Self-training and Pre-training are Complementary for Speech Recognition

Qiantong Xu[*]   **Alexei Baevski**[*]   **Tatiana Likhomanenko**   **Paden Tomasello**

Alexis Conneau   **Ronan Collobert**   **Gabriel Synnaeve**   **Michael Auli**

Facebook AI Research

Abstract

Self-training and unsupervised pre-training have emerged as effective approaches to improve speech recognition systems using unlabeled data. However, it is not clear whether they learn similar patterns or if they can be effectively combined. In this paper, we show that pseudo-labeling and pre-training with wav2vec 2.0 are complementary in a variety of labeled data setups. Using just 10 minutes of labeled data from Libri-light as well as 53k hours of unlabeled data from LibriVox achieves WERs of 3.0%/5.2% on the clean and other test sets of Librispeech – rivaling the best published systems trained on 960 hours of labeled data only a year ago. Training on all labeled data of Librispeech achieves WERs of 1.5%/3.1%.

1   Introduction

Speech recognition models trained on labeled speech data has progressed substantially in the recent past [1, 2, 3, 4]. A drawback of these models is that they require a lot of labeled data to perform well which is usually only available for English and a few other languages. Therefore, purely supervised training is impractical for the vast majority of the 7,000 languages spoken around the world [5] which is why there has been a lot of interest in how to better use unlabeled speech data [6, 7, 8].

This includes classical self-training [9, 10, 11] which demonstrated strong results [12, 13, 2, 14, 15] by pseudo-labeling unannotated audio data and then retraining the final system with the additional labeled data. Another line of work is pre-training representations on unlabeled speech followed by fine-tuning on labeled data [16, 17, 18, 19, 20, 21, 22, 23, 24].

In this paper we combine self-training and unsupervised pre-training which are different approaches to leveraging unlabeled data. Both achieved excellent results on competitive benchmarks and the central question we explore is whether the two methods are complementary to each other. Specifically, we build on the recently introduced wav2vec 2.0 model [24] and the self-training approach of Kahn et al. (2020; [13]) and Xu et al. (2020; [14]). We explore training models on the pseudo-labeled data from scratch or by fine-tuning the pre-trained model. To better understand how complementary the two methods are, we use the same unlabeled data for both.

Experiments on the full Librispeech corpus as well as the low-resource labeled data setups of Libri-light show that self-training and unsupervised pre-training are indeed complementary, a finding that is inline with recent work in natural language understanding [25]. In a very low resource setup with just 10 minutes of labeled data and LibriVox as unlabeled data, the combination of wav2vec 2.0 and self-training achieves a WER of 3.0%/5.2% on the clean and other test sets of Librispeech, a

---

[*]Equal contribution.

relative WER reduction of 25% and 40% over recent work in pre-training alone [24]. Using just the acoustic model without a language model achieves WER 3.7%/6.5% - supporting the hypothesis that self-training distills the language model used for pseudo-labeling into the final model. When all 960 hours of labeled training data are used we achieve 1.5%/3.1% WER on Librispeech.

2 Background

2.1 Unsupervised Pre-training Model

We experiment with the recently introduced wav2vec 2.0 model of Baevski et al. (2020; [24]). This model contains a convolutional feature encoder $f : \mathcal{X} \mapsto \mathcal{Z}$ to map raw audio $\mathcal{X}$ to latent speech representations $\mathbf{z}_1, \ldots, \mathbf{z}_T$ which are input to a Transformer $g : \mathcal{Z} \mapsto \mathcal{C}$ to output context representations $\mathbf{c}_1, \ldots, \mathbf{c}_T$ [26, 18, 27]. Each $\mathbf{z}_t$ represents about 25ms of audio strided by 20ms and the Transformer architecture follows BERT [28, 26]. During training, feature encoder representations are discretized to $\mathbf{q}_1, \ldots, \mathbf{q}_T$ with a quantization module $\mathcal{Z} \mapsto \mathcal{Q}$ to represent the targets in the objective. The quantization module uses a Gumbel softmax to choose entries from $G = 2$ codebooks with $V = 320$ entries each and the chosen entries are concatenated to obtain $\mathbf{q}$ [29, 30, 18].

The model is trained by solving a contrastive task over masked feature encoder outputs. At training time, spans of ten time steps with random starting indices are masked. The objective requires identifying the true quantized latent $\mathbf{q}_t$ for a masked time-step within a set of $K = 100$ distractors $\mathbf{Q}_t$ sampled from other masked time steps: $-\log \frac{\exp(sim(\mathbf{c}_t, \mathbf{q}_t))}{\sum_{\tilde{\mathbf{q}} \sim \mathbf{Q}_t} \exp(sim(\mathbf{c}_t, \tilde{\mathbf{q}}))}$ where $\mathbf{c}_t$ is the output of the Transformer, and $sim(\mathbf{a}, \mathbf{b})$ denotes cosine similarity. The objective is augmented by a codebook diversity penalty to encourage the model to use all codebook entries [31].

2.2 Self-training Approach

We adopt the pseudo-labeling strategy of Kahn et al. (2020; [13]) and Synnaeve et al. (2020; [2]). This first trains an initial acoustic model on the available labeled data and then labels the unlabeled data with the initial model as well as a language model in a step we call pseudo-labeling. Finally, a new acoustic model is trained on the pseudo-labeled data as well as the original labeled data.

Previous work considered multiple rounds of pseudo-labeling where the labeling step is repeated with each new model to train another model [14]. While iterative pseudo-labeling is more accurate, we opt for a single iteration which is computationally less demanding while still enabling us to reason about whether unsupervised pre-training and pseudo-labeling are complementary. Another line of work investigated filtering the resulting pseudo-labeled data to match the distribution of the original labeled data [15]. Both methods may improve results and we leave them to future work.

2.3 Combining the two Approaches

To combine the approaches, we replace the initial model for pseudo-labeling with a pre-trained model. The resulting training pipeline is as follows: we first pre-train a wav2vec 2.0 model on the unlabeled data, fine-tune it on the available labeled data, use the model to label the unlabeled data, and finally use the pseudo-labeled data to train the final model. In our experiments, we also consider a variant where we fine-tune the original wav2vec 2.0 model on the pseudo-labeled data.

3 Experimental Setup

3.1 Datasets

As unlabeled data for pre-training and self-training we consider the speech audio of the Librispeech corpus (LS-960; [32]) without transcriptions containing 960h of audio as well as the audio data of LibriVox (LV-60k). For the latter we follow the pre-processing of Kahn et al. (2020; [33]) resulting in 53.2k hours of audio. We consider five labeled data setups: all 960h of transcribed Librispeech, the train-clean-100 subset comprising 100h, as well as the Libri-light limited resource training subsets of train-10h (10h), train-1h (1h), and train-10min (10min). We evaluate on the standard Librispeech dev-other/clean and test-clean/other sets.

3.2 Pre-trained models

Pre-trained models are implemented in fairseq [34] and we obtain them from the public fairseq github repository.[2] The repository provides fine-tuned models for the five labeled data setups we consider (§ 3.1). We experiment with the LARGE configuration comprising 24 transformer blocks with model dimension 1,024, inner dimension 4,096 and 16 attention heads, comprising a total of about 300M parameters. The feature encoder contains seven blocks and the temporal convolutions in each block have 512 channels with strides (5,2,2,2,2,2,2) and kernel widths (10,3,3,3,3,2,2), resulting in a receptive field of about 25ms and a stride of about 20ms. After pre-training on the unlabeled data, this model is fine-tuned on the labeled data using Connectionist Temporal Classification (CTC; [35]) and a letter-based output vocabulary.

3.3 Self-training

We pseudo-label the audio data of either LS-960 or LV-60k using wav2vec 2.0 LARGE fine-tuned on different labeled data splits. For labeling, we follow the two-pass rescoring procedure of Synnaeve et al. (2020; [2]): first, we generate a list of candidate transcriptions by combining wav2vec 2.0 and the standard Librispeech 4-gram language model during beam-search with beam 800. Next, the n-best list is pruned to the 50 highest scoring entries and then rescored with a Transformer LM trained on the Librispeech language corpus [36, 2]. The Transformer LM has 20 blocks with model dimension 1,280, inner dimension 6,144 and 16 attention heads. The n-gram model obtains perplexity 150.3 on the development set and the Transformer language model 49.2. We found this to be more efficient than directly integrating the Transformer LM into beam search at little loss in accuracy. Decoding and rescoring hyper-parameters are tuned on dev-other of Librispeech for each experiment using a random parameter search. The LM weight and the word insertion penalty [2] is tuned by randomly sampling values in the range of [0, 5] and [-5, 5] over 128 trials.

3.4 Final Model

We follow Synnaeve et al. (2020; [2]) and train a Transformer-based sequence to sequence model with log-Mel filterbank inputs after pseudo-labeling using wav2letter++ [37]. The encoder uses a convolutional frontend containing 4 layers of temporal convolutions with kernel width 3, followed by 36 Transformer blocks with model dimension 768, 4 attention heads and feed-forward network (FFN) dimension 3072 [28, 2]. The model contains about 300M parameters.

We use a 10k word piece output vocabulary computed from the training transcriptions if the whole Librispeech training set is used as labeled data [38]. Otherwise, we switch to the 5k WP estimated on the train-clean-100 transcriptions [14, 8]. Language models are incorporated similar to § 3.3. We use a 4-gram language model and then rescore with a Transformer LM. The beam size used in both decoding and rescoring is 50.

4 Results

4.1 Low-Resource Labeled Data

Pre-training has been shown to be very effective in both high- and low-resource labeled training data setups whereas self-training has been most effective when at least a moderate amount of labeled data is available ($\geq$ 100h; [14, 15]). To get a sense of whether the combination of both methods can be even more effective, we start with experiments on the Libri-light setups with 10min, 1h and 10h of labeled data. For pre-training and pseudo-labeling we use either the 960h of Librispeech without transcriptions or the 53.2k hours of LibriVox (§ 3.1). As baseline we consider wav2vec 2.0 pre-trained on Librispeech and fine-tuned on one of the labeled data splits.

We use the publicly available wav2vec 2.0 models to pseudo-label (ST) the unlabeled data and then evaluate two options to train the final model on the resulting labels: one is to train a new sequence to sequence model from random initialization with a word-piece vocabulary (s2s scratch) following Synnaeve et al. (2019; [2]; § 3.4). Another option is to fine-tune wav2vec 2.0 on the pseudo-labeled data with CTC and a letter-based vocabulary (ctc ft).

---

[2]`https://github.com/pytorch/fairseq/tree/master/examples/wav2vec`

Table 1: WER on the Librispeech dev and test sets for the Libri-light low-resource labeled data setups of 10 min, 1 hour and 10 hours. As unlabeled data we use the audio of Librispeech (LS-960) or the larger LibriVox (LV-60k). ST (s2s scratch) trains a sequence to sequence model with a word-piece vocabulary on the pseudo-labeled data from random initialization while as ST (ctc ft) fine-tunes wav2vec 2.0 with the pseudo-labels using CTC and a letter-based vocabulary. All results are with language models at inference time.

Model	Unlbld data	dev		test	
		clean	other	clean	other
10 min labeled					
Discr. BERT [27]	LS-960	15.7	24.1	16.3	25.2
wav2vec 2.0 [24]	LS-960	6.6	10.6	6.8	10.8
+ ST (s2s scratch)	LS-960	4.1	7.0	5.0	8.1
+ ST (ctc ft)	LS-960	3.6	6.6	4.0	7.2
wav2vec 2.0 [24]	LV-60k	5.0	8.4	5.2	8.6
+ ST (s2s scratch)	LV-60k	2.6	4.7	3.1	5.4
+ ST (ctc ft)	LV-60k	2.8	4.6	3.0	5.2
1h labeled					
Discr. BERT [27]	LS-960	8.5	16.4	9.0	17.6
wav2vec 2.0 [24]	LS-960	3.8	7.1	3.9	7.6
+ ST (s2s scratch)	LS-960	2.9	5.6	3.4	6.6
+ ST (ctc ft)	LS-960	2.8	5.5	3.1	6.3
10h labeled					
Discr. BERT [27]	LS-960	5.3	13.2	5.9	14.1
IPL [14]	LS-960	23.5	25.5	24.4	26.0
wav2vec 2.0 [24]	LS-960	2.9	5.7	3.2	6.1
+ ST (s2s scratch)	LS-960	2.5	5.1	3.5	5.9
+ ST (ctc ft)	LS-960	2.6	5.2	2.9	5.7

Table 1 shows that the combination of pre-training and self-training (wav2vec 2.0 + ST) outperforms pre-training alone (wav2vec 2.0) across all low-resource setups. It also achieves a very large improvement over iterative pseudo-labeling [14] in the 10h labeled setup. This is because the initial model is much stronger due to pre-training and it is very difficult to train a good supervised-only model on just 10h of labeled data.

With just 10 minutes of labeled data, the combination of pre-training and pseudo-labeling with LibriVox achieves WER 5.2% on test-other. Using Librispeech (LS-960) as unlabeled data and 10 minutes of labeled data, wav2vec 2.0 + ST achieves 4.0%/7.2% WER on test-clean/other compared to 4.2%/8.6% for the best known pseudo-labeling approach [15] which uses 100 hours of labeled data. More unlabeled data leads to large improvements, reducing WER from 4.0%/7.2% for LS-960 to 3.0%/5.2% for LV-60k, a relative WER reduction of 25-28%. But increasing the amount of labeled data without more unlabeled data leads to diminishing returns - an issue we return to in § 5. Fine-tuning (ctc ft) generally outperforms from scratch training of a sequence to sequence model with a WP vocabulary (s2s scratch). This is likely because the model can leverage the pre-trained representations.

4.2 High-Resource Labeled Data

Next, we evaluate performance with more labeled data. We consider the 100h clean subset of Librispeech as well as all 960h of labeled data in Librispeech. Table 2 shows that LS-960 as unlabeled data is not enough to outperform the baseline when 100h of labeled data is available. However, performance improves when using the much larger LV-60k, achieving a 10% relative WER reduction on test-other over wav2vec 2.0.

When using the full Librispeech benchmark as labeled data, combining wav2vec 2.0 and pseudo-labeling achieves WER 1.5%/3.1%. This result was achieved with a strong sequence to sequence

Table 2: WER on Librispeech when using the clean 100h subset as labeled data or all 960h of Librispeech (cf. Table 1). Prior work used 860 unlabeled hours (LS-860) but the total with labeled data is 960h and comparable to our setup.

Model	Unlbld data	dev		test	
		clean	other	clean	other
100h labeled					
Discr. BERT [27]	LS-960	4.0	10.9	4.5	12.1
ST [13]	LS-860	5.4	19.0	5.8	20.1
IPL [14]	LS-860	5.0	8.0	5.6	9.0
Noisy student [15]	LS-860	3.9	8.8	4.2	8.6
wav2vec 2.0 [24]	LS-960	2.1	4.8	2.3	5.0
+ ST (s2s scratch)	LS-960	2.3	4.6	2.7	5.4
+ ST (ctc ft)	LS-960	2.2	4.6	2.4	5.0
IPL [14]	LV-60k	3.19	6.14	3.72	7.11
wav2vec 2.0 [24]	LV-60k	1.9	4.0	2.0	4.0
+ ST (s2s scratch)	LV-60k	1.4	2.8	1.9	3.8
+ ST (ctc ft)	LV-60k	1.7	3.2	1.9	3.6
960h labeled					
Supervised					
SpecAugment [1]	-	-	-	2.5	5.8
ContextNet [3]	-	1.9	3.9	1.9	4.1
Conformer [4]	-	2.1	4.3	1.9	3.9
Semi-supervised					
IPL [14]	LV-60k	1.85	3.26	2.10	4.01
Noisy Student [15]	LV-60k	1.6	3.4	1.7	3.4
wav2vec 2.0 [24]	LV-60k	1.6	3.0	1.8	3.3
+ ST (s2s scratch)	LV-60k	1.1	2.7	1.5	3.1
+ ST (ctc fine-tune)	LV-60k	1.6	2.9	1.8	3.3

model trained from scratch. While less effective than fine-tuning with CTC on smaller setups, a powerful sequence to sequence model excels in this larger setting since the decoder part of the model, which acts in part like a language model, does not overfit. The lower performance of fine tuning is likely due to CTC not being as competitive as more elaborate sequence to sequence models when a lot of pseudo-labeled data is available [2].

4.3 Results without a Language Model at Inference Time

Table 3 shows that combined training models have very good performance even without a language model. This is because the language model used during pseudo-labeling was partly distilled into the pseudo-labeled data [2]. This effect is particularly striking for the 10 min labeled setup without LM where wav2vec 2.0 + ST (s2s scratch) reduces the WER of the baseline (wav2vec 2.0 - LM) by 83% relative on test-other. As more labeled data becomes available, the performance of the acoustic model without a language model improves but there is still a clear effect of self-trained models having distilled the language model. Generally, the sequence to sequence model in the (s2s scratch) setting is better able to distill the language model used at pseudo-labeling time compared to the CTC model used in fine-tuning.

5 Analysis

We previously saw that improvements decreased with more labeled data (§ 4.1). To better understand this, we perform an experiment on Librispeech where we consider data setups with a fixed ratio between the unlabeled and labeled data. Table 4 shows that relative improvements are a function of the amount of unlabeled data relative to the labeled data, rather than the amount of labeled data alone. Table 1 showed much larger improvements for the 10 min labeled split but with a fixed ratio

Table 3: WER on Librispeech with and without a language model (LM) for 10 min, and 960h of labeled data and LibriVox as unlabeled data.

Model	dev		test	
	clean	other	clean	other
10 min labeled				
wav2vec 2.0 [24]	5.0	8.4	5.2	8.6
- LM	38.3	41.0	40.2	38.7
wav2vec 2.0 + ST (s2s scratch)	2.6	4.7	3.1	5.4
- LM	3.3	5.9	3.7	6.5
wav2vec 2.0 + ST (ctc ft)	2.8	4.6	3.0	5.2
- LM	4.2	6.9	4.3	7.2
960h labeled				
wav2vec 2.0 [24]	1.6	3.0	1.8	3.3
- LM	2.1	4.5	2.2	4.5
wav2vec 2.0 + ST (s2s scratch)	1.1	2.7	1.5	3.1
- LM	1.3	3.1	1.7	3.5
wav2vec 2.0 + ST (ctc ft)	1.6	2.9	1.8	3.3
- LM	1.7	3.6	1.9	3.9

Table 4: The main driver of performance is the ratio between labeled and unlabeled data. We add 8.6 times as much unlabeled data to each labeled setup. Results are on dev-other with an n-gram model and subsets of LS-960 as unlabeled data.

	labeled	unlab	dev-other	% change
wav2vec 2.0	10min	86 min	12.9	
+ ST	10min	86 min	12.0	7%
wav2vec 2.0	1h	8.6h	8.5	
+ ST	1h	8.6h	7.6	11%
wav2vec 2.0	10h	86h	6.9	
+ ST	10h	86h	6.5	6%
wav2vec 2.0	100h	860h	5.7	
+ ST	100h	860h	5.3	7%

of labeled and unlabeled data, the relative improvement is comparable to the 100h labeled setup (Table 2).

6 Conclusion

Unsupervised pre-training and pseudo-labeling are complementary for speech recognition. This enables building speech recognition systems with as little as 10 minutes of transcribed speech with word error rates that only a year ago were reserved to the best systems trained on 960 hours of labeled data.

References

[1] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le. Specaugment: A simple data augmentation method for automatic speech recognition. In *Proc. of Interspeech*, 2019.

[2] G. Synnaeve et al. End-to-end ASR: from Supervised to Semi-Supervised Learning with Modern Architectures. *arXiv*, abs/1911.08460, 2019.

[3] W. Han et al. Contextnet: Improving convolutional neural networks for automatic speech recognition with global context. *arXiv*, 2020.

[4] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, and et al. Conformer: Convolution-augmented transformer for speech recognition. *arXiv*, 2020.

[5] M. P. Lewis, G. F. Simon, and C. D. Fennig. Ethnologue: Languages of the world, nineteenth edition. Online version: `http://www.ethnologue.com`, 2016.

[6] A. H. Liu, H.-Y. Lee, and L.-S. Lee. Adversarial training of end-to-end speech recognition using a criticizing language model. *arXiv*, 2018.

[7] M. K. Baskar, S. Watanabe, R. Astudillo, T. Hori, L. Burget, and J. Černocký. Semi-supervised sequence-to-sequence asr using unpaired speech and text. *arXiv*, 2019.

[8] W.-N. Hsu, A. Lee, G. Synnaeve, and A. Hannun. Semi-supervised speech recognition via local prior matching. *arXiv*, 2020.

[9] H. Scudder. Probability of error of some adaptive pattern-recognition machines. *IEEE Trans. on Inform. Theory*, 1965.

[10] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proc. of ACL*, 1995.

[11] E. Riloff. Automatically generating extraction patterns from untagged text. In *Proc. of AAAI*, 1996.

[12] S. H. K. Parthasarathi and N. Strom. Lessons from building acoustic models with a million hours of speech. *arXiv*, 2019.

[13] J. Kahn, A. Lee, and A. Hannun. Self-training for end-to-end speech recognition. In *Proc. of ICASSP*, 2020.

[14] Q. Xu, T. Likhomanenko, J. Kahn, A. Hannun, G. Synnaeve, and R. Collobert. Iterative pseudo-labeling for speech recognition. *arXiv*, 2020.

[15] D. S. Park, Y. Zhang, Y. Jia, W. Han, C.-C. Chiu, and et al. Improved noisy student training for automatic speech recognition. *arXiv*, 2020.

[16] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv*, abs/1807.03748, 2018.

[17] S. Schneider, A. Baevski, R. Collobert, and M. Auli. wav2vec: Unsupervised pre-training for speech recognition. In *Proc. of Interspeech*, 2019.

[18] A. Baevski, S. Schneider, and M. Auli. vq-wav2vec: Self-supervised learning of discrete speech representations. In *Proc. of ICLR*, 2020.

[19] Y.-A. Chung, W.-N. Hsu, H. Tang, and J. R. Glass. An unsupervised autoregressive model for speech representation learning. *arXiv*, abs/1904.03240, 2019.

[20] D. Jiang, X. Lei, W. Li, N. Luo, Y. Hu, and et al. Improving transformer-based speech recognition using unsupervised pre-training. *arXiv*, abs/1910.09932, 2019.

[21] K. Kawakami, L. Wang, C. Dyer, P. Blunsom, and A. v. d. Oord. Learning robust and multilingual speech representations. *arXiv*, 2020.

[22] M. Rivière, A. Joulin, P.-E. Mazaré, and E. Dupoux. Unsupervised pretraining transfers well across languages. *arXiv*, abs/2002.02848, 2020.

[23] W. Wang, Q. Tang, and K. Livescu. Unsupervised pre-training of bidirectional speech encoders via masked reconstruction. *arXiv*, 2020.

[24] A. Baevski, H. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Proc. of NeurIPS*, 2020.

[25] J. Du, E. Grave, B. Gunel, V. Chaudhary, O. Celebi, and et al. Self-training improves pre-training for natural language understanding. *arXiv*, 2020.

[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv*, abs/1810.04805, 2018.

[27] A. Baevski, M. Auli, and A. Mohamed. Effectiveness of self-supervised pre-training for speech recognition. *arXiv*, abs/1911.03912, 2019.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, and et al. Attention is all you need. In *Proc. of NIPS*, 2017.

[29] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2011.

[30] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv*, abs/1611.01144, 2016.

[31] S. Dieleman, A. v. d. Oord, and K. Simonyan. The challenge of realistic music generation: modelling raw audio at scale. *arXiv*, 2018.

[32] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *Proc. of ICASSP*, pages 5206–5210. IEEE, 2015.

[33] J. Kahn and et al. Libri-light: A benchmark for asr with limited or no supervision. In *Proc. of ICASSP*, 2020.

[34] M. Ott et al. fairseq: A fast, extensible toolkit for sequence modeling. In *Proc. of NAACL Sys. Demo.*, 2019.

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

[36] A. Baevski and M. Auli. Adaptive input representations for neural language modeling. In *Proc. of ICLR*, 2018.

[37] V. Pratap et al. Wav2letter++: A fast open-source speech recognition system. In *Proc. of ICASSP*, 2019.

[38] T. Kudo and J. Richardson. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proc. of EMNLP Sys. Demo.*, 2018.