Learning Robust Latent Representations for Controllable Speech Synthesis

Shakti Kumar[*†] **Jithin Pradeep**[*] **Hussain Zaidi**[*]

[*]Vanguard Center for Analytics and Insights
[†]Computer Science, University of Toronto

{shakti_kumar, jithin_pradeep, hussain_zaidi}@vanguard.com

Abstract

State-of-the-art Variational Auto-Encoders (VAEs) for learning disentangled latent representations give impressive results in discovering features like pitch, pause duration, and accent in speech data, leading to highly controllable text-to-speech (TTS) synthesis. However, these LSTM-based VAEs fail to learn latent clusters of speaker attributes when trained on limited or noisy datasets. Further, different latent variables are found to encode the same features, limiting the control and expressiveness during speech synthesis. To resolve these issues, we propose REMMI (Reordered transformer Encoder with Minimal Mutual Information) where we minimize the mutual information between different latent variables and devise a modified Transformer architecture with layer reordering to learn controllable latent representations in speech data. We show that REMMI reduces the cluster overlap of speaker attributes by at least 30% over LSTM-VAE.

1 Introduction

Learning disentangled latent representations in speech is an active area of research (Hsu et al., 2017; Chou et al., 2018; Park et al., 2020) with applications in controlling the style (for example, pitch, pause duration, and accent) of synthesized speech. Recurrent architectures like Long Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) networks in Variational Autoencoders (VAE) have been state-of-the-art in discovering disentangled latent representations in speech (Wang et al., 2018; Jia et al., 2018; Skerry-Ryan et al., 2018) as well as sequential data more generally. For example Li and Mandt (2018) attempt to disentangle global and local features of video/speech in different latent variables. Hsu et al. (2019) disentangled different dimensions of the latent variables to discover meaningful representations and hence proposed a speech synthesis model with controllable pitch, pause duration, and speed.

These papers as well as several others (Chung et al., 2015; Hsu et al., 2019; Leglaive et al., 2020; Hono et al., 2020; Sun et al., 2020) make one limiting assumption— the availability of hundreds of hours of speech data for training deep learning networks. As we show in our experiments, state-of-the-art VAEs fail to learn meaningful separation of speaking styles in speech data when presented with small datasets. In addition, different latent variables learned by the VAE are no longer uncorrelated. Both these shortcomings lead to poor control of speaking styles during synthesis.

While LSTMs are state-of-the-art in learning latent variables in speech, Transformers have been used for understanding latent representations for text completion (Wang and Wan, 2019) and Transformer-based VAEs were used in Jiang et al. (2020) to model independent style attributes in music generation.

Inspired by these limitations of LSTM-based VAEs and the promise of more "attentive" networks, we modify the loss function of the state-of-the-art VAEs (Hsu et al., 2019) by explicitly minimizing the mutual information between latent variables, thereby penalizing common learned features between different representations. We then modify Transformer architecture for learning robust disentangled latent representations of speech from limited and noisy data. We show that our proposed architecture– REMMI (Reordered transformer Encoder with Minimal Mutual Information) discovers compact stable latent representations of speaker attributes even on datasets as small as 4 hours of total speech samples while state-of-the-art fails. Our proposed VAE outperforms LSTM and vanilla Transformers even on challenging dataset like Common Voice which has considerable background noise, low recording quality and large num-

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 3562–3575
August 1–6, 2021. ©2021 Association for Computational Linguistics

ber of speakers with the same style or accent. To summarize, following are the main contributions of our work,

1. Formulate a modified VAE loss function for speech data and a novel Transformer-based VAE for learning uncorrelated latent variables, thereby allowing more precise control over synthesis compared to the existing state-of-the-art.

2. Show that our latent clusters of speaking styles are better separated than existing LSTM and vanilla Transformer based VAEs on noisy and small datasets.

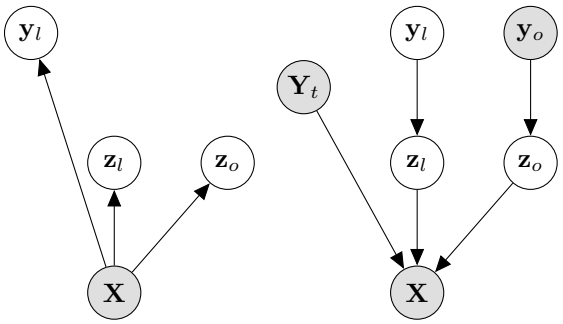
3. Show that the our modified Transformer architecture allows a faster convergence of the variational lower bound compared to both vanilla Transformer and LSTM based VAEs.

2 Related Work

Multiple previous work have targeted this problem of learning latent representations for sequential data like speech (Wang et al., 2018; Jia et al., 2018; Skerry-Ryan et al., 2018). As discussed, the main advantage of learning such representations is that it allows creating diverse examples during reconstruction by manipulating the encoded latent variable. Li and Mandt (2018) propose two sets of latents which learn global features like the generated sequence contents and local dynamic features such as pitch, speed etc. However, a limitation of this approach is the lack of interpretability of the learnt dimensions— it is known that the different dimensions of the latent variables are learning some features but there is little to no visibility into what those actual features are.

Modifying Text-to-Speech systems by introducing additional encoders has been a standard way to discover meaningful representations. Zhang et al. (2019) build on top of Tacotron-2 (Shen et al., 2018) architecture and use Gaussians to model their latent variables. An improved version can be seen in Hsu et al. (2019) where a hierarchical latent with mixture of Gaussians is used. Hsu et al. (2019) propose adversarial training to further improve latent variables and the features discovered by disentangling the background noise and reverberation along with speaker identity from the recording conditions.

While all these prior work aim to discover latent representations, there is a lot of room for improving those representations especially in cases where we have very limited hours of speech dataset. As we show in our experiments, in the absence of explicit restrictions on the training objective these VAEs easily collapse when presented with smaller datasets. Thus we focus on improving the representations, specifically latent clusters of speaker attributes, in cases of extremely limited datasets. Our contributions, however are not limited to smaller datasets and we see similar improved performance on larger and noisy datasets too.



(a) Encoder (b) Generator

Figure 1: Graphical model of controllable TTS system. Note that $q(\mathbf{y}_l|\mathbf{X})$ in the Encoder can be approximated in terms of $q(\mathbf{z}_l|\mathbf{X})$, in which case node $\mathbf{y}_l$ will have an edge from $\mathbf{z}_l$ instead of $\mathbf{X}$ as done in Hsu et al. (2019).

3 Background

Controllable text-to-speech (TTS) VAE-based systems like in Hsu et al. (2019) take an input text sequence $\mathbf{Y}_t$ and an optional observed categorical label $\mathbf{y}_o$ (e.g., speaker identity or accent) as input and learn to synthesize a sequence, usually mel-spectrogram frames $\mathbf{X}$ as output. Additional latent variables $\mathbf{z}_o$ and $\mathbf{z}_l$ can be introduced to discover meaningful representations during this process. Here $\mathbf{z}_o$ is a continuous latent learnt on top of shown labels $\mathbf{y}_o$, hence $\mathbf{z}_o$ captures the variation in features correlated with the speaker attribute $\mathbf{y}_o$. $\mathbf{z}_l$ is a completely unsupervised continuous variable learnt on top of standard Expectation-Maximization style latent mixture components $\mathbf{y}_l$. This graphical model is depicted in Figure 1. The objective function for learning such model, i.e. synthesizing sequence $\mathbf{X}$ given $\mathbf{Y}_t$ and $\mathbf{y}_o$, can be formulated as the variational lower bound[1],

[1]Complete derivation is given in the Appendix A.

$$
\begin{aligned}
log\ p(\mathbf{X}|\mathbf{Y}_t, \mathbf{y}_o) \geq\ & log\ p(\mathbf{X}|\mathbf{Y}_t, \widetilde{\mathbf{z}_o}, \widetilde{\mathbf{z}_l}) \\
& - \sum_{\mathbf{y}_l=1}^{K} q(\mathbf{y}_l|\mathbf{X}) D_{KL}[\ q(\mathbf{z}_l|\mathbf{X})\ ||\ p(\mathbf{z}_l|\mathbf{y}_l)\ ] \\
& - D_{KL}[\ q(\mathbf{y}_l|\mathbf{X})\ ||\ p(\mathbf{y}_l)\ ] \\
& - D_{KL}[q(\mathbf{z}_o|\mathbf{X})\ ||\ p(\mathbf{z}_o|\mathbf{y}_o)\ ] \\
& = -L_{mel} - L_{KL}
\end{aligned}
$$

where $L_{mel} = -log\ p(\mathbf{X}|\mathbf{Y}_t, \widetilde{\mathbf{z}_o}, \widetilde{\mathbf{z}_l})$ and L_{KL} refers to the remaining terms. Here $\widetilde{\mathbf{z}_o}, \widetilde{\mathbf{z}_l}$ are sampled points and are reparameterized (Kingma and Welling, 2014) as $\widetilde{\mathbf{z}_o} = \hat{\mu_o} + \hat{\sigma_o} \odot \epsilon_o$ and $\widetilde{\mathbf{z}_l} = \hat{\mu_l} + \hat{\sigma_l} \odot \epsilon_l$ with $\hat{\mu_o}, \hat{\mu_l}, \hat{\sigma_o}, \hat{\sigma_l}$ as the mean and standard deviation of the posterior distributions $q(\mathbf{z}_o|\mathbf{X})$ and $q(\mathbf{z}_l|\mathbf{X})$ respectively and with auxiliary noise variable $\epsilon_o, \epsilon_l \sim \mathcal{N}(0, I)$. Following Higgins et al. (2017) the loss L can be written in a more general form as,

$$
L = L_{mel} + \beta L_{KL} \tag{1}
$$

with β balancing the relative weighing between the latent channels and reconstruction accuracy. Here L_{mel} is the mel loss which controls the quality of the mel-spectrograms produced and L_{KL} refers to the total KL Loss controlling the features learnt in latent variables.

This VAE can be used in the Tacotron-2 architecture (Hsu et al., 2019) as shown in Figure 2(a) to learn the text to mel-spectrogram mapping and the latent features controlled by L_{KL}.

4 Methodology

We now describe the two main components, 1) Minimizing mutual information and 2) Layer reordering in our proposed REMMI architecture.

4.1 Minimizing Mutual Information

The latent $\mathbf{z}_l$ in Figure 1 is unsupervised while the latent $\mathbf{z}_o$ learns features correlated with the shown label $\mathbf{y}_o$. Our experiments showed that both $\mathbf{z}_l, \mathbf{z}_o$ can end up encoding the same set of features, which leads to poor control in synthesizing speech. An intuition into why this happens lies in the fact that $\mathbf{z}_l$ is an unsupervised variable and it can discover any feature hidden in the input speech sequence. There is no term in the loss function (1) which prevents the features of $\mathbf{z}_l$ from being correlated with the observed labels $\mathbf{y}_o$ (Klys et al., 2018).

This can be resolved by minimizing the mutual information I between latents $\mathbf{z}_o$ (equivalently $\mathbf{y}_o$) and $\mathbf{z}_l$. We can formulate this as,

$$
\begin{aligned}
& min\ I(\mathbf{y}_o; \mathbf{z}_l) \triangleq max\ H(\mathbf{y}_o|\mathbf{z}_l) \\
& = min \int_{\mathbf{z}_l} \int_{\mathbf{y}_o} p(\mathbf{z}_l)\ p(\mathbf{y}_o|\mathbf{z}_l)\ log\ p(\mathbf{y}_o|\mathbf{z}_l) d\mathbf{y}_o d\mathbf{z}_l \\
& = min \int_{\mathbf{X}} \int_{\mathbf{z}_l} \int_{\mathbf{y}_o} p(\mathbf{X})\ p(\mathbf{z}_l|\mathbf{X})\ p(\mathbf{y}_o|\mathbf{z}_l) \\
& \qquad\qquad log\ p(\mathbf{y}_o|\mathbf{z}_l)\ d\mathbf{y}_o\ d\mathbf{z}_l\ d\mathbf{X}
\end{aligned}
$$

Since integral over $\mathbf{z}_l$ is intractable, we replace $p(\mathbf{z}_l|\mathbf{X})$ with an approximate posterior $q(\mathbf{z}_l|\mathbf{X})$. Further, since the true distribution $p(\mathbf{y}_o|\mathbf{z}_l)$ is unknown, we approximate it by introducing a new network $q_\psi(\mathbf{y}_o|\mathbf{z}_l)$ leading to $min\ I(\mathbf{y}_o; \mathbf{z}_l)$

$$
\begin{aligned}
& \approx min \int_{\mathbf{X}} \int_{\mathbf{z}_l} \int_{\mathbf{y}_o} p(\mathbf{X})\ q(\mathbf{z}_l|\mathbf{X})\ q_\psi(\mathbf{y}_o|\mathbf{z}_l) \\
& \qquad\qquad log\ q_\psi(\mathbf{y}_o|\mathbf{z}_l) d\mathbf{y}_o\ d\mathbf{z}_l\ d\mathbf{X} \\
& = min\ E_{D(\mathbf{X})q(\mathbf{z}_l|\mathbf{X})} \left[\int_{\mathbf{y}_o} q_\psi(\mathbf{y}_o|\mathbf{z}_l) \right. \\
& \qquad\qquad \left. log\ q_\psi(\mathbf{y}_o|\mathbf{z}_l)\ d\mathbf{y}_o \right] \\
& \approx min\ \frac{1}{N} \sum_{a} \left[q_\psi(\mathbf{y}_o = a|\mathbf{z}_l{}') \right. \\
& \qquad\qquad \left. log\ q_\psi(\mathbf{y}_o = a|\mathbf{z}_l{}') \right]
\end{aligned} \tag{2}
$$

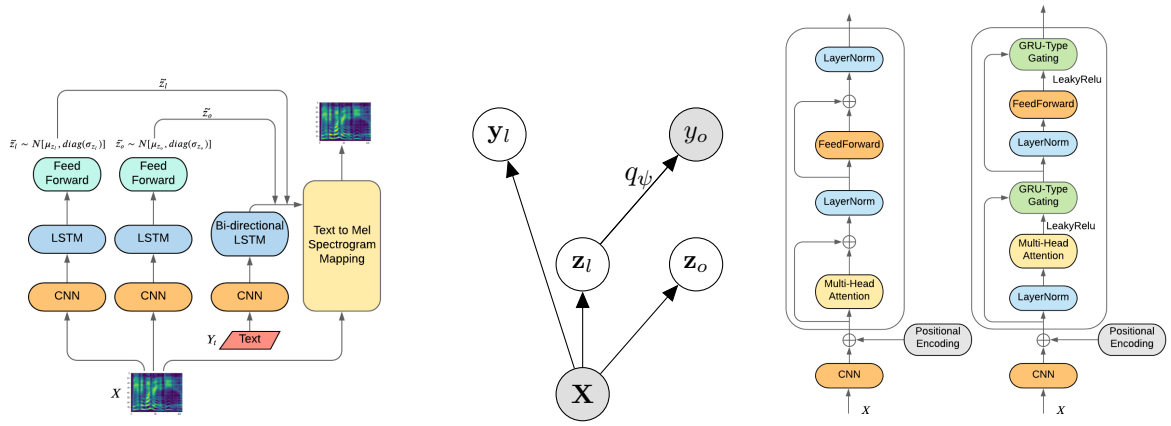
where $\mathbf{z}_l{}' \sim q(\mathbf{z}_l|\mathbf{X})$, $a \in \{0, 1, 2...A\}$, A is total number of unique classes of $\mathbf{y}_o$, N is the number of samples used for Monte Carlo estimates, and $D(\mathbf{X})$ is the underlying distribution of the input points $\mathbf{X}$. Our proposed encoder is depicted in Figure 2(b). Since we are using q_ψ to make predictions for $\mathbf{y}_o$, this network needs to be learnt itself. Hence we need to subtract an additional $q_\psi(\mathbf{y}_{o_T}|\mathbf{z}_l{}')$ from the loss function, where $\mathbf{y}_{o_T}$ is the ground truth $\mathbf{y}_o$ for the input $\mathbf{X}$. With $N = 1$ our proposed term is,

$$
\begin{aligned}
L_{MI} = & \sum_{a} q_\psi(\mathbf{y}_o = a|\mathbf{z}_l{}') log\ q_\psi(\mathbf{y}_o = a|\mathbf{z}_l{}') \\
& - q_\psi(\mathbf{y}_{o_T}|\mathbf{z}_l{}')
\end{aligned} \tag{3}
$$

Combining equations (1) and (3), the total loss function in our proposed model is,

$$
\begin{aligned}
L_{total} &= L_{mel} + \beta L_{KL} + \gamma L_{MI} \\
&= L_{mel} + L_{cond}
\end{aligned} \tag{4}
$$

To summarize, L_{mel} controls the quality of the mel-spectrogram produced during decoding, L_{KL} controls the features learnt in the latent variables $\mathbf{z}_l, \mathbf{z}_o$ and L_{MI} makes sure that $\mathbf{z}_l, \mathbf{z}_o$ encode different features. We will be referring to L_{mel} as the reconstruction or mel loss, L_{KL} as the KL loss and $L_{cond} = \beta L_{KL} + \gamma L_{MI}$ as the conditional loss respectively throughout this paper.



(a) Tacotron Architecture (b) Proposed Encoder (c) Original versus Proposed Transformer

Figure 2: **Left**: The Tacotron-2 architecture. VAE consists of two left parts where LSTMs predict mean μ and variance σ^2 of multivariate Gaussians $\mathcal{N}(\mu_{\mathbf{z}_l}, diag(\sigma^2_{\mathbf{z}_l})), \mathcal{N}(\mu_{\mathbf{z}_o}, diag(\sigma^2_{\mathbf{z}_o}))$. $\widetilde{\mathbf{z}}_l, \widetilde{\mathbf{z}}_o$ from this distribution are sampled and concatenated to the text encoding to conditionally learn the text to mel-spectrogram mapping. **Center**: Proposed encoder with the network q_ψ. The generator stays the same as in Figure 1. **Right**: The original and the proposed Transformers replace the LSTMs shown in the VAE of Tacotron-2 architecture.

4.2 Layer Reordering in Transformer

Introducing the above loss helps disentangle the learning of $\mathbf{z}_o$ and $\mathbf{z}_l$, but there is another problem that remains. Our experiments on MAILABS and Common Voice data, discussed in section 5.3, indicated that clusters of $\mathbf{z}_o$ corresponding to different shown labels $\mathbf{y}_o$ start sharing regions in the latent space. Hence for any given label $\mathbf{y}_o$ the sampled $\hat{\mathbf{z}_o} \sim p(\mathbf{z}_o|\mathbf{y}_o)$ may or may not belong to the style which $\mathbf{y}_o$ denotes. This leads to speech samples where the style correlated with the shown attribute $\mathbf{y}_o$ is not under control while sampling from the priors.

We tackle this problem by replacing LSTMs with Transformers. We expected that the ability of Transformers to attend to specific frames of interest where features could be localized or have a higher expression density, with a higher weight in the input speech sequence should bring down the dataset volume required for convergence by a considerable amount. Hence the lower bound on dataset size needed for modelling non overlapping clusters of $\mathbf{z}_o$ should be smaller while still keeping the sampled style under control. This should also accelerate the separation between latent clusters for larger datasets. Our experiments with vanilla Transformer-based VAEs confirm our predictions.

We next drew some inspiration from Parisotto et al. (2019) and modified the Transformer encoder. This was an attempt at changing the learning paradigm— instead of directly learning to translate $\mathbf{Y}_t$ to $\mathbf{X}$ in different $\mathbf{y}_o$ styles, we first learn to synthesize a general representation for all $\mathbf{X}$, and then learn specific deviations of each style $\mathbf{y}_o$ from this general representation. For example, instead of learning directly to speak in different accents first we learn to speak, and then we learn the subtleties of different accents. Our hypothesis was that learning different $\mathbf{y}_o$ styles should be a lot faster if a common understanding of all $\mathbf{X}$ in the dataset is gained first. The accent specific speech frames $\mathbf{X}$ (or style specific as per $\mathbf{y}_o$) should just be a slight deviation from this common representation.

Our proposed architecture is shown in Figure 2c where we switch the order of `LayerNorm` forming a direct connection between the input and the output. Due to this layer reordering if we make sure that all the modules `MHA`, `LayerNorm`, `FeedForward` are initialized with their expectation near 0, a direct path is formed early in training allowing a general representation of speech to be learnt independent of the shown labels $\mathbf{y}_o$. Now as training progresses and these modules warm up, the accent or $\mathbf{y}_o$ specific features will be learnt by conditioning the encoder.

We also introduce GRU-type gating (Chung et al., December 2014) to stabilize learning by minimizing the maximum gradient norms produced, and apply a small nonlinearity via $LeakyRelu$ at the outputs of the `MHA` and `FeedForward` modules to balance the observed trade-off between frequent gradient updates and maximum gradient

d	Feature	$\mu_{\mathbf{z}_l,d} - 3\sigma_{\mathbf{z}_l,d}$	$\mu_{\mathbf{z}_l,d}$	$\mu_{\mathbf{z}_l,d} + 3\sigma_{\mathbf{z}_l,d}$
0	Speaking Rate (sec)	3.0 ± 0.2	3.7 ± 0.3	4.4 ± 0.3
1	F_0 (Hz)	240.5 ± 12.57	211.4 ± 15.66	184 ± 10.43
2	Pause Duration (msec)	70 ± 3.40	79 ± 3.30	91 ± 3.50

Table 1: Length of the mel-spectrogram synthesized and pause durations increase while pitch decreases with increasing dth dimension of $\mathbf{z}_l$ from its marginal prior mean in REMMI.

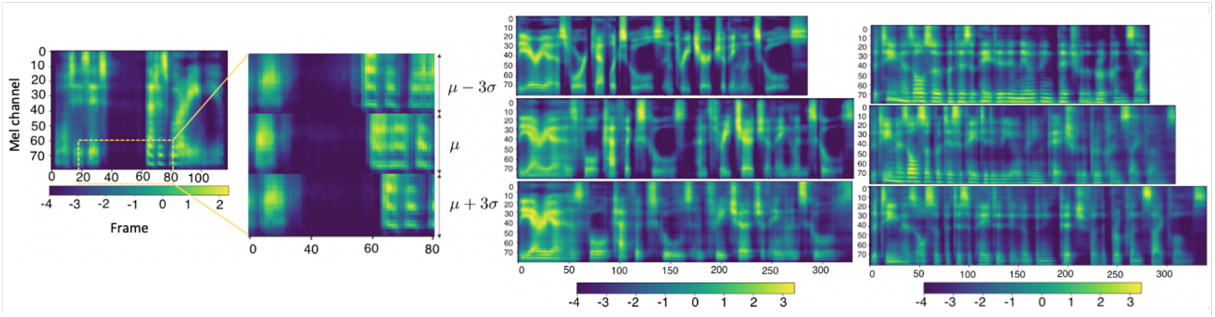


Figure 3: **Left**: Synthesized mel spectrogram for *"What is it, that is worrying you today?"* The stack of 3 mel spectrograms on the right are zoomed areas from frames 20 to 80 for each of their original mel-spectrogram. It can be seen that the pause duration denoted by the dark region increases as you synthesize the same text moving from $\mu_i - 3\sigma_i$ to $\mu_i + 3\sigma_i$. **Center**: Three mel-spectrograms synthesized for the text *"The area has four catholic schools and three church of England schools"*, corresponding to three random sampling of $\widetilde{\mathbf{z}}_o, \widetilde{\mathbf{z}}_l$ from their posteriors. First synthesis is considerably shorter than the second and third. Notice the different positions of voids between frames 50 and 100, and at frame 150 in the third spectrogram being considerably different. **Right**: Mel-spectrograms synthesized for the text *"The team has also participated in the opening pitch of the Brooklyn Cyclones"*. The third spectrogram shows smooth areas in the higher mel channels compared to the second and the first. These random latent sampling affects intonation and spectrogram texture.

norm[2].

5 Experiments

We refer to our proposed VAE with modifications from sections 4.1 (L_{MI} term) and 4.2 as REMMI, the vanilla Transformer with L_{MI} term as Transformer-VAE and the LSTM based state-of-the-art Tacotron-2 without L_{MI} term (Hsu et al., 2019) as LSTM-VAE. We trained each model on two datasets— 1) MAILABS (Solak, 2018 (accessed November 11, 2020) with a total 35hrs of UK and 39hrs of US speech in studio quality recorded by 4 professional speakers, 2) Common Voice (Ardila et al., 2020) with 4hrs of UK and 19hrs of US speech crowd-sourced from 477 volunteers with varying background noise, microphone qualities and other recording conditions. The input feature $\mathbf{X}$ were mel-scale spectrograms, the label $\mathbf{y}_o$ was set to be 0 for all $\mathbf{X}$ belonging to US and 1 for all UK. Dimension of $\mathbf{z}_o$ and $\mathbf{z}_l$ were picked to be 2 and 3 respectively and $K = 3$ for all experiments [3].

5.1 Features Learnt

Before we demonstrate our latent cluster improvements over Transformer-VAE and LSTM-VAE, we show that REMMI does learn important latent features in speech. Our experiments (focused on learning the speaking rate, the fundamental frequency F_0, and the pause duration) are summarized in Table 1. $\mu_{\mathbf{z}_l,d}$ and $\sigma_{\mathbf{z}_l,d}$ are the dth dimension mean and standard deviations of the marginal prior $p(\mathbf{z}_l) = \sum_k p(\mathbf{z}_l|\mathbf{y}_l = k)p(\mathbf{y}_l = k)$. All other dimensions of $\mathbf{z}_l$ are kept fixed at their own marginal priors while analyzing dth dimension.

For demonstrating control on speaking rate, we did 25 different synthesis for the text *"We had been wandering, indeed, in the leafless shrubbery an hour in the morning"*. It can be seen from Table 1 that the length of the synthesized mel-spectrogram increases as the value of $\mathbf{z}_l$ dimension 0 increases.

Next, we synthesized 25 texts, with 10 samples for each text to show control on pause duration and

[2]Importance of Gates and the specific choice of $LeakyRelu$ is discussed in the Ablation Study in Appendix D.

[3]Other hyperparameters of our VAE and training details of Tacotron-2 are given in Appendix F, G, H.

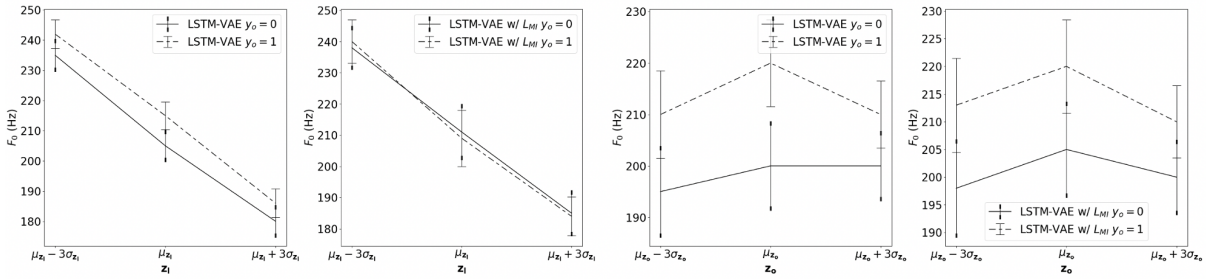


Figure 4: In LSTM-VAE F_0 encoded by $\mathbf{z}_l$ is significantly different for $\mathbf{y}_o = 0, 1$ showing that y_o specific information is encoded by $\mathbf{z}_l$. However this difference is no longer significant once we include our proposed L_{MI} terms in LSTM-VAE w/ L_{MI} experiment. $\mathbf{z}_o$ keeps showing different values of F_0 for $y_o = 0, 1$ in both LSTM-VAE and LSTM-VAE w/ L_{MI} experiments demonstrating learnt features which are conditional on y_o.

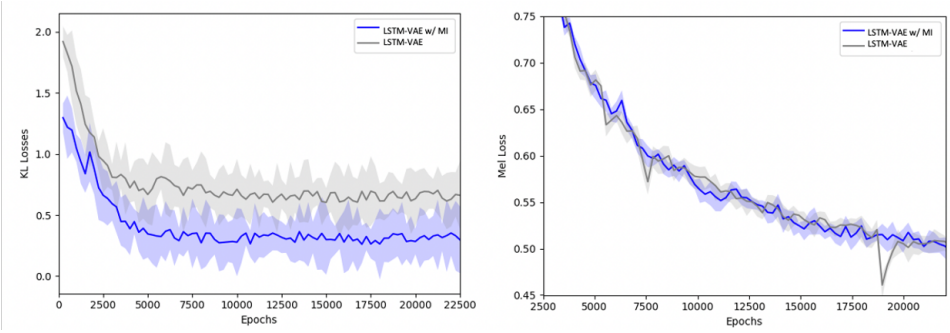


Figure 5: **Left**: Test L_{KL} versus epochs. Including L_{MI} in loss function decreases L_{KL} pointing to improved latent variables. **Right**: Test L_{mel} versus epoch. The L_{mel} remains the same even upon including L_{MI} demonstrating our proposed L_{MI} does not hurt the synthesized mel-spectrogram quality.

pitch (or the fundamental frequency F_0). For pause duration experiments each text contained at least one comma and we measured the maximum period of intermediate silence for each synthesis. To calculate F_0 we used the YIN algorithm (Guyot, 2018). In Table 1 it can be seen that the pause duration increases and F_0 decreases with increasing values of 2nd and 1st dimensions of $\mathbf{z}_l$, respectively.

Furthermore the sampled variables $\widetilde{\mathbf{z}_o}, \widetilde{\mathbf{z}_l}$ from their respective posterior distributions $q(\mathbf{z}_o|\mathbf{X}), q(\mathbf{z}_l|\mathbf{X})$ in L_{mel} gives the effect of different intonations with different speakers every time we synthesize a given text $\mathbf{Y}_t$. We demonstrate concrete examples in Figure 3.

5.2 Importance of L_{MI}

Our experiment on MAILABS dataset shows that the latent variable $\mathbf{z}_l$ starts encoding $\mathbf{y}_o$ specific features in the absence of an explicit L_{MI} term in the total loss, contrary to the expectation that $\mathbf{z}_l$ should not encode any $\mathbf{y}_o$ style specific information. As shown in Figure 4, $\mathbf{z}_l$ shows different values of F_0 for classes $\mathbf{y}_o = 0, 1$ in the absence of L_{MI}, while $\mathbf{z}_o$ continues to show accent specific values for both $\mathbf{y}_o$ classes with and without L_{MI} terms. The values in Figure 4 are plotted for a synthesis of 25 different texts with 10 samples for each text. We show similar trends for speaking rate in the Appendix.

A consequence of including L_{MI} in the loss function (4) can also be seen in the test curve of L_{KL}. We can see in Figure 5 that LSTM-VAE w/ MI has a lower value of L_{KL}. Also note that as shown in Figure 5, L_{mel} remains the same in both the experiments hence there is an overall decrease in the total loss value. We also observe that the two terms of L_{MI} in equation (3) are in contention to each other. The first term tries to learn a representation $\mathbf{z}_l$ such that it does not have any information about label $\mathbf{y}_o$ whereas the second term tries to maximize the probability of predicting true label $\mathbf{y}_o$ given $\mathbf{z}_l$. We verify from our experiments that at convergence $\mathbf{z}_l$ acts as a complete random input for estimating $\mathbf{y}_o$ with $q_\psi(\mathbf{y}_o|\mathbf{z}_l) = 0.5$ for both $\mathbf{y}_o = 0, 1$.

Model	4hrs US+4hrs UK		20hrs US+20hrs UK		39hrs US+35hrs UK	
	DI	DBI	DI	DBI	DI	DBI
LSTM-VAE	0.55±0.15	2.11±0.24	1.41±0.21	1.60±0.29	2.10±0.29	1.12±0.24
Transformer-VAE	1.22±0.26	0.44±0.05	2.24±0.05	0.30±0.15	2.48±0.23	0.27±0.09
REMMI	**1.85±0.59**	**0.35±0.07**	**2.33±0.21**	**0.29±0.10**	**2.80±0.26**	**0.26±0.07**

Table 2: REMMI consistently increases DI and reduces DBI for different sizes of MAILABS dataset and performs at least 3% better (DBI for 20hrs US+20hrs UK) on MAILABS dataset compared to all existing architectures.

Model	4hrs US+4hrs UK		10hrs US+4hrs UK		19hrs US+4hrs UK	
	DI	DBI	DI	DBI	DI	DBI
LSTM-VAE	0.98±0.17	83.18±13.66	0.85±0.23	85.53±15.10	0.80±0.30	98.20±24.68
Transformer-VAE	0.99±0.15	0.19±0.01	0.98±0.22	0.18±0.18	0.94±0.29	0.17±0.30
REMMI	**1.03±0.40**	**0.15±0.005**	**0.99±0.20**	**0.16±0.04**	**0.99±0.25**	**0.16±0.05**

Table 3: REMMI performs at least 4% better (DI for 4hrs US+4hrs UK Common Voice compared to Transformer-VAE) on all sizes of noisy Common Voice dataset than all existing LSTM and Transformer-VAE architectures.

Model	Overlap on MAILABS			Overlap on Common Voice		
	4+4	20+20	39+35	4+4	10+4	19+4
LSTM-VAE	30%	11%	0%	92%	94%	96%
Transformer-VAE	7%	0%	0%	52%	65%	81%
REMMI	**0%**	0%	0%	**47%**	**56%**	**65%**

Table 4: Overlap percentages for datasets of size $M + N$ with M hrs US and N hrs UK speech. REMMI reduces the overlap percentage by 30% for limited MAILABS dataset and by half for limited Common Voice dataset. The reduction difference for entire Common Voice dataset is 31% compared to LSTM and 16% compared to Transformer-VAE.

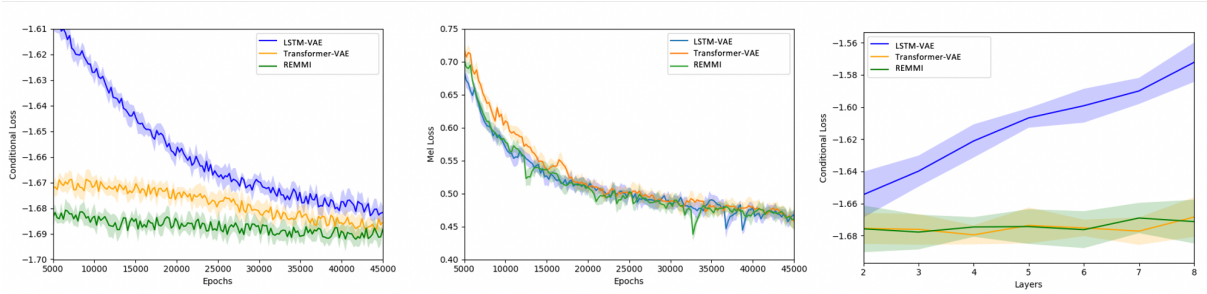


Figure 6: Loss Curves on MAILABS dataset. **Left**: Test L_{cond} versus Epochs. REMMI converges faster compared to both Transformer-VAE and LSTM-VAE. **Center**: Test L_{mel} versus Epochs. REMMI accelerates L_{cond} without compromising the mel-spectrogram quality or L_{mel}. **Right**: Test L_{cond} versus model depth. Transformer and REMMI do not overfit to a given dataset with increasing model depth unlike LSTM-VAE.

5.3 Cluster Quality

As discussed in section 4.2, we want clusters of $p(\mathbf{z}_o|\mathbf{y}_o = 0)$ and $p(\mathbf{z}_o|\mathbf{y}_o = 1)$ to be far from each other with no overlaps so that we can control $\mathbf{y}_o$ styles during synthesis. Hence we objectively measured the cluster quality with Dunn Index (DI) (Bezdek and Pal, 1995) and DB Index (DBI) (Davies and Bouldin, 1979) where $DI = \frac{min_{1\leq i<j\leq n} d(i,j)}{max_{1\leq k\leq n} d'(k)}$, $DBI = \frac{1}{n}\sum_{i=1}^{n} max_{j\neq i}\left(\frac{\sigma_i+\sigma_j}{d(\mu_i,\mu_j)}\right)$, j, i are cluster indices, $d(i, j)$ denotes the distance between the clusters i and j, n is the total number of points, $d'(k)$ is the maximal intra-cluster distance and $\mu_i, \sigma_i, \mu_j, \sigma_j$ are the means and standard deviations of the clusters i, j respectively. Thus DI is the ratio of minimal inter-cluster distance to the maximal intra-cluster distance. Similarly, DBI is the ratio of spread in each cluster to the distance between their means.

In Tables 2 and 3, we compare the test DI and DBI for different dataset sizes between

Dataset	CMOS	CS
MAILABS4hrs US+4hrs UK	0.2581 +- 0.1249	0.576 +- 0.0947
CommonVoice 4hrs US+4hrs UK	0.0541 +- 0.0966	0.327 +- 0.0888

Table 5: Positive CMOS confirms that REMMI produces speech that sounds more British than LSTM-VAE. A higher CS also shows that REMMI has better control over synthesized accent than LSTM-VAE.

REMMI, Transformer-VAE and LSTM-VAE. We see that REMMI performs consistently better than Transformer-VAE and LSTM-VAE for both MAILABS and Common Voice dataset. We also observe that as dataset size decreases, the performance gap between our REMMI and LSTM-VAE increases.

In Table 4 we calculate the percentage of overlap between clusters with test points $\hat{\mathbf{z}_o} \sim p(\mathbf{z}_o|\mathbf{y}_o = i)$ marked as overlapping with cluster $p(\mathbf{z}_o|\mathbf{y}_o = j)$ if they fall within $[\mu_{p(\mathbf{z}_o|\mathbf{y}_o=j)} - \sigma_{p(\mathbf{z}_o|\mathbf{y}_o=j)}, \mu_{p(\mathbf{z}_o|\mathbf{y}_o=j)} + \sigma_{p(\mathbf{z}_o|\mathbf{y}_o=j)}]$, with $i, j = 0, 1$. We observe that our REMMI consistently decreases the overlap regions by large margins even on challenging datasets like Common Voice, where more than 90% overlap exists for existing state-of-the-art. As discussed earlier this better separation provides improved control on synthesis and prevents uncontrolled styles when sampling speech from the priors.

5.4 Synthesis Quality

To get opinion scores on the quality of the synthesized British accent between LSTM-VAE and REMMI, we used Griffin-Lim reconstruction (Griffin and Lim, 1983) to convert the Mel spectrograms to waveforms for models trained on Common Voice 4hrs US+4hrs UK and MAILABS 4hrs US+4hrs UK data. To compare the accents, we synthesized 30 pairs of speech samples (LSTM was sample 1, REMMI was sample 2) and asked 20 Mechanical Turk (MTurk) (Crowston, 2012) participants to rate which sample sounded more British. The rating scale given to MTurk participants was: +2: 2nd sample sounds more British than 1st, +1: 2nd sounds slightly more British than 1st, 0: 2nd and 1st sound equally British, -1: 1st sounds slightly more British than 2nd, -2: 1st sounds more British than 2nd. We repeated the experiment with REMMI as sample 1 and LSTM as sample 2 (reversing the corresponding rating scale) and averaged the scores of the experiments to counter any ordering bias. We calculated the CMOS by averaging the difference in the mean scores for REMMI and LSTM-VAE.

Second, to check if REMMI provided more control on synthesized accent (whether US or British) than LSTM-VAE and provide human verification that the separation in latent clusters led to controllable synthesis, we generated 50 random pairs of (US sample, UK synthesized sample) using LSTM-VAE and REMMI each. We asked 10 MTurk participants to rate if the US and UK samples sounded different. The scale was: 0- Samples sound the same, 1- Samples sound slightly different, 2- Samples sound different. We calculated the Control Score (CS) by averaging the difference in the mean scores for REMMI and LSTM-VAE.

The resulting CMOS and CS with 95% confidence intervals in Table 5 show that in MAILABS 4hrs US+4hrs UK our approach is superior in both producing speech that sounds more British and providing controlled synthesis. In Common Voice due to noisy synthesis, LSTM-VAE and REMMI produce nearly the same accent quality, but a significantly positive CS provides better synthesis control for REMMI. In practice, this means that LSTM-VAE cannot be controlled at test time to produce US/British speech, while REMMI can be better controlled at this task.

5.5 Loss Curves

The conditional loss L_{cond} in equation (4) controls the latent variables being modelled namely $\mathbf{z}_l, \mathbf{z}_o$ and $\mathbf{y}_l$. The trend in Figure 6 for MAILABS dataset shows that REMMI has an accelerated convergence compared to both Transformer-VAE and LSTM-VAE. It can also be seen in Figure 6 that L_{mel} remains the same in all the 3 experiments, LSTM-VAE, Transformer-VAE and REMMI. This shows that while our REMMI is successful in lowering L_{cond}, it does so without hurting L_{mel} or the synthesized mel-spectrogram quality.

We also observed that for a given dataset size in LSTM-VAE, L_{cond} increases with increasing model depth which points towards inferior latent features. This trend is summarized in Figure 6 and shows that Transformer-VAE and REMMI do not overfit to a given dataset size with increasing

layers.

6 Conclusion

In this work we showed that REMMI discovers disentangled latent representations of speech with uncorrelated latent variables allowing better control of speech synthesis. Our layer reordering in Transformers produces notably improved latent clusters of speaker attributes keeping the speaker styles under control on varying dataset sizes with different noise conditions. We can generate mel spectrograms for different text with controllable pitch, pause durations, speaking speed and accent. We also showed that there is a significant boost both in convergence and in the stability of the learnt representations with our proposed method. Going forward we would like to explore the application of REMMI beyond speech, e.g, image captionining with sentiments or text to image rendering with different emotions.

References

Rosana Ardila, Megan Branson, Kelly Davis, Michael Henretty, M. Kohler, Josh Meyer, Reuben Morais, Lindsay Saunders, Francis M. Tyers, and Gregor Weber. 2020. Common voice: A massively-multilingual speech corpus. In *LREC*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate.

J. C. Bezdek and N. R. Pal. 1995. Cluster validation with generalized dunn's indices. In *Proceedings 1995 Second New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, pages 190–193.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation.

Ju-Chieh Chou, Cheng-chieh Yeh, Hung-yi Lee, and Lin-Shan Lee. 2018. Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations. In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association, Hyderabad, India, 2-6 September 2018*, pages 501–505. ISCA.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. December 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Workshop on Deep Learning*.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. *CoRR*, abs/1506.02216.

Kevin Crowston. 2012. Amazon mechanical turk: A research tool for organizations and information systems scholars. In *Shaping the Future of ICT Research. Methods and Approaches*, pages 210–221, Berlin, Heidelberg. Springer Berlin Heidelberg.

D. L. Davies and D. W. Bouldin. 1979. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227.

D. Griffin and Jae Lim. 1983. Signal estimation from modified short-time fourier transform. In *ICASSP '83. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 8, pages 804–807.

Patrice Guyot. 2018. Fast python implementation of the yin algorithm.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Yukiya Hono, Kazuna Tsuboi, Kei Sawada, Kei Hashimoto, Keiichiro Oura, Yoshihiko Nankaku, and Keiichi Tokuda. 2020. Hierarchical Multi-Grained Generative Model for Expressive Speech Synthesis. In *Proc. Interspeech 2020*, pages 3441–3445.

W. Hsu, Y. Zhang, R. J. Weiss, Y. Chung, Y. Wang, Y. Wu, and J. Glass. 2019. Disentangling correlated speaker and noise for speech synthesis via data augmentation and adversarial factorization. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5901–5905.

Wei-Ning Hsu, Yu Zhang, and James R. Glass. 2017. Unsupervised learning of disentangled and interpretable representations from sequential data. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1878–1889.

Wei-Ning Hsu, Yu Zhang, Ron Weiss, Heiga Zen, Yonghui Wu, Yuxuan Wang, Yuan Cao, Ye Jia, Zhifeng Chen, Jonathan Shen, Patrick Nguyen, and Ruoming Pang. 2019. Hierarchical generative modeling for controllable speech synthesis. In *International Conference on Learning Representations (ICLR)*.

Ye Jia, Yu Zhang, Ron J. Weiss, Quan Wang, Jonathan Shen, Fei Ren, Zhifeng Chen, Patrick Nguyen, Ruoming Pang, Ignacio Lopez-Moreno, and Yonghui Wu. 2018. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, pages 4485–4495.

J. Jiang, G. G. Xia, D. B. Carlton, C. N. Anderson, and R. H. Miyakawa. 2020. Transformer vae: A hierarchical model for structure-aware and interpretable music representation learning. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 516–520.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Jack Klys, Jake Snell, and Richard Zemel. 2018. Learning latent subspaces in variational autoencoders. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 6445–6455, Red Hook, NY, USA. Curran Associates Inc.

Simon Leglaive, Xavier Alameda-Pineda, Laurent Girin, and Radu Horaud. 2020. A recurrent variational autoencoder for speech enhancement. In *2020 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2020, Barcelona, Spain, May 4-8, 2020*, pages 371–375. IEEE.

Yingzhen Li and Stephan Mandt. 2018. Disentangled sequential autoencoder. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5656–5665. PMLR.

Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Çaglar Gülçehre, Siddhant M. Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, Matthew M. Botvinick, Nicolas Heess, and Raia Hadsell. 2019. Stabilizing transformers for reinforcement learning. *CoRR*, abs/1910.06764.

Seungwon Park, Dooyoung Kim, and Myun chul Joe. 2020. Cotatron: Transcription-guided speech encoder for any-to-many voice conversion without parallel data. In *INTERSPEECH*.

J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan, R. A. Saurous, Y. Agiomvrgiannakis, and Y. Wu. 2018. Natural tts synthesis by conditioning wavenet on mel spectrogram predictions. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4779–4783.

RJ Skerry-Ryan, Eric Battenberg, Ying Xiao, Yuxuan Wang, Daisy Stanton, Joel Shor, Ron J. Weiss, Rob Clark, and Rif A. Saurous. 2018. Towards end-to-end prosody transfer for expressive speech synthesis with tacotron.

Imdat Solak. 2018 (accessed November 11, 2020). *The M-AILABS Speech Dataset*. https://www.caito.de/2019/01/the-m-ailabs-speech-dataset/.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *CoRR*, abs/1505.00387.

G. Sun, Y. Zhang, R. J. Weiss, Y. Cao, H. Zen, and Y. Wu. 2020. Fully-hierarchical fine-grained prosody modeling for interpretable speech synthesis. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6264–6268.

Tianming Wang and Xiaojun Wan. 2019. T-cvae: Transformer-based conditioned variational autoencoder for story completion. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5233–5239. International Joint Conferences on Artificial Intelligence Organization.

Yuxuan Wang, Daisy Stanton, Yu Zhang, R. J. Skerry-Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Ye Jia, Fei Ren, and Rif A. Saurous. 2018. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 5167–5176. PMLR.

Ya-Jie Zhang, Shifeng Pan, Lei He, and Zhen-Hua Ling. 2019. Learning latent representations for style control and transfer in end-to-end speech synthesis. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, pages 6945–6949. IEEE.

Appendix

A Variational Lower Bound

For an input text sequence Y_t and an observed categorical label y_o frames X can be learnt via the joint distribution $log\ p(X, Y_t, y_o)$. Additional latent variables z_o and z_l can be introduced to discover meaningful representations during this process. Here z_o is a continuous latent learnt on top of shown labels y_o, hence the features z_o discovers is correlated with what is shown to the model via y_o, while z_l is a completely unsupervised continuous variable learnt on top of standard Expectation-Maximization style latent mixture components y_l. Note that y_l is a K-way categorical discrete variable. The variational lower bound can then be formulated as,

$$
\begin{aligned}
log\ p(X|Y_t, y_o) &\geq \mathbb{E}_{q(z_o|X)q(z_l|X)q(y_l|X)} \\
&\left[log \frac{p(X|Y_t, z_o, z_l)p(z_o|y_o)p(z_l|y_l)p(y_l)}{q(z_o|X)q(z_l|X)q(y_l|X)}\right] \\
&= \mathbb{E}_{q(z_o|X)q(z_l|X)}[log\ p(X|Y_t, z_o, z_l)] \quad (5) \\
&- D_{KL}(q(z_o|X)\ ||\ p(z_o|y_o)) \\
&- \mathbb{E}_{q(y_l|X)}[D_{KL}(q(z_l|X)\ ||\ p(z_l|y_l))] \\
&- D_{KL}(q(y_l|X)\ ||\ p(y_l)) \\
&\approx\ log\ p(X|Y_t, \widetilde{z_o}, \widetilde{z_l}) \quad (6) \\
&- \sum_{y_l=1}^{K} q(y_l|X) D_{KL}[\ q(z_l|X)\ ||\ p(z_l|y_l)\ ] \\
& \quad (7) \\
&- D_{KL}[\ q(y_l|X)\ ||\ p(y_l)\ ] \\
&- D_{KL}[q(z_o|X)\ ||\ p(z_o|y_o)\ ] \\
&= -L_{mel} - L_{KL}
\end{aligned}
$$

B Gated Architecture

In the past multiplicative interactions have been successful at stabilizing learning across different architectures (Cho et al., 2014; Srivastava et al., 2015). This motivated us to try out GRU-type gating at the heads of the proposed Transformers. The outputs at the GRU-type gating is controlled by the following equation,

$$
\begin{aligned}
r &= \sigma(W_r^{(l)} y + U_r^{(l)} x), \\
z &= \sigma(W_z^{(l)} y + U_z^{(l)} x - b_g^{(l)}), \\
\hat{h} &= tanh(W_g^{(l)} y + U_g^{(l)} (r \odot x)) \\
g^{(l)}(x, y) &= (1 - z) \odot x + z \odot \hat{h}
\end{aligned}
$$

where r stands for the reset gates, z is the update gates, $\hat{h}$ is the candidate activation similar to other recurrent units (Bahdanau et al., 2016). The overall gate activation $g(x, y)$ takes input x as the residual connection and y the output of the `FeedForward` or `Multi-Head Attention` modules. $g(x, y)$ is basically an interpolation between the previous activations $\hat{h}$ and the residual input x.

C Speaking Rate for $\mathbf{y}_o = 0, 1$

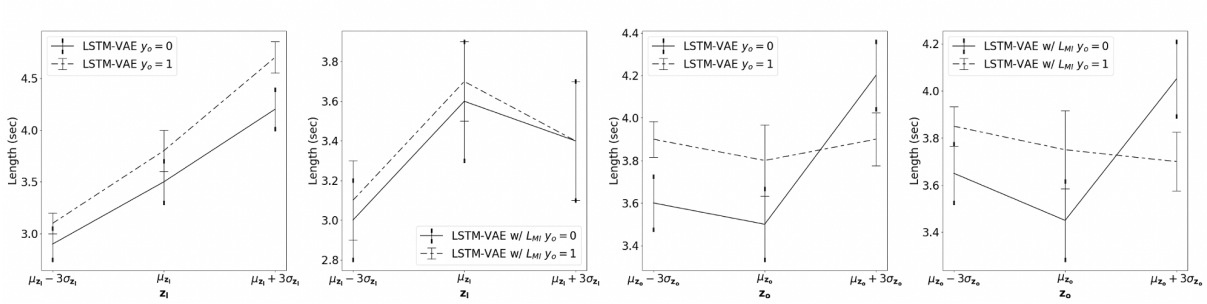


Figure 7: Length of mel-spectrogram synthesized by $\mathbf{z}_l$ in LSTM-VAE for MAILABS is significantly different for $\mathbf{y}_o = 0, 1$ showing that y_o specific information is encoded by $\mathbf{z}_l$. However this difference is no longer significant once we include our proposed L_{MI} terms in LSTM-VAE w/ L_{MI} experiment. $\mathbf{z}_o$ keeps showing different lengths for $y_o = 0, 1$ in both LSTM-VAE and LSTM-VAE w/ L_{MI} experiments demonstrating learnt features which are conditional on y_o.

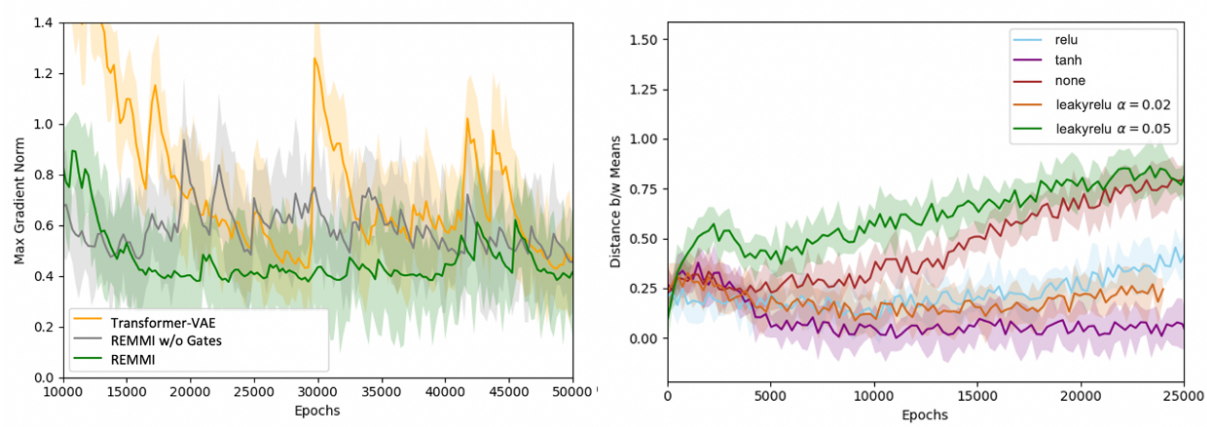


Figure 8: **Left**: Lower gradient norm for REMMI w/ Gates along with smaller variance compared to Transformers-VAE and REMMI w/o Gates. **Right**: Distance between the means of $z_o|y_o$ for $y_o = 0, 1$ for different activation functions at the output of `Multi-Head Attention` and `FeedForward` modules. We see that $LeakyRelu$ with $\alpha = 0.05$ performs the best in segregating the prior clusters among all experiments.

D Ablation Study

D.1 Importance of Gates

Our comparison of Gated architectures with non-Gated ones in Figure 8 shows that the maximum gradient norm which directly influences the convergence is much lower and stable with a lower variance for REMMI (which includes gates) compared to REMMI without (w/o) Gates and Transformer-VAE.

D.2 Choosing the Right Activation

In Figure 8 we see that the distance between $z_o|y_o$ cluster means is very small when the output from `Multi-Head Attention` and `FeedForward` modules are fed to `GRU-Type Gating` layers without any non linearity. Hence our choice of this non linearity was inspired by the trade-off between number of gradient updates and the maximum gradient norm. We see in Table 6 that $relu$ has a high maximum gradient norm ∇_{norm} which led to convergence instability and small distance between $z_o|y_o$ cluster means. But for $tanh$, almost all activations were producing gradient updates and this frequent update was leading to small cluster distance as shown in Figure 8. Hence we needed a function somewhere between $relu$ and $tanh$, which has a small gradient norm while also having fewer gradient updates. $LeakyRelu$ turns out to be the best candidate for this with its high distance between means as shown in Figure 8.

Experiment	% activation	max ∇_{norm}
relu	84.5 (< 0)	40.96
tanh	0 (>+2,<-2)	10.68
leakyrelu	-	7.17

Table 6: Comparing the percentage of activations for which gradient saturates and maximum gradient norm ∇_{norm}

E Compute Information

We ran all our experiments on NVIDIA Tesla V100 GPU with 16GB of GPU memory. Our LSTM-VAE (both with and without L_{MI}) experiments take average 5.81sec/step (seconds per step) with convergence near 40k steps. Transformer-VAE takes an average 2.81sec/step with convergence near 25k steps, and REMMI takes average 2.81sec/step with convergence near 25k steps. Total number of parameters are 28.03mn (million) for LSTM-VAE w/ and w/o MI, 27.84mn for Tranformer-VAE and 28.03mn for REMMI.

F Audio Hyperparameters

Parameter	Value
num mels	80
num freq	1025
max mel frames	900
silence threshold	2
n fft	2048
hop size	275
win size	1100
sample rate	16000
magnitude power	2.0
trim silence	True
trim fft size	2048
trim hop size	512
trim top db	50
preemphasize	True
preemphasis	0.97
min level db	-100
ref level db	20
fmin	55
fmax	7600
power	1.5

Table 7: Parameters for converting wav files to mel-spectrograms

G Tacotron-2 Hyperparameters

Parameter	Value
batch size	64
output frames per step	4
max training iterations	100k
optimizer	Adam
β_1	0.9
β_2	0.999
ϵ	1e-6
L2 regularization weight	1-e6
learning rate decay	exponential
initial learning rate	1e-3
decay start epoch	40k
decay epochs	18k
final learning rate	1e-4
clip gradients	True
teacher forcing	constant at 1

Table 8: Hyperparameters common for all experiments

H VAE Hyperparameters

Parameter	Value
z_l dim	3
z_o dim	2
$\|y_o\|$	2 (UK, US)
z_o, z_l convolution channels	128
activation function for convolution	$tanh$
kernel size	3x3
MC estimate num_samples	1
num_units for LSTM	128
min logvariance for $q(z_l\|X)$	-4
min logvariance for $q(z_o\|X)$	-6
initial mean for $p(z_l\|y_l)$	
$p(z_l\|y_l = 0)$	(1,0,0)
$p(z_l\|y_l = 1)$	(0,1,0)
$p(z_l\|y_l = 2)$	(0,0,1)
initial logvariance for $p(z_l\|y_l)$	-4
initial mean for $q(z_o\|y_o)$	
$p(z_o\|y_o = 0)$	(-0.5, -0.5)
$p(z_o\|y_o = 1)$	(+0.5, +0.5)
initial logvariance for $p(z_o\|y_o)$	-5
dropout	0.1
zoneout (for LSTM)	0.1
q_ψ num_layers	4
q_ψ num_units	8
q_ψ activations	$tanh$
Transformer d_model	64
Transformer num_heads	4
Transformer feedforward_dimension	256
max positional encoding	584

Table 9: Hyperparameters used for our VAEs