Institut für Maschinelle Sprachverarbeitung
Universität Stuttgart
Pfaffenwaldring 5B
D-70569 Stuttgart

Master thesis

Controllable Text-to-Speech System: Speaking Style Control Using Hierarchical Variational Autoencoder

Yung-Ching Yang
3516821

Studiengang:	M.Sc. Computational Linguistics
Prüfer*innen:	Prof. Dr. Ngoc Thang Vu
	Dr. Antje Schweitzer
Betreuer:	Florian Lux
Beginn der Arbeit:	25.07.2023
Ende der Arbeit:	25.01.2024

Erklärung (Statement of Authorship)

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst habe und dabei keine andere als die angegebene Literatur verwendet habe. Alle Zitate und sinngemäßen Entlehnungen sind als solche unter genauer Angabe der Quelle gekennzeichnet. Die eingereichte Arbeit ist weder vollständig noch in wesentlichen Teilen Gegenstand eines anderen Prüfungsverfahrens gewesen. Sie ist weder vollständig noch in Teilen bereits veröffentlicht. Die beigefügte elektronische Version stimmt mit dem Druckexemplar überein.

Hereby, I declare that I have independently authored this present work and have not used any sources other than those specified. All direct and indirect sources are acknowledged and cited with precise references. The submitted work has not been part of any other examination procedure, either in whole or in significant parts. It has not been published either in whole or in parts. The enclosed electronic version is consistent with the printed copy.

(Yung-Ching Yang)

Abstract

This research proposes an utterance embedding model that provides disentangling and scalable control over latent attributes in human speech. Our model is formulated as a hierarchical generative model based on the Variational Autoencoder (VAE) framework, integrated with the FastSpeech2 Text-to-Speech (TTS) system. The work demonstrates that image initiative networks on hierarchical pattern learning can be adapted to model complex distributions in speaking styles and prosody. This work merges advancements in VAE research—particularly those addressing critical statistical challenges such as posterior collapse and unbounded KL divergence—with recent studies focusing on structural enhancements of architectures in VAEs. We introduce a hierarchical structure in latent variable modeling and augment the learning objective with hierarchical information to ensure the latent variables at each level are hierarchically factorized. This approach learns the smooth latent prosody space and deepens our understanding of the relationship between the hierarchical nature of prosody and neural network architecture. Through our customized control mechanism, integrated into various levels of the latent spaces, the model is capable of manipulation of prosodic elements, allowing for both independent and scalable adjustments. By incorporating these techniques, our model is capable of capturing a wide range of prosodic variations, offering a refined level of control and expressiveness in speech synthesis in unsupervised learning contexts.

Contents

List of Tables

List of Figures

1 Introduction

In recent years, many attempts have been made to integrate style information and controllability into Text-to-speech (TTS) approaches. In most end-to-end setups, which inherently learn prosody in the model, the style conversion or controllability is done by integrating additional style modules in supervised and unsupervised ways. Among these developments, extensions from the concept of Global Style Token (GST)(52) focus on learning refined utterance-level prosody embeddings. In parallel, deep generative models such as Normalizing Flows (NF)(48), Generative Adversarial Networks (GANs)(14), and Variational Autoencoders (VAEs)(30) have emerged as powerful tools in latent representation learning, particularly for complex data types like images and speech. Recent studies based on generative models offer a framework to capture and manipulate hidden patterns in data, bridging the gap between rigid style control, such as categorical label usage, and dynamic variations inherent in natural speech. One of the primary challenges has been generating informative single embeddings that represent all characteristics of an utterance. To address this, recent studies such as (9) have developed a fine-grained encoder capable of extracting variable-length style information from utterances. In a similar vein, the Quantized Fine-Grained VAE (QFVAE)(53) was proposed to merge categorical clarity with the adaptability of latent space modeling by quantizing prosody latent representations into a set number of classes, thereby enhancing the naturalness of audio samples generated from continuous latent spaces. To cater to the few-shot adaptation needs, especially in multi-lingual and multi-speaker scenarios, TTS systems have adopted the strategy of conditioning on multiple specialized embeddings, such as speaker embeddings, language embeddings, and stress and tone embeddings(39).

In our research, inspired by the inherent hierarchy in human speech characteristics and the architectural design of Nouveau VAE (NVAE)(58). We introduce hierarchical blocks in the VAE structure and reinforce the learning objectives' hierarchical nature through residual distribution and leveled reconstruction losses. This hierarchical structure facilitates the disentanglement of prosodic features, and we integrate a control mechanism across our latent spaces, which allows independent and scalable manipulation of speaking styles. Our model is able to capture a broad range of prosodic nuances and provides more expressive and controllable speech synthesis.

1.1 Motivation and Use Case

Data privacy is a critical concern, especially in protecting Personally Identifiable Information (PII). Traditional masking techniques, such as distortion or pixelation, have been widely used to conceal PII in images. Recently, this focus has broadened to cover more multimedia types, such as speech and video data. For audio data, confidentiality concerns extend beyond mere content to encompass elements like voice characteristics and background noise, which could reveal geographical locations or personal identifiers. Early approaches to address speech privacy employed techniques like adding nonsensical masking noise to obscure speech information or voice distortion, which often lacked naturalness and intelligibility. With the advent of data anonymization techniques in image domains, such as the website- This Person Does Not Exist, anonymity has become more sophisticated while retaining robustness. Although there are existing anonymous voice generators that allow users to adopt various voices, including those of celebrities or fictional characters, these techniques often have potential security vulnerabilities and lack naturalness. In this regard, we aim to provide a system that offers vivid and explicit voice masking capabilities while still maintaining control over the output. This balance of privacy with usability opens new possibilities for secure and private speech communication.

1.2 Research Question

In this research, we build a controllable TTS system by integrating an utterance embedding generator based on a hierarchical variational autoencoder (HVAE) with a control mechanism at the latent space level. The objective is to obtain disentangling and scalable style representation and enable flexible speech generation control. Besides, we are particularly interested in the difference between features learned in different hierarchical levels and the relation between them within our proposed framework. The research questions can be broken down into the following points:

1. **Implementation of Hierarchical VAE as Embedding Function**: How can a hierarchical VAE be developed as an utterance embedding function within a TTS system?
2. **Control Integration in Non-Isotropic Latent Spaces**: How can controls be integrated in latent spaces beyond the isotropic prior distribution $p(z)$ in VAEs?
3. **Speaking Style Control and Hierarchical Analysis**: How does the hierarchical structure of the VAE influence the system's ability to capture varying levels of speaking style? What is the interrelationship between these captured features across different levels and dimensions?
4. **Disentanglement and Scalability of Prosodic Features**: To what extent can the controlled prosodic features be disentangled, combined, and scaled?

2 Theoretical Background

In this section, we introduce the theoretical background of our work, including the concepts behind different deep generative models and their recent advancements in various applications, especially in image and speech domains. This serves as the cornerstone for us because it provides related frameworks for modeling complex data distributions and the possible integration with the control mechanism. In our work, we integrate VAE as a backbone model to the TTS system to model the latent characteristics in human speaking styles and other potential hidden features, like microphone characteristics and noise levels. We extend the concept of generative models and focus more on latent representation learning by looking into methods for encoding and decoding speech data into meaningful latent space. Next, we introduce more general TTS methods and compare different variance modeling and prediction methods essential for capturing the variability inherent in human speech and enabling dynamic control over speech synthesis. In the end, with the hierarchy nature of acoustic features, we look into the relationship between speaking style and model architecture, illustrating how our proposed model leverages hierarchical structures to achieve fine-grained control over various aspects of speech, such as pitch, pace, and energy.

2.1 Deep Generative Model

Generative models have become a common tool in diverse subfields of artificial intelligence and machine learning due to their power of content creation and creativity. Integrating deep neural networks for parameterizing these models has marked a significant advancement, particularly with the strides made in stochastic optimization techniques, increasing the potential for scalable and efficient modeling of complex and high-dimensional data and capturing more underlying dynamics. The core concept behind generative models is to learn the underlying patterns or distributions of data to generate similar or new data. The concept of autoregressive models plays a vital role in developing and evolving modern generative modeling techniques. The complex data distributions can be used in different ways, such as latent space exploration and reconstruction, reversible transformations, adversarial training by integrating discrimination, or a stochastic denoising process. These methods can be generalized into the based generative models like VAEs, Flow-based models, GANs, and Diffusion-based models(21), with relative strengths and weaknesses. For example, in optimizing the closeness between the distributions of data and model distributions, autoregressive models are good at accurately assessing how well they fit the data distribution by offering tractable likelihoods. However, they lack a direct mechanism for abstract feature learning, focusing mainly on sequence prediction. On the one hand, VAEs benefit from learning latent feature representations of data through their encoding-decoding structure. Still, they face a limitation in directly computing the marginal likelihoods of data samples because they rely on variational methods, which approximate the true data distribution, thus providing a bound on the likelihood rather than an exact value. In contrast, the flow-based models and diffusion-based models have similar approach, which learns to convert from a simple prior distribution to the unknown complex target feature distribution corresponding to the conditional information, but with different approaches in likelihood computation and sample generation, where flow-based models offering exact likelihood computations and tractable modeling of the complex distributions by applying reversible transformations and bijective mappings; whereas diffusion models focusing on sequences of stochastic denoising steps that iteratively refining noise into structured data. Lastly, GANs solve these issues differently by turning to likelihood-free training but with indirect quality assessment measures. They focus on generating data that cannot be distinguished from real data to a discriminator network that avoids the direct computation of likelihoods. We will give more details into the distribution modeling and latent representation learning in the next section 2.2.

Based on these natures of different backbone models, nowadays, researchers have taken their strengths and extended to focus on adding expressiveness and controllability in the generation process. In the image domain, recent research, such as DALL-E 2(44) and GLIDE(41), use discrete image tokens guided by text diffusion models for direct image generation. Specifically, both are based on diffusion models; DALL-E 2 employs a complex process in which the model constructs visual content by gradual denoising a random noise pattern, thereby translating textual input into a coherent visual representation. Similarly, GLIDE extends the concept of diffusion models by conditioning the model on additional textual information, thereby enabling text-conditional image generation. As in the speech domain, different topics also benefit from integrating generative models in realizing speech enhancement and style adaptation. For example, Stylebook (38) proposes methods for any-to-any voice conversion by extracting content-dependent target style embeddings and feeding them into a diffusion-based U-Net (49) decoder to generate the styled speech mel-spectrogram. Similarly, Noise-Aware Speech Enhancement (NASE) (24) also utilizes diffusion, and it realizes speech enhancement by adding noise-specific information inside noisy speech to guide the reverse denoising process based on the conditional diffusion probabilistic model. Moreover, to increase accuracy in variance prediction, VarianceFlow(34) integrates normalizing flow to a traditional deterministic variance predictor to capture the full range of speech characteristics during the training stage. We will introduce more recent TTS approaches and discuss the controllability and variance that can be made deeper in Section3.

2.2 Latent Representation Learning

Earlier speaker embedding generation approaches in TTS systems involved techniques like one-hot encoding, clustering, and factor analysis. These methods either directly encoded categorical speaker information or learned speaker embeddings based on the distribution of speech data. However, they were limited in their ability to generate new samples as they did not model the full probability distribution of the data. As introduced in the previous section, deep generative models such as VAEs and GANs have been employed to learn an approximate mapping between Gaussian latent variables and data samples, especially when the true latent variables have an intractable posterior distribution (30). GANs, in particular, have shown promise in achieving relative and explicit control over generated outputs. A typical GAN optimizes the following objective function:

$$\min_{G} \max_{D} \mathbb{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{1}$$

This formula reflects the adversarial game between the generator (G) and the discriminator (D), where the generator strives to produce data-like samples, and the discriminator aims to distinguish between real and generated samples. Recent GAN-based approaches realized both relative control and explicit control over image generation by exploiting the inherent disentanglement properties of their latent space and leveraging conditional GANs, respectively. For example, InterFaceGAN (51) used off-the-shelf binary classifiers to find separation boundaries in the latent space where each side of the boundary corresponds to an opposite semantic attribute (e.g., young v.s. old). In conditional GANs, which have been widely used in the image domain, the generator is conditioned on some input variables, such as a class label, an attribute vector, or a style code, which enables explicit control over the generated images. Some recent approaches introduced additional regularization or disentanglement techniques, such as adversarial regularization or information bottleneck, to encourage the generator to learn a disentangled representation of the input variables. While these methods provide flexibility and the prospect of optimizing different objectives for high likelihood and high-quality sample generation, GANs do not explicitly measure how well the generated samples match the data distribution. This can lead to the generator producing only a limited set of samples, resulting in a lack of diversity in the generated samples. This can be problematic in the context of posterior approximation. Though an improvement can be made to posterior approximation by using Wasserstein Distance as an objective function (3), the control mechanisms remain a challenge.

On the other hand, the objective function of VAEs ensures that the generated samples are diverse and representative of the true distribution. The vanilla VAE has been used to Tacotron2 (52) to learn the latent representation of speaking style. The approaches that try to increase controllability to VAE can be divided into 2 categories. The first group of works focuses on solving the Kullback-Leibler (KL) vanishing problem and tries to exercise control over KL-divergence to add diversity to the output yet maintain authenticity. The objective function of VAEs, consisting of log-likelihood and KL-divergence

$$logp_\theta(x) \geq \mathcal{L}_{\text{VAE}} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x} \mid \mathbf{z})] - \mathrm{KL}(q_\phi(\mathbf{z} \mid \mathbf{x}) \parallel p_\theta(\mathbf{z})) \tag{2}$$

which posts a trade-off between reconstruction accuracy bounds and the manipulation of output diversity. β-VAE blindly controlled the KL-divergence by adding a hyperparameter β as a weight. Following this work, to improve dynamic adjustment to the weight and avoid the large reconstruction error, ControlVAE (50) deployed a non-linear Proportional Integral (PI) controller (4) to further tune the hyperparameter β by reducing the distance between desired KL-divergence and the actual KL-divergence using the concept of PID control (4). These approaches usually sacrifice interpretability and have less stable controllability. As the latent space is continuous, one problem with using the vanilla VAE is that the decoder may not capture all variations in the original data distribution because the prior distribution is assumed to be a Gaussian unit. As a result, another group of works proposed more recently focuses on making variations, such as adding conditions and multiple layers. This then increases the flexibility in modeling the data distribution and allows the model to learn different levels of features. To make use of the hierarchical property of speech data, latent spaces can also be modeled in different levels to learn latent variables in different hierarchies (22) (31). In work (22), one level is a categorical variable, and the second level, conditioned on the first, is a multivariate Gaussian variable; the two levels are responsible for the attribute group (e.g., clean/noisy), and the attribute configurations (e.g., noise level, speaking rate) respectively. With a similar concept, NAVE proposed a deep hierarchical VAE to improve the expressiveness of the model by partitioning the latent variables into different groups based on the complexity of the embedding to capture hierarchical structures in the image data.

GANs and VAEs have demonstrated great variations and flexibility, though they still face challenges in explicitly learning the probability density function of real data. Flow-based models can address this limitation with the aid of normalizing flows(48). Utilizing the property of the Jacobian of invertible functions and the change of variables formula, Flow-based models approximate the data distribution of $\mathbf{x}$ using the following relationship:

$$p_\theta(\mathbf{x}) = p_\theta(\mathbf{z}) \left| det \frac{\partial f^{-1}}{\partial x} \right|, \quad log p_\theta(\mathbf{x}) = log p_\theta(\mathbf{z}) + log \left| det \frac{\partial f^{-1}}{\partial x} \right| \tag{3}$$

Here, the log determinant term in Equation 3 is replaced by the sum of the log determinants of each intermediate Jacobian, as shown in Equation 4:

$$log p_\theta(\mathbf{x}) = log p_\theta(\mathbf{z}) + \sum_{i=1}^{K} log \left| det \frac{\partial f_i^{-1}}{\partial z_i} \right| \tag{4}$$

This formulation enables the model to be trained directly using maximum log-likelihood. Consequently, the training objective of a Flow-based generative model is the negative log-likelihood (NLL) over the training dataset $\mathcal{D}$, as specified in Equation 5:

$$\mathcal{L}(\mathcal{D}) = -\frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} log p_\theta(\mathbf{x}) \tag{5}$$

In contrast to VAEs and GANs, which approximate the lower bound of the log-likelihood (ELBO) and rely on discriminator-generator optimization, respectively, Flow-based models offer a more direct and flexible approach. This makes them ideally suited for modeling variational distributions that are complex enough to encompass the true posterior distribution.

2.3 Variance Modelling and Prediction

One of the critical issues in the TTS system is the one-to-many mapping problem, where a given text can be pronounced in multiple ways depending on the speaker's characteristics, speaking styles, and other variances. More explicit variance prediction approaches in TTS models, such as FastSpeech (47) and its variant FastSpeech 2 (45), have been trained to predict the ground-truth variance factors based on the input text using mean squared error (MSE) loss. The variance is explicitly modeled through pitch, energy, and duration predictors. FastSpeech 2 addressed the complexity of energy and pitch by using a quantile regression network to predict the quantiles of the energy distribution and deploying continuous pitch representation respectively (55; 19). Another approach to predicting non-textual information, including variance and speaking style, is to consider them as a representation in the latent space and use them as conditions with the input signal. In some extension works for prosody control based on Tacotron (61), this is achieved by introducing a reference encoder to learn a fixed-dimensional embedding as the prosody space from the reference signal (52). Flowtron (59) used a flow-based Mel-spectrogram encoder as the latent encoder, proposed in GMVAE-Tacotron (22), to map the distribution over Mel-spectrograms to a latent space. The aforementioned works often have problems modeling complex variance patterns and controlling those factors independently because of their deterministic prediction approaches and assumptions on dimensionality. To address these issues, recent approaches have shown growing interest in using stochastic methods to model and predict the distribution of variance features. For example, in VAE and NF, variance prediction can be seen as the posterior distribution approximation. Since they do not make common simplifying assumptions about the marginal or posterior distribution, they have higher flexibility in target distribution approximation. In particular, flow-based generative models can capture complex multi-model distributions of speech variance and speaker characteristics by transforming simple distributions into more complex ones using a series of invertible transformations. The Mel encoder in Flowtron (59) is the first TTS application that uses affine coupling layer (12) as building blocks in their invertible neural networks. The concept is further deployed to a stochastic duration predictor, which learns the joint distribution of the estimated phoneme duration in Variational Inference with adversarial learning for end-to-end Text-to-Speech (VITS) (27). In this case, the forward and inverse transformations control the learning flow between the input distribution (a complex distribution, e.g., phoneme duration labels) and output distribution (a simple distribution, e.g., Gaussian random variables). To overcome the limitations of discontinuation and high dimensionality, duration is dequantized and augmented using the methods of variational dequantization (20) and data augmentation (8). Based on FastSpeech 2, VarainceFlow (34) used similar approaches to replace the deterministic functions in variance predictors with flow-based stochastic pitch and energy predictors.

2.4 Hierarchy in Speaking Style

Recent research has highlighted the hierarchical nature of speech features, covering the spectral elements processed in the auditory cortex to the semantic understanding in higher brain areas. This hierarchical representation is important for TTS systems, particularly in unsupervised expressive speech synthesis (UESS), which the concept is proposed in (2). The models with similar UESS concepts aim to synthesize expressive speech without relying on explicit speech expression labels. While labels have been shown to aid in modeling, unsupervised methods are increasingly preferred due to the ease of obtaining expressive speech from sources like online videos or audiobooks, where manual annotation is impractical(13). Also, the reliability of manually annotated labels, such as those categorizing emotions(18), can vary significantly, with differences in expression intensity within the same category.

Focusing on latent representations for styles and prosody, recent TTS studies have utilized VAEs (2; 22; 54). These studies acknowledge that speech features manifest distinctly across various levels, such as phonetic, word, and utterance. For instance, the work (54) proposed a multilevel model integrated with a hierarchical latent variable model based on Tacotron 2(61) captures prosody representations at both the utterance level and finer levels like words and phones. NVAE is one of the first works that proposed adding hierarchy in VAEs to enhance the expressiveness and controllability of latent variables. This is achieved by learning a hierarchical prior $p_\theta(\mathbf{z})$ and partitioning it into disjoint groups $z = \{z_1, z_2, \ldots, z_L\}$, where L represents the number of groups. The prior and approximated posterior are represented by $p(z) = \prod_l p(z_l|z_{<l})$ and $q(z|\mathbf{x}) = \prod_l q(z_l|z_{<l}, \mathbf{x})$, respectively. Each level in the hierarchy l has a posterior, as shown in Equation 7, modeled by individual VAE layers that stack to form a hierarchical structure. Each VAE layer, both in the encoder and decoder, models a specific level of abstraction corresponding to a group of latent variables. The generator $p(\mathbf{x}, z)$ in each VAE is a top-down network that generates parameters of conditional distributions by sampling from each hierarchy (group), passing each sampled latent variable z_l to the next group $z_{>l}$. During inference, a bidirectional encoder infers these latent variable groups sequentially, maintaining the order between $q(\mathbf{z}|\mathbf{x})$ and $p(\mathbf{z})$. This method has shown great performance in addressing high-fidelity images through downscaling and upscaling images in different groups. Though utilizing latent representations of speech, such as speaker embeddings, does not take advantage of the spectral format as images do, the nature of speech characteristics and works based on this assumption provides similar opportunities for representation and control.

$$\begin{aligned}\mathcal{L}_{\text{VAE}(\mathbf{x})} &:= \mathbb{E}_{q_{(\mathbf{z}|\mathbf{x})}}[\log p(\mathbf{x} \mid \mathbf{z})] \\ &\quad - \text{KL}(q(\mathbf{z_1} \mid \mathbf{x}) \parallel p(\mathbf{z_1})) - \sum_{l=2}^{L} \mathbb{E}_{q_{(\mathbf{z}_{<\mathbf{l}}|\mathbf{x})}}[\text{KL}(q(\mathbf{z_l}|\mathbf{x}, \mathbf{z_{<l}}) \parallel p(\mathbf{z_l})|\mathbf{z_{<l}}))]\end{aligned} \tag{6}$$

$$q(z_{<l}|\mathbf{x}) := \prod_{i=1}^{l-1} q(z_i|\mathbf{x}, z_{<i}) \tag{7}$$

3 Related Work

Initial efforts in end-to-end expressive TTS models primarily focused on explicit style control using categorical labels like emotions and speaker identity (36). These early methods, however, faced challenges such as the intensive labeling effort and limited expressive diversity. Subsequently, the field evolved towards unsupervised approaches, notably through the use of reference encoders. These encoders extract global style tokens (GSTs) from reference audio, as seen in (52). Further developments include multi-scale reference encoders, which capture both global-scale and local-scale prosodic features (37), and reference attention mechanisms for finer alignment of prosody embeddings with phoneme sequences (35). Recently, some researchers had taken the success in text and image generation guided by a text description prompt and applied the concept to the speech domain. PromptTTS (17) guides the TTS process using style prompts and content prompts that control the acoustic features and synthesized content, respectively. Similarly, StyleTagging-TTS (ST-TTS)(28) proposes a non-autoregressive expressive TTS model utilizing style tags, which are extremely short sentences like ”in a loud voice” as additional style modules. The style encoder in instructTTS (63) consists of an audio encoder that encodes style information from the target mel-spectrogram, then uses the style information in the ground-truth audio.

There is growing interest in developing controllable speaker embeddings by customizing model architectures based on the nature of prosodic characteristics in different speech types. These embeddings, essentially low-dimensional vector representations, encapsulate information relevant to an individual's speaking style (6). To achieve control using speaker embeddings, auxiliary generative models have been deployed, as seen in works like (22). These models aim to learn speaker embeddings that are both representative and controllable, allowing for precise manipulation of speech characteristics. One major challenge is interpreting the phone-level latent space, as latent dimensions can entangle. Consequently, some works, motivated by goals similar to our research in modeling speaking styles, focus on model engineering to learn disentangled latent representations for prosody. This involves adding conditions in VAEs to learn controllable speaker embeddings (22) (2) or imposing multilevel modeling strategies in VAEs, taking advantage of the hierarchical structure of spoken language (54).

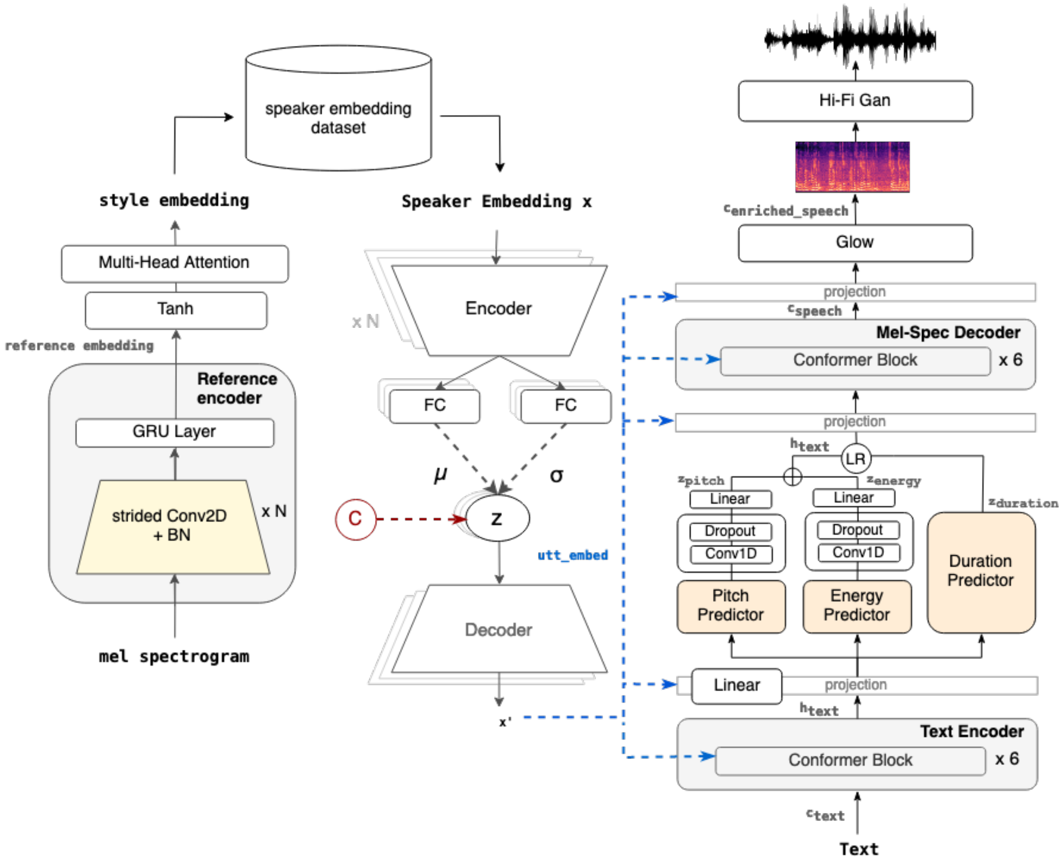


Figure 1: Architecture of the Controllable TTS System

4 Methods

4.1 System Architecture

The overall model architecture of a controllable TTS system is shown in Figure 1, which consists of two main components: the speaker embedding function and the TTS model. Our TTS component is adapted from the variance adaptor in FastSpeech2 (45) integrated with the Flow-based post-processing network from PortaSpeech (46). We use pre-trained GST model(62) introduced in Tacotron (61) as the speaker embedding generator to create a training dataset for our controllable utterance embedding function, which we employ a hierarchical Variational Autoencoder (HVAE) framework with multiple levels within the encoder and decoder, integrated with control mechanisms in the latent representation spaces. Once our utterance embedding function is trained, the GST model is frozen, and the embeddings are sampled and controlled via the latent space in the specified levels.

4.1.1 GST as Speaker Embedding Function

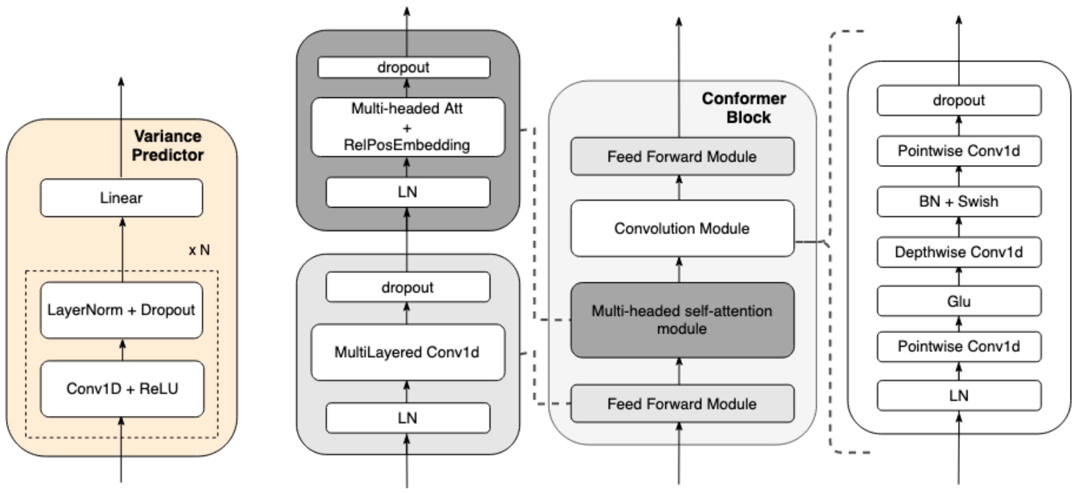
The pre-trained speaker embedding function comprises two main components: a reference encoder and a style token layer, shown in the left part of Figure1.

Reference Encoder: The reference encoder, adapted from an extension to the Tacotron architecture (52), comprises an N-layer stack of 2D convolutions. Each convolutional layer utilizes 3×3 filters with a 2×2 stride, followed by batch normalization and a ReLU activation function. The final layer is a Gated Recurrent Unit (GRU) (10)layer. In our setup, the pre-trained model uses $N = 8$ layers and has the following number of filters: 32, 32, 64, 64, 128, 128, 256, 256. The process of obtaining the reference embedding involves two stages: first, the reference signal is downsampled by this CNN architecture. Secondly, the intermediate output is then compressed into a single fixed-length vector using a recurrent neural network with a single D-width GRU. The D-dimensional reference embedding of the GRU is the pooled summarization of the sequence. Here, D denotes the number of GST units, indicating the output dimension of the reference encoder. We use $D = 256$ in our pre-trained model.

Style Token Layer: The reference embedding obtained from the reference encoder is fed to the style token layer, which consists of a bank of K style token embeddings and a Tanh activation function, followed by the Multi-Headed Attention module. The reference embedding is used as the query vector to the attention module, which outputs a set of combination weights that represent the contribution of each style token to the encoded reference embedding. The weighted sum of the GSTs is the output style embedding. Here, K implies the number of GST tokens, a set of learnable vectors within the model. Each token represents a different aspect or ”style” of speech, such as intonation, rhythm, stress, or any other stylistic characteristic that can be captured from speech data. Our pre-trained model uses $K = 2000$, and the output style embedding is a 64-dimensional vector.

4.1.2 Text-to-Speech System

The TTS model comprises five main parts: a text encoder, a variance predictor, a Mel-spectrogram decoder, a Flow-based Post-Net (26), and a HiFi-GAN (33) vocoder. Both encoder and decoder have similar architectures as convolution-augmented transformer (Conformer) (16) shown in Figure 2 (b). A Conformer block comprises two macaron-like feed-forward layers with half-step residual connections sandwiching the multi-headed



(a) Variance Predictor

(b) Conformer Model Architecture

Figure 2: Architecture of the Variance Predictor and Conformer Blocks

Note: The figure illustrates the detailed structure of (a) the variance predictor, which is responsible for capturing prosodic variations, and (b) the conformer blocks, which facilitate the encoding and decoding processes within the TTS component.

self-attention and convolution modules. The input text is first turned into phonemes and then vectorized as articulatory features c_{text} as the input to the text encoder to obtain the encoded text, denoted as h_{text}. With a framework similar to FastSpeech2, the variance predictor consists of a pitch, energy, and duration predictor. Figure 2 (a) shows the structure of the three predictors, consisting of a N-layer (pitch predictor:$N = 5$, energy, and duration predictor: $N = 2$) 1D-convolutional network with ReLU as an activation function, followed by a layer normalization layer and the dropout layer. The final layer is a linear layer for projection. During training, taking the pitch predictor as an example, the encoded text h_{text} and observed pitch variance information are fed into the predictor to obtain the pitch z_{pitch}. The process involves modeling the feature values of pitch across the text using a pitch predictor. The output is then further transformed using an additional embedding layer comprising 1D convolutional layers, dropout layers, and a linear layer at the end. Once the pitch and energy values are embedded, represented as z_pitch and z_energy respectively, they are integrated with the encoded texts as the input to the length regulator, which utilizes predicted durations, denoted as $z_duration$, to adjust the temporal structure of the encoded sequence. This ensures the duration of each phoneme in the encoded text aligns with the durations predicted by the model. The duration predictor uses the phoneme duration obtained by forced alignment as training objectives, allowing for more accurate mapping. The system has additional pitch and energy variance scaling functions to tune

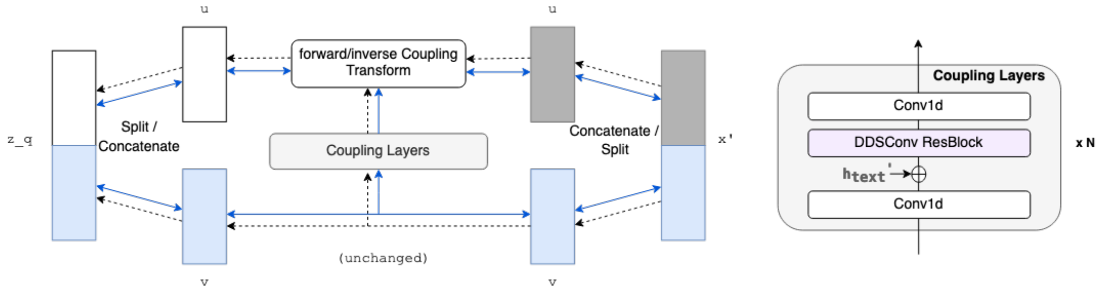


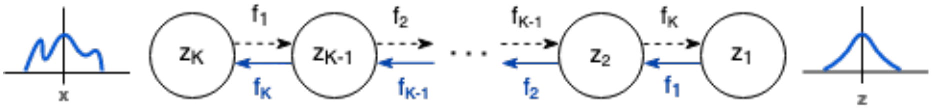
Figure 3: Affine Coupling Layer in Glow Module

Note: In this example, the posterior flow module transforms Gaussian noise z_q into random variables u and v to approximate the posterior distribution $q_\phi(u, v|x, h'_{\text{text}})$. This approximation is achieved by learning the complex distribution x'. The forward coupling employs piecewise rational quadratic and log transforms, while the inverse coupling performs the reverse transformation to approximate the likelihood $q_\phi(x'|z_q, h'_{\text{text}})$.

these prosodic features to enhance speech expressiveness. Adjusting these factors impacts the dynamic range of speech; above the standard midpoint, it increases variance for more expressiveness, while below, it yields a more uniform pitch or energy profile. Similarly, duration scaling affects the overall speech length, with variations influencing the speed and flow of the utterance. Additionally, a separate factor for pause duration scaling finely tunes the rhythm and pacing of speech. In the pre-trained model, we employ a standard scaling value of 1.0 (midpoint) for all three factors, ensuring a balanced modulation of these prosodic elements.

The encoded texts, enriched with variance information, are fed into the mel-decoder to obtain the predicted spectrogram denoted as c_{speech}. The integration of the speaker embedding utt_{embed} and the encoded text h_{text} is realized by adding projection layers, which normalizes the embedding along their feature dimension and concatenates it with the latent text embedding. In the first projection layer, followed by the text encoder, speaker embedding is integrated with the encoded text h_{text} using a linear bottleneck layer before projection.

Glow Module: In our TTS model, we incorporate a flow-based post-net as an enrichment layer to enhance the quality of the generated mel-spectrograms. The central element of this post-net is the affine coupling layer, shown in Figure 3. This layer transforms mel-spectrogram samples into a latent prior distribution, typically an isotropic multivariate Gaussian, during the training phase. This transformation is achieved through a series of invertible functions known as flow steps. As illustrated, the architecture of our post-net is conditioned on the outputs of the preceding network layers and encoder, similar to the configuration in PortaSpeech(46). During training, the post-net computes the exact log-likelihood of the data by executing a forward transformation. This transformation process involves converting the mel-spectrogram samples c_{speech} into the latent prior distribution (isotropic multivariate Gaussian z) through a chain of K transformations f_k, where $z = f_K \circ f_{K-1} \circ \ldots \circ f_2 \circ f_1(x)$.



During inference, the process is reversed: $x = z_K = f_K \circ f_{K-1} \circ \ldots \circ f_2 \circ f_1(z)$, where latent variables sampled from the Gaussian distribution are transformed back into mel-spectrogram features, resulting in high-quality speech synthesis. Based on the exact likelihood estimation of normalizing flow, this method differs from simpler loss-based (L1 or MSE) or VAE-based models. Still, it addresses the issue of over-smoothing, often encountered in speech synthesis, thereby producing more realistic and detailed outputs.

4.1.3 Hierarchical VAE as Utterance Embedding Function

Our controllable utterance embedding function is adapted from the NVAE framework, initially formulated for 3-dimensional image datasets; modifications include redesigning the network architecture for 1D data, integrating hierarchical loss functions, and establishing a control mechanism within the latent space across various levels. The architecture comprises six hierarchical levels in both the encoder and decoder, characterized by the following downsampling and upsampling dimensions: $[56, 48, 40, 32, 24]$ and $[24, 32, 40, 48, 56]$, respectively, as Figure 5 shown. Within the encoder and decoder blocks, as illustrated in Figure 4 (a), (b), we employ a combination of 1D convolutions, batch

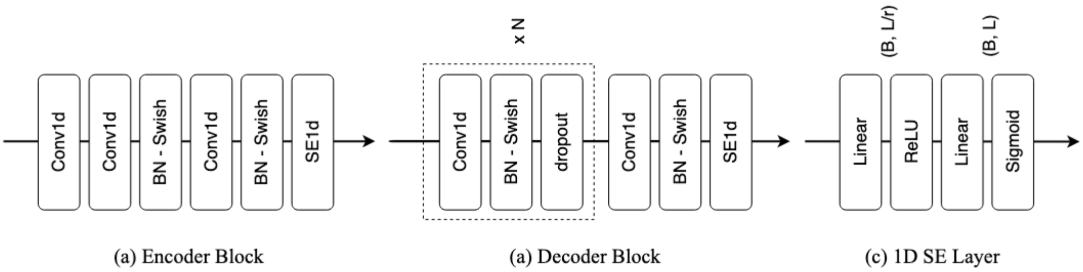


Figure 4: Architecture of Downsampling and Upsampling Blocks

Note: This figure details the architectural components of the VAE embedding function, delineating (a) the encoder's downsampling blocks, (b) the decoder's upsampling blocks, and (c) the implementation of the 1D Squeeze-and-Excitation (SE) as the last layer in both the encoder and decoder blocks.

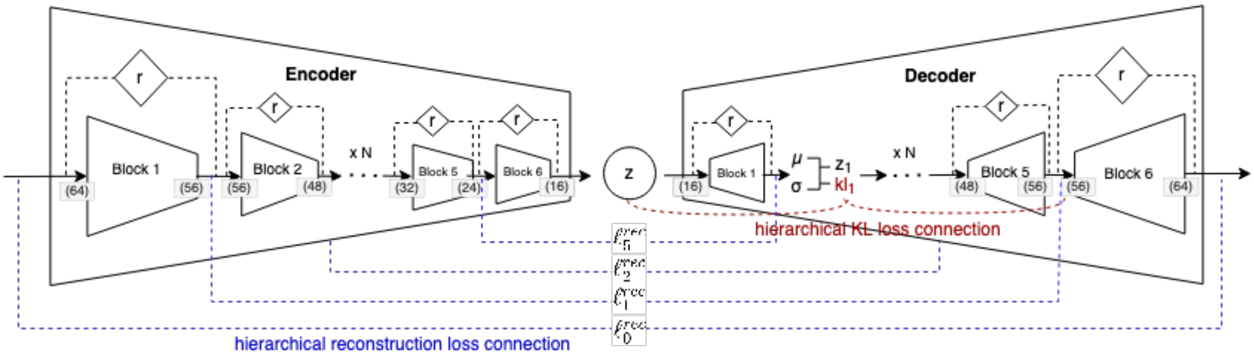


Figure 5: Architecture of Hierarchical VAE Model

normalization, and the Swish activation function, where the Swish function is defined as $\mathrm{swish}(x) = x \cdot \mathrm{sigmoid}(\beta x) = \frac{x}{1+e^{-\beta x}}$. This is followed by a Squeeze and Excitation (SE) layer (23), a channel-wise gating mechanism, to enhance the representational power of the model.

The input of the model is batches of style embeddings $\mathbf{X} \in \mathbb{R}^{B\times C\times L}$ with the shape of (B, C, L), obtained from the GST model. Here, B, C, and L denote the batch size, the number of channels(1 in our case), and the temporal dimension (length of the embedding) of the input, respectively. Each encoder block first outputs the intermediate feature map from level l (denoted as $\mathbf{x}_l^{enc}$ where $\mathbf{x}_l^{enc} \in \mathbf{xs} = [\mathbf{x}_1^{enc}, \ldots, \mathbf{x}_l^{enc}]$ and $l = 1, 2, 3, \ldots, N-1$) for the hierarchical reconstruction loss calculation. Each block is followed by a residual condition layer to predict further the mean (denoted as $\boldsymbol{\mu_l}$) and variance (denoted as $\boldsymbol{\sigma_l}^2$) independently. The reparameterization for each mean and variance within the batch is conducted using:

$$\mathbf{z}_l = \boldsymbol{\mu}_l + \boldsymbol{\epsilon}_l \cdot \exp\left(\frac{1}{2}\log(\boldsymbol{\sigma}_l^2)\right) \tag{8}$$

to obtain the latent variables $\mathbf{Z}$. During the decoding process, with a similar structure as the encoder, intermediate decoded feature maps (denoted as $\mathbf{x}\hat{_{N-l}^{dec}}$ where $\mathbf{x}\hat{_{N-l}^{dec}} \in \hat{\mathbf{xs}} = [\mathbf{x}\hat{_1^{dec}}, \ldots, \mathbf{x}\hat{_{N-l}^{dec}}]$) are stored. Each block is followed by condition layers to obtain the decoded mean and variance, as well as incorporate additional residual KL losses. These layers work in combination to condition the latent variable z on both the previous layer's latent variable and the deterministic feature maps produced by the decoder. Section 4.2.1 explains the hierarchical losses obtained by modeling residual normal distributions within the VAE framework.

To address long-range dependencies, a challenge in deep generative models, each block is augmented with a residual connection consisting of 3 layers of 1D convolutions,

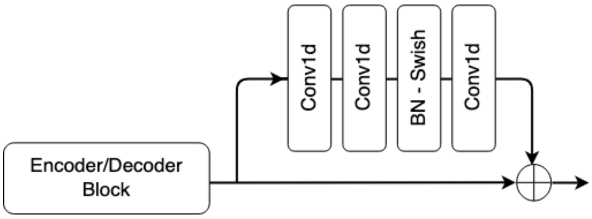


Figure 6: Residual Cells in Encoder and Decoder

Note: The figure depicts the residual cells within the encoder and decoder structures. Each cell incorporates a scaling factor of 0.1 applied to the output of the sequential convolutional and activation operations, enhancing the learning stability by weighting the contribution of the residual connection.

batch normalization, and Swish activation. Figure 6 shows the architecture of each encoder and decoder block. Residual connections help mitigate issues like vanishing gradients, especially in KL loss optimization, by allowing direct paths for gradients during backpropagation.

Feature Combination: In the decoding process, we follow similar tricks as the NVAE framework to combine the sampled z_l with deterministic feature maps, serving as the info learned by the decoder during the generation process as the input to the next level to improve the information flow. The concatenation is applied long temporal dimension, and the integration is formalized at each layer l of the decoder as follows:

$$h_l = Dec\_Block_l(Res\_Block_l(h_{l-1} \oplus z_l)), \tag{9}$$

where $\oplus$ denotes the concatenation operation, h_{l-1} represents the deterministic feature map from the previous layer, and Dec_Block_l and Res_Block_l indicates the upsampling block and residual block operation respectively at the l-th layer. We expect each layer to draw upon the randomness inherent in the samples and adapt to consistent, learned features, leading to a more stable and effective generative process. We initialize the feature map as zero tensor, and it is updated as the decoding progresses through different levels.

Squeeze and Excitation: In adapting the SE layer to accommodate our single-channel data, we have implemented a modification focusing on recalibration across the temporal dimension. It is realized through two-phase processes: a squeeze and an excitation phase. In the squeeze phase, we apply global average pooling along the channel dimension of the input hidden state tensor $H \in \mathbb{R}^{B\times C\times L}$, transforming its shape from (B, C, L) to (B, L). The squeeze operation for each batch b and temporal location l can be represented as:

$$S_{b,l} = \frac{1}{C}\sum_{c=1}^{C} H_{b,c,l}, \tag{10}$$

where $S \in \mathbb{R}^{B \times L}$ is the output of the squeeze phase. Next, the excitation module consists of two fully connected layers with ReLU and Sigmoid activation functions in the excitation phase, as Figure 4 (c) shows. The first linear layer in the excitation module first processes the temporally compressed representation, reducing the input dimensionality based on a pre-defined reduction ratio, denoted as r. The second linear layer then expands back to the original temporal dimension. Let $\text{FC}_1 : \mathbb{R}^L \to \mathbb{R}^{L/r}$ and $\text{FC}_2 : \mathbb{R}^{L/r} \to \mathbb{R}^L$ be the two fully connected layers, where r is the predefined reduction ratio. The excitation operation can be defined as:

$$E = \sigma(\text{FC}_2(\delta(\text{FC}_1(S)))), \tag{11}$$

resulting in $E \in \mathbb{R}^{B \times L}$, where δ denotes the ReLU activation function and σ represents the Sigmoid activation function. The final output tensor Y, which maintains the shape (B, C, L), is obtained by applying the excitation output to the original input tensor:

$$Y_{b,c,l} = H_{b,c,l} \cdot E_{b,l}, \tag{12}$$

for $b = 1, \ldots, B$; $c = 1$; and $l = 1, \ldots, L$.

4.2 Training Stability in Hierarchical VAE

We implement several techniques to enhance training stability and extract more features from 1-dimensional embedding data in our hierarchical VAE setup. Firstly, we incorporate residual KL losses by modeling the residual normal distribution. This approach involves sampling from a distribution at one level of the hierarchy and then, at the subsequent level, constructing a normal distribution centered around that sample, repeated throughout the hierarchy. Additionally, we employ hierarchical reconstruction and KL losses by integrating the losses from each level into the overall optimization process. During the initial experimental phase, we experimented with spectral normalization, utilizing Lipschitz constant regularization as detailed in (15), applied to the residual cells along the temporal dimension of the data. While spectral normalization helps to keep the latent variables predicted by the encoder within a bounded range, thereby stabilizing the model's response to input variations, we observed that in our specific context, it led to over-generalization in the output. This over-generalization resulted in a reduced variety within the learned embeddings, leading us to ultimately exclude spectral normalization from our model.

4.2.1 Residual Normal Distribution

In VAEs, a challenge lies in balancing the two objectives of the training process, particularly in managing the unbounded KL divergence between the distributions $q(z_l \mid x, z_{<l})$ and $p(z_l \mid z_{<l})$. This divergence is a critical component of our objective function, but its unconstrained nature can lead to gradient explosion, adversely affecting the optimization process. To mitigate this issue, we adapt the residual distribution approach to parameterize $q(z \mid x)$ relative to $p(z)$ as mentioned in the original paper. Specifically, we define the distribution of each latent variable z_{li} within the layer z_l as a normal distribution, $p(z_{li} \mid z_{<l}) := \mathcal{N}(\mu_i(z_{<l}), \sigma_i(z_{<l}))$, based on the preceding layers $z_{<l}$. For the approximate posterior, we employ $q(z_{li} \mid z_{<l}, x) := \mathcal{N}(\mu_i(z_{<l}) + \Delta\mu_i(z_{<l}, x), \sigma_i(z_{<l}) \cdot \Delta\sigma_i(z_{<l}, x))$. For levels other than the last level ($l < N$), the KL divergence is calculated using this approach13, allowing the model to capture more subtle variations in the data by learning adjustments (Δ) relative to a stable baseline distribution (the prior), rather than learning the entire distribution outright.

$$\mathrm{KL}_{l<N}(q(z^i \mid x) \parallel p(z^j)) = \frac{1}{2}\sum_i \left(\frac{(\Delta\mu_i)^2}{\sigma_i^2} + \Delta\sigma_i^2 - \log(\Delta\sigma_i^2) - 1 \right), \tag{13}$$

, where $\Delta\mu_i$ and $\Delta\sigma_i$ are the deviations in the mean and variance, respectively, of the distribution $q(z_{li} \mid z_{<l}, x)$ from those of the prior $p(z_{li} \mid z_{<l})$.

$$\mathrm{KL}(q(\mathbf{z_1} \mid \mathbf{x}) \parallel p(\mathbf{z_1})) = -\frac{1}{2}(1 + \log(\boldsymbol{\sigma}^2) - \boldsymbol{\mu}^2 - \boldsymbol{\sigma}^2), \tag{14}$$

We take the KL losses from Equation 13 and the standard KL term14, calculated between the reparameterized $\mathbf{z_1}$ from the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$ outputted from the encoder and the Gaussian distribution $q(z \mid x)$, as our hierarchical KL loss, denoted as $\mathcal{L}_{\text{VAE}}$. The reparameterization in upper-level loss terms stabilizes the training by preventing dramatic shifts in the distribution parameters. It ensures a more efficient encoding of information, enhancing the model's ability to learn complex hierarchical representations.

4.2.2 Hierarchical Losses

The optimization process in VAEs typically involves two fundamental loss components: the KL divergence and the reconstruction loss. The KL divergence principally aligns the latent space distribution $\mathbf{z}$ with a Gaussian distribution, a standard objective in VAE architectures as mentioned in section4.2.1, where we introduce the hierarchical and residual relationship in KL losses from different levels. In our utterance embedding function framework comprising N levels, we extend this hierarchical concept by incorporating hierarchical reconstruction losses. This dual approach is suggested to further enhance the mapping accuracy between successive blocks or levels within the intermediate output from the encoder and their corresponding reconstructed feature maps. We denote the output of the encoder at the l^{th} level as $\mathbf{x}_l^{enc}$ and their corresponding decoded output as $\mathbf{x}_{N-l}^{\hat{dec}}$ across each hierarchical level l where $l = 1, 2, 3, \ldots, N-1$. Both standard reconstruction loss between input embedding (denoted as $\mathbf{x}$) and predicted embedding (denoted as $\hat{\mathbf{x}}$) and intermediate reconstruction losses at level l are calculated using a composite function, a combination of L1 loss, cosine similarity, and mean squared error (MSE). Their corresponding weights denoted as w_{L_1}, w_{Cos}, and w_{MSE}, respectively. Equation 15 shows how one loss is calculated in one level. In our setup, we give different weights (w_{Rec} and w_{KL}) to standard losses (ℓ_0^{rec} and ℓ_0^{kl}) and intermediate losses ($w_l^{rec/kl}$, $\ell_l^{rec/kl}$) in Equation 16.

$$\begin{aligned}\text{Recon}(\mathbf{x}_l^{enc}, \mathbf{x}_{N-l}^{\hat{dec}}) = {} & w_{L_1} \cdot \text{L}_1(\mathbf{x}_l^{enc}, \mathbf{x}_{N-l}^{\hat{dec}}) \\ & + w_{\text{Cos}} \cdot \text{Cos}(\mathbf{x}_l^{enc}, \mathbf{x}_{N-l}^{\hat{dec}}) + w_{\text{MSE}} \cdot \text{MSE}(\mathbf{x}_l^{enc}, \mathbf{x}_{N-l}^{\hat{dec}}),\end{aligned} \tag{15}$$

$$\mathcal{L}_{\text{Recon}} = \sum_{l=1}^{N-1} w_l^{rec} \cdot \text{Recon}(\mathbf{x}_l^{enc}, \mathbf{x}_{N-l}^{\hat{dec}}) + w_{\text{Rec}} \cdot \text{Recon}(\mathbf{x}, \hat{\mathbf{x}}). \tag{16}$$

The total loss for the model, denoted as $\mathcal{L}_{\text{VAE}}$, is the summation of the weighted hierarchical reconstruction loss $\mathcal{L}_{\text{Recon}}$ and KL loss $\mathcal{L}_{\text{KL}}$:

$$\mathcal{L}_{\text{KL}} = w_{\text{KL}} \cdot \text{KL}(q(\mathbf{z_1} \mid \mathbf{x}) \parallel p(\mathbf{z_1})) + \sum_{l=1}^{N} w_l^{kl} \cdot \text{KL}_{l<N}(q(z^i \mid x) \parallel p(z^j)), \tag{17}$$

$$\mathcal{L}_{\text{VAE}} = \mathcal{L}_{\text{Recon}} + \mathcal{L}_{\text{KL}}. \tag{18}$$

4.3 Speaking Style Control Mechanism

The decoder of our HVAE model is structured hierarchically with latent variables represented at each level, z_l, where l denotes the level index. As the prior $p(z)$ is an isotropic distribution, which contains no information, our control mechanism is built in other latent spaces between the conversion. The control mechanism alters the latent variables sampled from other non-prior spaces in each decoder level, assuming that these latent variables capture different features of the speech output.

Initialization and Parameter Setup: A random Gaussian noise vector $\mathbf{z}$ is initialized as the starting point:

$$\mathbf{z} \in \mathbb{R}^{1\times1\times16} \sim \mathcal{N}(0, I), \tag{19}$$

where $\mathcal{N}(0, I)$ denotes a multivariate normal distribution with zero mean and identity covariance matrix. In our architecture, the dimensionality of the latent variable $\mathbf{z}$ is structured as a three-dimensional tensor, whose dimensions correspond to batch size, number of channels, and feature space equal to 16 resulting from the downsampling process the encoder. In our control mechanism, several parameters are introduced:

- $freeze\_level$: This parameter specifies the level in the decoder at which the latent variables are frozen ('fixed' mode). In our case, the controllable levels can iterate from 0 to 4, representing the dimensional structure of the latent space at each hierarchy of the decoder: level $= \{24, 32, 40, 48, 56\}$.
- m_N: The specified parameters demarcate discrete dimensions within the latent variables at $freeze\_level$. m_N spans a range from 0 to $N - 1$, where N is a constituent of the set level $= \{24, 32, 40, 48, 56\}$, contingent upon the pre-established $freeze\_level$.
- α: Refer to as the control step size, this parameter determines the magnitude of change applied to the controlled dimensions (m_N) of the latent variables. For example, if $m_N = 0$, then the 0^{th} component of that latent variable will be adjusted by $\pm\alpha$. A smaller α allows for finer, more subtle adjustments, while a larger α leads to more significant changes. This parameter is crucial for tuning the sensitivity of the control mechanism.
- $window_m$: This parameter defines the window size to control the scope, which refers to the number of dimensions affected during manipulation. For example, setting $window_m = 3$ and $m_N = 5$ implies that the modification, characterized by adding α to dimension 5 (m_N), extends to the elements of the latent variable corresponding to dimensions 4, 5, and 6.

Control Process: The control mechanism within our decoder navigates different levels and dimensions during the decoding process, allowing for detailed manipulation of each or windowed component(s) in latent variables during inference. Each control run starts when a random noise z is initialized, and the modification is done with fixed feature maps by disabling the sampling during the inference. The control process at each decoder level l is defined as follows:

$$\mathbf{Z}_l^{\text{ctrl}} = \begin{cases} f(z_l), & \text{if } l \neq l_{\text{freeze}}, \\ f(z_l)_{\Delta_m} \pm \alpha, & \text{if } l = l_{\text{freeze}}, \end{cases} \tag{20}$$

where $f(\cdot)$ denotes the reparameterization function applied to the latent variables. This function is defined as:

$$f(\mu, \log \sigma^2, c) = \mu + \exp\left(\frac{\log \sigma^2}{2}\right) \cdot c, \tag{21}$$

where μ and $\log \sigma^2$ represent the mean and log-variance of the latent variables, respectively. The term c, denoting the constant prior, is a predetermined constant vector derived from a predefined prior distribution. This ensures deterministic output for a given random input in the 'fix' mode for manipulation. Variables at levels other than l_{freeze} remain constant during the inference process, maintaining stability in other aspects of the generated output. Variables at level l_{freeze} are dynamically modified based on the adjustment Δ_m by α, representing changes to a specific dimension of z_l.

Real-Time Interaction: During control, we capture keyword inputs to enable interactive manipulation of the model's output by varying parameters in dimensions at the freeze level. This process involves a continuous loop of interaction, where new samples are generated using the same z but with a delay time of 2 seconds setting for modifications. With the predefined α, the changes can be made in real-time, including level adjustment, modification of a single dimension, modification within a windowed dimension, and reset to the base state, which is the state without any modifications.

5 Experiments

5.1 Experimental Setup

5.1.1 Datasets

The speech datasets we use to build our speaker embedding training dataset include subsets of the following datasets: LJ Speech Dataset (25), LibriTTS-R (32), Multilingual LibriSpeech (MLS) Dataset (42), GigaSpeech Dataset (7), Emotional Voices Database (EmoV-DB) (1), Emotional Speech Dataset (65), ADEPT Dataset (57), RyanSpeech Dataset (64), Blizzard Challenge 2013 (29), Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)(40), and Voice Cloning Toolkit (VCTK) (60) from Centre for Speech Technology, differing in speaking styles, fidelity, microphone features, and languages. Table 1 shows the features of each dataset and the amount of training data included. All speech data is resampled to 16kHz, and then we normalize it by adjusting the amplitude of the signal by its decibel levels, standardizing the loudness across different speech data. We utilize the toolkit from Silero Voice Activity Detector (VAD)(56) to remove the silence by detecting the non-speech segments from the speech based on the timestamps of speech segments. In converting audio to Mel-spectrograms, we apply the Short-Time Fourier Transform (STFT) from librosa, with the frame length of 1024, hop length of 256, and a raised cosine ('hann') window function. The Mel filter bank is within the frequency bounds between 40 Hz and 8000 Hz. We use the logarithmic transformation to get the output log-Mel spectrogram.

These Mel-spectrograms are converted into speaker embeddings utilizing the GST model. The total amount of our training data includes 1151017 embeddings extracted from approximately 974 hours of audio.

5.1.2 Model Configuration

We train the VAE utterance embedding model with 6 hierarchical levels using the 64-dimensional embeddings, and the model compresses this information into a 16-dimensional latent space.

5.1.3 Loss Balancing

To stabilize the training and balance between the reconstruction loss and KL loss in our model, we employ a combination of Beta-annealing and cyclic annealing strategies on our

Dataset	Duration	Feature	Type
LJ Speech	≈ 24 hrs	Single speaker	TTS
LibriTTS-R	≈ 480 hrs	Multi-speaker	TTS with various styles and accents
MLS	≈ 107 hrs	Multilingual, Multi-speaker	ASR
Gigaspeech	≈ 212 hrs	Multilingual, Multi-speaker	Conversations, Interviews, and other spoken content
EmoV-DB	≈ 80 hrs	Multi-speaker	Emotion
ESD	< 1 hr	Multi-speaker, Multilingual	Emotion
ADEPT	≈ 1 hr	Multi-speaker	TTS with prosodic variations
RAVDESS	≈ 24 hrs	Multi-speaker	Emotion
VCTK	≈ 43 hrs	Multi-speaker	TTS with various English accents

Table 1: Datasets for Baseline VAE Model.

Note: This table provides an overview of datasets for training the baseline VAE model. It details the types of content each dataset offers, ranging from TTS and automatic speech recognition (ASR) to emotional speech and conversations in various languages. The 'Feature' column describes whether the dataset includes a single speaker or multiple speakers and notes the presence of emotional content or cross-lingual data. 'Type' characterizes the primary use case of the dataset.

weighted reconstruction and KL terms 22 across different hierarchies, given the definition of the loss function defined in Equations 16 and 17.

$$\text{weighted losses} = \begin{cases} \mathcal{L}_{\text{Recon}} = \sum_{l=1}^{5} w_l^{rec} \times \ell_l^{rec} + (w_{\text{Rec}} \times \ell_0^{rec}), \\ \mathcal{L}_{\text{KL}} = (w_{\text{Kl}} \times \ell_0^{kl}) + \sum_{l=1}^{4} w_l^{kl} \times \ell_l^{kl} + (w_5^{kl} \times \ell_5^{kl}), \end{cases} \tag{22}$$

, which can be simplified as $\mathcal{L}_{\text{KL}} = [(w_{\text{KL}} \times \ell_0^{kl}), (w_1^{kl} \times \ell_1^{kl}), (w_2^{kl} \times \ell_2^{kl}), (w_3^{kl} \times \ell_3^{kl}), (w_4^{kl} \times \ell_4^{kl}), (w_5^{kl} \times \ell_5^{kl})]$ and $\mathcal{L}_{\text{Recon}} = [(w_{\text{Rec}} \times \ell_0^{rec}), (w_1^{rec} \times \ell_1^{rec}), (w_2^{rec} \times \ell_2^{rec}), (w_3^{rec} \times \ell_3^{rec}), (w_4^{rec} \times \ell_4^{rec}), (w_5^{rec} \times \ell_5^{rec})]$ with the corresponding dimension $\text{dim}_{kl} = [16, 24, 32, 40, 48, 56]$, $\text{dim}_{rec} = [64, 56, 48, 40, 32, 24]$. During our experiments, in the case of reconstruction losses, the last reconstruction term ℓ_0^{rec} is the most difficult to optimize, as it corresponds to the reconstruction between the input embedding. On the other hand, the first KL term ℓ_0^{kl} and the last KL term ℓ_5^{kl} are rather difficult to reduce. We weight the losses in different levels by setting: $w_{\text{Rec}} = 5.5, w_l^{rec} = 1.0, w_l^{kl} = 0.8, w_{\text{Kl}} = 1.0, w_5^{kl} = 10$. The Beta-annealing technique is applied to the KL loss component due to the KL vanishing issue as the following:

$$\beta = \begin{cases} \beta_{\text{initial}}, & \text{if } S_{current} < S_0 \\ \beta_{\text{initial}} + \left(\frac{S_{current}}{S_{epoch}}\right) \times (\beta_{\text{final}} - \beta_{\text{initial}}), & \text{otherwise} \end{cases} \tag{23}$$

, where $S_{current}$, S_{epoch} and S_0 represent the current step, the number of steps in each epoch, and the number of steps where β stays zero at the beginning of each epoch, respectively. Initially, the annealing factor β is set to $\beta_{\text{initial}} = 0.0000001$. It remains constant until the number of training steps exceeds $S_0 = 100$ within one epoch. Beyond this point, beta gradually increases linearly from $\beta_{\text{initial}} = 0.0000001$ to $\beta_{\text{final}} = 0.0001$, which helps in warming up KL loss and smoothly transitioning the model's focus from reconstruction at the early stages to a balanced emphasis on both terms. Figure 7 shows the cyclic schedule with an example of 7 epochs.

5.1.4 Training and Evaluation of VAE Model

Traning: The VAE utterance embedding model is trained using a single NVIDIA GeForce GTX TITAN X GPU, with a batch size of 512 for 5000 epochs. We use the Adamax optimizer with default beta coefficients, $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and the LambdaLR scheduler defined using a lambda function where the learning rate γ for each epoch is defined as $\gamma = \gamma_0 \times \alpha^{\text{epoch}}$, where $\gamma_0 = 0.01$ represents the initial learning, and $\alpha = 0.8$ denotes the scheduler rate that controls the rate of decay. The scheduler implies an exponential decrease in the learning rate with each epoch. We apply the gradient clipping with the max norm of 1.0 and the early stopping with the patience of 5 epochs. During the training, we discovered that the deepest KL loss from the last level has high instability, often manifesting as gradient explosion, which poses a challenge in achieving a balanced training dynamic. Figure 7 compares the sum of KL losses from all levels and the one excluding the last level. However, the trade-off of the deepest-level KL loss plays a crucial role in stabilizing the KL losses at the other levels, allowing the steady decrease and mitigating the KL vanishing issue. In hierarchical models, deeper levels often capture more abstract and complex features of the data, however this is different in our setup, where the first-level KL loss is the most essential and standard, as it ensures that it aligns well with the assumed prior distribution. As a result, we adopt a strategy that, while acknowledging the deepest-level KL loss's role in providing regularization and indirect guidance to other levels, intentionally de-emphasizes its optimization to prevent it from dominating the training process.

Evaluation: In our evaluation framework, we evaluate our base VAE model by comparing the differences between the distribution of our input embeddings and the distribution obtained from the decoded embeddings over the latent space by taking 1000 data points from both. Given that these embeddings are in a high-dimensional space, we employ the dimensionality reduction using the t-Distributed Stochastic Neighbor Embedding (t-SNE) technique to obtain a 2-dimensional representation of our embeddings to enable

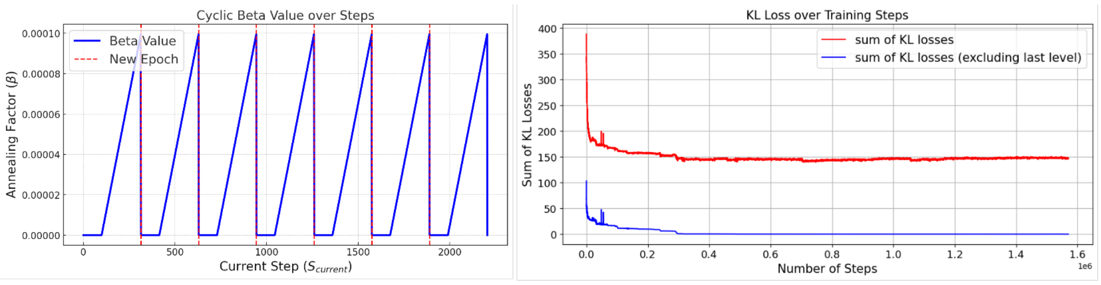


Figure 7: Cyclic-Beta Annealing and KL Losses in Training

Note: The left plot illustrates the cyclic beta values applied over training steps, indicating the annealing strategy employed to regulate the contribution of the reconstruction loss. The right plot provides the sum of KL losses over the training steps, distinguishing between the total KL loss and the sum excluding the final level. This delineation allows observation of the training stability and the effect of excluding the last level's contribution to the overall loss.

the visualization. The distributions are visualized in two ways: through a scatter plot and a kernel density estimation (KDE) plot. The scatter plot visually differentiates between the generated and real samples using distinct colors, visually representing how closely the VAE-generated embeddings align with those taken from the dataset. Furthermore, we also include kernel density estimation (KDE), as KDE is a non-parametric method used to estimate the probability density function of a random variable. In our analysis, we apply KDE to both axes of the 2D embeddings for both predicted and real samples. This is achieved by fitting a Gaussian kernel to the data, which provides a smooth density estimate. We set the bandwidth to 0.1, giving us a more detailed curve that closely follows the data. Then, we plot these densities for both the x and y axes, using different colors to distinguish between the predicted and real data. As KDE plots offer deeper insights into the distributional properties of the embeddings, they give a bit more information on the consistency of the x and y-dimensional mapping.

5.2 Experiments on Speaking Style Control

In this section, we first introduce the general control setup, standard pre-trained models, and objective metrics employed for evaluation in both single-feature and multi-feature control setups. Subsequently, we explain our investigation of the independence between controlled features within the multi-feature control setup and explore the scalability in feature modulation depth achieved through varying control step sizes, assessing how changes in alpha values influence the extent of feature alteration and if it is in a consistent direction.

5.2.1 General Control Setupt and Metrics

In our experiment, we focus on the control of the following acoustic features: pitch, pitch range, duration, energy, and spectral tilt, as the sentence-wise control in these five intuitive prosodic features is stated to be efficient enough to generate a wide variety of speaking styles(43). We calculate the average pitch and the pitch range using Yin pitch estimation algorithm(11). For the five features, as described in Section 4.3, we set the control parameters with $freeze\_level = 4$, $\alpha = 0.5$, $m_N = x$, and $window_m = 3$, and apply both positive and negative adjustments using α by 3 steps. Here, x means the linked dimension of the component controlling the target features. Beginning from the base latent state, denoted as z_B representing the state without any α modifications, we adjust the compoment at m_N dimension in z_B by adding it with α for three consecutive steps, leading to $z_B^{m_N} + 3\alpha$. Subsequently, we revert to the base z_B and then substracting it by α for another three steps, resulting in $z_B^{m_N} - 3\alpha$. This process results in a total of 7 seven different states for each sample, ranging from $z_B^{m_N} - 3\alpha$ to $z_B^{m_N} + 3\alpha$, through the unmodified base state z_B. After being decoded and synthesized, the outputs are initially structured as $S = [(S_0), S_1, S_2, S_3, S_4, S_5, S_6, S_7]$, where S_0 represents the initial output decoded from the base state without any control. Positive adjustments are applied first, resulting in S_1 to S_3 gradually, and S_4 is the manually reset state where we switch to negative adjustments. To be more intuitive, we restructure the outputs as $S = [S_3, S_2, S_1, S_{reset}, S_5, S_6, S_7]$. We evaluate the effectiveness of our manipulations by calculating the average values for the seven states and determining the average differences between these characteristics for the six consecutive state pairs.

Besides the five acoustic features, we additionally measure the possible gender switches and the timing of the transitions at a higher level. We utilize the audio classification model finetuned from XLM-R for Speech(5), a Facebook AI's large-scale multilingual pre-trained model, on Librispeech-clean-100 for gender recognition. In this experiment, we resample the synthesized audios to 16000Hz due to the model configuration, and we adopt a setup using $freeze\_level = 4$, $\alpha = 0.5$, and $m_N = 12$. Adjustments are made to z_B^{12} by modifying it with α for six consecutive steps in both positive and negative directions, resulting in a range of $z_B^{12} \pm 6\alpha$ to enable more precise evaluation on the gender switch.

Duration, Energy, and Pitch: For each i-th audio file, we calculate the duration (denoted as D_i), energy (denoted as E_i), and pitch (denoted as P_i). The duration D_i is computed as the audio file length divided by the sampling rate. The energy E_i is calculated as the average RMS energy across all frames. For the average pitch, we take the mean from the voiced array where each element corresponds to the estimated pitch (in Hz) for a given frame. The following shows the formulas for the three:

$$D_i = \frac{L_i}{sr}, \qquad E_i = \frac{1}{N}\sum_{n=1}^{N} \text{RMS}(f_{i,n}), \qquad P_i = \begin{cases} \frac{1}{V}\sum_{n=1}^{V} p_n & \text{if } V > 0 \\ 0 & \text{otherwise} \end{cases}$$

where L_i is the length (number of samples) of the i-th file, and sr is the sampling rate (set at 240000 Hz). In the case of RMS, $f_{i,n}$ is the n-th frame of the i-th file, p_n represents the pitch of the n-th frame that is voiced, N is the total number of frames in the file; and V is the total number of voiced frames in the i-th audio file, determined by filtering the pitch estimates to include only positive values. The difference in these three features between consecutive files is calculated as $\Delta D_i = |D_i - D_{i-1}|$ for the duration, $\Delta E_i = |E_i - E_{i-1}|$ for energy, and $\Delta P_i = |P_i - P_{i-1}|$ for pitch. These calculations provide the duration difference ΔD_i, the energy difference ΔE_i, and the pitch difference ΔP_i between each pair of consecutive files. The overall average and the average difference between consecutive files for each feature are calculated using the following formulas:

$$\bar{X} = \frac{1}{7}\sum_{i=1}^{7} X_i, \qquad \bar{\Delta X} = \frac{1}{6}\sum_{i=2}^{7} \Delta X_i$$

where X represents either the duration (D), energy (E), or pitch (P) of the files. $\bar{X}$ denotes the average value of X over the seven inputs, and $\bar{\Delta X}$ represents the average difference in X between each pair of consecutive files. These analyses provide insights into the variations in duration, energy, and pitch across the controlled audio samples, offering information to determine if the changes are significant.

Pitch Range: For each i-th audio file, the pitch range (denoted as R_i) is calculated by taking the difference between the minimum and maximum pitch values within the voiced segments as:

$$R_i = \max(p_{voiced,i}) - \min(p_{voiced,i}), \qquad \bar{R} = \frac{1}{M}\sum_{i=1}^{M} R_i$$

where $p_{voiced,i}$ represents the array of pitch estimations for all voiced frames within that file (similar to how we calculated P_i earlier). We calculate the average pitch range, denoted as $\bar{R}$, across 7 audio files in the dataset.

Spectral Tilt: For each i-th audio file, the spectral tilt (denoted as T_i) is evaluated to characterize the slope of the log power spectrum concerning frequency and to look into spectral energy distribution across frequency components within the audio signal. We calculate it by using linear regression on the log power spectrum against the frequency bins. The formula for calculating the spectral tilt is given as:

$$T_i = \text{LinearRegression}(f_i, \log(P_i)), \qquad \bar{T} = \frac{1}{7}\sum_{i=1}^{7} T_i$$

Here, f_i represents the frequency bins, and $\log(P_i)$ denotes the logarithm of the power spectrum for the i-th file. This procedure is executed for each audio file, outputting individual spectral tilt values. Additionally, the average spectral tilt (denoted as $\bar{T}$) is computed across the seven audio files. The average difference in spectral tilt (denoted as ΔT_i) between consecutive files is also evaluated, following a similar approach to the calculations of ΔD_i, ΔE_i, and ΔP_i for the duration, energy, and pitch respectively. Other than comparing T_i and $\bar{T}$, we also evaluate it by plotting each audio file's spectral centroid over time. As the spectral centroid represents the 'center of mass' of the power spectrum, providing a measure of the brightness or sharpness of a sound, we think it is relevant to see the possible differences in microphone features.

Gender: To obtain more reliable information on the timing of the gender switch, we analyze 13 consecutive audio segments. For each i-th audio, we extract the audio features and feed them into the specified pre-trained gender detection model and evaluate mainly the number and timing of gender switches. Let S_i represent the binary gender classification (0 for "female" and 1 for "male") for the i-th audio segment. We analyze the sequence $S = [(S_0), S_1, S_2, S_3, S_4, S_5, S_6, S_7, S_8, S_9, S_{10}, S_{11}, S_{12}, S_{13}]$, where S_0 and S_7 are the original baseline conditions. After S_0 is automatically generated, we first conduct a sequence of adjustments by $+\alpha$, then reset the condition back to the baseline (S_7), and subsequently perform a negative control. A gender switch is identified when $S_i \neq S_{i+1}$ and $i \neq 6$. The switching time (ST) is calculated by counting the number of gender classification changes across the sequence, excluding the transition from S_6 to S_7. For this experiment, we repeat the process 10 times by randomly sampling 10 latent variables with the same parameter setting, resulting in 10 sets of 13 audio segments. We calculate the average of the switching points (Avg$_{SP}$) and switching times (Avg$_{ST}$) across the 10 runs.

5.2.2 Multi-Feature Control Setup

In the multi-feature control setup, we configure the parameters as follows: $freeze\_level = 4$, $\alpha = 1.0$, and for multiple features $m_{Ni} = x_i$, where we increase α from 0.5 to 1.0 in order to record more obvious changes in the multi-feature setup. In this setup, given a base latent state z_B, we manipulate the feature dimensions corresponding to the target paired features, both individually and in combination. This manipulation is achieved by alternating between the level and dimension, facilitated by our real-time interaction mechanism.

5.2.3 Scalabililty Control Setup

To measure the scalability of feature changes in response to varying values of the control parameter α, where we aim to assess whether larger or smaller values of α correspondingly result in more substantial or minor alterations in the features. To this end, the experiment is structured using a single feature control setup with two distinct values of α, $\alpha = 0.5$ and $\alpha = 1.0$. As the random seeds differ, this is quantified by only observing the average difference $\bar{\Delta X}$ between the controlled inputs when applying different α values.

5.2.4 Scalability Control Setup

To measure the scalability of feature changes in response to different values of the control parameter α, the investigation is conducted using a single feature control setup, employing two distinct α values: $\alpha = 0.5$ and $\alpha = 1.0$. In this way, we aim to know whether larger or smaller values of α correspondingly result in more substantial or minor alterations in the features. Given that different random seeds are used and we have different initial inputs z to the VAE model, a direct comparison of average values across different inputs is not meaningful. Instead, the focus is placed on analyzing the average difference $\bar{\Delta X}$ between the controlled inputs under varying α settings and keeping the other control parameters identical. This approach allows for assessing the effect of α on the degree of feature change, independent of the initial state variations.

6 Experimental Result

In this section, we first present the results of our HVAE base model's performance, followed by its capability to control speaking style. This includes an assessment of the general effectiveness of control, its consistency and scalability, as well as the degree of disentanglement achieved.

6.1 HVAE Basemodel Evaluation

Our evaluation of the HVAE base model's performance employs a visualization approach, as shown in Figures 9 and 10. Figure 9 presents a 2D scatter plot representation of the input and predicted distribution formed by the sampled embeddings, while Figure 10 portrays the kernel density estimation of these embeddings across various training epochs. The model reaches convergence after 1500 epochs, and in most of our experiment runs, the distribution along the y-axis appears to converge faster than along the x-axis. Figure 8 shows the loss curves along the training process, with the left plot showing the sum of reconstruction losses and the right plot displaying the sum of KL losses from the first five levels, excluding the last level. The reconstruction loss has a sharp decline in the initial epochs, followed by a plateau, and the KL loss plot, on the other hand, highlights the model's learning of the efficient latent space representation, as evidenced by the rapid initial decrease and subsequent stabilization. At the end of our training, the two losses are:

$$\begin{aligned}\mathcal{L}_{\text{Recon}} :& [1.669e-06, 4.247e-14, 0.192, 0.157, 0.023, 0.081],\\ \mathcal{L}_{\text{KL}} :& [2.384e-07, 1.490e-07, 0.097, 0.0002, 0.007, 146.863]\end{aligned}$$

During the experimental phase, we tried to calculate the Jensen-Shannon (JS) distance between the two distributions by taking the high-dimensional embeddings and the 2D ones. We discovered that the non-overlapping distributions in specific dimensions led to infinite average distance values. This occurs even when distributions are close; as long as one is not entirely encompassed by another, the metric becomes infinite. In our hierarchical setup, which complicates distribution modeling due to the combination of hierarchical losses considered, using JS distance gives us relatively little information about how the distributions look.

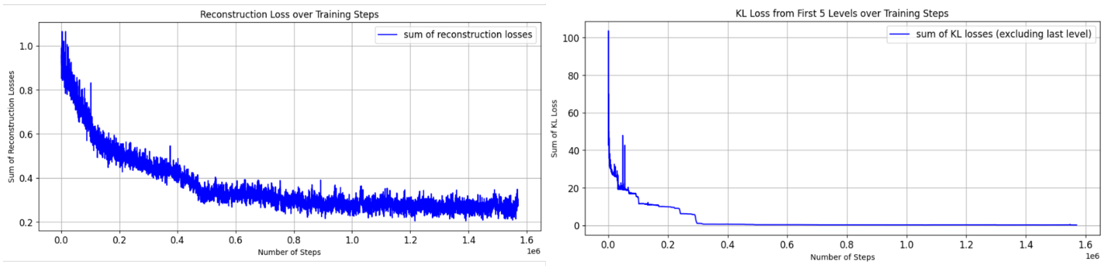


Figure 8: The Reconstruction Loss and KL Loss

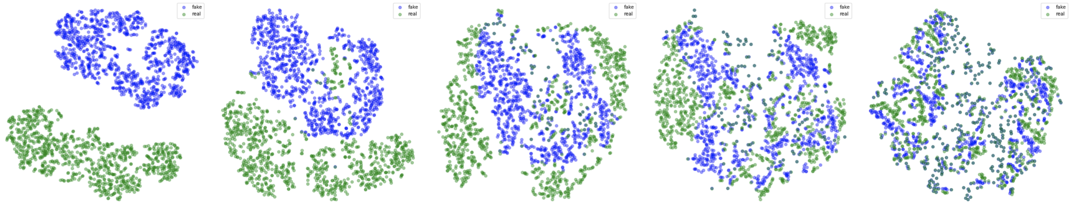


Figure 9: Evolution of HVAE Base model: 2D Scatter Plot Representation.

Note: This scatter plot shows 1000 input (green) and 1000 reconstructed (blue) embeddings. It is arranged chronologically from epochs 0 to 1200, with each subplot representing a distinct 300-epoch stage.

6.2 Speaking Style Control

6.2.1 Control Consistency

Throughout our experiments, we observed a consistent control mechanism across various aspects, including the dimensionality, control direction, and the effect magnitude of control step size. First, we found that the dimensionality remains constant for different randomly initialized inputs, where the selected dimensions influence the intended acoustic features, regardless of the initial state or different parameters. Moreover, the direction of influence exerted by the selected controlling factors also exhibited uniformity. For instance, if adding α within the first dimension consistently decreases the duration across one sample, this directional influence remains identical across all samples and different α sizes. Lastly, the effect magnitude of α on the controlled features is the same given the same controlling factor. During the preliminary stages of our experiment, we explored a broad range of α values, from 0.1 to 2.0, across different features. However, for analytical simplicity and to avoid the complexities introduced by an overabundance of variables, we standardized the same α value for all features during analysis. From Table 2, we can see that, even with this standardization, different features have varying degrees of sensitivity to larger control steps. Detailed insights are further discussed in Section 6.2.3.

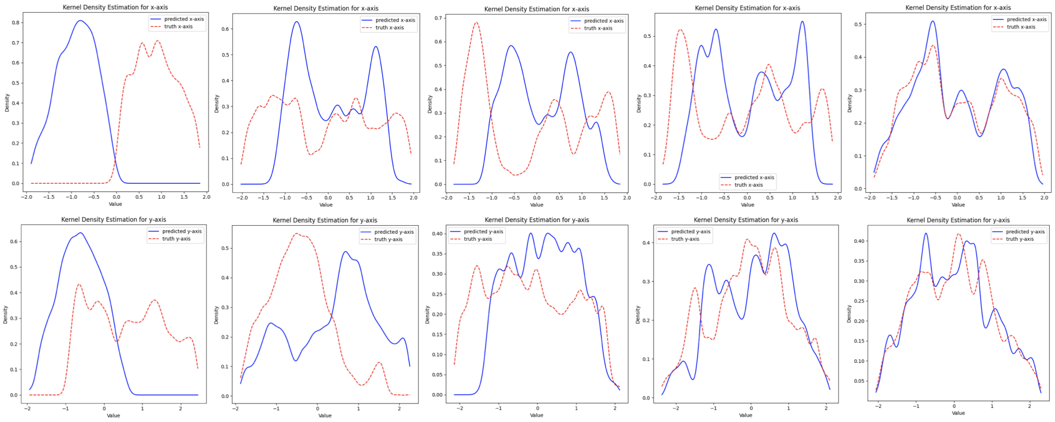


Figure 10: Evolution of HVAE Base model: Kernel Density Estimation Representation

Note: The figure shows kernel density estimations of speaker embeddings, with x-axis distributions in the top row and y-axis distributions in the bottom row. The sequence of plots, ordered from left to right, represents the evolution from epoch 0 to epoch 1200 in 300-epoch increments, contrasting the predicted embeddings (blue solid lines) with the actual embeddings (red dashed lines).

6.2.2 Controlled Features

From a hierarchical level perspective, modifications done at the highest hierarchy level ($dim = 56$) have the most significant control over the five target features. From a dimensional perspective fixed at the last level, most of the five features can be individually controlled through single-dimensional adjustments, showing the independence among neighboring dimensions. For example, at this level, while dimension 5 is responsible for controlling pitch, its adjacent dimensions, dimension 4 and dimension 6, exhibit distinct behaviors: dimension 4 shows no substantial influence, acting as a neutral element, and dimension 6 is involved in controlling the unstable noise level. Among all the target features, the pitch range is controlled by the components at dimensions 15 to 17 using the windowed control. Table 2 gives an overview of the latent variable's controlling factor (dimension) and the metrics, including average value $\bar{X}$, the average difference between the seven controlled inputs $\bar{\Delta X}$, and the interval when $\alpha = 0.5$.

Pitch: Our experiments have identified that multiple components across various levels and dimensions influence pitch control. However, many exhibit inconsistent behaviors and limited scalability. Our experiments indicate that the 5^{th} and the dimensions spanning 7^{th} to 9^{th} in the final level of the latent variable exhibit the highest stability. Among these options, the 7^{th} to 9^{th} dimensions demonstrate stable control in only one direction—either consistently increasing or decreasing pitch with the addition of α, but not inversely. Consequently, the 5^{th} dimension has been designated as the principal factor for pitch control. There is a positive relationship between pitch values and α adjustments: positive

Feature	Index	Control Step: $\alpha = 0.5/\alpha = 1.0$
pitch (Hz)	avg.	($\alpha = 0.5$) 126.51
(dim=6)	avg. gap	($\alpha = 0.5$) 11.99 ($\alpha = 1.0$) 14.21
	interval	[171.09, 155.64, 137.62, 116.34, 101.94, 102.56, 100.42]
pitch range (Hz)	avg.	($\alpha = 0.5$) 76.870
(dim=16 $\sim$ 18)	avg. gap	($\alpha = 0.5$) 7.934 ($\alpha = 1.0$) 8.02
	interval	[44.160, 69.658, 71.010, 83.350, 87.912, 90.239, 91.762]
duration (seconds)	avg.	($\alpha = 0.5$) 1.682
(dim=2)	avg. gap	($\alpha = 0.5$) 0.117 ($\alpha = 1.0$) 0.274
	interval	[1.440, 1.472, 1.504, 1.536, 1.728, 1.952, 2.144]
energy (norm. units)	avg.	($\alpha = 0.5$) 0.058
(dim=17)	avg. gap	($\alpha = 0.5$) 0.019 ($\alpha = 1.0$) 0.092
	interval	[0.136, 0.100, 0.0619, 0.0401, 0.028, 0.021, 0.021]
spectral tilt (dB/octave)	avg.	($\alpha = 0.5$) -0.00064
(dim=25)	avg. gap	($\alpha = 0.5$) 3.79e-05 ($\alpha = 1.0$) 1.04e-04
	interval	[-6.16e-4, -6.48e-4, -5.89e-4, -6.80e-4, -6.61e-4, -6.49e-4, -6.33e-4]
gender	Avg$_{SP}$	($\alpha = 0.5$) 9.1 ($\alpha = 1.0$) 7.9
(dim=13)	Avg$_{ST}$	($\alpha = 0.5$) 0.9 ($\alpha = 1.0$) 1.0

Table 2: Result of Acoustic Feature Control

α adjustments lead to an increase in pitch, whereas negative α adjustments result in a decrease. The graph in Figure 11 illustrates these pitch variations, where the spectral representation provides a visual confirmation of the pitch modulation in response to the α adjustments.

Duration: Duration is controlled by the first component in the latent variable at the last level and has a clear response to changes in the control parameter α. Table 2 shows that an increase in the value of α correlates with a reduction in the duration output, whereas a decrease in α increases the length. Figure 11 suggests that the model exhibits greater sensitivity to negative adjustments than positive ones. As for the scalability, when α is increased from 0.5 to 1.0, there is a notable augmentation in the average gap of the duration feature, underscoring the model's ability to modulate feature duration effectively.

Energy: The energy feature is primarily governed by the 17^{th} component of the latent variable. While adjustments in other dimensions also result in some changes in energy, their controls have less stability and consistency compared to the 17^{th} dimension. As illustrated in Table 12, the RMS energy of the controlled elements tends to increase with positive adjustments in α, indicating a greater sensitivity to positive rather than negative changes.

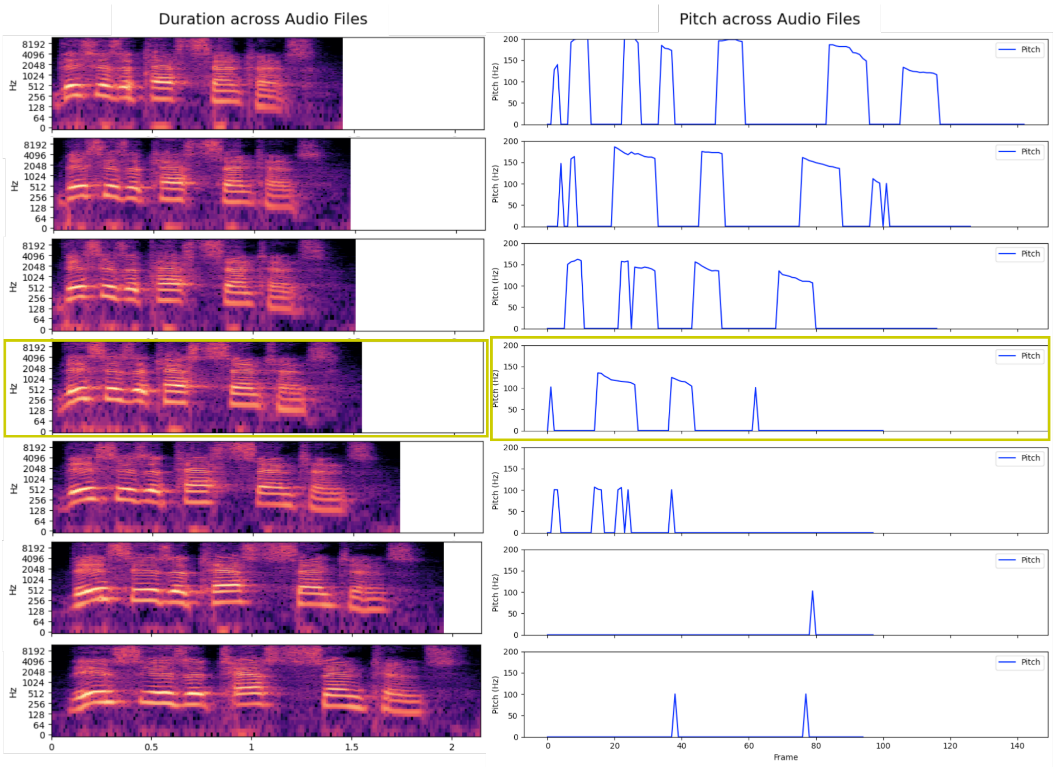


Figure 11: Variations in Duration and Pitch

Note: The left plot is a spectrogram displaying the frequency content over time, with the x-axis showing time in seconds and the y-axis indicating frequency bins on a logarithmic scale. The right plot is the fundamental frequency of the audio signals, with the x-axis representing sequential frames and the y-axis displaying pitch in Hertz (Hz), ranging from 0 Hz to 200 Hz. Yellow-marked plots denote the base state.

However, as α is varied from 0.5 to 1.0, the average differential in energy levels shows only a marginal change, suggesting a nonlinear response to control parameter scaling. Figure 12 gives an example of the differences.

Spectral Tilt: Spectral tilt is controlled by the 24^{th} dimension in the latent variable. It demonstrates a positive correlation with the adjustment of α, where an increase in α leads to a rising trend, suggesting a tilt toward higher frequencies. In Figure 12, we can see that as the positive adjustment is applied, the plot has more frequent peaks, indicating the moments where the sound has higher frequency content and is assumed to be perceived as brighter. Conversely, a decrease in α yields smoother output fluctuations, with lower centroid values, denoting a darker or more mellow sound quality. The average spectral tilt $\bar{T}$ across all seven samples is -0.00064, indicating a general bias toward lower-frequency energy in the audio spectrum. And the slight average difference $\Delta\bar{T}$ between consecutive samples' tilts shows that the changes in spectral tilt, controlled by $\alpha = 0.5$, are relatively subtle.

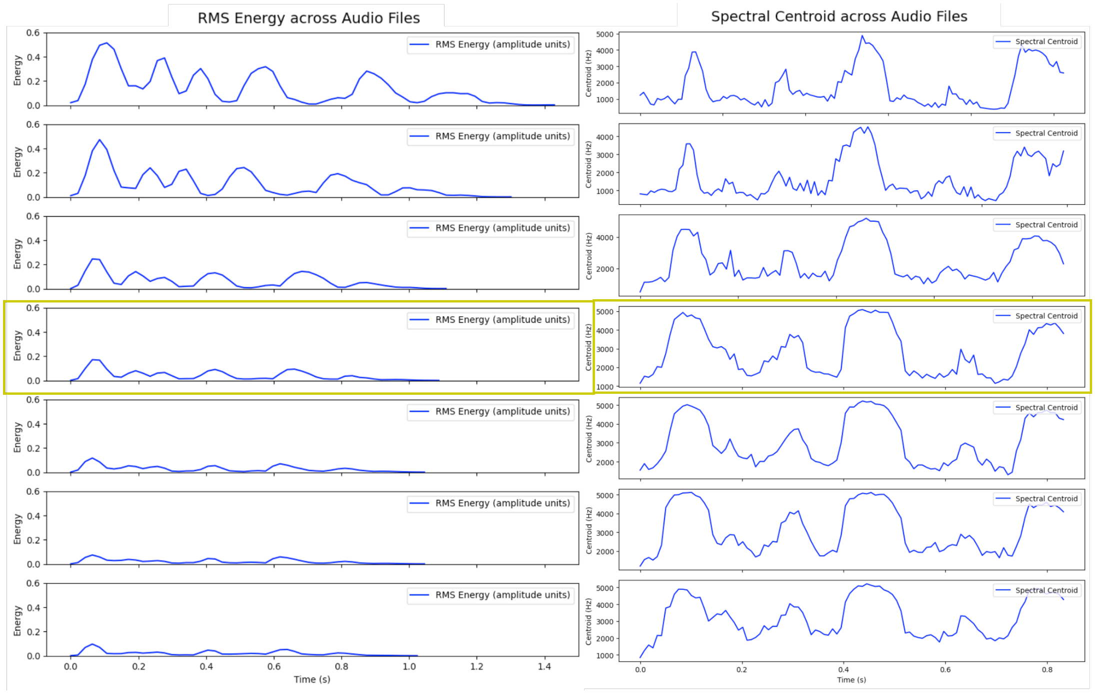


Figure 12: Varinations in RMS Energy and Spectral Centroid

Note: The left plot depicts the Root Mean Square (RMS) Energy of audio signals, measured in amplitude units, with the x-axis representing time in seconds (s) and the y-axis indicating energy levels. The right plot is the Spectral Centroid measurements for the corresponding audio files, where the x-axis denotes the number of frames over time, and the y-axis displays the centroid frequency in Hertz (Hz), extending from 0 Hz to 5000 Hz. Yellow-marked plots denote the base state.

Gender: The 12^{th} component in the latent variable at the last level is the most significant factor controlling gender and fundamental frequency. A positive α adjustment leads to a more feminine direction change, while a negative α adjustment shifts the voice towards more masculine characteristics. The results from the 10 experiments show that the average switching point (denoted as Avg_{SP}) is 9.1, and the average number of switches (denoted as Avg_{ST}) is 0.9. Figure 13 illustrates one of the runs where the switching point occurs between S_9 and S_{10}, indicating a change in gender from female to male. Typically, a female voice has a higher fundamental frequency than a male voice. We mark in spectrograms using horizontal lines (harmonics) spacing to represent the F0 and its multiples: closer spacing signifies a lower pitch, while wider spacing signifies a higher pitch. Additionally, formant frequencies, particularly the first formant (F1) and the second formant (F2) are often higher in female voices.

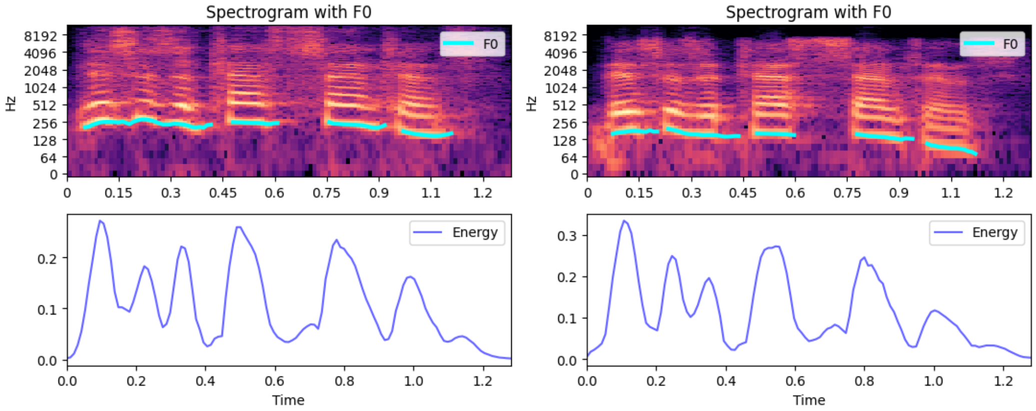


Figure 13: Spectrogram and Energy Contour Illustration of Gender Transition

6.2.3 Feature Scalability

In our analysis, we measure the average difference, $\Delta\bar{X}$, which represents the average change in feature X between each pair of consecutive files that differ in the amount of α added or subtracted. As shown in Table 2, all six features, including pitch, pitch range, duration, energy, spectral tilt, and gender, have varying degrees of difference in $\Delta\bar{X}$ when controlled using $\alpha = 0.5$ and $\alpha = 1.0$. Pitch and pitch range have a relatively high degree of scalability, that there is a notable increase in $\Delta\bar{X}$ when α increases. On the contrary, gender shows the least variability, suggesting lower scalability in response to changes in α. It is assumed that gender switch is higher level and contains more factors than the five intuitive prosodic features. The choice to limit the control to three positive and three negative steps is based on our observations that, with $\alpha = 1.0$ and $\alpha = 0.5$, further steps do not induce significant changes in the features. Additionally, excessive adjustment steps can sometimes introduce noise into the output, particularly in less stable dimensions under control.

6.2.4 Feature Disentanglement

Figure 3 presents four sets of multi-feature controls that we found to be stable and informative. In these experiments, the interaction between duration and other prosodic features, such as pitch or energy, demonstrates significant stability, suggesting a higher degree of independence for duration in our model's control mechanism. This finding aligns with the intrinsic properties of these acoustic features, where duration appears to be less correlated with pitch or energy, especially when juxtaposed with the correlation observed between pitch and gender. In the second control set, which focuses on the control

MultiFeature Control: $\alpha = 1.0$		
Duration (seconds, dim=2) vs. Energy (norm. units, dim=17)		
Set 1	Duration ↑	Duration ↓
Energy ↑	$D_i = 1.732, E_i = 0.088$	$D_i = 1.505, E_i = 0.092$
Energy ↓	$D_i = 1.810, E_i = 0.071$	$D_i = 1.603, E_i = 0.073$
Duration (seconds, dim=2) vs. Pitch (Hz, dim=6)		
Set 2	Duration ↑	Duration ↓
Pitch ↑	$D_i = 1.582, P_i = 163.552$	$D_i = 1.210, P_i = 156.040$
Pitch ↓	$D_i = 1.591, P_i = 142.012$	$D_i = 1.388, P_i = 138.263$
Duration (seconds, dim=2) vs. Spectral Tilt (dB/octave, dim=25)		
Set 3	Duration ↑	Duration ↓
Spectral Tilt ↑	$D_i = 2.089, T_i = -0.00072$	$D_i = 1.600, T_i = -0.00069$
Spectral Tilt ↓	$D_i = 1.881, T_i = -0.00077$	$D_i = 1.603, T_i = -0.00076$
Pitch (Hz, dim=6) vs. Energy (norm. units, dim=17)		
Set 4	Pitch ↑	Pitch ↓
Energy ↑	$P_i = 211.562, E_i = 0.149$	$P_i = 189.105, E_i = 0.152$
Energy ↓	$P_i = 214.002, E_i = 0.100$	$P_i = 183.720, E_i = 0.105$

Table 3: Comprehensive MultiFeature Control

over duration and pitch, we observe that adjustments to duration have a dominant effect over pitch. When we adjust pitch, the resulting changes in duration are relatively minor, indicating that modifications in pitch have a limited impact on duration. This demonstrates that duration maintains robust consistency in our model's framework, highlighting its relative independence and effectiveness in maintaining temporal aspects of speech.

In set 3, where duration and spectral tilt are controlled, the spectral tilt (T_i) remains relatively stable despite substantial changes in duration. While the changes in spectral tilt are subtle, they still exhibit a degree of independence and stability, reinforcing the model's capacity for nuanced control. Conversely, the control involving pitch and energy (Set 4) reveals a more intertwined relationship. Modification in pitch leads to corresponding shifts in energy levels, although these shifts are less significant compared to the direct impact on pitch. This interdependence suggests that pitch and energy are more closely related to acoustic features. Therefore, their simultaneous control presents more challenges and offers less independence than controlling duration in conjunction with either of these features.

7 Conclusions

Our work introduces a novel utterance embedding function adapted from hierarchical VAE to learn distributions in sequential data, such as speaker embeddings. Our function uses scaling convolutions with residual cells, and we customize the loss function by integrating the concept of residual distribution and symmetric mapping in feature maps. Regarding training stability and generalization, we combine the techniques of Beta-annealing and cyclic scheduling in weights in our hierarchical loss design to address the loss balancing and KL vanishing issues. In response to our research questions, our controllable utterance embedding function can capture the speaker embedding distribution and apply explicit controls to the synthesized speech. This control is achieved by manipulating variables in latent spaces that extend beyond the standard isotropic prior distribution. We found that the deepest level in our model learns the most stable and disentangling acoustic features, including pitch, pitch range, energy, duration, and spectral tilt. These features can typically be controlled via a single factor in the latent space, offering scalable control at varying intensities, and the controlling factors remain consistent across different random seeds. Our exploration of feature disentanglement revealed that multi-feature control is stable and exhibits independence between different features, especially when considering pairs of features. The result highlights that duration demonstrates a high degree of independence, particularly when controlled with features like pitch or energy. This independence is less pronounced when pitch and energy are controlled together, reflecting their closer relationship as acoustic features.

Overall, our results demonstrate the efficacy of hierarchical VAEs in enabling fine-grained and independent control of multiple speech characteristics, providing deeper insights into the correlation between the hierarchy of acoustic features and the hierarchical design in neural networks. This understanding and our experiments offer new possibilities for more expressive TTS applications.

8 Discussions and Future Work

The primary challenges encountered in this work fall into three categories: optimizing the information captured in speaker embeddings, managing the complexity of an over-engineered model framework, and developing an intuitive and efficient control mechanism. Initially, our efforts were focused on achieving convergence, mostly on addressing the KL vanishing issue and only learning partial distribution. In the beginning stage, we utilized 128-dimensional speaker embeddings with a 16-dimensional latent space, assuming that higher-dimensional representations would capture more information within a well-defined framework. While the 128-dimensional mode converged faster and required fewer loss-balancing techniques, it captured a narrower range of speaking styles than the 64-dimensional mode. Consequently, we shifted back to the 64-dimensional data, applying different techniques to balance the losses. This decision underscored the trade-offs between convergence speed, model complexity, and the richness of captured features. Lastly, the control setup in our work, although comprehensive, turned out to be non-intuitive and involved a lot of factors, making it extra-consuming to navigate through every level and variable across different dimensions and parameter combinations. This complexity led us to believe that there might be undiscovered features or relationships between neighboring levels or dimensions that have yet to be fully understood or utilized.

9 References

[1] Adaeze Adigwe, Noé Tits, Kevin El Haddad, Sarah Ostadabbas, and Thierry Dutoit. The emotional voices database: Towards controlling the emotion dimension in voice generation systems. *arXiv preprint arXiv:1806.09514*, 2018.

[2] Kei Akuzawa, Yusuke Iwasawa, and Yutaka Matsuo. Expressive speech synthesis via modeling expressions with variational autoencoder, 2019.

[3] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017. cite arxiv:1701.07875.

[4] Karl Johan Åström and Tore Hägglund. *Advanced PID Control.* ISA - The Instrumentation, Systems and Automation Society, 2006.

[5] Arun Babu, Changhan Wang, Andros Tjandra, Kushal Lakhotia, Qiantong Xu, Naman Goyal, Kritika Singh, Patrick von Platen, Yatharth Saraf, Juan Pino, Alexei Baevski, Alexis Conneau, and Michael Auli. Xls-r: Self-supervised cross-lingual speech representation learning at scale, 2021.

[6] Gautam Bhattacharya, Joao Monteiro, Jahangir Alam, and Patrick Kenny. Generative adversarial speaker embedding networks for domain robust end-to-end speaker verification, 2018.

[7] Guoguo Chen, Shuzhou Chai, Guanbo Wang, Jiayu Du, Wei-Qiang Zhang, Chao Weng, Dan Su, Daniel Povey, Jan Trmal, Junbo Zhang, Mingjie Jin, Sanjeev Khudanpur, Shinji Watanabe, Shuaijiang Zhao, Wei Zou, Xiangang Li, Xuchen Yao, Yongqing Wang, Yujun Wang, Zhao You, and Zhiyong Yan. Gigaspeech: An evolving, multi-domain asr corpus with 10,000 hours of transcribed audio, 2021.

[8] Jianfei Chen, Cheng Lu, Biqi Chenli, Jun Zhu, and Tian Tian. VFlow: More expressive generative flows with variational data augmentation. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1660–1669. PMLR, 13–18 Jul 2020.

[9] Seungwoo Choi, Seungju Han, Dongyoung Kim, and Sungjoo Ha. Attentron: Few-shot text-to-speech utilizing attention-based variable-length embedding, 2020.

[10] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling, 2014.

[11] Alain de Cheveigné and Hideki Kawahara. Yin, a fundamental frequency estimator for speech and music. *The Journal of the Acoustical Society of America*, 111 4:1917–30, 2002.

[12] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2017.

[13] Florian Eyben, Sabine Buchholz, Norbert Braunschweiler, Javier Latorre, Vincent Wan, Mark John Francis Gales, and Kate Knill. Unsupervised clustering of emotion and voice styles for expressive tts. *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4009–4012, 2012.

[14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

[15] Henry Gouk, Eibe Frank, Bernhard Pfahringer, and Michael J. Cree. Regularisation of neural networks by enforcing lipschitz continuity, 2020.

[16] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, and Ruoming Pang. Conformer: Convolution-augmented transformer for speech recognition, 2020.

[17] Zhifang Guo, Yichong Leng, Yihan Wu, Sheng Zhao, and Xu Tan. Prompttts: Controllable text-to-speech with text descriptions, 2022.

[18] Gustav Eje Henter, Jaime Lorenzo-Trueba, Xin Wang, and Junichi Yamagishi. Principles for learning controllable tts from annotated and latent variation. In *Interspeech*, 2017.

[19] Keikichi Hirose and Jianhua Tao. Speech prosody in speech synthesis: Modeling and generation of prosody for high quality and flexible speech synthesis. 2015.

[20] Jonathan Ho, Xi Chen, Aravind Srinivas, Yan Duan, and Pieter Abbeel. Flow++: Improving flow-based generative models with variational dequantization and architecture design, 2019.

[21] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models, 2020.

[22] Wei-Ning Hsu, Yu Zhang, Ron Weiss, Heiga Zen, Yonghui Wu, Yuan Cao, and Yuxuan Wang. Hierarchical generative modeling for controllable speech synthesis. In *International Conference on Learning Representations*, 2019.

[23] Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-excitation networks, 2019.

[24] Yuchen Hu, Chen Chen, Ruizhe Li, Qiushi Zhu, and Eng Siong Chng. Noise-aware speech enhancement using diffusion probabilistic model, 2023.

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

[26] Jaehyeon Kim, Sungwon Kim, Jungil Kong, and Sungroh Yoon. Glow-tts: A generative flow for text-to-speech via monotonic alignment search, 2020.

[27] Jaehyeon Kim, Jungil Kong, and Juhee Son. Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech. *CoRR*, abs/2106.06103, 2021.

[28] Minchan Kim, Sung Jun Cheon, Byoung Jin Choi, Jong Jin Kim, and Nam Soo Kim. Expressive text-to-speech using style tag. In *Interspeech 2021*. ISCA, August 2021.

[29] Simon King and Vasilis Karaiskos. The blizzard challenge 2013. 2014.

[30] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022.

[31] Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and Patrick van der Smagt. Learning hierarchical priors in vaes. In *Neural Information Processing Systems*, 2019.

[32] Yuma Koizumi, Heiga Zen, Shigeki Karita, Yifan Ding, Kohei Yatabe, Nobuyuki Morioka, Michiel Bacchiani, Yu Zhang, Wei Han, and Ankur Bapna. Libritts-r: A restored multi-speaker text-to-speech corpus, 2023.

[33] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae. Hifi-gan: Generative adversarial networks for efficient and high fidelity speech synthesis, 2020.

[34] Yoonhyung Lee, Jinhyeok Yang, and Kyomin Jung. Varianceflow: High-quality and controllable text-to-speech using variance information via normalizing flow. *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7477–7481, 2022.

[35] Younggun Lee and Taesu Kim. Robust and fine-grained prosody control of end-to-end speech synthesis, 2019.

[36] Younggun Lee, Azam Rabiee, and Soo-Young Lee. Emotional end-to-end neural speech synthesizer, 2017.

[37] Xiang Li, Changhe Song, Jingbei Li, Zhiyong Wu, Jia Jia, and Helen Meng. Towards multi-scale style control for expressive speech synthesis, 2021.

[38] Hyungseob Lim, Kyungguen Byun, Sunkuk Moon, and Erik Visser. Stylebook: Content-dependent speaking style modeling for any-to-any voice conversion using only speech data, 2023.

[39] Zhaoyu Liu and Brian Mak. Cross-lingual multi-speaker text-to-speech synthesis for voice cloning without using parallel corpus for unseen speakers, 2019.

[40] Steven R. Livingstone and Frank A. Russo. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. *PLOS ONE*, 13(5):1–35, 05 2018.

[41] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models, 2022.

[42] Vineel Pratap, Qiantong Xu, Anuroop Sriram, Gabriel Synnaeve, and Ronan Collobert. Mls: A large-scale multilingual dataset for speech research. *ArXiv*, abs/2012.03411, 2020.

[43] Tuomo Raitio, Ramya Rasipuram, and Dan Castellani. Controllable neural text-to-speech synthesis using intuitive prosodic features, 2020.

[44] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022.

[45] Yi Ren, Chenxu Hu, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech 2: Fast and high-quality end-to-end text to speech, 2022.

[46] Yi Ren, Jinglin Liu, and Zhou Zhao. Portaspeech: Portable and high-quality generative text-to-speech, 2022.

[47] Yi Ren, Yangjun Ruan, Xu Tan, Tao Qin, Sheng Zhao, Zhou Zhao, and Tie-Yan Liu. Fastspeech: Fast, robust and controllable text to speech, 2019.

[48] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows, 2016.

[49] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[50] Huajie Shao, Shuochao Yao, Dachun Sun, Aston Zhang, Shengzhong Liu, Dongxin Liu, Jun Wang, and Tarek Abdelzaher. ControlVAE: Controllable variational autoencoder. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8655–8664. PMLR, 13–18 Jul 2020.

[51] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. Interpreting the latent space of gans for semantic face editing, 2020.

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

[53] Guangzhi Sun, Yu Zhang, Ron J. Weiss, Yuan Cao, Heiga Zen, Andrew Rosenberg, Bhuvana Ramabhadran, and Yonghui Wu. Generating diverse and natural text-to-speech samples using a quantized fine-grained vae and auto-regressive prosody prior, 2020.

[54] Guangzhi Sun, Yu Zhang, Ron J. Weiss, Yuan Cao, Heiga Zen, and Yonghui Wu. Fully-hierarchical fine-grained prosody modeling for interpretable speech synthesis, 2020.

[55] Antti Suni, Daniel Aalto, Tuomo Raitio, Paavo Alku, and Martti Vainio. Wavelets for intonation modeling in HMM speech synthesis. In *Proc. 8th ISCA Workshop on Speech Synthesis (SSW 8)*, pages 285–290, 2013.

[56] Silero Team. Silero vad: pre-trained enterprise-grade voice activity detector (vad), number detector and language classifier. `https://github.com/snakers4/silero-vad`, 2021.

[57] Alexandra Torresquintero, Tian Huey Teh, Christopher G. R. Wallis, Marlene Staib, Devang S Ram Mohan, Vivian Hu, Lorenzo Foglianti, Jiameng Gao, and Simon King. Adept: A dataset for evaluating prosody transfer, 2021.

[58] Arash Vahdat and Jan Kautz. Nvae: A deep hierarchical variational autoencoder. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc.

[59] Rafael Valle, Kevin J. Shih, Ryan Prenger, and Bryan Catanzaro. Flowtron: an autoregressive flow-based generative network for text-to-speech synthesis. In *International Conference on Learning Representations*, 2021.

[60] Christophe Veaux, Junichi Yamagishi, and Kirsten MacDonald. Cstr vctk corpus: English multi-speaker corpus for cstr voice cloning toolkit. 2017.

[61] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, and Rif A. Saurous. Tacotron: Towards end-to-end speech synthesis, 2017.

[62] Yuxuan Wang, Daisy Stanton, Yu Zhang, RJ Skerry-Ryan, Eric Battenberg, Joel Shor, Ying Xiao, Fei Ren, Ye Jia, and Rif A. Saurous. Style tokens: Unsupervised style modeling, control and transfer in end-to-end speech synthesis, 2018.

[63] Dongchao Yang, Songxiang Liu, Rongjie Huang, Chao Weng, and Helen Meng. Instructtts: Modelling expressive tts in discrete latent space with natural language style prompt, 2023.

[64] Rohola Zandie, Mohammad H. Mahoor, Julia Madsen, and Eshrat S. Emamian. Ryanspeech: A corpus for conversational text-to-speech synthesis, 2021.

[65] Kun Zhou, Berrak Sisman, Rui Liu, and Haizhou Li. Seen and unseen emotional style transfer for voice conversion with a new emotional speech dataset. *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 920–924, 2020.