Improved Masking Strategies for Self-Supervised Speech Representation Learning

Anuj Jitendra Diwan
Department of Computer Science
University of Texas at Austin
Austin, TX
anuj.diwan@utexas.edu

Abstract

Models trained using self-supervised speech representation learning algorithms have achieved state-of-the-art performance on competitive downstream task benchmarks. Inspired by advances in masking strategies for text-based models, we propose a novel linguistically-guided masking strategy called **RandomPhoneme Masking** for Masked Language Modelling-based speech models like HuBERT. We propose two variations of this strategy and conduct pretraining experiments using our proposed strategy. We demonstrate that our proposed strategy's obtains better metrics when using high-quality supervision-derived phoneme boundaries on the downstream Automatic Speech Recognition, Phoneme Recognition and Keyword Spotting tasks in the SUPERB benchmark of speech tasks, while observing a small gap in performance when using unsupervised boundaries.

1 Introduction

Pretraining large neural network models using self-supervised representation learning has shown great success for learning powerful latent representations in fields like Natural Language Processing (Devlin et al. [2019b], Brown et al. [2020]), Computer Vision (Grill et al. [2020], Zbontar et al. [2021]) and Speech (Baevski et al. [2020], Hsu et al. [2021]). Typically, neural network models are *pretrained* i.e. trained using a unsupervised algorithm on unlabelled datasets (natural or human-generated data that can be collected relatively easily and does not require additional human annotation or supervision). Then, the representations generated by these models are used to *finetune* the model on labelled data for a given downstream task. The eventual goal of unsupervised learning is generating high-quality representations that can be used to achieve high performance on any downstream task. For example, in speech, such representations are expected to be able to represent the spoken content (including its semantic meaning, lexical aspects, etc.) as well as non-lexical aspects like emotion, prosody, speaker characteristics, etc. Indeed, self-supervised pretraining algorithms have achieved state-of-the-art performance on several downstream tasks such as the SuperGLUE (Wang et al. [2020a]) benchmark for NLP and the SUPERB (wen Yang et al. [2021]) benchmark for Speech.

Self-supervised representation learning is a form of unsupervised learning where the data provides some form of implicit labelling or supervision. For example, Contrastive Learning (surveyed by Le-Khac et al. [2020]) involves training the model to learn to contrast similar from dissimilar data using its own representations; this forces the model to encode useful information. Masked (Language) Modelling (Devlin et al. [2019b]), on the other hand, involves masking (hiding) part of the input data and tasking the model with predicted the masked part from the remaining part of data. For example, one can mask part of a sentence (`I go to the [MASK] to buy eggs`) and require the model to predict the masked input using the remaining context. While these two approaches are in no way representative of the vast number of self-supervised algorithms proposed over the past few decades,

Preprint. Under review.

these are the two approaches used in recent state-of-the-art self-supervised speech representation algorithms: wav2vec2.0 (Baevski et al. [2020]) and HuBERT (Hsu et al. [2021]).

In Masked Language Modelling, the *masking strategy* is an important modelling choice that can affect the quality of learned representations. The masking strategy is the algorithm used to choose which parts of the input data instance to mask. BERT (Devlin et al. [2019b]) was one of the first models to propose Masked Language Modelling and used a fairly simple masking strategy that involved picking a random proportion $p = 0.15$ of all sentence tokens to mask. In subsequent years, many improved masking strategies have been proposed for Masked Language Modelling in pretrained text models.

We discuss these masking strategies and other related work in Section 2. However, the current state-of-the-art approaches for speech (Chen et al. [2021b], Hsu et al. [2021]) still use a fairly simple random strategy similar to the one used by the BERT model. We discuss the HuBERT model used by both these approaches in Section 3. In Section 4, we describe the masking strategy used by HuBERT, qualitatively discuss its potential pitfalls, and propose a novel masking strategy, **RandPhoneme Masking**, that is based on using phoneme boundaries (discovered in an unsupervised manner). In Section 5, we describe our downstream evaluation methodology. In Sections 6 and 7, we describe our experimental setup and our experiment results as well as a qualitative discussion of our results. Finally, in Section 8, we conclude.

2 Related Work

In the past, many improved masking strategies have been proposed for Masked Language Modelling for text models. While Random-Token Masking, originally proposed in the BERT paper (Devlin et al. [2019b]) picks a random 15% of subword tokens to mask, Whole-Word Masking (Devlin et al. [2019a]) ensures that entire words are masked, picking these words at random (such that the total number of masked subword tokens is still around 15%). Random-Span Masking (Joshi et al. [2020]) masks an entire span of multiple consecutive words, sampling the span length and span location at random (again, ensuring the masked subword tokens are 15%). These approaches extend the 'masking unit' from subword tokens to entire words, a linguistically meaningful unit. Our approach is inspired by this type of extension; we extend the 'masking unit' from frame spans to entire phonemes, an acoustically and linguistically meaningful unit. Approaches like Knowledge Masking (Sun et al. [2019]) and Salient Span Masking (Guu et al. [2020]) use external tools like parsers (like we use external phoneme segmentation models) in order to identify meaningful units of text to mask; PMI-Masking (Levine et al. [2020]) identifies meaningful units of text automatically using a Pointwise Mutual Information-based metric.

In the speech field, semantic masking has been proposed by Wang et al. [2020b] for supervised speech recognition, where they mask out random words in the input speech signal as a form of input signal corruption, encouraging the decoder to use contextual language-model-like information to predict these words rather than purely acoustic information. Chen et al. [2021a] applies SpanBERT-like masking for Speech-to-Text Translation. This approach, while similar to the original HuBERT masking strategy, differs in the span lengths being a random variable rather than fixed.

3 Pretraining the HuBERT Model

Figure 1 depicts the HuBERT architecture. We shall describe the architecture in brief in order to describe and motivate our masking strategy in Section 4.

We are provided a dataset of speech audio examples $x \in X$, where each x is a single speech audio example. Since speech is a continuous waveform, in order to use it for computation, it is sampled at a large sampling rate (16000 Hz in all our datasets) and stored as a sequence of amplitude values. Given such a speech example x, one first passes it through a CNN audio encoder with a 320x downsampling rate. Thus, the CNN encoder generates a frame feature sequence at a frame rate of 20 ms.

First, a discretized version of the frame feature sequence is generated by passing the audio sequence into an acoustic unit discovery system. To do this, our acoustic unit discovery system is trained by passing the training set audio sequences into the pretrained HuBERT Base model and generating a HuBERT representation for each frame. Then, these frame HuBERT representations are used for k-means clustering. For any given audio sequence, it is passed into the pretrained HuBERT Base

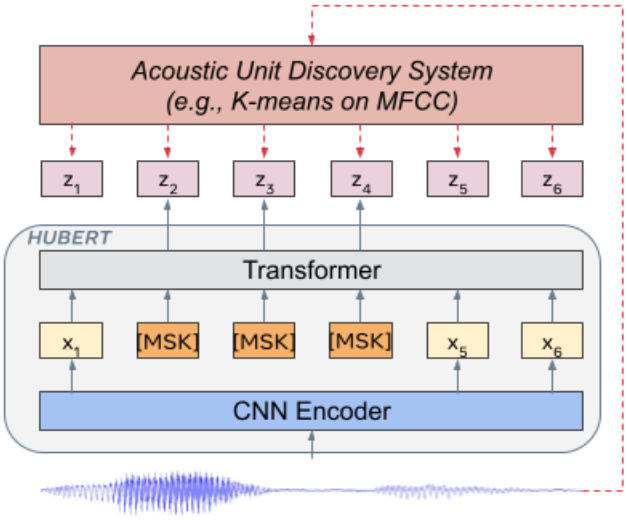

Figure 1: The HuBERT architecture. Diagram has been taken from Hsu et al. [2021]

model and each frame is classified into a k-means cluster using the trained k-means model. The cluster index is then used as the discretized representation of the frame.

Second, using a masking strategy, part of the input frame sequence is replaced with a `[MASK]` token. We explore different masking strategies in this project. This modified frame feature sequence is then passed into the Transformer. For each masked frame, a probability distribution is computed over the cluster indices. Cross-entropy loss is applied to each masked frame and summed up to encourage the model to predict the correct cluster indices for the masked frames. For more details regarding how the probability distribution and cross-entropy loss is computed, please refer to the original HuBERT paper (Hsu et al. [2021]).

During inference, the whole frame feature sequence is passed unmasked into the Transformer encoder. The output frame representations can be used for downstream task finetuning.

In order to complete pretraining in a reasonable amount of time, we initialize the CNN encoder and the Transformer of all our models using a pretrained HuBERT Base model and run continued pretraining from this model using different masking strategies.

4 RandomPhoneme Masking

First, we discuss the masking strategy used by the original HuBERT paper. In this work, we refer to it as **RandomFrameSpan Masking**. This strategy masks out *spans* of consecutive frames. Given an input frame sequence, $p\%$ of the timesteps are randomly chosen to be start indices of masking spans. Then, a fixed span of l steps is masked starting from each start index, noting that overlapping spans are allowed. $p = 8\%$ and $l = 10$ is chosen in the paper, resulting in about 56% of all frames being masked.

We can observe that masking out entire spans, rather than choosing frames to mask at random, is important due to the highly redundant nature of the audio signal. Masking entire spans means that frames in the interior of the span have their local context masked out as well, forcing the model to use some sort of global context in order to make correct predictions. However, we argue that the random choice of the *location* of the masking spans may be suboptimal. If, say, the left part of the masked span has a high correlation to the unmasked frames to the left of the masked span (perhaps because they constitute a single phoneme), it would be possible to predict the left part of the masked span by simply using these local unmasked frames, which will not allow the model to learn global context. By taking inspiration from approaches like Knowledge Masking (Sun et al. [2019]) and PMI Masking (Levine et al. [2020]), we argue that masking *semantically meaningful* spans that have high internal correlations may be useful. In this work, we consider the span of frames that constitute a

single *phoneme* to be a meaningful span that should be harder to predict than a randomly chosen span.

Therefore, we propose our masking strategy, *RandomPhoneme Masking*. Given an input frame sequence, we first run a phoneme segmentation algorithm (described in Section 4.1) to retrieve a predicted list of phoneme boundaries in the input audio i.e. the start and end frame index of each phoneme in the audio. We propose two types of masking strategies:

- **RandomPhoneme Vanilla**: In this strategy, we choose a random $q\%$ of phonemes and then fully mask them out. To ensure that approximately the same number of frames as in the RandomFrameSpan Masking are masked, we choose $q = 56\%$.
- **RandomPhoneme Iterative**: In this strategy, we choose a proportion $q\%$ of total frames that we intend to mask and a phoneme span length m. Suppose there are n phonemes; then, we initialize a set S of phonemes from phoneme 0 to phoneme $n - m$. In each iteration, we randomly sample a phoneme (say it is the i^{th} phoneme) from set S and mask out an entire span of m phonemes from phoneme i to phoneme $i + m - 1$. Then, we remove phoneme i from set S (we never resample phonemes). To ensure that approximately the same number of frames as in the RandomFrameSpan Masking are masked, we choose $q = 56\%$.

4.1 Phoneme Segmentation Algorithms

Since we are interested in self-supervised learning, our phoneme segmentation algorithm cannot require any supervision (in the form of phoneme labels, text, ground truth segmentation, etc.) during training nor during inference. We use a self-supervised segmentation algorithm proposed by Kreuk et al. [2020], referring to it as the **Unsupervised** approach. In this approach, inspired by Noise Contrastive Estimation and Contrastive Predictive Coding (van den Oord et al. [2019]), the model is optimized to detect spectral changes in the signal; abrupt frame representation changes are indicative of phoneme boundaries. The model is trained to learn a frame-level representation that is capable of distinguishing pairs of adjacent frames from pairs of non-adjacent frames through contrastive learning. At inference time, the adjacent frame pairs with the least similar representations are identified as phoneme boundaries.

We also run a phoneme segmentation algorithm that we refer to as a **Supervised** approach that makes use of ground truth text in order to generate phoneme boundaries, which are expected to be of higher quality than the Unsupervised approach. This algorithm is intended to be used as an 'oracle' experiment that shows the best that our masking strategy can be expected to do, since the phoneme segmentation is near-perfect. Our pretraining dataset is LibriSpeech (Panayotov et al. [2015]) which also comes with labelled text (note that we do not use the labels for the self-supervised pretraining itself); thus, we can test this approach. First, we train a hybrid HMM-TDNN based ASR system on the labelled training data using the standard Librispeech recipe from Kaldi (Povey et al. [2011]). Then, using a phonetic dictionary to convert the ground truth word sequence to a ground truth phoneme sequence, we run forced Viterbi alignment using the trained ASR model to align the speech signal to the phoneme sequence. As a result, we obtain the start and end timestamps for each phoneme, resulting in a phoneme segmentation.

5 Evaluation

In order to evaluate the quality of the representations generated by each approach, we run downstream task evaluation using the SUPERB (wen Yang et al. [2021]) benchmark. We now describe the evaluation methodology for a given downstream task. The architecture of the provided pretrained model is first modified in order to support predictions for the downstream task. To do this, one representation is obtained from every layer of the model and these layer representations are all summed together with layer-specific weights in order to obtain the final model representation. That is, given an input utterance x passed through a model M with l layers $m_1, \ldots, m_l$, the output at each layer gives rise to l layer representations $m_1(x), \ldots, m_l(x)$. Assuming that each layer representation has the same dimension d, we can generate a final representation

$$M(x) = \sum_{i=1}^{l} w_i m_i(x)$$

where w_i are layer-specific weights. Then, a lightweight task-specific head is placed on top of the final representation and the output of the head is the desired task-specific output. Finally, the original model weights are frozen and this modified model is trained on the task training set i.e. only the layer weights and the task-specific head are trained. Since only a small number of weights are being trained for each downstream task, a pretrained model that performs well on each task, despite such a restriction, must be sufficiently universal and generalizable, which is what we are interested in testing.

In this project, we focus on three of the many tasks provided by the benchmark: a) **Automatic Speech Recognition (ASR)**, which is a task where one must identify the textual contents of a given speech signal, outputting a sequence of characters. It uses the Librispeech `train-clean-100/dev-clean/test-clean` datasets for train/development/testing. The downstream task head is a vanilla 2-layer 1024-unit BLSTM optimized by CTC loss at the character level. The evaluation metric is Word Error Rate (WER); lower it is, the better. b) **Phoneme Recognition (PR)**, which is a task where one must identify the phonemes in a given speech signal, outputting a sequence of phonemes. It uses the Librispeech `train-clean-100/dev-clean/test-clean` datasets, converted into phoneme sequences using the official Librispeech phoneme lexicon `g2p-model-5`, for train/development/testing. The downstream task head is a shared linear layer applied on each frame; the output is optimized using CTC loss. The evaluation metric is Phoneme Error Rate (PER); lower it is, the better. c) **Keyword Spotting (KS)**, which is a task where one must identify preregistered keywords in a speech signal and classify the signal into this keyword class. It uses the Speech Commands v1.0 dataset (Warden [2017]) for training and testing. The downstream task head is a mean-pooling layer followed by a linear layer; the output is optimized using cross-entropy loss. The evaluation metric is Accuracy (Acc); higher it is, the better.

6 Implementation Details

6.1 Phoneme Segmentation

Dataset Details. In order to train the HMM-TDNN model neeeded for the **Supervised** approach, we use the LibriSpeech (Panayotov et al. [2015]) dataset. This dataset consists of 960 hours of training data (combining the three training splits `train-clean-100, train-clean-360, train-other-500`) and 10 hours of development set data (combining the dev splits `dev-clean` and `dev-other`). It also contains the phoneme lexicons needed for training the ASR model and running the forced Viterbi alignment. For the **Unsupervised** approach, we use the already-trained model released by the authors of Kreuk et al. [2020] that is trained on the Buckeye dataset (Pitt et al. [2005]) combined with the `train-other-500` portion of Librispeech.

Experimental Setup. For the **Supervised** approach, we train the HMM-TDNN system using the Kaldi (Povey et al. [2011]) Librispeech recipe. This recipe trains a sequence of stages, starting with a monophone model followed by a series of triphone models that get increasingly complex (LDA+MLLT followed by LDA+MLLT+SAT), ending in a TDNN model. At specific intermediate stages, the training dataset is sequentially expanded from 100 hr to 460 hr to 960 hr. For hyperparameters and specific details of each stage, please refer to the actual recipe at `https://github.com/kaldi-asr/kaldi/tree/master/egs/librispeech/s5`. Each previous stage model is used to generate (forced Viterbi) alignments that are used to supervise the next stage model. We run forced Viterbi alignment using the final model on the entire training and validation sets using `align_fmllr.sh` and get a time-aligned phoneme boundary file using this alignment using `ali-to-phones`. For the **Unsupervised** approach, we use the Buckeye+ model released by the authors of the Kreuk et al. [2020] paper at `https://github.com/felixkreuk/UnsupSeg`. We use the provided `predict.py` script with the `--prominence 0.05` argument on each audio file in the train and val sets. After obtaining phoneme boundaries from either approach, we convert timestamps to frame indices by multiplying by 50 and rounding off (since the framerate is 20 ms). We also record the last frame index for each audio sample (the end timestamp of the last phoneme).

6.2 HuBERT Pretraining and Evaluation

Dataset Details. We use the LibriSpeech (Panayotov et al. [2015]) dataset for pretraining all our models, following the policy used to train the HuBERT Base models in the original paper. As mentioned before, this dataset consists of 960 hours of training data (combining the three training

splits `train-clean-100, train-clean-360, train-other-500`) and 10 hours of development set data (combining the dev splits `dev-clean` and `dev-other`). For evaluation, we use the datasets mentioned in Section 5.

Experimental Setup. We run all our experiments using the `fairseq` toolkit (Ott et al. [2019]), modifying the dataloader and model code to support our **RandomPhoneme Masking** strategy. We use the HuBERT Base architecture, consisting of 12 Transformer layers, a hidden layer size of 3072 with 8 attention heads in each layer, with a total of 95M parameters. For more hyperparameter details, refer to the HuBERT (Hsu et al. [2021]) paper. First, we train the acoustic unit discovery system. To do this, we run k-means clustering with 500 clusters on the representations generated by the 6^{th} layer of the pretrained HuBERT Base model at `https://github.com/pytorch/fairseq/tree/main/examples/hubert` on a randomly sampled 10% subset of the Librispeech training data. Then, for each of our experiments, we initialize from the pretrained HuBERT Base model and run continued pretraining on the Librispeech 960-hr training set for 40000 additional steps with a batch size of 4200000 tokens (corresponding to 262.5 seconds of audio). We use the same learning rate schedule as the original paper, but with a smaller peak learning rate (0.00005) since we are continuing from a pretrained checkpoint. To run evaluation, we use the `s3prl` toolkit provided by SUPERB and run the provided scripts for each downstream task with the default hyperparameters.

7 Experiments

Exp. No	Continued Pretraining Strategy	ASR (WER)	PR (PER)	KS (Acc)
E0	**None (Leaderboard)**	**6.42**	**5.41**	96.30
E1	**None (Our evaluation)**	**7.09**	6.10	96.55
E2	**RandFrameSpan**	7.18	5.61	96.62
E3	**RandPhon Vanilla + Sup**	7.49	5.73	96.62
E4	**RandPhon Iter w/ Span 1 + Sup**	7.13	5.57	96.72
E5	**RandPhon Iter w/ Span 2 + Sup**	7.11	**5.53**	**96.88**
E6	**RandPhon Iter w/ Span 3 + Unsup**	7.49	5.73	96.78

Table 1: For Automatic Speech Recognition ASR (WER) and Phoneme Recognition PR (PER), lower is better. For Keyword Spotting KS (Acc), higher is better. Each pretraining experiment is described as follows: a) **None** refers to the original HuBERT pretrained checkpoint with no continued pretraining. The *Leaderboard* results are the results reported on the public SUPERB leaderboard. The *Our evaluation* results are results obtained after finetuning the checkpoint on each downstream task ourselves. b) **RandomFrameSpan** refers to continued pretraining using the RandomFrameSpan Masking strategy. c) **RandomPhoneme Vanilla + Supervised** performs pretraining using the RandomPhoneme Vanilla masking strategy using the Supervised phoneme boundaries. d) The three **RandomPhoneme Iterative** experiments use increasing phoneme spans m of 1,2 and 3. The last experiment is conducted with the Unsupervised phoneme boundaries while the first two are conducted using the Supervised phoneme boundaries. The ASR experiment in E6 is finetuned with a slightly smaller number of steps (196000/200000) due to insufficient time to completely train.

Table 1 contains the description as well as the results of all our experiments. Here, we describe our motivation behind these experiments. First, we consider the natural None baseline with no continued pretraining. We note a large discrepancy between the results as reported on the public leaderboard (Experiment E0) and as obtained by our evaluation using exactly the same pretrained checkpoint (Experiment E1) While we have been unable to pinpoint why this discrepancy exists, one obvious reason could be that the public leaderboard HuBERT submission uses dev-set tuned hyperparameters for each downstream task while we use the default hyperparameters. Since these tuned hyperparameters, if any, have not been reported in the literature, and due to insufficient compute budgets to tune these hyperparameters ourselves, we consider the 'Our evaluation' results to be a fair baseline to compare our experiment results to, rather than the 'Leaderboard' results, since we use the default hyperparameters in all our experiments.

Second, we run continued pretraining using the original strategy RandomFrameSpan (Experiment E2). We observe that, as compared to no continued pretraining in E1, we obtain a large improvement in the PR task and some improvement in the KS task; however, ASR slightly suffers. Thus, continued

pretraining, as expected, helps in general. With the Vanilla RandomPhoneme strategy, however, (Experiment E3) we get much worse performance than E2 in both ASR and PR and match the performance in KS. Thus, the Vanilla strategy seems to be a strictly worse pretraining strategy. In Experiments E4 and E5 with the Iterative RandomPhoneme Strategy, we observe that these approaches are able to beat the previous approaches in both PR and KS. In ASR, they are close to the performance of the best model for ASR, E1. This shows that our proposed pretraining strategy is better or nearly comparable to the baselines (E1, E2) in all the tasks! However, note that all these experiments were run with the Supervised phoneme boundaries, which we won't have access to in a true self-supervised setting. Thus, finally we run Experiment E6 with the RandomPhoneme Iterative Strategy with Unsupervised boundaries. We can see that it still beats the baselines (E1, E2) for the KS task; however, it fails to beat baseline E2 for the CTC and ASR task.

8 Conclusion

In this project, we propose a novel linguistically-guided masking strategy for self-supervised speech representation learning called **RandomPhoneme Masking**. We explore different variations of the approach with Supervised and Unsupervised phoneme discovery. We demonstrate its superiority using high-quality supervision-derived phoneme boundaries over simpler fixed-span based strategies, but observe a gap in performance when using unsupervised boundaries. Future work will investigate the reasons for this gap in performance, conduct more ablation studies, run improved postprocessing on unsupervised boundary detection algorithms, and investigate using phoneme segmentation-based ideas to propose a more efficient pretraining paradigm for speech representation learning.

References

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations, 2020.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language Models are Few-Shot Learners, 2020.

Junkun Chen, Mingbo Ma, Renjie Zheng, and Liang Huang. Mam: Masked acoustic modeling for end-to-end speech-to-text translation, 2021a.

Sanyuan Chen, Chengyi Wang, Zhengyang Chen, Yu Wu, Shujie Liu, Zhuo Chen, Jinyu Li, Naoyuki Kanda, Takuya Yoshioka, Xiong Xiao, Jian Wu, Long Zhou, Shuo Ren, Yanmin Qian, Yao Qian, Jian Wu, Michael Zeng, and Furu Wei. WavLM: Large-Scale Self-Supervised Pre-Training for Full Stack Speech Processing, 2021b.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. The original BERT repository, 2019a. URL https://github.com/google-research/bert.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2019b.

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, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised Learning, 2020.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training, 2020.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units, 2021.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving Pre-training by Representing and Predicting Spans, 2020.

Felix Kreuk, Joseph Keshet, and Yossi Adi. Self-Supervised Contrastive Learning for Unsupervised Phoneme Segmentation, 2020.

Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. Contrastive Representation Learning: A Framework and Review. *IEEE Access*, 8:193907–193934, 2020. ISSN 2169-3536. doi: 10.1109/access.2020.3031549. URL http://dx.doi.org/10.1109/ACCESS.2020.3031549.

Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. PMI-Masking: Principled masking of correlated spans, 2020.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A Fast, Extensible Toolkit for Sequence Modeling, 2019.

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)*, pages 5206–5210, 2015. doi: 10.1109/ICASSP.2015.7178964.

Mark A. Pitt, Keith Johnson, Elizabeth Hume, Scott Kiesling, and William Raymond. The Buckeye corpus of conversational speech: labeling conventions and a test of transcriber reliability. *Speech Communication*, 45(1):89–95, 2005. ISSN 0167-6393. doi: https://doi.org/10.1016/j.specom.2004.09.001. URL https://www.sciencedirect.com/science/article/pii/S0167639304000974.

Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovsky, Georg Stemmer, and Karel Vesely. The Kaldi Speech Recognition Toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. ERNIE: Enhanced Representation through Knowledge Integration, 2019.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding, 2019.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems, 2020a.

Chengyi Wang, Yu Wu, Yujiao Du, Jinyu Li, Shujie Liu, Liang Lu, Shuo Ren, Guoli Ye, Sheng Zhao, and Ming Zhou. Semantic mask for transformer based end-to-end speech recognition, 2020b.

Pete Warden. Speech Commands: A public dataset for single-word speech recognition. *Dataset available online*, 2017. URL http://download.tensorflow.org/data/speech_commands_v0.01.tar.gz.

Shu wen Yang, Po-Han Chi, Yung-Sung Chuang, Cheng-I Jeff Lai, Kushal Lakhotia, Yist Y. Lin, Andy T. Liu, Jiatong Shi, Xuankai Chang, Guan-Ting Lin, Tzu-Hsien Huang, Wei-Cheng Tseng, Ko tik Lee, Da-Rong Liu, Zili Huang, Shuyan Dong, Shang-Wen Li, Shinji Watanabe, Abdelrahman Mohamed, and Hung yi Lee. SUPERB: Speech processing Universal PERformance Benchmark, 2021.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow Twins: Self-Supervised Learning via Redundancy Reduction, 2021.