Autoregressive Predictive Coding: A Comprehensive Study

Gene-Ping Yang, Sung-Lin Yeh, Yu-An Chung, James Glass, and Hao Tang

***Abstract*—We review autoregressive predictive coding (APC), an approach to learn speech representation by predicting a future frame given the past frames. We present three different views of interpreting APC, and provide a historical account to the approach. To study the speech representation learned by APC, we use common speech tasks, such as automatic speech recognition and speaker verification, to demonstrate the utility of the learned representation. In addition, we design a suite of fine-grained tasks, including frame classification, segment classification, fundamental frequency tracking, and duration prediction, to probe the phonetic and prosodic content of the representation. The three views of the APC objective welcome various generalizations and algorithms to learn speech representations. Probing on the suite of fine-grained tasks suggests that APC makes a wide range of high-level speech information accessible in its learned representation.**

***Index Terms*—Automatic speech recognition, predictive coding, representation learning, self-supervised learning, speaker verification.**

I. Introduction

PREDICTING the future given the past and learning from sequential data is two sides of the same coin. Being able to predict the future given the past is a fundamental requirement for data compression [1], [2], [3], and being able to compress data implies learning [4], [5]. Even at the intuitive level, it is not difficult to accept that if a model knows something about the data, then the model, to a certain extent, is able to anticipate the future given the past.

This paper focuses on the converse and asks the question: If a model is able to predict the future, does the model learn something about the data? Predicting the future has been the subject of information theory, coding, and language modeling. However, not until the seminal work of Peters et al. [6] (known as ELMo), has it been shown that learning to predict the future can significantly improve many other tasks (often called downstream tasks to differentiate from the task of predicting the future). Since then, lots of variants in the text domain [7], [8] and in the speech domain [9], [10], [11], [12], [13], [14] have been proposed. This family of approaches is called self-supervised learning, in which models are trained to solve a task based on transformations of the input alone. This paper focuses on the simplest form of this approach, namely Autoregressive Predictive Coding (APC) [15].

In this paper, we will focus on the discussion of APC for speech data. The approach, however, is general and can be applied to other sequential data, such as texts or videos. The goal of APC is to predict a future acoustic frame given the past frames. Predicting the frame immediately after, however, can be too simple a task, as acoustic frames are temporally smooth. Instead, APC requires models to predict a frame that is a few frames into the future. It might be difficult to reason why a model learns anything by simply predicting the future. This paper will provide three views, connecting the APC objective with maximum likelihood and mutual information.

To understand what a model trained with APC learns, we study, typically, a sequence of vectors produced by the model given a speech utterance. The sequence of vectors is often called a representation of the input utterance. It is odd to ask whether the sequence of vectors represents the speech utterance better, because the waveform itself is the most faithful representation of the utterance. It is equally odd to ask whether more information about the utterance is in the sequence of vectors, because all the information is in the waveform itself. Instead, we ask whether certain information is, not only encoded in the representation, but also easily accessible. In this paper, we define accessibility similar to Chung et al. [15] as how much the information can be extracted by a simple linear classifier.

Self-supervised models commonly serve as initialization for another model. The training of downstream tasks with self-supervised models as initialization is called fine-tuning; the training of self-supervised models, in contrast, is often called pre-training. Fine-tuning with self-supervised models as initialization often brings significant improvements over models trained on the fine-tuned data set alone [10], [16], and is a common approach to leverage unlabeled data for semi-supervised learning, as pre-training does not require human annotations. Practitioners aiming for better performance, by all means, should fine-tune the models on downstream tasks. Fine-tuning, however, involves additional data sets and training objectives, complicating the analysis of self-supervised training. Our goal is to study what models learn during pre-training, for example with APC. In other words, for the sake of analysis, we freeze the parameters of self-supervised models and do not update them further in our setting.

Manuscript received 15 January 2022; revised 20 April 2022, 21 June 2022, and 4 August 2022; accepted 18 August 2022. Date of publication 1 September 2022; date of current version 14 October 2022. The guest editor coordinating the review of this manuscript and approving it for publication was Dr. Tara N Sainath. *(Corresponding author: Hao Tang.)*

Gene-Ping Yang, Sung-Lin Yeh, and Hao Tang are with the University of Edinburgh, Edinburgh EH8 9YL, U.K. (e-mail: s2064029@ed.ac.uk; sunglin.yeh@ed.ac.uk; hao.tang@ed.ac.uk).

Yu-An Chung and James Glass are with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: andyyuan@mit.edu; glass@mit.edu).

Digital Object Identifier 10.1109/JSTSP.2022.3203608

Practitioners have primarily focused on downstream tasks that have direct applications, such as automatic speech recognition (ASR) and speaker verification. In addition to these tasks, we evaluate our models on a suite of fine-grained speech tasks, such as frame classification, segment classification, fundamental frequency tracking, and duration prediction for vowels and pauses. Frame and segment classification are proxies for studying the phonetic content of the representation, while fundamental frequency tracking and duration prediction are proxies for studying the prosodic content of the representation. These fine-grained tasks are designed to reveal more properties of the representation than the tasks that bear direct applications.

Self-supervised learning is often advertised as requiring little to no human supervision. However, we stress that the choice of objectives, data sets, model architectures, and training hyperparameters critically affect what is learned in the representations. The design choices prior to learning, also known as inductive bias, constitute the assumptions we make, and we will discuss in the paper how these assumptions limit what models can learn.

The contributions of the paper are summarized as follows.

- We review the simplest form of self-supervised learning, namely APC, and present a historical account of the approach.
- We present three alternative views of the same objective, welcoming various directions to generalize APC.
- We study the utility of the learned representation both for direct applications, such as ASR and speaker verification, and for fine-grained analyses, such as frame classification, segment classification, fundamental frequency tracking, and duration prediction.

II. Autoregressive Predictive Coding

Autoregressive predictive coding (APC) is an objective function with the goal of predicting frames k time steps into the future. Formally, the objective function

$$L = \frac{1}{2}\sum_{t=1}^{T-k} \|x_{t+k} - Wh_t\|_2^2, \tag{1}$$

where $x_1, \ldots, x_T$ (or $x_{1:T}$ in short) is a sequence of log Mel features of length T, and $h_t = f(x_{1:t})$ for some neural network f such as a 3-layer LSTM. The weight matrix W is the regression layer that predicts x_{t+k} with h_t.

Once the model f is trained to minimize this objective, the hidden vectors $h_1, \ldots, h_T$ or other intermediate layers can be used as a representation of the original speech utterance. Downstream tasks (as opposed to the APC task) that take log Mel features as input can instead use the representation of the utterances as input. The focus of this paper is to study whether the learned representations bring us any benefits.

The time shift k is a hyperparameter that controls the difficulty of the APC task. When the time shift $k = 1$, the objective is similar to regular language modeling. In contrast to language modeling, there is temporal smoothness in the spectrogram, and setting $k = 1$ might be too simple a task for the model f. In general, the larger the k, the more difficult the task. The time shift k also affects what information is encoded in the representations, as we will see in the experiments.

A. A Probabilistic View

The objective L does not necessarily need to use the ℓ_2 norm. In fact, the original APC uses the ℓ_1 norm [15], drawing inspiration from end-to-end TTS systems [17]. One approach to generalize the loss function is to consider the probabilistic view. For example, the ℓ_2 norm can be seen as an isotropic Gaussian. Formally, if we define

$$p(x_{t+k}|h_t) \propto \exp\left(-\frac{1}{2}(x_{t+k} - Wh_t)^\top I^{-1}(x_{t+k} - Wh_t)\right), \tag{2}$$

i.e., a Gaussian centered at Wh_t, we can rewrite the objective as

$$L = -\sum_{t=1}^{T-k} \log p(x_{t+k}|h_t) = -\sum_{t=1}^{T-k} \log p(x_{t+k}|x_{1:t}) \tag{3}$$

$$= -\log \prod_{t=1}^{T-k} p(x_{t+k}|x_1, \ldots, x_t), \tag{4}$$

giving the objective a likelihood interpretation.

The probabilistic view is liberating, because the future frame x_{t+k} no longer needs to be the "output" of a model. For example, we can consider

$$p(x_{t+k}|h_t) \propto \exp\left(x_{t+k}^\top Wh_t\right), \tag{5}$$

i.e., a von Mises–Fisher distribution centered at Wh_t.[1] More generally, we can have any arbitrary processing over x_{t+k}. For example, we can choose

$$p(x_{t+k}|h_t) \propto \exp\left(g(x_{t+k})^\top Wh_t\right), \tag{6}$$

where g is another neural network, such as a convolutional neural network. This particular choice of $p(x_{t+k}|h_t)$ makes the objective reminiscent to contrastive predictive coding [18]. In other words, the probabilistic view blurs the distinction made by many [19], [20], [21] between reconstructive and contrastive self-supervised learning. The probabilistic view also allows for the modeling of uncertainty. For example, instead of assuming an isotropic Gaussian for $p(x_{t+k}|h_t)$, uncertainty can be modeled by a Gaussian mixture model as in mixture density networks [22].

B. A Variational View

The objective L does not explicitly reveal the dependencies among framewise representation $h_{1:T}$. Given the probabilistic view, we introduce a sequence of latent variables $z_{1:T}$, and assume a generative process that x_{t+k} is generated from z_t. The variational bound for the generative process is

$$\log p(x_{k+1:T}|x_{1:k}) \geq \mathbb{E}_{z_{1:T-k}\sim q}\left[\sum_{t=1}^{T-k} \log p(x_{t+k}|z_t)\right] - \mathrm{KL}[q(z_{1:T-k})\|p(z_{1:T-k})], \tag{7}$$

where $z_{1:T-k}$ is a sequence of latent variables, q is an auxiliary distribution of our choice from which the latent variables are

[1] Note that Wh_t needs to have an ℓ_2 norm of 1. We ignore the normalization for clarity.

sampled. If we choose $q(z_t|x_{1:t}) = \delta_{z_t=h_t}$, where δ_c is the Dirac delta centered at c, the first term in the variational bound falls back to the original objective L. This particular choice of q only allow z_t to depend on $x_{1:t}$ and assume z_t is no different from h_t. The KL term can be ignored if we assume $p(z_{1:T-k})$ to be uniform, but can always be reinstated with different assumptions. Note that q in principle can depend on anything, including the entire sequence $x_{1:T}$, but it is important for z_t to only depend on frames before t in q for the objective to be APC. If q has access to frames beyond $t+k$ when producing z_t, the objective becomes autoencoding where p and q can collaborate, and can lead to degenerate solutions (sometimes called collapse [23]). A common remedy if we want q to condition on the entire $x_{1:T}$ is to have a masking noise. The family of masked reconstruction losses [13], [14], [24] becomes a special case of this objective. In fact, this objective exactly matches denoising autoencoders [25] if each frame is assumed to be independent.

There are several benefits to formulating APC as a variational bound. Having an explicit generative process makes it simpler to incorporate constraints and to form generalizations of APC. For example, as we have discussed, the family of masked reconstruction losses [13], [14], [24] is a special case of this objective. As another example, VQ-APC [26] becomes a special case with this objective when the latent variable z_t's are discrete. We can also assume the latent variables to be structured, providing alternatives to discrete and temporally smooth representations [27], [28]. We will lightly compare a few special cases in the experiments, but a full treatment of potential generalizations is beyond the scope of this paper.

C. A Co-Training View

Learning to predict the future can also be seen as learning a distribution to maximize the mutual information between the past and the future. At time t, information-theoretic co-training assumes that the latent variable z_t is best predicted by the future frame x_{t+k} with a distribution $p(z_t|x_{t+k})$. In other words, the data $x_{1:T}$ can be drawn from the data distribution D, and the latent variable is drawn from $p(z_t|x_{t+k})$. By the data processing inequality, we and have

$$I(x_{1:t}, x_{t+k}) \geq I(x_{1:t}, z_t) = H(z_t) - H(z_t|x_{1:t}), \quad (8)$$

where $H(z_t)$ is the entropy of z_t and $H(z_t|x_{1:t})$ is the conditional entropy of z_t given $x_{1:t}$. We can further bound the conditional entropy with the cross entropy

$$\begin{aligned} H(z_t|x_{1:t}) &= -\mathbb{E}_{x_{1:t},x_{t+k}\sim D}\mathbb{E}_{z_t\sim p(z_t|x_{t+k})} \log p(z_t|x_{1:t}) \quad (9)\\ &= -\mathbb{E}_{x_{1:t},x_{t+k}\sim D}\big[\mathrm{KL}[p(z_t|x_{1:t})\|q(z_t|x_{1:t})] \\ &\quad - \mathbb{E}_{z_t\sim p(z_t|x_{t+k})}[\log q(z_t|x_{1:t})]\big] \quad (10)\\ &\leq \mathbb{E}_{x_{1:t},x_{t+k}\sim D}\mathbb{E}_{z_t\sim p(z_t|x_{t+k})}[\log q(z_t|x_{1:t})] \quad (11) \end{aligned}$$

by introducing an auxiliary distribution q. Note that q is allowed to depend on anything as long as it is a distribution of z_t, but we restrict it to $x_{1:t}$. Putting everything together, we aim to maximize the objective

$$H(z_t) - \mathbb{E}_{x_{1:t},x_{t+k}\sim D}\mathbb{E}_{z_t\sim p(z_t|x_{t+k})}[\log q(z_t|x_{1:t})], \quad (12)$$

i.e., the entropy subtracted by the cross entropy. The expectation over sequences from D is approximated by averaging over sequences in the training set. The derivation is due to McAllester [29].

The objective involves training $q(z_t|x_{1:t})$ and $p(z_t|x_{t+k})$ together, hence the name co-training [29]. In addition, $q(z_t|x_{1:t})$ is called the prediction model, predicting a representation z_t given the past $x_{1:t}$, and $p(z_t|x_{t+k})$ is called the confirmation model, confirming that the predicted representation agrees with the future x_{t+k} [29].

To recover the APC objective, we assume z_t is of the same type as the input frames, p is a $\mathcal{N}(x_{t+k}, I)$, and q is $\mathcal{N}(Wh_t, I)$. In words, we confirm the prediction of a frame with the actual future frame. Because the cross entropy of the two Gaussian distributions is

$$H(p,q) = \frac{1}{2}\left(d\log 2\pi + d + \|x_{t+k} - Wh_t\|_2^2\right), \quad (13)$$

where d is the dimension of the x_t's, we recover the regular APC objective with a few constant terms. The entropy term $H(z_t)$ can again be bounded with cross entropy by introducing yet another auxiliary distribution. We simply choose the auxiliary distribution to be uniform, and drop the entropy term. The entropy term can be reinstated by choosing a different auxiliary distribution.

The co-training view draws an explicit connection between APC and mutual information. Various generalizations of APC can be introduced by making assumptions on the prediction model and the confirmation model. The co-training approach is, in fact, extremely general and can subsume the variational view and many other adversarial approaches [29].

Various other approaches [30], [31], CPC [18] included, attempt to draw connections between self-supervised learning and mutual information. Our approach fits information-theoretic co-training well, as it explicitly makes the distinction between the past and the future.

III. Related Work

The approach, predicting the future given the past, has its roots in predictive coding [2], [3], and being autoregressive means there exists an order in which the sequence is generated.[2] Predictive coding is used in the context of sending a message from a transmitter to a receiver. The transmitter and the receiver both run a predictive model, for example, an LSTM trained with APC. The transmitter only sends the residual, the part that the predictive model fails to predict, to the receiver. The more accurate the predictive model, the fewer bits in the residual to send. Though compression is the original intended use, APC is not explicitly designed to maximize compression, and maximizing compression does not necessarily lead to better learned representations.

Predictive coding has been shown to be useful for learning speech representations. Long before APC, the coefficients learned from linear predictive coding have been a successful

[2]The modifier "autoregressive" is mostly redundant in classical predictive coding, because the order is always the one in which the message is sent.

speech representation for automatic speech recognition [32]. WaveNet [33], as another example, is a nonlinear version of predictive coding, and its learned representations can be applied to acoustic unit discovery [34].

The original design of APC [15] is heavily influenced by contrastive predictive coding (CPC) [18] and embedding from language models (ELMo) [6]. APC can be seen as an extension of ELMo, allowing the model to predict a target several frames ahead. The experimental methodology of APC follows the study of CPC [18], validating the existence of phonetic and speaker information with probing tasks, such as phone classification. In this paper, we further extend this methodology to studying prosodic information with fundamental frequency tracking and duration prediction. Following the study of ELMo [6], the model of choice is a stacked LSTM, because of the autoregressive nature of LSTMs and their strong performance in both small- and large-data regimes. However, the approach can be and has been applied to other model architectures, such as Transformers [35].

Representation learning, particularly based on the past and the future in sequences, has been studied, if not rediscovered, many times. Collobert and Weston [36] use a variant of language modeling to learn representations of texts from unlabeled data. Mobahi et al. [37] use video frames of the past and the future to learn visual representations. The two studies even antedate the term "representation learning" popularized by Bengio et al. [38] and "self-supervised learning" popularized by Doersch et al. [39]. Dai and Le [40] use language modeling to pre-train a sequence-to-sequence model, which is one of the early successes of pre-training with language modeling before ELMo. Srivastava et al. [41] propose to predict future video frames based on the past to learn visual representations.

The wide adoption of the pre-training paradigm for semi-supervised learning stems from computer vision. An instance of the pre-training paradigm is supervised pre-training, the approach of learning representations with a supervised task. The representations learned from image classification have been shown to be useful for object detection and semantic segmentation [42]. This approach has been extended to other supervised tasks without manual annotations, and is termed self-supervised learning [39]. Various tasks, such as colorization [43], solving jigsaw puzzles [44], and inpainting [45], in the vision domain have been proposed for self-supervised learning. Subsequent development in the texts domain based on language modeling, such as ELMo [6], BERT [7], and XLNet [8], also belongs to self-supervised learning.

The key difference between supervised pre-training and self-supervised pre-training is the type and the amount of information about the input retained in the representation. The representations learned through a supervised task tend to only include information useful to perform the task, while information irrelevant to the task tends to be discarded [46], [47]. As a result, representations learned by supervised pre-training are only suitable for tasks similar to the supervised task. This explains why, for example, representations trained on image classification can be useful for object detection and semantic segmentation, given the similarity among these tasks. The representations learned from self-supervised tasks, however, tend to include various aspects of the input, not for a specific task, in particular when the self-supervised task includes (re-)constructing parts of the input. This explains the wide applicability of representations learned by self-supervised learning.

Many [19], [20], [21] have categorized self-supervised approaches as generative (or sometimes reconstructive) or contrastive. For example, CPC [18] and wav2vec [9], [10] belong to the contrastive approach, while APC [15], MPC [13], [14], DeCoAR [11], [12] belong to the generative approach. The distinction is again in the type and the amount of information about the input retained in the representations. Contrastive approaches often define objectives (based on contrast) in the space of hidden vectors projected from the input, and by the data processing inequality, certain information could be discarded as long as the necessary contrast can be made. Compared to the reconstructive approach, where all the nuance of the input needs to be reconstructed, it is necessary for the representation to include all the nuance of the input. Though the probabilistic view of APC in Section II-A blurs the distinction between generative and contrastive approaches, the argument about the types and amounts of information retained still holds.

A comprehensive summary of the recent advance in self-supervised learning, or even in the field of speech alone, is beyond the scope of the paper. Many [16], [48], [49], [50], [51], [52] have scaled self-supervised models to large data sets with large models, with a particular focus on ASR. Readers interested in large-scale experiments should consult the reference therein.

IV. Experiments

We present experiments to study the representations learned by APC on a suite of tasks. We compare several representations, the log Mel features, hidden vectors produced by random LSTMs, and hidden vectors produced by LSTMs trained with APC. As we have argued in Section I, based on the data processing inequality, no representations can include more information about the input than the input log Mel features themselves. Instead, we focus on the accessibility of information, defined as the performance of linear classifiers taking the representations as input. All LSTMs are frozen after pre-training, to avoid the confounding factor of fine-tuning. Comparing to random LSTMs helps us rule out the effect of random features [53].

It is important to acknowledge that models trained with APC do not necessarily need to improve accessibility. As we have argued in Section I, the information encoded in the learned representations depends on objectives, data sets, model architectures, and training hyperparameters. The goal is to study how the design choices affect what is learned in the representations.

The suite of tasks includes frame classification of phones, segment classification of phones, fundamental frequency tracking, vowel duration prediction, and silence duration prediction. Frame and segment classification are meant to study the phonetic information in the representations. Fundamental frequency tracking, vowel duration prediction, and silence duration prediction are meant to study the prosodic information in the representations. Finally, we also include ASR and speaker verification

to demonstrate the utility of the learned representations for practical applications.

A. Data Preparation

Three data sets, namely, Librispeech [54], Wall Street Journal (WSJ) [55], and VoxCeleb1 [56], are used in our experiments.

The LibriSpeech data sets consist of audio books crawled from LibriVox, including about 1,000 hours of read speech. It comes with three training subsets, containing 100 hours, 360 hours, and 500 hours, respectively. The data set is designed mainly for ASR, but it also provides a wide variety of speakers and recording conditions. We will use the 360-hour subset and the 960 hours combined for pre-training.

The Wall Street Journal data set consists of news texts dictated by journalists, and includes about 80 hours of read speech. It is designed mainly for ASR, and is suitable for studying phonetic properties in speech utterances. The data set does not come with a development set, so we choose 10% of the `si284` training set for tuning hyperparameters, such as learning rate and early stopping. The experiments involve training on the WSJ training set `si284`, and testing on the test sets, `eval92` and `dev93`.

The VoxCeleb1 data set consists of recordings of celebrities extracted from YouTube videos, including 1,251 speakers and a total of 352 hours of speech. The data set is designed for speaker verification, and we follow its protocol to study the learned representations.

For all data sets, we extract 40-dimensional log Mel features with a 25 ms window and a 10 ms hop. Global (instead of speaker-based) mean and variance normalization is applied to the log Mel features. The mean and variance are computed on the respective training set.

A speaker-adaptive GMM-HMM, based on Kaldi's `tri3b` recipe [57] is trained on WSJ for generating forced alignments. Though there are pronunciation variations and other sources of errors, forced alignments are a good proxy to the actual phonetic transcription. We therefore use the forced alignments as the ground truth when necessary.

B. Pre-Training on LibriSpeech

We attempt to extend the experiments in Chung et al. [15], and follow their setting closely. We train 3-layer and 6-layer unidirectional LSTMs with the APC objective on either the 360-hour subset or the 960-hour subset of LibriSpeech. All hidden layers have 512 dimensions. A linear layer is applied to the last layer for predicting the 40-dimensional log Mel features. The APC objective, as discussed in Section II, involves the prediction of future frames. We explore three numbers of time steps into the future, namely, $k \in \{3, 10, 20\}$, as target frames in the APC objective.

All LSTMs of the same type start from the same random initialization, where the Glorot initialization [58] is used. We train each model for 15 epochs with learning rate 10^{-3} using the Adam optimizer [59]. We choose a batch size of 32 for 3-layer LSTMs and 16 for 6-layer LSTMs.

There are a few simplification to the design choices in Chung et al. [15]. Among the many distance metrics, we use the ℓ_2 norm instead of the ℓ_1 norm. We do not include residual connections between layers. No dropout or data augmentation are used. We train our models for 15 epochs, instead of the reported 100 epochs.

C. Frame Classification of Phones on WSJ

We evaluate the representations with frame classification of phones on WSJ. Achieving a low error rate would suggest that the representations not only include phonetic information but also at the right timing and for the right duration. A sparse representation, such as the spikes learned by CTC [60], would not be able to low error. Note that neither is more desirable, and the goal is to reveal the properties of the learned representations.

We treat the forced alignments generated by the speaker-adaptive GMM-HMM in Section IV-B as the ground truth. We use a phone set that include 39 phones, with stressed markers removed. It also includes one special token `sil` for silence, one special token `spn` for spoken noise, and another `nsn` for non-spoken noise.

During training, we freeze the LSTM model parameters and only update the linear classifier with the cross entropy loss. We train the linear classifier using a batch size of 32 and a learning rate of 10^{-3} for 10 epochs with the Adam optimizer. We use learning rate 5×10^{-2} for training the linear layer on top of the random LSTM.

We have two sets of baselines for this task. The first baseline is a linear classifier with log Mel features as input. The other set of baselines consists of the different layers of a random LSTM. We also include the supervised counterpart of 3-layer and 6-layer LSTMs.[3]

Results for different LSTM layers and two sets of baselines are shown in Fig. 1. The PERs vary from layer to layer, but it is encouraging that most of the layers have better PERs than the baselines. The LSTM trained with $k = 3$ have better PERs across layers. Increasing the amount of pre-training data improves PERs, but the difference is less pronounced compared to varying the target time steps in APC and choosing a different layer to use. The best PER is observed when $k = 3$ for both the 3-layer LSTM and 6-layer LSTM, with the 3-layer LSTM achieving 20.8% and the 6-layer LSTM achieving 18.6%. The best PERs are surprisingly close to the supervised counterpart, suggesting the strong connection between achieving a low APC objective and acquiring phonetic information.

Our PERs across board are stronger by a wide margin than the results reported in other studies [15], [26], [62] (e.g., 7% absolute even with a 3-layer LSTM). The improvement is due to a more consistent experimental pipeline across data sets.

D. Segment Classification of Phones on WSJ

Segment classification of phones serves a similar purpose to frame classification of phones. There are subtle differences, however. For example, as we argued, segment representations can be sparse in time, while frame representations need to be

[3]The supervised 3-layer LSTM is on par with the results in Tang and Glass [61].

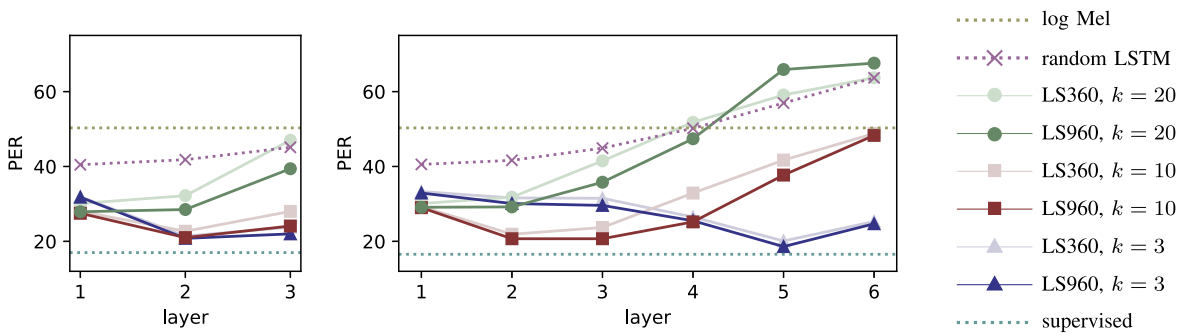

Fig. 1. Phone error rates (PERs) of frame classification on `dev93` with representations produced by 3-layer LSTMs (*left*) and 6-layer LSTMs (*right*). We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective.

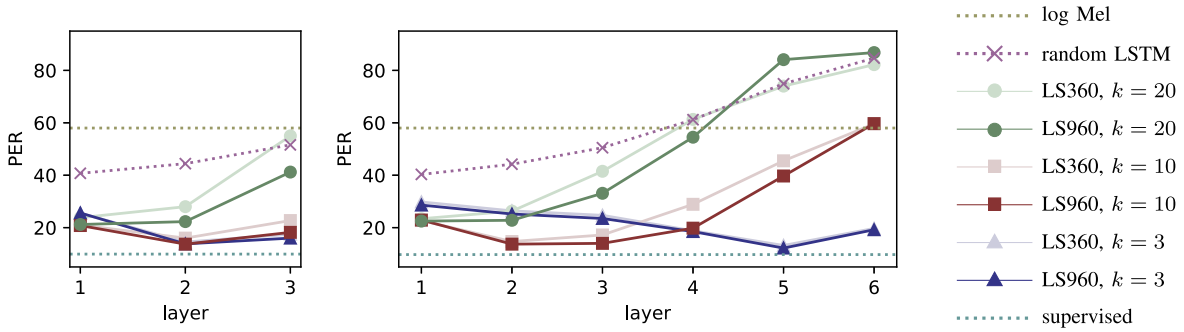

Fig. 2. Phone error rates (PERs) of segment classification on `dev93` with representations produced by 3-layer LSTMs (*left*) and 6-layer LSTMs (*right*). We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective.

dense. In terms of evaluation, frame classification favors phones with longer duration, while segment classification is more tolerable to timing and duration errors. Achieving a low error rate in segment classification suggests that the frame representations can be compressed into fixed-dimensional vectors.

We use the forced alignments as the ground truth of phone boundaries. For segment representation, we simply average the frame representations within a segment. For example, the segment representations are the log Mel features averaged within a segment or the hidden vectors produced by the LSTMs averaged within a segment. This type of segment representation is used in various other studies [28], [63]. We follow the same experimental setting as in frame classification, training and testing on the same subsets with the same model architectures and hyperparameters. We also include the same sets of baselines and the supervised counterpart.

Results are shown in Fig. 2. The findings are largely the same as in frame classification. The gap between the supervised counterpart and best LSTM representation learned with APC is again surprisingly small.

E. Fundamental Frequency Tracking on WSJ

Fundamental frequency (f_0) has been shown to correlate well with the perception and production of word prominence, and is part of several prosodic features humans exercise to convey meaning and intent [64]. Fundamental frequency can in principle be extracted based on harmonics in narrow-band spectrograms. We evaluate the representations on f_0 tracking to identify whether f_0 information is encoded in the representations.

We use the fundamental frequency extracted by the PYIN algorithm [65], an approach based on autocorrelation, as the ground truth. We set the minimum and maximum frequency in PYIN to 50 Hz and 600 Hz respectively. A frame rate of 10 ms (as opposed to 5 ms commonly used in TTS [33]) is used to be consistent with the frame rate of the log Mel features.

We present two baselines, a linear regression model based on log Mel features, and a linear regression on top of a 3-layer random LSTM taking log Mel features as input. We also include a supervised counterpart for the 3-layer LSTM. The LSTMs has 512 dimensions in each layer. The mean squared error is used for training. We use Adam with a learning rate of 10^{-3} for the supervised LSTM, and 10^{-2} for the linear regression on top of the LSTMs, random or trained with APC. We use root-mean-square error (RMSE) in Hz to evaluate the performance of f_0 prediction only on sonorants identified based on the forced alignments prepared in Section IV-B.

The experiment results are shown in Fig. 3. Most of the cases are significantly better than the baselines, with the best performing result close to the supervised counterpart.

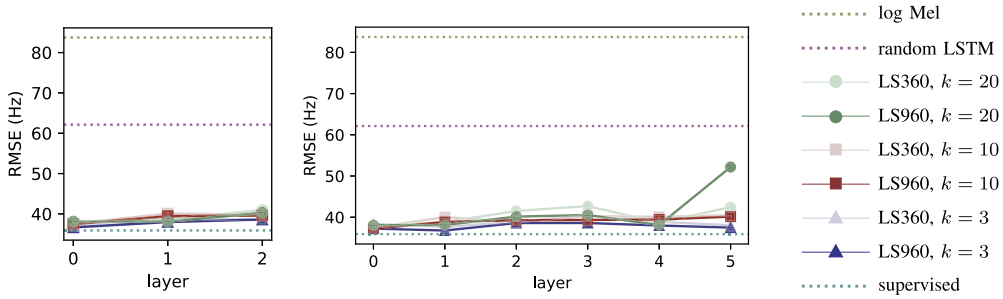

Fig. 3. Root-mean-square errors (RMSE) of f_0 tracking in Hz with representations produced by 3-layer LSTMs (*left*) and 6-layer LSTMs (*right*). We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective. The results for log Mel and the last layer of the random 3-layer LSTM are 83.7 and 60.2, respectively.

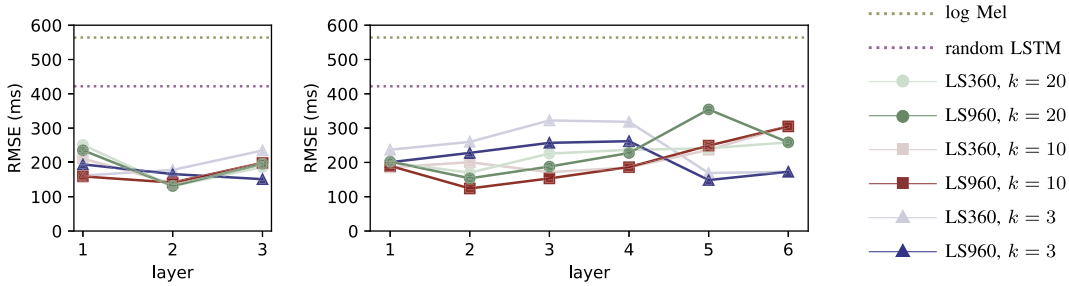

Fig. 4. Root-mean-square error (RMSE) of silence duration prediction in milliseconds with representations produced by 3-layer LSTMs (left) and 6-layer LSTMs (right). We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective. Predicting with the mean duration achieves 594 ms and is not shown for clarity.

F. Duration Prediction for Vowels and Silence on WSJ

Similar to the fundamental frequency, duration has also been shown to correlate well with the perception and production of word prominence [64]. For example, vowel lengthening and pauses are the common prosodic cues for prominence. In our experiments, we focus on two proxy tasks, vowel duration prediction and silence duration prediction. Achieving a low error rate suggests that the representations are able to signal these prosodic cues when they are present.

Similar to segment classification, we extract segment representations by simply taking the average, and use linear regression to predict the duration of segments. Since the goal is to target pauses and not the silence at the beginning and at the end of an utterance, we only consider silence within an utterance. We make use the forced alignments prepared in Section IV-B to identify the segments and use their duration as the ground truth. Three baselines are considered. First, we have a linear model taking averaged log Mel features within a segment as input. The second baseline is a random 3-layer LSTM, where a linear model takes the average frame representation produced by the top layer of the LSTM as input. The third is to predict the duration based on the phone or silence identity itself, and this amounts to predicting the mean duration of a phone or silence. Note that the task is trivial if an LSTM embeds a counter in the representation, so we do not provide a supervised counterpart for these experiments. We use Adam with a learning rate of 0.01 to train the linear layers. The root-mean-square errors (RMSE) in milliseconds are reported.

Results for predicting silence duration are shown in Fig. 4. The representations learned by APC are better than the log Mel features, the random LSTM, and the mean duration. However, predicting silence duration seems difficult in general. The overall RMSE range spans tens, even hundreds, of milliseconds.

Results for predicting vowel duration are shown in Fig. 5. The representations learned by APC are better than log Mel features and predicting the mean duration. The RMSE's of a supervised model purely based on phone identities (not limited to vowels) are typically within 40 ms [66]. What is surprising is the strong performance of the random LSTM. Since counting between boundaries is sufficient to do well on this task, we suspect random LSTMs are capable of counting between sharp changes. In general, LSTMs trained with APC do not have satisfactory performance on duration-related tasks. More analyses are needed to understand the root cause of this issue.

G. Automatic Speech Recognition on WSJ

As reviewed in Section III, various other studies have shown the usefulness of self-supervised representations for ASR. Our goal for evaluating representations on ASR, besides showing the usefulness on a practical application, is to study the properties of the representations. Compared to frame classification and segment classification, ASR is a sequence prediction task, even more tolerable to errors in timing and duration. For example, it is common to have representations learned with a unidirectional LSTM on ASR shifted in time, to accommodate the delay in real time and lookahead [60]. Achieving a low error rate in ASR suggests that the sequential order of the phonetic information is preserved.

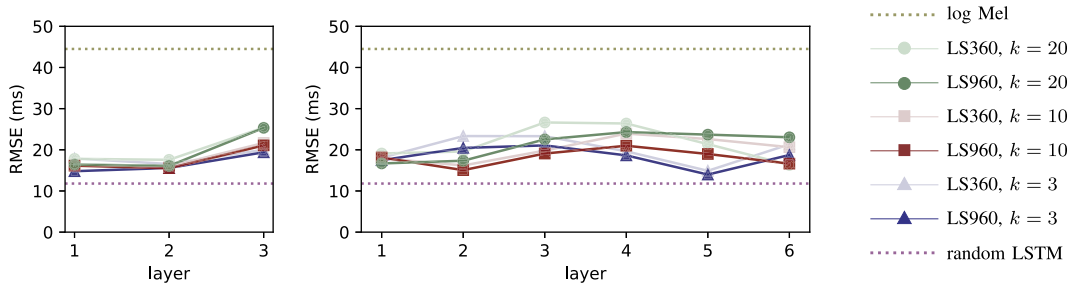

Fig. 5. Root-mean-square error (RMSE) of vowels duration prediction in milliseconds with representations produced by 3-layer LSTMs (left) and 6-layer LSTMs (right). We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective. Predicting with the mean duration achieves 40.4 ms and is not shown for clarity.

TABLE I
WORD AND CHARACTER ERROR RATES (%) ON WSJ TEST SETS COMPARING DIFFERENT INPUT REPRESENTATION TO A SEQUENCE-TO-SEQUENCE MODEL

Representation		dev93		eval92	
		WER	CER	WER	CER
log Mel		18.2	6.8	14.7	5.1
3-layer LSTM	LS360	18.0	6.5	14.2	4.8
	LS960	17.2	6.0	13.6	4.6
6-layer LSTM	LS360	17.8	6.2	14.4	4.9
	LS960	16.4	5.9	12.6	4.3

We follow the experimental setting of Kim et al. [67] and Chung and Glass [35], training character-based sequence-to-sequence models [68] end to end on WSJ. The character set includes 26 alphabets, common punctuation marks, a noise token <NOISE>, and a whitespace token <space> to delineate words. We limit ourselves to the case of $k = 3$ and only use the last layer of LSTMs (consistent with Chung and Glass [35]) for training the ASR system, because, as we know from Section IV-C, $k = 3$ produces representations that contain the most phonetic information.

In terms of the model architecture, the encoder has two convolutional layers with 32 and 32 channels each and strides of 2 and 2, followed by a 4-layer bidirectional GRU [69] with 256 dimensions for each layer. This results in a total down-sampling factor of four on both time and feature dimensions. The decoder is a unidirectional 256-dimension GRU. Models are trained with a fixed scheduled sampling probability of 0.4. Beam search with a beam size of 5 is used for decoding. We use Adam with learning rates of 2×10^{-4} for log Mel features and 10^{-4} for APC representations. After 100 epochs, another 50 epochs are run with a learning rate with a factor of 0.05 for log Mel features, while only 10 epochs are run for APC representations. We apply a dropout rate of 0.2, and a label smoothing rate of 0.1 for regularization. Layer normalization is added after each bidirectional GRU layer in the encoder. We acknowledge that the model architecture is relatively small compared to the ones that perform well on larger data sets [70], but the setting is sufficient to answer whether the sequential order of phones can be extracted from the learned representations.

We report word error rates (WERs) and character error rates (CERs) on both test sets, namely eval92 and dev93, in Table I. A baseline with 40-dimensional log Mel features as input is also presented and achieves results similar to prior work [71]. APC representations improve WERs and CERs up to 10% relative. The improvement is more pronounced compared to frame and segment classification of phones as we increase the amount of pre-training data, suggesting that special properties in the representations, for example of syllables or even words, become more accessible but cannot be reflected in frame and segment classification of phones.

H. Speaker Verification on VoxCeleb1

It is difficult to quantify what exactly constitutes a speaker identity. Factors, such as voice characteristics related to the vocal tract, accent and phonological preferences, speaking style, and word choices, all play a role in identifying a speaker. We evaluate our representations broadly on speaker verification, and do not attempt to study the individual factors. Typically, the frame representations need to be aggregated, using averaging for example, to obtain utterance-level representations. Achieving a low error rate in this task suggests that the frame representations are able to be aggregated and can capture one or more of the factors that differentiate speakers.

We use averaging to aggregate the frame representations as is typically done for neural network-based speaker verification [72]. We include two baselines, one with the log Mel features as frame representations, and the other a random 3-layer LSTM. The averaged vectors directly serve as speaker representations. We compute the cosine of vectors as the similarity score, and sweep a threshold to compute the equal error rates (EERs).

The EERs are shown in Fig. 6. While the results are all better than the baselines, there does not seem to have a clear trend. Inspired by the success of Fan et al. [73], we suspect that averaging is too crude an approach with no training involved. We hypothesize that there exists a subspace that is better for averaging to aggregate speaker information.

Instead of directly averaging, we extract speaker representation by training a linear layer to identify speakers [74]. We first apply a 512×512 linear transformation to the frame representations, followed by an averaging to obtain the speaker representation. We then apply an additional linear layer to predict the speaker identity, and the two linear layers are trained on the VoxCeleb1 training set. Only the first linear layer is used to

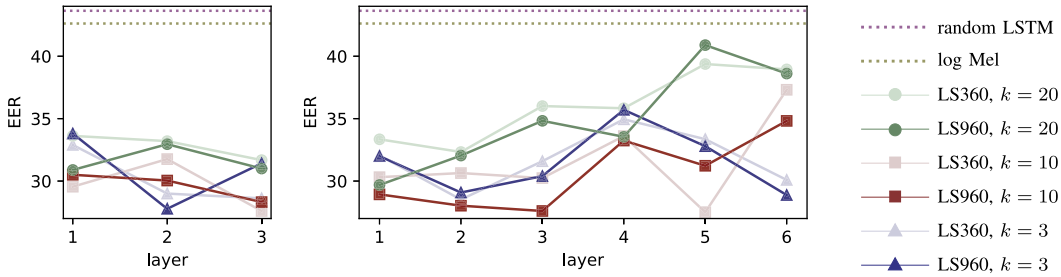

Fig. 6. Equal error rates (EERs) of speaker verification with representations produced by 3-layer LSTMs (left) and 6-layer LSTMs (right) without any training process. We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective.

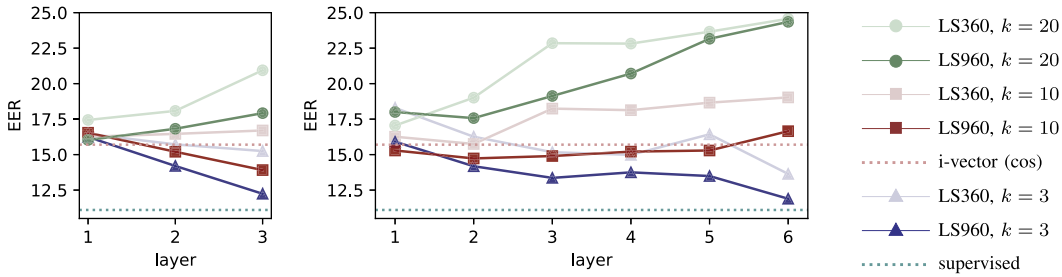

Fig. 7. Equal error rates (EERs) of speaker verification with representations produced by 3-layer LSTMs (left) and 6-layer LSTMs (right), two linear layers are trained. We use LS360 and LS960 to denote the LSTMs trained on the 360-hour subset and the 960 hours combined of LibriSpeech, respectively. We use k to denote the number of time steps into the future in the APC objective. The i-vector with cosine similarity achieves 15.7%, 8.6% with LDA, and 5.3% with PLDA.

extract and evaluate the speaker representations. We also train a 3-layer LSTM (with the additional linear layer) as the supervised counterpart, and include a few i-vector results. We use the cosine similarity to compute the EERs. Cross entropy is optimized with Adam and a learning rate of 10^{-3} for 10 epochs.

Results are shown in Fig. 7. With the additional linear layer, the results are substantially better than the ones in Fig. 6. The best performing result is approaching the supervised counterpart, and better than a vanilla i-vector with cosine similarity. The case of $k = 3$ dominates all the others, with the best 3-layer LSTM achieving 12.2% and 6-layer LSTM achieving 11.9%. Our results are also significantly better than the 15.6% with wav2vec 2.0 reported in [73]. The benefit of increasing the pre-training is also more pronounced.

I. Comparison to Existing Self-Supervised Approaches

Finally, we compare several generalizations of APC on phone classification, including VQ-APC [26], MPC [14], band reconstruction [24], and a version of wav2vec 2.0 termed Mel2vec [52] based on log Mel features instead of waveforms. Since the losses are all generalizations of APC, we are able to make comparisons while holding the model architecture fixed.

For VQ-APC, we follow [26], adding a quantization layer on top of the last LSTM layer to learn discrete speech representations for future prediction. We experiment different codebook sizes of 100, 256, 512 for quantization and report the one that gives the best PER.

For MPC, we follow Jiang et al. [14], dividing the input frames into 4-frame chunks, and randomly masking them with a probability of 0.25. For band reconstruction, we follow Wang et al. [24], masking two bands of consecutive frames of sizes randomly sampled from $\{16, 17, \ldots, 32\}$, and masking one frequency band of size sampled from $\{1, 2, \ldots, 8\}$. Losses are computed only on the masked portion.

TABLE II
PHONE ERROR RATES (%) OF FRAME CLASSIFICATION ON WSJ `dev93` COMPARING DIFFERENT TRAINING LOSSES FOR PRE-TRAINING

	PER(%) ↓		
	h_1	h_2	h_3
APC	31.9	21.4	23.8
VQ-APC	28.4	21.7	28.9
MPC	39.8	33.0	24.4
band recon.	39.8	35.6	30.0
Mel2vec	32.2	23.7	26.7

For Mel2vec, we follow Misra et al. [52], sampling 100 negative samples from the same utterance while randomly masking 10-frame chunks with a probability of 0.065 (the same masking strategy as wav2vec 2.0 [10]). However, we do not quantize the log Mel features prior to computing the contrastive loss.

The setting is identical to the ones in Section IV-C. Results are shown in Table II. The performance of MPC and Mel2vec are close to APC, while band reconstruction has a larger gap behind others. Though APC achieves the best performance, this finding might only hold for unidirectional LSTMs, as band reconstruction has been shown to work well with bidirectional LSTMs [24] and masked reconstruction has been shown to work well with Transformers [14].

To provide a fair comparison among all the training approaches, it would require a significant amount of tuning for each individual approach in Table II. One goal of unifying the views of these approaches is to provide a single training objective that practitioners can use to compare across approaches. For

example, log likelihood is a potential candidate, given that all approaches are inherently generative models. It is in fact a common practice to measure and compare log likelihoods for studying variational autoencoders [75], [76]. However, computing log likelihoods often requires approximation, especially for the approaches that are only tractable via computing variational lower bounds. There are other measurements, such as conditional independence and cluster tightness [77], [78]. In general, it is still an open problem as to what characterizes the connection between self-supervised training objectives and performance on the downstream tasks. One future direction is to estimate either log likelihood or mutual information of the approaches and to establish the connection between these measurements and the performance on the downstream tasks.

V. Discussion

In this paper, we have presented three different interpretations of the APC objective that welcome various generalizations. We have also shown the utility and properties of the representations learned by APC. Increasing the amount of pre-training data is useful across the board. Several best results are even close to the supervised counterparts. The representations for $k = 3$ are particularly strong for various tasks, reflecting phonetic and prosodic cues when they are present. Based on frame classification and fundamental frequency tracking, it also suggests that the representation for $k = 3$ is likely to be time synchronous, reflecting accurate timing of the acoustic events in speech.

Though APC serves as a strong objective for learning various aspects of speech, it also has limitations. Bender and Koller [79] have argued that, without proper grounding, it is impossible to learn semantics purely based on the surface form of texts. In other words, it is impossible to infer hidden variables without making assumptions when only given the observable variables. This is generally known as the identifiability problem. A corollary of this argument is that, without making the proper assumptions, none of the models, those trained with APC included, are able to learn semantics from speech alone.

Whether self-supervised models are unable to learn semantics still needs to be formally proved. However, the statement does suggest that self-supervised models might struggle to learn whether a phonetic property is phonemic or not. For example, subtle differences in spectrograms, such as vowel nasalization, might be difficult to learn. Self-supervised models might also struggle to differentiate homophones (such as *there* and *their*) and words with different senses (such as a financial *bank* and a river *bank*), as there are no phonetic distinction to help separate these words.

In practice, it is still entirely possible to learn representations that correlate with properties that are difficult to identify exactly. These limitations, if exist at all, can be overcome through grounding and better modeling, for example, including other modalities and more modeling assumptions.

References

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[2] P. Elias, "Predictive coding–I," *IRE Trans. Inf. Theory*, vol. 1, no. 1, pp. 16–24, Mar. 1955.

[3] P. Elias, "Predictive coding–II," *IRE Trans. Inf. Theory*, vol. 1, no. 1, pp. 24–33, Mar. 1955.

[4] N. Littlestone and M. K. Warmuth, "Relating data compression and learnability," Univ. of California at Santa Cruz, Tech. Rep., 1986.

[5] O. David, S. Moran, and A. Yehudayoff, "On statistical learning via the lens of compression," in *Proc. 30th Int. Conf. Neural Inf. Process. Syst.*, 2016, pp. 2792–2800.

[6] M. E. Peters et al., "Deep contextualized word representations," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2018, pp. 2227–2237.

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[8] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "XLNet: Generalized autoregressive pretraining for language understanding," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 5753–5763.

[9] S. Schneider, R. C. A. Baevski, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," in *Proc. 20th Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 3465–3469.

[10] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. 34th Int. Conf. Neural Inf. Process. Syst.*, 2020, pp. 12449–12460.

[11] S. Ling, Y. Liu, J. Salazar, and K. Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 6429–6433.

[12] S. Ling and Y. Liu, "DeCoAR 2.0: Deep contextualized acoustic representations with vector quantization," 2021, *arXiv:2012.06659*.

[13] D. Jiang et al., "Improving transformer-based speech recognition using unsupervised pre-training," 2019, *arXiv:1910.09932*.

[14] D. Jiang et al., "A further study of unsupervised pretraining for transformer based speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2021, pp. 6538–6542.

[15] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. Glass, "An unsupervised autoregressive model for speech representation learning," in *Proc. 20th Annu. Conf. Int. Speech Commun. Assoc.*, 2019, pp. 146–150.

[16] Y. Zhang et al., "BigSSL: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition," *IEEE J. Sel. Top. Signal Process.*, pp. 1–14, 2022, doi: 10.1109/JSTSP.2022.3182537.

[17] Y. Wang et al., "Tacotron: Towards end-to-end speech synthesis," in *Proc. Interspeech*, 2017, pp. 4006–4010, doi: 10.21437/Interspeech.2017-1452.

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

[19] D. Jiang, W. Li, M. Cao, W. Zou, and X. Li, "Speech SimCLR: Combining contrastive and reconstruction bbjective for self-supervised speech representation learning," in *Proc. Interspeech*, 2021, pp. 1544–1548, doi: 10.21437/Interspeech.2021-391.

[20] A. T. Liu, S.-W. Li, and H.-Y. Lee, "TERA: Self-supervised learning of transformer encoder representation for speech," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 2351–2366, 2021.

[21] X. Liu et al., "Self-supervised learning: Generative or contrastive," *IEEE Trans. Knowl. Data Eng.*, early access, Jun. 22, 2021, doi: 10.1109/TKDE.2021.3090866.

[22] C. M. Bishop, "Mixture density networks," Neural Comput. Res. Group, Aston Univ., Tech. Rep., 1994.

[23] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6309–6318.

[24] W. Wang, Q. Tang, and K. Livescu, "Unsupervised pre-training of bidirectional speech encoders via masked reconstruction," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 6889–6893.

[25] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1096–1103.

[26] Y.-A. Chung, H. Tang, and J. Glass, "Vector-quantized autoregressive predictive coding," in *Proc. 22nd Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 3760–3764.

[27] J. Chorowski et al., "Aligned contrastive predictive coding," in *Proc. 22nd Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 976–980.

[28] S. Bhati, J. Villalba, P. Zelasko, L. Moro-Velazquez, and N. Dehak, "Unsupervised speech segmentation and variable rate representation learning using segmental contrastive predictive coding," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 2002–2014, 2022.

[29] D. McAllester, "Information theoretic co-training," 2018, *arXiv:1802.07572*.

[30] L. Kong, C. de Masson d'Autume, W. Ling, L. Yu, Z. Dai, and D. Yogatama, "A mutual information maximization perspective of language representation learning," in *Proc. Int. Conf. Learn. Representations*, 2020.

[31] J. Bai, W. Wang, Y. Zhou, and C. Xiong, "Representation learning for sequence data with deep autoencoding predictive components," in *Proc. Int. Conf. Learn. Representations*, 2021.

[32] K.-F. Lee, *Automatic Speech Recognition: The Development of the SPHINX System*. Berlin, Germany: Springer, 1988.

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

[34] J. Chorowski, R. J. Weiss, S. Bengio, and A. van den Oord, "Unsupervised speech representation learning using WaveNet autoencoders," *IEEE/ACM Trans. Audio, Speech Lang. Process.*, vol. 27, no. 12, pp. 2041–2053, Dec. 2019.

[35] Y.-A. Chung and J. Glass, "Generative pre-training for speech with autoregressive predictive coding," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2020, pp. 3497–3501.

[36] R. Collobert and J. Weston, "A unified architecture for natural language processing: Deep neural networks with multitask learning," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 160–167.

[37] H. Mobahi, R. Collobert, and J. Weston, "Deep learning from temporal coherence in video," in *Proc. 26th Int. Conf. Mach. Learn.*, 2009, pp. 737–744.

[38] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[39] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1422–1430.

[40] A. M. Dai and Q. V. Le, "Semi-supervised sequence learning," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 3079–3087.

[41] N. Srivastava, E. Mansimov, and R. Salakhutdinov, "Unsupervised learning of video representations using LSTMs," in *Proc. 32nd Int. Conf. Mach. Learn.*, 2015, pp. 843–852.

[42] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.

[43] G. Larsson, M. Maire, and G. Shakhnarovich, "Learning representations for automatic colorization," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 577–593.

[44] M. Noroozi and P. Favaro, "Unsupervised learning of visual representations by solving jigsaw puzzles," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 69–84.

[45] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 2536–2544.

[46] Y. Belinkov and J. Glass, "Analyzing hidden representations in end-to-end automatic speech recognition systems," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 2438–2448.

[47] S. A. Chowdhury, N. Durrani, and A. Ali, "What do end-to-end speech models learn about speaker, language and channel information? A layer-wise and neuron-level analysis," 2021, *arXiv:2107.00439*.

[48] Y. Zhang et al., "Pushing the limits of semi-supervised learning for automatic speech recognition," in *Proc. NeurIPS SAS Workshop*, 2020.

[49] Y.-A. Chung et al., "W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *Proc. IEEE Autom. Speech Recognit. Understanding Workshop*, 2021, pp. 244–250.

[50] C. Wang et al., "UniSpeech at scale: An empirical study of pre-training method on large-scale speech recognition dataset," 2021, *arXiv:2101.07597*.

[51] C. Wang et al., "UniSpeech: Unified speech representation learning with labeled and unlabeled data," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 10937–10947.

[52] A. Misra et al., "A comparison of supervised and unsupervised pre-training of end-to-end models," in *Proc. 22nd Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 731–735.

[53] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Proc. 20th Int. Conf. Neural Inf. Process. Syst.*, 2007, pp. 1177–1184.

[54] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2015, pp. 5206–5210.

[55] D. B. Paul and J. M. Baker, "The design for the wall street journal-based CSR corpus," in *Proc. Workshop Speech Natural Lang.*, 1992, pp. 357–362.

[56] A. Nagrani, J. S. Chung, W. Xie, and A. Zisserman, "Voxceleb: Large-scale speaker verification in the wild," *Comput. Speech Lang.*, vol. 60, 2020, Art. no. 101027.

[57] D. Povey et al., "The Kaldi speech recognition toolkit," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding*, 2011.

[58] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[59] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.

[60] H. Sak, A. Senior, K. Rao, and F. Beaufays, "Fast and accurate recurrent neural network acoustic models for speech recognition," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2015, pp. 1468–1472.

[61] H. Tang and J. Glass, "On training recurrent networks with truncated backpropagation through time in speech recognition," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 48–55.

[62] A. H. Liu, Y.-A. Chung, and J. Glass, "Non-autoregressive predictive coding for learning speech representations from local dependencies," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2021, pp. 3730–3734.

[63] H. Tang et al., "End-to-end neural segmental models for speech recognition," *IEEE J. Sel. Top. Signal Process.*, vol. 11, no. 8, pp. 1254–1264, Dec. 2017.

[64] J. Cole, "Prosody in context: A review," *Lang. Cogn. Neurosci.*, vol. 30, pp. 1–31, 2015.

[65] M. Mauch and S. Dixon, "PYIN: A fundamental frequency estimator using probabilistic threshold distributions," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2014, pp. 659–663.

[66] G. E. Henter, S. Ronanki, O. Watts, M. Wester, Z. Wu, and S. King, "Robust TTS duration modelling using DNNs," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 5130–5134.

[67] S. Kim, T. Hori, and S. Watanabe, "Joint CTC-attention based end-to-end speech recognition using multi-task learning," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process.*, 2017, pp. 4835–4839.

[68] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2015, pp. 577–585.

[69] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder–decoder approaches," in *Proc. Workshop Syntax Semantics Struct. Stat. Transl.*, 2014, pp. 103–111.

[70] Z. Tuske, G. Saon, K. Audhkhasi, and B. Kingsbury, "Single headed attention based sequence-to-sequence model for state-of-the-art results on Switchboard," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2020, pp. 551–555.

[71] J. Chorowski and N. Jaitly, "Towards better decoding and language model integration in sequence to sequence models," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 523–527.

[72] S. Shon, H. Tang, and J. Glass, "Frame-level speaker embeddings for text-independent speaker recognition and analysis of end-to-end model," in *Proc. IEEE Spoken Lang. Technol. Workshop*, 2018, pp. 1007–1013.

[73] Z. Fan, M. Li, S. Zhou, and B. Xu, "Exploring wav2vec 2.0 on speaker verification and language identification," in *Proc. Interspeech*, 2020, pp. 1509–1513, doi: 10.21437/Interspeech.2021-1280.

[74] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *Proc. Annu. Conf. Int. Speech Commun. Assoc.*, 2017, pp. 999–1003.

[75] J. Tomczak and M. Welling, "VAE with a VampPrior," in *Proc. 21st Int. Conf. Artif. Intell. Statist.*, 2018, pp. 1214–1223.

[76] M. Wu and N. Goodman, "Multimodal generative models for scalable weakly-supervised learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 5580–5590.

[77] J. D. Lee, Q. Lei, N. Saunshi, and J. Zhuo, "Predicting what you already know helps: Provable self-supervised learning," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2021, pp. 309–323.

[78] S. Arora, H. Khandeparkar, M. Khodak, O. Plevrakis, and N. Saunshi, "A theoretical analysis of contrastive unsupervised representation learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 5628–5637.

[79] E. M. Bender and A. Koller, "Climbing towards NLU: On meaning, form, and understanding in the age of data," in *Proc. 58th Annu. Meeting Assoc. Comput. Linguistics*, 2020, pp. 5185–5198.