

Discrete Acoustic Space for an Efficient Sampling in Neural Text-To-Speech

Marek Strefec, Jonas Rohnke, Antonio Bonafonte, Mateusz Łajszczak, Trevor Wood

Amazon, Alexa AI

{strefecm, rohnj, bonafont, mateuszl, trevowoo}@amazon.com

Abstract

We present a Split Vector Quantized Variational Autoencoder (SVQ-VAE) architecture using a split vector quantizer for NTTS, as an enhancement to the well-known Variational Autoencoder (VAE) and Vector Quantized Variational Autoencoder (VQ-VAE) architectures. Compared to these previous architectures, our proposed model retains the benefits of using an utterance-level bottleneck, while keeping significant representation power and a discretized latent space small enough for efficient prediction from text. We train the model on recordings in the expressive task-oriented dialogues domain and show that SVQ-VAE achieves a statistically significant improvement in naturalness over the VAE and VQ-VAE models. Furthermore, we demonstrate that the SVQ-VAE latent acoustic space is predictable from text, reducing the gap between the standard constant vector synthesis and vocoded recordings by 32%.

Index Terms: prosody control, expressive speech synthesis, VQ-VAE, NTTS

1. Introduction

Generating speech for complex scenarios such as multi-turn dialogues requires modeling the rich prosodic aspects of speech to capture the natural variability of human conversation and assign the appropriate conversational style. Recent advances in TTS [1, 2] have allowed for the generation of high-quality speech but suffer from a lack of appropriate variation. As a result, the generated audio often sounds monotonous and unnatural.

One of the popular techniques for improving prosodic variation is based on extracting prosodic features from ground-truth data via Variational Auto-Encoders (VAE) [3, 4, 5, 6]. These methods use reference mel-spectrograms during training but have to provide the acoustic features from a different source during inference, as mel-spectrograms are unavailable. For example, Ezzerg et al. [7] computed a centroid (mean) over training data to represent the acoustic space. While using a constant vector provides a simple solution, this approach often results in monotonous speech. Akuzawa et al. [3] experimented with random sampling from the VAE’s prior, and Hodari et al. [8] proposed sampling focusing on low-probability regions of the prior distribution to produce speech with varied intonation. These methods introduce variation but can lead to unnatural speech due to the randomness in sampling. Karlapati et al. [9] introduced more sophisticated samplers that combine BERT and Graph samplers to predict Gaussian parameters from which they sample the final acoustic vector. Tyagi et al. [10] applied selection-based strategies to search a specific set of utterances for the most appropriate match based on syntactic structure.

All of these approaches, however, suffer from the fact that the underlying acoustic latent space often represents both high-level (e.g., general speaking style) and low-level (e.g., prosodic patterns) aspects of speech. This is not desirable for synthesis with constant acoustic vectors (e.g., centroids) because they are likely to encode noise and unique characteristics of particular utterances, resulting in instabilities and inconsistent style generation. Moreover, predicting such a multi-modal distribution

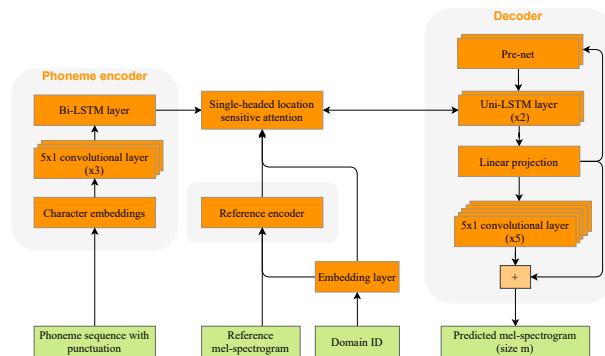


Figure 1: NTTS model architecture

from text is difficult.

To simplify the acoustic latent space for efficient sampling, we apply Vector Quantized-Variational AutoEncoder (VQ-VAE), which has been shown to learn a high-level abstract space of features [11, 12]. In this work, we take inspiration from [13] and present a modified version of VQ-VAE, called Split Vector Quantized Variational Autoencoder (SVQ-VAE), that partitions the latent vector into several splits that are each discretized independently. This allows the model to keep a smaller number of codes while increasing the number of information bits that can be encoded. This architecture gives the model significant representation power while keeping the discretized latent space small enough for efficient prediction from text.

While our work focuses on utterance-level prosody representations, more fine-grained representations have been proposed as well. A recent study by Hodari et al. [14] applies a two-stage training framework for prosody modeling, where word-level prosody representation is first learned separately and then predicted from text for TTS. Hono et al. [15] proposed a hierarchical approach that captures three different time resolutions (utterance-level, phrase-level, and word-level), where the prediction of each level is conditioned on latent variables from the coarser-level. Our work demonstrates that significant improvements can be achieved by using simpler utterance-level representations. Sun et al. [16] also used a hierarchical VAE model to disentangle prosodic features on different levels. However, their work concerns the control of the prosody style rather than predictability from context. Similarly, Sun et al. [17] investigates fine-grained architectures with auto-regressive prosody priors, but they focus on random sampling and diverse prosodic variations for ASR.

This work makes two contributions: 1) We present an SVQ-VAE model for TTS and show that it outperforms its VQ-VAE and VAE counterparts on constant-vector synthesis. 2) We demonstrate that the SVQ-VAE acoustic space can be predicted from BERT’s word-piece embeddings, reducing the gap between centroid synthesis and vocoded recordings by 32%.

2. Model

Our TTS system is based on a Tacotron2-like [1] architecture with an additional module for learning high-level speaking styles in an unsupervised manner. As shown in Figure 1, this model takes phonemes and domain ID as inputs, where domain ID is used to identify each of the pre-defined subsets of data (e.g., neutral, emotions, dialogues, etc.). The output of the model is acoustic features (mel-spectrograms), which are converted into audio waveforms using a vocoder [18, 1]. This section shows the details of our proposed model, comparing VAE, VQ-VAE, and SVQ-VAE reference encoders. Finally, in section 2.5, we describe the architecture to predict the latent codes from the input text.

2.1. Acoustic model

The acoustic model consists of the phoneme encoder, reference encoder, domain encoder, single-headed location-sensitive attention, and decoder.

In order to synthesize speech, the three encoders of the acoustic model first encode all available input features. The phoneme encoder uses stacked convolutional layers and a Bi-LSTM layer to encode the input phoneme sequence. The reference encoder [19] takes mel-spectrograms and applies a bottleneck to model utterance-level speech styles (see sections 2.2-2.4). The domain encoder uses an embedding lookup layer on a one-hot vector representing each of the pre-defined domains in our training dataset. The encoder outputs are concatenated and fed to an attention layer, which summarizes the entire encoded sequence as a fixed-length context vector for each decoder output step. The layer uses the location-sensitive attention mechanism from [20], which uses cumulative attention weights to prevent the model from skipping or repeating segments of speech.

Finally, the decoder predicts 5 frames at each decoding step. Each generated frame has a dimensionality of 80 and represents 50 ms of speech with an overlap of 12.5 ms.

2.2. VAE reference encoder

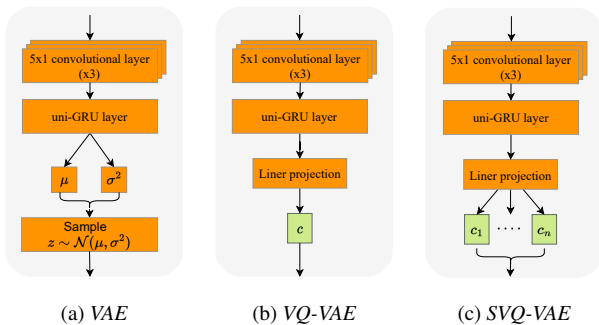


Figure 2: A detailed view into the three reference encoders used for centroid-based synthesis. The SVQ-VAE encoder was also used for experiments with predicted synthesis.

The baseline model uses a variational reference encoder, as shown in Figure 2a. During training, the encoder summarizes the mel-spectrogram input into a single 128-dimensional vector and predicts the mean and standard deviation of a Gaussian distribution μ and σ . We assume a prior distribution $p_\theta(z) = N(0, I)$, where θ represents the parameters of the reference encoder and z is the produced latent variable. The variable z is sampled using the reparameterization trick [21], and the model is trained to maximize the evidence lower bound (ELBO) [22]. Similar to [6], we apply KLD loss annealing and scheduling to avoid posterior collapse.

2.3. VQ-VAE reference encoder

While the continuous nature of the VAE latent space provides significant representation power, it also results in capturing properties of speech that generally do not need to be modeled with such high fidelity. Consequently, we use VQ-VAE (Figure 2b) to apply a strong bottleneck on these acoustic features.

As introduced by Oord et al. [11], the VQ-VAE encoder uses vector quantization in order to obtain a discrete latent representation of the continuous space. More specifically, our VQ-VAE encoder first summarizes the mel-spectrogram input into a single 128-dimensional vector. It then projects it into a codebook space C , which consists of K codebook vectors (or codes) c_1, c_2, \dots, c_K , each of dimensionality D . During the forward pass, VQ-VAE computes the L2-normalized distance between the encoder output and all codebook vectors, choosing the one with the minimum distance. Finally, this vector is passed to the decoder.

The choice of the codebook size K and the dimensionality of codes D have a major impact on the acoustic representation of the input. Small codebooks allow for an easy interpretation of codes and clear separation of styles, but we have experimentally observed that the overall speech quality is reduced. In our experiments, a codebook size of at least 512 is necessary to produce good quality speech. While the speech quality improves further as we increase the size of the codebook, using too many codes can affect generalization as there are only 57,975 utterances in our training dataset. In our internal evaluations, we achieved the best results with 8,192 codes, which is the value used for experiments in this paper.

In the context of VQ-VAE, a codebook collapse is the equivalent of a posterior collapse in VAE. It occurs when a portion of the codes become underutilized, causing the approximate posterior $q_\theta(z|x)$ to be supported by only a handful of codes. To address this problem, we use random restarts as Dhariwal et al. [23].

2.4. SVQ-VAE reference encoder

The empirical observations of the previous section regarding optimal codebook size can be better understood if we consider the information capacity required to learn useful audio representations. Let us assume we want to optimally represent global attributes of audio (e.g. pace, intonation, emotion) in a discrete VQ-VAE space, and that these attributes can be represented by a random variable with x_1, x_2, \dots, x_F features. If we assume that each feature has y_1, y_2, \dots, y_N possible values, in order to capture all combinations of these values in the VQ-VAE space, we need a codebook with F^N codes. Thus, in domains with high acoustic diversity, we need large codebooks for good coverage of possible speech variations. This presents several challenges. First, the VQ-VAE approach becomes impractical as the memory footprint gets too high for large codebooks. Second, a codebook with a large number of codes is more likely to result in a codebook collapse. Frequently used codes receive a stronger signal than other codes, leading to a further increase in their usage and, ultimately, only a small subset of codes being utilized. Lastly, increasing the number of codes can have a negative impact on the prediction of codes from text, increasing the total number of targets and thus making them more sparse.

In order to be able to encode more information while keeping the number of codes in a codebook reasonably low, we apply a split vector quantizer [24, 13] shown in Figure 2c. It encodes different parts of the latent vector individually, using separate codebooks. More specifically, our SVQ-VAE reference encoder uses S codebooks C_1, C_2, \dots, C_S . The output vector of the GRU is partitioned into S splits, and each split is passed into a dedicated codebook. Similar to the standard VQ-VAE, a code

is selected by performing a nearest-neighbor lookup, resulting in S selected codebook vectors. These vectors are concatenated to obtain the final acoustic representation of the input. Note that for $S = 1$, we get the original VQ-VAE reference encoder. Comparing VQ-VAE with SVQ-VAE, the former can encode only $\log_2 K$ information bits for a codebook size of K while the latter can encode $S \log_2 K$.

We conducted an extensive hyper-parameter search on our development dataset and achieved the best results with 8 splits, 1,024 codes per codebook, and codebook dimensionality of 8. Therefore, these parameters are used throughout all experiments presented.

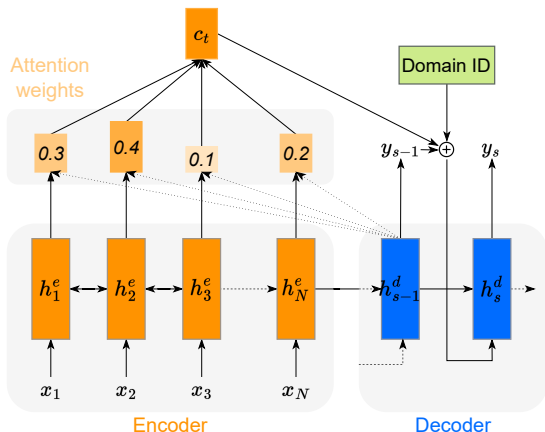


Figure 3: The proposed prediction model takes pre-trained BERT word embeddings (x_1, \dots, x_N) and domain ID as inputs and predicts a sequence of utterance-level codes (y_1, \dots, y_S).

2.4.1. Centroid code

Since oracle mel-spectrograms are not available for reference during inference time, a latent vector from the learned acoustic space must be chosen differently. A common approach for the VAE latent space is to use the *centroid* of a given domain by calculating the mean of all latent vectors corresponding to audio from that domain. This results in a good general representation of that domain. However, since the VQ-VAE and SVQ-VAE spaces are discrete, calculating the centroid in the same way is likely to result in a latent vector that is not represented in the codebooks and was, therefore, never observed during training. This can lead to instabilities and quality degradation. Instead, we selected a centroid code by calculating the mean of all domain vectors and performed a nearest-neighbor lookup.

2.5. Prediction of the acoustic space from the input text

In order to predict the acoustic latent space, we train the model depicted in Figure 3. Rather than using a sequence of phonemes as input, the model utilizes BERT’s word-piece embeddings that can capture the semantic and syntactic meaning of the text [25].

The word embeddings are consumed by the encoder’s bi-directional GRU, which further summarizes the input utterance and outputs a new representation of each word. The decoder is then conditioned on this representation to predict a code for each split y_s . y_1, y_2, \dots, y_S then altogether represent the sentence-level acoustic features of the utterance. At decoding step s , the model uses Bahdanau-style attention [26] to obtain alignment scores for each input word, representing how important each word is for the prediction of the code y_s , which corresponds to the s th split. These scores are then multiplied with the corresponding hidden states of the encoder’s bi-directional

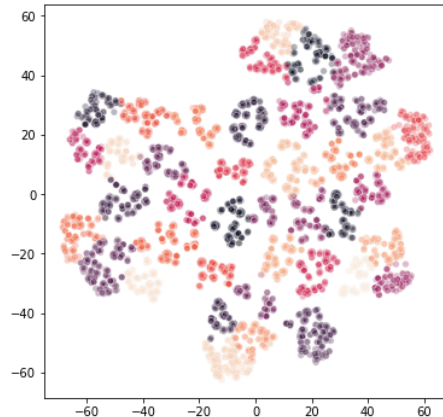


Figure 4: t -SNE plot of the SVQ acoustic latent space. Different colours show the result of k -means clustering for 40 clusters.

GRU to produce a context vector c_s . Finally, the domain ID, the context vector c_s , the prediction from the previous step y_{s-1} , and the previous decoder hidden step h_{s-1} are concatenated and fed into the decoder’s GRU to predict the following code y_s .

We hypothesize that autoregressive connections give the model more power to represent entangled audio features captured in codes. In addition, our experiments with non-autoregressive models often lead to a collapse, where the model predicts the same output for any input text. The Bahdanau-style attention is applied so that the model can focus on different parts of input for different splits.

2.5.1. K -means clustering

In order to train an SVQ-VAE reference encoder that is good at representing the audio, a reasonably large number of codes and splits has to be used. Indeed, training a model with fewer splits or codes causes a deterioration in the audio quality. However, this requirement may not be necessary for the prediction task. By reducing the number of prediction targets, we can significantly simplify the training objective while still improving the naturalness and expressivity of the generated speech. As shown in Figure 4, visualizing the latent space with t -SNE, the codebook space forms distinct clusters. This means that we can simplify the task to the prediction of cluster centroids, significantly reducing the number of prediction targets.

We applied the K -means clustering algorithm to each codebook split and set the number of centroids to 40 using the Elbow method [27]. Thus, we have 40 classification targets per split in our prediction task, and each cluster is represented by the closest codeword to the cluster mean.

While the k -means clustering step is not necessary for predicting the acoustic space, our internal evaluations show an improvement over the baseline model. Therefore, this approach is used in the final evaluations.

3. Experiments

In this section, we present two experiments. The first shows that the SVQ-VAE model outperforms VQ-VAE and VAE when using domain centroids for constant latent vector synthesis. The second experiment shows that the discretized space can be predicted from text, improving audio quality over synthesis with constant latent vectors. Both experiments are evaluated in the task-oriented dialogue domain.

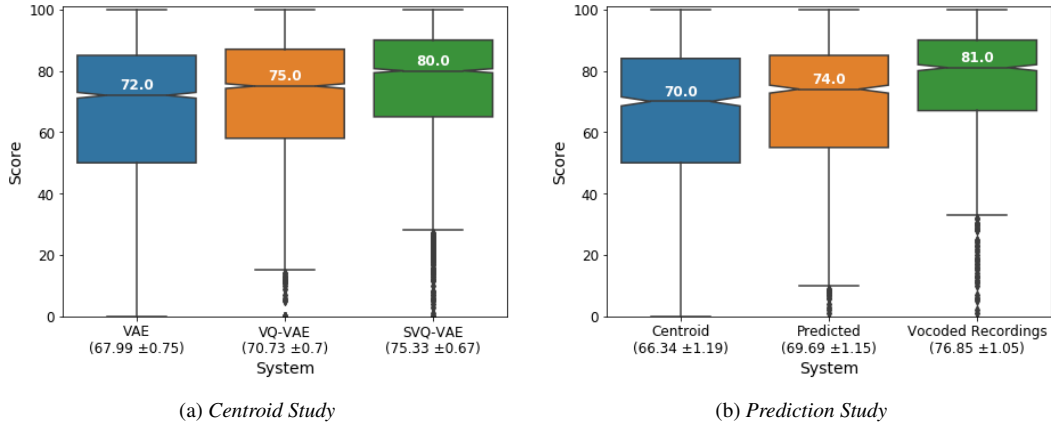


Figure 5: Results of the MUSHRA evaluations. Median scores are shown in the boxes, mean scores with 95% confidence interval range are shown underneath.

3.1. Voice data

To train the speech synthesis model, we recorded scripted dialogues in English from various task-oriented domains (ticket booking, recommender systems, food orderings, etc.). A professional speaker read the turns of the agent, and a second person read the user’s turns. By providing additional context to the professional speaker (i.e., including dialogue partner results), we obtained a more natural rendition of the recording script. We did not annotate the recording scripts with labels and instructed the speaker to vary their manner of speech in accordance with the context of the dialogue while maintaining a natural, conversational style. During the recordings, the speaker was also encouraged to modify the script to sound more conversational [28], resulting in additional contractions and the addition of filler words and pauses (e.g., *OK, so, well, Hmm*).

The number of recorded dialogues is 610, consisting of 4,796 agent turns, totaling ~5.5 hours of audio. We combined this data with additional recordings from the same professional speaker: ~24 hours of audio spoken in a neutral style in the general domain and ~50 hours of expressive recordings in a variety of domains (news reading, emotions, etc.).

3.2. Constant vector synthesis

The results of the first experiment show that the SVQ-VAE model outperforms the baseline models when all utterances in the test set are synthesized using the same constant latent vector. For the constant vector, we chose the domain centroid of the recorded dialogues as a general representation of the speaking style. We used a test set of utterances consisting of 31 dialogues with a total of 125 turns. The set was synthesized with the VAE, VQ-VAE, and SVQ-VAE models using their respective versions of the domain centroid (see section 2.4.1). In a MUSHRA evaluation, we asked US native expert listeners to ‘Please rate the systems in terms of their naturalness,’ targeting 30 ratings per utterance. We received a total of 3,375 ratings. The results in Figure 5a show that the SVQ-VAE model has a rating mean of 75.33, which is higher than both the VQ-VAE and VAE models, rated 70.73 and 67.99, respectively. Two-sided pairwise t-tests between all model pairs show statistical significance with p-values < 0.01.

3.3. Predicted vector synthesis

The results of the second experiment show that appropriate sampling from the latent space can benefit synthesis quality. However, deciding how to sample from the latent space for a given

utterance is a challenging problem. In this work, we predict the latent space only from the text of the current utterance, leaving more complicated methods (e.g., prediction from dialogue context) to be explored in future work.

Our test set consists of 13 task-oriented dialogues with a total of 75 turns. It includes scenarios with successful and unsuccessful interactions in which the voice is expected to show appropriate variations (e.g., positive when a recommendation is successful).

We synthesized the set with the SVQ-VAE model using the domain centroid code described in section 2.4.1 and with codes predicted from text using the model described in section 2.5. Finally, we evaluated the models in a MUSHRA test containing 3 systems: 1) SVQ-VAE using centroid; 2) SVQ-VAE with predicted codes; and 3) vocoded recordings as an upper anchor.

We asked US native expert listeners to ‘Please rate the systems in terms of their naturalness and expressivity,’ targeting 20 ratings per utterance. In this experiment, we are interested in the overall quality of speech which includes adjusting the expressivity to match the utterance text. Therefore, we asked the listeners to additionally consider expressivity to capture this effect in the evaluation. We received a total of 1,317 ratings.

The results in Figure 5b show that the model using predicted codes is preferred over the one using the centroid. It has a rating mean of 69.69 compared to 66.34, reducing the gap to the vocoded recordings by 32%. Two-sided pairwise t-tests between all model pairs show statistical significance with p-values < 0.01.

4. Conclusion

We introduced SVQ-VAE, a novel NTTS architecture that allows modeling a discrete space of global prosodic representations. We applied it to the challenging domain of task-oriented dialogues. We demonstrated that our architecture outperforms standard VAE and VQ-VAE baselines when inferring with a centroid. Additionally, we proposed an autoregressive predictor model that utilises contextualised representation from BERT embeddings to predict SVQ-VAE representations to further improve our system. Inferring with predicted representations outperformed centroid-based synthesis and reduced the gap to vocoded recordings by 32% in terms of average MUSHRA score. For future work, we plan to improve the prediction of the latent space by utilizing additional contextual features (e.g., dialogue embeddings), addressing the appropriateness of multi-turn dialogues rather than individual utterances.

5. References

- [1] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan *et al.*, “Natural TTS synthesis by conditioning wavenet on mel spectrogram predictions,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.
- [2] N. Prateek, M. Łajszczak, R. Barra-Chicote, T. Drugman, J. Lorenzo-Trueba, T. Merritt, S. Ronanki, and T. Wood, “In other news: a bi-style text-to-speech model for synthesizing newscaster voice with limited data,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, 2019, pp. 205–213.
- [3] K. Akuzawa, Y. Iwasawa, and Y. Matsuo, “Expressive speech synthesis via modeling expressions with variational autoencoder,” *Proc. Interspeech 2018*, pp. 3067–3071, 2018.
- [4] W.-N. Hsu, Y. Zhang, R. J. Weiss, H. Zen, Y. Wu, Y. Wang, Y. Cao, Y. Jia, Z. Chen, J. Shen *et al.*, “Hierarchical generative modeling for controllable speech synthesis,” *arXiv preprint arXiv:1810.07217*, 2018.
- [5] P. Karanasou, S. Karlapati, A. Moinet, A. Joly, A. Abbas, S. Slanzen, J. Lorenzo-Trueba, and T. Drugman, “A Learned Conditional Prior for the VAE Acoustic Space of a TTS System,” in *Proc. Interspeech 2021*, 2021, pp. 3620–3624.
- [6] Y.-J. Zhang, S. Pan, L. He, and Z.-H. Ling, “Learning latent representations for style control and transfer in end-to-end speech synthesis,” in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6945–6949.
- [7] A. Ezzerg, A. Gabrys, B. Putrycz, D. Korzekwa, D. Saez-Trigueros, D. McHardy, K. Pokora, J. Lachowicz, J. Lorenzo-Trueba, and V. Klimkov, “Enhancing audio quality for expressive neural text-to-speech,” *arXiv preprint arXiv:2108.06270*, 2021.
- [8] Z. Hodari, O. Watts, and S. King, “Using generative modelling to produce varied intonation for speech synthesis,” in *Proc. 10th ISCA Speech Synthesis Workshop*, 2019, pp. 239–244.
- [9] S. Karlapati, A. Abbas, Z. Hodari, A. Moinet, A. Joly, P. Karanasou, and T. Drugman, “Prosodic representation learning and contextual sampling for neural text-to-speech,” in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6573–6577.
- [10] S. Tyagi, M. Nicolis, J. Rohne, T. Drugman, and J. Lorenzo-Trueba, “Dynamic prosody generation for speech synthesis using linguistics-driven acoustic embedding selection,” *Proc. Interspeech 2020*, pp. 4407–4411, 2020.
- [11] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu, “Neural discrete representation learning,” *Advances in neural information processing systems*, vol. 30, 2017.
- [12] A. Razavi, A. van den Oord, and O. Vinyals, “Generating diverse high-fidelity images with VQ-VAE-2,” in *Advances in neural information processing systems*, 2019, pp. 14 866–14 876.
- [13] L. Kaiser, S. Bengio, A. Roy, A. Vaswani, N. Parmar, J. Uszkoreit, and N. Shazeer, “Fast decoding in sequence models using discrete latent variables,” in *International Conference on Machine Learning*. PMLR, 2018, pp. 2390–2399.
- [14] Z. Hodari, A. Moinet, S. Karlapati, J. Lorenzo-Trueba, T. Merritt, A. Joly, A. Abbas, P. Karanasou, and T. Drugman, “Camp: a two-stage approach to modelling prosody in context,” in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6578–6582.
- [15] Y. Hono, K. Tsuboi, K. Sawada, K. Hashimoto, K. Oura, Y. Nankaku, and K. Tokuda, “Hierarchical multi-grained generative model for expressive speech synthesis,” *Proc. Interspeech 2020*, pp. 3441–3445, 2020.
- [16] G. Sun, Y. Zhang, R. J. Weiss, Y. Cao, H. Zen, and Y. Wu, “Fully-hierarchical fine-grained prosody modeling for interpretable speech synthesis,” in *2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2020, pp. 6264–6268.
- [17] G. Sun, Y. Zhang, R. J. Weiss, Y. Cao, H. Zen, A. Rosenberg, B. Ramabhadran, and Y. Wu, “Generating diverse and natural text-to-speech samples using a quantized fine-grained VAE and autoregressive prosody prior,” in *2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6699–6703.
- [18] Y. Jiao, A. Gabryś, G. Tinchev, B. Putrycz, D. Korzekwa, and V. Klimkov, “Universal neural vocoding with parallel wavenet,” in *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 6044–6048.
- [19] R. Skerry-Ryan, E. Battenberg, Y. Xiao, Y. Wang, D. Stanton, J. Shor, R. Weiss, R. Clark, and R. A. Saurous, “Towards end-to-end prosody transfer for expressive speech synthesis with tacotron,” in *International Conference on machine Learning*. PMLR, 2018, pp. 4693–4702.
- [20] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” *Advances in neural information processing systems*, vol. 28, 2015.
- [21] D. P. Kingma and M. Welling, “Auto-encoding variational bayes,” *stat*, vol. 1050, p. 1, 2014.
- [22] D. P. Kingma, M. Welling *et al.*, “An introduction to variational autoencoders,” *Foundations and Trends® in Machine Learning*, vol. 12, no. 4, pp. 307–392, 2019.
- [23] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, “Jukebox: A generative model for music,” *arXiv preprint arXiv:2005.00341*, 2020.
- [24] K. K. Paliwal and B. S. Atal, “Efficient vector quantization of lpc parameters at 24 bits/frame,” *IEEE transactions on speech and audio processing*, vol. 1, no. 1, pp. 3–14, 1993.
- [25] A. Rogers, O. Kovaleva, and A. Rumshisky, “A primer in bertology: What we know about how bert works,” *Transactions of the Association for Computational Linguistics*, vol. 8, pp. 842–866, 2020.
- [26] D. Bahdanau, K. H. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” in *3rd International Conference on Learning Representations, ICLR*, 2015.
- [27] C. A. Sugar and G. M. James, “Finding the number of clusters in a dataset: An information-theoretic approach,” *Journal of the American Statistical Association*, vol. 98, no. 463, pp. 750–763, 2003.
- [28] N. Campbell, “Towards conversational speech synthesis; lessons learned from the expressive speech processing project,” in *Proc. 6th ISCA Speech Synthesis Workshop*, 2007, pp. 22–27.