# Unsupervised acoustic unit discovery for speech synthesis using discrete latent-variable neural networks

*Ryan Eloff*[1*], *André Nortje*[1*], *Benjamin van Niekerk*[1*], *Avashna Govender*[2], *Leanne Nortje*[1],
*Arnu Pretorius*[1], *Elan van Biljon*[1], *Ewald van der Westhuizen*[1], *Lisa van Staden*[1], *Herman Kamper*[1*]

[1]Dept. E&E Engineering, Stellenbosch University, South Africa
[2]CSTR, School of Informatics, University of Edinburgh, UK

kamperh@sun.ac.za

## Abstract

For our submission to the ZeroSpeech 2019 challenge, we apply discrete latent-variable neural networks to unlabelled speech and use the discovered units for speech synthesis. Unsupervised discrete subword modelling could be useful for studies of phonetic category learning in infants or in low-resource speech technology requiring symbolic input. We use an autoencoder (AE) architecture with intermediate discretisation. We decouple acoustic unit discovery from speaker modelling by conditioning the AE's decoder on the training speaker identity. At test time, unit discovery is performed on speech from an unseen speaker, followed by unit decoding conditioned on a known target speaker to obtain reconstructed filterbanks. This output is fed to a neural vocoder to synthesise speech in the target speaker's voice. For discretisation, categorical variational autoencoders (CatVAEs), vector-quantised VAEs (VQ-VAEs) and straight-through estimation are compared at different compression levels on two languages. Our final model uses convolutional encoding, VQ-VAE discretisation, deconvolutional decoding and an FFTNet vocoder. We show that decoupled speaker conditioning intrinsically improves discrete acoustic representations, yielding competitive synthesis quality compared to the challenge baseline.

**Index Terms**: unsupervised speech processing, zero-resource challenge, acoustic unit discovery, low-resource speech synthesis

## 1. Introduction

For many languages, it is difficult or impossible to collect the annotated resources required for training supervised automatic speech recognition (ASR) models [1]. Zero-resource speech processing aims to develop methods that can learn directly from unlabelled speech audio. A central problem is finding frame-level feature representations that capture meaningful linguistic contrasts, such as phonetic categories, while being invariant to factors such as a speaker's identity, gender or accent [2–5]. As part of the 2015 and 2017 Zero Resource Speech Challenges (ZRSC), several unsupervised methods were proposed for learning *continuous* acoustic representations [6–11]. Here we consider *discrete* representation learning, also referred to as 'acoustic unit discovery' [12, 13] or 'discrete subword learning' [14]. From a scientific perspective, discrete representations could be useful in cognitive models to study phonetic category learning in human infants [15–18]. From a technology perspective, such features could be used in downstream speech applications requiring symbolic or sparse input, e.g., for faster retrieval in speech search systems [19, 20]. Here we consider the downstream task of speech synthesis within the context of the ZRSC'19.

Specifically, the task is to perform acoustic unit discovery on an unlabelled speech collection, to build a synthesiser in a target speaker's voice using the discovered units, and to then synthesise speech from unseen speakers in the target speaker's voice. This is related to voice conversion [21, 22], where speech from one speaker needs to be synthesised in another speaker's voice. But, importantly, the task here involves unit discovery which compresses spoken input [23, 24]. The trade-off between the level of compression and intelligibility is explicitly evaluated.

For unit discovery, we use an autoencoder (AE) neural network architecture, which compresses its input and then reconstructs it from a latent layer. AEs are well-suited to our task since compression can be directly controlled through the latent layer dimensionality [14, 25, 26]. Unfortunately, naive discretisation within AEs is not possible since gradient updates cannot be calculated directly. We build on a number of recent stochastic methods which allows these gradients to be estimated [27–32].

Concretely, we investigate different unsupervised discrete latent-variable neural networks for acoustic unit discovery and subsequent speech synthesis. Our AE architecture takes Mel-frequency cepstral coefficients (MFCCs) as input, performs intermediate discretisation to find discrete units, and produces log-Mel filterbank features as output. These are fed to an autoregressive neural vocoder, producing synthesised speech. During evaluation, the goal is to synthesise speech in a target speaker's voice given input from an unseen speaker. In order to separate subword from speaker modelling, we feed a speaker's identity to the AE's decoder module during training. Speaker conditioning is not used in the AE's encoder and discretisation steps, so discrete units can be obtained for an arbitrary speaker. At test time, units are obtained for an unseen test speaker; these units are then decoded by conditioning on the target speaker's voice using the combination of the AE's decoder and the vocoder (trained on the target speaker's voice). In contrast to the feed-forward AEs of [14, 25, 26], we incorporate temporal context by using a convolutional encoder and deconvolutional decoder in our AE.

A similar model using a vector-quantised variational autoencoder (VQ-VAE) was recently proposed in [33]. Our approach differs from this model in several ways, corresponding to the main contributions of our work. Firstly, we do not train our model end-to-end, but rather train separate compression and vocoding models. This is not necessarily superior, but it has benefits. E.g., simpler/more complex vocoders could be used with the same compression model depending on computational resources. Secondly, we specifically consider the impact of decoupling unit discovery from speaker modelling. We show that this produces better discrete latent features in an intrinsic cross-speaker evaluation task. Thirdly, we consider different discretisation methods; we show that the categorical VAE (Cat-VAE) [30, 31] and straight-through estimation (STE) [27–29] both perform competitively to the VQ-VAE. Finally, we evaluate

---

speech synthesis performance using human evaluations as part of ZRSC'19 on both English and Indonesian data.

## 2. Unsupervised discrete representation learning for speech synthesis

A corpus of unlabelled speech from multiple speakers in a single language is used for training our discrete latent-variable neural networks. A known training-set speaker is specified as the target voice in which to synthesise speech. At test time, the proposed system is provided with new utterances from unseen speakers. For each test utterance, the system performs unit discovery on the input, producing a transcription-like encoding using the learned discrete symbols. The test utterance is then re-synthesised in the target speaker's voice based on these symbols [34].

Our specific approach is shown in Figure 1. The system takes MFCCs with deltas and double-deltas as input, $\mathbf{x}_{1:T} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$, with each $\mathbf{x}_t \in \mathbb{R}^{39}$. These are encoded into latent continuous representations $\mathbf{h}_{1:N}$, which are discretised into symbolic representations $\mathbf{z}_{1:N}$. The sequence of symbols $\mathbf{z}_{1:N}$ preserves time-information at a fixed rate, with $N$ depending on the length of the input $T$ and the amount of downsampling in the encoder (§2.4). A symbol $\mathbf{z}$ takes on a different form depending on the discretisation method: it can be binarised (STE), one-hot (CatVAE), or an embedding selected from a codebook (VQ-VAE). But, importantly, the input is encoded into a finite set of symbols. These symbols are decoded to 45-dimensional filterbanks $\mathbf{y}_{1:N}$ using a decoder module. Since the encoder input and the decoder output is obtained from the same original waveform, we refer to this compression model as an autoencoder (AE).

At training time, the AE's decoder is conditioned on the input speaker identity. This means that, in principle, the encoder does not need to preserve speaker-specific information, thereby decoupling acoustic unit discovery from speaker modelling. For this same reason we use MFCCs, which tend to be speaker-independent [35], as input, while fitlerbanks, which retain speaker-specific properties, are used as intermediate output. We also experimented with filterbank input, but MFCCs worked better. The encoder and discretiser are speaker-independent and can therefore be applied to unseen speakers at test time. The decoder is then conditioned on the target speaker and its output is provided to an FFTNet vocoder [36], trained only on the target speaker, to obtain synthesised speech in the desired voice. Voice conversion therefore occurs in the AE's decoder, not the vocoder.

The whole system can be trained without parallel data. In contrast to the end-to-end methodology of [33], we train the compression model separately from the vocoder. The combination of the AE's decoder with the vocoder could thus be described as a symbol-to-speech module, as shown in Figure 1.

Building on recent advances in discrete representations in neural networks, we use three discretisation strategies to convert the encoder output $\mathbf{h}$ to a discrete latent representation $\mathbf{z}$.[1]

### 2.1. Straight-through estimation (STE) binarisation

A direct way to discretise would be to binarise the continuous encoder output vectors. If $\mathbf{h}$ is a $K$-dimensional continuous vector obtained through a $\mathtt{tanh}$ activation, i.e. $\mathbf{h} \in [-1, 1]^K$, each dimension of $\mathbf{z}$ could be set so that $z_k = 1$ if $h_k \geq 0$ or $z_k = -1$ otherwise. The result would be a binarised vector $\mathbf{z} \in \{-1, 1\}^K$. The gradients required for back-propagating

---

[1] Here, $\mathbf{h}$ refers to a single latent vector in the sequence $\mathbf{h}_{1:N}$. The scalar $h_k$ is the $k^{\text{th}}$ dimension of $\mathbf{h}$. The same convention is used for $\mathbf{z}$.
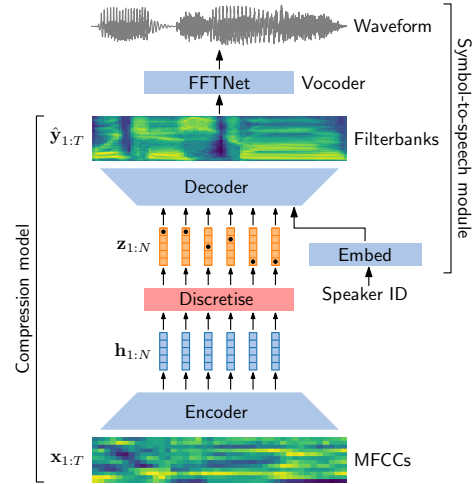


Figure 1: *Discrete latent-variable neural networks are used for unit discovery and speech synthesis. At training time, the speaker ID matches the input. At test time it is set to a target speaker. The compression model and vocoder is trained separately.*

through such a thresholding operation cannot be directly derived, so we use the straight-through estimation (STE) method [28, 29]. During training, binarisation is performed stochastically with $z_k = h_k + \epsilon$, where $\epsilon$ is sampled noise:

$$\epsilon = \begin{cases} 1 - h_k & \text{with probability } \frac{1+h_k}{2} \\ -h_k - 1 & \text{with probability } \frac{1-h_k}{2} \end{cases}$$

Since $\epsilon$ is zero-mean, the derivative of the expected value of $z_k$ is $\frac{\partial \mathbb{E}[z_k]}{\partial h_k} = 1$, so gradients are passed unchanged through the binarisation layer: $\frac{\partial L}{\partial \mathbf{h}} \approx \frac{\partial L}{\partial \mathbf{z}}$. We use a squared loss $L$.

### 2.2. Vector-quantised variational autoencoder (VQ-VAE)

The vector-quantised variational autoencoder (VQ-VAE) [32] maintains a codebook of prototype embedding vectors $\{\mathbf{e}_k\}_{k=1}^K$. During inference, the encoder output $\mathbf{h}$ is used to select the closest codebook embedding to serve as the quantised representation, i.e., $\mathbf{z} = \mathbf{e}_k$ where $k = \text{argmin}_j ||\mathbf{h} - \mathbf{e}_j||^2$. Once again, gradients cannot be derived for this operation, so we approximate it as $\frac{\partial L}{\partial \mathbf{h}} \approx \frac{\partial L}{\partial \mathbf{z}}$. VQ-VAE discretisation uses a loss $L$ composed of three terms. The first is the negative log likelihood, which reduces to a reconstruction loss when assuming a spherical Gaussian output distribution: $-\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:T}|\mathbf{z}_{1:N}) = -\sum_{t=1}^T \left( c - \frac{1}{2\sigma^2} ||\mathbf{y}_t - \hat{\mathbf{y}}_t||^2 \right)$, with $\boldsymbol{\theta}$ denoting the decoder parameters, $\mathbf{y}_t$ the target filterbanks, and $\hat{\mathbf{y}}_t$ the predicted decoder output. The second loss term, $||\text{sg}(\mathbf{h}) - \mathbf{z}||^2$, updates the embeddings in the codebook, with $\text{sg}(\cdot)$ denoting the stop-gradient operation. Third, a 'commitment loss' encourages the encoder output $\mathbf{h}$ to lie close to a codebook embedding: $\beta||\mathbf{h} - \text{sg}(\mathbf{z})||^2$. The three terms can be weighed by changing the hyperparameters $\sigma$ and $\beta$. We use $\sigma = 10^{-6}$ and $\beta = 25$.

### 2.3. Categorical variational autoencoder (CatVAE)

As in a standard VAE [37], the categorical VAE (CatVAE) [30, 31] optimises a lower bound for $\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:T})$ using an encoder $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_{1:T})$ to approximate the intractable posterior $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x}_{1:T})$, with $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ denoting the encoder and decoder weights, respectively. In a standard VAE, $\mathbf{z}$ is drawn from a multivariate Gaussian using the reparametrisation trick. For the CatVAE, $\mathbf{z}$ is a one-hot vector sampled from a categorical distribution, which does not have a direct reparametrised form.

The Gumbel-softmax distribution approximates the categorical distribution and can be reparametrised [30]. It is defined as $P(z_k) = \frac{\exp\{(\log \pi_k + g_k)/\tau\}}{\sum_{k=1}^{K} \exp\{(\log \pi_k + g_k)/\tau\}}$ where $g_1, g_2, \ldots, g_K$ are samples from a $\text{Gumbel}(0, 1)$ distribution and $\tau$ is a temperature parameter. Sampling from this distribution with small $\tau$ resembles a $K$-component categorical distribution with weights $\boldsymbol{\pi} = \{\pi_k\}_{k=1}^{K}$. Our AE encoder outputs these weights $\mathbf{h} = \log \boldsymbol{\pi}$.

The model is trained using the evidence lower bound for $\log p_{\boldsymbol{\theta}}(\mathbf{y}_{1:T})$, which reduces to a reconstruction term (the same as for the VQ-VAE) and a Kullback-Leibler (KL) regularisation term. This KL term encourages the latent features $\mathbf{z}$ to be similar to a prior $p(\mathbf{z})$, chosen as the uniform categorical distribution. The term can thus be optimised directly since it is the KL divergence between two categorical distributions: the one predicted for $\mathbf{z}$, with mass $\boldsymbol{\pi}$ given by the encoder, and the uniform prior. During training, we anneal $\tau$ linearly from 1 to 0.1.

### 2.4. Neural network architectures

Each of the above methods are used as the discretisation layer in Figure 1. Our goal is to compress input speech into a discrete set of symbols which can be used for speech synthesis. The degree of compression (the bitrate) can be controlled by changing the number of unique symbols, set through $K$. To further reduce bitrate, the encoder architecture can downsample the input, producing discrete features at a fixed but lower rate than the input.

Specifically, all our models use a first pre-processing convolutional layer without downsampling. This is followed by a convolutional layer with a stride of 2, producing output at half the rate of its input. This strided convolutional layer can be repeated, each downsampling by a factor of 2. If not specified, we use two such layers, i.e. for MFCC input with $T$ frames, the model produces $N = T/4$ discrete vectors. For decoding, transposed convolutions mirror the convolutions in the encoder. A linear output layer produces the $T$-frame filterbank output. We also experimented with recurrent architectures, but the convolutional approach proved superior and was faster to train.

Based on development experiments on English data (§3), we chose the same architecture for VQ-VAE and CatVAE discretisation, but a slightly different architecture for STE. For the STE model, we use convolutional gated recurrent units (ConvGRU) [38] to downsample. A ConvGRU is a type of recurrent cell which applies convolutions over its input. Rather than straight-forward transposed convolutions, our STE model also uses the more efficient pixel shuffling operation [39]. The VQ-VAE/CatVAE models use residual convolutions with batch normalisation and standard transposed convolutions for decoding. In the STE model, the decoder is conditioned on a 250-dimensional speaker embedding, while the VQ-VAE/CatVAE models use a speaker dimensionality of 128; these are trained jointly with the rest of the model. We use Adam optimisation [40]. Other architectural choices are given in our code repository.[2]

For speech synthesis, we use FFTNet as a vocoder (Figure 1 top). FFTNet is an auto-regressive neural model which generates raw audio waveforms given filterbanks as input. We also experimented with WaveNet [41], which produced slighly better quality but required much longer training times and occasionally generated noise. Our final model consists of a stack of 11 FFTNet layers with 256 channels each, giving a receptive field of 2048 samples. We quantise the raw waveforms using $\mu$-law encoding and a softmax over 256 classes to model the distribution over the next sample. We train FFTNet on clean filterbanks,

extracted directly from the target speaker. We also tried to train directly on the output of the different AEs, but this performed worse. Further work is needed to understand this observation.

## 3. Experimental setup and evaluation

We perform experiments on two languages. We compare models on English data, and then apply the best models unaltered on the ZRSC'19 evaluation language, Indonesian, a low-resource Austronesian language widely used as a lingua franca [42, 43]. For both languages, training data for acoustic unit discovery consists of around 15 h from 100 speakers. Data from a target voice is used both for acoustic discovery and for training the vocoder. Two English and one female Indonesian target voice dataset of around 2 h each are used. Test data consists of 30 min in both English and Indonesian from 24 and 15 speakers, respectively.

As an intrinsic measure of discriminability, we use the ABX task [44]. Using a particular feature representation, ABX asks whether a triphone $X$ is more similar to triphone $A$ or triphone $B$, where $A$ and $X$ are different instances of the same triphone (e.g. 'beg') and $B$ differs in the middle phone (e.g. 'bag'). To explicitly measure invariance to speaker, $A$ and $B$ comes from the same speaker, while $X$ comes from another. As a distance metric, the average frame-wise cosine distance along the dynamic time warping (DTW) alignment path is used. ABX is reported as an aggregated error rate over minimal pairs.

To measure the degree of compression, bitrate is calculated as $\frac{M \sum_{m=1}^{M} P(\mathbf{z}_m) \log_2 P(\mathbf{z}_m)}{D}$, where $M$ is the total number of symbols in the test data, $P(\mathbf{z}_m)$ is the estimated probability of symbol $\mathbf{z}_m$, and $D$ is the total duration of the data in seconds. To measure synthesis quality, human judges transcribe the synthesised speech. By comparing this to the ground truth, a character error rate (CER) is calculated. Humans also rate similarity to the target voice and give a mean opinion score (MOS), both on a scale from 1 to 5 (higher is better).

The ZRSC'19 baseline system uses a pipeline of a Dirichlet process Gaussian mixture model (DPGMM) for acoustic unit discovery [13], and the Merlin neural network speech synthesiser [45] trained on the discovered units from the target speaker. A topline system feeds output from a supervised ASR system to a supervised speech synthesiser, both trained on transcriptions.

## 4. Experimental results

**Discretisation method comparison.** The 'ABX on latent' column in Table 1 compares the different discretisation methods on the English test data, with ABX performed directly on the discrete symbols. $K$ is set so that 512 unique symbols can be obtained in all models, but not all the symbols are necessarily utilised, so the bitrates differ. The form of the latent symbols $\mathbf{z}$ is very different for the different methods: binarised for STE, one-hot for CatVAE, or an embedding selected from a codebook for VQ-VAE. The cosine-DTW distance metric used in ABX might not be equally well-suited to all of these. To make a fairer comparison, we pass the encoded symbols through the decoder for each model, and compare all the models on the reconstructed filterbank $\hat{\mathbf{y}}$ obtained at the output from each AE.[3] All methods are therefore compared based on the same feature type. ABX scores based on decoder outputs with and without speaker conditioning are shown in the second and third columns of Table 1. In all cases, VQ-VAE performs best. We also observe that the

---

[3]One way to interpret this evaluation of the symbolic features is to see the decoder as part of the ABX distance metric for each of the methods.

Table 1: *ABX (%) on English for different discrete latent-variable neural models with and without speaker conditioning.*

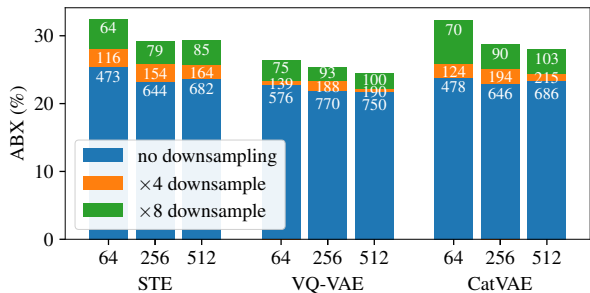| Model | ABX on decoder output | | ABX on latent | Bitrate |
| | No spkr cond. | Spkr cond. | | |
| --- | --- | --- | --- | --- |
| STE | 27.5 | 26.4 | 31.5 | 116 |
| VQ-VAE | 26.0 | 22.1 | 28.6 | 190 |
| CatVAE | 28.7 | 24.3 | 45.6 | 215 |
| Filterbanks | - | - | 29.5 | 1735 |
| MFCCs | - | - | 22.7 | 1738 |



Figure 2: *ABX on English at different downsampling levels and numbers of symbols (x-axis labels). Models are evaluated on their decoder outputs. Bitrates are shown in white on the bars.*

CatVAE's one-hot latent symbols are particularly ill-suited to direct ABX evaluation: 45.6% error rate compared to 24.3% and 28.7% when using decoder outputs.

**Speaker conditioning.** Our main hypothesis is that decoupling unit discovery from speaker modelling produces intrinsically better discrete representations. A comparison of the columns with and without speaker conditioning in Table 1 shows that, when speaker conditioning is not used, ABX is worse for all models. For the last two rows in the Table 1, ABX was performed directly on MFCCs (inputs) and filterbanks (target outputs). Our best overall result of 22.1% when using the VQ-VAE is better than both of these. This is particularly noteworthy since this model outperforms both the input and output features on which it is trained, and it does so while performing intermediate discretisation which encodes the input at a much lower bitrate. Discretisation on its own (ABX of 26.0% for VQ-VAE) already leads to improvements over filterbanks (29.5%), but it is the combination of discretisation and decoupled speaker conditioning which leads to the best score (22.1%).

**Results across different bitrates.** As noted in §2.4, there are two ways to control the level of compression: $K$ can be varied, which sets the number of unique symbols, or strided convolutions in the model's encoder can be used to perform downsampling. Figure 2 shows ABX scores on English as we vary the number of symbols (64, 256, 512) and the downsampling factor ($\times 1$, $\times 4$ and $\times 8$) for the three discretisation methods. The general trend is that ABX improves when using more symbols and/or less downsampling, which is expected since this results in higher bitrates. Another trend is that it is typically more beneficial to use downsampling to reduce the bitrate rather than reducing the symbols. E.g., the difference between the 256- and 512-symbol STE bars are smaller than the differences within each bar (when downsampling is changed).

**Final results and speech synthesis.** Based on the development experiments on English, we submitted two versions of the VQ-VAE model for official evaluation to the ZRSC'19: the one

Table 2: *Human and machine ZRSC'19 evaluation on the English and Indonesian test data. Higher MOS and similarity scores are better, while lower is better for the other metrics.*

| Model | CER (%) | MOS [1, 5] | Similarity [1, 5] | ABX (%) | | Bitrate |
| | | | | latent | output | |
| --- | --- | --- | --- | --- | --- | --- |
| *English:* | | | | | | |
| DPGMM-Merlin | 75 | 2.50 | 2.97 | 35.6 | - | 72 |
| VQ-VAE-x8 | 75 | 2.31 | 2.49 | 30.8 | 25.1 | 88 |
| VQ-VAE-x4 | 67 | 2.18 | 2.51 | 27.6 | 23.0 | 173 |
| Supervised | 44 | 2.77 | 2.99 | 29.9 | - | 38 |
| *Indonesian:* | | | | | | |
| DPGMM-Merlin | 62 | 2.07 | 3.41 | 27.5 | - | 75 |
| VQ-VAE-x8 | 58 | 1.94 | 1.95 | 26.5 | 17.6 | 69 |
| VQ-VAE-x4 | 60 | 1.96 | 1.76 | 19.8 | 14.5 | 140 |
| Supervised | 28 | 3.92 | 3.95 | 16.1 | - | 35 |

uses $\times 4$ and the other $\times 8$ downsampling. We did this to measure the impact of bitrate on speech synthesis quality. Results on the English and Indonesian test data are reported in Table 2, together with the DPGMM-Merlin baseline and the supervised top-line. Our approach performs worse on MOS and similarity compared to DPGMM-Merlin. On CER, however, we perform similarly or better in all cases. In terms of ABX directly on the discrete latent features, both VQ-VAE-x8 and VQ-VAE-x4 outperform the baseline—on English the latter even outperforms the supervised topline system (27.6% compared to 29.9% in ABX).

We also submitted reconstructed filterbanks $\hat{y}$ as auxiliary features to ZRSC'19, with the results reported in the 'output' column in Table 2. As in Table 1, these features give better scores than when evaluating on the symbols directly. On English, output features from both our models outperform the supervised topline system while on Indonesian the VQ-VAE-x4 model does. These results suggests that the main bottleneck in our overall approach is not in the AE compression model (Figure 1), but in the vocoder. We use FFTNet mainly for computational reasons. Although not conclusive, a comparison of our English MOS ($\sim$2.3) to the MOS for FFTNet reported in [36] ($\sim$3.2) might indicate that we are reaching a performance ceiling with this vocoder.

## 5. Conclusions and future work

We have proposed and evaluated different discrete latent-variable neural network models for unsupervised acoustic unit discovery and subsequent speech synthesis. A convolutional encoder is used and acoustic unit discovery is decoupled from speaker modelling by conditioning a deconvolutional decoder on the training speaker. At test time, the decoder is conditioned on a target speaker. In terms of intelligibility as measured by human character error rate, our models perform similarly or better than the ZRSC'19 baseline system. In an intrinsic evaluation of feature discriminability and speaker invariance, our discrete symbolic representation outperform the baseline by around 50% relative, even outperforming a supervised topline system. Experiments indicate that synthesis quality is impaired most by the neural vocoder, which is fed with reconstructed filterbanks from our decoder. In future work, we plan to consider more complex neural vocoders such as WaveNet [41], and to compare our approach to a fully end-to-end model.

# 6. References

[1] L. Besacier, E. Barnard, A. Karpov, and T. Schultz, "Automatic speech recognition for under-resourced languages: A survey," *Speech Commun.*, vol. 56, pp. 85–100, 2014.

[2] Y. Zhang and J. R. Glass, "Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriorgrams," in *Proc. ASRU*, 2009.

[3] A. Jansen *et al.*, "A summary of the 2012 JHU CLSP workshop on zero resource speech technologies and models of early language acquisition," in *Proc. ICASSP*, 2013.

[4] A. Jansen, S. Thomas, and H. Hermansky, "Weak top-down constraints for unsupervised acoustic model training," in *Proc. ICASSP*, 2013.

[5] W.-N. Hsu, Y. Zhang, and J. R. Glass, "Unsupervised learning of disentangled and interpretable representations from sequential data," in *Proc. NIPS*, 2017.

[6] D. Renshaw, H. Kamper, A. Jansen, and S. J. Goldwater, "A comparison of neural network methods for unsupervised representation learning on the Zero Resource Speech Challenge," in *Proc. Interspeech*, 2015.

[7] R. Thiollière, E. Dunbar, G. Synnaeve, M. Versteegh, and E. Dupoux, "A hybrid dynamic time warping-deep neural network architecture for unsupervised acoustic modeling," in *Proc. Interspeech*, 2015.

[8] M. Versteegh, X. Anguera, A. Jansen, and E. Dupoux, "The Zero Resource Speech Challenge 2015: Proposed approaches and results," in *Proc. SLTU*, 2016.

[9] E. Dunbar *et al.*, "The Zero Resource Speech Challenge 2017," in *Proc. ASRU*, 2017.

[10] M. Heck, S. Sakti, and S. Nakamura, "Feature optimized DPGMM clustering for unsupervised subword modeling: A contribution to ZeroSpeech 2017," in *Proc. ASRU*, 2017.

[11] T. Tsuchiya, N. Tawara, T. Ogawa, and T. Kobayashi, "Speaker invariant feature extraction for zero-resource languages with adversarial learning," in *Proc. ICASSP*, 2018.

[12] C.-y. Lee, T. O'Donnell, and J. R. Glass, "Unsupervised lexicon discovery from acoustic input," *Trans. ACL*, vol. 3, pp. 389–403, 2015.

[13] L. Ondel, L. Burget, and J. Černocký, "Variational inference for acoustic unit discovery," *Procedia Comput. Sci.*, vol. 81, pp. 80–86, 2016.

[14] L. Badino, A. Mereta, and L. Rosasco, "Discovering discrete subword units with binarized autoencoders and hidden-Markov-model encoders," in *Proc. Interspeech*, 2015.

[15] N. H. Feldman, T. L. Griffiths, and J. L. Morgan, "Learning phonetic categories by learning a lexicon," in *Proc. CCSS*, 2009.

[16] O. J. Räsänen, "Computational modeling of phonetic and lexical learning in early language acquisition: Existing models and future directions," *Speech Commun.*, vol. 54, pp. 975–997, 2012.

[17] T. Schatz and N. H. Feldman, "Neural network vs. hmm speech recognition systems as models of human cross-linguistic phonetic perception," in *Proc. CCN*, 2018.

[18] C. Shain and M. Elsner, "Measuring the perceptual availability of phonological features during language acquisition using unsupervised binary stochastic autoencoders," in *Proc. HLT-NAACL*, 2019.

[19] K. Levin, A. Jansen, and B. Van Durme, "Segmental acoustic indexing for zero resource keyword search," in *Proc. ICASSP*, 2015.

[20] S. Settle, K. Levin, H. Kamper, and K. Livescu, "Query-by-example search with discriminative neural acoustic word embeddings," in *Proc. Interspeech*, 2017.

[21] A. Kain and M. W. Macon, "Spectral voice conversion for text-to-speech synthesis," in *Proc. ICASSP*, 1998.

[22] J.-c. Chou, C.-c. Yeh, H.-y. Lee, and L.-s. Lee, "Multi-target voice conversion without parallel data by adversarially learning disentangled audio representations," in *Proc. Interspeech*, 2018.

[23] P. K. Muthukumar and A. W. Black, "Automatic discovery of a phonetic inventory for unwritten languages for statistical speech synthesis," in *Proc. ICASSP*, 2014.

[24] O. Scharenborg *et al.*, "Linguistic unit discovery from multi-modal inputs in unwritten languages: Summary of the "Speaking Rosetta" JSALT 2017 Workshop," in *Proc. ICASSP*, 2018.

[25] M. D. Zeiler *et al.*, "On rectified linear units for speech processing," in *Proc. ICASSP*, 2013.

[26] L. Badino, C. Canevari, L. Fadiga, and G. Metta, "An auto-encoder based approach to unsupervised learning of subword units," in *Proc. ICASSP*, 2014.

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

[28] T. Raiko, M. Berglund, G. Alain, and L. Dinh, "Techniques for learning binary stochastic feedforward neural networks," in *Proc. ICLR*, 2015.

[29] G. Toderici *et al.*, "Variable rate image compression with recurrent neural networks," in *Proc. ICLR*, 2016.

[30] E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with gumbel-softmax," in *Proc. ICLR*, 2017.

[31] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," in *Proc. ICLR*, 2017.

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

[33] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using WaveNet autoencoders," *arXiv preprint arXiv:1901.08810*, 2019.

[34] E. Dunbar *et al.*, "The Zero Resource Speech Challenge 2019: TTS without T," in *Submitted to Interspeech*, 2019.

[35] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE Trans. Acoust., Speech, Signal Process.*, vol. 28, no. 4, pp. 357–366, 1980.

[36] Z. Jin, A. Finkelstein, G. J. Mysore, and J. Lu, "FFTNet: A real-time speaker-dependent neural vocoder," in *Proc. ICASSP*, 2018.

[37] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[38] G. Toderici *et al.*, "Full resolution image compression with recurrent neural networks," in *Proc. CVPR*, 2017.

[39] W. Shi *et al.*, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. CVPR*, 2016.

[40] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[41] A. van den Oord *et al.*, "WaveNet: a generative model for raw audio," *arXiv preprint arXiv:1609.03499*, 2016.

[42] S. Sakti, R. Maia, S. Sakai, T. Shimizu, and S. Nakamura, "Development of HMM-based Indonesian speech synthesis," in *Proc. O-COCOSDA*, 2008.

[43] S. Sakti, E. Kelana, H. Riza, S. Sakai, K. Markov, and S. Nakamura, "Development of Indonesian large vocabulary continuous speech recognition system within A-STAR project," in *Proc. TCAST*, 2008.

[44] T. Schatz, V. Peddinti, F. Bach, A. Jansen, H. Hermansky, and E. Dupoux, "Evaluating speech features with the minimal-pair ABX task: Analysis of the classical MFC/PLP pipeline," in *Proc. Interspeech*, 2013.

[45] Z. Wu, O. Watts, and S. King, "Merlin: An open source neural network speech synthesis system." in *Proc. SSW*, 2016.