Pre-Training Audio Representations With Self-Supervision

Marco Tagliasacchi, Beat Gfeller, Félix de Chaumont Quitry, and Dominik Roblek

***Abstract*—We explore self-supervision as a way to learn general purpose audio representations. Specifically, we propose two self-supervised tasks: `Audio2Vec`, which aims at reconstructing a spectrogram slice from past and future slices and `TemporalGap`, which estimates the distance between two short audio segments extracted at random from the same audio clip. We evaluate how the representations learned via self-supervision transfer to different downstream tasks, either training a task-specific linear classifier on top of the pretrained embeddings, or fine-tuning a model end-to-end for each downstream task. Our results show that the representations learned with `Audio2Vec` transfer better than those learned by fully-supervised training on Audioset. In addition, by fine-tuning `Audio2Vec` representations it is possible to outperform fully-supervised models trained from scratch on each task, when limited data is available, thus improving label efficiency.**

***Index Terms*—Self-supervised learning, audio processing.**

I. Introduction

THANKS to advances in supervised learning, it is now possible to train models that are able to successfully perform a variety of audio tasks. Despite the indisputable success, this approach suffers from two main shortcomings. First, it requires collecting large annotated datasets specific to each task to be solved. Second, separate models are typically trained for each task, making it difficult to reuse computational resources when multiple such models are deployed on a mobile device.

Unsupervised learning attempts to overcome these limitations, by making it possible to learn from widely available unlabelled datasets and by learning general purpose representations that can be reused for different downstream tasks. In the area of unsupervised learning, self-supervised learning has emerged as an attractive approach [1]–[3].

In a nutshell, this approach formulates an auxiliary task based on the available unlabelled data and a fully-supervised model is trained to solve this task. The key idea is that by solving the auxiliary task, the model is also learning some general purpose representations in a lower dimensional embedding space. Therefore, the embedding encoder, e.g., the portion of the model architecture mapping the input data to the embedding space, can be reused as a feature extractor for different downstream tasks.

One of the earliest successes of self-supervised learning was obtained in the context of language models, where `Word2Vec` is used to map one-hot-encoded words to word embeddings [4]. `Word2Vec` can be formulated in two variants: i) continuous bag-of-words (CBoW), or ii) skip-gram. In the former, the model predicts the current word based on the context of surrounding words. In the latter, the model predicts surrounding words given the current word. Recently, a similar approach has been proposed to map speech to fixed-dimensional embeddings [5]–[7]. The model used to solve the `Speech2Vec` task relies on a speech segmentation step to isolate the temporal slices of the spectrogram corresponding to different words. Then, a RNN encoder-decoder is used to handle variable-length inputs and outputs. In contrast, in our work we do not rely on the specific characteristics of speech and we evaluate our results also on non-speech related tasks.

In this letter we explore self-supervised learning of audio representations. We posit that contextual temporal information can be exploited in the case of general audio signals without resorting to any form of explicit supervision. We argue that solving properly designed tasks that involve the temporal context requires extracting some sort of high level semantic information from the underlying raw data, thus leading to reusable embeddings. In this respect, this letter makes the following main contributions:

(i) We propose `Audio2Vec`, a self-supervised learning task that is inspired by `Word2Vec`, but applied to audio spectrograms. In the CBoW formulation (Fig. 1(a)) the auxiliary task consists of reconstructing a temporal slice of pre-determined duration from a number of past and future slices. In the skip-gram formulation (Fig. 1(b)) the roles of the target and surrounding slices are reversed.

(ii) We propose `TemporalGap`, a self-supervised learning task that consists of estimating the distance in time between any two pairs of audio segments extracted at random from a longer audio clip (Fig. 1(c)).

(iii) We quantitatively evaluate the quality of the embeddings produced by the feature encoders obtained by solving the aforementioned self-supervised tasks, and compare it to existing approaches such as autoencoder and triplet loss. To this end we consider a wide variety of downstream tasks, including speech detection, music detection, speaker identification and language identification, among others. Our results show that the representations

Manuscript received February 5, 2020; revised March 30, 2020; accepted March 30, 2020. Date of publication April 8, 2020; date of current version April 30, 2020. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Maximo Cobos. *(Corresponding author: Marco Tagliasacchi.)*

The authors are with the Google Research, 8002 Zurich, Switzerland (e-mail: mtagliasacchi@google.com; beatg@google.com; fcq@google.com; droblek@google.com).

Digital Object Identifier 10.1109/LSP.2020.2985586

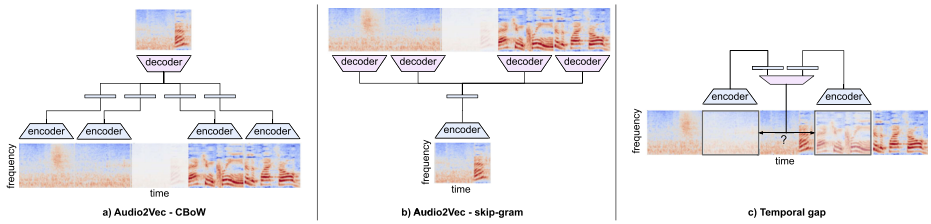


Fig. 1. Overview of the proposed self-supervised learning tasks.

learned with `Audio2Vec` transfer better than those learned by full-supervised training on Audioset [8]. In addition, by fine-tuning `Audio2Vec` representations it is possible to outperform fully-supervised models trained from scratch on each task, in a setting where the number of training examples per downstream task is limited to 1000, thus improving label efficiency.

II. Related Work

Learning audio representations: Unsupervised feature learning can lead to more descriptive representations than traditional handcrafted features, e.g., MFCCs. For example, different autoencoder architectures have been explored, e.g., denoising [9], convolutional LSTM autoencoders [10] and sequence-to-sequence autoencoders [11]. A self-supervised version of the triplet loss is proposed in [3]. In the absence of labels, the authors create anchor-positive pairs by adding noise, shifting in time and/or frequency, and sampling temporal neighbors. A self-supervised task based on temporal data is proposed in [12]. Speech-specific representations based on self-supervision have been explored in [5]–[7], [13], [14]

Learning visual representations: Several auxiliary tasks have been explored to learn image representations, e.g., predicting the relative position of a pair of patches extracted from the same image [15], re-ordering image patches and solving jigsaw puzzles [1], or asking the model to discriminate between a patch and transformed version of it [16]. In some cases solving seemingly simple tasks can lead to very powerful representations such as, for example, detecting image rotations [2]. In other cases, representations can be learned as a by-product of solving useful tasks, e.g., in the case of image colorization [17] and image inpainting [18]. In the case of video, it is possible to exploit the temporal dimension to learn visual representations by asking a model to learn whether frames are in the correct temporal order [19], [20], to infer motion by observing a static image [21], or detect whether a video is playing forwards or backwards [22].

Learning multimodal representations: Several letters have recently investigated learning audio representations exploiting the correlation with other modalities, e.g., text [14], [23], images [24] and videos [25]–[29].

III. Methods

Let $x = \{x_1, x_2, \ldots, x_n\}$ denote an audio clip of n samples in the time domain and $X \in \mathbb{R}^{T \times F}$ the corresponding mel spectrogram, which consists of T temporal frames and F frequency bins. Note that we compute the logarithm of the modulus of the spectrogram to compress the dynamic range of the amplitudes. Let X_i denote a $N \times F$ slice of the spectrogram X, starting at frame i with N temporal frames and $z_i = Enc(X_i)$ a d-dimensional embedding computed by processing the input spectrogram X_i with an encoder $Enc()$, whose architecture is detailed in Section IV. Using this notation, in the following we describe the proposed self-supervised learning models.

Audio2Vec: The task comes in two variants. In the CBoW variant, we first select a target slice at random, together with a set of surrounding slices used for prediction. Each of the predictor slices is processed by the same encoder, which maps its input into a fixed-dimensional embedding. These embeddings are then concatenated and fed into a decoder, whose architecture mirrors the one of the encoder, which computes a reconstruction of the target slice. More specifically, let $X_{(0)} = X_i$ be a slice selected at random from X. Then, a set of past $(X_{(-P)}, \ldots, X_{(-1)})$ and future slices $(X_{(1)}, \ldots, X_{(P)})$ are extracted from the same audio clip. The temporal location of the start of slice $X_{(p)}$ is equal to $X_{i+p(N+G)}$, i.e., we consider non-overlapping slices of size N, with an extra gap of G temporal frames between any two consecutive slices. The gap is introduced to avoid that the self-supervised model exploits the leakage between adjacent STFT temporal frames as a *shortcut* [1] to solve the task. Each slice is processed by the same encoder to obtain $z_{(p)} = Enc(X_{(p)})$. Then, a vector $\bar{z}_{(0)} = [z_{(-P)}, \ldots, z_{(-1)}, z_{(1)}, \ldots, z_{(P)}]$ is obtained by concatenating the embeddings of each of the predictor slices and fed into a convolutional decoder to obtain a reconstruction $\hat{X}_{(0)} = Dec(\bar{z}_{(0)})$. Note that the architecture of the decoder is obtained by reversing the order of the layers in the encoder and replacing max-pooling with nearest-neighbor upsampling. The overall encoder-decoder architecture is trained end-to-end by minimizing the mean-square error loss function $\|X_{(0)} - \hat{X}_{(0)}\|$.

The skip-gram variant of `Audio2Vec` uses a similar architecture. In this case we compute the embeddings of the middle slice $z_{(0)} = Enc(X_{(0)})$, and then let the decoder reconstruct the surrounding slices, i.e., $[\hat{X}_{(-P)}, \ldots, \hat{X}_{(-1)}, \hat{X}_{(1)}, \ldots, \hat{X}_{(P)}] = Dec(z_{(0)})$. The decoder is identical to the one used by the CBoW variant, except for one important difference: the last convolutional layer has $2P$ output channels, one for each of the slices to be reconstructed. The loss function minimizes the average mean-square error computed across the $2P$ reconstructed slices.

Temporal gap: For the `TemporalGap` task, we ask the model to estimate the absolute value of the distance in time between two slices sampled at random from the same audio clip. More specifically, we sample the ground truth temporal gap from a uniform distribution, i.e., $\Delta \sim \mathcal{U}(0, N_{\max} - N)$,

where N and $N_{\max}$ are the lengths (in time frames) of the slices and the original sample, respectively, and define the normalized temporal gap as $\delta = \Delta/(N_{\max} - N) \in [0, 1]$. Then, we extract two slices X_i and X_j such that $\Delta = |i - j|$. Note that we do not impose a temporal order between the two slices. We concatenate the embedding representations in a single $2d$-dimensional vector $z = [Enc(X_i), Enc(X_j)]$ and we feed this vector into a fully connected feed forward network with a single hidden layer of size 64 that produces the scalar output $\hat{\delta}$. We train the model end-to-end so as to minimize a cross-entropy loss $\mathcal{L}_{CE}(\delta, \hat{\delta})$ between the ground-truth and the predicted gap. In our experiments, we found that this loss is to be preferred to the mean-square error, presumably because it gives more weight to errors when the ground truth δ is small.

IV. Experiments

To evaluate the quality of the embeddings produced by different self-supervised learning methods, we observe how the learned representations transfer to several, potentially heterogeneous, downstream tasks. To this end we consider a model that consists of the sequence of a pre-trained encoder and a linear layer, which maps the embeddings to the logits corresponding to the class predictions of the downstream task. For each combination of pre-trained encoder and downstream task, we train the model in a supervised fashion using the labels available for the downstream task, either using the whole training dataset, or subsets containing 1000 training examples sampled at random from the original training set. In the latter case we do not explicitly constrain the sampling process to obtain exactly the original class label balance. We evaluate our results in two scenarios. In the first scenario, the encoder weights are frozen and only the weights of the linear layer are trained. In the second scenario the whole model is trained end-to-end, so that also the encoder weights can be fine-tuned.

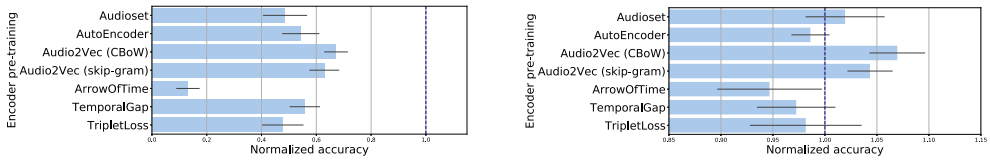
Encoder architecture: In our work we consistently use the same audio frontend, which processes input sequences sampled at 16 kHz, with a window size of 25 ms and a hop size equal to 10 ms to compute the short-time Fourier transform (STFT), and then computes $F = 64$ mel-spaced frequency bins in the range 60–7800 Hz. For the encoder $Enc()$, we use a convolutional neural network with 6 layers, each using 3×3 filters and respectively $[64, 128, 256, 256, 512, 512]$ channels. All activation functions are ReLUs and batch normalization is used in all convolutional layers. Max-pooling is used whenever we increase the number of channels in the next layer, to reduce the time-frequency dimensions by a factor of two. Finally, a global max-pooling layer produces a vector, which is further processed by a fully-connected layer to get the embeddings. We set $N = 96$ (corresponding to 975 ms) and $d = 128$.

Encoder pre-training: We use *AudioSet* [8] to pre-train different encoders, either with full or self-supervision. *AudioSet* contains excerpts of 10 seconds from the soundtracks of YouTube videos annotated with labels of 527 classes. In the case of self-supervised pre-training, we ignore the labels.

We consider encoders pre-trained with six different self-supervised models: `Audio2Vec`, in its two variants, CBoW and skip-gram, `TemporalGap`, `AutoEncoder`, `ArrowOfTime` and `TripletLoss`. The `ArrowOfTime` model is the audio equivalent of the video-based arrow-of-time task proposed in [22], where the task is to predict the temporal direction of the clip (forward vs. backward) The `TripletLoss` model was proposed in [3], in which positive/negative pairs are obtained by extracting a slice from, respectively, the same or a different original sample. The `AutoEncoder` model shares the same encoder and decoder architectures as `Audio2Vec`. For `Audio2Vec` we use $P = 2$ slices on each side of the target, and a gap of $G = 2$ temporal frames between consecutive slices. In our preliminary experiments we observed that the selection of the hyperparameter G is not particularly critical to the quality of the pre-trained representations and similar results are observed with larger values of G (e.g., 4 or 8 frames). The work in [30] shows that embeddings trained on ImageNet transfer to wide range a downstream visual tasks. Motivated by this observation, we include in our study an encoder pre-trained with full-supervision on *AudioSet* classes. All models are trained with stochastic gradient descent and Adam optimizer with default hyperparameters. The learning rate was set to 10^{-3} for `Audio2Vec`, `AutoEncoder`, while it was set to 10^{-4} for `TemporalGap`, `TripletLoss`, `ArrowOfTime` and the supervised model trained on *AudioSet*. We use a mini-batch size equal to 256 and we stop training after 3 million iterations.

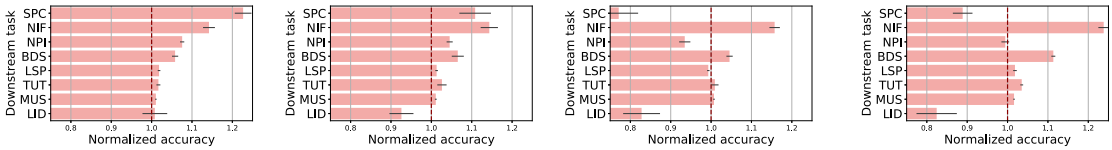
Downstream tasks: We consider different publicly available datasets to evaluate a variety of eight downstream tasks, covering both speech and non-speech related tasks. We use the *Speech Commands* dataset [31] (SPC) to evaluate keyword spotting on 35 distinct keywords. *LibriSpeech* [32] (LSP) contains audio books read by 251 different speakers. We use the 100 hours training set to evaluate a speaker identification task. The *Spoken Language Identification* dataset [33] (LID) contains samples that belong to three different languages: English, Spanish and German, while the *MUSAN* dataset [34] (MUS) distinguishes across three classes, namely music, speech and noise. The *NSynth* dataset [35] contains synthesized musical notes and we used this dataset for two different downstream tasks, i.e., distinguish 128 distinct pitch values (NPI) and classify instruments into 11 families (NIF). Finally, we use two datasets released in the context of the DCASE2018 Challenge, *Bird Audio Detection* [36] and *TUT Urban Acoustic Scenes 2018* [37], which contains labeled audio samples from 10 different urban environments. Note that we deliberately avoid exploiting during pre-training any of the datasets used for evaluating the downstream tasks. Since each dataset is characterized by samples having different durations, during both training and evaluation we extract equal-length slices uniformly at random from the original sample and assign the corresponding label to all of the extracted slices. We use input samples with a duration of $T = 975$ ms, so as to match the size of the temporal slices used when training the self-supervised tasks. The choice of the tasks used for the evaluation is consistent with the selected temporal granularity. When training for the downstream task, we set the learning rate to 10^{-4} and we stop after 100 k steps.

Metrics: For each combination of pre-trained encoder $e \in \mathcal{E}$ and downstream task $t \in \mathcal{T}$, we compute the accuracy on the eval set $a_{e,t,r}$. The index r denotes the fact that we repeat training of each configuration over four replicas. Since downstream tasks



(a) Frozen encoder (b) Fine-tuned encoder.

Fig. 2. Normalized accuracy (averaged over tasks and replicas) obtained using different methods to pre-train the encoder, and using 1 k examples for each downstream tasks. Note that a different scale is used for the x-axis in the two figures.



(a) `Audio2Vec` (CBoW). (b) `Audio2Vec` (skip-gram). (c) `TemporalGap`. (d) *AudioSet*.

Fig. 3. Normalized accuracy (averaged over replicas) for different downstream tasks, using 1 k examples per downstream task.

TABLE I
NORMALIZED ACCURACY (AVERAGED OVER TASKS AND REPLICAS) OBTAINED USING DIFFERENT ENCODER PRE-TRAINING METHODS

	Frozen		Fine-tuned	
	full	1k	full	1k
Audioset	0.40 ± 0.09	0.48 ± 0.09	0.98 ± 0.02	1.02 ± 0.04
AutoEncoder	0.47 ± 0.06	0.55 ± 0.07	0.99 ± 0.01	0.99 ± 0.02
A2V (CBoW)	0.58 ± 0.06	0.68 ± 0.05	1.00 ± 0.01	1.07 ± 0.03
A2V (SG)	0.53 ± 0.07	0.63 ± 0.06	1.00 ± 0.01	1.04 ± 0.02
ArrowOfTime	0.10 ± 0.03	0.12 ± 0.05	0.97 ± 0.02	0.95 ± 0.05
TemporalGap	0.48 ± 0.06	0.55 ± 0.06	0.97 ± 0.02	0.97 ± 0.04
TripletLoss	0.46 ± 0.07	0.46 ± 0.08	0.95 ± 0.05	0.98 ± 0.06

are characterized by a different number of classes and intrinsic level of difficulty, we normalize the accuracy as follows:

$$\tilde{a}_{e,t,r} = \frac{a_{e,t,r} - \bar{a}_{\texttt{Untrained},t}}{\bar{a}_{\texttt{Supervised},t} - \bar{a}_{\texttt{Untrained},t}} \tag{1}$$

where $\bar{a}_{\texttt{Supervised},t}$ is the average accuracy attained by a fully-supervised model in which both the encoder and the linear head are trained from scratch, while $\bar{a}_{\texttt{Untrained},t}$ is the average accuracy attained by an untrained model in which the encoder is randomly initialized (and frozen), while the linear head is trained from scratch. In our results, we report averages across replicas ($\bar{a}_{e,t}$) as well as across replicas and tasks ($\bar{a}_e$). We also experimented with the metrics proposed in [38], which averages logit-scaled accuracy values and we reached the same conclusions as using the normalized accuracy in (1).

Main results: Table I summarizes the main results of the letter, reporting the normalized accuracy averaged across replicas and downstream tasks. For each scenario (frozen vs. fine-tuned encoder), we consider training using either the full downstream task datasets or subsets of 1 k examples extracted at random. The reported intervals reflect the uncertainty of the results due to the selection of the downstream task. Specifically, we applied 50 bootstrap iterations, resampling with replacement the downstream tasks used to compute the average. Intuitively, smaller intervals imply that the results are less dependent on the specific choice of downstream tasks. First, we focus the attention on the results obtained when training on 1 k examples and keeping the encoder weights frozen (column 2 in Table I and Fig. 2(a)). In this scenario we can make a few important observations: i) training a simple linear head, it is only partially possible to bridge the gap with a fully-supervised model (max normalized accuracy equal to 0.68); ii) there is a significant variation in the level of normalized accuracy across different methods (between 0.10 and 0.68); iii) the `Audio2Vec` models produce representations that outperform, on average, the ones obtained by all other models, including a fully-supervised model trained on Audioset (0.68 vs. 0.40). Fig. 2(b) shows how the normalized accuracy substantially increases when applying fine-tuning (note the change of the scale of the x-axis). In this case all methods achieve a level of accuracy comparable with a fully-supervised model (normalized accuracy between 0.95 and 1.07). Still, the models based on `Audio2Vec` seem to outperform fully-supervised models by a significant margin. Indeed, Fig. 3(a) shows the per-task level of normalized accuracy for a model pre-trained using `Audio2Vec` (confidence intervals represent the variation across replicas). On all tasks, pre-training using `Audio2Vec` is better than a fully-supervised model trained from scratch. The gain is particularly remarkable for *Speech Commands*, *NSynth Pitch* and *NSynth Intrument Family*. Fig. 3(d) shows a similar breakdown for an encoder pre-trained with full supervision on *AudioSet*. In this case pre-training outperforms the baseline for some tasks, and underperforms for others. When a fully supervised model is trained having access to the whole training set (column 1 and 3 in Table I), pre-training using `Audio2Vec` still outperforms all other models, although the differences are smaller. In this case, it achieves the same level of normalized accuracy as a model trained from scratch, but it takes fewer iterations to converge.

V. CONCLUSION

We present a comprehensive evaluation of self-supervised learning models on a variety of downstream audio tasks. We evaluate novel methods, like `Audio2Vec` and `TemporalGap`, as well as previously proposed methods. Our results show that when the number of examples per class are limited to 1 k, fine-tuning the `Audio2Vec` representation outperforms fully-supervised models trained from scratch on each downstream task.

REFERENCES

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

[2] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *Proc. Int. Conf. Learn. Representations*, Mar. 2018.

[3] A. Jansen *et al.*, "Unsupervised learning of semantic audio representations," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Nov. 2018, pp. 126–130.

[4] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. Int. Conf. Learn. Representations*, Jan. 2013.

[5] Y.-A. Chung and J. Glass, "Speech2Vec: A sequence-to-sequence framework for learning word embeddings from speech," in *Proc. Interspeech*, Mar. 2018, pp. 811–815.

[6] Y.-A. Chung, W.-H. Weng, S. Tong, and J. Glass, "Unsupervised cross-modal alignment of speech and text embedding spaces," in *Proc. Neural Inf. Process. Syst.*, May 2018, pp. 7365–7375.

[7] B. Milde and C. Biemann, "Unspeech: Unsupervised speech context embeddings," in *Proc. Interspeech*, 2018, pp. 2693–2697.

[8] J. F. Gemmeke *et al.*, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2017, pp. 776–780.

[9] Y. Xu *et al.*, "Unsupervised feature learning based on deep models for environmental audio tagging," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 25, no. 6, pp. 1230–1241, Jun. 2017.

[10] M. Meyer, J. Beutel, and L. Thiele, "Unsupervised feature learning for audio analysis," in *Proc. Workshop Track*, 2017.

[11] Y.-A. Chung, C.-C. Wu, C.-H. Shen, H.-Y. Lee, and L.-S. Lee, "Audio Word2Vec: Unsupervised learning of audio segment representations using sequence-to-sequence autoencoder," Tech. Rep., Mar. 2016, *arXiv:1603.00982*.

[12] M. Cartwright, J. Cramer, J. Salamon, and J. P. Bello, "Tricycle: Audio representation learning from sensor network data using self-supervision," in *Proc. IEEE Workshop Appl. Signal Process. Audio Acoust.*, 2019, pp. 278–282.

[13] S. Pascual, M. Ravanelli, J. Serrà, A. Bonafonte, and Y. Bengio, "Learning problem-agnostic speech representations from multiple self-supervised tasks," Tech. Rep., 2019, *arXiv:1904.03416*.

[14] A. Haque, M. Guo, P. Verma, and L. Fei-Fei, "Audio-linguistic embeddings for spoken sentences," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2019, pp. 7355–7359.

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

[16] A. Dosovitskiy, P. Fischer, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with exemplar convolutional neural networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 9, pp. 1734–1747, Jun. 2016.

[17] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proc. Eur. Conf. Comput. Vision*, Mar. 2016, pp. 649–666.

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

[19] I. Misra, C. L. Zitnick, and M. Hebert, "Shuffle and learn: Unsupervised learning using temporal order verification," in *Proc. Eur. Conf. Comput. Vision*, Mar. 2016, pp. 527–544.

[20] B. Fernando, H. Bilen, E. Gavves, and S. Gould, "Self-supervised video representation learning with odd-one-out networks," in *Proc. Comput. Vision Pattern Recognit. Conf.*, Nov. 2017, pp. 3636–3645.

[21] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan, "Learning features by watching objects move," in *Proc. Comput. Vision Pattern Recognit. Conf.*, Dec. 2017, pp. 2701–2710.

[22] D. Wei, J. Lim, A. Zisserman, and W. T. Freeman, "Learning and using the arrow of time," in *Proc. Comput. Vision Pattern Recognit. Conf.*, 2018, pp. 8052–8060.

[23] Y.-A. Chung, W.-H. Weng, S. Tong, and J. Glass, "Unsupervised cross-modal alignment of speech and text embedding spaces," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 7354–7364.

[24] A. Owens, J. Wu, J. H. Mcdermott, W. T. Freeman, and A. Torralba, "Learning sight from sound: Ambient sound provides supervision for visual learning," *Int. J. Comput. Vision*, vol. 126, no. 10, pp. 1120–1137, 2018.

[25] A. Owens and A. A. Efros, "Audio-visual scene analysis with self-supervised multisensory features," in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 631–648.

[26] R. Gao, R. Feris, and K. Grauman, "Learning to separate object sounds by watching unlabeled video," in *Proc. Eur. Conf. Comput. Vision*, Apr. 2018, pp. 35–53.

[27] R. Arandjelović and A. Zisserman, "Objects that sound," in *Proc. Eur. Conf. Comput. Vision*, Dec. 2018, pp. 451–466.

[28] B. Korbar, D. Tran, and L. Torresani, "Cooperative learning of audio and video models from self-supervised synchronization," in *Proc. Neural Inf. Process. Syst.*, 2018, pp. 7774–7785.

[29] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen and learn more: Design choices for deep audio embeddings," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2019, pp. 3852–3856.

[30] X. Zhai *et al.*, "A large-scale study of representation learning with the visual task adaptation benchmark," Tech. Rep., Oct. 2019, *arXiv:1910.04867*.

[31] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," Tech. Rep., 2018, *arXiv:1804.03209*.

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

[33] T. Oponowicz, "Spoken language identification," 2018. [Online]. Available: https://www.kaggle.com/toponowicz/spoken-language-identification

[34] D. Snyder, G. Chen, and D. Povey, "MUSAN: A music, speech, and noise corpus," Tech. Rep., 2015, *arXiv:1510.08484*.

[35] J. Engel *et al.*, "Neural audio synthesis of musical notes with wavenet autoencoders," in *Proc. 34th Int. Conf. Mach. Learn. (ICML'17)*, vol. 70, Aug. 2017, pp. 1068–1077.

[36] D. Stowell, M. Wood, H. Pamuła, Y. Stylianou, and H. Glotin, "Automatic acoustic detection of birds through deep learning: The first bird audio detection challenge," *Method. Ecol. Evol.*, vol. 10, no. 3, pp. 368–380, Mar. 2019. [Online]. Available: https://doi.org/10.1111/2041-210X.13103

[37] A. Mesaros, T. Heittola, and T. Virtanen, "Detection and classification of acoustic scenes and events," in *Proc. Detection Classification Acoust. Scenes Event*, 2018, pp. 9–13.

[38] S. Kornblith, J. Shlens, and Q. V. Le, "Do better imagenet models transfer better?" in *Proc. Comput. Vision Pattern Recognit. Conf.*, Jun. 2019, pp. 2656–2666.