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

Non-Autoregressive Predictive Coding for Learning Speech Representations from Local Dependencies

Alexander H. Liu, Yu-An Chung, James Glass

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology, USA
{alexhliu, andyyuan, glass}@mit.edu

Abstract

Self-supervised speech representations have been shown to be effective in a variety of speech applications. However, existing representation learning methods generally rely on the autoregressive model and/or observed global dependencies while generating the representation. In this work, we propose Non-Autoregressive Predictive Coding (NPC), a self-supervised method, to learn a speech representation in a non-autoregressive manner by relying only on local dependencies of speech. NPC has a conceptually simple objective and can be implemented easily with the introduced Masked Convolution Blocks. NPC offers a significant speedup for inference since it is parallelizable in time and has a fixed inference time for each time step regardless of the input sequence length. We discuss and verify the effectiveness of NPC by theoretically and empirically comparing it with other methods. We show that the NPC representation is comparable to other methods in our experiments while being more efficient.

Index Terms: speech representation, self-supervised learning, non-autoregressive model

1. Introduction

Speech representation learning aims to extract high-level representations from surface features such as waveforms or spectrograms. Ideally, these representations make latent information in speech such as phonetic content and speaker characteristics more accessible to downstream tasks. While speech representations can be computed via different transformations of the surface feature, recent research has successfully combined neural networks and self-supervised learning (i.e., where learning targets can be derived from the input itself) [1, 2, 3, 4, 5, 6, 7, 8, 9].

Contrastive Predictive Coding (CPC) [1] is one such approach whereby a surface feature sequence is transformed into a latent representation by an encoder network. An autoregressive model summarizes the latent sequence history into a higher-level representation that is used to predict future latent representations. CPC and its extensions are effective for learning expressive and robust representations of speech [2, 3, 4, 5].

Instead of predicting future *latent* representations, Autoregressive Predictive Coding (APC) suggests that simply predicting future surface features is suitable for learning an effective representation of speech [6]. APC has also been extended and improved by enforcing constraints that information from past sequences be stored in the representation [10] or by imposing an information bottleneck via vector quantization (VQ) [11].

Inspired by the left-to-right nature of speech, both CPC and APC achieve self-supervision by using future features in a uni-directional ordered learning. Masked Language Modeling (MLM) relaxes this constraint and uses a different self-supervised learning strategy whereby parts of the input sequence are randomly masked and set as the prediction target, allowing models to observe the entire surface feature sequence without seeing the target and derive a representation from contextual information [12]. A bidirectional RNN [8] or Transformer [9] can be used to learn MLM speech representations.

To introduce our work, we first formulate our task and mark two properties of the aforementioned methods. Our goal is to derive a high-level representation $(h_1, h_2, ..., h_T)$ from the surface feature sequence of audio $(x_1, x_2, ..., x_T)$ with length T. In APC and CPC, the representation h_t at time t is learned by predicting the unseen future frame x_{t+n} (or its latent counterpart) based on the current frame x_t and the previous latent representation h_{t-1}. These methods 1) are *autoregressive*: the previous representation h_{t-1} is required at each timestep; and 2) incorporate *global dependency*: h_{t-1} encodes all the past inputs $(x_1, ..., x_{t-1})$, making h_t depend on $(x_1, .., x_t)$. These properties also apply to MLM[1], but with a *stronger* global dependency since the full input sequence is always observed, i.e. h_t depends on $(x_1, ..., x_T)$ for any t. Note that these two properties have a large impact on the efficiency of representation models. The autoregressive property implies that the extraction process cannot be parallelized in time, and relying on global dependency results in time complexity bounded by the input sequence length as we verify later in our experiments (Sec. 3.2.1).

To this end, we propose Non-Autoregressive Predictive Coding (NPC) to learn latent representations in a *non-autoregressive* manner by observing only the *local dependency* of speech. Without the autoregressive property, NPC offers a significant speedup for deriving speech representations by enabling parallelizing in time. By observing only local dependencies, NPC allows representations to be derived efficiently regardless of the input sequence length, which is useful for downstream tasks requiring low latency such as streaming speech recognition. Furthermore, we show that representations derived by NPC, relying only on local dependencies and incorporating a non-autoregressive model, is empirically comparable to different self-supervised learning strategies.

2. Proposed method

2.1. Non-autoregressive Predictive Coding

To derive the high-level feature h_t at time t without a global dependency or autoregressive property, we restrict it to depend only on the neighbors of x_t within a receptive field $(x_{t-r}, ..., x_t, ..., x_{t+r})$ of size $R = 2r + 1$. While any model architecture with a fixed-size receptive field can be used, we stack Convolution Blocks (ConvBlock, Fig. 1(b)) to build the representation extraction model in this work.

[1]The MLM autoregressive property does not apply to transformers [13] but increases compute complexity in terms of input sequence length.

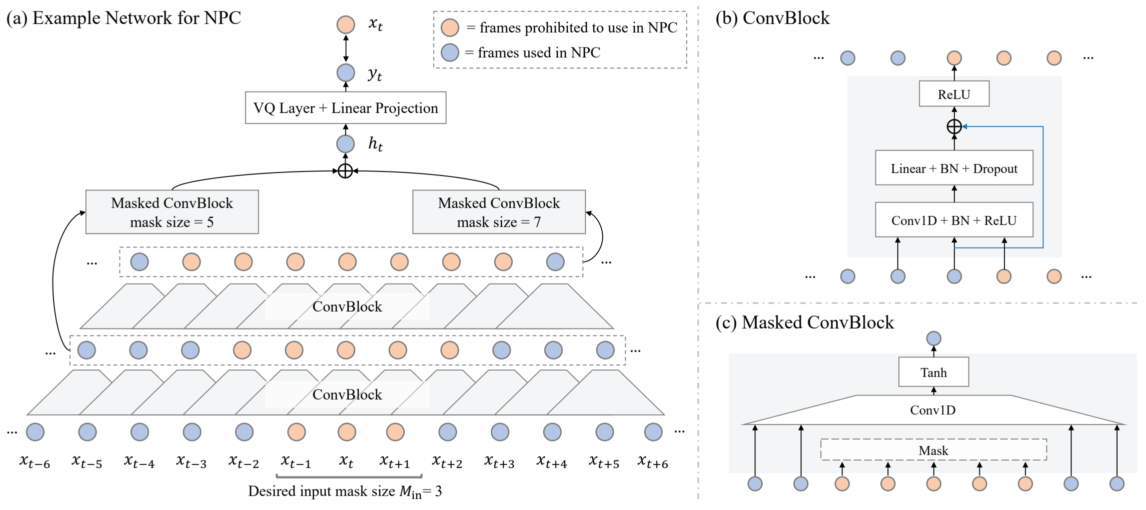
(a) Example Network for NPC
= frames prohibited to use in NPC
= frames used in NPC
x_t
y_t
VQ Layer + Linear Projection
h_t
Masked ConvBlock mask size = 5
Masked ConvBlock mask size = 7
ConvBlock
ConvBlock
x_{t-6} x_{t-5} x_{t-4} x_{t-3} x_{t-2} x_{t-1} x_t x_{t+1} x_{t+2} x_{t+3} x_{t+4} x_{t+5} x_{t+6}
Desired input mask size M_{in}= 3
(b) ConvBlock
ReLU
Linear + BN + Dropout
Conv1D + BN + ReLU
(c) Masked ConvBlock
Tanh
Conv1D
Mask

Figure 1: *Illustration of NPC at time t with desired input mask size $M_{in} = 3$ on an example network having receptive field size $R = 13$. In all figures, orange nodes represent the frames that contain information of the target frame x_t, therefore should not be used for prediction; blue nodes are the rest of frames that can be used. (a) An example of 2-layered feedforward network for NPC. Input frames are processed by layers of ConvBlock, features from Masked ConvBlock at each layer are summed up to the context representation h_t, which will be passed into Vector Quantization layer followed by a linear projection predicting y_t to match the target frame x_t. (b) ConvBlock applies a CNN along the time axis, resulting target-related information to spread to its neighbor at next layer. (c) Masked ConvBlock generates representation only based on unmasked frames containing no prohibited information.*

To ensure that the high-level feature h_t is indeed representative of x_t, it is linearly transformed into y_t to predict x_t. Following previous work [11, 4, 14], we adopt a VQ layer [15] before the linear projection to serve as an information bottleneck on h_t to yield a better representation. The NPC objective is to minimize the $L1$ difference between surface feature x_t and prediction y_t based on h_t for all time steps:

$$\sum_{t=1}^{T} |y_t - x_t| \,. \tag{1}$$

Note that the target x_t of representation h_t is in the receptive field $(x_{t-r}, ..., x_t, ..., x_{t+r})$, which might cause h_t to be uninformative, since the network can implicitly learn to copy the target directly from the input. Therefore, NPC requires an additional restriction where the target and its close neighbors in time cannot be observed by h_t. Concretely, given the receptive field $(x_{t-r}, ..., x_t, ..., x_{t+r})$ of the high-level representation h_t, the nearest $2m$ neighbors of x_t and itself, i.e. $(x_{t-m}, ..., x_t, ..., x_{t+m})$, cannot be observed, forming an input mask size $M_{\text{in}} = 2m + 1$ for h_t. As the receptive field of each layer in the model varies, the desired mask size changes accordingly, e.g., the choice of ConvBlock with receptive field of size 3 results in the desired mask size to increase by 2 (see orange nodes in Fig. 1(a)(b)).

2.2. Masked Convolution Blocks for NPC

To implement the desired restriction, we introduce the Masked Convolution Block (Masked ConvBlock), where the kernel-wise convolution operation can be written as

$$(W \odot D) * Z \tag{2}$$

with $Z \in \mathbb{R}^{T\times d}$ denoting the intermediate features from model with sequence length T and dimension d, $W \in \mathbb{R}^{k\times d}$ denoting the learnable kernel weight with size k, and $D \in \{0, 1\}^{k\times d}$ denoting the mask with each element $d_{ij} = \mathbb{1}_{i\leq \frac{k}{2}-m} + \mathbb{1}_{i\geq \frac{k}{2}+m}$. For example, Fig. 1(c) illustrates a Masked ConvBlock with $k = 7$ and $m = 2$. The Masked ConvBlock prevents high-level feature h_t from observing any surface feature within the desired input mask. Moreover, it can be applied to any intermediate level feature as long as the desired mask size can be calculated at each layer. In practice, we find this property valuable as it allows aggregation of representations at different depths.

3. Experiments

3.1. Self-supervised Learning Setup

We learn speech representations from LibriSpeech [17], where the 360-hour clean subset is used for a probing task in Sect. 3.2 and the full 960-hour training set is used for semi-supervised speech recognition experiments in Sect. 3.3. An 80-dimensional log Mel spectrogram is selected as the surface feature of speech. Unless otherwise specified, each channel is normalized to have zero mean and unit variance across the same utterance. For the NPC model, we use multi-layer convolution networks, where each layer consists of a ConvBlock and Masked ConvBlock as shown in Fig. 1. Given a desired receptive field R, since ConvBlocks have a fixed receptive field of 3, the kernel size of Masked ConvBlock can be set to $R-2\times L$ where L is the depth of NPC model. The dimensionality of the representation and all intermediate layers is set to 512 for probing tasks, while 768 is used in the recognition task. We use the Gumbel-softmax VQ layer described in [4] with a group of 4 codebooks each consisting of 64 codewords. We train NPC using Adam [18] with a learning rate of 10^{-3} and a batch size of 32 for 50 epochs.

3.2. Evaluation of Representation with Probing Tasks

We first evaluate the NPC representation and study the effect of hyper-parameter choice via probing tasks. We follow prior work [1, 6, 9] to define the "effectiveness" of representations such as accessibility to latent information, i.e., their linear separability with respect to underlying phonetic label and speaker

Table 1: *Efficiency and performance of different self-supervised methods. All representations have dimension of 512. A speaker-wise normalized log Mel spectrogram is used as the surface feature. All numbers in the phone and speaker error rate columns except those of NPC are directly taken from [11]. See Sec. 3.2.1 for more setup details.*

Method	Network	Frame dependency	Theoretical† complexity	Empirical§ inference time	Phone error rate	Speaker error rate
log Mel-spectrogram	-	-	-	-	50.3	17.6
CPC [1]	3-layer GRU	Left-to-right	$\mathcal{O}(T \cdot d^2)$	29x	34.1	9.7
APC [6]					33.3	8.5
MT-APC [10]					30.5	7.3
VQ-APC [11]					28.4	5.5
RNN-MLM [8]	3-layer Bi-GRU	Global	$\mathcal{O}(T \cdot d^2)$	72x	32.4	6.2
Transformer-MLM [9]	3-layer Transformer		$\mathcal{O}(T^2 \cdot d)$	33x	30.8	5.1
NPC (ours)	3-layer Masked Conv.	Local	$\mathcal{O}(k \cdot d^2)$	1x	27.9	6.1

† Frame-wise time complexity. T denotes the sequence length, d the representation dimension, and k the kernel size.
§ Averaged time cost over 10K runs on a single GPU with PyTorch [16] without further optimization on all networks

ID. The NPC model pre-trained on LibriSpeech is fixed and used to extract representations from the Wall Street Journal corpus (WSJ) [19] for the tasks defined in [11]. For phone classification, we use 90% of the utterances in the standard `si284` split to train a linear classifier, the remaining 10% as a validation set, and report frame-wise test accuracy on `dev93`. For speaker classification, the extracted representations are averaged utterance-wise to serve as input to a linear classifier. We consider the first 259 speakers in `si284` and use 80% of the utterances as a training set, 10% as a validation set, and report the frame-wise test accuracy on the last 10%. All reported numbers are averaged over 3 runs with negligible variance.

3.2.1. Performance

In Table 1, we compare NPC with prior speech representation learning models, including CPC [1], APC family [6, 10, 11], and MLM family [8, 9] as introduced in Sec. 1. We note that utterance-wise zero mean unit variance normalization on log Mel spectrograms is more suitable for NPC (and potentially all other methods), but we use speaker-wise normalization following previous work specifically in Table 1 for a fair comparison to the reported results in [11].

Efficiency: To study the speed advantage of NPC brought by the non-autoregressive and local-only dependent property, we first compare the time complexity and empirical inference speed to others as shown in Table 1. For time complexity, we consider the worst-case complexity per frame in terms of the input sequence length T, the representation dimension d, and the convolution kernel size k for the NPC model.[2] For empirical inference time, we average run time over 10K runs for all models with fixed sequence length $T = 1000$ (approximately corresponding to a 10 sec. utterance), $d = 512$, and batch size 32.

For NPC, the time complexity is $\mathcal{O}(k \cdot d^2)$ since the representation at any time step has a fixed-size receptive field depending on k, which is independent of the sequence length T. We set the average running time of a 3-layer NPC as the standard (denoted "1x" in Table 1) and compare it against other methods. For APC and CPC based methods, the worst case is the representation at the end of the sequence which must process through all T inputs, resulting in the complexity $\mathcal{O}(T \cdot d^2)$. With the choice of 3-layer GRU, we observed 29 times longer inference time on APC and CPC models. For RNN-MLM, the time complexity is identical to the previous case since the representation is the combination of 2 GRU hidden states. However, in practice, bi-directional autoregressive representations can be up to 72 times slower than NPC without further optimization. For the Transformer-MLM, the time complexity is $\mathcal{O}(T^2 \cdot d)$ since each representation is a weighted sum of the complete sequence of hidden states of transformer encoders as noted in [13]. As speech signals are generally longer ($T > d$), we observed a slightly longer inference time than APC/CPC models.[3]

Effectiveness: Given that NPC provides a significantly faster inference, we now take a look into the accessibility of speaker characteristics and phonetic information compared to other methods. For the task of speaker classification, an NPC representation produced a 6.1% error rate where the best from Transformer-MLM is 1% better. This suggests that NPC may not be as effective as other representation models when the task explicitly requires global information. For phone classification, which depends less on global information compared to speaker classification, we observe a better performance compared to other methods, indicating that NPC can be applied for tasks focusing on local dependencies without a trade-off.

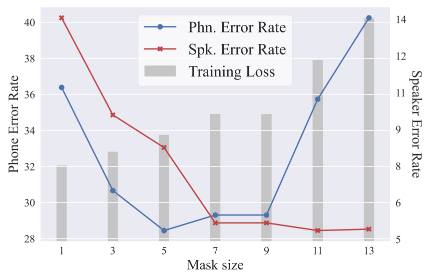

Figure 2: *Phone/speaker error rate and training loss with respect to different mask size on 2-layer NPC with* $R = 23$.

3.2.2. Importance of the Mask Size M_{in}

Fig. 2 shows the result of varying the mask size with a fixed receptive field of size 23, i.e. restricting inputs to be 11 frames on both sides of the target. Intuitively, increasing the mask size

[2]We treat the depth of models c as a constant since all models discussed in this paper have $c \ll T$ and $c \ll d$.

[3]In practice, this can be addressed by downsampling the feature sequence at the cost of making frame-wise representation unavailable.

Table 2: *Ablation study on NPC with input mask size $M_{in} = 5$, receptive field size $R = 23$. Single MaskedConv indicates applying Masked ConvBlock at the last layer only.*

Method	PER
NPC 4-layer	27.2
- remove 1 layer	27.7
- remove 2 layer	28.8
- remove VQ layer	27.9
- Single MaskedConv	29.7

Table 3: *Word error rate on LibriSpeech test set with 100 hour labeled data and representations from 960 hours training set, comparing to results from prior work [20].*

Representation	Depth / Model	clean / other
filterbank	-	9.36 / 30.20
wav2vec 2.0 [5]	12 Transformer	5.10 / 11.94
VQ-APC [11]	3 uni-GRU	7.42 / 23.38
DeCoAR [21]	4 bi-LSTM	6.10 / 17.43
DeCoAR 2.0 [20]	12 Transformer	5.02 / 12.07
NPC	3 ConvBlock	7.39 / 22.49

will increase the difficulty of predicting the target frame, and the training loss increases accordingly as a consequence. It can be observed that with the mask size less than 5, NPC representations begin to lose speaker and phonetic accuracy despite having a lower loss, which verifies our assumption in Sec. 2.1 where observing the target and its close neighbor will result in a less informative representation. On the other hand, a dramatic increase in phone error rate but not the speaker error rate is observed as the mask size exceeds 9, indicating that proper constraint on mask size is important for NPC to capture phonetic content. This matches the fact that phonetic content may change within a short time period while speaker characteristics tend to persist across time, hence are less affected by a larger mask.

3.2.3. Ablation study

To verify the importance of the model architecture, we performed an ablation study and list the results in Table 2. We note that the difference in speaker error rate is not significant and we only report phone error rate. We start with a 4-layer NPC model with receptive field size $R = 23$ and input mask size $M_{\text{in}} = 5$. By either reducing the depth of the NPC model or removing the vector quantization layer, the phone error rate slightly increases but varied no more than 1.6%. In contrast, phone error rate drops over 2% when applying the Masked ConvBlock on the last layer only (29.7). Nevertheless, we observe that none of the architectural decisions have a huge impact on NPC as we also saw for the input mask size M_{in}, which demonstrates the robustness of the NPC model in terms of architecture.

3.3. Application on Semi-supervised Speech Recognition

Speech recognition is used as the downstream application for NPC. We follow the semi-supervised setting of prior work [20] using the following pipeline: 1) Pre-train the self-supervised model on all 960 hours of LibriSpeech data and freeze the representation extraction parameters for the downstream task. 2) Train a 2-layer bidirectional LSTM acoustic model using the CTC objective [22]. The targets are phone sequences consisting of 71 phone labels plus 1 blank symbol. 3) Apply WFST-based ASR decoding from EESEN [23] together with the 4-gram language model provided by LibriSpeech. The *dev-clean* subset is used for validation and we report the word error rates (WERs) on *test-clean* and *test-other* in Table 3. The NPC used in this pipeline has 3 layers and 768 dimensions, the final ReLU function in ConvBlock is replaced with BatchNorm, and the output of the last ConvBlock is used as the input of downstream ASR.

Clearly, there is a gap between NPC (shallow network with local dependencies only) and state-of-the-art models like wav2vec2.0 [5] and DeCoAR2.0 [20] (which are deeper and more contextualized). This suggests that speeding up self-supervised speech representation without hurting downstream applications is still an unsolved problem. Nevertheless, NPC still made a significant improvement over baseline surface feature (spectrogram) similar to VQ-APC [11] with a lower computational cost that benefits computationally limited scenarios.

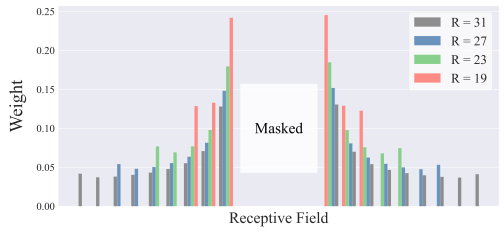

Figure 3: *Normalized magnitude of weights of CNN in Masked ConvBlock with different receptive field size R.*

3.4. NPC Analysis

Conceptually, NPC relies on local context to predict a target frame. The idea of learning an embedding based on local neighbors has been useful for learning word embeddings [24, 12] and also for speech representation learning [25, 26, 21, 27, 28]. However, NPC defines *explicit* masks, and uses a simple reconstruction loss, which differentiates it from other methods.

To better understand how NPC derives its representation from speech, we take the Masked ConvBlock kernels from the pre-trained 2-layer NPC model of different receptive fields R and compute the magnitude of these kernel weights at the second layer. This can be viewed as the importance of the adjacent frames of the target learned by NPC for generating a speech representation. Results are normalized and visualized in Fig. 3.

Unsurprisingly, frames adjacent to the masked input possess the largest magnitude, indicating they are the most important part for NPC to produce a representation. Conversely, the inputs farthest from the target usually have less than 5% of the total magnitude. This supports our view that local dependency is sufficient for learning effective speech representations.

4. Conclusion

In this work we propose Non-Autoregressive Predictive Coding (NPC) to significantly speed up the inference time required for speech representation learning. This is done by learning only from local contexts of speech with a fix-sized receptive field. Target-related information masking is implemented by a Masked ConvBlock. In our experiments, we examine and discuss the importance of each NPC component to demonstrate the robustness of the proposed framework. Evaluations on the learned representation and analysis of the model were carried out to support the conclusion that speech representation can be obtained more efficiently by NPC.

Code at: `https://github.com/Alexander-H-Liu/NPC`

5. References

[1] A. v. d. Oord, Y. Li, and O. Vinyals, “Representation learning with contrastive predictive coding,” *arXiv preprint arXiv:1807.03748*, 2018.

[2] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux, “Unsupervised pretraining transfers well across languages,” in *ICASSP*, 2020.

[3] S. Schneider, A. Baevski, R. Collobert, and M. Auli, “wav2vec: Unsupervised pre-training for speech recognition,” in *Interspeech*, 2019.

[4] A. Baevski, S. Schneider, and M. Auli, “vq-wav2vec: Self-supervised learning of discrete speech representations,” in *ICLR*, 2020.

[5] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in *NeurIPS*, 2020.

[6] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, “An unsupervised autoregressive model for speech representation learning,” in *Interspeech*, 2019.

[7] S. Pascual, M. Ravanelli, J. Serrà, A. Bonafonte, and Y. Bengio, “Learning problem-agnostic speech representations from multiple self-supervised tasks,” in *Interspeech*, 2019.

[8] W. Wang, Q. Tang, and K. Livescu, “Unsupervised pre-training of bidirectional speech encoders via masked reconstruction,” in *ICASSP*, 2020.

[9] A. Liu, S.-W. Yang, P.-H. Chi, P.-C. Hsu, and H.-Y. Lee, “Mockingjay: Unsupervised speech representation learning with deep bidirectional Transformer encoders,” in *ICASSP*, 2020.

[10] Y.-A. Chung and J. Glass, “Improved speech representations with multi-target autoregressive predictive coding,” in *ACL*, 2020.

[11] Y.-A. Chung, H. Tang, and J. Glass, “Vector-quantized autoregressive predictive coding,” in *Interspeech*, 2020.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional Transformersfor language understanding,” in *NAACL-HLT*, 2019.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *NIPS*, 2017.

[14] J. Chorowski, R. Weiss, S. Bengio, and A. van den Oord, “Unsupervised speech representation learning using wavenet autoencoders,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 12, pp. 2041–2053, 2019.

[15] A. van den Oord, O. Vinyals *et al.*, “Neural discrete representation learning,” in *NIPS*, 2017.

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury *et al.*, “PyTorch: An imperative style, high-performance deep learning library,” in *NeurIPS*, 2019.

[17] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, “LibriSpeech: An ASR corpus based on public domain audio books,” in *ICASSP*, 2015.

[18] D. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *ICLR*, 2015.

[19] D. Paul and J. Baker, “The design for the Wall Street Journal-based CSR corpus,” in *Speech and Natural Language Workshop*, 1992.

[20] S. Ling and Y. Liu, “DeCoAR 2.0: Deep contextualized acoustic representations with vector quantization,” *arXiv preprint arXiv:2012.06659*, 2020.

[21] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, “Deep contextualized acoustic representations for semi-supervised speech recognition,” in *ICASSP*, 2020.

[22] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *ICML*, 2006.

[23] Y. Miao, M. Gowayyed, and F. Metze, “EESEN: End-to-end speech recognition using deep RNN models and WFST-based decoding,” in *ASRU*, 2015.

[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[25] B. Milde and C. Biemann, “Unspeech: Unsupervised speech context embeddings,” in *Interspeech*, 2018.

[26] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep contextualized word representations,” in *NAACL-HLT*, 2018.

[27] Y.-S. Chuang, C.-L. Liu, and H.-Y. Lee, “SpeechBERT: Cross-modal pre-trained language model for end-to-end spoken question answering,” in *Interspeech*, 2020.

[28] X. Song, G. Wang, Z. Wu, Y. Huang, D. Su, D. Yu, and H. Meng, “Speech-XLNet: Unsupervised acoustic model pretraining for self-attention networks,” in *Interspeech*, 2020.