INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Unsupervised Neural-based Graph Clustering for Variable-Length Speech Representation Discovery of Zero-Resource Languages

Shun Takahashi[1], *Sakriani Sakti*[1,2], *and Satoshi Nakamura*[1,2]

[1]Nara Institute of Science and Technology, Japan
[2]RIKEN, Center for Advanced Intelligence Project (AIP), Japan
{takahashi.shun.tp0, ssakti, s-nakamura}@is.naist.jp

Abstract

Discovering symbolic units from unannotated speech data is fundamental in zero resource speech technology. Previous studies focused on learning fixed-length frame units based on acoustic features. Although they achieve high quality, they also suffer from a high bit-rate due to time-frame encoding. In this work, to discover variable-length, low bit-rate speech representation from a limited amount of unannotated speech data, we propose an approach based on graph neural networks (GNNs), and we study the temporal closeness of salient speech features. Our approach is built upon vector-quantized neural networks (VQNNs), which learn discrete encoding by contrastive predictive coding (CPC). We exploit the predetermined finite set of embeddings (a codebook) used by VQNNs to encode input data. We consider a codebook a set of nodes in a directed graph, where each arc represents the transition from one feature to another. Subsequently, we extract and encode the topological features of nodes in the graph to cluster them using graph convolution. By this process, we can obtain coarsened speech representation. We evaluated our model on the English data set of the ZeroSpeech 2020 challenge on Track 2019. Our model successfully drops the bit rate while achieving high unit quality.
Index Terms: Zero resource, speech representation learning, graph neural networks, graph clustering

1. Introduction

Spoken language technology applications and services are only available to about 100 of the world's languages. This is because most state-of-the-art speech processing technologies were built using costly supervised learning methods that rely on a massive amount of parallel speech and orthographic/linguistic transcriptions. However, the assumption that such information is available is not always valid for the world's languages in practice. Since many languages have no standardized written form, little to no access exists to reliable transcribed data.

On the other hand, human infants spontaneously learn discrete speech representation without any supervision and develop language skills. Just as they acquire languages, zero-resource speech processing develops systems that directly learn spoken languages directly from raw sensory data in an unsupervised manner. The *Zero Resource Speech Challenge (ZeroSpeech)* series [1, 2, 3, 4] addresses this demanding task and offers a set of shared tasks, data sets, and evaluation methods to allow for comparison among various approaches.

The top performances in speech representation discovery in ZeroSpeech 2015 [1] and 2017 [2] were achieved by Bayesian models such as the Dirichlet process Gaussian mixture model (DPGMM) [5, 6], which can deal with unknown model complexity. But it is susceptible to acoustic variations and often produces too many subword units and a relatively high-dimensional posteriogram [7].

In ZeroSpeech 2019 [3] and 2020 [4], the paradigm shifted, and many studies explored vector quantization neural networks that map continuous speech features onto a finite set of discrete representations [8, 9, 10, 11, 12]. The best-performing models used vector quantized neural networks for acoustic unit discovery, incorporating either a variational autoencoder (VQ-VAE) [13] or a contrastive predictive coding (VQ-CPC) [14].

However, the models proposed so far focused on learning speech representation by classifying acoustic features at each fixed-length frame-based unit. As most speech features of a single phoneme consist of several unit frames, such approaches cause redundancy of the learned representation, which is indicated by the high bit-rate.

This work addresses the high bit-rate problem by proposing a novel, neural-graph clustering that is applied on top of the discrete autoencoder model. In other words, the approach will learn to discover, so to speak, a coarser or more abstract form of speech representation that may span over the time frames. We specifically incorporate graph clustering based on topology-adaptive graph convolutional networks (TAGCN) [15] and deep modularity networks (DMoN) [16] into vector-quantized contrastive predictive coding (VQCPC) [12].

2. Related Works

Recently, the use of graph-based neural network models has gained increasing attention. It is a subtype of neural networks that operate on data structured as graphs. Unlike standard neural networks, Graph Neural Networks (GNNs) retain a state representing information from their neighborhood with arbitrary depth.

Some studies have already shown the advantages of utilizing GNNs on text and speech processing tasks. Defferrard et al. [17] addressed a text classification task as node classification by using graph convolutional neural networks (GCNs) [18] and successfully outperformed the traditional convolutional neural network (CNN) models. Bastings et al. [19] also proposed GCN-based encoders for syntax-aware neural machine translation. In speech processing, GNN approaches have been applied to solve several tasks, including speech separation [20], speech enhancement [21], conversational emotion recognition [22], and even text-to-speech synthesis [23, 24].

However, to the best of our knowledge, no study has applied the GNN framework to the acoustic unit discovery of zero-resource languages. Furthermore, most existing works only explored these mechanisms within a semi-supervised or supervised framework. Our study focuses on discovering variable-length acoustic speech representation with unsupervised learning based on TAGCN [15] and DMoN [16]. The motivation might resemble a recent work [25], although instead of seg-

http://dx.doi.org/10.21437/Interspeech.2021-1340

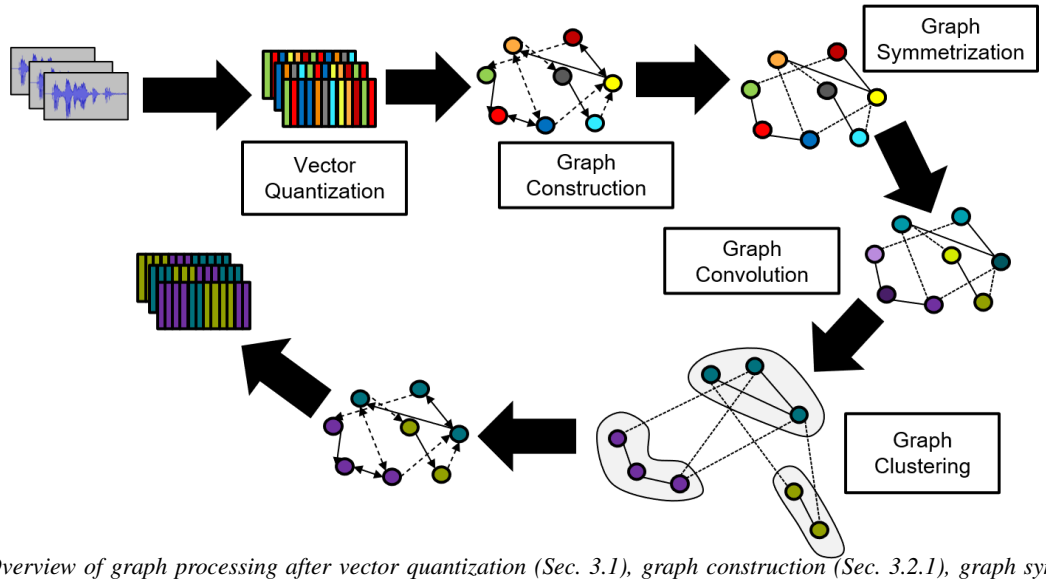

Figure 1: *Overview of graph processing after vector quantization (Sec. 3.1), graph construction (Sec. 3.2.1), graph symmetrization (Sec. 3.2.2), graph convolution (Sec. 3.2.3), graph clustering (Sec. 3.2.3).*

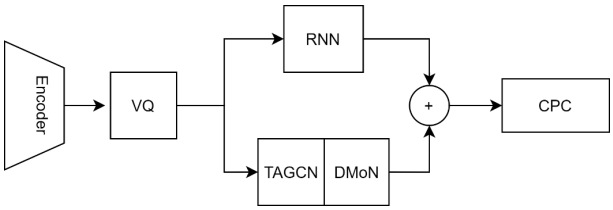

Figure 2: *Our proposed framework: Utilizing VQCPC, TAGCN, and DMoN.*

menting with greedy approach or dynamic programming, we cluster based on the interconnection among the speech representations. First, we discretize the speech waveform into a sequence of nodes and perform convolutions on graphs using TAGCN. Then we cluster those nodes on modularity in unsupervised training using DMoN.

3. Proposed Approach

In this section, we present our approach for discovering variable-length speech representation. Our proposed model consists of two different clustering layers: (1) a VQ that *categorizes* the acoustic features of each input time frame into latent classes; (2) graph clustering that *segments* the intertwined sequences of the categorized acoustic features, based on their temporal closeness to each other. Fig. 1 illustrates an overview of graph processing, and Fig. 2 shows our proposed framework. The details are described in the following sections.

3.1. Vector Quantized Contrastive Predictive Coding

For the discretization of acoustic features, we use a previously proposed efficient discrete neural network model, VQCPC [12] and the released implementation by its authors. For further details about VQCPC and its training process, refer to their original paper.

3.1.1. Vector Quantization

Vector-quantized neural networks (VQNNs) discretize the hidden representation of input speech samples $x_{1:T}$. VQNNs have a trainable latent embedding space called a *codebook* $\mathbf{C} \in \mathbb{R}^{N \times D}$, where N is the size of the categorical latent space (codewords), and D is the dimensionality of each latent embedding vector $\mathbf{c}_i$. The VQ replaces latent representation $\mathbf{x}$ of the encoder layer by the nearest codeword:

$$\mathbf{z} = \mathbf{c}_n, \text{ where } n = \arg\min_j ||\mathbf{x} - \mathbf{c}_j||^2. \quad (1)$$

In forward computation, nearest codeword z is passed to the next layer. In backward computation, gradient $\nabla_z \mathcal{L}$ is passed directly to the encoder layer using a straight-through estimator [26].

The VQ has an auxiliary objective, which encourages the encoder to commit to the embedding:

$$\mathcal{L}_{VQ} = \alpha \frac{1}{T} \sum_{i=1}^{T} ||\mathbf{x}_i - \mathtt{sg}(\mathbf{z}_i)||^2, \quad (2)$$

where α is a commitment cost weight, and $\mathtt{sg}$ is a stop-gradient operator that prevents update of its operand. The codebook is updated independently by the exponential moving average (EMA) of x. For further detail of the VQNNs, refer to this work [13].

3.1.2. Contrastive Predictive Coding

In this framework, instead of training by reconstruction loss as in typical autoencoder models, we train the neural networks to predict M steps of future observations $\hat{\mathbf{z}}$ by distinguishing them from negative examples $\tilde{\mathbf{z}}$ from set $\mathcal{N}_{t,m}$ [14]. The objective loss function, $\mathcal{L}_{CPC}$ is given as follows:

$$\mathcal{L}_{CPC} = -\frac{1}{M} \sum_{m=1}^{M} \log \left[\frac{\exp\left(\hat{\mathbf{z}}_{t+m}^{\top} \mathbf{W}_m \mathbf{c}_t\right)}{\sum_{\tilde{\mathbf{z}} \in \mathcal{N}_{t,m}} \exp\left(\tilde{\mathbf{z}}^{\top} \mathbf{W}_m \mathbf{c}_t\right)} \right], \quad (3)$$

where $\mathbf{W}$ is a learned weight matrix and $\mathbf{c}_t$ is a latent context representation. For $\mathbf{c}_t$ we used a concatenation of two different context representation of $\mathbf{z}$, each of which was produced by an autoregressive model and a GNN model addressed below.

3.2. Neural-based Graph Clustering

3.2.1. Graph Construction

We consider the codewords to be nodes, and the transitions between them to be edges in a directed graph. For each batch t, we construct a graph from the batched sequences of N codeword indices, produced by the VQ layer by encoding the input

speech data. We simply count each bigram $\{i, j\} \in N$ over the sequences, where we exclude consecutive patterns $\{i, i\}$. The bigrams represent the edges in a graph where their frequencies directly correspond to their weights. In other words, bigram $\{i, j\}$ is equivalent to $a_{i,j}$ in an adjacency matrix of sample graph $\mathbf{A}_t \in \mathbb{R}^{N \times N}$. Note that since the consecutive patterns are excluded, the graph does not contain any self-loop edges ($a_{i,i}$). To effectively capture the latent graph structure, we update the edge weights by their exponential moving averages at every batch throughout the training:

$$\overline{\mathbf{A}}_t = \overline{\mathbf{A}}_{t-1} + \beta(\mathbf{A}_t - \overline{\mathbf{A}}_{t-1}), \text{ where } \overline{\mathbf{A}}_1 = \mathbf{A}_1. \quad (4)$$

3.2.2. Graph Symmetrization

To apply graph clustering, we transform the adjacency matrix to symmetric affinity matrix so that edge weight $a_{i,j}$ is equal to $a_{j,i}$.We consider 2 symmetrization methods: *simple symmetrization* (denoted as "sim") and *degree-discounted bibliometric symmetrization* (denoted as "bib") [27].

Affinity matrix $\mathbf{U_s}$ obtained by the simple symmetrization can be understood intuitively as representing *temporal closeness* of each codeword to the others with the directionality of the original graph disregarded:

$$\mathbf{U}_s = \overline{\mathbf{A}}_t + \overline{\mathbf{A}}_t^T. \quad (5)$$

Degree-discounted bibliometric symmetrization, on the other hand, measures the similarity of each pair of nodes with respect to how many shared in- and out-links they have, and discounts the edge weights to or from the nodes with excessive links. Note that a pair of nodes with no edge in the original directed graph may have an edge if they share in- or out-links and, furthermore, the more they have, the more weight their edge will have in the resultant affinity matrix.

To obtain affinity matrix $\mathbf{U_d}$ from this transformation, we calculate degree-discounted out-link similarity $\mathbf{B_d}$ and in-link similarity $\mathbf{C_d}$, and simply add them together:

$$\mathbf{B}_d = \mathbf{D}_o^{-1/2}\overline{\mathbf{A}}_t\mathbf{D}_i^{-1/2}\overline{\mathbf{A}}_t^T\mathbf{D}_o^{-1/2}, \quad (6)$$

$$\mathbf{C}_d = \mathbf{D}_i^{-1/2}\overline{\mathbf{A}}_t\mathbf{D}_o^{-1/2}\overline{\mathbf{A}}_t^T\mathbf{D}_i^{-1/2}, \quad (7)$$

$$\mathbf{U}_d = \mathbf{B}_d + \mathbf{C}_d. \quad (8)$$

where $\mathbf{D}_o = \texttt{diag}(\overline{\mathbf{A}}_t\mathbf{1}_N)$ and $\mathbf{D}_i = \texttt{diag}(\overline{\mathbf{A}}_t^T\mathbf{1}_N)$. Last, for numerical stability, we symmetrically normalize the affinity matrix in both cases:

$$\tilde{\mathbf{U}} = \mathbf{D}^{-1/2}\mathbf{U}\mathbf{D}^{-1/2}, \text{ where } \mathbf{D} = \texttt{diag}(\mathbf{U}\mathbf{1}_N). \quad (9)$$

3.2.3. Graph Convolution and Clustering

By graph convolution, the highly correlated nodes (codewords) share similar features, and subsequently by graph clustering the graph is partitioned into K clusters based on the nodes' features and their edge weights. In our implementation, we set the number of codewords to 512 and partition them into 64 clusters.

Given codebook $\mathbf{C}$ as node features and symmetrically normalized affinity matrix $\tilde{\mathbf{U}}$, which is updated with the VQ's output codeword sequences at every step, we find a (soft) cluster assignment matrix $\mathbf{S} \in \mathbb{R}^{N \times K}$ for codebook $\mathbf{C}$ by optimizing objective $\mathcal{L}_{DMoN}$ [16]. From the clustered codebook and the sequences of the original codeword indices, we obtain *coarsened* vector-quantized speech representation.

We train l-layered topology-adaptive graph convolutional networks (TAGCN), each of which is followed by rectified linear unit (ReLU) activation. TAGCN extracts local features $\hat{\mathbf{C}} \in \mathbb{R}^{N \times \hat{D}}$ of the nodes with respect to the graph using a size-κ filter that aggregates κ-hop neighbor nodes' features weighted by the edge weights. For further details of TAGCN, see the original paper [15]. We implemented the TAGCN model using Pytorch Geometric [28], and stacked two TAGCN layers, one with a size-2 filter, and another with a size-3 filter:

$$\hat{\mathbf{C}} = \texttt{SeLU}(\texttt{TAGCN}_l^{\kappa}(\mathbf{C},\ \tilde{\mathbf{U}})). \quad (10)$$

We applied softmax on extracted node features $\hat{\mathbf{C}}$ multiplied by trainable weight matrix $\mathbf{W} \in \mathbb{R}^{\hat{D} \times K}$ to obtain the assignment matrix $\mathbf{S}$:

$$\mathbf{S} = \texttt{softmax}(\hat{\mathbf{C}}\mathbf{W}). \quad (11)$$

The optimal assignment matrix maximizes modularity Q of the graph. Modularity matrix $\mathbf{B}$ is defined:

$$\mathbf{B} = \tilde{\mathbf{U}} - \frac{\mathbf{d}^T\mathbf{d}}{2\mathcal{M}}, \quad (12)$$

where $\mathbf{d} = \sum_j \tilde{\mathbf{U}}$, and $\mathcal{M} = \sum_{i,j} \tilde{\mathbf{U}}$, and modularity Q:

$$Q = \frac{1}{2\mathcal{M}}\texttt{Tr}(\mathbf{S}^T\mathbf{B}\mathbf{S}). \quad (13)$$

With *collapse regularization* term R,

$$R = \frac{\sqrt{K}}{N}||\mathbf{S}^T||_F - 1, \quad (14)$$

where $||\cdot||_F$ is a frobenius norm, we find the best assignment matrix by minimizing the objective of the DMoN[16]:

$$\mathcal{L}_{DMoN} = -Q + \gamma R, \quad (15)$$

where hyperparameter γ adjusts the strength of regularizer R. We set $\gamma = 0.1$ in our implementation.

We can obtain clustered codebook $\hat{\mathbf{C}}_\mathbf{S}$ by,

$$\hat{\mathbf{C}}_\mathbf{S} = \mathbf{S}^T\hat{\mathbf{C}}\mathbf{S}. \quad (16)$$

We consider it a look-up table from which we can obtain new speech representation sequences, using the sequences of the original codeword indices. Each representation of the obtained sequences is concatenated with corresponding contextual vectors produced by the autoregressive model for the original codeword sequences, and they are trained jointly by the CPC.

3.2.4. Variable-length Speech Representation

During inference, we discretize the obtained speech representation. A discretized version of $\mathbf{S}$ can be obtained simply by converting each assignment vector $\mathbf{s}_i$ into a one-hot vector:

$$s_{i,j} = \begin{cases} 1 & \text{if } j = \arg\max \mathbf{s}_i \\ 0 & \text{otherwise} \end{cases} \quad (17)$$

The obtained speech representations are coarser than the original vector-quantized ones, since those which are likely to co-occur coalesce by the graph clustering: they can span across the time-frames.

4. Experimental Set-up and Results

4.1. Experimental Set-up

We evaluated our model by the English data set of the ZeroSpeech Challenge 2019. The *Train Unit Discovery* dataset contains around 10 minutes per speaker of read text from 100 speakers (16 hours of speech in total), and the test data set contains new utterances about 1-4 minutes long from another 20 speakers. Further details can be found here [3].

The speech waveforms were sampled at 16 kHz as input, and then we computed log-Mel spectrograms. After that, we applied VQCPC with the same training configuration and hyperparameter settings as in a previous work [12]. Our proposed graph clustering layer could be optimized jointly without any change to the VQCPC configuration.

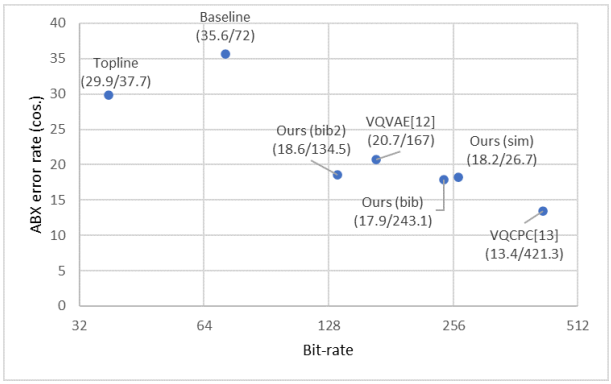

Figure 3: *The unit quality comparison between the models submitted in ZeroSpeech 2020 and our proposed models: ABX error with DTW-cosine distance as function of bit-rate for unit discovery. The values show "(ABX/bit-rate)" scores.*

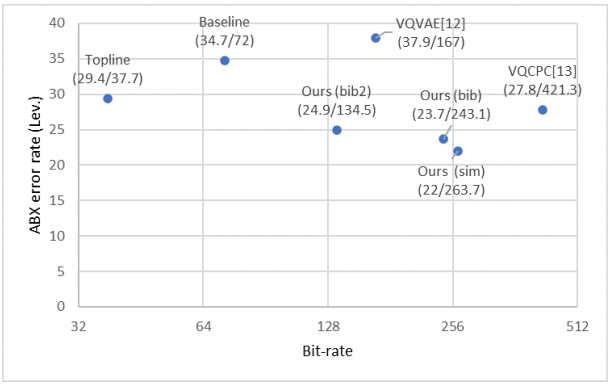

Figure 4: *The unit quality comparison between the models submitted in ZeroSpeech 2020 and our proposed models: ABX error with Levenshtein distance as function of bit-rate for unit discovery. The values show "(ABX/bit-rate)" scores.*

4.2. Evaluation Metrics

4.2.1. ABX Phone Discrimination

We evaluated the discovered unit quality by ABX discriminability test [29, 30], which measures whether X is more similar to A or B based on the discovered units (i.e., A and X are tokens of 'beg', and B is a token of 'bag'). Here the similarity is measured using normalized Levenshtein distance and the average frame-wise cosine distance of the tokens' representations along a DTW-realigned path, denoted as the "ABX error rate (Lev.)" and "ABX error rate (cos.)". The former evaluates the units as distinct symbols, and the latter evaluates them as vectors that may contain contextual information.

4.2.2. Bit-rate

The bit-rate is a product of the number of the symbols per unit time and their entropy:

$$B(\mathbf{V}) = \frac{L}{T} \sum_{i=1}^{L} p(\mathbf{v_i}) \log p(\mathbf{v_i}), \tag{18}$$

where $\mathbf{V}$ is a sequence of symbolic representation vectors $\mathbf{v_i}$ of length L, T is the time duration of $\mathbf{V}$ in seconds, and $p(\mathbf{v}_i)$ the probability of vectors $\mathbf{v_i}$.

4.3. Results and Discussion

We evaluated our proposed method's performance with two graph symmetrization methods ("sim" and "bib"). Furthermore, although our proposed graph clustering method produced variable-length speech representation, we investigated two ways during ABX evaluations: (1) providing the speech representation for each time frame by repeating the symbols ("bib1") that span several time frames or (2) providing variable-length speech representations independent of the time frame ("bib2").

Figures 3 and 4 show evaluation results of comparing our proposed models with the ZeroSpeech 2020 challenge on the Track 2019 official *Topline* and *Baseline* [4] as well as other submitted state-of-the-art models from ZeroSpeech 2020, VQ-VAE [11] and VQCPC [12]. Fig. 3 shows the ABX error with DTW-cosine distance as a function of the bit-rate for unit discovery/synthesis, and Fig. 4 shows the ABX error with Levenshtein distance as function of bit-rate for unit discovery/synthesis. For both of ABX error rate and the bit-rate, a lower rate is better.

The two proposed graph symmetrization methods ("sim" and "bib") did not have any significant performance differences in the performance. Although degree-discounted bibliometric symmetrization requires complicated computation, the simple symmetrization already suffices for our task, which may be due to graph convolution. Among the two results that used degree-discounted bibliometric symmetrization ("bib1" with fixed-length frame-based units and the "bib2" with variable-length time-frame independence), the results reveal that the one with variable-length speech representations gave optimum performance.

Since our model was built upon VQCPC, which achieved the lowest ABX error rate as of ZeroSpeech 2020, our concern investigated whether our proposed approach could lower the bit rate while retaining VQCPC's original unit quality. In this respect, our models were successful. They even outperformed VQCPC in terms of Levenshtein distance. Overall, our proposed models with variable-length speech representation achieved the best trade-off between unit quality and bit-rate.

5. Conclusions

We present neural-based graph clustering for variable-length symbolic unit discovery from raw speech data. We simultaneously discovered acoustic units by vector-quantized neural networks, constructed a graph of the acoustic units based on their temporal closeness, and clustered them by the modularity-based neural graph clustering. We evaluated our model on the English data set of the ZeroSpeech 2020 challenge on Track 2019, and compared our model with the state-of-the-art models submitted in ZeroSpeech 2020. Our models with variable-length speech representation achieved the best trade-off between unit quality and bit-rate.

6. Acknowledgements

Part of this work was supported by JSPS KAKENHI Grant Numbers JP17H06101 and JP21H03467.

7. References

[1] M. Versteegh, R. Thiolliere, T. Schatz, X. N. Cao, X. Anguera, A. Jansen, and E. Dupoux, "The zero resource speech challenge 2015," in *Proc. INTERSPEECH*, 2015, pp. 3169–3173.

[2] E. Dunbar, X. N. Cao, J. Benjumea, J. Karadayi, M. Bernard,

L. Besacier, X. Anguera, and E. Dupoux, "The Zero Resource Speech Challenge 2017," in *Proc. IEEE ASRU*, 2017, pp. 323–330.

[3] E. Dunbar, R. Algayres, J. Karadayi, M. Bernard, X.-N. C. J. Benjumea, C. D. L. Miskic, L. Ondel, A. W. Black, L. Besacier, S. Sakti, and E. Dupoux, "The Zero Resource Speech Challenge 2019: TTS without T," in *Proc. INTERSPEECH*, pp. 1088–1092.

[4] E. Dunbar, J. Karadayi, M. Bernard, X.-N. Cao, R. Algayres, L. Ondel, L. Besacier, S. Sakti, and E. Dupoux, "The Zero Resource Speech Challenge 2020: Discovering Discrete Subword and Word Units," in *Proc. INTERSPEECH*, pp. 4831–4835.

[5] H. Chen, C.-C. Leung, L. Xie, B. Ma, and H. Li, "Parallel inference of Dirichlet process Gaussian mixture models for unsupervised acoustic modeling: A feasibility study," in *Proc. INTERSPEECH*, 2015, pp. 3189–3193.

[6] M. Heck, S. Sakti, and S. Nakamura, "Feature optimized DPGMM clustering for unsupervised subword modeling: A contribution to zerospeech 2017," in *Proc. IEEE ASRU*, 2017, pp. 740–746.

[7] B. Wu, S. Sakti, J. Zhang, and S. Nakamura, "Optimizing DPGMM clustering in zero-resource setting based on functional load," in *Proc. The 6th Intl. Workshop on Spoken Language Technologies for Under-Resourced Languages*, pp. 1–5.

[8] L. Badino, A. Mereta, and L. Rosasco, "Discovering discrete subword units with binarized autoencoders and hidden-Markov-model encoders," in *Proc. INTERSPEECH*, 2015, pp. 3174–3178.

[9] A. Tjandra, B. Sisman, M. Zhang, S. Sakti, H. Li, and S. Nakamura, "VQVAE unsupervised unit discovery and multi-scale code2spec inverter for zerospeech challenge 2019," in *Proc. INTERSPEECH*, pp. 1118–1122.

[10] R. Eloff, A. Nortje, B. L. V. Niekerk, A. Govender, L. Nortje, A. Pretorius, E. V. Biljon, E. V. der Westhuizen, L. V. Staden, and H. Kamper, "Unsupervised acoustic unit discovery for speech synthesis using discrete latent-variable neural networks," in *Proc. INTERSPEECH*, 2019.

[11] A. Tjandra, S. Sakti, and S. Nakamura, "Transformer VQ-VAE for Unsupervised Unit Discovery and Speech Synthesis: ZeroSpeech 2020 Challenge," in *Proc. INTERSPEECH*, pp. 4851–4855.

[12] B. van Niekerk, L. Nortje, and H. Kamper, "Vector-quantized neural networks for acoustic unit discovery in the zerospeech 2020 challenge," in *Proc. INTERSPEECH*, 2020.

[13] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. NeurIPS*, 2017.

[14] A. van den Oord, Y. Li, and O. Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[15] J. Du, S. Zhang, G. Wu, J. M. F. Moura, and S. Kar, "Topology Adaptive Graph Convolutional Networks," *arXiv preprint arXiv:1710.10370*, 2018.

[16] A. Tsitsulin, J. Palowitch, B. Perozzi, and E. Müller, "Graph clustering with graph neural networks," *arXiv preprint arXiv:2006.16904*, 2020.

[17] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering," in *Proc. Advances in Neural Information Processing Systems*, 2016.

[18] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. ICLR*, 2017.

[19] J. Bastings, I. Titov, W. Aziz, D. Marcheggiani, and K. Simaan, "Graph convolutional encoders for syntax-aware neural machine translation," in *Proc. EMNLP*, 2015, p. 1957–1967.

[20] S. Qin, T. Jiang, S. Wu, N. Wang, and X. Zhao, "Graph convolution-based deep clustering for speech separation," *IEEE Access*, vol. 8, pp. 82 571–82 580, 2020.

[21] P. Tzirakis, A. Kumar, and J. Donley, "Multi-channel speech enhancement using graph neural networks," in *Proc. IEEE ICASSP*, 2021.

[22] Z. Lian, J. Tao, B. Liu, J. Huang, Z. Yang, and R. Li, "Conversational emotion recognition using self-attention mechanisms and graph neural networks," in *Proc. INTERSPEECH*, 2020, pp. 2347–2351.

[23] A. Sun, J. Wang, N. Cheng, H. Peng, Z. Zeng, and J. Xiao, "GraphTTS: Graph-to-sequence modelling in neural text-to-speech," in *Proc. IEEE ICASSP*, 2020.

[24] R. Liu, B. Sisman, and H. Li, "Graphspeech: Syntax-aware graph attention network for neural speech synthesis," in *Proc. IEEE ICASSP*, 2021.

[25] H. Kamper and B. van Niekerk, "Towards unsupervised phone and word segmentation using self-supervised vector-quantized neural networks," *arXiv preprint arXiv:2012.07551*, 2020.

[26] N. L. Yoshua Bengio and A. C. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation." *arXiv preprint arXiv:1308.3432*, 2013.

[27] V. Satuluri and S. Parthasarathy, "Symmetrizations for clustering directed graphs," in *Proc. EDBT/ICDT*, 2011.

[28] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

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

[30] T. Schatz, V. Peddinti, X.-N. Cao, F. Bach, H. Hermansky, and E. Dupoux, "Evaluating speech features with the minimal pair ABX task (ii): Resistance to noise," in *Proc. INTERSPEECH*, 2014.