A Vector Quantized Variational Autoencoder (VQ-VAE) Autoregressive Neural F0 Model for Statistical Parametric Speech Synthesis

Citation for published version:
Wang, X, Takaki, S, Yamagishi, J, King, S & Tokuda, K 2020, 'A Vector Quantized Variational Autoencoder (VQ-VAE) Autoregressive Neural F0 Model for Statistical Parametric Speech Synthesis', *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 28, pp. 157-170. https://doi.org/10.1109/TASLP.2019.2950099

Digital Object Identifier (DOI):
10.1109/TASLP.2019.2950099

Link:
Link to publication record in Edinburgh Research Explorer

Document Version:
Peer reviewed version

Published In:
IEEE/ACM Transactions on Audio, Speech and Language Processing

A Vector Quantized Variational Autoencoder (VQ-VAE) Autoregressive Neural F_0 Model for Statistical Parametric Speech Synthesis

Xin Wang, *Member, IEEE,* Shinji Takaki, *Member, IEEE,* Junichi Yamagishi, *Senior Member, IEEE*
Simon King, *Fellow, IEEE,* Keiichi Tokuda, *Fellow, IEEE*

***Abstract*—Recurrent neural networks (RNNs) can predict fundamental frequency (F0) for statistical parametric speech synthesis systems, given linguistic features as input. However, these models assume conditional independence between consecutive F_0 values, given the RNN state. In a previous study, we proposed autoregressive (AR) neural F_0 models to capture the causal dependency of successive F_0 values. In subjective evaluations, a deep AR model (DAR) outperformed an RNN.**

Here, we propose a Vector Quantized Variational Autoencoder (VQ-VAE) neural F_0 model that is both more efficient and more interpretable than the DAR. This model has two stages: one uses the VQ-VAE framework to learn a latent code for the F_0 contour of each linguistic unit, and other learns to map from linguistic features to latent codes. In contrast to the DAR and RNN, which process the input linguistic features frame-by-frame, the new model converts one linguistic feature vector into one latent code for each linguistic unit. The new model achieves better objective scores than the DAR, has a smaller memory footprint and is computationally faster. Visualization of the latent codes for phones and moras reveals that each latent code represents an F_0 shape for a linguistic unit.

***Index Terms*—fundamental frequency, speech synthesis, neural network, variational auto-encoder**

I. Introduction

The fundamental frequency (F_0) is the frequency of vibration of the vocal folds during voiced speech sounds. The perceptual consequence of the F_0 is *pitch*. The F_0 carries phonological information in tonal languages and suprasegmental information in all languages [1]. For example: in Japanese which is a pitch-accent language, the change of F0 height at the lexical level differentiates ‘alcohol’ (/sake/) from ‘salmon’ (/sa’ke/) [2]; in English, the F_0 is used to make a particular word more prominent, or to differentiate a statement from a question. In text-to-speech synthesis (TTS), where the achievable segmental quality is now very high [3], the F_0 of the generated waveform is arguably the most important place to look for improvements in naturalness.

Although some recent TTS systems [4], [5] predict (Mel) spectrograms, in which the F_0 is implicit, many other systems still represent the F_0 explicitly as a separate acoustic feature [6], [7]. In most statistical parametric speech synthesis (SPSS) systems [8], [9], the F_0 contour is predicted from a sequence of linguistic features that have been extracted from input text using the front-end text processor. Given the predicted F_0 and spectral features, SPSS systems generate a speech waveform using a deterministic or neural vocoder [10], [11].

We focus on the task of F_0 prediction for SPSS. We assume a traditional SPSS approach as the baseline in which the input linguistic feature and output acoustic feature sequences are equal in length and aligned in time; each time step in these sequences is a *speech frame*. In such a framework, F_0 prediction can be achieved using a neural network (NN), such as a recurrent NN (RNN) with long-short-term memory (LSTM) units [12]. Although such a model is straightforward to use and performs reasonably well, conditional independence is assumed between the F_0 at different frames. To alleviate this incorrect assumption, we previously proposed autoregressive (AR) F_0 models that can capture the causal dependency of each value in the F_0 sequence (i.e., the F_0 *contour*). A subjective evaluation demonstrated that one of our AR F_0 models – the deep AR F_0 model (DAR) – outperformed an RNN [13].

Although the DAR models the probability distribution of F_0 contours better than an RNN, it still processes input linguistic features frame by frame. However, most values in a frame-rate linguistic feature vector sequence have a constant value from one frame to the next because they change only every segment, syllable, word, etc. When operating at the frame rate, the recurrent layer(s) must propagate information across many frames before it can be used to predict the F_0 of the next segment. If the model uses a convolutional layer, however, it has to use a large convolution receptive field.

In this paper, we improved both the efficiency and interpretability of the DAR and developed a two-stage Vector Quantized Variational Autoencoder (VQ-VAE)-based AR F_0 model. During F_0 generation in a TTS system, the proposed model uses a linguistic *linker* to convert the linguistic features of one linguistic unit into a latent code vector. The model then uses a DAR-based F_0 contour *generator* to output an F_0 contour conditioned on the latent vector and duration of

Xin Wang is with National Institute of Informatics, Tokyo, 101-8340, Japan. e-mail:wangxin@nii.ac.jp

Junichi Yamagishi is with the National Institute of Informatics, Tokyo, 101-8340, Japan and also with Centre for Speech Technology Research, University of Edinburgh, UK. e-mail: jyamagis@nii.ac.jp.

Simon King is with Centre for Speech Technology Research, University of Edinburgh, UK. e-mail: Simon.King@ed.ac.uk

Shinji Takaki and Keiichi Tokuda are with the Nagoya Institute of Technology, Japan. e-mail: takaki@nii.ac.jp, tokuda@nitech.ac.jp

The guest editor coordinating the review of this manuscript was Dr. Hirokazu Kameoka. This work was partially supported by JST CREST Grant Number JPMJCR18A6, Japan and by MEXT KAKENHI Grant Numbers (16H06302, 16K16096, 17H04687, 18H04120, 18H04112, 18KT0051), Japan.

the unit. In contrast to the DAR and to conventional RNN F_0 models, the linker in the proposed model operates at the segment rate rather than the frame rate. It processes only one input linguistic feature vector and produces one latent code for each segment, which reduces processing time and facilitates the use of wider context.

The training algorithm has two steps. We first use the VQ-VAE framework [14] and train the DAR-based F_0 contour generator (i.e., the VQ-VAE's decoder) jointly with the VQ-VAE's encoder and VQ codebook. We then use the trained VQ-VAE encoder and codebook to extract the quantized latent codes from the training data and train the linker to predict the latent codes from the input linguistic features. The trained model only uses the linker, VQ-VAE codebook, and decoder for F_0 generation, leaving the VQ-VAE encoder unused.

We conducted experiments in which the linguistic unit was defined as a phone, syllable, or word. We also combined latent codes from linguistic units at multiple levels. The experiments demonstrated that the model using the phone and syllable levels provided objective and subjective results no worse than the DAR, even though the proposed model is smaller. More interestingly, the learned latent codes encode the F_0 shape and average height of the phone and syllable. The latent code of the phone also encodes the voicing status.

Section II gives an overview of related neural F_0 models, then Section III explains the proposed model, including its implementation. Section IV presents the experimental evaluation, Section V discusses remaining issues and Section VI concludes the paper.

II. Brief Review of related F_0 models

We consider the F_0 modeling as a task to learn the mapping from the input linguistic features to the output F_0 sequence. At each time step, the target F_0 datum encodes the voicing status and the F_0 value if the frame is voiced.

A. Classical models

For SPSS-based TTS systems, hidden Markov models (HMMs) plus decision-trees have been widely used to model the F_0 and other spectral features. Specifically, multi-space probability distribution HMMs directly model the raw F_0 contours that may contain unvoiced frames [15], whereas HMM-based continuous F_0 models use interpolated F_0 contours and the voicing status as the target [16]. Based on these two models, many other models or methods have been proposed. For example, some F_0 models generate the F_0 contour by adding F_0 components at different linguistic levels [17], [18], [19]; some other models use an alternative F_0 representation rather than the raw or interpolated F_0 contour [20], [21].

Recently, researchers have used NNs as alternatives to HMMs to jointly model the F_0 and other spectral features [22], [23], [24]. Some researchers use NNs exclusively for F_0 modeling [25], [12], which may be reasonable because it was recently found that NNs may prioritize the spectral features over the F_0 [26]. In fact, many NN-based F_0 models have been proposed before the advent of SPSS-based TTS systems [27], [28], [29]. Some of the NN-based models may only predict F_0s for a few target points per linguistic unit [30]. Many other models generate the F_0 for each frame, but they only use simple input features and limited network size due to the limitation of computation resources at that time.

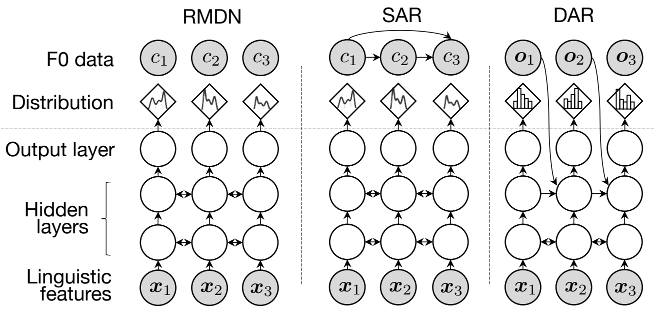


Fig. 1. RMDN and AR F_0 models. c_t and $\boldsymbol{o}_t$ denote interpolated continuous-valued F_0 (iF0) and un-interpolated quantized F_0 (qF0), respectively.

B. Autoregressive neural F_0 models

We focused on recent NN-based F_0 models because they are straightforward to use and perform well. The goal with an NN-based F_0 model is to convert a linguistic feature sequence of T frames $\boldsymbol{x}_{1:T} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T\}$ into an F_0 sequence $\boldsymbol{c}_{1:T} = \{c_1, \cdots, c_T\}$, where $c_t \in \mathbb{R}$ is the F_0 value at the t-th frame of an interpolated continuous-valued F_0 contour (iF0) [1]. For an RNN-based F_0 model, its output at the t-th frame can be written as $\widehat{c}_t = \boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}}(\boldsymbol{x}_{1:T}, t)$, where $\boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}}(\cdot)$ and $\boldsymbol{\Theta}$ denote the non-linear transformation conducted by the network and the network's weights, respectively. Usually, $\boldsymbol{\Theta}$ can be learned by minimizing a mean square error (MSE) $E = \sum_{t=1}^{T} ||\boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}}(\boldsymbol{x}_{1:T}, t) - c_t||^2$ over a training data set. After training, the RNN can produce $\widehat{\boldsymbol{c}}_{1:\tilde{T}}$ for a new input sequence $\tilde{\boldsymbol{x}}_{1:\tilde{T}}$ by generating $\widehat{c}_t = \boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}^*}(\tilde{\boldsymbol{x}}_{1:\tilde{T}}, t)$, $\forall t \in \{1, \cdots, \tilde{T}\}$.

The above method implicitly treats the RNN as a probabilistic model [31] with a distribution of $\boldsymbol{c}_{1:T}$ as[2]

$$\begin{aligned} &p(\boldsymbol{c}_{1:T}|\boldsymbol{x}_{1:T}; \boldsymbol{\Theta}) \\ =&\prod_{t=1}^{T} p(c_t|\boldsymbol{x}_{1:T}; \boldsymbol{\Theta}) = \prod_{t=1}^{T} \mathcal{N}(c_t; \boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}}(\boldsymbol{x}_{1:T}, t), \beta \boldsymbol{I}), \end{aligned} \tag{1}$$

where $\mathcal{N}(\cdot)$ is the Gaussian distribution, $\boldsymbol{I}$ is an identity matrix, and β is a variance parameter. Training the RNN using the MSE criterion is equivalent to a maximum-likelihood training on the probabilistic model; using $\boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}^*}(\tilde{\boldsymbol{x}}_{1:\tilde{T}}, t)$ as the generated F_0 is equivalent to taking the mean of the Gaussian distribution as the output. This probabilistic interpretation also allows the RNN to be extended as a recurrent mixture density network (RMDN) [32], [33] where the Gaussian distribution is replaced with a Gaussian mixture model (GMM).

An RNN-based F_0 model has been demonstrated to be better than those based on feedforward NNs [12]. However, as Equation (1) shows, a conditional independency between c_{t_1} and c_{t_2}, $\forall t_2 \neq t_1$ is assumed with an RNN (and RMND),

[1] The interpolated F_0 contour is modeled jointly with a sequence of voicing statuses [16]. The voicing status is omitted here for the purpose of explanation.

[2] In this paper, we use $p(\cdot)$ and $P(\cdot)$ to denote the probability density function for a continuous random variable and probability mass function for a discrete random variable, respectively.

which is incompatible with the fact that F_0 values in neighboring frames may be highly correlated. We recently proposed AR neural F_0 models that learn the causal temporal dependency of F_0 contours [13]. One of these models, the shallow AR model (SAR), defines the distribution of $\boldsymbol{c}_{1:T}$ as

$$\begin{aligned} p(\boldsymbol{c}_{1:T}|\boldsymbol{x}_{1:T};\boldsymbol{\Theta},\boldsymbol{\Psi}) &= \prod_{t=1}^{T} p(c_t|\boldsymbol{c}_{t-K:t-1},\boldsymbol{x}_{1:T};\boldsymbol{\Theta},\boldsymbol{\Psi}) \\ &= \prod_{t=1}^{T}\mathcal{N}(c_t;\boldsymbol{\mathcal{H}}_{\boldsymbol{\Theta}}(\boldsymbol{x}_{1:T},t)+\sum_{k=1}^{K}a_k c_{t-k}+b,\beta\boldsymbol{I}) \end{aligned}, \quad (2)$$

where $\boldsymbol{\Psi} = \{a_1,\cdots,a_K,b\}$. In contrast to an RNN, the distribution of c_t is assumed with SAR to be dependent not only on $\boldsymbol{x}_{1:T}$ but also on the observations $\boldsymbol{c}_{t-K:t-1}$ in the previous K frames. Note that the SAR can also be defined on the basis of the RMDN [13].

Although the SAR is theoretically better, it fails to outperform an RNN-based model for F_0 modeling because the SAR only models the local dependency [13]. A more powerful of our previous models is the DAR [13], which defines:

$$P(\boldsymbol{o}_{1:T}|\boldsymbol{x}_{1:T};\boldsymbol{\Theta}) = \prod_{t=1}^{T} P(\boldsymbol{o}_t|\boldsymbol{o}_{1:t-1},\boldsymbol{x}_{1:T};\boldsymbol{\Theta}), \quad (3)$$

To avoid the impact of the interpolated F_0 curves, the DAR uses quantized F_0s (qF0s) rather than interpolated continuous-valued F_0s as the target. The quantized F_0 of the t-th frame is represented as a one-hot vector $\boldsymbol{o}_t = [o_{t,0}, o_{t,1},\cdots,o_{t,N}]$, where N is the number of quantization levels. For example, $\boldsymbol{o}_t = [1,0,\cdots,0]$ denotes an unvoiced frame, while $\boldsymbol{o}_t = [0,0,\cdots,o_{t,n} = 1,\cdots,0]$ means an F_0 value at the n-th quantization level. The DAR then uses a hierarchical softmax layer to calculate a categorical distribution for each quantized F_0 datum. Suppose that the j-dimension of $\boldsymbol{o}_t$ is 1, the probability of $\boldsymbol{o}_t$ can then be written as $P(\boldsymbol{o}_t|\boldsymbol{o}_{1:t-1},\boldsymbol{x}_{1:T};\boldsymbol{\Theta}) \triangleq P_t(J=j)$, where

$$P_t(J=j) = \begin{cases} \dfrac{e^{h_{t,0}}}{1+e^{h_{t,0}}}, & j\in\{0\} \\ \dfrac{1}{1+e^{h_{t,0}}}\dfrac{e^{h_{t,j}}}{\sum_{k=1}^{N}e^{h_{t,k}}}, & j\in[1,N] \end{cases}. \quad (4)$$

Here, $h_{t,j}$ is the j-th dimension of the input vector $\boldsymbol{h}_t$ to the softmax layer. In implementation, the hierarchical softmax uses a sigmoid function to compute the probability of being unvoiced, i.e., $P_t(\text{unvoiced}) \triangleq P_t(J=0) = \frac{e^{h_{t,0}}}{1+e^{h_{t,0}}}$. It then uses a normal softmax to compute the conditional probability of each quantized F_0 level as $P_t(J=j|\text{voiced}) = \frac{e^{h_{t,j}}}{\sum_{k=1}^{N}e^{h_{t,k}}}, j\in[1,N]$.

More importantly, the DAR introduces the links that feed the previous observation into a uni-directional recurrent layer, which is plotted in Figure 1. The feedback datum is transformed non-linearly and propagated to the rest of the utterances, which theoretically enables the distribution of $\boldsymbol{o}_t$ to depend on $\boldsymbol{o}_{1:t-1}$. Compared with the SAR, the DAR theoretically models non-linear temporal dependency in a longer time-span. Our previous experiments demonstrated that the DAR outperformed the SAR and an RNN-based model with a statistically significant difference for F_0 modeling [13].

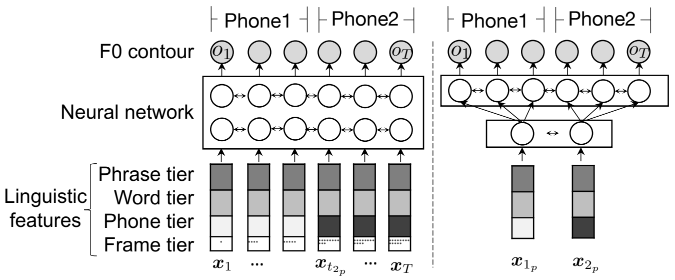


Fig. 2. Illustration of conventional F_0 model (left) and proposed F_0 model (right) that both output an F_0 contour of T frames. Subscripts 1_p and 2_p denote 1st and 2nd phone, respectively. t_{2_p} indexes first frame of 2nd phone. Linguistic features with different values are displayed in different colors.

III. Proposed VQ-VAE-based AR F_0 model

A. *Shortcomings of conventional NN-based F_0 models*

The input linguistic features $\boldsymbol{x}_{1:T}$ to an F_0 model mainly contain the linguistic properties of phone, syllable, and other high-level segments. When a TTS front-end generates $\boldsymbol{x}_{1:T}$, it simply replicates the features of the n-th segment to $\{\boldsymbol{x}_{t_n},\boldsymbol{x}_{t_n+1},\cdots,\boldsymbol{x}_{t_{(n+1)}-1}\}$, where t_n and $t_{(n+1)-1}$ denote the first and last frames of that segment. Although the frame index t may be added, most dimensions of the linguistic feature vectors remain the same within one segment. An example is illustrated on the left side of Figure 2.

It is inefficient for the DAR and other conventional NN-based F_0 models to convert $\boldsymbol{x}_{1:T}$ into the F_0 contour $\boldsymbol{o}_{1:T}$ frame by frame. First, the input layers spend much time processing replicated linguistic features in one segment. Furthermore, they cannot easily retrieve the features of neighboring segments. In a recurrent layer, the features of the surrounding segments may be attenuated when they are propagated across frames. If a convolutional layer is used, its receptive field must be sufficiently large to cover surrounding segments.

Another shortcoming is that the hidden features of conventional NN-based F_0 models cannot be easily interpreted. In some classical F_0 modeling frameworks, such as the combination of decision trees and the Tilt model [34], the linguistic features are converted into meaningful discretized F_0 events (e.g., Tilt events) then transformed into an F_0 contour. If an NN-based F_0 model provides meaningful hidden features, it would facilitate theoretical research on speech prosody.

B. *Introducing latent variables to NN-based F_0 models*

As the right side of Figure 2 illustrates, a more efficient and interpretable model may only need to process a short input sequence $\boldsymbol{x}_{1:N_p}$, where N_p is the number of segments such as phones in the utterance. To convert $\boldsymbol{x}_{1:N_p}$ into $\boldsymbol{o}_{1:T}$ (or $\boldsymbol{c}_{1:T}$), the F_0 model may introduce an intermediate feature sequence $\boldsymbol{e}_{1:N_p}$ that satisfies the following three conditions:

- $\boldsymbol{e}_{1:N_p}$ can be easily predicted from $\boldsymbol{x}_{1:N_p}$;
- $\boldsymbol{e}_{1:N_p}$ can be easily converted to a frame-level $\boldsymbol{e}_{1:T}$ then $\boldsymbol{o}_{1:T}$, given the duration of each phone;
- $\boldsymbol{e}_{1:N_p}$ is interpretable.

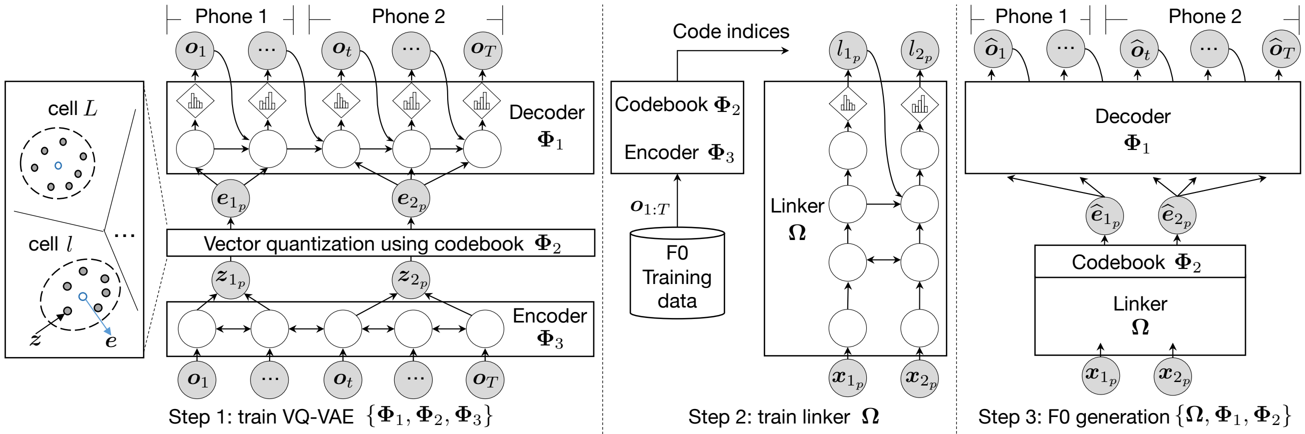


Fig. 3. Framework of proposed VQ-VAE-based F_0 model. Note that example utterance contains two phones and T frames. $\boldsymbol{z}$, $\boldsymbol{e}$, and l denote raw latent code, codeword vector (centroid of code cell), and code index in codebook, respectively. Subscripts 1_p and 2_p denote first and second phones, respectively.

One potential implementation for the above idea is to replace the hidden layer near the input side of the DAR with a clockwork recurrent layer [35], which extracts $\boldsymbol{e}_{1:N_p}$ from $\boldsymbol{x}_{1:N_p}$ and replicates $\boldsymbol{e}_{1:N_p}$ into $\boldsymbol{e}_{1:T}$ given the duration of each phone. However, such a clockwork DAR model did not outperform the normal DAR in our pilot experiments. Furthermore, the clockwork recurrent layer did not produce meaningful hidden features.

Another approach is to treat $\boldsymbol{e}_{1:N_p}$ as latent variables and formulate a two-staged F_0 model as

$$\begin{aligned} &P(\boldsymbol{o}_{1:T}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Theta}) \\ &\quad= \int_{\boldsymbol{e}_{1:N_p}} P(\boldsymbol{o}_{1:T}|\boldsymbol{e}_{1:N_p};\boldsymbol{\Phi})p(\boldsymbol{e}_{1:N_p}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Omega})d\boldsymbol{e}_{1:N_p}, \end{aligned} \tag{5}$$

where $p(\boldsymbol{e}_{1:N_p}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Omega})$ converts $\boldsymbol{x}_{1:N_p}$ into $\boldsymbol{e}_{1:N_p}$, $P(\boldsymbol{o}_{1:T}|\boldsymbol{e}_{1:N_p};\boldsymbol{\Phi})$ generates the F_0 contour given $\boldsymbol{e}_{1:N_p}$, and $\boldsymbol{\Theta} = \{\boldsymbol{\Phi}, \boldsymbol{\Omega}\}$ [3]. Such a model can be trained under the framework of a conditional variational autoencoder (VAE) by maximizing an evidence lower bound (ELBO) [37]

$$\begin{aligned} &\log P(\boldsymbol{o}_{1:T}|\boldsymbol{x}_{1:N_p}) \\ &\geq \mathbb{E}_{q(\boldsymbol{e}_{1:N_p}|\boldsymbol{o}_{1:T},\boldsymbol{x}_{1:N_p};\boldsymbol{\Gamma})} \log P(\boldsymbol{o}_{1:T}|\boldsymbol{e}_{1:N_p};\boldsymbol{\Phi}) \\ &\quad - \text{KLD}\big[q(\boldsymbol{e}_{1:N_p}|\boldsymbol{o}_{1:T},\boldsymbol{x}_{1:N_p};\boldsymbol{\Gamma})||p(\boldsymbol{e}_{1:N_p}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Omega})\big], \end{aligned} \tag{6}$$

where $P(\boldsymbol{o}|\boldsymbol{e};\boldsymbol{\Phi})$, $q(\boldsymbol{e}|\boldsymbol{o},\boldsymbol{x};\boldsymbol{\Gamma})$, and $p(\boldsymbol{e}|\boldsymbol{x};\boldsymbol{\Omega})$ denote the decoder, encoder, and prior distribution of the latent variables, respectively, and KLD is the Kullback-Leibler divergence.

As an implementation, the encoder $q(\boldsymbol{e}|\boldsymbol{o},\boldsymbol{x};\boldsymbol{\Gamma})$ may use a linguistic-boundary-based pooling strategy to summarize the F_0 features from $\boldsymbol{o}_{1:T}$ for each of the N_p phones (see Section III-C1) and generate the latent variables $\boldsymbol{e}_{1:N_p}$. The decoder $P(\boldsymbol{o}|\boldsymbol{e};\boldsymbol{\Phi})$ can be a DAR that receives the replicated latent variables $\boldsymbol{e}_{1:T}$ as input.

However, our pilot experiments showed that the KLD decreased to zero after only two training epochs, and the decoder ignored the latent variables. The resulting model became similar to the DAR and the latent variables became useless. The above phenomenon has been reported in VAE-based text and image generation models under the name of 'posterior collapse' [38], [39]. One possible reason is that, without using latent variables, the DAR-based decoder has found sufficient information from the feedback data $\boldsymbol{o}_{1:t-1}$ to predict $\boldsymbol{o}_t$.

C. *Proposed model and implementation*

To avoid 'posterior collapse', we define the proposed F_0 model on the basis of the VQ-VAE framework [14]:

$$\begin{aligned} &P(\boldsymbol{o}_{1:T}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Theta}) \\ &\quad= \sum_{\boldsymbol{e}_{1:N_p}\in\boldsymbol{\Phi}_2} P(\boldsymbol{o}_{1:T}|\boldsymbol{e}_{1:N_p};\boldsymbol{\Phi}_1)P(\boldsymbol{e}_{1:N_p}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Omega}), \end{aligned} \tag{7}$$

where the latent vectors $\boldsymbol{e}_{1:N_p}$ are assumed to be selected from a codebook $\boldsymbol{\Phi}_2$ using a vector quantization process.

Although other frameworks, such as the annealing trick [38], may also alleviate the 'posterior collapse', we chose the VQ-VAE framework because the quantized latent codes are more compatible with the linguistic assumption that F_0 can be abstracted as discretized prosodic events [40], [41]. Another motivation is that the VQ-VAE framework has not been used for F_0 modeling. A probabilistic interpretation on the VQ-VAE and its relationship with a conventional VAE are reported in another of our studies [36].

We list the steps to train and use the proposed VQ-VAE-based F_0 model based on the definition in Equation (7):

1) Train the decoder $P(\boldsymbol{o}_{1:T}|\boldsymbol{e}_{1:N_p};\boldsymbol{\Phi}_1)$ with the codebook $\boldsymbol{\Phi}_2$ and an encoder $q(\boldsymbol{e}_{1:N_p}|\boldsymbol{o}_{1:T};\boldsymbol{\Phi}_3)$ using the standard VQ-VAE training method;
2) Use the trained encoder and codebook to extract the quantized codes $\boldsymbol{e}_{1:N_p}$ from the F_0 training data and train a model $P(\boldsymbol{l}_{1:N_p}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Omega})$ to predict the indices $\boldsymbol{l}_{1:N_p}$ of the quantized codes $\boldsymbol{e}_{1:N_p}$ from the input linguistic features $\boldsymbol{x}_{1:N_p}$. Because $P(\boldsymbol{l}_{1:N_p}|\boldsymbol{x}_{1:N_p};\boldsymbol{\Omega})$ links the latent F_0 space with the linguistic feature space, it is referred to as a *linguistic linker*.
3) Combine the linker, VQ-VAE codebook, and decoder as a full-fledged model for F_0 generation, leaving the VQ-VAE encoder unused.

[3] With Equation (5), it is assumed that $\boldsymbol{o}$ is conditionally independent from $\boldsymbol{x}$ given $\boldsymbol{e}$. Another definition is to assume $\boldsymbol{o}$ depends on both $\boldsymbol{x}$ and $\boldsymbol{e}$, i.e., $p(\boldsymbol{o}_{1:T}|\boldsymbol{e}_{1:N_p},\boldsymbol{x}_{1:N_p};\boldsymbol{\Phi})$. In this case $\boldsymbol{e}$ only encodes $\boldsymbol{o}$'s variation that cannot be explained by $\boldsymbol{x}$ [36]. We used the definition in Equation (5) because we hope that $\boldsymbol{e}$ can fully encode the F_0 variation in a meaningful manner.

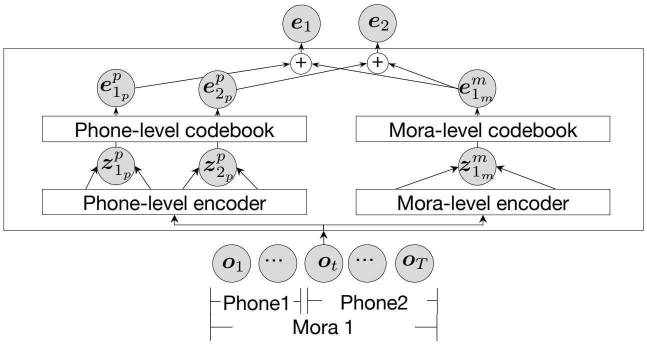


Fig. 4. Example of phone-mora VQ-VAE encoder that extracts latent codes from utterance with T frames. This utterance contains single mora that consists of two phones. Phone and mora latent codes are equal in dimension.

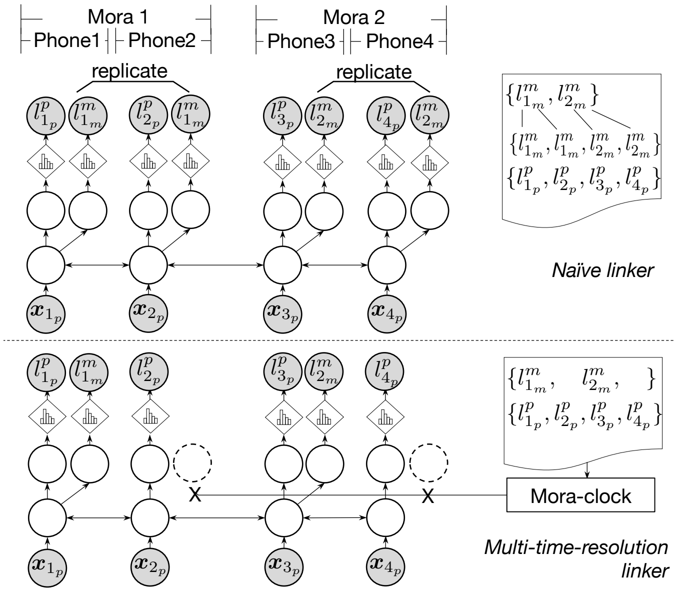

Mora 1
Phone1
Phone2
Mora 2
Phone3
Phone4
replicate
replicate
$l^p_{1_p}$ $l^m_{1_m}$ $l^p_{2_p}$ $l^m_{1_m}$ $l^p_{3_p}$ $l^m_{2_m}$ $l^p_{4_p}$ $l^m_{2_m}$
$\{l^m_{1_m}, l^m_{2_m}\}$
$\{l^m_{1_m}, l^m_{1_m}, l^m_{2_m}, l^m_{2_m}\}$
$\{l^p_{1_p}, l^p_{2_p}, l^p_{3_p}, l^p_{4_p}\}$
x_{1_p} x_{2_p} x_{3_p} x_{4_p}
Naïve linker
$\{l^m_{1_m}, l^m_{2_m}, \}$
$\{l^p_{1_p}, l^p_{2_p}, l^p_{3_p}, l^p_{4_p}\}$
X
X
Mora-clock
Multi-time-resolution linker

Fig. 5. Example of naive linker (top) and multi-time-resolution linker (bottom) that predict mora-level code indices $\{l^m_{1_m}, l^m_{2_m}\}$ and phone-level code indices $\{l^p_{1_p}, \cdots, l^p_{4_p}\}$. Subscripts n_p and n_m index phone and mora, respectively. Feedback links are not plotted for this example.

These three steps are illustrated in Figure 3. Details of the three steps are explained in the following sections. Note that, because $e_{1:N_p}$ are quantized, the combination of the codebook and linker is equivalent to $P(e_{1:N_p}|x_{1:N_p};\mathbf{\Omega})$ in Equation (7). The VQ-VAE part is trained in an almost-unsupervised manner since it only requires the F_0 and linguistic boundary of linguistic segments.

*1) **VQ-VAE:*** The first step is to use the VQ-VAE framework and learn the parameter of $p(o_{1:T}|e_{1:N_p};\mathbf{\Phi}_1)$ together with the codebook and encoder. The three components are trained by minimizing

$$\begin{aligned}\mathcal{L}(\mathbf{\Phi}) = &-\log p(o_{1:T}|e_{1:N_p};\mathbf{\Phi}_1)\\ &+\big\|e_{1:N_p}-\mathrm{sg}[z_{1:N_p}]\big\|_2^2+\beta\big\|z_{1:N_p}-\mathrm{sg}[e_{1:N_p}]\big\|_2^2,\end{aligned} \tag{8}$$

where the quantized code vectors $e_{1:N_p}$ (i.e., centroid of code cell) and raw latent vectors $z_{1:N_p}$ are given by

$$e_{1:N_p} = \text{Vector\_quantization}_{\mathbf{\Phi}_2}(z_{1:N_p}), \tag{9}$$

$$z_{1:N_p} = \text{Encoder}_{\mathbf{\Phi}_3}(o_{1:T}), \tag{10}$$

and $\mathbf{\Phi} = \{\mathbf{\Phi}_1, \mathbf{\Phi}_2, \mathbf{\Phi}_3\}$. The first term in Equation (8) measures the reconstruction loss; the second term drives the quantized vectors towards the raw latent vectors; the third term is a 'commitment loss' that prevents the output of the encoder from growing arbitrarily large [14]. The hyper-parameter β scales the commitment loss and is set to 0.25 as in the original paper [14]. The operator $\mathrm{sg}[\cdot]$ zeros out the gradient back propagated to the argument.

Training the VQ-VAE is not straightforward because the gradients of $z_{1:N_p}$ w.r.t the reconstruction loss is undefined due to the undifferentiable VQ process in Equation (9). Following the original paper [14], we use the straight-through estimator [42] to copy the gradients from $e_{1:N_p}$ to $z_{1:N_p}$ during the back-propagation on Equation (9). Accordingly, the encoder $\mathbf{\Phi}_3$ is trained by minimizing the first and third terms of Equation (8), while the codebook $\mathbf{\Phi}_2$ and decoder $\mathbf{\Phi}_1$ are updated based on the second and first terms, respectively.

Remember that we only use a single code e_{n_p} to encode the F_0 curve for the n-th segment. This is implemented using a linguistic-boundary-based pooling strategy. As step 1 in Figure 3 shows, the encoder uses a bi-directional recurrent layer to process $o_{1:T}$ and extracts z_{n_p} for the n-th phone by concatenating the hidden features of the first and last frames of that phone. After that, the centroid vector e_{n_p} closest to z_{n_p} is retrieved from the codebook. Given the quantized code sequence $\{e_1, \cdots, e_{N_p}\}$, the decoder simply replicates each code to all the frames in the corresponding phone and reconstructs $o_{1:T}$ frame by frame. Note that the decoder uses a similar structure to the DAR.

The explanation above only concerns the latent codes for the phone. However, it is straightforward to extract latent codes of other linguistic units using additional encoders. In Figure 4, for example, the mora encoder is added to extract the mora latent codes [4]. This mora encoder has a similar structure to the phone encoder except that it outputs one code per mora. Each mora code is replicated to every phone within that mora, and the sum of the phone and mora codes is used as the latent representation for the phone. Note that the latent codes of different linguistic layers are equal in dimension. The latent vectors can also be concatenated. However, summing the latent vectors was found to be slightly better in our pilot experiments.

Encoders of multiple linguistic levels should be trained in a top-down manner. For the case in Figure 4, the mora encoder and codebook are first trained, after which the phone part is updated with the mora encoder and codebook fixed. The single decoder, however, is always updated. Such a training approach prevents the VQ-VAE from ignoring the latent codes provided by a high-level encoder.

*2) **Linguistic linker:*** After the VQ-VAE is trained and the latent codes are extracted from the training F_0 data, the linguistic linker learns to map the linguistic features into the latent codes. Because the latent codes are codewords in a codebook, the task of the linker is equivalent to a sequential classification task, in which case the input is the sequence of

[4] A Japanese mora is a phonological unit that determines the timing of speech. For example, the word 'Japan' in Japanese has two pronunciations: Ni-ho-n (3 moras) and Ni-p-po-n (4 moras).

linguistic features $\boldsymbol{x}_{1:N} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N\}$ and the target is the code index sequence $\boldsymbol{l}_{1:N} = \{l_1, \cdots, l_N\}$, where l_n is the index of the code $\boldsymbol{e}_n$ for the n-th segment. Such a linker can be implemented using an RMDN with a softmax output layer. Other techniques for classification models can also be applied, for example the skip-connection [43] and dropout [44].

When the VQ-VAE uses latent codes at multiple linguistic levels, the linker must predict multiple code index sequences. However, the index sequences may differ in length because the number of linguistic segments may differ at different levels. A naive implementation is to replicate the higher-level indices to the segments at the lowest level. In the example plotted at the top of Figure 5, the mora level indices $\{l^m_{1_m}, l^m_{2_m}\}$ are replicated to the phones, and the linker predicts the mora and phone code indices simultaneously for every phone. Note that the alignment between different linguistic levels can be easily retrieved from the linguistic features.

By replicating the mora code indices to the phones, the naive linker treats the replicated indices as the values of different random variables and may assign them different probabilities. This model assumption is against the fact that the replicated indices should have the same emitting probability. A statistically better linker may adopt a multi-time-resolution architecture similar to a clockwork RNN [35]. Suppose the mora and phone levels are used and their code indices are denoted as $l^m_{1:N_m}$ and $l^p_{1:N_p}$, respectively. The idea is to emit a hidden state and output index for a mora only when the current time step hits the first phone in the mora. Such a linker is plotted at the bottom of Figure 5.

3) ***F0 generation****:* Once the linker is trained, the proposed F_0 model can be built by concatenating the linker, codebook, and VQ-VAE decoder. During F_0 generation, the linker can generate a sequence of probability vectors $\boldsymbol{P}_{1:N} = \{\boldsymbol{P}_1, \cdots, \boldsymbol{P}_N\}$, where the vector $\boldsymbol{P}_n = [P(l_n = 1|\boldsymbol{x}_{1:N}), \cdots, P(l_n = L|\boldsymbol{x}_{1:N})]$ contains the probabilities of emitting each of the L code vectors at the n-th step. After that, a soft code vector can be computed as $\widehat{e}_n = \sum_{l=1}^{L} P(l_n = l|\boldsymbol{x}_{1:N})\boldsymbol{e}_l$, where $\boldsymbol{e}_l$ is the code with the index l. This soft code can be fed to the decoder for F_0 generation, which approximates the summation in Equation (7).

IV. Experiments

Following the explanation on the proposed VQ-VAE-based F_0 model, we now discuss the experiments. After describing the data and model configuration in Sections IV-A and IV-B, we describe the evaluation of the VQ-VAE part and linker of the proposed model in Sections IV-C and IV-D, respectively. We then show the results of comparing the proposed model with other NN-based F_0 models in Section IV-E.

A. Corpus and data

The experiments were conducted using the same speech corpus and configuration as in our previous study on the DAR [13]. Among the speakers in the corpus [45], we used fifty hours of recordings from a female speaker (F009). This subset contained 30,016 news-reading utterances, among which 500 were randomly selected as the validation set and another 500 as the test set. The waveform sampling rate was 48 kHz.

Natural F_0 data were extracted using an ensemble of multiple F_0 trackers with a frame shift of 5 ms [46]. The raw F_0 data were transformed to the Mel scale using $m = 1127\log(1+\text{F0}/700)$. The interpolated continuous-valued F_0s (iF0s) were obtained after interpolating the unvoiced frames. In the case of the quantized F_0 (qF0), the Mel-scale F_0 values were quantized into 255 levels between 66 and 529 on the Mel scale, the same recipe as in our previous study [13]. The quantized F_0 values and an unvoiced symbol were encoded as a one-hot vector $\boldsymbol{o}_t \in \{0, 1\}^{256}$ for each frame. The F_0 delta and delta-delta components were not used.

Linguistic features were extracted from the text using a Japanese TTS front-end called OpenJTalk [47]. The basic linguistic unit is the phone, above which there are mora, word, and phrase levels. A linguistic feature vector for a phone has 386 dimensions and encodes the quin-phone identity, word part-of-speech, phase accent type, and so on [48]. To create the input linguistic feature sequences for conventional NN-based F_0 models, the linguistic features were replicated to each frame in the phone and augmented with three additional features: the frame position in the utterance (forward and backward) and the total number of frames.

Speech waveforms for listening tests were synthesized using the WORLD vocoder [49] given natural Mel-generalized cepstral coefficients [50] (60 coefficients per frame) and band aperiodicity features (25 dimensions per frame). Natural duration was used for both model training and testing.

B. Model configuration, training, and testing

We evaluated the proposed VQ-VAE-based F_0 model with different configurations on the VQ-VAE part and linker[5]. Some of the configurations are plotted in Figures 6, and other configurations are explained in Sections IV-C and IV-D. Given the best configuration, we then compared the proposed VQ-VAE-based model with baseline NN-based F_0 models.

Specifically, Figure 6 plots two configurations of the VQ-VAE part, one using only the phone level latent codes and the other using both the phone and mora levels. Due to the limited space, configurations using the latent codes of other linguistic levels are not plotted. In all the VQ-VAE configurations, the codebooks contained 128 code words of 64 dimensions [6], and the decoder used the same structure as the upper part of `DAR` in Figure 7. The 'extractor' layers executed the linguistic-boundary-based pooling and generated the latent vectors for each linguistic unit. The 'code summation' layer summed the codes of different linguistic levels, as Figure 4 shows. The dropout rate of 50% in the feedback link was identical to that used in `DAR`. The VQ-VAE part was trained using

[5] All models were implemented using CURRENNT [51]. The toolkit and training recipes will be available online (https://nii-yamagishilab.github.io).

[6] We chose the number of codes based on how frequently each code was used in the validation set. While the phone-level codebook used around 128 codes, high-level codebooks used fewer codes. However, it is harmless to leave unused codes in the codebook. We also tried to increase code dimensions from 64 to 128 but found that the performance of TTS F_0 modeling did not improve.

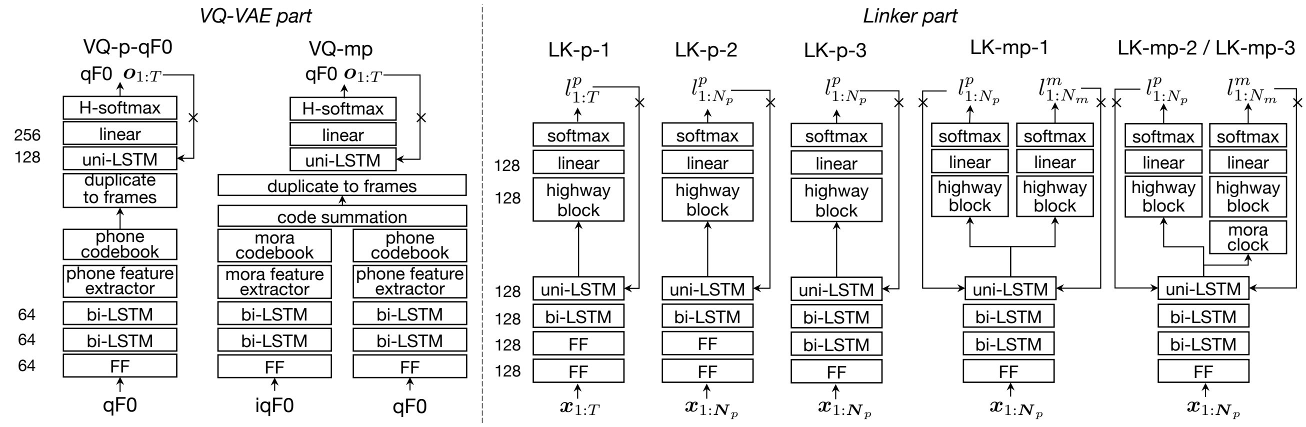


Fig. 6. Example components of proposed VQ-VAE-based F_0 model, including two VQ-VAEs in Table II and six linkers in Table III. `VQ-p-qF0` and `VQ-mp` denote phone-level and mora-phone-level VQ-VAEs, respectively. `LK-p-*` and `LK-mp-*` are linkers designed for `VQ-p-qF0` and `VQ-mp`, respectively. `LK-mp-3` is identical to `LK-mp-2` except that `LK-mp-3` used dropout in every hidden layer. Numbers near hidden layer denote layer size. Subscripts T, N_p, and N_m denote number of frames, phones, and moras, respectively. FF, H-softmax, bi-LSTM, and un-LSTM denote feedforward, hierarchical-softmax, bi-directional, and uni-directional LSTM layer, respectively. × denotes dropout in feedback link.

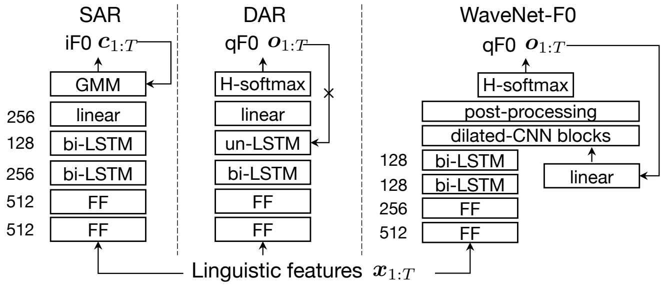


Fig. 7. Network structure of three baseline experimental F_0 models

TABLE I
MEDIAN DURATION OF LINGUISTIC UNIT

	Phrase	Word	Mora	Phone
Median duration (frames)	95	51	25	13

the top-down training strategy explained in Section III-C1 and the Adam optimizer with default configuration (learning rate= 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$) [52].

Figure 6 also plots some of the linker configurations. In all the configurations, each highway block contained five layers that converts the input vector $\boldsymbol{x}$ into $\boldsymbol{y} = (1 - \boldsymbol{g}) \odot \boldsymbol{x} + \boldsymbol{g} \odot \text{ReLU}(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b})$, where $\boldsymbol{g} = \text{sigmoid}(\boldsymbol{W}_g\boldsymbol{x} + \boldsymbol{b}_g)$, and $\odot$ is the element-wise product. The dropout rate in the feedback links was set to 25% based on a pilot test. The linker was trained using conventional gradient descent with early stopping (learning rate $= 10^{-5}$) and further tuned using AdaGrad (learning rate $= 0.001$) with early stopping.

Two baseline NN-based F_0 models `SAR` and `DAR` were trained in our previous study [13]. The `SAR` used a GMM with two mixture components to model iF0 and a binomial distribution to model the voicing status. It was initialized using a trained RMDN and fine-tuned after adding the AR dependency of order $K = 1$. The `DAR` modeled qF0 and used a dropout rate of 50% in the feedback link. It was initialized using the layer-size-dependent strategy [53] and trained from scratch.

Because both `SAR` and `DAR` are based on RNNs, we included a convolution NN (CNN) as a reference system. Specifically, we used the WaveNet [3] to model the F0 (`WaveNet-F0`) since its dilated convolution architecture is reported to be able to cover contexts in a long range. This `WaveNet-F0` uses a similar structure and training recipe to the WaveNet vocoder in another of our studies [54]. It contained 40 dilated CNN blocks with the dilation size of the n-th block as $2^{\text{mod}(n-1,10)}, n \in [1, 40]$. The sizes of the residual- and skip- channels were 64 and 256, respectively. Different from the WaveNet vocoder, the `WaveNet-F0` used a hierarchical softmax output layer and modeled qF0 frame by frame.

Subjective and objective tests were conducted on the test set to evaluate model performance. For the objective test, several metrics were calculated on the generated F_0 given the natural duration information, including a root mean square error (**RMSE**), correlation coefficient (**CORR**), and unvoiced/voiced (U/V) classification error rate (**U/V err.**). The **RMSE** and **CORR** were calculated on frames where both natural and generated F_0s were voiced. The subjective test is discussed in Section IV-E.

C. *Pilot test I: F_0 encoding-decoding using VQ-VAE*

This pilot test compared VQ-VAEs at the phone, mora, word, and phrase levels. Each VQ-VAE was trained on the training set and evaluated on the test set given the natural duration of linguistic units.

Table I lists the median duration of the linguistic units at each level, which can be used to compute the 'bit rate' of a VQ-VAE. For example, since 13 is the median duration of the phone in frames, a phone-level VQ-VAE extracts $1/13$ code per frame on average. Because the codebook size is $128 = 2^7$, the bit rate of this VQ-VAE is $7/13 = 0.538$ bit/frame. For reference, the bit rate of the natural quantized F_0 is 8 bit/frame since each frame encodes 255 quantized F_0 levels and an unvoiced flag.

*1) **VQ-VAEs using single linguistic level:*** This test involved seven VQ-VAEs with a single linguistic level. The first four VQ-VAEs modeled the qF0 and operated on the phone,

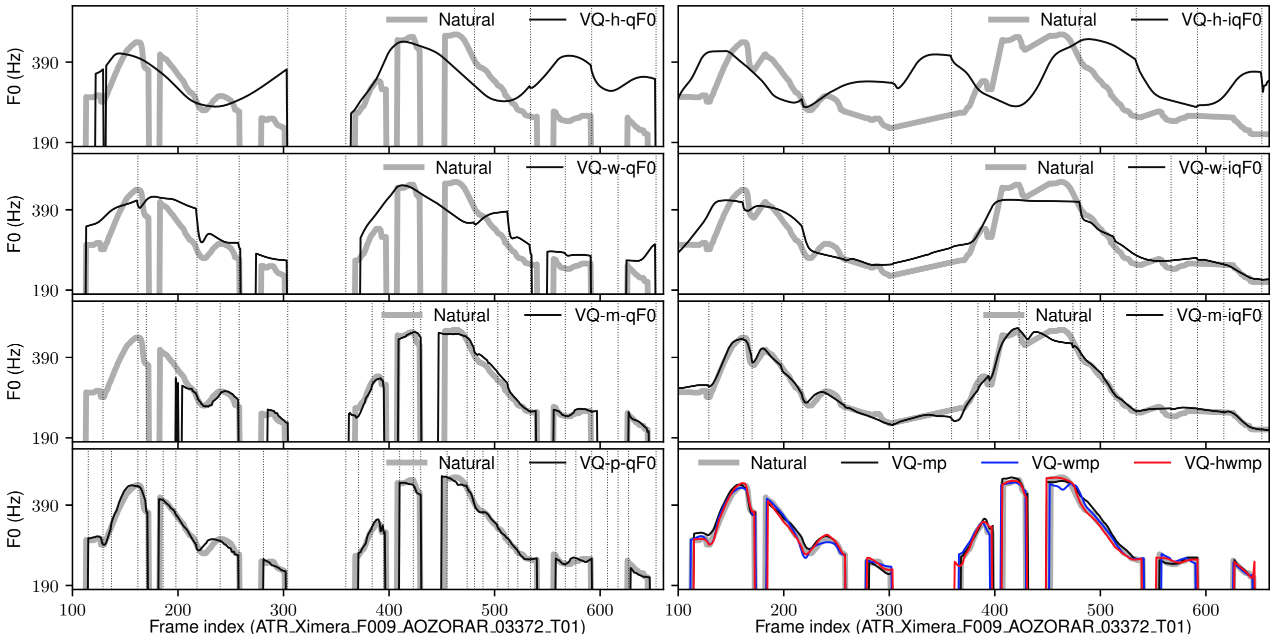


Fig. 8. F0 contour reconstructed with each VQ-VAE model through encoding and decoding one test utterance. Dotted lines denote linguistic unit boundary.

TABLE II
OBJECTIVE RESULTS OF F_0 ENCODING-CODING USING VQ-VAES.

	Bit rate (bit/frame)	**RMSE**	**CORR**	**U/V err.**
Quantized F_0	8	01.19	0.999	0.00%
`VQ-h-qF0`	0.074 (7/95)	61.87	0.425	28.78%
`VQ-w-qF0`	0.137 (7/51)	41.81	0.739	22.94%
`VQ-m-qF0`	0.280 (7/25)	22.27	0.919	14.55%
`VQ-p-qF0`	0.538 (7/13)	13.60	0.972	6.88%
`VQ-h-iqF0`	0.074	58.83	0.469	-
`VQ-w-iqF0`	0.137	37.05	0.796	-
`VQ-m-iqF0`	0.280	21.57	0.934	-
`VQ-hwmp`	1.029	11.46	0.982	4.58%
`VQ-wmp`	0.956	11.54	0.982	4.27%
`VQ-mp`	0.818	12.11	0.981	4.60%

Note: ‘qF0’ and ‘iqF0’ denote quantized F_0 and interpolated quantized F_0 , respectively. ’VQ-hwmp’, ’VQ-wmp’, and ’VQ-mp’ use phrase (h) + word (w) + mora (m) + phone (p), word + mora + phone, and mora + phone levels, respectively.

mora, word, and phrase levels, respectively. They are referred to as `VQ-p-qF0`, `VQ-m-qF0`, `VQ-w-qF0`, and `VQ-h-qF0`. The network structure of `VQ-p-qF0` is plotted in Figure 6, and the other three VQ-VAEs used similar structures except the linguistic level to extract latent codes.

Because the qF0 contains the voicing status, it may be inappropriate to model it at the linguistic levels above the phone. Therefore, three additional VQ-VAEs were added to model the interpolated quantized F_0 (iqF0): `VQ-h-iqF0`, `VQ-w-iqF0`, and `VQ-m-iqF0`. An iqF0 contour was quantized from an interpolated F_0 contour using the same F_0 quantization recipe as the qF0.

After training, these VQ-VAEs were evaluated by encoding and decoding the F_0 data in the test set, and the objective results are listed in Table II. If we compare the results of using the iqF0 and qF0, we can see that it is better for high-level VQ-VAEs to model the iqF0 rather than qF0. As Figure 8 shows, the voicing status of natural speech may change multiple times within a phrase or word. It was difficult for a phrase- or word-level VQ-VAE to encode both the F_0 shape and location where the voicing status changes. Only `VQ-p-qF0` performed well on the qF0, which is reasonable because the voicing status is a mainly phonetic property.

Even using the iqF0, `VQ-h-iqF0` failed to encode and decode the F_0 using one code per phrase. One hypothesis may be that the model is incapable of modeling the F_0 curve of roughly 95 frames in length. Another hypothesis is that the shape of the artificially interpolated F_0 curves may have interfered with the model learning. When a VQ-VAE generated latent codes at the mora or phone level, the F_0 was encoded and reconstructed with high accuracy. Particularly, `VQ-p-qF0` achieved a correlation score of 0.97, and the reconstructed F_0 was very close to the natural one.

These results are reasonable because `VQ-p-qF0` used more bits to encode the F_0 contours. However, `VQ-p-qF0` was still imperfect because its bit rate is only $7/13$ bit/frame, which is much lower than the 8 bits/frame of the qF0. This is the trade-off with a VQ-VAE as it attempts to use a single quantized latent code to describe the F_0 curve of one linguistic unit.

*2) **VQ-VAEs using multiple linguistic levels**:* Based on the above results, the second experiment involved three VQ-VAEs using multiple linguistic levels: phrase + word + mora + phone (`VQ-hwmp`), word + mora + phone (`VQ-wmp`), and mora + phone (`VQ-mp`). The structure of `VQ-mp` is plotted in Figure 6, and `VQ-hwmp` and `VQ-wmp` had similar structures. The three VQ-VAEs were trained in a top-down manner, as explained in Section III-C1. The encoders at the phrase, word, and mora levels took the iqF0 as input. The results in Table II indicate that the VQ-VAEs using multiple levels improved the objective performance compared with `VQ-p-qF0`, especially on the U/V. However, the improvement due to high-level latent codes seems to be limited because most of the variation was encoded by the phone-level latent codes.

TABLE III
OBJECTIVE RESULTS COMPARING DIFFERENT LINKERS

Linker (s/epoch)	VQ-VAE	**RMSE**	**CORR**	**U/V err.**
`LK-p-1` (1300)	`VQ-p-qF0`	32.79	0.856	7.61%
`LK-p-2` (54)		27.11	0.906	6.36%
`LK-p-3` (61)		26.74	0.908	6.36%
`LK-mp-1` (59)	`VQ-mp`	27.35	0.907	6.22%
`LK-mp-2` (63)		26.46	0.912	6.24%
`LK-mp-3` (65)		25.55	0.916	4.87%
`LK-wmp` (70)	`VQ-wmp`	26.72	0.909	5.04%
`LK-hwmp` (76)	`VQ-hwmp`	26.18	0.909	4.81%

Note: the numbers inside the brackets denote the time to train the linker for one epoch.

D. Pilot test II: linguistic linkers

The second pilot test focused on the linguistic linker. With a trained VQ-VAE, the latent codes were extracted from the training and validation sets and used to train a linker. The trained linker was then combined with the VQ-VAE's decoder and codebook into a full-fledged VQ-VAE-based F_0 model. The linker's performance, or equivalently the performance of the proposed VQ-VAE-based F_0 model, was evaluated after generating F_0 contours from linguistics features on the test set.

1) ***Phone-level experiment:*** Given the phone-level VQ-VAE model `VQ-p-qF0`, we compared three linguistic linkers (`LK-p-1`, `LK-p-2`, and `LK-p-3` plotted in Figure 6). In the case of `LK-p-1`, the phone code indices $\boldsymbol{l}^p_{1:N_p}$ in the training set were replicated to the frame-level as $\boldsymbol{l}^p_{1:T}$. `LK-p-1` was then trained to convert the linguistic features $\boldsymbol{x}_{1:T}$ into $\boldsymbol{l}^p_{1:T}$ frame by frame in the same manner as conventional NN-based F_0 models. `LK-p-2`, however, directly converted $\boldsymbol{x}_{1:N_p}$ to $\boldsymbol{l}^p_{1:N_p}$ phone by phone. `LK-p-3` was identical to `LK-p-2` except that `LK-p-3` replaced the second feedforward layer with a bi-directional LSTM. The objective results are listed in the first three rows of Table III.

First, `LK-p-1` performed worse than `LK-p-2` and required much more time in training (and of course generation). As argued in Section III-A, `LK-p-1` is inefficient because it processes redundant linguistic features frame by frame. In contrast, `LK-p-2` operates phone by phone and can easily access the linguistic features of multiple phones. This may be the main reason for the different performances of `LK-p-1` and `LK-p-2`. As the median duration of the phone is around 13 frames (see Table I), the time `LK-p-2` spends in both forward and backward computation is only $\frac{1}{13}$ of `LK-p-1`. The `LK-p-3` further improved the performance of `LK-p-2` by replacing a feedforward layer with a recurrent layer. Since the length of the input/output sequence is short, a recurrent layer would not increase the training or generation time too much compared with a feedforward layer.

2) ***Multiple-level experiment:*** Three linkers (`LK-mp-1`, `LK-mp-2`, and `LK-mp-3`) were evaluated given the phone-mora VQ-VAE (`VQ-mp`). The three linkers operated phone by phone. While `LK-mp-1` predicted the mora and phone code indices using the naive implementation in Figure 5, `LK-mp-2` and `LK-mp-3` used the mora clock. Based on `LK-mp-2`, `LK-mp-3` used dropout (rate 5%) in all the hidden layers.

The results are listed in the middle of Table III. First, `LK-mp-2` performed better than `LK-mp-1` in terms of RMSE and CORR. This result supports the use of the mora clock, even though this improvement was small since one mora in Japanese usually contains one or two phones. `LK-mp-3` further improved in performance over `LK-mp-2`, especially regarding the U/V err. This result suggests the importance of dropout for the linker. Because the number of training samples for the linker was equal to that of phones, which was only 1/13 the number of frames in the corpus, dropout may regularize the network during training.

For more linguistic levels, `LK-wmp` and `LK-hwmp` were built for `VQ-wmp` and `VQ-hwmp`. Their network structures were the same as `LK-mp-3` except additional branches to predict the word and phrase code indices. They also used dropout in hidden layers as `LK-mp-3` did. However, the results in Table III indicate that using the phrase and word levels did not improve F_0 modeling performance. This result may be reasonable because the word and phrase latent codes were not informative, as suggested from pilot test I in Section IV-C.

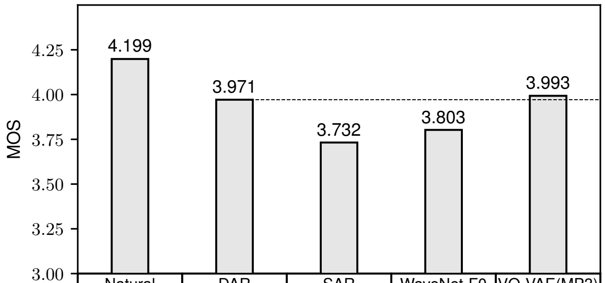
TABLE IV
OBJECTIVE RESULTS OF NN-BASED F_0 MODELS.

		Time cost				
	#. Para. million	Train h/epoch	Gen. ms/frame	**RMSE**	**CORR**	**U/V err.**
`SAR`	1.30	0.528	0.141	34.57	0.897	3.85%
`DAR`	1.48	0.555	0.205	28.30	0.903	3.46%
`WaveNet-F0`	3.28	1.194	4.109	28.04	0.903	3.52%
`VQ-VAE(MP3)`	1.11	0.435	0.147	25.55	0.916	4.87%

E. Comparing proposed VQ-VAE-based F_0 model with baseline models

The results of the pilot tests indicate that `VQ-mp` with `LK-mp-3` performed the best among different configurations. We refer to the proposed VQ-VAE-based model using this configuration as `VQ-VAE(MP3)` and compared it with the baseline NN-based F_0 models. As the results in Table IV indicate, `VQ-VAE(MP3)` achieved better RMSE and CORR scores even though the U/V err. increased. The improved RMSE and CORR scores may be the result of the unit-by-unit processing of linguistic features in the linker. The U/V err. increased because `VQ-VAE(MP3)` tended to treat a whole unit as either being voiced or unvoiced, which is shown in the analysis of latent codes in Section IV-F. This phone-level U/V classification caused more errors around the phone boundaries than the frame-level U/V classification conducted using conventional NN-based F_0 models.

A mean-opinion-score (MOS) test was conducted to compare the perceptual quality of the generated F_0. This test involved 109 paid native Japanese speakers. Each participant listened to the samples and evaluated their quality in terms of intonation using a score from 1 (unnatural) to 5 (natural). In each testing round, one vocoded speech sample and the synthetic samples from each model were played in a randomly shuffled order. In total, 1702 testing rounds were conducted. The results plotted in Figure 9 indicate that `VQ-VAE(MP3)` slightly outperformed the `DAR`, even



	Natural	DAR	SAR	WaveNet-F0	VQ-VAE(MP3)
Natural		<1e-10	<1e-10	<1e-10	<1e-10
DAR	<1e-10		<1e-10	0.000102	0.392125
SAR	<1e-10	<1e-10		0.000176	<1e-10
WaveNet-F0	<1e-10	0.000102	0.000176		0.000002
VQ-VAE(MP3)	<1e-10	0.392125	<1e-10	0.000002	

Fig. 9. Results of subjective evaluation. Numbers in table denote $p-$values calculated using two-sided Mann-Whitney U test.

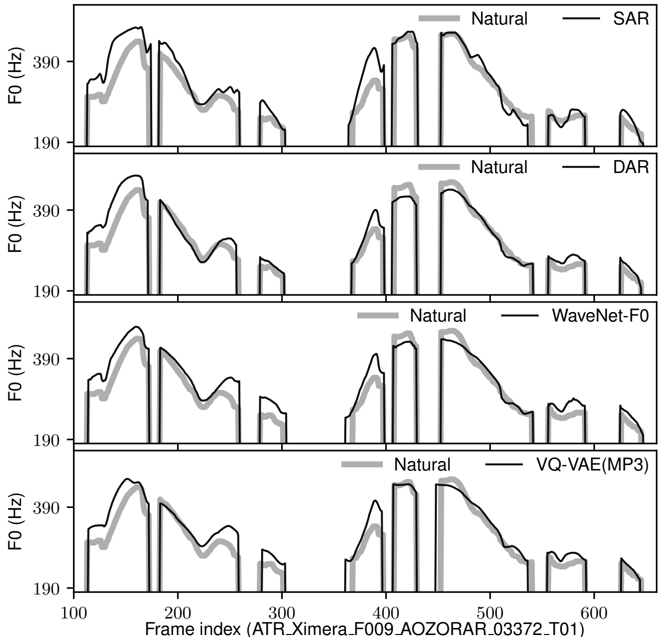


Fig. 10. F0 contours generated using baseline models and proposed VQ-VAE-based F_0 model given linguistic features of one test utterance

though the difference is not statistically significant. The results also indicate that `WaveNet-F0` and `SAR` performed worse than `VQ-VAE(MP3)`. The performance of `SAR` was expected because it only modeled local F_0 dependency [13].

The reader may wonder whether `VQ-VAE(MP3)` performed well because of its large and deep network structures, particularly the linker. In fact, `VQ-VAE(MP3)` had a smaller number of model parameters (1.11M = VQ-VAE 0.44 + linker 0.67). Since the VQ-VAE encoder is not used during F_0 generation, the parameter of `VQ-VAE(MP3)` becomes smaller when it is used for F_0 generation (0.93M = VQ-VAE decoder&codebook 0.26 + linker 0.67). The results also suggest that the mapping between linguistic features and F0 patterns may be more difficult than F_0 encoding-decoding. It may be better to assign more parameters to the linker.

Even though the linker is larger and deeper than the VQ-VAE part, it operates at the segment rate rather than frame rate. It is easier for the linker to cover a large linguistic context, as we argued before. Furthermore, the linker would not significantly increase the training and generation time because the number of segments in one utterance is much smaller that the number of frames. In our implementation, the linker part `LK-mp-3` only costs 0.019 ms out of the 0.147 ms that `VQ-VAE(MP3)` costs in generating one frame of F_0.

A conventional F_0 model such as `DAR` is less efficient because it spends extra time processing the redundant linguistic features frame by frame. Neither does it factorize the F_0 modeling task nor use a specific network structure to process the linguistic features. This may be the reason it used more parameters but did not outperform `VQ-VAE(MP3)` [7].

F. Visualization of latent F_0 code space

Since the proposed VQ-VAE-based F_0 model has shown promising results, it is interesting to investigate the 'meaning' of the latent codes learned using the VQ-VAE. For this analysis, the codewords from the phone or mora codebooks of `VQ-VAE (MP3)` were compressed to two dimensions using t-SNE [55]. Its VQ-VAE encoder was then used to extract the code indices from an F_0 contour in the test set. The results are plotted in Figure 11 (a), where the top and bottom sub-figures show the results of the phone and mora codes, respectively.

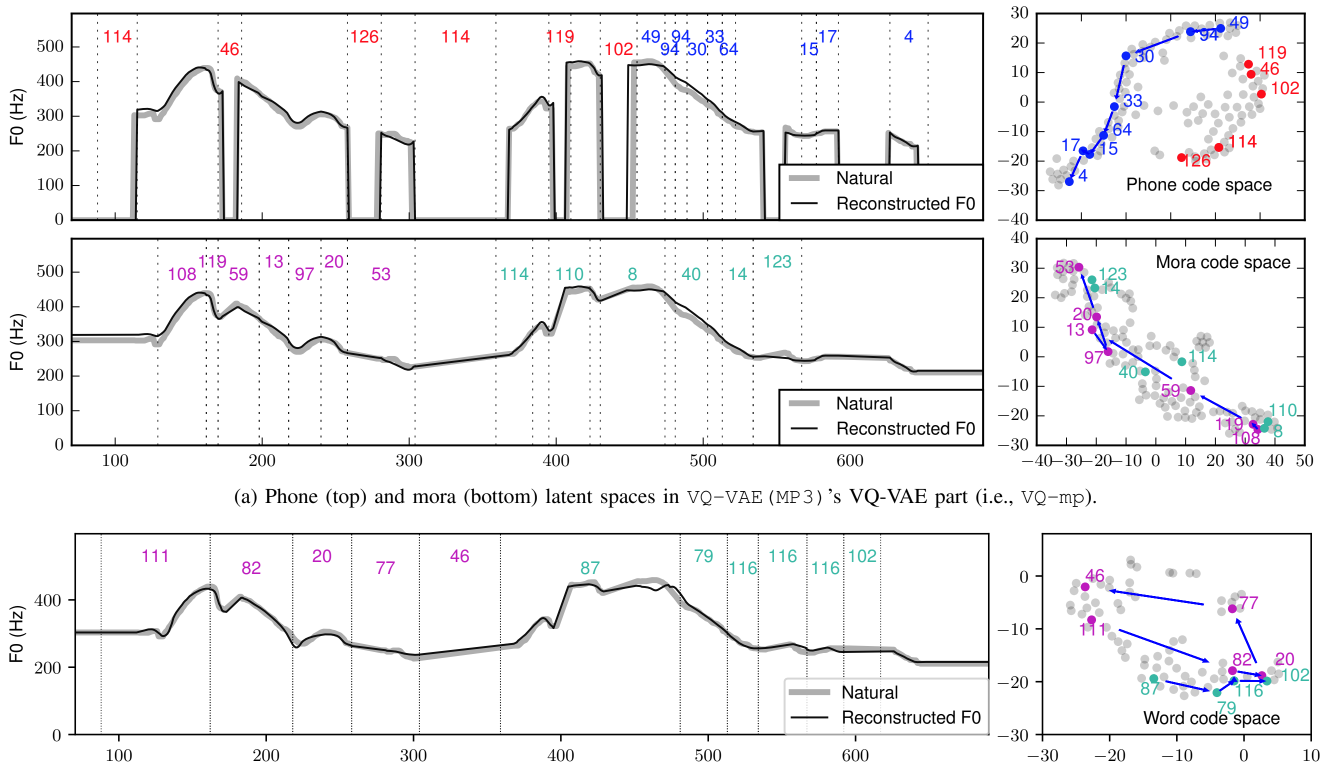
At the mora level, the code indices assigned to the first part of the F_0 contour were $\{108, 119, 59, 13, 97, 20, 53\}$. The location of these codes generally reflects the average F_0 height of the corresponding linguistic unit. For example, the mora with the code 108 contained an F_0 peak, and the following moras contained a decreasing F_0 curve. Accordingly, the code sequence $108 \rightarrow 59 \rightarrow 13 \rightarrow 20 \rightarrow 53$ formed a line starting from the right-bottom corner of the code space and ending at the left-top corner, except code 97 where the F_0 curve slightly increased. On the second part of the F_0 contour, we can observe that the F_0 curve increased at first, reached a plateau, and finally decreased. The corresponding code sequence $\{114, 110, 8, 40, 4, 123\}$ started at the center of the code space, reached the right-bottom corner, and moved to the left-top corner.

Similar to the case at the mora level, the phone-level codes generally learned the average F_0 height of phones. More interestingly, some of the codes seemed to also encode the unvoiced status and were found in the right-hand side manifold in the phone-code space, e.g., the code indices $\{114, 46, 126, 119, 102\}$ in the phone-code space of Figure 11.

The above results indicate that the VQ-VAE encoder at the phone or mora level mainly encodes the skeleton of the F_0 contour, e.g., average F_0 height and voicing status, while the decoder fills in the detailed F_0 variations. The average F_0 height may be a general representation to encode the F_0 contour at the phone or mora levels.

Analysis on the latent space at the word or higher levels did not show patterns as regular as those observed at the phone and mora levels. For example, Figure 11 (b) plots the results for the word-level latent space in `VQ-wmp`, which is difficult to see meaningful patterns. One hypothesis is that the latent

[7] We also trained a smaller `DAR` with a number of parameters equal to 1.08 million. However, while the generation speed was still around 0.185 ms/frame, the RMSE and CORR degraded to 30.01 and 0.890, respectively.



(a) Phone (top) and mora (bottom) latent spaces in VQ-VAE(MP3)'s VQ-VAE part (i.e., VQ-mp).

(b) Word latent space in VQ-wmp.

Fig. 11. Latent code indices (integer numbers) assigned to linguistic unit and their locations in 2D latent code space. Left column shows F_0 contours with linguistic unit boundaries (grey dot lines) and code indices. Right column shows codes used in left figure (colored dots) and other codes (grey dots)

space may be affected by the artificially interpolated F0 curves in unvoiced regions, for example the first word unit with the code 111 and that with code 46.

V. Discussion

The proposed VQ-VAE-based F_0 model resembles the classical two-step F_0 modeling approaches [56], [57], [58], in which the first step uses an F_0 parametric model, such as the Fujisaki model, to extract F_0 parameters for each linguistic unit, and the second step learns the mapping from linguistic features to the F_0 parameters of each linguistic unit. However, these classical approaches rely on deterministic and expert-designed F_0 parametric models. In contrast, our proposed model avoids any assumption about the F_0 shape.

Some recent studies also used a VAE [59] or auto-encoder [60] for F_0 modeling. However, the VAE-based model still relied on the constraints of the Fujisaki model. More importantly, how a VAE-based F_0 model performs for TTS has not yet been reported. The F_0 auto-encoder only handles fixed-length F_0 contours [60]. Its performance for TTS tasks has not been reported either. Compared with these models, our proposed model is more flexible and has performed well for TTS.

The linker in the proposed model may be extended to jointly predict the duration and latent F_0 codes for each linguistic unit, which may potentially model the dependency between these two acoustic cues of speech prosody. It is also possible to jointly train the linker with the VQ-VAE decoder before F_0 generation, which will be explored in future work.

Our proposed model is by no means the only model to avoid the frame-by-frame processing of linguistic features. Sequence-to-sequence models [61], [62] may also be used for the F_0-modeling task. Compared with a sequence-to-sequence F0 model, however, our proposed model can explicitly use the linguistic boundaries rather than learning the alignment from scratch, which may ease the learning process. It is also interesting to introduce sequence-to-sequence models in the linker part of the proposed model.

VI. Conclusion

We investigated the issue of model efficiency for NN-based F_0 models in TTS systems. Although we previously proposed the DAR, which outperformed an RNN-based F_0 model, this DAR processes the linguistic features frame by frame in the same manner as the RNN-based model. Because frames within the same linguistic unit carry the same input linguistic features, the frame-by-frame processing of linguistic features is unnecessary. It also prevents the model from easily retrieving linguistic features of the neighboring units.

To improve model efficiency, we proposed a VQ-VAE-based F_0 model that consists of two components. The first component is an F_0 contour model that generates the F_0 contour based on sparse latent codes of linguistic units. To jointly learn this component and latent codes, we proposed to use the VQ-VAE framework and estimate the model parameter and codebooks together with additional encoders. Experiments showed that the meaningful latent codes can be learned at the phone and mora levels, and these codes can be used to encode and decode the F_0 contours quite accurately.

Given the latent codes extracted using the VQ-VAE encoder on the training set, the second component of the proposed model, which is referred to as the linker, can be easily trained as a sequential classification model. This linker only converts the linguistic features into a latent code for each linguistic unit. It operates much faster than the layers of a conventional NN-based F_0 model that processes the linguistic feature frame by frame. By combining a linker that predicts mora and phone latent codes and the corresponding VQ-VAE decoder and codebooks, the proposed VQ-VAE-based F_0 model outperformed the DAR in objective tests and performed equally well in subjective tests. The proposed VQ-VAE-based F_0 model is also smaller in model size and faster in training and F_0 contour generation.

References

[1] C. Gussenhoven, *The phonology of tone and intonation*. Cambridge University Press, 2004, pp. xvii–xix.

[2] K. Shigeto, "The phonology of Japanese accent," in *Handbook of Japanese Phonetics and Phonology*. De Gruyter Mouton CY, 2015.

[3] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A generative model for raw audio," *arXiv:1609.03499*, 2016.

[4] 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 *Proc. ICASSP*, 2017, pp. 4779–4783.

[5] W. Ping, K. Peng, A. Gibiansky, S. O. Arik, A. Kannan, S. Narang, J. Raiman, and J. Miller, "Deep voice 3: Scaling text-to-speech with convolutional sequence learning," in *Proc. ICLR*, 2018.

[6] T. Merritt, S. Ronanki, Z. Wu, and O. Watts, "The CSTR entry to the Blizzard Challenge 2016," in *Blizzard Challenge workshop*, 2016.

[7] Y. Hu, C. Ding, L. J. Liu, Z. H. Ling, and L. R. Dai, "The USTC system for Blizzard Challenge 2017," in *Blizzard Challenge Workshop*, 2017.

[8] K. Tokuda, Y. Nankaku, T. Toda, H. Zen, J. Yamagishi, and K. Oura, "Speech synthesis based on hidden Markov models," *Proceedings of the IEEE*, vol. 101, no. 5, pp. 1234–1252, 2013.

[9] H. Zen, K. Tokuda, and A. W. Black, "Statistical parametric speech synthesis," *Speech Communication*, vol. 51, pp. 1039–1064, 2009.

[10] H. Kawahara, I. Masuda-Katsuse, and A. d. Cheveigne, "Restructuring speech representations using a pitch-adaptive time-frequency smoothing and an instantaneous-frequency-based F0 extraction: Possible role of a repetitive structure in sounds," *Speech Communication*, vol. 27, pp. 187–207, 1999.

[11] A. Tamamori, T. Hayashi, K. Kobayashi, K. Takeda, and T. Toda, "Speaker-dependent WaveNet vocoder," in *Proc. Interspeech*, 2017, pp. 1118–1122.

[12] R. Fernandez, A. Rendel, B. Ramabhadran, and R. Hoory, "Prosody contour prediction with long short-term memory, bi-directional, deep recurrent neural networks." in *Proc. Interspeech*, 2014, pp. 2268–2272.

[13] X. Wang, S. Takaki, and J. Yamagishi, "Autoregressive neural F0 model for statistical parametric speech synthesis," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 26, no. 8, pp. 1406–1419, 2018.

[14] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. NIPS*, 2017, pp. 6306–6315.

[15] K. Tokuda, T. Masuko, N. Miyazaki, and T. Kobayashi, "Multi-space probability distribution HMM," *IEICE Trans. on Information and Systems*, vol. 85, no. 3, pp. 455–464, 2002.

[16] K. Yu and S. Young, "Continuous F0 modeling for HMM based statistical parametric speech synthesis," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 5, pp. 1071–1079, 2011.

[17] H. Zen and N. Braunschweiler, "Context-dependent additive log F0 model for HMM-based speech synthesis," in *Proc. Interspeech*, 2009, pp. 2091–2094.

[18] M. Lei, Y. Wu, F. K. Soong, Z. H. Ling, and L. Dai, "A hierarchical F0 modeling method for HMM-based speech synthesis," in *Proc. Interspeech*, 2010, pp. 2170–2173.

[19] J. Latorre and M. Akamine, "Multilevel parametric-base F0 model for speech synthesis." in *Proc. Interspeech*, 2008, pp. 2274–2277.

[20] Y. Qian, Z. Wu, B. Gao, and F. K. Soong, "Improved prosody generation by maximizing joint probability of state and longer units," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, no. 6, pp. 1702–1710, 2011.

[21] L. Gao, Z.-H. Ling, L.-H. Chen, and L.-R. Dai, "Improving F0 prediction using bidirectional associative memories and syllable-level F0 features for HMM-based Mandarin speech synthesis," in *Proc. ISCSLP*. IEEE, 2014, pp. 275–279.

[22] H. Zen, A. Senior, and M. Schuster, "Statistical parametric speech synthesis using deep neural networks," in *Proc. ICASSP*, 2013, pp. 7962–7966.

[23] Y. Fan, Y. Qian, F. Xie, and F. K. Soong, "TTS synthesis with bidirectional LSTM based recurrent neural networks," in *Proc. Interspeech*, 2014, pp. 1964–1968.

[24] M. S. Ribeiro, "Suprasegmental representations for the modeling of fundamental frequency in statistical parametric speech synthesis," Ph.D. dissertation, The University of Edinburgh, 2018.

[25] X. Yin, M. Lei, Y. Qian, F. K. Soong, L. He, Z.-H. Ling, and L.-R. Dai, "Modeling DCT parameterized F0 trajectory at intonation phrase level with DNN or decision tree," in *Proc. Interspeech*, 2014, pp. 2273–2277.

[26] X. Wang, S. Takaki, and J. Yamagishi, "Investigating very deep highway networks for parametric speech synthesis," *Speech Communication*, vol. 96, pp. 1 – 9, 2018.

[27] M. Scordilis and J. Gowdy, "Neural network based generation of fundamental frequency contours," in *Proc. ICASSP*, 1989, pp. 219–222.

[28] C. Traber, "F0 generation with a data base of natural F0 patterns and with a neural network," in *Proc. ESCA Workshop on Speech Synthesis*, 1991, pp. 141–144.

[29] S. H. Chen, S. H. Hwang, and Y. R. Wang, "An RNN-based prosodic information synthesizer for Mandarin text-to-speech," *IEEE Transactions on Speech and Audio Processing*, vol. 6, no. 3, pp. 226–239, 1998.

[30] Y. Sagisaka, "On the prediction of global F0 shape for Japanese text-to-speech," in *Proc. ICASSP*, 1990, pp. 325–328.

[31] C. M. Bishop, *Neural networks for pattern recognition*. Oxford university press, 1995.

[32] ——, "Mixture Density Networks," Aston University, Tech. Rep., 2004. [Online]. Available: http://eprints.aston.ac.uk/373/

[33] M. Schuster, "Better generative models for sequential data problems: Bidirectional recurrent mixture density networks," in *Proc. NIPS*, 1999, pp. 589–595.

[34] K. Dusterhoff and A. W. Black, "Generating f0 contours for speech synthesis using the tilt intonation theory," in *Intonation: Theory, Models and Applications*, 1997.

[35] J. Koutnik, K. Greff, F. Gomez, and J. Schmidhuber, "A Clockwork RNN," in *Proc. ICML*, 2014, pp. 1863–1871.

[36] G. E. Henter, X. Wang, and J. Yamagishi, "Deep encoder-decoder models for unsupervised learning of controllable speech synthesis," *arXiv:1807.11470*, 2018.

[37] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. ICLR*, 2014.

[38] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio, "Generating sentences from a continuous space," in *Proc. CoNLL*, 2016, pp. 10–21.

[39] X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel, "Variational lossy autoencoder," in *Proc. ICLR*, 2017.

[40] K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. B. Pierrehumbert, and J. Hirschberg, "ToBI: a standard for labeling English prosody," in *Proc. ICSLP*, 1992, pp. 867–870.

[41] P. Taylor, "Analysis and synthesis of intonation using the Tilt model," *JASA*, vol. 107, no. 3, pp. 1697–1714, 2000.

[42] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. CVPR*, 2016, pp. 770–778.

[44] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[45] H. Kawai, T. Toda, J. Ni, M. Tsuzaki, and K. Tokuda, "XIMERA: A new TTS from ATR based on corpus-based technologies," in *Proc. SSW5*, 2004, pp. 179–184.

[46] L. Juvela, X. Wang, S. Takaki, S. Kim, M. Airaksinen, and J. Yamagishi, "The NII speech synthesis entry for Blizzard Challenge 2016," in *Blizzard Challenge Workshop*, 2016.

[47] HTS Working Group, "The Japanese TTS System 'Open JTalk'," 2015. [Online]. Available: http://open-jtalk.sourceforge.net/

[48] ——, "An example of context-dependent label format for HMM-based speech synthesis in Japanese," 2015.

[49] M. Morise, F. Yokomori, and K. Ozawa, "WORLD: A vocoder-based high-quality speech synthesis system for real-time applications," *IEICE Trans. on Information and Systems*, vol. 99, no. 7, pp. 1877–1884, 2016.

[50] K. Tokuda, T. Kobayashi, T. Masuko, and S. Imai, "Mel-generalized cepstral analysis a unified approach," in *Proc. ICSLP*, 1994, pp. 1043–1046.

[51] F. Weninger, J. Bergmann, and B. Schuller, "Introducing CURRENNT: The Munich open-source CUDA recurrent neural network toolkit," *The Journal of Machine Learning Research*, vol. 16, no. 1, pp. 547–551, 2015.

[52] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2014.

[53] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. AISTATS*, 2010, pp. 249–256.

[54] X. Wang, J. Lorenzo-Trueba, S. Takaki, L. Juvela, and J. Yamagishi, "A comparison of recent waveform generation and acoustic modeling methods for neural-network-based speech synthesis," in *Proc. ICASSP*, 2018, pp. 4804–4808.

[55] L. v. d. Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, no. Nov, pp. 2579–2605, 2008.

[56] K. E. Dusterhoff, A. W. Black, and P. A. Taylor, "Using decision trees within the Tilt intonation model to predict F0 contours," *Proc. Eurospeech*, pp. 1627–1630, 1999.

[57] K. Hirose, K. Sato, Y. Asano, and N. Minematsu, "Synthesis of F0 contours using generation process model parameters predicted from unlabeled corpora: Application to emotional speech synthesis," *Speech communication*, vol. 46, no. 3, pp. 385–404, 2005.

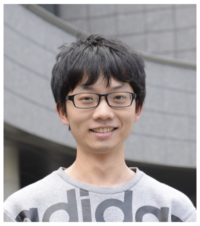
[58] H. Liu, "Fundamental frequency modelling: an articulatory perspective with target approximation and deep learning," Ph.D. dissertation, UCL, 2017.

[59] K. Tanaka, H. Kameoka, and K. Morikawa, "VAE-SPACE: Deep generative model for voice fundamental frequency contours," in *Proc. ICASSP*, 2018, pp. 5779–5793.

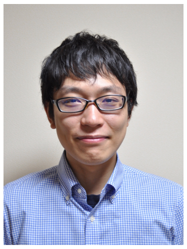
[60] N. Obin and J. Beliao, "Sparse coding of pitch contours with deep auto-encoders," in *Proc. Speech Prosody*, 2018, pp. 799–803.

[61] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. NIPS*, 2014, pp. 3104–3112.

[62] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, 2017, pp. 5998–6008.



Xin Wang (S'16 - M'18) received his Ph.D. degree from Department of Informatics, SOKENDAI / National Institute of Informatics, Japan, 2018. He is now a project researcher at National Institute of Informatics, Japan. His research interests include statistical speech synthesis and machine learning.



Shinji Takaki (M'16) received the B.E. degree in computer science, the M.E. and Ph.D. degrees in scientific and engineering simulation from Nagoya Institute of Technology, Nagoya, Japan in 2009, 2011, and 2014, respectively. From September 2013 to January 2014, he was a visiting researcher at University of Edinburgh. From April 2014, he was a project researcher at National Institute of Informatics. From April 2019, he is a project researcher at Nagoya Institute of Technology. He received the IPSJ Yamashita SIG Research Award, the ASJ Awaya Kiyoshi Award, in 2017 and 2019, respectively. His research interests include statistical machine learning and speech synthesis.

Junichi Yamagishi (SM'13) is a professor at National Institute of Informatics in Japan. He is also a senior research fellow in the Centre for Speech Technology Research (CSTR) at the University of Edinburgh, UK. He was awarded a Ph.D. by Tokyo Institute of Technology in 2006 for a thesis that pioneered speaker-adaptive speech synthesis and was awarded the Tejima Prize as the best Ph.D. thesis of Tokyo Institute of Technology in 2007. Since 2006, he has authored and co-authored over 200 refereed papers in international journals and conferences. He was awarded the Itakura Prize from the Acoustic Society of Japan, the Kiyasu Special Industrial Achievement Award from the Information Processing Society of Japan, and the Young Scientists' Prize from the Minister of Education, Science and Technology, the JSPS prize in 2010, 2013, 2014, and 2016, respectively.

He was one of organizers for special sessions on "Spoofing and Countermeasures for Automatic Speaker Verification" at Interspeech 2013, "ASVspoof evaluation" at Interspeech 2015, "Voice conversion challenge 2016" at Interspeech 2016, and "2nd ASVspoof evaluation" at Interspeech 2017. He has been a member of the Speech & Language Technical Committee (SLTC). He served as an Associate Editor of the IEEE/ACM Transactions on Audio, Speech and Language Processing and a Lead Guest Editor for the IEEE Journal of Selected Topics in Signal Processing (JSTSP) special issue on Spoofing and Countermeasures for Automatic Speaker Verification.

Simon King (M'95–SM'08–F'14) holds M.A.(Cantab) and M.Phil. degrees from Cambridge and a Ph.D. from Edinburgh. He is the director of CSTR and Professor of Speech Processing. His interests include speech synthesis, recognition and signal processing, with over 200 publications in these areas. He currently co-organises the Blizzard Challenge. He has served on the IEEE SLTC and as an associate editor of IEEE Transactions on Audio, Speech and Language Processing and is a current editor of Computer Speech and Language.

Keiichi Tokuda (M'89 - SM'09 - F'14) received the B.E. degree in electrical and electronic engineering from Nagoya Institute of Technology, Nagoya, Japan, the M.E. and Dr.Eng. degrees in information processing from the Tokyo Institute of Technology, Tokyo, Japan, in 1984, 1986, and 1989, respectively.

From 1989 to 1996 he was a Research Associate at the Department of Electronic and Electric Engineering, Tokyo Institute of Technology. From 1996 to 2004 he was a Associate Professor at the Department of Computer Science, Nagoya Institute of Technology as Associate Professor, and now he is a Professor at the same institute. He is also an Honorary Professor at the University of Edinburgh. He was an Invited Researcher at ATR Spoken Language Translation Research Laboratories, Japan from 2000 to 2013 and a Visiting Researcher at Carnegie Mellon University from 2001 to 2002 and at Google from 2014 to 2015.

He published over 90 journal papers and over 200 conference papers, and received six paper awards and four achievement awards, including the ISCA Medal for Scientific Achievement. He was a member of the Speech Technical Committee of the IEEE Signal Processing Society from 2000 to 2003, a member of ISCA Advisory Council and an associate editor of IEEE Transactions on Audio, Speech & Language Processing, and acts as organizer and reviewer for many major speech conferences, workshops and journals. He is a IEEE Fellow and ISCA Fellow. His research interests include speech coding, speech synthesis and recognition, and statistical machine learning.