe that CTC outperforms CE while being competitive with CTC+CE. In general, CTC+OT is the best performing method and is competitive with all other recipes even without MT pre-training.

Table 4. BLEU on the *dev* sets. The prefixes "(b)" and "(m)" mean bilingual and multilingual (on average). See Tables 12–14 in Appendix D for detailed results.

	Method	MuST-C			CoVoST-2	
		(b)**De**	(b)**Fr**	(m)**avg**	**En→X**	**X*→En**
w/o MT	CE	22.39	28.48	25.42	20.50	19.64
	CTC	23.25	**29.87**	26.51	21.71	20.27
	CTC+CE	23.77	29.74	26.47	21.83	20.26
	CTC+OT	**23.91**	29.83	**26.60**	**22.36**	**21.12**
with MT	CE	23.04	28.89	25.62	21.17	20.07
	CTC	24.08	29.91	26.52	22.10	20.53
	CTC+CE	24.28	30.21	26.57	22.22	20.56
	CTC+OT	**24.74**	**30.31**	**26.67**	**22.82**	**21.46**

6.4. Comparison to state-of-the-art methods

In this section, we compare our *multilingual* models (with MT pre-training) to state-of-the-art methods. Recall that we only use a simple *vanilla* encoder-decoder architecture. Tables 5 and 6 show the results on CoVoST-2 and MuST-C (*test* sets), respectively. For reference, we name our proposed pre-training method **Siamese-PT** (CTC+OT). We should note that due to the low-resource nature of CoVoST-2, a comparison to methods using external data would be highly unfair. Therefore, we only compare against Wang et al. (2020c) whose results were obtained under the same settings as ours. Our proposed method can certainly use external data, which is left for future work.

We observe the effectiveness of CTC pre-training from the MuST-C results in Table 6. It surpasses CE by 0.4 points (29.2 vs. 28.8; for the medium model the gap is 1.2 points, see Table 17); and adding CE to CTC did not help. Siamese pre-training with OT helps improve BLEU score to 29.8. Moreover, we have improved from the previous best results in the no-external-data and multilingual settings (Le et al., 2021) (4th row) by 3.2 points (29.8 vs. 26.6). Finally, our best result even surpasses previous strong and sophisticated multi-task learning systems (Tang et al., 2021b; Ye et al., 2022) that were trained on external data. The effectiveness of CTC and OT is again confirmed on CoVoST, as shown in Table 5, where Siamese-PT also achieved the best results.

Table 5. BLEU on CoVoST-2 *test* set. "X**→En" denotes the average excluding very low-resource languages (see Section 6.3) and those not reported in Wang et al. (2020c, Table 3). See Tables 15 and 16 (Appendix D) for details. The reported results are obtained using large model for En→X and medium model for X**→En, which is better than the large one (Wang et al., 2020c).

Method	En→X	X**→En
Wang et al. (2020c)	19.4	24.5
CE	19.2	24.6
CTC	19.8	24.7
CTC+CE	19.7	24.5
Siamese-PT (this work)	**21.5**	**25.5**

6.5. Application to multi-task learning

As a proof of concept of the wide applicability of our method, we apply it on top of the multi-task learning system by Tang et al. (2021b), where they pre-trained the ASR component with CE loss. We replicate their results on MuST-C En-De and perform further experiments using different ASR pre-training methods discussed in the paper. The results are shown in Table 7. We observe again the superiority of CTC over CE. Simply replacing CE with CTC yields an improvement of 0.26 points, whereas using them together slightly decreases the performance. Our Siamese-PT reached the

Table 6. Performance on MuST-C test set. Due to space constraints, only the *large* model and only some existing methods are included. Our results for the *medium* model are also competitive. See Table 17 in Appendix D for a complete list. The column "Multi" denotes multilingual models (instead of individual bilingual ones).

Method		Multi	External Data		BLEU								
			Unlabeled	*Labeled*	de	es	fr	it	nl	pt	ro	ru	avg
FAIRSEQ S2T (Wang et al., 2020b)		✓	-	-	24.5	28.2	34.9	24.6	28.6	31.1	23.8	16.0	26.5
ESPnet-ST (Inaguma et al., 2020)		✓	-	-	22.9	28.0	32.7	23.8	27.4	28.0	21.9	15.8	25.1
Dual-decoder (Le et al., 2020)		✓	-	-	23.6	28.1	33.5	24.2	27.6	30.0	22.9	15.2	25.6
Adapters (Le et al., 2021)		✓	-	-	24.7	28.7	35.0	25.0	28.8	31.1	23.8	16.4	26.6
BiKD (Inaguma et al., 2021)		-	-	-	25.3	-	35.3	-	-	-	-	-	-
JointSpeechText (Tang et al., 2021b)		-	-	✓	26.8	31.0	37.4	-	-	-	-	-	-
TaskAware (Indurthi et al., 2021)		-	-	✓	**28.9**	-	-	-	-	-	-	-	-
ConST (Ye et al., 2022)		-	✓	✓	28.3	32.0	38.3	27.2	**31.7**	33.1	25.6	**18.9**	29.4
STPT (Tang et al., 2022)		-	✓	✓	-	**33.1**	**39.7**	-	-	-	-	-	-
CE pre-training	MEDIUM	✓	-	-	24.6	28.7	34.9	24.6	28.4	30.7	23.7	15.9	26.4
CTC pre-training	MEDIUM	✓	-	-	25.9	29.7	36.6	25.6	29.6	32.0	24.6	16.7	27.6
CTC+CE pre-training	MEDIUM	✓	-	-	25.6	29.5	36.4	25.2	29.5	31.6	24.5	16.5	27.4
Siamese-PT (this work)	MEDIUM	✓	-	-	26.2	29.8	36.9	25.9	29.8	32.1	24.8	16.8	27.8
CE pre-training	LARGE	✓	-	-	26.9	30.8	37.7	26.7	30.8	33.3	26.2	17.9	28.8
CTC pre-training	LARGE	✓	-	-	27.6	31.4	38.2	27.2	31.1	33.6	26.4	18.4	29.2
CTC+CE pre-training	LARGE	✓	-	-	27.2	31.2	38.0	27.0	31.5	33.7	26.2	18.3	29.1
Siamese-PT (this work)	LARGE	✓	-	-	27.9	31.8	39.2	**27.7**	**31.7**	**34.2**	**27.0**	18.5	**29.8**

highest accuracy with an improvement of 0.46, showing that our pre-training method is not only effective for vanilla ST but also complementary to multi-task learning.

Table 7. Results of the multi-task learning system of Tang et al. (2021b), using different ASR pre-training methods.

ASR pre-training method	En-De
CE (reported in Tang et al. (2021b))	26.74
CE (reproduced)	26.78
CTC	27.04
CTC+CE	26.69
Siamese-PT (CTC+OT, proposed)	**27.20**

7. Conclusion

In this paper, we have discussed the gap between speech and text modalities as a challenge in speech-to-text translation, and proposed to overcome it at the pre-training stage. We have shown that ASR pre-training using the CTC loss can reduce this modality gap, but only partially. To further mitigate this issue, we have proposed a novel pre-training method in which we train a Siamese-like model composed of a speech encoder and a text encoder, such that they produce representations that are close in the Wasserstein space. Extensive experiments on two popular ST datasets have demonstrated the effectiveness of our method. In particular, our results surpassed previous work in the non-external-data setting, while being on par with recent strong multi-task learning systems trained with external data. Our pre-training method can also be applied to the multi-task learning system, yielding further improvements for these methods.

Limitations and future work Even though the proposed pre-training recipe (Section 5.3) can use additional ASR and MT data, we have only considered the setting without external data. While this setting is important in many real-world scenarios, by restricting to it we have missed the opportunity to show the effectiveness of the proposed method on very low-resource languages, as discussed in Section 6.3. On the other hand, the proposed method for learning to align speech and text features (Section 5.2) is quite generic and has potential applications beyond pre-training for ST. For example, the proposed OT speech-text alignment could be, in principle, used as an additional loss in multi-task learning. We aim to address all these open questions in future work.

Societal impact The technology developed in this work can be used in surveillance systems, which might raise privacy concerns. Therefore, considerations should be taken when releasing or deploying these models to the public.

Acknowledgements

This work was supported by a Meta AI SRA grant and partially by the Multidisciplinary Institute in Artificial Intelligence (MIAI@Grenoble Alpes, ANR-19-P3IA-0003). It was granted access to the HPC/AI resources of IDRIS under the allocations 2022-AD011012538 and 2022-AD011013744 made by GENCI. We thank Yun Tang and Anne Wu for their help in reproducing the results reported in Tang et al. (2021b) and Wang et al. (2020c), respectively. We also thank Aidan Mannion and the anonymous reviewers for their feedback that helped greatly improve the manuscript.

References

Alinejad, A. and Sarkar, A. Effectively pretraining a speech translation decoder with machine translation data. In *EMNLP (1)*, pp. 8014–8020. Association for Computational Linguistics, 2020.

Alqahtani, S., Lalwani, G., Zhang, Y., Romeo, S., and Mansour, S. Using optimal transport as alignment objective for fine-tuning multilingual contextualized embeddings. *CoRR*, abs/2110.02887, 2021.

Alvarez-Melis, D. and Jaakkola, T. S. Gromov-wasserstein alignment of word embedding spaces. In *EMNLP*, pp. 1881–1890. Association for Computational Linguistics, 2018.

Alvarez-Melis, D., Jegelka, S., and Jaakkola, T. S. Towards optimal transport with global invariances. In *AISTATS*, volume 89 of *Proceedings of Machine Learning Research*, pp. 1870–1879. PMLR, 2019.

Anastasopoulos, A. and Chiang, D. Tied multitask learning for neural speech translation. In Walker, M. A., Ji, H., and Stent, A. (eds.), *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pp. 82–91. Association for Computational Linguistics, 2018. doi: 10.18653/v1/n18-1008. URL https://doi.org/10.18653/v1/n18-1008.

Anastasopoulos, A., Bojar, O., Bremerman, J., Cattoni, R., Elbayad, M., Federico, M., Ma, X., Nakamura, S., Negri, M., Niehues, J., Pino, J. M., Salesky, E., Stüker, S., Sudoh, K., Turchi, M., Waibel, A., Wang, C., and Wiesner, M. Findings of the IWSLT 2021 evaluation campaign. In *IWSLT*, pp. 1–29. Association for Computational Linguistics, 2021.

Anastasopoulos, A., Barrault, L., Bentivogli, L., Boito, M. Z., Bojar, O., Cattoni, R., Currey, A., Dinu, G., Duh, K., Elbayad, M., et al. Findings of the iwslt 2022 evaluation campaign. In *Proceedings of the 19th International Conference on Spoken Language Translation (IWSLT 2022)*, pp. 98–157, 2022.

Ao, J., Wang, R., Zhou, L., Wang, C., Ren, S., Wu, Y., Liu, S., Ko, T., Li, Q., Zhang, Y., Wei, Z., Qian, Y., Li, J., and Wei, F. Speecht5: Unified-modal encoder-decoder pre-training for spoken language processing. In *ACL (1)*, pp. 5723–5738. Association for Computational Linguistics, 2022a.

Ao, J., Zhang, Z., Zhou, L., Liu, S., Li, H., Ko, T., Dai, L., Li, J., Qian, Y., and Wei, F. Pre-training transformer decoder for end-to-end ASR model with unpaired speech data. In *INTERSPEECH*, pp. 2658–2662. ISCA, 2022b.

Ardila, R., Branson, M., Davis, K., Kohler, M., Meyer, J., Henretty, M., Morais, R., Saunders, L., Tyers, F., and Weber, G. Common voice: A massively-multilingual speech corpus. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pp. 4218–4222, 2020.

Bahar, P., Bieschke, T., and Ney, H. A comparative study on end-to-end speech to text translation. In *ASRU*, pp. 792–799. IEEE, 2019.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Bahdanau, D., Chorowski, J., Serdyuk, D., Brakel, P., and Bengio, Y. End-to-end attention-based large vocabulary speech recognition. In *ICASSP*, pp. 4945–4949. IEEE, 2016.

Bansal, S., Kamper, H., Livescu, K., Lopez, A., and Goldwater, S. Pre-training on high-resource speech recognition improves low-resource speech-to-text translation. In *NAACL-HLT (1)*, pp. 58–68. Association for Computational Linguistics, 2019.

Bapna, A., Chung, Y., Wu, N., Gulati, A., Jia, Y., Clark, J. H., Johnson, M., Riesa, J., Conneau, A., and Zhang, Y. SLAM: A unified encoder for speech and language modeling via speech-text joint pre-training. *CoRR*, abs/2110.10329, 2021.

Bapna, A., Cherry, C., Zhang, Y., Jia, Y., Johnson, M., Cheng, Y., Khanuja, S., Riesa, J., and Conneau, A. mslam: Massively multilingual joint pre-training for speech and text. *CoRR*, abs/2202.01374, 2022.

Bentivogli, L., Cettolo, M., Gaido, M., Karakanta, A., Martinelli, A., Negri, M., and Turchi, M. Cascade versus direct speech translation: Do the differences still make a difference? In *ACL/IJCNLP (1)*, pp. 2873–2887. Association for Computational Linguistics, 2021.

Berard, A., Pietquin, O., Servan, C., and Besacier, L. Listen and translate: A proof of concept for end-to-end speech-to-text translation. *CoRR*, abs/1612.01744, 2016.

Bérard, A., Besacier, L., Kocabiyikoglu, A. C., and Pietquin, O. End-to-end automatic speech translation of audiobooks. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6224–6228. IEEE, 2018.

Berndt, D. J. and Clifford, J. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pp. 359–370. Seattle, WA, USA:, 1994.

Chen, L., Zhang, Y., Zhang, R., Tao, C., Gan, Z., Zhang, H., Li, B., Shen, D., Chen, C., and Carin, L. Improving sequence-to-sequence learning via optimal transport. In *ICLR (Poster)*. OpenReview.net, 2019.

Chen, Z., Zhang, Y., Rosenberg, A., Ramabhadran, B., Moreno, P. J., Bapna, A., and Zen, H. MAESTRO: matched speech text representations through modality matching. In *INTERSPEECH*. ISCA, 2022.

Chuang, S., Sung, T., Liu, A. H., and Lee, H. Worse wer, but better bleu? leveraging word embedding as intermediate in multitask end-to-end speech translation. In Jurafsky, D., Chai, J., Schluter, N., and Tetreault, J. R. (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pp. 5998–6003. Association for Computational Linguistics, 2020. URL https://www.aclweb.org/anthology/2020.acl-main.533/.

Cuturi, M. Sinkhorn distances: Lightspeed computation of optimal transport. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/af21d0c97db2e27e13572cbf59eb343d-Paper.pdf.

Cuturi, M. and Blondel, M. Soft-dtw: a differentiable loss function for time-series. In *International conference on machine learning*, pp. 894–903. PMLR, 2017.

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

Dong, Q., Ye, R., Wang, M., Zhou, H., Xu, S., Xu, B., and Li, L. Listen, understand and translate: Triple supervision decouples end-to-end speech-to-text translation. In *AAAI*, pp. 12749–12759. AAAI Press, 2021.

Du, Y., Zhang, Z., Wang, W., Chen, B., Xie, J., and Xu, T. Regularizing end-to-end speech translation with triangular decomposition agreement. In *AAAI*, pp. 10590–10598. AAAI Press, 2022.

Duong, L., Anastasopoulos, A., Chiang, D., Bird, S., and Cohn, T. An attentional model for speech translation without transcription. In *HLT-NAACL*, pp. 949–959. The Association for Computational Linguistics, 2016.

Fang, Q., Ye, R., Li, L., Feng, Y., and Wang, M. STEMM: self-learning with speech-text manifold mixup for speech translation. In *ACL (1)*, pp. 7050–7062. Association for Computational Linguistics, 2022.

Feydy, J., Séjourné, T., Vialard, F.-X., Amari, S.-i., Trouve, A., and Peyré, G. Interpolating between optimal transport and mmd using sinkhorn divergences. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2681–2690, 2019.

Frogner, C., Zhang, C., Mobahi, H., Araya, M., and Poggio, T. A. Learning with a wasserstein loss. *Advances in neural information processing systems*, 28, 2015.

Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., and Lempitsky, V. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. C., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pp. 2672–2680, 2014.

Grave, E., Joulin, A., and Berthet, Q. Unsupervised alignment of embeddings with wasserstein procrustes. In *AISTATS*, volume 89 of *Proceedings of Machine Learning Research*, pp. 1880–1890. PMLR, 2019.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006.

Graves, A., Wayne, G., and Danihelka, I. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Han, C., Wang, M., Ji, H., and Li, L. Learning shared semantic space for speech-to-text translation. In *ACL/IJCNLP (Findings)*, volume ACL/IJCNLP 2021 of *Findings of ACL*, pp. 2214–2225. Association for Computational Linguistics, 2021.

Haubold, A. and Kender, J. R. Alignment of speech to highly imperfect text transcriptions. In *2007 IEEE International Conference on Multimedia and Expo*, pp. 224–227. IEEE, 2007.

Inaguma, H., Duh, K., Kawahara, T., and Watanabe, S. Multilingual end-to-end speech translation. In *IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2019, Singapore, December 14-18, 2019*, pp. 570–577. IEEE, 2019. doi: 10.1109/ASRU46091.2019.9003832. URL https://doi.org/10.1109/ASRU46091.2019.9003832.

Inaguma, H., Kiyono, S., Duh, K., Karita, S., Soplin, N. E. Y., Hayashi, T., and Watanabe, S. Espnet-st: All-in-one speech translation toolkit. *arXiv preprint arXiv:2004.10234*, 2020.

Inaguma, H., Kawahara, T., and Watanabe, S. Source and target bidirectional knowledge distillation for end-to-end speech translation. In *NAACL-HLT*, pp. 1872–1881. Association for Computational Linguistics, 2021.

Indurthi, S., Zaidi, M. A., Kumar Lakumarapu, N., Lee, B., Han, H., Ahn, S., Kim, S., Kim, C., and Hwang, I. Task aware multi-task learning for speech to text tasks. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 7723–7727, 2021. doi: 10.1109/ICASSP39728.2021.9414703.

Juang, B.-H. On the hidden markov model and dynamic time warping for speech recognition—a unified view. *AT&T Bell Laboratories Technical Journal*, 63(7):1213–1243, 1984.

Karita, S., Chen, N., Hayashi, T., Hori, T., Inaguma, H., Jiang, Z., Someki, M., Soplin, N. E. Y., Yamamoto, R., Wang, X., et al. A comparative study on transformer vs rnn in speech applications. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pp. 449–456. IEEE, 2019.

Kim, S., Hori, T., and Watanabe, S. Joint ctc-attention based end-to-end speech recognition using multi-task learning. In *ICASSP*, pp. 4835–4839. IEEE, 2017.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980.

Kudo, T. and Richardson, J. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In Blanco, E. and Lu, W. (eds.), *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018*, pp. 66–71. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-2012. URL https://doi.org/10.18653/v1/d18-2012.

Lample, G., Conneau, A., Denoyer, L., and Ranzato, M. Unsupervised machine translation using monolingual corpora only. In *ICLR (Poster)*. OpenReview.net, 2018.

Le, H., Pino, J., Wang, C., Gu, J., Schwab, D., and Besacier, L. Dual-decoder transformer for joint automatic speech recognition and multilingual speech translation. In Scott, D., Bel, N., and Zong, C. (eds.), *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pp. 3520–3533. International Committee on Computational Linguistics, 2020. doi: 10.18653/v1/2020.coling-main.314. URL https://doi.org/10.18653/v1/2020.coling-main.314.

Le, H., Pino, J. M., Wang, C., Gu, J., Schwab, D., and Besacier, L. Lightweight adapter tuning for multilingual speech translation. In *ACL/IJCNLP (2)*, pp. 817–824. Association for Computational Linguistics, 2021.

Lee, Y. and Kim, T. Learning pronunciation from a foreign language in speech synthesis networks. *CoRR*, abs/1811.09364, 2018.

Li, X., Wang, C., Tang, Y., Tran, C., Tang, Y., Pino, J., Baevski, A., Conneau, A., and Auli, M. Multilingual speech translation with efficient finetuning of pretrained models. *arXiv e-prints*, pp. arXiv–2010, 2020.

Liang, P. P., Zadeh, A., and Morency, L. Foundations and recent trends in multimodal machine learning: Principles, challenges, and open questions. *CoRR*, abs/2209.03430, 2022.

Lin, H., Tseng, H., Lu, X., and Tsao, Y. Unsupervised noise adaptive speech enhancement by discriminator-constrained optimal transport. In *NeurIPS*, pp. 19935–19946, 2021.

Liu, Y., Zhu, J., Zhang, J., and Zong, C. Bridging the modality gap for speech-to-text translation. *CoRR*, abs/2010.14920, 2020.

Muda, L., Begam, M., and Elamvazuthi, I. Voice recognition algorithms using mel frequency cepstral coefficient (mfcc) and dynamic time warping (dtw) techniques. *Journal of Computing*, 2(3):138–143, 2010.

Ney, H. Speech translation: coupling of recognition and translation. In *ICASSP*, pp. 517–520. IEEE Computer Society, 1999.

Ouyang, S., Ye, R., and Li, L. WACO: word-aligned contrastive learning for speech translation. *CoRR*, abs/2212.09359, 2022.

Papi, S., Gaido, M., Negri, M., and Turchi, M. Speechformer: Reducing information loss in direct speech translation. In *EMNLP (1)*, pp. 1698–1706. Association for Computational Linguistics, 2021.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the*

Association for Computational Linguistics, pp. 311–318, 2002.

Park, D. S., Chan, W., Zhang, Y., Chiu, C., Zoph, B., Cubuk, E. D., and Le, Q. V. Specaugment: A simple data augmentation method for automatic speech recognition. In Kubin, G. and Kacic, Z. (eds.), *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019*, pp. 2613–2617. ISCA, 2019. doi: 10.21437/Interspeech.2019-2680. URL https://doi.org/10.21437/Interspeech.2019-2680.

Peyré, G., Cuturi, M., et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Pino, J., Xu, Q., Ma, X., Dousti, M. J., and Tang, Y. Self-training for end-to-end speech translation. *arXiv preprint arXiv:2006.02490*, 2020.

Post, M. A call for clarity in reporting BLEU scores. In *WMT*, pp. 186–191. Association for Computational Linguistics, 2018.

Sakoe, H. and Chiba, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1): 43–49, 1978.

Sinkhorn, R. and Knopp, P. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967.

Sperber, M., Neubig, G., Niehues, J., and Waibel, A. Attention-passing models for robust and data-efficient end-to-end speech translation. *Transactions of the Association for Computational Linguistics*, 7:313–325, 2019.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

Tang, Y., Pino, J., Wang, C., Ma, X., and Genzel, D. A general multi-task learning framework to leverage text data for speech to text tasks. In *ICASSP*, 2021a.

Tang, Y., Pino, J. M., Li, X., Wang, C., and Genzel, D. Improving speech translation by understanding and learning from the auxiliary text translation task. In *ACL/IJCNLP (1)*, pp. 4252–4261. Association for Computational Linguistics, 2021b.

Tang, Y., Gong, H., Dong, N., Wang, C., Hsu, W., Gu, J., Baevski, A., Li, X., Mohamed, A., Auli, M., and Pino, J. Unified speech-text pre-training for speech translation and recognition. In *ACL (1)*, pp. 1488–1499. Association for Computational Linguistics, 2022.

Viterbi, A. J. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. *IEEE Trans. Inf. Theory*, 13(2):260–269, 1967.

Wang, C., Pino, J., Wu, A., and Gu, J. Covost: A diverse multilingual speech-to-text translation corpus. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pp. 4197–4203, Marseille, France, May 2020a. European Language Resources Association. URL https://www.aclweb.org/anthology/2020.lrec-1.517.

Wang, C., Tang, Y., Ma, X., Wu, A., Okhonko, D., and Pino, J. fairseq s2t: Fast speech-to-text modeling with fairseq. *arXiv preprint arXiv:2010.05171*, 2020b.

Wang, C., Wu, A., and Pino, J. Covost 2 and massively multilingual speech-to-text translation. *arXiv e-prints*, pp. arXiv–2007, 2020c.

Wang, C., Wu, Y., Liu, S., Yang, Z., and Zhou, M. Bridging the gap between pre-training and fine-tuning for end-to-end speech translation. In *AAAI*, pp. 9161–9168. AAAI Press, 2020d.

Wang, C., Wu, Y., Liu, S., Zhou, M., and Yang, Z. Curriculum pre-training for end-to-end speech translation. In *ACL*, pp. 3728–3738. Association for Computational Linguistics, 2020e.

Watanabe, S., Hori, T., Kim, S., Hershey, J. R., and Hayashi, T. Hybrid ctc/attention architecture for end-to-end speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 11(8):1240–1253, 2017.

Weiss, R. J., Chorowski, J., Jaitly, N., Wu, Y., and Chen, Z. Sequence-to-sequence models can directly translate foreign speech. In Lacerda, F. (ed.), *Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017*, pp. 2625–2629. ISCA, 2017. URL http://www.isca-speech.org/archive/Interspeech_2017/abstracts/0503.html.

Xu, C., Hu, B., Li, Y., Zhang, Y., Huang, S., Ju, Q., Xiao, T., and Zhu, J. Stacked acoustic-and-textual encoding: Integrating the pre-trained models into speech translation encoders. In *ACL/IJCNLP (1)*, pp. 2619–2630. Association for Computational Linguistics, 2021.

Ye, R., Wang, M., and Li, L. End-to-end speech translation via cross-modal progressive training. In *Interspeech*, pp. 2267–2271. ISCA, 2021.

Ye, R., Wang, M., and Li, L. Cross-modal contrastive learning for speech translation. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 2022.

Yuan, J., Liberman, M., et al. Speaker identification on the scotus corpus. *Journal of the Acoustical Society of America*, 123(5):3878, 2008.

Zhang, B., Titov, I., Haddow, B., and Sennrich, R. Adaptive feature selection for end-to-end speech translation. In *EMNLP (Findings)*, volume EMNLP 2020 of *Findings of ACL*, pp. 2533–2544. Association for Computational Linguistics, 2020.

Zhao, C., Wang, M., Dong, Q., Ye, R., and Li, L. Neurst: Neural speech translation toolkit. In *ACL (demo)*, pp. 55–62. Association for Computational Linguistics, 2021a.

Zhao, J., Luo, W., Chen, B., and Gilman, A. Mutual-learning improves end-to-end speech translation. In *EMNLP (1)*, pp. 3989–3994. Association for Computational Linguistics, 2021b.

Zheng, R., Chen, J., Ma, M., and Huang, L. Fused acoustic and text encoding for multimodal bilingual pretraining and speech translation. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12736–12746. PMLR, 2021.

A. Distance Functions for Speech-Text Alignment

We provide a more detailed description of alternative loss functions (other than OT) for learning to align speech and text features, as discussed in Section 5.2. Recall that our notations for the output speech and text features are $\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_m) \in \mathbb{R}^{d\times m}$ and $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_n) \in \mathbb{R}^{d\times n}$, respectively, where $\mathbf{u}_i \in \mathbb{R}^d$ and $\mathbf{v}_j \in \mathbb{R}^d$ for all i, j. As some distance functions require the sequence lengths to be the same (which is not the case, recall that typically $m \gg n$), we first present methods for making this happen.

A.1. Length-matching operations

In general, the sequences $\mathbf{U}$ and $\mathbf{V}$ are processed to obtain new sequences $\tilde{\mathbf{U}} \in \mathbb{R}^{d\times p}$ and $\tilde{\mathbf{V}} \in \mathbb{R}^{d\times p}$ that have the same length p (which needs not equal m or n):

$$\tilde{\mathbf{U}}, \tilde{\mathbf{V}} = \textsc{LengthMatching}(\mathbf{U}, \mathbf{V}). \tag{7}$$

We have experimented with the following three length-matching operations.

Average pooling This consists in simply taking the global average of each sequence (Liu et al., 2020; Dong et al., 2021):

$$\tilde{\mathbf{U}} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{u}_i, \qquad \tilde{\mathbf{V}} = \frac{1}{n}\sum_{j=1}^{n} \mathbf{v}_j. \tag{8}$$

Note that before using this averaging operation, one may also apply some kind of *shrinking* mechanisms to reduce the length of the speech representations such as using CTC labels as guidance to remove contiguous blanks or repeated predictions (Liu et al., 2020), or using a shrinking layer, for example a 2D convolutional layer followed by a normalization one (Dong et al., 2021). These shrinking operations do not necessarily make the length of the speech output equal to that of the text one. For simplicity, we did not experiment with any type of shrinking mechanisms.

Interpolation This consists in resizing the longer sequence to the same length as the shorter one:

$$\tilde{\mathbf{U}} = \textsc{Interpolate}(\mathbf{U}, n), \qquad \tilde{\mathbf{V}} = \mathbf{V}, \tag{9}$$

The function $\textsc{Interpolate}(\cdot,\cdot)$ takes as inputs the sequence $\mathbf{U}$ (of length m) together with the desired dimension n and produces the interpolated sequence $\tilde{\mathbf{U}}$ of length n. The most popular method of interpolation is perhaps *linear* interpolation, which we use in our experiments, but more advanced ones such as *cubic* interpolation can also be used. In our implementation, we used PyTorch's `nn.functional.interpolate` function (with argument `mode='linear'`). Since interpolation is a rather standard operation, we omit further details.

Attention-based transformation This was proposed by Tang et al. (2021b) under the name Cross-Attentive Regularization (CAR), which consists in using cosine attention (Graves et al., 2014) to obtain:

$$\tilde{\mathbf{U}} = \mathbf{U}\,\mathrm{softmax}(\bar{\mathbf{U}}^\top \bar{\mathbf{V}}) \in \mathbb{R}^{d\times n}, \tag{10}$$

$$\tilde{\mathbf{V}} = \mathbf{V}\,\mathrm{softmax}(\bar{\mathbf{V}}^\top \bar{\mathbf{V}}) \in \mathbb{R}^{d\times n}, \tag{11}$$

where $\bar{\mathbf{U}} = (\bar{\mathbf{u}}_1, \dots, \bar{\mathbf{u}}_m)$ and $\bar{\mathbf{V}} = (\bar{\mathbf{v}}_1, \dots, \bar{\mathbf{v}}_n)$ are sequences of normalized features:

$$\bar{\mathbf{u}}_i = \frac{\mathbf{u}_i}{\|\mathbf{u}_i\|_2}, \quad i = 1, \dots, m, \tag{12}$$

$$\bar{\mathbf{v}}_j = \frac{\mathbf{v}_j}{\|\mathbf{v}_j\|_2}, \quad j = 1, \dots, n. \tag{13}$$

A.2. Distance functions

We present four different distance functions: Euclidean distance, Kullback-Leibler (KL) divergence, adversarial loss, and dynamic time warping (DTW). For the first two, we apply one of the length-matching operations presented in the previous section.

Euclidean distance This is computed as

$$d_{\mathrm{euc}}(\tilde{\mathbf{U}}, \tilde{\mathbf{V}}) = \|\tilde{\mathbf{U}} - \tilde{\mathbf{V}}\|_2 = \sqrt{\sum_{i=1}^{p} \|\tilde{\mathbf{u}}_i - \tilde{\mathbf{v}}_i\|_2^2}, \tag{14}$$

where p denotes the length of the sequences resulting from the length-matching mechanism. In particular, $p = 1$ for *average pooling* and $p = n$ for *interpolation* or *attention*.

Kullback-Leibler divergence Let $\mathbf{a} = (a_1, \ldots, a_d)$ and $\mathbf{b} = (b_1, \ldots, b_d)$ (where $a_i, b_i \in [0, 1]$ and $\mathbf{1}^\top \mathbf{a} = \mathbf{1}^\top \mathbf{b} = 1$) be the masses representing two discrete probability distributions. The KL divergence between these distributions is given by

$$\mathrm{KL}(\mathbf{a} \parallel \mathbf{b}) = \sum_{k=1}^{d} a_k \log \frac{a_k}{b_k}. \tag{15}$$

For two input sequences $\tilde{\mathbf{U}} \in \mathbb{R}^{d \times p}$ and $\tilde{\mathbf{V}} \in \mathbb{R}^{d \times p}$ (already length-matched), we first normalize the features to obtain probability vectors:

$$\bar{\mathbf{u}}_i = \mathrm{softmax}(\tilde{\mathbf{u}}_i), \quad i = 1, \ldots, p, \tag{16}$$

$$\bar{\mathbf{v}}_i = \mathrm{softmax}(\tilde{\mathbf{v}}_i), \quad i = 1, \ldots, p. \tag{17}$$

Then our KL-based distance between the input sequences is given by:

$$d_{\mathrm{KL}}(\tilde{\mathbf{U}}, \tilde{\mathbf{V}}) = \sum_{i=1}^{p} \mathrm{KL}(\bar{\mathbf{u}}_i \parallel \bar{\mathbf{v}}_i). \tag{18}$$

Adversarial loss The idea is to add a *discriminator* module that learns to distinguish speech features from text features (Ganin et al., 2016; Lample et al., 2018; Alinejad & Sarkar, 2020) (highly inspired by adversarial generative networks (Goodfellow et al., 2014)). In our case, the speech and text encoders act as a *generator* that tries to deceive the discriminator. Intuitively, if the discriminator fails to distinguish between the speech and text features, then we can say that the encoders are successful at producing speech and text features that are close to each other. To achieve this, we add two loss functions (in addition to existing ones such as CTC):

1. A *discriminator loss* that helps train the discriminator to distinguish between the speech and text features.
2. A *generator loss* that helps train the generator (i.e., the two encoders) to fool the discriminator.

Then, training alternates between updating the discriminator's parameters (while freezing the generator) and updating the generator's parameters (while freezing the discriminator). More formally, given an input feature vector $\mathbf{u} \in \mathbb{R}^d$, the discriminator outputs a value $p \in [0, 1]$ representing the probability of $\mathbf{u}$ being produced by a *speech* encoder (or equivalently, $1 - p$ denotes the probability of $\mathbf{u}$ being produced by the *text* encoder):

$$p = \textsc{Discriminator}(\mathbf{u}). \tag{19}$$

For a pair of sequences $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_m) \in \mathbb{R}^{d \times m}$ and $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_n) \in \mathbb{R}^{d \times n}$ produced by the speech and text encoders, respectively, we compute the discriminator loss as

$$L_{\mathrm{disc}}(\mathbf{U}, \mathbf{V}) = \sum_{i=1}^{m} \textsc{bce}(p_i, 1) + \sum_{j=1}^{n} \textsc{bce}(q_j, 0), \tag{20}$$

where

$$p_i = \textsc{Discriminator}(\mathbf{u}_i), \tag{21}$$

$$q_j = \textsc{Discriminator}(\mathbf{v}_j), \tag{22}$$

and $\textsc{bce}(\cdot, \cdot)$ is the binary cross-entropy loss:

$$\textsc{bce}(p, y) = -y \log p - (1 - y) \log(1 - p), \tag{23}$$

where $y \in \{0, 1\}$ is the ground-truth label.

The generator loss is similar, except that we switch the labels:

$$L_{\mathrm{gen}}(\mathbf{U}, \mathbf{V}) = \sum_{i=1}^{m} \textsc{bce}(p_i, 0) + \sum_{j=1}^{n} \textsc{bce}(q_j, 1). \tag{24}$$

We use the same set of hyper-parameters as Lample et al. (2018) (which was also used by Alinejad & Sarkar (2020)). In particular, our discriminator is a 3-layer feedforward network with hidden dimension 1024 and leaky ReLU activations (with negative slope 0.2), each followed by a Dropout layer of probability 0.1.

Differentiable DTW Dynamic time warping (Sakoe & Chiba, 1978) is a popular approach to measuring the similarity between two time series, making it a natural choice to the problem of aligning speech and text, both of which are temporal sequences. A differentiable variant of this function has been proposed (Cuturi & Blondel, 2017) to allow training with stochastic gradient descent. Even though this is also available in our implementation, we did not include it in our experiments due to its prohibitively high memory usage. Therefore, we refer the reader to Cuturi & Blondel (2017) for further details.

B. Further Implementation Details

Data processing The speech input features are 80-dimensional log Mel filter-bank. Utterances having more than 3000 frames are removed for GPU efficiency. For data augmentation, following (Wang et al., 2020b;c) we used SpecAugment (Park et al., 2019) with LibriSpeech basic (LB) policy (no time warping) in the bilingual MuST-C and multilingual settings of CoVoST-2. For multilingual MuST-C, we applied LibriSpeech double (LD) policy. The inputs to the text encoder in our ASR and MT pre-training phases are in their phoneme pronunciation form, except for

the many-to-one case in which we used the 10k unigram vocabulary (Kudo & Richardson, 2018) learned on the training text of all source languages. Following (Tang et al., 2021b), the grapheme-to-phoneme conversion is done using the g2p_en package (Lee & Kim, 2018). The resulting English phoneme vocabulary has the size of 134 tokens. On the target side for MT and ST models, we used 10K unigram vocabulary learned on all relevant target training text. More specifically, the vocabulary is learned on German or French training text in the bilingual case, and on the concatenation of all target training texts in the multilingual ones.

Training hyper-parameters We used the Adam optimizer (Kingma & Ba, 2015) with learning rate linearly increased for the first N warmup steps to a value $\eta_{\max}$, then decreased proportionally to the inverse square root of the step counter. $\eta_{\max}$ is set to 2×10^{-3} in medium ASR/ST experiments and to 5×10^{-4} in experiments using large architecture. For MT experiments, $\eta_{\max}=5\times10^{-3}$. N is set to 10000 in ASR/ST experiments and to 4000 in MT experiments. Label smoothing is set to 0.1 (Szegedy et al., 2016) for models using cross-entropy loss. All ST models on MuST-C are trained up to 50 epochs and those on CoVoST-2 are trained up to 50 and 100 epochs for En→X and X→En, respectively. The last 10 checkpoints are averaged for decoding using a beam size of 5.

Metrics We report the case-sensitive detokenized BLEU scores (Papineni et al., 2002; Post, 2018), except for English-Japanese and English-Chinese where we report character-level BLEU (Wang et al., 2020c). For ASR performence, we report Word Error Rate (WER).

Hardware and training time Medium ASR/ST models were trained on 8 NVIDIA V100 GPUs while large ASR/ST ones were trained on 32 A100 GPUs. All MT models were trained on 8 V100 GPUs. A comparison of the training time is given in Table 8.

Table 8. Training time for 100 epochs on MuST-C En-Fr (batch size 40K tokens).

Method	Hours	Slowdown vs. CTC
CE	7.10	1.13×
CTC	6.27	1.00×
CTC+CE	7.78	1.24×
CTC+OT	7.62	1.22×

C. Further Ablation Studies

C.1. CTC encoder vs. CE encoder

In Section 4.2 we claim that the speech encoder trained by CTC is stronger than the one trained by CE. The experimental results in Section 6 already support this claim. Indeed, the ST results labeled with "CTC" are consistently better than those with "CE". Given that the two models are identical except for the encoder initialization, any difference in the ST performance between them can technically be attributed to the quality of the initialized encoders. This allows us to conclude that the CTC encoder is superior to the CE encoder (in terms of producing better embeddings for the subsequent ST task).

In this section, we present additional results on the ASR task to further demonstrate the superiority of the CTC encoder.

- Consider two pre-trained speech encoders, one pre-trained with CE and the other with CTC, as shown in Figures 1b and 2 (without the CE branch), respectively. Previously we have referred to these as "CE encoder" and "CTC encoder", respectively.
- We freeze these encoders, plug them into an encoder-decoder ASR model, and train it with the CE loss.
- We do the same as the previous item, but this time with an encoder-only model, trained with the CTC loss. Here we add a linear layer between the encoder and the CTC loss to project the encoder's output hidden dimension to the vocabulary size.

The results are shown in Table 9.

Table 9. WER on MuST-C En-De *dev* set for comparing CTC encoder and CE encoder. "Enc" and "Dec" mean "encoder" and "decoder", respectively.

Frozen Enc	Enc-Dec-CE	Enc-Linear-CTC
CE	12.7	29.9
CTC	12.4	15.1

We observe that, for the encoder-decoder model, the pre-trained CTC encoder outperforms the pre-trained CE encoder (WER 12.4 vs. 12.7) despite the fact that the CE encoder has been previously pre-trained in a similar fashion (using an encoder-decoder model). This provides strong empirical evidence to support our hypothesis that the CTC encoder is superior.

For the encoder-only model, the results are similar, but with a much larger gap (WER 15.1 for CTC vs. 29.9 for CE). We note that this model has a smaller number of trainable parameters compared to the previous one (5M parameters of the linear layer compared to 35M parameters of the decoder).

C.2. Different loss weighting values in CTC+CE

The hybrid architecture using CE and CTC has been shown to obtain strong ASR performance (Watanabe et al., 2017; Karita et al., 2019). Hence, there are several works using

Table 10. BLEU on MuST-C dev sets for different loss weighting values in CTC+CE.

	Method	Bilingual		Multilingual (one-to-many)								
		de	fr	de	es	fr	it	nl	pt	ro	ru	avg
w/o MT	0.7CE + 0.3CTC	**24.10**	29.59	24.13	32.71	29.77	24.54	26.13	**30.24**	23.27	15.86	25.83
	CTC + 0.1CE	24.09	**29.74**	**25.32**	**33.80**	**30.54**	**25.64**	**26.97**	30.07	**23.44**	**15.95**	**26.47**
with MT	0.7CE + 0.3CTC	24.04	29.77	**25.34**	33.57	30.42	**25.61**	**26.98**	30.31	23.14	15.90	26.41
	CTC + 0.1CE	**24.28**	**30.21**	25.01	**34.01**	**30.55**	25.49	**26.94**	**30.56**	**23.76**	**16.23**	**26.57**

Table 11. BLEU on CoVoST-2 dev sets in En→X settings for different choices of tokenization: character ("char") vs. Sentencepiece 10K unigram ("spm10k").

Method	Vocab	ar	ca	cy	de	et	fa	id	ja	lv	mn	sl	sv	ta	tr	zh	avg
CE w/o MT	char	14.95	27.01	26.94	20.83	**14.06**	**15.77**	25.07	36.04	13.92	12.31	**16.70**	22.63	14.17	**13.90**	30.00	20.29
	spm10k	**15.04**	**27.23**	**27.19**	**20.98**	**14.06**	15.76	**25.50**	**36.54**	**14.06**	**12.57**	**16.70**	**22.95**	**14.46**	13.84	**30.58**	**20.50**
CE w/ MT	char	15.86	27.99	27.92	21.71	14.53	16.25	26.05	36.89	**14.89**	**12.96**	**17.77**	23.51	**14.86**	**14.61**	30.49	21.09
	spm10k	**15.94**	**28.13**	**28.09**	**21.80**	**14.76**	**16.34**	**26.14**	**37.00**	14.68	12.94	17.70	**23.67**	14.72	**14.61**	**31.03**	**21.17**

this pre-trained ASR model to initialize subsequent ST models with the combination of 0.7CE + 0.3CTC as proposed in (Watanabe et al., 2017; Karita et al., 2019). We additionally compare this combination with our choice (CTC + 0.1CE) in order to find the better performing recipe for the CTC+CE pre-training method. Results in Table 10 show that our combination of CTC+0.1CE works better for the ST initialization in both *with* (improvement of $+0.16$) and *without* (improvement of $+0.64$) MT pre-training settings. The results again highlight the importance of CTC in the ASR pre-training stage.

C.3. Character vs. SentencePiece tokenization

For CoVoST one-to-many settings, we compare different choices of vocabulary: character vs. 10K unigram vocabulary. The results (see Table 11) show that using the 10K unigram slightly outperforms using character dictionary.

D. Detailed Results and Discussion

D.1. Results without MT pre-training

Results are shown in the upper halves of Tables 12 and 14 for CoVoST-2 and MuST-C, respectively.

Bilingual The results (2nd and 3rd columns of Table 14, upper half) highlight the significant contribution of CTC in the final ST performance: all three methods using CTC in pre-training surpass the common method using CE (0.89–1.40 for En-De, and 1.35–1.39 for En-Fr). Specifically, pre-training with CTC alone outperforms pre-training with CE by a large margin ($+0.89$ and $+1.39$, respectively, for En-De and En-Fr). Adding CE or OT on top of CTC is helpful for En-De, though it slightly hurts the performance on En-Fr.

Multilingual The results (Tables 12 and 14, upper halves) show that CTC consistently outperforms CE on both datasets: the scores are higher across all language pairs by a large margin (on average, $+1.1$ and $+1.2$ on MuST-C and CoVoST-2, respectively). Adding CE on top of CTC slightly degrades the results (26.47 vs. 26.51) while using OT marginally improves the performance (26.60 vs. 26.51).

D.2. Results with MT pre-training

Results are shown in the lower halves of Tables 12 and 14 for CoVoST-2 and MuST-C, respectively.

Bilingual The results (2nd and 3rd columns of Table 14, lower half) show that CTC outperforms CE by around $+1.0$ BLEU point for both En-De and En-Fr. Compare with the without-MT-pre-training results (same columns but in the upper halves), we observe that MT pre-training are helpful for all methods. Especially, the gains for our proposed method are the highest (0.83/0.48 on En-De/En-Fr, compared to 0.35/0.41, 0.50/0.04, and 0.19/0.47 for CE, CTC, and CTC+CE, respectively).

Multilingual The results (Tables 12 and 14, lower halves) again show that CTC remains superior to CE. On the other hand, MT pre-training neither helps nor hurts the performance for MuST-C, while the improvements are more noticeable on CoVoST-2 (0.4–0.7 on average).

Table 12. Detailed BLEU on CoVoST-2 dev sets in En→X settings for all directions.

	Method	ar	ca	cy	de	et	fa	id	ja	lv	mn	sl	sv	ta	tr	zh	avg
w/o MT	CE	15.04	27.23	27.19	20.98	14.06	15.76	25.50	36.54	14.06	12.57	16.70	22.95	14.46	13.84	30.58	20.50
	CTC	16.46	28.90	28.68	22.40	15.05	16.66	26.83	37.74	15.04	13.46	18.09	24.30	15.39	15.08	31.57	21.71
	CTC+CE	16.50	28.84	28.84	22.38	15.16	16.90	26.89	38.11	15.38	13.60	18.36	24.31	15.38	15.16	31.59	21.83
	CTC+OT	**17.19**	**29.52**	**29.77**	**23.12**	**15.68**	**17.29**	**27.61**	**38.21**	**15.44**	**14.07**	**18.68**	**24.94**	**16.05**	**15.68**	**32.11**	**22.36**
with MT	CE	15.94	28.13	28.09	21.80	14.76	16.34	26.14	37.00	14.68	12.94	17.70	23.67	14.72	14.61	31.03	21.17
	CTC	16.92	29.20	29.07	22.76	15.45	17.00	27.29	38.03	15.56	13.73	18.47	24.57	15.77	15.24	32.02	22.10
	CTC+CE	16.99	29.31	29.34	22.73	15.64	17.14	27.37	38.12	15.86	13.72	18.82	24.68	16.05	15.55	32.03	22.22
	CTC+OT	**17.61**	**30.21**	**30.19**	**23.58**	**16.20**	**17.60**	**28.09**	**38.63**	**16.21**	**14.38**	**19.30**	**25.33**	**16.35**	**16.20**	**32.36**	**22.82**

Table 13. Detailed BLEU on CoVoST-2 dev sets in X→En settings for all directions.

	Method	ar	ca	cy	de	es	et	fa	fr	id	it	ja	lv	mn	nl	pt	ru	sl	sv	ta	tr	zh	avg
	Data (h)	2	136	2	184	113	3	49	264	1	44	1	2	3	7	10	18	2	2	2	4	10	
w/o MT	CE	1.32	22.06	3.01	19.32	25.53	1.06	3.94	27.31	**0.68**	20.18	0.35	0.78	**0.24**	3.95	8.65	14.44	0.75	1.02	**0.25**	3.04	7.11	7.86
	CTC	2.97	22.43	3.26	20.13	25.48	0.94	4.05	27.40	0.30	20.21	0.39	1.50	**0.19**	3.74	10.18	16.03	0.34	1.51	**0.23**	3.46	7.06	8.18
	CTC+CE	2.80	22.06	2.99	19.67	25.35	0.85	3.94	27.27	0.38	19.66	0.46	**1.65**	**0.27**	3.81	10.10	17.71	**0.96**	**1.87**	**0.15**	3.27	6.92	8.20
	CTC+OT	**3.50**	**23.10**	**4.15**	**20.68**	**26.22**	**1.33**	**4.56**	**28.09**	0.27	**21.06**	**0.89**	1.31	**0.26**	**4.33**	**10.70**	**17.98**	0.65	1.69	**0.20**	**3.63**	7.18	**8.66**
with MT	CE	1.72	22.42	2.85	20.01	25.78	0.75	3.66	27.84	0.22	20.17	0.22	0.86	0.14	3.70	9.09	15.20	0.32	1.63	0.19	3.00	6.90	7.94
	CTC	2.20	22.79	2.90	20.67	25.76	**0.96**	3.88	27.79	**0.44**	20.59	0.70	1.16	**0.19**	**4.22**	10.37	15.74	**0.97**	**1.68**	**0.24**	3.05	6.92	8.25
	CTC+CE	1.81	22.71	2.87	20.25	25.48	0.79	3.79	27.49	0.39	20.20	**0.81**	**1.66**	**0.19**	3.59	10.63	**17.16**	**1.12**	1.46	**0.22**	3.28	6.43	8.21
	CTC+OT	**3.26**	**23.73**	**3.26**	**21.14**	**26.52**	**0.93**	**4.45**	**28.87**	**0.52**	**21.32**	0.31	1.25	**0.25**	**4.20**	**11.39**	**17.23**	**0.97**	**1.77**	0.13	**3.53**	7.19	**8.68**

Table 14. Detailed BLEU on MuST-C dev sets.

	Method	Bilingual		Multilingual (one-to-many)								
		de	fr	de	es	fr	it	nl	pt	ro	ru	avg
w/o MT	CE	22.39	28.48	23.87	32.38	29.54	24.57	25.79	29.21	22.90	15.10	25.42
	CTC	23.25	**29.87**	25.16	33.66	30.36	25.74	26.94	30.40	23.63	16.19	26.51
	CTC+CE	23.77	29.74	**25.32**	33.80	30.54	25.64	26.97	30.07	23.44	**15.95**	26.47
	CTC+OT	**23.91**	29.83	25.25	**33.82**	**30.63**	**25.92**	**27.01**	**30.61**	**23.61**	**15.95**	**26.60**
with MT	CE	23.04	28.89	24.29	32.83	29.86	25.09	26.06	29.49	22.32	15.05	25.62
	CTC	24.08	29.91	25.41	33.60	30.54	25.63	26.79	30.45	**23.80**	15.92	26.52
	CTC+CE	24.28	30.21	25.01	**34.01**	**30.55**	25.49	26.94	**30.56**	23.76	**16.23**	26.57
	CTC+OT	**24.74**	**30.31**	**25.53**	33.89	30.51	**26.17**	**27.27**	30.49	23.55	15.97	**26.67**

Table 15. BLEU on CoVoST-2 test sets for En→X. †results from baselines in (Wang et al., 2020c). "CE" is the same model as the corresponding ones in CoVoST-2 but with MT pre-training.

	Method	ar	ca	cy	de	et	fa	id	ja	lv	mn	sl	sv	ta	tr	zh	avg
Medium	CoVoST-2 †	11.2	21.6	22.9	15.9	12.8	13.8	19.7	31.5	12.4	9.2	15.2	21.5	10.6	9.8	29.3	17.2
	CE	12.2	23.1	24.5	17.3	13.9	15.1	21.7	32.0	13.5	10.5	16.5	22.7	11.9	11.1	29.3	18.4
	CTC	12.9	24.1	25.4	18.1	14.7	15.7	22.7	32.8	14.3	11.1	17.7	23.6	12.6	11.7	30.5	19.2
	CTC+CE	13.1	24.3	25.6	18.2	14.8	15.9	22.8	32.8	14.4	11.1	17.7	24.0	12.9	11.8	30.3	19.3
	CTC+OT	**13.5**	**25.1**	**26.2**	**18.9**	**15.5**	**16.3**	**23.2**	**33.4**	**14.9**	**11.6**	**18.3**	**24.5**	**13.3**	**12.1**	**31.0**	**19.9**
Large	CoVoST-2 †	13.9	23.6	25.1	18.4	15.1	15.5	22.0	33.0	15.2	11.0	18.3	24.1	12.8	11.7	31.3	19.4
	CE	13.5	23.8	25.2	18.3	14.9	15.7	22.6	32.8	14.6	11.1	17.8	23.8	12.6	11.6	30.1	19.2
	CTC	13.8	24.5	25.9	18.8	15.4	16.1	23.0	33.6	15.2	11.5	18.6	24.4	13.2	11.9	31.1	19.8
	CTC+OT	**15.3**	**26.5**	**28.1**	**20.6**	**17.1**	**17.5**	**25.1**	**35.7**	**16.6**	**12.8**	**20.6**	**26.5**	**14.8**	**13.5**	**32.4**	**21.5**

Table 16. BLEU on CoVoST-2 test sets for X→En. † results from baselines in (Wang et al., 2020c). "CE" is the same model as the corresponding ones in CoVoST-2 but with MT pre-training.

	Method	fr	de	es	ca	avg
Medium	CoVoST-2 †	27.0	18.9	28.0	23.9	24.5
	CE	27.9	19.3	28.3	22.9	24.6
	CTC	27.7	19.7	28.2	23.2	24.7
	CTC+CE	27.4	19.4	28.2	22.9	24.5
	CTC+OT	**28.4**	**20.4**	**29.2**	**24.1**	**25.5**

Table 17. BLEU on MuST-C test sets (`tst-COMMON`), comparing with state-of-the-art methods. "Multi" means multilingual models. Note that "CE pre-training" (8th row from the bottom) is the same as FAIRSEQ S2T (Wang et al., 2020b) (1st row) but with MT pre-training, which turns out to be unhelpful.

Method		Multi	External Data		BLEU								
			Unlabeled	*Labeled*	de	es	fr	it	nl	pt	ro	ru	avg
FAIRSEQ S2T (Wang et al., 2020b)		✓	-	-	24.5	28.2	34.9	24.6	28.6	31.1	23.8	16.0	26.5
ESPnet-ST (Inaguma et al., 2020)		✓	-	-	22.9	28.0	32.7	23.8	27.4	28.0	21.9	15.8	25.1
NeurST (Zhao et al., 2021a)		✓	-	-	22.8	27.4	33.3	22.9	27.2	28.7	22.2	15.1	24.9
Dual-decoder (Le et al., 2020)		✓	-	-	23.6	28.1	33.5	24.2	27.6	30.0	22.9	15.2	25.6
Adapters (Le et al., 2021)		✓	-	-	24.7	28.7	35.0	25.0	28.8	31.1	23.8	16.4	26.6
AFS (Zhang et al., 2020)		-	-	-	22.4	26.9	31.6	23.0	24.9	26.3	21.0	14.7	23.9
Speechformer (Papi et al., 2021)		-	-	-	23.6	28.5	-	-	27.7	-	-	-	-
Mutual-learning (Zhao et al., 2021b)		-	-	-	-	28.7	36.3	-	-	-	-	-	-
BiKD (Inaguma et al., 2021)		-	-	-	25.3	-	35.3	-	-	-	-	-	-
TDA (Du et al., 2022)		-	-	-	25.4	29.6	36.1	25.1	29.6	31.1	23.9	16.4	27.2
Adversarial (Alinejad & Sarkar, 2020)		-	-	✓	20.2	-	17.0	-	-	-	-	-	-
MTL (Tang et al., 2021a)		-	-	✓	23.9	28.6	33.1	-	-	-	-	-	-
Joint Speech Text (Tang et al., 2021b)		-	-	✓	26.8	31.0	37.4	-	-	-	-	-	-
SATE (Xu et al., 2021)		-	-	✓	28.1	-	-	-	-	-	-	-	-
TaskAware (Indurthi et al., 2021)		-	-	✓	**28.9**	-	-	-	-	-	-	-	-
Self-training (Pino et al., 2020)		-	✓	✓	25.2	-	34.5	-	-	-	-	-	-
FAT-ST (Big) (Zheng et al., 2021)		-	✓	✓	25.5	30.8	-	-	30.1	-	-	-	-
XSTNet (Ye et al., 2021)		-	✓	✓	27.1	30.8	38.0	26.4	31.2	32.4	25.7	18.5	28.8
Chimera (Han et al., 2021)		-	✓	✓	27.1	30.6	35.6	25.0	29.2	30.2	24.0	17.4	27.4
STEMM (Fang et al., 2022)		-	✓	✓	28.7	31.0	37.4	25.8	30.5	31.7	24.5	17.8	28.4
ConST (Ye et al., 2022)		-	✓	✓	28.3	32.0	38.3	27.2	**31.7**	33.1	25.6	**18.9**	29.4
STPT (Tang et al., 2022)		-	✓	✓	-	**33.1**	**39.7**	-	-	-	-	-	-
CE pre-training	MEDIUM	✓	-	-	24.6	28.7	34.9	24.6	28.4	30.7	23.7	15.9	26.4
CTC pre-training	MEDIUM	✓	-	-	25.9	29.7	36.6	25.6	29.6	32.0	24.6	16.7	27.6
CTC+CE pre-training	MEDIUM	✓	-	-	25.6	29.5	36.4	25.2	29.5	31.6	24.5	16.5	27.4
Siamese-PT (this work)	MEDIUM	✓	-	-	26.2	29.8	36.9	25.9	29.8	32.1	24.8	16.8	27.8
CE pre-training	LARGE	✓	-	-	26.9	30.8	37.7	26.7	30.8	33.3	26.2	17.9	28.8
CTC pre-training	LARGE	✓	-	-	27.6	31.4	38.2	27.2	31.1	33.6	26.4	18.4	29.2
CTC+CE pre-training	LARGE	✓	-	-	27.2	31.2	38.0	27.0	31.5	33.7	26.2	18.3	29.1
Siamese-PT (this work)	LARGE	✓	-	-	27.9	31.8	39.2	**27.7**	**31.7**	**34.2**	**27.0**	18.5	**29.8**