CONTRASTIVE SIAMESE NETWORK FOR SEMI-SUPERVISED SPEECH RECOGNITION

Soheil Khorram, Jaeyoung Kim*, Anshuman Tripathi, Han Lu, Qian Zhang, Hasim Sak*

{soheilkhorram, jaeykim, anshumant, luha, zhaqian, hasim}@google.com
Google Inc., USA

ABSTRACT

This paper introduces contrastive siamese (*c-siam*) network, an architecture for leveraging unlabeled acoustic data in speech recognition. *c-siam* is the first network that extracts high-level linguistic information from speech by matching outputs of two identical transformer encoders. It contains augmented and target branches which are trained by: (1) masking inputs and matching outputs with a contrastive loss, (2) incorporating a stop gradient operation on the target branch, (3) using an extra learnable transformation on the augmented branch, (4) introducing new temporal augment functions to prevent the shortcut learning problem. We use the Libri-light 60k unsupervised data and the LibriSpeech 100hrs/960hrs supervised data to compare c-siam and other best-performing systems. Our experiments show that c-siam provides 20% relative word error rate improvement over wav2vec baselines. A c-siam network with 450M parameters achieves competitive results compared to the state-of-the-art networks with 600M parameters.

***Index Terms*—** semi-supervised learning, siamese network, speech recognition, temporal augmentation

1. INTRODUCTION

Collecting large transcribed datasets is expensive and time consuming. Also, because of privacy concerns of the users, we always prefer to minimize transcribing datasets while keeping the quality of the systems unchanged. A common method to this end is to leverage unlabeled data using self/semi-supervised techniques. This work is an attempt to improve the performance of the existing self/semi-supervised speech recognition techniques.

Most approaches in learning effective representations without human supervision fall into one of three categories that predict (1) input-level, (2) intermediate-level and (3) output-level representations. It is easier to train the first category as there is no degenerate solutions for them. For example, in autoregressive predictive coding (APC) [1] the goal is to generate future frames by conditioning on past frames using a unidirectional network; or similarly DeCoAR [2, 3], TERA [4] and MOCKINGJAY [5] are systems that mask inputs and generate the mask regions with a bidirectional network. All these methods benefit from L1 reconstruction loss which is stable and easy to optimize. However, there is a major problem: In order to generate inputs, the network needs to learn details of the inputs that are not necessary for supervised tasks. As a result, it is not optimal to incorporate these techniques alongside supervised losses in semi-supervised frameworks.

The second category is based on predicting intermediate-level representations. Some methods in this category include CPC [6], wav2vec [7] and vq-wav2vec [8] which are similar to unidirectional APC [1], but instead of generating future inputs they predict future intermediate representations. Wav2vec 2.0 [9] and w2v-BERT [10] extend these unidirectional structures to bidirectional ones. They mask the intermediate representations and predict the masked regions. These techniques incorporate contrastive [11] or clustering [12] losses which are more consistent with supervised tasks, but there is still room for improvement; it is more effective to predict outputs of audio encoders [13].

Speech SimCLR [14, 13] uses both input and output level prediction losses. An augmentation module transforms inputs into two correlated views, then a transformer extracts output-level representations for both views. Speech SimCLR minimizes two losses: a contrastive loss that matches output representations, and a reconstruction loss that matches inputs and outputs. The reconstruction loss is employed to prevent the *shortcut learning problem* [15]: SimCLR can easily minimize the contrastive loss by ignoring the inputs and by using positional embeddings of the transformers. However, this reconstruction loss prevents SimCLR to be consistent with supervised tasks. In this paper, we propose to use temporal augmentation to deal with the shortcut learning problem. We introduce two temporal augmentation methods that are consistent with speech recognition and they can effectively reduce the shortcut learning problem.

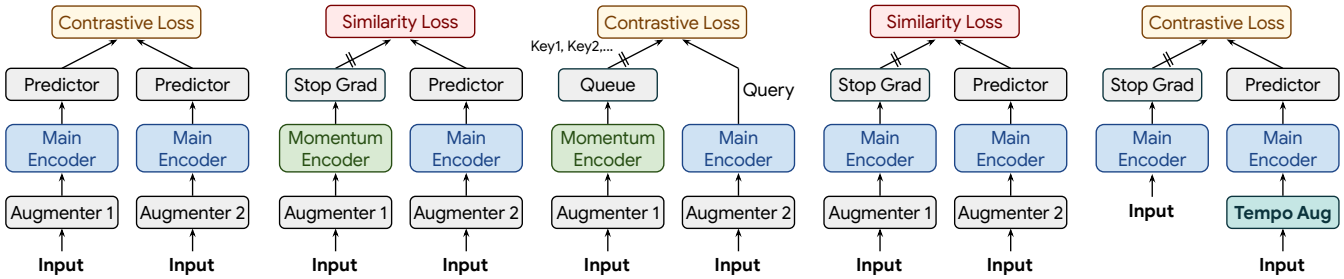
To evaluate the efficacy of the c-siam network, we developed semi-supervised speech recognition systems using the Libri-light 60k unsupervised data and the LibriSpeech 100hrs/960hrs supervised data. We compared c-siam with current state-of-the-art methods and the results show that c-siam provides 20% relative word error rate (WER) improvement when compared with the wav2vec-based self/semi-supervised systems.

2. RELATED WORK

Our work is most related to SimCLR [13], BYOL [16], MoCo [17] and SimSiam [18] methods. Figure 1 shows the architecture of these methods compared to c-siam. They all follow the core idea of matching high-level representations generated from two branches. A trivial solution of matching high-level representations is all outputs "*collapsing*" to a constant vector [18]. In this section, we discuss how these methods prevent the collapsing problem.

SimCLR learns representations by maximizing agreement between two different augmentations of the same data sample. It benefits from a contrastive loss to prevent the collapsing problem. SimCLR experiments showed that a learnable nonlinear mapping on top of the encoders can significantly improve the quality of the representations [13]. *BYOL* relies on similarity losses, but it does not collapse as it uses a momentum encoder. It contains online and target branches. The online branch is trained to predict the target outputs and the target branch is an exponential average of the online branch. BYOL also incorporates a learnable predictor on the online branch [13]. *MoCo* leverages both the contrastive loss and the momentum encoder. It considers contrastive learning as a dictionary

* These authors contributed equally to this work



(a) SimCLR [13] (b) BYOL [16] (c) MoCo [17] (d) SimSiam [18] (e) C-Siam

Fig. 1. Architecture of self/semi-supervised networks that are based on the idea of matching encoders' outputs.

look-up such that the main encoder extracts a query representation and the momentum encoder extracts a queue of keys. MoCo uses the contrastive loss to match queries with their corresponding keys [17]. *SimSiam* uses the same encoders in both branches and it matches the encoders' outputs using the cosine similarity function. SimSiam deals with the collapsing problem by applying a stop gradient operation and a learnable projection network [18].

Although these methods are effective for learning from unlabeled images, they are not suitable for transformer-based speech recognition. The main issue is the *"shortcut learning problem"* [15]: when we process sequences of inputs, transformers tend to minimize the training loss by just using the positional information and by ignoring the inputs. To overcome this problem, we propose to use temporal augmentation in c-siam. We modify the temporal characteristics of the inputs before processing them by the encoders. We also modify our contrastive loss to align the representations before defining positive and negative pairs.

3. PRELIMINARY EXPERIMENT

We first conduct an experiment to figure out if wav2vec-style training [9] is consistent with supervised speech recognition. The experiment consists of two steps: (1) we train an audio encoder using the wav2vec 2.0 scheme on the Libri-light 60k data; (2) we extract intermediate representations of the encoder and train a simple classifier on each representation to recognize frame-level phonemes. We expect to see constant improvement in phoneme recognition accuracy when we get closer to the output of the encoder.

Our audio encoder starts with two layers of strided convolutions, each of which downsamples its input by the factor of two. The encoder follows by a 20-layer transformer-xl with relative attention mechanism [19]. We train this encoder with the wav2vec 2.0 scheme explained in [20]. In order to perform phoneme recognition, we apply two dense and a softmax layers after each transformer layer. We train them with the cross-entropy (CE) loss that maximizes the likelihood of predicting frame-level phonemes. We follow [6] for preparing our frame-level phoneme recognition dataset.

Figure 2 (red, square points) shows phoneme recognition results achieved by wav2vec training. The accuracy increases from 65.9% at layer 1 to 91.2% at layer 17. In the last 3 layers, the accuracy drops to 70.2%. This performance reduction is a result of wav2vec trying to match the inputs of the audio encoder and it is not easy to predict phonemes from the inputs. To address this problem, we propose to match higher-level representations using siamese networks.

We also repeat the above experiment with our c-siam network (see Section 4 for the details of c-siam). We just use unsupervised part of the c-siam network in this experiment to be consistent with the wav2vec experiment. Figure 1 (blue, circular points) shows c-siam's phoneme recognition results. It does not show any performance drop at the end of the audio encoder. The constant improvement in phoneme accuracy confirms the consistency between the c-siam structure and the phoneme recognition task, and it motivates us to use c-siam alongside supervised speech recognition losses.

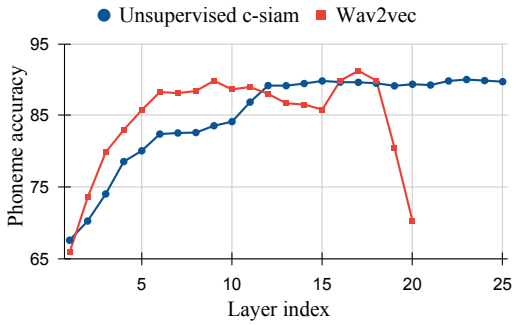


Fig. 2. Frame-level phoneme recognition accuracy using two dense layers for both wav2vec 2.0 (red, square points) and unsupervised c-siam (blue, circular points).

4. CONTRASTIVE SIAMESE NETWORK

An illustration of the c-siam network is shown in Figure 3. The network contains supervised and unsupervised parts. Both parts share the same audio encoder and they are trained together using the same Adam optimizer and the same learning rate function.

4.1. Supervised Network

Supervised network is consistent with the RNN-T based transformer transducer structure introduced in [21]. In this structure, likelihood of labels given input features are factorized through three modules: an audio encoder, a label encoder and a logit function. *Audio encoder* is a stack of strided convolution layers followed by transformers. Two strided convolution layers downsample log-mel features by the factor of four. Then, multiple layers of transformer-xl [19] extract our acoustic embeddings. *Label encoder* is a streaming transformer-xl that does not attend to the future labels. *Logit function* takes both acoustic and label embeddings as inputs and generates logit embeddings using this module:

$$r = \text{Linear}(\text{Tanh}(\text{Linear}(a) + \text{Linear}(l))), \qquad (1)$$

where a, l and r are acoustic, label and logit embeddings respectively. We pass the logits to a softmax and calculate label probabilities which are used in RNN-T's forward/backward algorithm [22].

4.2. Unsupervised Network

Unsupervised network contains two branches: augmented and target. *Augmented branch* takes log-mel features and applies temporal

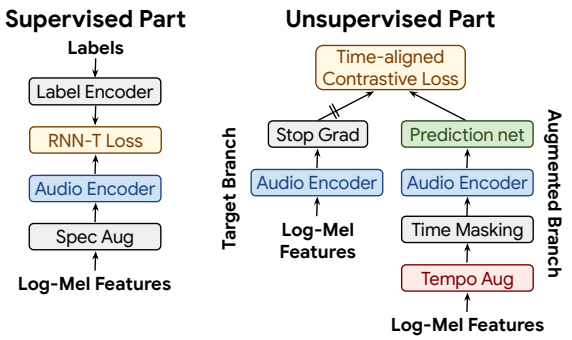


Fig. 3. Supervised and unsupervised parts of the c-siam network.

augmentation (TempoAug), time masking, audio encoder and prediction network to predict the outputs of the target branch. The *target branch* extracts clean outputs by passing log-mel features to the audio encoder. It is designed to just generate the clean outputs and we do not train parameters through that. To do so, we apply a stop gradient operation at the end of the target branch.

Stop gradient and prediction network – These components are introduced in SimSiam architecture [18] to improve convergence properties of siamese networks. By using these components, target network generates expected outputs based on the knowledge learned up to the current state of training, and the augmented branch tries to match these expected outputs [18]. We use 5 layers of transformer-xl as the prediction network in our experiments.

Time-aligned contrastive loss – In order to match target and augmented outputs, we mask the features of the augmented branch and we minimize a contrastive loss over the masked regions. Masking is done by simply setting continuous regions of features to zero, and contrastive loss is a softmax-based negative log-likelihood function defined over cosine similarities. Assume a_t is an output vector of the augmented branch, $q_{t'}$ is a positive target vector that has to match a_t, and Q is a set of negative target vectors randomly selected from the masked regions of the same utterance. Contrastive loss at time t is:

$$\mathcal{L}_t = -\log \frac{\exp(\text{sim}(a_t, q_{t'})/\tau)}{\sum_{q \in Q \cup \{q_{t'}\}} \exp(\text{sim}(a_t, q)/\tau)}, \quad (2)$$

where sim is the cosine similarity and τ is the temperature parameter. In c-siam, positive pairs may come from different time indexes (t and t' in Eq. 2). It is because of the temporal augment (TempoAug) module used at the beginning of the augmented branch.

TempoAug – modifies temporal characteristics of the inputs by shifting log-mel features in time. The goal is to prevent *"shortcut learning problem"* [15] caused by transformers' positional embeddings. If we do not modify time characteristics of the branches, network can ignore its input and minimize the contrastive loss just based on the positional embeddings. We introduce two TempoAug techniques in this work: uniform and non-uniform.

Uniform TempoAug – uniformly compresses or stretches time-domain audio signals with a randomly drawn tempo ratio. It uses a waveform similarity based overlap and add (WSOLA) [23] method to linearly change tempo of speech (i.e., changing t to αt) without modifying its pitch contours. In our experiments, the ratio α is randomly drawn from a uniform distribution for each utterance. Since each utterance takes different tempo ratio, our audio encoder cannot easily model it and therefore positional counting can be avoided.

Non-uniform TempoAug – non-uniformly compresses or stretches log-mel trajectories such that it modifies temporal characteristics, but it does not negatively affect speech recognition. We do this by applying a time-warping function to feature trajectories. Assume $x(t)$ denotes speech features at time t, we transform it to $x(w(t))$, where $w(t)$ denotes our time-warping function. We also apply the same time-warping function to the outputs of the target branch in our time-aligned contrastive loss. This warping function must satisfy three constraints to preserve the nature of log-mel trajectories: (1) Monotonicity – $w(t)$ must be monotonically increasing, otherwise it will not preserve the order of the input samples. (2) Smoothness – sudden changes in the warping function distort the overall shape of the feature trajectories. (3) Boundary conditions – the warping function must start from time 0, $w(0) = 0$, and it must end to the last frame, $w(T-1) = T-1$, where T is the number of input frames. Boundary condition ensures that the warping function does not eliminate any part of the inputs. We propose the following time-warping function that satisfies all these constraints:

$$w(t) = t + \sum_{r=1}^{R} a_r \sin(\frac{\pi r t}{T-1}), \quad t \in \{0, 1, ..., T-1\}. \quad (3)$$

In this equation, R is the order of warping function and a_r specifies the amplitude of the r-th sin component. These parameters can control smoothness and monotonicity of the warping function. In our experiments, we use five sin components ($R = 5$) and we randomly select the values of a_r for each utterance from the normal distribution with mean 0 and std 0.2. We empirically found that these parameters lead to our best performing systems.

After generating a time-warping function, we must apply it to the input features, i.e., we must calculate $x(w(t))$. We use a linear interpolation technique to do this:

$$x(w) \approx (w - \lfloor w \rfloor) x(\lceil w \rceil) + (\lceil w \rceil - w) x(\lfloor w \rfloor), \quad (4)$$

where $\lfloor w \rfloor$ and $\lceil w \rceil$ are floor and ceil values of w. Our experiments showed that applying time-warping function introduced in Eq. 3 using the simple linear interpolation expressed by Eq. 4 can significantly reduce the shortcut learning problem.

5. EXPERIMENTS

Data – We evaluate the c-siam network using the LibriSpeech dataset [24]. It is a read speech corpus collected from LibriVox audio books. We use two standard sets, 100 hrs and 960 hrs sets, of LibriSpeech as our supervised datasets. Our unsupervised dataset is the Libri-light corpus which is also derived from the LibriVox audio books. Libri-light contains 60k hours of 16kHz audio. Our language model (LM) training dataset is a combination of LibriSpeech text transcripts with 10M word tokens and an additional text-only dataset with 800M word tokens. We extract 80-dimensional 10ms log-mel filter bank coefficients as our acoustic features [20]. Also, we use SpecAugment [25] for the supervised part of the network. SpecAugment applies two frequency masks with the size of 27 and 10 time masks with the maximum ratio of 0.05.

Models – We train two types of models, a small model with 100M parameters and a large model with 450M parameters. The small model contains 2 conv layers with the kernel size of 3 and 512 channels, 20 layers of transformers, 8 attention heads, 512 dimensional outer embeddings, 2048 dimensional inner embeddings, 48 dimensional Value vectors and 16 dimensional Query/Key vectors. The large model contains 24 transformer layers, 16 heads, 1024 dimensional outer embeddings, 4096 dimensional inner embeddings and other parameters are the same as the small model. We train our c-siam models on 16x16 TPUs with a per-core batch size of 2 (experiments on 100hrs) and 4 (experiments on 960hrs). This results in a global batch size of 1024 and 2048. Both supervised and unsupervised utterances share the same batch size in our c-siam network.

Method	Size (B)	Clean Dev.	Clean Test	Other Dev.	Other Test
Baseline Models					
random init.	.45	1.9	2.0	4.2	4.5
wav2vec init.	.45	1.9	1.9	4.1	4.0
wav2vec cotrain	.45	1.7	1.8	3.8	3.5
Proposed Models					
uniform c-siam	.45	1.6	1.7	2.9	**2.8**
uniform c-siam + LM		1.5	1.6	2.7	**2.8**
non-uniform c-siam	.45	1.5	1.6	2.7	2.9
non-uniform c-siam + LM		**1.4**	1.6	**2.6**	**2.8**
SOTA Models					
NST-conformer [20]	.1	1.6	1.7	3.3	3.5
w2v-CTC [9]	.3	2.1	2.2	4.5	4.5
w2v-conformer [20]	.6	1.7	1.7	3.5	3.5
w2v-BERT [10]	.6	1.5	**1.5**	2.9	2.9
w2v-BERT + LM [10]		**1.4**	**1.5**	2.8	**2.8**

Table 1. WER results on LibriSpeech 960 hrs data. The size of the networks are reported in billion parameters. Also, rows with "+ LM" report the results obtained after applying a language model.

Training parameters – We train both models with the Adam optimizer [26]. Learning rate is ramped up linearly to 2e-3 during first 10k steps, and then decays exponentially to 2.5e-6 at 200k steps. We also apply gradient scaling to limit the norm of the gradient vectors to 60. To reduce the over-training problem, we set dropout factor, variational noise power and L2 loss weight to 0.3, 0.02 and 1.5e-4 for our small model and we set them to 0.1, 0.03 and 3e-5 for our large model. We apply the variational noise after 4k iterations. To do masking we sample initial time steps of the masks randomly with probability 0.016 and we mask the subsequent 28 steps.

5.1. Experiment Results

Table 1 and 2 report the WER results when LibriSpeech 960 hrs and 100 hrs are used as the supervised data. The WER numbers are calculated on the standard LibriSpeech development and test sets. Both tables are divided into three sections: baseline, proposed and state-of-the-art (SOTA). We trained our transformer-xl network using the existing self/semi-supervised methods and we summarize the results in the baseline section. In this section, *"Random init."* refers to the base system that does not leverage any unlabeled data, *"wav2vec init."* is considered for the pretraining/finetuning experiment with wav2vec 2.0, *"Wav2vec cotrain"* is a semi-supervised model in which both RNN-T and wav2vec losses are trained together in a similar way that we train our c-siam network. In the proposed section, we report the WER numbers obtained for the c-siam network with both uniform and non-uniform TempoAug approaches. We also report our LM fusion results. We follow transformer-based shallow LM fusion explained in [21] to obtain these numbers. The last section, SOTA section, contains WER of the best-performing systems that we could find from other papers. There are many differences between these systems and our c-siam system, for example "NST-conformer", "w2v-conformer" and "w2v-BERT" use the conformer encoders [27] instead of the transformer-xl and "w2v-CTC" is trained with CTC loss instead of the RNN-T loss. However, all these systems are similar in one aspect that they are trained with the same supervised and unsupervised datasets.

The results reported in the tables show that (1) c-siam significantly improves all the baseline models on the "other" set. The best c-siam network provides 15% (on 100 hrs) and 20% (on 960 hrs) relative WER improvement compared to the best baseline system (i.e., "wav2vec cotrain"). (2) The improvement achieved by c-siam on the "clean" set (11% for 960 hrs and 3% for 100 hrs) is less than the "other" set. This is also True when we compare "wav2vec cotrain" and "wav2vec init.". It is due to the nature of our unsupervised data that is more similar to the "other" set; therefore, all cotrain systems are more effective for the "other" set. (3) c-siam with 450M parameters achieves competitive results compared to the best performing models in the literature with 600M parameters. This is an important achievement for practical applications in which there is a limitation in using large models. (4) Uniform and non-uniform TempoAug perform similarly in 960 hrs experiments, but non-uniform TempoAug is better in 100 hrs. It shows more augmentation provided by the non-uniform TempoAug is helpful for smaller datasets. LibriSpeech is divided into two separate evaluation sets: "Clean" and 'Other'. The "Other" set is noisier and therefore it contains more challenging utterances. C-Siam provides more improvement in the "Other" set, because C-Siam's target embeddings are close to the output labels and they are less impacted by the input noise.

Method	Size (B)	Clean Dev.	Clean Test	Other Dev.	Other Test
Baseline Models					
random init.	.1	5.3	5.5	16.	15.8
wav2vec init.	.1	3.1	3.0	7.3	6.8
wav2vec cotrain	.1	–	3.2	–	6.2
Proposed Models					
uniform c-siam (S)	.1	3.0	2.9	5.3	5.3
non-uniform c-siam (S)	.1	2.9	3.0	5.2	5.3
uniform c-siam (L)	.45	3.0	2.9	4.6	4.8
non-uniform c-siam (L)	.45	2.7	2.6	**4.3**	**4.6**
SOTA Models					
w2v-CTC [9]	.3	3.3	3.1	6.5	6.3
w2v-conformer [20]	.6	2.5	2.6	4.7	4.9
w2v-BERT [10]	.6	**2.4**	**2.5**	4.4	**4.6**

Table 2. WER results for LibriSpeech 100 hrs data. "(S)" and "(L)" are used to refer to small and large c-siam networks.

There is an important limitation with the c-siam architecture. In each training step, c-siam needs to run the forward pass of the audio encoder three times, which leads to a slow training loop and also high memory usage. Training our large network with c-siam takes 5 days, while with "wav2vec cotrain" it takes 3.5 days. Due to this problem, it is challenging to train c-siam for very large networks.

6. CONCLUSION

We introduce contrastive siamese (c-siam) network, a new architecture for training semi-supervised speech recognition systems. c-siam simultaneously trains a supervised RNN-T model and an unsupervised siamese network. The siamese network contains target and augmented branches. It extracts clean and augmented representations from target and augmented branches; it then modifies the augmented representations to be correlated with the clean ones using a contrastive loss. We train c-siam on 60k hours Libri-Light unlabeled data and LibriSpeech labeled data. We show that c-siam either outperforms or matches state-of-the-art systems. For future work, we plan to explore different techniques to train large networks (with 600M parameters or more) using c-siam.

7. REFERENCES

[1] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass, "An unsupervised autoregressive model for speech representation learning," *arXiv preprint arXiv:1904.03240*, 2019.

[2] Shaoshi Ling, Yuzong Liu, Julian Salazar, and Katrin Kirchhoff, "Deep contextualized acoustic representations for semi-supervised speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6429–6433.

[3] Shaoshi Ling and Yuzong Liu, "Decoar 2.0: Deep contextualized acoustic representations with vector quantization," *arXiv preprint arXiv:2012.06659*, 2020.

[4] Andy T Liu, Shang-Wen Li, and Hung-yi Lee, "Tera: Self-supervised learning of transformer encoder representation for speech," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 2351–2366, 2021.

[5] Andy T Liu, Shu-wen Yang, Po-Han Chi, Po-chun Hsu, and Hung-yi Lee, "Mockingjay: Unsupervised speech representation learning with deep bidirectional transformer encoders," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6419–6423.

[6] Aaron van den Oord, Yazhe Li, and Oriol Vinyals, "Representation learning with contrastive predictive coding," *arXiv preprint arXiv:1807.03748*, 2018.

[7] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, "wav2vec: Unsupervised pre-training for speech recognition," *arXiv preprint arXiv:1904.05862*, 2019.

[8] Alexei Baevski, Steffen Schneider, and Michael Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," *arXiv preprint arXiv:1910.05453*, 2019.

[9] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *arXiv preprint arXiv:2006.11477*, 2020.

[10] Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu, "W2v-bert: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," *arXiv preprint arXiv:2108.06209*, 2021.

[11] Michael Gutmann and Aapo Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2010, pp. 297–304.

[12] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze, "Deep clustering for unsupervised learning of visual features," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 132–149.

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[14] Dongwei Jiang, Wubo Li, Miao Cao, Ruixiong Zhang, Wei Zou, Kun Han, and Xiangang Li, "Speech simclr: Combining contrastive and reconstruction objective for self-supervised speech representation learning," *arXiv preprint arXiv:2010.13991*, 2020.

[15] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann, "Shortcut learning in deep neural networks," *Nature Machine Intelligence*, vol. 2, no. 11, pp. 665–673, 2020.

[16] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al., "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[17] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[18] Xinlei Chen and Kaiming He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15750–15758.

[19] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov, "Transformer-xl: Attentive language models beyond a fixed-length context," *arXiv preprint arXiv:1901.02860*, 2019.

[20] Yu Zhang, James Qin, Daniel S Park, Wei Han, Chung-Cheng Chiu, Ruoming Pang, Quoc V Le, and Yonghui Wu, "Pushing the limits of semi-supervised learning for automatic speech recognition," *arXiv preprint arXiv:2010.10504*, 2020.

[21] Qian Zhang, Han Lu, Hasim Sak, Anshuman Tripathi, Erik McDermott, Stephen Koo, and Shankar Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7829–7833.

[22] Alex Graves, "Sequence transduction with recurrent neural networks," *arXiv preprint arXiv:1211.3711*, 2012.

[23] Werner Verhelst and Marc Roelands, "An overlap-add technique based on waveform similarity (wsola) for high quality time-scale modification of speech," in *1993 IEEE International Conference on Acoustics, Speech, and Signal Processing*. IEEE, 1993, vol. 2, pp. 554–557.

[24] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[25] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le, "Specaugment: A simple data augmentation method for automatic speech recognition," *arXiv preprint arXiv:1904.08779*, 2019.

[26] Diederik P Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[27] Anmol Gulati, James Qin, Chung-Cheng Chiu, Niki Parmar, Yu Zhang, Jiahui Yu, Wei Han, Shibo Wang, Zhengdong Zhang, Yonghui Wu, et al., "Conformer: Convolution-augmented transformer for speech recognition," *arXiv preprint arXiv:2005.08100*, 2020.