Near Unsupervised Adversarial Text to Speech and Automatic Speech Recognition

Lucas Kabela[*]
Department of Computer Science
University of Texas at Austin
lucaskabela@utexas.edu

Bill Yang[†]
Department of Computer Science
University of Texas at Austin
billyang@utexas.edu

Jie Hao Liao[‡]
Department of Computer Science
University of Texas at Austin
jhliao@utexas.edu

Abstract

Low resource/unsupervised approaches treating TTS and ASR independently do not incorporate the closed feedback loop of speech, whereas humans speak, listen, and process what has been spoken. The insight that TTS and ASR are dual tasks gave rise to a learning method called dual transformation. Ren et al. (2019) show its effectiveness using 200 paired and 12,300 unpaired samples of text and speech, reaching 11.7% PER and 2.68 MOS. The data is trained on supervised ASR/TTS, denoising autoencoding, and dual transformation tasks. However, an implicit assumption in these objectives is encoded speech and text representations align to similar areas of latent space. We make this constraint explicit through adding an adversarial loss to enforce alignment. We find the discriminator consistently improves performance across RNN and Transformer architectures and combinations of all three tasks. Our best model is able to achieve 72.62% PER and 0.034 average MSE loss against the ground truth mel-spectrogram with teacher forcing. While we were not able to reproduce the performance in Ren et al. (2019) due to difficulty balancing text and speech losses, we were able to produce evidence that the adversarial term improves learning for joint TTS and ASR systems.

1 Introduction

As a result of advances in end-to-end (E2E) neural approaches, data efficiency is of significant concern for text to speech (TTS) and automatic speech recognition (ASR). To address this, many techniques for low resource/unsupervised learning have been proposed in TTS and ASR. However, training either system in isolation does not take advantage of how humans experience speech: humans speak, listen, and review via a closed auditory feedback loop. One technique which leverages this is "dual learning", which is similar to "back translation" in neural machine translation (NMT) (Sennrich et al., 2016) and was first used for speech applications by Tjandra et al. (2017). In its original formulation, the dual learning task only used the TTS and ASR objectives, which does not fully leverage the data. Ren et al. (2019) introduce an "almost" unsupervised approach for TTS and ASR, which adds a denoising autoencoder (DAE) task to the dual transformation (DT) and supervised ASR/TTS objectives, achieving 11.7% PER and 2.68 MOS.

[*]Contributed RNN and Transformer implementations and training code.
[†]Contributed dataset, processing, and discriminator implementation.
[‡]Contributed pre/post nets, vocoder, implementation fix-ups, and trained models.

Preprint. Under review.

In these dual learning approaches for speech applications, there has been no constraint on the latent representations the encoders produce, despite the implicit requirement they are aligned for the decoders to learn efficiently, as decoders use the outputs from either encoder depending on the task. This oversight allows the potential for loose or no alignment, which could harm performance. In NMT, Lample et al. (2017) leverage an adversarial discriminator to enforce alignment of embedding spaces. Consequently, we propose the use of an adversarial discriminator with the speech tasks introduced in Ren et al. (2019). We hypothesize the adversary will enforce greater alignment between semantically identical speech and text, and make learning more efficient between the various tasks.

We evaluate this hypothesis on dual speech and text autoencoders[4], with an RNN model based on Tacotron 2 (Shen et al., 2017) and a Transformer model based on Transformer TTS (Li et al., 2018). These architectures are then trained on 200 paired and 12,300 unpaired examples from the LJSpeech dataset (Ito and Johnson, 2017). Our best full model achieves 72.62% PER, and we note that the discriminator aids learning for all objectives. To summarize, we contribute: (1) Evaluation of supervised, unsupervised, and dual transformation objectives with an adversarial loss in speech applications which demonstrates the adversarial term consistently improves model performance. (2) A potential unsupervised learning technique for ASR and TTS given phonetic transcription of text. Our performance using fully unsupervised data with an adversary improves by a 64.22% difference in PER on RNNs. (3) Evaluation of the tasks introduced in Ren et al. (2019) on a RNN architecture, which is within 4% PER and .01 per-frame average MSE from ground truth mel spectogram predictions of Transformers.

2 Related Works

TTS and ASR systems have transitioned from complex systems with independent components (Juang and Rabiner, 2005) to E2E deep neural networks with many potential formulations. To maximize the similarity in our objectives and architectures, we use the Seq2Seq (Chorowski et al., 2014; Wang et al., 2017) formulation for our speech and text models. While transformers (Vaswani et al., 2017) have caused a shift away from RNNs, and there has been some comparative work between Transformers and RNNs for TTS and ASR applications (Karita et al., 2019), direct comparison has been absent from the speech dual learning literature. Hence, we include an RNN architecture (Shen et al., 2017) and a Transformer architecture (Li et al., 2018) in our experiments.

Inspired by data efficiency and how humans learn, there have been attempts at low resource and unsupervised learning in TTS and ASR. Unsupervised approaches primarily focused on automatically discovering sub-word units in both TTS and ASR applications (Yeh et al., 2018; Muthukumar and Black, 2014; Chen et al., 2018), but such approaches still lag behind supervised models (Dunbar et al., 2020). Low resource learning on the other hand has seen greater success, with approaches commonly using a model to bootstrap and augment data via pseudo-labeling or transfer learning (Kahn et al., 2019; Zhou et al., 2018). Our work is closely related to Drexler and Glass (2018), which uses text and speech autoencoders with a discriminator; however, they only train an ASR model, and do not use the dual task of TTS.

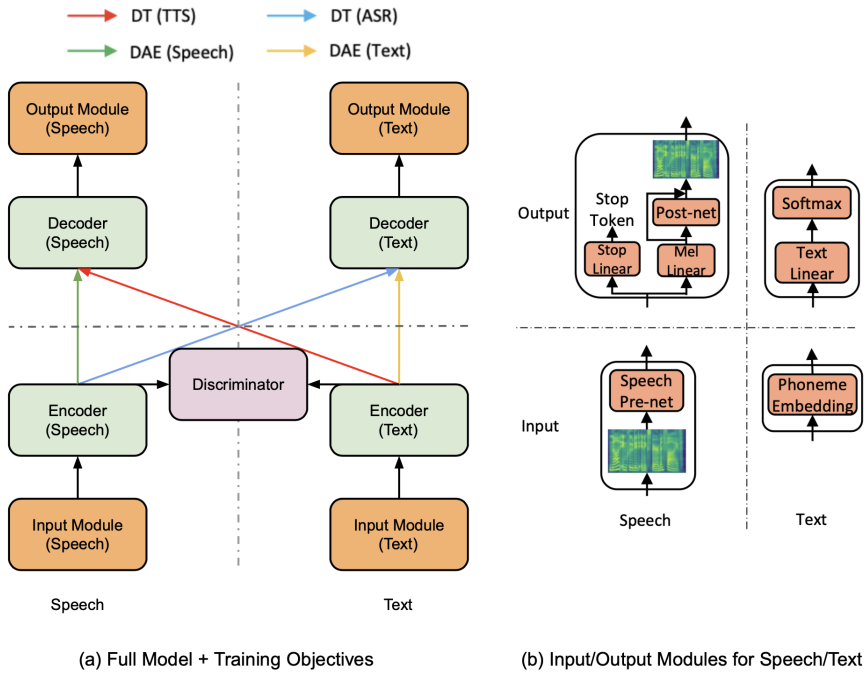
The observation that some tasks are the "dual" of each other influenced a branch of research known as dual learning. This became popular in neural machine translation (NMT) with Sennrich et al. (2016); Xia et al. (2016) where translation systems could be trained on monolingual corpus by translating sentences from a source language to a target language, then back-translating to the source language. Lample et al. (2017) found adding a discriminator adversary to enforce alignment between the encoder latent spaces of different languages improves learning in this setup considerably. In speech, the idea that speech is a closed system, or humans hear their own speech and can further process it, led Tjandra et al. (2017) to use TTS as the dual of ASR. This work uses a large dataset of 10,000 paired examples, and achieves as low as a 5% CER and 6 MSE loss (summed). Ren et al. (2019), which our work closely resembles, added autoencoding losses to this formulation, and was able to achieve similar performance in low resource learning by using only 200 paired examples. However, there was no constraint on the latent space alignment, which we attempt to add via a discriminator.

[4]Our code is available at `https://github.com/Lucaskabela/UNAST`

3 Methodology

3.1 Architecture

Our model incorporates 4 basic input/output net modules for speech and text, and 5 primary components, seen in Figure 1: speech (θ_s^{enc}) and text encoders (θ_t^{enc}), speech (θ_s^{dec}) and text decoders (θ_t^{dec}), and a discriminator (θ^D), which attempts to predict the input domain, speech or text, of an encoded sequence representation. We report the number of parameters on our full model in our results, and a summary of hyperparameters in our Appendix Table 5.



(a) Full Model + Training Objectives (b) Input/Output Modules for Speech/Text

Figure 1: Figure (a): Full model adapted from Ren et al. (2019) with colored arrows showing the flow of data for the various training tasks. Figure (b): Input and ouput modules for the encoder/decoder model from Ren et al. (2019).

Encoder and Decoder For our speech and text autoencoders, $\theta_s^{enc}/\theta_s^{dec}$, and $\theta_t^{enc}/\theta_t^{dec}$, we choose to investigate both RNNs from Shen et al. (2017) and Transformers from Li et al. (2018). Both transformers and RNNs are able to attend to entire encoded representations of sequences, but RNNs additionally generate autoregressively from a single encoded hidden state. Thus we anticipate the forced alignment of speech and text encoder representations might have a stronger affect on RNNs and wish to compare the two architectures.

Input/Output Nets Following (Ren et al., 2019), we use the same input and output nets that enhance and format the raw input and output of our network. Our speech input module is a 2 layer fully-connected network (FCN), while our text input module is an embedding layer. We also experiment with adding a 3 layer convolutional neural network (CNN) before our text encoder to enable temporal context. Our speech output module is a linear layer that outputs a stop probability and a 5 layer CNN with residual to enhance the resolution of our mel-spectograms. Our text postnet is a linear layer and softmax which predicts phonemes directly. These nets can be seen in Figure 1b.

Discriminator Some related works use feed-forward networks (Li et al., 2020; Jiang et al., 2021) as adversaries. However, due to our use of the transformer model and LSTMs with attention, we want to enforce alignment of the entire latent representation or sequence instead of any one state, since the decoder is given access to the entire embedded sequence of the input. Thus, we use a two-layer bidirectional-LSTM (Huang et al., 2015) as our discriminator θ^D, which we believe should be a sufficiently effective adversary for our model.

3.2 Training Objectives and Loss Functions

Our goal is to train our model using unpaired data with few paired examples. Following (Ren et al., 2019), we use both supervised and unsupervised training objectives. Each of these training objectives operate on either a speech input x or text input y. For speech objectives, we use mean squared error (MSE) as the primary loss function following (Shen et al., 2017), while text objectives use cross entropy (CE). Since our speech model also outputs a stop prediction used in inference, the speech loss also includes a stop prediction loss, which is the binary cross entropy between the stop frame predictions and the ground truth stop, which is the last frame.

Denoising Autoencoder Denoising autoencoding (Vincent et al., 2008) has proven to be an effective method for unsupervised learning by corrupting input values and attempting to reproduce the original inputs. We use a simple noising function, $C()$ in which phonemes in text and frames in speech are masked by zero with probability p=0.3, following Ren et al. (2019). The model must then reconstruct the original data from the corrupted version, giving the loss functions:

$$\mathcal{L}_s^{dae} = MSE\left(x, \theta_s^{dec}(\theta_s^{enc}(C(x)))\right);\ L_t^{dae} = CE\left(y, \theta_t^{dec}(\theta_t^{enc}(C(y)))\right) \tag{1}$$

Thus, our total loss function for denoising autoencoders is the sum of speech and text

$$\mathcal{L}^{dae} = \mathcal{L}_s^{dae} + \mathcal{L}_t^{dae} \tag{2}$$

Dual Transformation Due to the dual nature of speech and text, we use the dual transformation to further utilize unsupervised training examples. Given speech input x, we use an ASR system to generate pseudo text $\hat{y} = \theta_t^{dec}(\theta_s^{enc}(x))$. Then we use a TTS system to recreate the speech input by $\theta_s^{dec}(\theta_t^{enc}(\hat{y}))$. Similarly for a text input y, we generate pseudo speech $\hat{x} = \theta_s^{dec}(\theta_t^{enc}(y))$, and recreate by $\theta_t^{dec}(\theta_s^{enc}(\hat{x}))$. We define the following loss functions for DT on speech and text

$$\mathcal{L}_s^{dt} = MSE\left(x, \theta_s^{dec}(\theta_t^{enc}(\hat{y}))\right);\ \mathcal{L}_t^{dt} = CE\left(y, \theta_t^{dec}(\theta_s^{enc}(\hat{x}))\right) \tag{3}$$

and the total DT loss as

$$\mathcal{L}^{dt} = \mathcal{L}_s^{dt} + \mathcal{L}_t^{dt} \tag{4}$$

Supervised Learning We also train our model using a limited number of paired training examples, which aids in learning additional structure. To avoid overfitting, we utilize frame and frequency masking components of SpecAugment (Park et al., 2019), with frequency mask of 20, time mask of 100, and replacement by mean. Thus, given a ground truth pair $x_i, y_i \in \mathcal{D}$, we can evaluate our model on both the TTS and ASR tasks. This results in the following loss functions for both TTS and ASR

$$\mathcal{L}^{tts} = MSE\left(x_i, \theta_s^{dec}(\theta_t^{enc}(y_i))\right);\ \mathcal{L}^{asr} = CE\left(y_i, \theta_t^{dec}(\theta_s^{enc}(x_i))\right) \tag{5}$$

and total supervised loss as

$$\mathcal{L}^{sup} = \mathcal{L}^{tts} + \mathcal{L}^{asr} \tag{6}$$

Discriminator Adversarial Training We also explore adding a discriminator on the output of encoders. This decision is motivated by the fact our decoder models must learn to decode the outputs from both encoders, so we wish to align the embedding spaces to improve the performance of the decoders. Following Lample et al. (2017), we use a smoothing factor of .1 on our discriminator predictions. To train the discriminator, we use the following loss:

$$\mathcal{L}^{disc} = CE\left(s, \theta^D(\theta_s^{enc}(x))\right) + CE\left(t, \theta^D(\theta_t^{enc}(y))\right) \tag{7}$$

where the discriminator outputs a single prediction, s speech or t text, and tries to learn the output distribution of each encoder. The encoder parameters are frozen for this step.

For our other tasks, the encoders try to fool the discriminator which requires aligning the encoders' embedding spaces. We freeze the discriminator parameters here. This task maximizes the adversary loss of the correct discriminator predictions by minimizing the loss of the encoder representations being incorrectly identified by the discriminator, or

$$\mathcal{L}^{adv} = CE\left(t, \theta^D(\theta_s^{enc}(x))\right) + CE\left(s, \theta^D(\theta_t^{enc}(y))\right) \tag{8}$$

Thus when training our model with the discriminator, our new losses for the three tasks become:

$$\mathcal{L}^{dae,adv} = \mathcal{L}^{dae} + \mathcal{L}^{adv};\ \mathcal{L}^{dt,adv} = \mathcal{L}^{dt} + \mathcal{L}^{adv};\ \mathcal{L}^{sup,adv} = \mathcal{L}^{sup} + \mathcal{L}^{adv} \tag{9}$$

4 Experimentation and Results

4.1 Dataset

We use the LJSpeech dataset (Ito and Johnson, 2017), which contains roughly 24 hours of labeled audio data. Following Ren et al. (2019), we split the 13,100 clips in the dataset into a train set of 200 supervised examples, 12,300 unsupervised examples, and a validation and test set of 300 examples each. This results in only about 20 minutes of supervised data.

We relax the problem by preprocessing the text into phonemes as done in Ren et al. (2019). While there are many neural methods to convert text to phonemes (Yolchuyeva et al., 2019), Li et al. (2018) found many learned techniques were not perfect and chose to use a rule based approach. For ease of experimentation and reproducibility, we used the CMU Pronouncing dictionary. We first clean the text, removing punctuation and converting to lower case, and then use the CMU pronouncing dictionary to get a phoneme transcription for each word in the input sequence.

We follow Li et al. (2018) and convert the speech audio waveforms into normalized mel spectrograms. We use the same parameters found in a re-implementation[5] of the paper: a sampling rate of 22050, a hop length of .0125 seconds, window length of .05 seconds, and 80 filterbanks.

4.2 Training Details

We train all models for 8 days or until convergence on NVIDIA P100 GPUs on Google Colab or NVIDIA GTX 1080-Ti GPUs on UT Condor Cluster and TACC. Due to the autoregressive generation for the DT task, the runtime is O(n^3) for Transformers. This results in much longer training time per epoch for transformers, and consequently we reach roughly 150 epochs for RNN models and only 80 epochs for Transformers. Each epoch involves 50 total steps, where each step comprised of accumulating gradients on 4 denoising autoencoding, 8 dual transformation, and 4 supervised batches. If a discriminator is present, then each step also accumulates gradients on 4 discriminator batches. Due to our small training environment, we were constrained to batch sizes of 4 examples for training, but the gradient accumulation should emulate larger batch sizes.

4.3 Evaluation Metrics

We evaluate the ASR model on phoneme error rate (PER), which is our primary form of evaluation. This is computed by finding the number of Substitutions ($\mathcal{S}$), Insertions ($\mathcal{I}$), and Deletions ($\mathcal{D}$) required to edit a prediction $\hat{y}$ to ground truth y with length $\mathcal{N}$:

$$\text{PER}(y, \hat{y}) = \frac{\mathcal{S} + \mathcal{I} + \mathcal{D}}{\mathcal{N}} \times 100 \tag{10}$$

To evaluate the TTS model, we report test-set averaged MSE loss as a quantitative measure in the difference between the predicted and ground truth spectrograms. While human evaluation and mean opinion score (MOS) are usually used to evaluate speech, we avoided collecting such scores due to resource limitations. Similarly, we also report the test-set ASR loss as a source of further metrics.

4.4 Results

Our main results can be found in Table 1. We use the models with the best PER on the dev set using early stopping, which commonly occurred around 130 epochs for RNN and 70 for Transformers. Due to resource limitations, some models were unable to train this long. A list of the best epoch and total epochs trained for each model is reported in the Appendix Table 8.

Our transformer model trained on all objectives and with discriminator loss produces the best PER, with the RNNs not far behind in PER in comparison. Unfortunately, none of our models achieved better than 70% PER nor produced intelligible speech. We attribute this to difficulty in the models learning both speech and text on so many objectives. Consequently, the models learned to optimize text predictions earlier since the CE loss, which is about 2 in the first epoch, is much greater than the MSE loss, which reached about .05 in the first few epochs. As seen in Lample et al. (2017), weighting

[5] https://github.com/soobinseo/Transformer-TTS

our objectives may be necessary for efficient learning. Fortunately, our hypothesis on discriminators is still supported, with both RNNs and Transformers acquiring similar gains of 3% PER improvement and qualitative improvements to the mel spectograms when using the discriminator loss. While the discriminator caused small increases in most test set loss values, this is expected as it enforced greater constraints on latent space. Finally, we note the RNN vs Transformer difference is not very pronounced in our results, but this is likely a result of failing to reach strong convergence.

	# Params	PER	TTS	ASR
Supervised (Best)	12.1 M	73.25	0.079	4.887
Unsupervised (Best)	12.1 M	74.96	0.106	1.307
RNN (LSA)	11.7 M	78.14	0.044	2.013
+ D	12.1 M	76.74	0.121	**1.331**
Transformer (Inverse)	17.1 M	75.39	**0.034**	1.582
+ D	17.3 M	**72.62**	0.087	1.426

Table 1: Table for full results

4.5 Ablations

Training Objectives To compare the effects of our training objectives, we analyze the results of training with only supervised or only unsupervised objectives in Table 2. Our supervised models train on only the 200 paired examples and supervised TTS and ASR tasks while the unsupervised models train on only the 12,300 unpaired examples and the DAE and DT tasks.

	# Params	PER	TTS	ASR
Supervised RNN	11.7 M	82.89	**0.041**	6.381
+ D	12.1 M	**73.25**	0.079	**4.887**
Supervised Transformer	17.1 M	74.33	0.102	4.985
+ D	17.3 M	74.05	0.120	5.342
Unsupervised RNN	11.7 M	145.87	**0.069**	1.344
+ D	12.1 M	**74.96**	0.106	**1.307**
Unsupervised Transformer	17.1 M	99.17	0.122	1.512
+ D (*)	17.3 M	172.54	0.113	1.636
Ours (Best)	17.3 M	72.62	0.087	1.426

Table 2: Table for ablating training objectives. (*) Denotes model was not able to train for full amount of epochs due to resource constraints, which harmed performance.

We find on most ablations, the discriminator improves performance on PER. On the few supervised examples, we also noticed that it helps prevent overfitting on the training data. Our unsupervised training shows substantial improvement with the discriminator, achieving close to our supervised levels in performance. This translates to a 64.22% improvement in PER for our RNN, and suggest that adding a discriminator may enable dual transformation to be used effectively for unsupervised learning, which has not been documented in previous research as far as we are aware. We note while our models with discriminator was unable to finish training, it showed consistent trends in performing better than the models without a discriminator in PER, and prevented some objectives from overfitting while the other objectives underfit.

Architecture Changes We performed small variations on the training architectures to try and improve the model performance and test hypothesis in the architectures we base our model on. We vary the attention of our RNN, Luong (Luong et al., 2015) vs. LSA (Chorowski et al., 2015), and the schedule of our transformer, a linear decay learning rate scheduler used in Ren et al. (2019) vs. the inverse decay learning rate scheduler from Vaswani et al. (2017). We report our the results in Table 3.

We find that LSA generally outperforms Luong for our RNNs, conforming with the hypothesis in Shen et al. (2017). Interestingly, the Transformer model trained on inverse LR decay outperforms the linear LR decay, which differs from the approach used in Ren et al. (2019), so we report Transformers with the inverse schedule as our best model. However, we did not reach the same performance as Ren et al. (2019), so it is possible this result would be different with more training time and resources.

	Params	PER	TTS	ASR
RNN				
+ Luong	11.7 M	86.65	0.044	1.963
+ Luong + D	12.1 M	81.38	0.129	**1.309**
+ LSA	11.7 M	78.14	**0.044**	2.013
+ LSA + D	12.1 M	**76.74**	0.121	1.331
Transformer				
+ Linear	17.1 M	85.12	0.118	1.491
+ Linear + D	17.3 M	75.98	0.099	**1.42**
+ Inverse	17.1 M	75.39	**0.034**	1.582
+ Inverse + D	17.3 M	**72.62**	0.087	1.426

Table 3: Table for minor architecture ablations

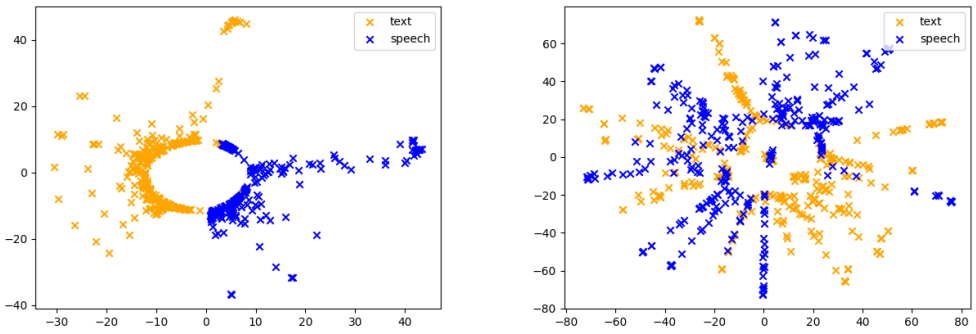
Speech Loss Function After noting our model was not learning speech properly, we found our MSE speech loss values were small relative to our other loss values. We attempted to change MSE loss to L1 loss and sum rather than taking the mean over the spectrograms. We compare these results to our RNN model trained using MSE loss. We still report test-set average MSE loss for both models, as this is a consistent measure of closeness to the ground-truth spectrograms in the test set. The results are reported in Table 4. While we were unable to fully train the L1 loss for the RNNs, only about 55 epochs, they converged surprisingly well. Nonetheless, we deemed it unable to be concluded if using the L1 loss would be an improvement, as it did not appear to be reaching performance in Ren et al. (2019), and we decided to continue using MSE loss for our experiments.

	PER	TTS	ASR
MSE RNN	78.14	**0.044**	2.013
+ D	76.74	0.121	**1.331**
L1Sum RNN (*)	**72.85**	0.123	2.406
+ D (*)	79.09	0.079	1.827

Table 4: Table for speech loss ablation

4.6 Qualitative Results

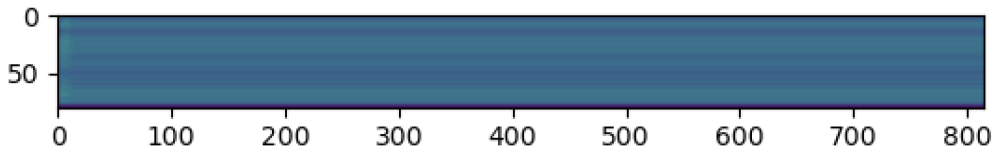
Effect of Discriminator on Latent States To evaluate the latent space of the encoders, we use T-SNE with a perplexity of 10 for a 2D visualization of the proximity of text and speech embeddings from our best RNN encoder on the test set in Figure 2. We notice the nearest neighbors without the discriminator are almost exclusively from the same source, while using the discriminator produces a more mixed visualization, which is what we had anticipated. Thus, it seems the latent representations of speech and text indeed are more aligned through the addition of the adversarial term, which empirically results in better performance.



(a) RNN without discriminator

(b) RNN with discriminator

Figure 2: 2D Latent space visualization without (left) and with (right) discriminator using T-SNE



(a) Mel Spectrogram Output

ðə ˈprɪzənərz wər ˈsteɪtɪd ðət hi wɑz ˈɔlsoʊ ˈsteɪtmənt əv ðə ˈprɛzɪdənt ˈkɛnədi əv ðə ˈprɛzɪdənt ˈkɛnədi ˈpɑrti~

(b) Phoneme Output

Figure 3: Prediction outputs from our model.

Example Outputs We provide some example mel spectrogram and phoneme outputs from our best performing transformer model with discriminator in Figure 3. We find that our speech outputs heavily suffer from the repetition problem and seems to have regressed to "safe" predictions, outputting some varied values for the first few frames but then repeating the same frame until predicting stop. Our text output is slightly better, but still has some repetition, especially near the end of the sequence - a common issue in Seq2Seq. Our speech predictions are essentially nonsense, but some examples can be found on our Google Drive[6].

5 Limitations and Future Work

Given the constrained resources in this project, we were required to both reduce the size of models from prior works and use smaller batches with more gradient accumulation. However, Karita et al. (2019) has shown the affects of larger batches are nontrivial when restricted by these resource constraints. Lack of time also impacted our ability to fully implement better noising functions such as shuffling and time warping as in SpecAugment. We also would have liked to perform more hyperparameter experiments, as these values have shown to critically affect model performance in other works, and Ren et al. (2019) lack thorough documentation and code on the hyperparameters.

Another limitation is we are still relying on a small supervised set of data to learn some structure. While we found unsupervised training with discriminator converged somewhat, it still requires phoneme level transcriptions to converge better. Combining approaches such as Yeh et al. (2018) to learn phonemes/subwords automatically might be one way to scale this approach to acting on raw text. Subword units may enable more distinctions than phonetic transcriptions and improve learning. Regardless, investigating the level at which our text operates as well as the strength of a fully unsupervised approach are important next steps.

Finally, we note there has not been as much work done in decoding algorithms for speech synthesis technologies. Following prior works, we use greedy decoding, and observed a large amount of repetition. Techniques such as nucleus sampling (Holtzman et al., 2020) claim to fix such issues, showing improvements in the text generation domain, and are worth investigating here.

6 Conclusion

In conclusion, we add a discriminator to the dual transformation learning task attempted in Ren et al. (2019). We experiment with both RNN and Transformer architectures, which perform comparably, and different objectives, and discover in all cases the discriminator improves performance. The discriminator also enables somewhat efficient unsupervised learning, resulting in a near 66% improvement in PER for RNNs. Finally, while we did not reach the same level of performance as Ren et al. (2019), we do not believe this is a limitation of our approach but instead our resources, and encourage future work to examine the use of adversarial objectives in dual transformation learning.

[6] https://drive.google.com/drive/folders/1w5FG5QH8F_jZXcCbdDQvTBswX2yMAoNR?usp=sharing

References

Yi-Chen Chen, Chia-Hao Shen, Sung-Feng Huang, and Hung-yi Lee. 2018. Towards unsupervised automatic speech recognition trained by unaligned speech and text only. *CoRR*, abs/1803.10952.

Jan Chorowski, Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio. 2014. End-to-end continuous speech recognition using attention-based recurrent nn: First results.

Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, KyungHyun Cho, and Yoshua Bengio. 2015. Attention-based models for speech recognition. *CoRR*, abs/1506.07503.

Jennifer Drexler and James Glass. 2018. Combining end-to-end and adversarial training for low-resource speech recognition. pages 361–368.

Ewan Dunbar, Julien Karadayi, Mathieu Bernard, Xuan-Nga Cao, Robin Algayres, Lucas Ondel, Laurent Besacier, Sakriani Sakti, and Emmanuel Dupoux. 2020. The zero resource speech challenge 2020: Discovering discrete subword and word units.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging.

Keith Ito and Linda Johnson. 2017. The lj speech dataset. `https://keithito.com/LJ-Speech-Dataset/`.

Yifan Jiang, Shiyu Chang, and Zhangyang Wang. 2021. Transgan: Two transformers can make one strong gan.

B. Juang and Lawrence Rabiner. 2005. Automatic speech recognition - a brief history of the technology development.

Jacob Kahn, Ann Lee, and Awni Hannun. 2019. Self-Training for End-to-End Speech Recognition.

Shigeki Karita, Nanxin Chen, Tomoki Hayashi, Takaaki Hori, Hirofumi Inaguma, Ziyan Jiang, Masao Someki, Nelson Enrique Yalta Soplin, Ryuichi Yamamoto, Xiaofei Wang, Shinji Watanabe, Takenori Yoshimura, and Wangyou Zhang. 2019. A Comparative Study on Transformer vs RNN in Speech Applications.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *CoRR*, abs/1711.00043.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, Ming Liu, and Ming Zhou. 2018. Neural Speech Synthesis with Transformer Network.

Xuancai Li, Chen Kehai, Tiejun Zhao, and Muyun Yang. 2020. End-to-end speech translation with adversarial training. In *Proceedings of the First Workshop on Automatic Simultaneous Translation*, pages 10–14, Seattle, Washington. Association for Computational Linguistics.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. *CoRR*, abs/1508.04025.

Prasanna Muthukumar and Alan Black. 2014. Automatic discovery of a phonetic inventory for unwritten languages for statistical speech synthesis. pages 2594–2598.

Daniel S. Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D. Cubuk, and Quoc V. Le. 2019. Specaugment: A simple data augmentation method for automatic speech recognition. *Interspeech 2019*.

Yi Ren, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. 2019. Almost unsupervised text to speech and automatic speech recognition. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5410–5419. PMLR.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Jonathan Shen, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, Yu Zhang, Yuxuan Wang, RJ Skerry-Ryan, Rif A. Saurous, Yannis Agiomyrgiannakis, and Yonghui Wu. 2017. Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions.

Andros Tjandra, Sakriani Sakti, and Satoshi Nakamura. 2017. Listening while speaking: Speech chain by deep learning. pages 301–308.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, page 1096–1103, New York, NY, USA. Association for Computing Machinery.

Yuxuan Wang, R. J. Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc V. Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. 2017. Tacotron: A fully end-to-end text-to-speech synthesis model. *CoRR*, abs/1703.10135.

Yingce Xia, Di He, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual Learning for Machine Translation.

Chih-Kuan Yeh, Jianshu Chen, Chengzhu Yu, and Dong Yu. 2018. Unsupervised speech recognition via segmental empirical output distribution matching.

Sevinj Yolchuyeva, Géza Németh, and Bálint Gyires-Tóth. 2019. Transformer based grapheme-to-phoneme conversion. *Interspeech 2019*.

Shiyu Zhou, Shuang Xu, and Bo Xu. 2018. Multilingual End-to-End Speech Recognition with A Single Transformer on Low-Resource Languages.

7 Appendix

7.1 Hyperparameters

Parameter	RNN	Transformer
optimizer	AdamW	AdamW
initial learning rate	$2*10^{-3}$	.0625 / 0
weight decay	1e-6	1e-6
batch size	4	4
accum. steps (ae/cm/sp/d)	4/8/4/4	4/8/4/4
epoch steps	50	50
epochs	150	80
encoder RNN	LSTM	–
bidirectional encoder	Yes	Yes
num layers	2	4
num heads	–	4
hidden dim	256	256
ffn dim	–	1024
encoder dropout	.5	.1
decoder dropout	.2	.1
attention type	Luong/LSA	Multihead-self
Learning Rate Schedule	Step (*.1) at 100, 140	Warmup/Linear

Table 5: Network Hyperparameters

Parameter	Text Input	Text Output	Speech Input	Speech Output
architecture	Embedding/Conv	FFN	FFN	Conv
Kernel size	5	–	–	5
Padding	2	–	–	4
input dim	1	256	80	80
hidden dim	256	256	256	256
output dim	256	num phonemes	80	80
nonlinearity	relu	relu	relu	relu
dropout	.5	.1	.5	.1

Table 6: Input/Output Net Hyperparameters

Parameter	Discriminator
encoder RNN	LSTM
bidirectional encoder	Yes
num layers	2
hidden dim	64
fully connected layers	2
dropout	.1

Table 7: Discriminator Hyperparameters

Model	Best Epoch	Total Epochs
Supervised RNN	60	70
Supervised RNN + D	39	70
Supervised Transformer	41	70
Supervised Transformer + D	31	70
Unsupervised RNN	140	145
Unsupervised RNN + D	72	145
Unsupervised Transformer	54	60
Unsupervised Transformer + D	48	48
RNN Luong	141	147
RNN Luong + D	121	133
RNN LSA	127	141
RNN LSA + D	96	122
Transformer Linear	89	96
Transformer Linear + D	82	91
Transformer Inverse	78	82
Transformer Inverse + D	73	79
L1Sum RNN	32	53
L1Sum RNN + D	57	59

Table 8: Best Model Epochs

7.2 Summary of All Runs (Best Model)

	Epoch (B)	Epoch (L)	Params	PER	TTS	ASR	(S) DAE	(T) DAE	(S) DT	(T) DT
RNN (Condor)										
+ Luong	138	155	11.7 M	83.26	0.045	1.931	0.041	0.319	0.045	1.438
+ Luong + D	130	133	12.1 M	91.31	0.096	1.3	0.09	1.292	0.096	1.301
+ LSA	127	141	11.7 M	78.14	0.044	2.013	0.042	0.198	0.044	1.574
+ LSA + D	96	122	12.1 M	76.74	0.121	1.331	0.075	1.305	0.119	1.328
Transformer (Condor)										
+ Linear	89	96	17.1 M	85.12	0.118	1.491	0.115	1.465	0.118	1.491
+ Linear + D	82	91	17.3 M	75.98	0.099	1.42	0.099	1.426	0.099	1.42
+ Inverse	86	96	17.1 M	70.85	0.117	1.926	0.118	1.507	0.117	1.926
+ Inverse + D	84	86	17.3 M	86.79	0.116	1.403	0.119	1.4	0.116	1.403
RNN (TACC)										
+ Luong	141	162	11.7 M	86.65	0.044	1.963	0.044	0.295	0.044	2.155
+ Luong + D	121	141	12.1 M	81.38	0.129	1.309	0.089	1.286	0.131	1.312
+ LSA	160	202	11.7 M	82.65	0.057	2.09	0.046	0.207	0.058	2.432
+ LSA + D	182	195	12.1 M	102.93	0.116	1.287	0.128	1.271	0.118	1.284
Transformer (TACC, 2k WM)										
+ Linear	74	79	17.1 M	90.57	0.119	1.558	0.118	1.456	0.119	1.558
+ Linear + D	77	78	17.3 M	87.19	0.087	1.435	0.088	1.432	0.087	1.435
+ Inverse	78	82	17.1 M	75.39	0.034	1.582	0.035	1.438	0.034	1.582
+ Inverse + D	73	79	17.3 M	72.62	0.087	1.426	0.087	1.418	0.087	1.426
Transformer (TACC, 4k WM)										
+ Linear	69	88	17.1 M	81.03	0.061	1.637	0.061	1.508	0.061	1.504
+ Linear + D	66	86	17.3 M	82.21	0.019	1.477	0.019	1.467	0.019	1.477
+ Inverse	84	89	17.1 M	85.53	0.022	3.175	0.014	1.463	0.022	1.701
+ Inverse + D	86	87	17.3 M	79.78	0.112	1.462	0.108	1.456	0.112	1.462
Supervised RNN	60	70	11.7 M	82.89	0.041	6.381	0.065	7.166	0.051	6.594
+ D	39	70	12.1 M	73.25	0.079	4.887	0.082	5.094	0.079	4.887
Supervised Transformer	41	70	17.1 M	74.33	0.102	4.985	0.119	5.749	0.102	4.989
+ D	31	70	17.3 M	74.05	0.120	5.342	0.097	5.469	0.120	5.342
Unsupervised RNN	140	145	11.7 M	145.87	0.069	1.344	0.053	0.209	0.062	1.267
+ D	72	145	12.1 M	74.96	0.106	1.307	0.103	1.281	0.106	1.310
Unsupervised Transformer	54	60	17.1 M	99.17	0.122	1.512	0.121	1.503	0.122	1.512
+ D	48	48	17.3 M	72.54	0.113	1.636	0.113	1.651	0.113	1.636
L1Sum RNN	32	53	11.7 M	72.85	0.123	2.406	0.126	2.404	0.124	2.410
+ D	57	59	12.1 M	79.09	0.079	1.827	0.083	1.806	0.080	1.835

Table 9: All Best Model Results. Epoch (B) correspond to the amount of epoch trained for the model with the best performance. Epoch (L) correspond to the maximum number of epochs the models have been trained. WM corresponds to the number of warmup steps in the learning rate scheduler. The discriminator usually achieved about 70-100% accuracy in distinguishing between the speech and text representations, so there are potential for more improvement.