SEMI-SUPERVISED END-TO-END SPEECH RECOGNITION USING TEXT-TO-SPEECH AND AUTOENCODERS

Shigeki Karita[1], *Shinji Watanabe*[2], *Tomoharu Iwata*[1], *Marc Delcroix*[1], *Atsunori Ogawa*[1], *Tomohiro Nakatani*[1]

[1]NTT Communication Science Laboratories
[2]Center for Language and Speech Processing Johns Hopkins University

ABSTRACT

We introduce speech and text autoencoders that share encoders and decoders with an automatic speech recognition (ASR) model to improve ASR performance with large speech only and text only training datasets. To build the speech and text autoencoders, we leverage state-of-the-art ASR and text-to-speech (TTS) encoder decoder architectures. These autoencoders learn features from speech only and text only datasets by switching the encoders and decoders used in the ASR and TTS models. Simultaneously, they aim to encode features to be compatible with ASR and TTS models by a multi-task loss. Additionally, we anticipate that TTS joint training can also improve the ASR performance because both ASR and TTS models learn transformations between speech and text. The experimental result we obtained with our semi-supervised end-to-end ASR/TTS training revealed reductions from a model initially trained with a small paired subset of the LibriSpeech corpus in the character error rate from 10.4% to 8.4% and word error rate from 20.6% to 18.0% by retraining the model with a large unpaired subset of the corpus.

Index Terms— speech recognition, semi-supervised learning, autoencoder, encoder-decoder

1. INTRODUCTION

End-to-end automatic speech recognition (ASR) [1], [2] and text-to-speech (TTS) [3], [4] models have a lot in common. Their architectures are both encoder-decoder [5] with attention mechanism [6]. Both tasks learn transformations between speech and text features directly. For example, the ASR model does "speech-to-feature-to-text" and the TTS model does "text-to-feature-to-speech", where the speech is often a log Mel filterbank sequence, the feature is an output of the encoder networks, and the text is a sequence of character ids. They do not need the hand-crafted lexicons (e.g., a pronunciation dictionary) or complex decoding procedures used in the non end-to-end ASR [7] and TTS [8] systems.

While the above end-to-end systems achieved comparable ASR performance [9] or TTS quality [4] to human beings when their supervised training datasets are sufficiently large, the performance degrades significantly when the dataset is small [9]. Hence, many previous works [10]–[14] aimed to improve the ASR performance with smaller paired and larger unpaired training datasets, where "paired" means a supervised pair consisting of speech and corresponding transcribed text, and "unpaired" means speech only or text only unsupervised data.

For non end-to-end ASR models, unsupervised learning of the speech only or text only data could be achieved using e.g., a restricted Boltzmann machine (RBM) [11], [15] and a language model (LM) integration [16], [17]. However, these approaches are not able to learn ASR from speech only and text only data simultaneously, in other words, not able to learn a common representation between these modalities. Although the end-to-end ASR models also have succeeded in the LM integration [1], [12], but we still demand for the unsupervised learning method with speech only data. One of the effective approach to train the ASR model with speech only and text only data jointly is a TTS integration [13] that aims to generate paired speech and text data from unpaired speech and text for training the ASR and TTS models each other. It builds speech and text autoencoders by chaining ASR into TTS or TTS into ASR models in serial. It reports that the joint training the autoencoders with speech only or text only datasets improved the ASR performance. However, obviously, this approach needs large machine resource because its unrolled computation graph is at least twice larger than ones for each ASR and TTS model individually.

Instead of the chain of ASR and TTS models in serial, our method switches ASR and TTS components in parallel that makes the computation graph as short as each ASR and TTS model. As unsupervised models to exploit the unpaired data in a fully end-to-end manner, we introduce speech and text autoencoders built by switching the encoders and decoders employed in ASR and TTS models. In other words, the speech autoencoder consists of the ASR model's speech encoder and the TTS model's speech decoder, while the text autoencoder consists of the TTS model's text encoder and the ASR model's text decoder. This switching mechanism assumes that the ASR and TTS models can be combined in a common latent space because they are designed to encode speech/text and decode text/speech, respectively. To ensure this assumption, in addition to the ASR, and speech/text autoencoder loss functions, our semi-supervised objective minimizes distance between two distributions of encoded speech and text features, which we call "inter-domain loss". This switching mechanism using an encoder-decoder and an autoencoder with the inter-domain loss is succeeded in unsupervised image-to-image [18] or machine translation [19]. For the ASR task, we anticipate that the unsupervised training for speech-to-feature and feature-to-text transformation by the autoencoders improves speech-to-feature-to-text transformation in the ASR model when they share these transformations and latent space. This kind of loss function to obtain the common latent space is also known as "invariant representation learning" in the encoder decoder based robust ASR [20]. Additionally, we incorporate TTS loss into the multi-task loss to help the ASR model learn the transformation because both the intermediate features in the ASR and TTS models can be common between speech and text in our framework.

This study describes two major changes from previous semi-supervised end-to-end methods [10], [14]. First, while the previous methods only have the ASR model and text autoencoder, our new method combine four neural networks; the ASR model, TTS model, speech autoencoder, and text autoencoder. Note that, unlike [10], speech and text encoders do not share output layers anymore because we combine the different state-of-the-art architectures in ASR [21] and TTS [4] into our joint training framework.

Second, we use new inter-domain loss based on "maximum mean discrepancy" (MMD), which is kernel-based distance between two distributions [22]. In our previous framework, the inter-domain loss was Gaussian Kullback–Leibler (KL) divergence, which assumes that encoded speech and text distributions are Gaussians and requires unstable matrix inverse computation. We found that MMD is a more stable method because it is nonparametric and does not require density estimation.

We summarize our contributions below:

- We extend our previous semi-supervised end-to-end ASR method [10] so that we can exploit more speech and text datasets with TTS and autoencoders.
- Our proposed MMD based inter-domain loss is more stable and performs better than the KL based loss proposed in [10].
- Our system reduced the character error rate (CER) from 10.4% to 8.4% and word error rate (WER) from 20.6% to 18.0% with the small paired data TTS joint training and large unpaired data autoencoding in the LibriSpeech corpus [23].

2. PRELIMINARY

In this section, we describe our implementation of the ASR and TTS models that can be trained jointly in our proposed system.

2.1. End-to-End Speech Recognition

In this paper, the architecture of our end-to-end ASR model is based on [12]. The ASR model is an encoder-decoder that consists of two parts called "encoder" and "decoder" networks [5]. A speech encoder $\mathrm{enc}^X(\cdot)$ consists of one VGG layer [24] and four bidirectional long short-tem memory (BLSTM) with projection layers. The encoder receives an utterance consisting of a sequence of 80 dim log Mel filterbank $\mathbf{x} \in \mathbb{R}^{S \times 80}$, where S is the number of speech frames, and transforms it into an intermediate representation $\mathbf{h}^X \in \mathbb{R}^{S' \times U}$, where S' is the number of sub-sampled speech frames and U is the size of the encoded speech feature vector.

A text decoder network $\mathrm{dec}^Y(\cdot)$ consists of one embedding layer, one unidirectional LSTM layer with location based attention [1], and one fully connected layer. The decoder predicts the current token y_t in a symbol (e.g., character, subword, word) set $\{$'a', 'b', ..., <EOS>$\}$ using the encoder's output $\mathbf{h}^X$, the decoder's state vector $\mathbf{g}_t^Y$ and the previous token y_{t-1}. We describe this processing pipeline as follows:

$$\mathbf{h}^X = \mathrm{enc}^X(\mathbf{x}), \tag{1}$$

$$\left[\Pr(y_t|y_{t-1}, \mathbf{h}^X), \mathbf{g}_t^Y\right] = \mathrm{dec}^Y(y_{t-1}, \mathbf{h}^X, \mathbf{g}_{t-1}^Y), \tag{2}$$

where the initial state of the text decoder is $\mathbf{g}_0^Y = \mathbf{0}$. For simplicity, in the remainder of this paper, we omit the state $\mathbf{g}_t^Y$ and describe ASR loss in a sequence form as follows:

$$L_{\mathrm{ASR}} = -\log \Pr(\mathbf{y}|\mathbf{h}^X) = -\log \mathrm{dec}^Y(\mathbf{y}, \mathrm{enc}^X(\mathbf{x})), \tag{3}$$

where $\mathbf{y} = [y_1, y_2, \ldots, y_{|\mathbf{y}|}]$ is a predicted text, and $|\mathbf{y}|$ denotes the length of $\mathbf{y}$. Note that the decoder emits an <EOS> token when it predicts the end of a sequence.

2.2. End-to-End Text-to-Speech

The architecture of the end-to-end TTS model used in this paper is based on Tacotron2 [4], which consists of encoder and decoder networks as with the ASR model described in Section 2.1. Its text encoder $\mathrm{enc}^Y(\cdot)$ converts the character sequence $\mathbf{y}$ into an intermediate representation $\mathbf{h}^Y \in \mathbb{R}^{T \times U}$, where T is the length of character ids. $\mathrm{enc}^Y(\cdot)$ is composed of one embedding layer, three convolution layers and one BLSTM layer. Note that the encoded text vector size U is equal to the encoded speech vector. This configuration enables us to train the speech and text encoders jointly with the inter-domain loss described in the next section.

The speech decoder dec^X predicts the next speech the log Mel filterbank frame $\mathbf{x}_s$ from the encoded text input $\mathbf{h}^Y$, the previous frame $\mathbf{x}_{s-1}$, and the state vector $\mathbf{g}_s^X$. The speech decoder network $\mathrm{dec}^X(\cdot)$ consists of two subnetworks. The first is called "pre-net" and it receives the previous state vector $\mathbf{g}_{s-1}^X$ and the previous target speech frame $\mathbf{x}_{s-1}$ during training. Then it emits the state vector $\mathbf{g}_s^X$, the next speech frame $\mathbf{x}_s^{\mathrm{pre}}$ and binary flag b to halt the sequence generation. The pre-net is composed of a two-layer feedforward network with ReLU activation, three unidirectional LSTM layers with location based attention [1], and two output linear layers for the next speech frame and binary flag, respectively. The second subnetwork is a 5-layer convolutional network called "post-net", which adds residual vectors as post processing on the pre-net prediction $\mathbf{x}'_s$ to achieve a more realistic log Mel filterbank $\mathbf{x}_s^{\mathrm{post}}$. We describe this TTS pipeline as follows:

$$\mathbf{h}^Y = \mathrm{enc}^Y(\mathbf{y}), \tag{4}$$

$$\left[b_s, \mathbf{x}_s^{\mathrm{pre}}, \mathbf{x}_s^{\mathrm{post}}, \mathbf{g}_s^X\right] = \mathrm{dec}^X(\mathbf{x}_{s-1}, \mathbf{h}^Y, \mathbf{g}_{s-1}^X), \tag{5}$$

where the initial state of the text decoder is $\mathbf{g}_0^X = \mathbf{0}$. For simplicity, we describe a sequence form of Eq. (5) as

$$\left[\mathbf{b}, \mathbf{x}^{\mathrm{pre}}, \mathbf{x}^{\mathrm{post}}\right] = \mathrm{dec}^X(\mathbf{x}, \mathbf{h}^Y) \tag{6}$$

Finally, we define the TTS loss $\mathrm{Taco}(\cdot)$ proposed in [25] that includes binary cross entropy for the flag b_s and L1 [26] / L2 [4] norm for speech errors as follows:

$$\begin{aligned} L_{\mathrm{TTS}} &= \mathrm{Taco}(\mathbf{x}, \mathrm{dec}^X(\mathrm{enc}^Y(\mathbf{y}))) \\ &= \sum_{s=1}^{S} \left(\sum_{n=1}^{2} \sum_{\mathbf{x}'_s \in \{\mathbf{x}^{\mathrm{pre}}, \mathbf{x}^{\mathrm{post}}\}} |\mathbf{x}_s - \mathbf{x}'_s|^n - \log \Pr(b_s) \right). \end{aligned} \tag{7}$$

3. PROPOSED SEMI-SUPERVISED METHOD

Figure 1 shows the basic pipeline of our proposed method for semi-supervised learning to train speech and text autoencoders jointly with the ASR and TTS tasks in previous sections. Algorithm 1 defines in detail the procedure inside the pipeline.

First, our system samples a paired speech/text Z, unpaired speech X and text Y minibatches and it encodes them using speech and text encoders from the ASR and TTS models. Then we obtain the inter-domain loss L_{dom} between encoded features of paired speech $H^{X'}$ and text $H^{Y'}$, and also between encoded unpaired speech H^X and text H^Y. As unsupervised learning, speech and text decoders from the ASR and TTS models reconstruct the unpaired speech X and text Y from the encoded speech H^X and text H^Y and we compute their speech and text reconstruction errors $L_{\mathrm{SAE}}, L_{\mathrm{TAE}}$. At the same time, as supervised learning, the decoders also predict paired speech and text Z from encoded features of paired speech

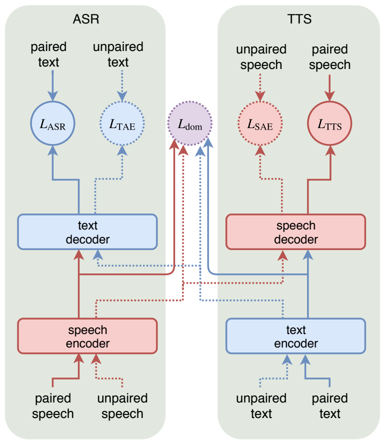

Fig. 1. Joint training pipeline of ASR and TTS models.

Algorithm 1 Semi-supervised ASR training algorithm using TTS and autoencoders.

```
unpaired speech and text datasets: $\mathcal{X}, \mathcal{Y}$.
paired speech-text dataset: $\mathcal{Z}$.
learning rates: $\alpha, \alpha_{\mathrm{TTS}}, \alpha_{\mathrm{SAE}}, \alpha_{\mathrm{TAE}}, \alpha_{\mathrm{dom}}$.
parameters of speech/text encoder, speech/text decoder: $\Phi^X, \Phi^Y, \Theta^X, \Theta^Y$.
procedure TRAIN($\Phi^X, \Phi^Y, \Theta^X, \Theta^Y$)
    for $i = 1, 2, \ldots, \max(\#\mathcal{X}, \#\mathcal{Y}, \#\mathcal{Z})$ do
        ▷ sample unpaired minibatches
        $X \sim \mathcal{X}, Y \sim \mathcal{Y}$
        ▷ sample one paired minibatch
        $Z \sim \mathcal{Z}$
        ▷ encode minibatches
        $H^X = \left\{\mathrm{enc}^X(\mathbf{x}) | \mathbf{x} \in X\right\}, H^Y = \left\{\mathrm{enc}^Y(\mathbf{y}) | \mathbf{y} \in Y\right\}$
        $H^{X'} = \left\{\mathrm{enc}^X(\mathbf{x}') | (\mathbf{x}', \mathbf{y}') \in Z\right\}$
        $H^{Y'} = \left\{\mathrm{enc}^Y(\mathbf{y}') | (\mathbf{x}', \mathbf{y}') \in Z\right\}$
        ▷ inter-domain loss
        if use KL then
            $L_{\mathrm{dom}} = \mathrm{KL}(\mathcal{N}(H^X) || \mathcal{N}(H^Y)) + \mathrm{KL}(\mathcal{N}(H^{X'}) || \mathcal{N}(H^{Y'}))$
        else if use MMD then
            $L_{\mathrm{dom}} = \mathrm{MMD}(H^X, H^Y) + \mathrm{MMD}(H^{X'}, H^{Y'})$
        else
            $L_{\mathrm{dom}} = 0$
        end if
        ▷ autoencoder loss
        $L_{\mathrm{SAE}} = \sum_{\mathbf{x} \in X} \mathrm{Taco}(\mathbf{x}, \mathrm{dec}^X(\mathrm{enc}^X(\mathbf{x})))$
        $L_{\mathrm{TAE}} = -\sum_{\mathbf{y} \in Y} \log \Pr(\mathbf{y})$
        ▷ supervised loss
        $L_{\mathrm{ASR}} = -\sum_{\mathbf{x}', \mathbf{y}' \in Z} \log \Pr(\mathbf{y}' | \mathbf{x}')$
        $L_{\mathrm{TTS}} = \sum_{(\mathbf{x}', \mathbf{y}') \in Z} \mathrm{Taco}(\mathbf{x}', \mathrm{dec}^X(\mathrm{enc}^Y(\mathbf{y}')))$
        ▷ multi task semi-supervised loss
        $L = L_{\mathrm{ASR}} + \alpha_{\mathrm{TTS}} L_{\mathrm{TTS}} + \alpha_{\mathrm{SAE}} L_{\mathrm{SAE}} + \alpha_{\mathrm{TAE}} L_{\mathrm{TAE}} + \alpha_{\mathrm{dom}} L_{\mathrm{dom}}$
        ▷ update parameters
        for $\Xi \in \left\{\Phi^X, \Phi^Y, \Theta^X, \Theta^Y\right\}$ do
            $\Xi \leftarrow \mathrm{Adam}_\alpha(\Xi, \frac{\partial L}{\partial \Xi})$
        end for
    end for
end procedure
```

$H^{X'}$ and text $H^{Y'}$ and we compute their ASR and TTS errors $L_{\mathrm{ASR}}, L_{\mathrm{TTS}}$ as described in the previous section. Finally, as joint optimization of these tasks, our system optimizes parameters of the ASR and TTS models to minimize the semi-supervised loss that is a weighted sum of $L_{\mathrm{dom}}, L_{\mathrm{SAE}}, L_{\mathrm{TAE}}, L_{\mathrm{ASR}}$, and L_{TTS}.

The reminder of this section describes more details of the speech/text autoencoder loss $L_{\mathrm{SAE}}, L_{\mathrm{TAE}}$ and inter-domain loss L_{dom} for unsupervised learning.

3.1. Autoencoder loss

The speech and text autoencoders reconstruct the input speech and text through their encoded feature, respectively, and they aim to minimize their reconstruction errors between the inputs and their decoded outputs. As the text autoencoder (TAE) loss, we use cross entropy loss

$$L_{\mathrm{TAE}} = -\log \Pr(\mathbf{y}) = -\log \mathrm{dec}^Y(\mathrm{enc}^Y(\mathbf{y})), \tag{8}$$

as well as the previous method [10]. ASR cross entropy loss defined by Eq. (3), and inter-domain loss. In addition, we newly introduce a speech autoencoder (SAE) loss

$$L_{\mathrm{SAE}} = \mathrm{Taco}(\mathbf{x}, \mathrm{dec}^X(\mathrm{enc}^X(\mathbf{x}))) \tag{9}$$

based on the TTS loss L_{TTS} defined by Eq. (7). Note that unlike the previous method [10], our new method simply switches encoders and decoders from speech or text to speech to text without sharing layers in the speech and text encoders because ASR and TTS models have different output layers in encoder networks, for example, BLSTM with projection in the speech encoder and BLSTM without projection in the text encoder.

3.2. Inter-domain loss

We use the inter-domain loss as dissimilarity between distributions of encoded speech H^X and text H^Y. It enforces speech and text encoders to map inputs into the common latent space for switching the decoders. In [10], we used Gaussian KL divergence as the inter-domain loss

$$L_{\mathrm{dom}} = \mathrm{KL}(\mathcal{N}(H^X) || \mathcal{N}(H^Y)), \tag{10}$$

where $\mathcal{N}(\cdot)$ is a multivariate Gaussian with empirical mean and covariance on the minibatch input. However, as regards the stability of Gaussian KL, we found that its computation of the covariance matrix inverse sometimes become unstable in our experiment.

Therefore, we introduce more stable inter-domain loss using kernel based distance called maximum mean discrepancy (MMD). MMD is a statistic used to measure the difference between two distributions [22]. Let minibatches of paired or unpaired encoded speech and text features $H^X \in \mathbb{R}^{\#X \times U}$ and $H^Y \in \mathbb{R}^{\#Y \times U}$, where $\#X, \#Y$, and U are the number of samples in a speech/text minibatch and encoded the feature size described in Section 2.1, 2.2, respectively. In our system, we compute the MMD between encoded speech/text minibatches by using Gaussian kernel in [22] as follows:

$$m_X = \sum_{i=1}^{\#X} \sum_{j=1}^{\#X} \sum_{k=1}^{U} H_{i,k}^X H_{j,k}^X,$$

$$m_Y = \sum_{i=1}^{\#Y} \sum_{j=1}^{\#Y} \sum_{k=1}^{U} H_{i,k}^Y H_{j,k}^Y,$$

$$k_X = \frac{\sum_{i=1}^{\#X} \sum_{j=1}^{\#X} \exp\left(\sum_{k=1}^{U} H_{i,k}^X H_{j,k}^X - m_X\right)}{\#X^2},$$

$$k_Y = \frac{\sum_{i=1}^{\#Y} \sum_{j=1}^{\#Y} \exp\left(\sum_{k=1}^{U} H_{i,k}^Y H_{j,k}^Y - m_Y\right)}{\#Y^2},$$

$$k_{X,Y} = \frac{\sum_{i=1}^{\#X}\sum_{j=1}^{\#Y} \exp\left(\sum_{k=1}^{U} H^X_{i,k} H^Y_{j,k} - \frac{m_X + m_Y}{2}\right)}{\#X\#Y},$$

$$\mathrm{MMD}(H^X, H^Y) = k_X - 2k_{X,Y} + k_Y. \tag{11}$$

Because this definition has simple operations only, MMD is more stable than Gaussian KL. Furthermore, MMD can be dissimilarity between any distribution because it does not need for a density estimation, while Gaussian KL only works as dissimilarity between normal distributions. In our preliminary experiment on the WSJ corpus, the MMD resulted in a CER 13.9% while it was 14.4% for KL [10][1].

4. EXPERIMENT

4.1. Settings

We used the LibriSpeech [23] corpus to investigate our method in this experiment because it has strong baselines for character based ASR and TTS models in ESPnet [21]. We used `train_clean_100` as a supervised training dataset and `dev_clean_100` as a validation dataset for ASR and TTS baseline systems. Because LibriSpeech has multiple speakers, we concatenated pretrained x-vector [27][2] based speaker embedding into the speech decoder input. In addition, we used speech only data from `train_clean_360` and and text only data from `train_other_500` as unsupervised training datasets for speech autoencoding and text autoencoding, respectively. We also used this internal text only dataset or external text dataset `librispeech-lm-norm.txt.gz` for training character and word based LMs. Note that the number suffix in the dataset name indicates recording time in hours.

First, we pretrained the supervised ASR and TTS models with only paired dataset `train_clean_100` by ESPnet's recipes until their convergence. We regard this pretrained ASR model as the baseline ASR system. Then, we retrained the baseline models using the Adam optimizer [28] with a minibatch size of 64 and a learning rate $\alpha = 10^{-5}$ for 10 epochs. We searched for the multi-task loss weights $\alpha_{\mathrm{TTS}}, \alpha_{\mathrm{SAE}}, \alpha_{\mathrm{TAE}}$ in Algorithm 1 from $\{0.1, 1.0, 10.0\}$ during training and LM weight from $\{0.1, 0.2, 0.3, 0.5\}$ during decoding, respectively. We searched the best training and decoding settings by monitoring the WER of `dev_clean`. Note that as described in Algorithm 1, it is important to sample alternately from paired and speech/text only datasets to make paired and unpaired training updates balanced. When we sampled data in proportion to the number of samples in paired/speech-only/text-only datasets, the ASR performance became worse than that of the pretrained baseline model because the numbers of training updates were unbalanced. To provide more details, we will share our public source code.

4.2. Results

Table 1 summarizes the CER/WER obtained with the LibriSpeech `test_clean` dataset. A checkmark "✓" in the table indicates the loss functions trained jointly with the ASR loss for each system. Unlike the results obtained with WSJ [10], we observed clear improvements by joint decoding with LMs. The word based RNNLM recently proposed in [29] that learned the external data (ext LM) achieved the better result than word based RNNLM (word LM) and the character based LM (char LM) that learned the internal text only data. This baseline system using the ext LM results in the CER of 10.4% and WER of 20.6%. Since the LM is unsupervised model using the text only dataset, our proposed system also used this ext LM for decoding. We observed further decreases in CER/WER with every task combination for semi-supervised training with the MMD based loss, while the KL-based loss especially using SAE degraded CER/WER. Finally, our proposed method using all the tasks with MMD provided the best CER of 8.4% and WER of 18.0%.

Table 1. Character/word error rate (CER/WER) on `dev_clean` and `test_clean` of LibriSpeech. Note that we used only `train_clean_100` as a paired ASR/TTS dataset.

	multi-task loss				dev clean	test clean
	ASR	TTS	SAE	TAE	CER/WER	CER/WER
baseline	✓	-	-	-	14.1 / 26.2	15.0 / 25.0
+ char LM	✓	-	-	-	12.2 / 22.8	11.9 / 22.5
+ word LM	✓	-	-	-	12.1 / 21.3	11.7 / 22.0
+ ext LM	✓	-	-	-	10.3 / 20.6	10.4 / 20.6
+ KL	✓	✓	-	-	9.6 / 19.4	9.5 / 19.2
	✓	-	✓	-	11.6 / 21.8	12.0 / 22.1
	✓	-	-	✓	10.3 / 19.7	10.5 / 19.9
	✓	✓	✓	✓	12.2 / 21.8	12.1 / 21.8
+MMD	✓	✓	-	-	9.6 / 19.1	9.5 / 18.9
	✓	-	✓	-	9.2 / 18.9	8.7 / 18.4
	✓	-	-	✓	9.4 / 18.9	9.0 / 18.4
	✓	✓	✓	✓	**8.9 / 18.5**	**8.4 / 18.0**

Altough the TTS loss only leveraged the small paired dataset, it improved the ASR performance as well as other unsupervised loss functions. This result indicates that the latent feature between ASR and TTS can be common. However, while the ASR loss decreased significantly during the TTS joint training, the TTS loss did not decrease from that of the baseline TTS model when it was combined with the ASR model. We expect that the TTS loss can be traded off against the ASR loss or a regularization term that helps the encoded feature to become common for both speech and text. Otherwise, we need to initialize the TTS model by a single speaker task as described in [13]. Overall, we confirmed that MMD was more stable and results lower CER/WER than KL. One plausible reason for the degradation in KL with the SAE is the instability of the matrix inverse that we observed some warnings from LAPACK. Note that, this computation had no problem in the previous experiment in [10] but it lacks of the SAE.

5. CONCLUSION

In this study, we proposed extensions of the semi-supervised end-to-end ASR. It can exploit 100 hours of paired speech/text, 300 hours of speech only and 460 hours of text only data with the TTS model, speech and text autoencoders. Through experiments, our semi-supervised model with MMD based inter-domain loss reduced the CER/WER from 10.4/20.6% to 8.4/18.0%. In future work, we plan to extend the proposed approach to fully unsupervised ASR.

Acknowledgement

We would like to thank the multilingual end-to-end ASR team for incomplete data project in the the Fifth Frederick Jelinek Memorial Summer Workshop for their valuable discussions and comments.

[1]For more details, see `https://github.com/nttcslab-sp/espnet-semi-supervised`

[2]For more details, see `https://david-ryan-snyder.github.io/2017/10/04/model_sre16_v2.html`

6. REFERENCES

[1] D. Bahdanau, J. Chorowski, D. Serdyuk, and Y. Bengio, "End-to-End Attention-based Large Vocabulary Speech Recognition," *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 4945–4949, 2016.

[2] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2016-May, pp. 4960–4964, 2016.

[3] W. Ping, K. Peng, A. Gibiansky, *et al.*, "Deep Voice 3: 2000-Speaker Neural Text-to-Speech," *CoRR*, vol. abs/1710.07654, 2017.

[4] J. Shen, R. Pang, R. J. Weiss, *et al.*, "Natural TTS Synthesis by Conditioning Wavenet on MEL Spectrogram Predictions," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018, pp. 4779–4783.

[5] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," *Neural Information Processing Systems*, pp. 3104–3112, 2014.

[6] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *International Conference on Learning Representations*, 2015.

[7] G. Hinton, L. Deng, D. Yu, *et al.*, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 82–97, 2012.

[8] P. Taylor, *Text-to-Speech Synthesis.*, 2009.

[9] D. Amodei, S. Ananthanarayanan, R. Anubhai, *et al.*, "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin," in *International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 48, 2016, pp. 173–182.

[10] S. Karita, S. Watanabe, T. Iwata, A. Ogawa, and M. Delcroix, "Semi-Supervised End-to-End Speech Recognition," in *Proceedings of International Conference on Spoken Language Processing, INTERSPEECH*, 2018, pp. 2–6.

[11] K. Vesely, M. Hannemann, and L. Burget, "Semi-Supervised Training of Deep Neural Networks," in *IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 267–272.

[12] T. Hori, S. Watanabe, Y. Zhang, and W. Chan, "Advances in Joint CTC-Attention Based End-to-End Speech Recognition with a Deep CNN Encoder and RNN-LM," in *Interspeech*, 2017, pp. 949–953.

[13] A. Tjandra, S. Sakti, and S. Nakamura, "Listening while Speaking: Speech Chain by Deep Learning," in *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2017, pp. 301–308.

[14] T. Hayashi, S. Watanabe, Y. Zhang, *et al.*, "Back-Translation-Style Data Augmentation for End-to-End ASR," in *IEEE Spoken Language Technology Workshop (to appear)*, 2018.

[15] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.

[16] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent Neural Network based Language Model," *Interspeech*, no. September, pp. 1045–1048, 2010.

[17] E. Dikici and M. Saraçlar, "Semi-supervised and Unsupervised Discriminative Language Model Training for Automatic Speech Recognition," *Speech Communication*, vol. 83, pp. 54–63, 2016.

[18] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised Image-to-Image Translation Networks," in *Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., 2017, pp. 700–708.

[19] G. Lample, L. Denoyer, and M. Ranzato, "Unsupervised Machine Translation Using Monolingual Corpora Only," *International Conference on Learning Representation*, 2018.

[20] D. Liang, Z. Huang, and Z. C. Lipton, "Learning noise-invariant representations for robust speech recognition," *arXiv preprint arXiv:1807.06610*, 2018.

[21] S. Watanabe, T. Hori, S. Karita, *et al.*, "ESPnet: End-to-End Speech Processing Toolkit," in *Proceedings of International Conference on Spoken Language Processing, INTERSPEECH*, 2018, pp. 2207–2211.

[22] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A Kernel Two-sample Test," *J. Mach. Learn. Res.*, vol. 13, pp. 723–773, 2012.

[23] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "LibriSpeech: An ASR corpus based on public domain audio books," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2015, pp. 5206–5210.

[24] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *International Conference on Learning Representation*, 2015.

[25] Y. Jia, Y. Zhang, R. J. Weiss, *et al.*, "Transfer Learning from Speaker Verification to Multispeaker Text-To-Speech Synthesis," *arXiv preprint arXiv:1806.04558*, 2018.

[26] Y. Wang, R. Skerry-Ryan, D. Stanton, *et al.*, "Tacotron: Towards End-to-End Speech Synthesis," in *Proceedings of International Conference on Spoken Language Processing, INTERSPEECH*, 2017, pp. 4006–4010.

[27] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN Embeddings for Speaker Recognition," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, IEEE, 2018.

[28] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *International Conference on Learning Representations*, pp. 1–13, 2014.

[29] T. Hori, J. Cho, and S. Watanabe, "End-to-end speech recognition with word-based rnn language models," *arXiv preprint arXiv:1808.02608*, 2018.