WAV2SEQ: PRE-TRAINING SPEECH-TO-TEXT ENCODER-DECODER MODELS USING PSEUDO LANGUAGES

Felix Wu† Kwangyoun Kim† Shinji Watanabe‡ Kyu J. Han† Ryan McDonald‡ Kilian Q. Weinberger‡‡ Yoav Artzi‡‡

†ASAPP Inc. ‡Carnegie Mellon University ‡‡Cornell University

ABSTRACT

We introduce Wav2Seq, the first self-supervised approach to pre-train both parts of encoder-decoder models for speech data. We induce a pseudo language as a compact discrete representation, and formulate a self-supervised pseudo speech recognition task — transcribing audio inputs into pseudo subword sequences. This process stands on its own, or can be applied as low-cost second-stage pre-training. We experiment with automatic speech recognition (ASR), spoken named entity recognition, and speech-to-text translation. We set new state-of-the-art results for end-to-end spoken named entity recognition, and show consistent improvements on 8 language pairs for speech-to-text translation, even when competing methods use additional text data for training. On ASR, our approach enables encoder-decoder methods to benefit from pre-training for all parts of the network, and shows comparable performance to highly optimized recent methods.

Index Terms— Self-supervision, Encoder-decoder, Speech recognition, Spoken named entity recognition, Speech translation

1. INTRODUCTION

Self-supervised pre-trained models have recently become a core part of speech models [1, 2, 3, 4], leading to impressive performances on a wide variety of speech tasks [5, 6, 7, 8]. Most of these approaches rely on pre-training an encoder to create expressive representations of input data. If a sequential decoder is needed for downstream tasks (i.e., for generative tasks), it is often trained with task-specific supervised data. The most common approaches for automatic speech recognition (ASR) follow this encoder-decoder paradigm [9, 10, 11, 12, 13, 14], regardless if they use sequence transducers [15] or sequence-to-sequence [16, 17, 18] architectures. Because all existing self-supervised learning approaches for speech focus on pre-training an encoder model only, when adapted to an encoder-decoder architecture, the decoder has to be randomly initialized or borrowed from a pre-trained NLP decoder [19, 20].

We propose Wav2Seq, a new† self-supervised approach to jointly pre-train the encoder and decoder for speech data. We automatically induce pseudo subwords that form a compact discrete representation of spoken language. We treat these as audio transcripts in a pseudo ASR task, and use them as the targets for Seq2Seq learning (Fig. 1). When fine-tuned on a downstream task (e.g., ASR or speech translation), the input and output embedding layers are replaced in order to adapt to natural language.

†Concurrent to the preprint version [21] of this work, Ao et al. [22] introduce Speech2C, which is similar to Wav2Seq trained on pseudo characters. Their experiments are only limited to Librispeech ASR task [5] where they additionally use joint CTC/Attention decoding to improve the performance.

We conduct extensive experiments on ASR, spoken named entity recognition (SNER), and speech-to-text translation (ST). We show, in settings with limited labeled audio data (≤10h), Wav2Seq boosts the performance of encoder-decoder models significantly and closes the gap to CTC ASR models.

When we apply Wav2Seq as a low-cost second-stage pre-training method, models based on existing pre-trained encoders (e.g., HuBERT) achieve even better results. On the SLUE-VoxPopuli SNER benchmark [8], Wav2Seq initialized with HuBERT achieves the best end-to-end results. For ST tasks, we conduct experiments on four from-English and four to-English language pairs. Wav2Seq consistently outperforms models initialized with HuBERT or XLS-R pre-trained models and achieves similar BLEU scores as models trained on additional machine annotated text data. Pre-trained models and code are available at https://github.com/asappresearch/wav2seq.

2. RELATED WORK

In NLP, it has been observed that pre-trained Seq2Seq models outperform pre-trained encoders on text generation tasks [23, 24] despite being less competitive in discriminative tasks. Lewis et al. [23] introduced BART, a Seq2Seq Transformer pre-trained on text denoising. Concurrently, T5 [24] is pre-trained on a large collection of NLP tasks, showing effective task transfer behavior.

CPC [1] and wav2vec [2] are two early approaches for self-supervised speech representation learning based on contrastive loss. vq-wav2vec [25] learns discretized speech features, which are then applied as inputs to a BERT-like model [26]. Wav2vec 2.0 [3] is the first self-supervised model to outperform purely supervised approaches on ASR. Instead of directly reconstructing masked features, it uses a contrastive loss — repelling the quantized version of the cor-
we also experiment with pre-training Wav2Seqs with a sequence (in terms of the sequence length) of the audio input. Of these sequences (i.e., wanting them to be short) with the quality of computation with respect to sequence length, we balance the length tokens from speech. Because Transformer decoders require quadratic computation and machine translation to balance between the benefits and costs of character and word representations (i.e., vocabulary size, unit semantics, handling of unseen words, etc). For example, a word “negotiation” can be represented by “me” “go” “in” and “ation” with a subword tokenizer, which allows the model to share the embedding of the common suffix “ation” across different words. Similarly, common pseudo characters sequences are merged to pseudo subwords.

4. EXPERIMENTAL SETUP

We use LibriSpeech [5] data for pre-training and focus on low-resource ASR using 10h and 100h subsets of labels. Similar to the training of wav2vec 2 large and HuBERT-large, we use LibriLight [30] to pre-train large models. We use CoVoST-2 speech translation data set [6] for ST experiments and use SLUE-VoxPopuli dataset [8] for spoken NER experiments. We select the hyper-parameters using the development set of each task.

Pretraining: We use the official baselines for most prior work [3, 4, 20], which are implemented in fairseq [31]. Our Wav2Seq is also implemented as a plugin of fairseq. We use the mini-batch k-means algorithm [32] with k-mean++ initialization [33] implementation in scikit-learn [34]. Following HuBERT’s best hyper-parameters, the number of clusters C is set to 500 by default unless specified separately. We use the byte-pair encoding (BPE) [29] implementation from Huggingface’s tokenizers library2 as the subtokenization method where the vocabulary size of subword tokens V is set to 30K (or 10K for the tiny models). Following Baevski et al. [3], we pre-trained models with a batch size of at most 87.5 seconds per GPU on 8 GPUs and an update frequency of 4 to simulate 32-GPU training as Baevski et al. [3]. We pre-train the models for 100K updates (or 25K updates in second-stage pre-training) unless specified otherwise. We use the same hyper-parameters as prior work [3, 35]. All models are pre-trained with eight NVIDIA V100 GPUs with half-precision. We tie the weights of input and output embeddings of the decoder during pre-training and fine-tuning [36, 37].

Fine-tuning: We use BPE with vocabulary size 1,000 to tokenize the text in ASR, spoken NER, or ST tasks. Similar to prior work [3, 4], we use learning rate $5 \times 10^{-5}$ for fine-tuning and use tri-stage learning rate scheduler with 2,000 steps for linear warm-up and the last 50% of the updates for exponential decay.3 The number of fine-tuning steps depends on the datasets. When fine-tuned on LibriSpeech 10h (or 100h) data, we use a batch size with at most 50 seconds per GPU of audio for 20K (or 80K) and updates on eight GPUs with half-

---

1. We use LibriSpeech [5] for pre-training and focus on low-resource ASR using 10h and 100h subsets of labels.
2. https://github.com/huggingface/tokenizers
3. We tune the learning rate $\epsilon \in \{2 \times 10^{-4}, 10^{-4}, 5 \times 10^{-5}, 2 \times 10^{-5}\}$ on the small models on ASR and fix them.

---

Fig. 2: Mapping an audio input into pseudo subwords. Here we use the first 0.6 seconds of a real audio from LibriSpeech [5] training set as an example. We use models with 25 clusters and 1000 BPE tokens in this example. Pseudo subword is a compact representation (in terms of the sequence length) of the audio input.
Wav2Seq can easily adapt to an ASR task with only limited super-
Wav2Seq models in this table are trained on hidden units and pseudo
with different fine-tuning setups. In the first two rows, as is already
we follow the hyperparameters provided by Hsu et al. [4].
we increase the number of updates to 320K because it has more data.
vision (row 6). Row 7 shows that it is more important to have a
training cost, Wav2Seq is able to significantly improve HuBERT's
Consequently, with only a relatively small amount of additional pre-
tuning cost, Wav2Seq is able to significantly improve HuBERT's
among all the approaches. The numbers in the first four rows are
provided by Shon et al. [8].

5. EXPERIMENTS

5.1. Automatic Speech Recognition (ASR)

Small Model Experiment. We conduct initial small-scale experiments using tiny models with embedding size 256, four attention heads, and feed-forward embedding size 1,024 in each Transformer block. Each model has 6 or 12 Transformer blocks. To speed up the model, we utilize a compact wave feature extractor (WFE-C-c128i0) [35]. This extractor has been shown to perform similar to the wave feature extractor used in wav2vec 2.0 and HuBERT, but faster. All models are pre-trained on LibriSpeech with a semi-supervised setup using 960h unlabelled recordings for pre-training and 10h labelled data for fine-tuning. To be comparable, we use the 9th layer of the official second iteration HuBERT-base model to extract both hidden units and pseudo subwords for pre-training HuBERT baselines and our Wav2Seq models. All HuBERT models in this experiment are third-iteration of HuBERT models. Table 1 shows word error rate (WER) of HuBERT and Wav2Seq with different fine-tuning setups. In the first two rows, as is already known, HuBERT performs strongly when fine-tuning with CTC objective which does not require a Transformer decoder. However, it is challenging to train a HuBERT encoder with a randomly initialized decoder with only 10h labelled data (rows 4 and 5). In contrast, our Wav2Seq can easily adapt to an ASR task with only limited supervision (row 6). Row 7 shows that it is more important to have a deeper encoder (8 layers) and a shallower decoder (4 layers), which allows closing the gap between CTC fine-tuning and Seq2Seq models. For row 8, we use Wav2Seq as a second stage pre-training — the encoder of Wav2Seq is initialized with a pre-trained HuBERT model. Consequently, with only a relatively small amount of additional pre-training cost, Wav2Seq is able to significantly improve HuBERT's

| # | Pre-training Method | Pre-train Iterations | Layers | dev-other WER (%) |
|---|---|---|---|---|
| CTC Models: | | | | |
| 1 | HuBERT | 100K | 6 / 0 | 51.7 |
| 2 | HuBERT | 100K | 12 / 0 | 36.5 |
| Seq2Seq Models: | | | | |
| 3 | No pre-training | 0 | 6 / 6 | ≥100.0 |
| 4 | HuBERT | 100K | 6 / 6 | ≥100.0 |
| 5 | HuBERT | 100K | 12 / 6 | 85.8 |
| 6 | Wav2Seq | 100K | 6 / 6 | 38.1 |
| 7 | Wav2Seq | 100K | 8 / 4 | 36.6 |
| 8 | Wav2Seq (from HuBERT) | 100K + 25K | 6 / 6 | 34.7 |
| Transducer Models: | | | | |
| 9 | No pre-training | 0 | 6 / 3 | 93.5 |
| 10 | HuBERT | 100K | 6 / 3 | 93.6 |
| 11 | Wav2Seq | 100K | 6 / 3 | 44.2 |
| 12 | Wav2Seq (from HuBERT) | 100K + 25K | 6 / 3 | 40.8 |

Table 1: Small model ASR experiment with unsupervised pretraining on LibriSpeech 960h and fine-tuning with LibriSpeech 10h ASR labels. All models have embedding size 256, FFN size 1,024, and four attention heads in their Transformer layers. All the HuBERT and Wav2Seq models in this table are trained on hidden units and pseudo language tokens produced by the official HuBERT-base.

5.2. Automatic Speech Recognition (ASR)

We conduct similar experiments on a Trans-
experiment using a randomly initialized decoder is inferior to the simple CTC objective [15] model architecture and show that Wav2Seq pre-training is not restricted to Seq2Seq models. Rows 11 & 12 in Table 1 demonstrate that Transducers pretrained using Wav2Seq also outperforms a Transducer with its encoder initialized with HuBERT (row 10).

Standard Model Size Model Experiments. We use the official HuBERT-base and HuBERT-large pre-trained on LibriSpeech and LibriLight as the initialization of Wav2Seq encoders following our observation that Wav2Seq works best as second stage pre-training. We further pre-train the models on the same corpus with relatively few updates (25K or 100K iterations compared to 400K interactions for HuBERT models). For the decoder, we use six Transformer blocks with the same width and number of heads as the encoder. Our Wav2Seq (from HuBERT-base) and Wav2Seq (from HuBERT-large) models take 14 and 49 hours to be fine-tuned for 25K updates on eight NVIDIA V100 GPUs, a relatively small compute budget (less than 10%) compared to training HuBERT models (usually 400K updates).

Table 2 shows WER for standard sized models. Even with a well trained HuBERT-base, fine-tuning with the Seq2Seq architecture using a randomly initialized decoder is inferior to the simple CTC objective (rows 2 and 3). Using Wav2Seq closes the gap or even makes Seq2Seq models outperform CTC counterparts (row 4). However, we observe that pre-training longer helps slightly for Wav2Seq (from HuBERT-base) (row 5). We also compare models with 100h labelled data to confirm our observation.

| # | Model Type | PT method | dev | test |
|---|---|---|---|---|
| 1 | Pipeline | Wav2Seq + DeBERTa大型 | 65.3 | 57.8 |
| 2 | Pipeline | Wav2Seq + DeBERTa大型 | 74.9 | 69.6 |
| 3 | E2E CTC | Wav2Seq大型 | 55.6 | 50.9 |
| 4 | E2E CTC | Wav2Seq大型 + DeBERTa大型 | 70.2 | 64.8 |
| 5 | E2E Seq2Seq | Wav2Seq大型 | 64.0 | 58.5 |
| 6 | E2E Seq2Seq | Wav2Seq (from HuBERT大型) | 71.7 | 65.4 |

Table 2: Librispeech dev-other WER (%) with different amount of labelled data. Wav2Seq as a second stage pre-training method significantly improves the Seq2Seq model performance and allows it to match or even outperform CTC fine-tuning. No LMs are used.

Table 3: Development and test F1 scores (%) on SLUE-VoxPopuli NER [8]. Wav2Seq achieves the best performance among all end-to-end methods without access to a language model. A pipeline model using an NLP pre-trained model (DeBERTa大型) remains the best among all the approaches. The numbers in the first four rows are provided by Shon et al. [8].

Seq2Seq performance — demonstrating that HuBERT and Wav2Seq pre-training are truly complimentary.

Transducer Models. We conduct similar experiments on a Trans-
ducer [15] model architecture and show that Wav2Seq pre-training is not restricted to Seq2Seq models. Rows 11 & 12 in Table 1 demonstrate that Transducers pretrained using Wav2Seq also outperforms a Transducer with its encoder initialized with HuBERT (row 10).

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Table 5: Test BLEU scores on CoVoST-2 X-to-En high-resource language pairs. *: numbers from XLS-R paper [20].

| #  | Model                        | Fr   | De  | Ca  | Es  | Avg  |
|----|------------------------------|------|-----|-----|-----|------|
| 1  | Wav2Seq (from HuBERT-large)  | 33.2 | 28.9| 34.0| 29.7| 31.0 |
| 2  | Wav2Seq (from XLS-R (0.3B)) | 33.0 | 28.0| 33.8| 29.9| 31.1 |

Use an mBART decoder (pretrained on multilingual machine translation data):

| #  | Model                        | Fr   | De  | Ca  | Es  | Avg  |
|----|------------------------------|------|-----|-----|-----|------|
| 3  | XLS-R (0.3B) [20] + mBART-ML50N1 | 32.9 | 26.7 | 34.1 | 28.7 | 30.6 |
| 4  | XLSR-53 [41] + mBART-ML50N1 | 32.3 | 26.9 | 33.3 | 28.6 | 30.3 |
| 5  | VP-100K [42] + mBART-ML50N1 | 30.4 | 23.4 | 31.1 | 25.7 | 27.7 |
| 6  | XMEF-En [43] + mBART-ML50N1 | 35.0 | 28.2 | 35.2 | 31.1 | 32.4 |
| 7  | XMEF-X [43] + mBART-ML50N1 | 36.1 | 30.6 | 38.1 | 34.8 | 34.2 |

Table 6: WER (%) on LibriSpeech dev-other set for pseudo ASR tasks with different target sequences. The length compression rate is the ratio between the decoder sequence length and the encoder sequence length (100% means no compression).

| Method                        | Target Tokens | Compress WER (%) |
|-------------------------------|---------------|------------------|
| Wav2Seq                        | Pseudo Subwords | 17.4 % | 38.1 |
|                                | - BPE tokenization | Pseudo Characters | 39.2 % | 44.1 |
|                                | - Deduplication Hidden Units | 100.0 % | 96.5 |

5.3. Speech-to-text Translation (ST)

We conduct speech-to-text translation experiments on 8 language pairs in CoVoST-2 [6] speech translation dataset including four English-to-X and four X-to-English tasks that are commonly used in prior work [19, 20]. Encoder-decoder (i.e., Seq2Seq) models are considered particularly suitable for speech-to-text translation tasks, where the input and output sequences are not monotonically aligned. Besides the English Wav2Seq (from HuBERT-large), we also experiment a Wav2Seq (from XLS-R (0.3B)) for X-to-English tasks which uses a multi-lingual XLS-R encoder [20] pre-trained on 500K hours of audio in 128 languages. We use LibriLight for second stage pre-training and pre-train the model for only 25K updates.

En-to-X. Table 5 shows the test BLEU scores on four English-to-X pairs with the full 430h labelled data used for fine-tuning. Our Wav2Seq (from HuBERT-large) model (row 2) outperforms most of the models with the same size encoder (rows 3 to 6) and matches the performance of the models with 2× or 3× encoder size (W2V2 (0.72B) and XLS-R (1B), i.e. rows 7 & 8). Notably, these models use a large mBART-ML50N1 decoder which has 4× the number of parameters compared to our decoder and is pre-trained on a gigantic text-based machine translation corpus. In contrast, our model is pre-trained on speech data only. Our model also matches the performance of a wav2vec 2.0 large using 60K hours of audio for self-training (row 10). Admittedly using a gigantic XLS-R (2B) [20] or decoding with LM [19] remains the state-of-the-art; however, these improvements are orthogonal to our proposed method and can be applied to our model as well.

X-to-En. Table 5 shows the test BLEU scores of the four X-to-English language pairs. Our Wav2Seq (from XLS-R (0.3B)) outperforms the counterpart using an mBART-ML50N1 decoder (row 1 vs. 3). Surprisingly, we do not observe any significant performance gain with a multi-lingual pre-trained encoder (row 1 vs. 2). Again, our Wav2Seq is able to achieve strong performance without using a large mBART-ML50N1 decoder pretrained on text-based machine translation corpus (row 3 to 7). Notably, our Wav2Seq (from XLS-R (0.3B)) outperforms XLS-R (0.3B) + mBART-ML50N1 (row 2 vs. 3).

5.4. Ablation Study

We conduct an ablation study with the small model in Subsec. 5.1. Table 6 shows the results of pre-training with different types of target sequences. Deduplication plays a vital role and using BPE tokenization to convert characters into subwords further reduces WER.

6. CONCLUSION

We present Wav2Seq, a self-supervised learning framework for pre-training speech encoder-decoder models. Instead of using aligned text, we create a pseudo ASR task in which models transcribe audio inputs into pseudo subword tokens. We show that Wav2Seq closes the performance gap between encoder-decoder models and CTC models under low-resource conditions in ASR and also achieves strong performance on ST and SNER tasks.
References

[1] A. van den Oord et al., “Representation learning with contrastive predictive coding,” arXiv:1807.03748, 2018.
[2] S. Schneider et al., “wav2vec: Unsupervised pre-training for speech recognition,” in Interspeech, 2019.
[3] A. Baevski et al., “wav2vec 2.0: A framework for self-supervised learning of speech representations,” in NeurIPS, 2020.
[4] W.-N. Hsu et al., “HuBERT: How much can a bad teacher benefit asr pre-training,” in SAS-NeurIPS, 2020.
[5] V. Panayotov, G. Chen, D. Povey et al., “LibriSpeech: An asr corpus based on public domain audio books,” ICASSP, 2015.
[6] C. Wang, A. Wu, and J. Pino, “Covost 2 and massively multilingual speech-to-text translation,” arXiv:2007.10310, 2020.
[7] S.-w. Yang et al., “SUPERB: Speech processing universal performance benchmark,” in Interspeech, 2021.
[8] S. Shon et al., “SLUE: New benchmark tasks for spoken language understanding evaluation on natural speech,” in ICASSP, 2022.
[9] R. Collobert et al., “Wav2letter: an end-to-end convnet-based speech recognition system,” arXiv:1609.03193, 2016.
[10] S. Watanabe, T. Hori, S. Karita et al., “ESPNet: End-to-end speech processing toolkit,” in Interspeech, 2018.
[11] V. Pratap et al., “Wav2letter++: A fast open-source speech recognition system,” in ICASSP, 2019.
[12] W. Han et al., “ContextNet: Improving convolutional neural networks for automatic speech recognition with global context,” in Interspeech, 2020.
[13] A. Gulati et al., “Conformer: Convolution-augmented transformer for speech recognition,” in Interspeech, 2020.
[14] A. Baevski, W.-N. Hsu, Q. Xu, A. Babu, J. Gu, and M. Auli, “Data2vec: A general framework for self-supervised learning in speech, vision and language,” in ICML, 2022.
[15] A. Graves, “Sequence transduction with recurrent neural networks,” in ICML, 2012.
[16] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” NeurIPS, 2014.
[17] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, “Attention-based models for speech recognition,” in ICASSP, 2015.
[18] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, “Listen, attend and spell: A neural network for large vocabulary conversational speech recognition,” in ICASSP, 2016.
[19] C. Wang, A. Wu, J. Pino, A. Baevski, M. Auli, and A. Conneau, “Large-scale self- and semi-supervised learning for speech translation,” in Interspeech, 2021.
[20] A. Babu, C. Wang, A. Tjandra et al., “XLS-R: Self-supervised cross-lingual speech representation learning at scale,” in Interspeech, 2022.
[21] F. Wu, K. Kim, S. Watanabe et al., “Wav2seq: Pre-training speech-to-text encoder-decoder models using pseudo languages,” arXiv:2205.01086, 2022.
[22] J. Ao, Z. Zhang, L. Zhou et al., “Pre-training transformer decoder for end-to-end asr model with unpaired speech data,” in Interspeech, 2022.
[23] M. Lewis, Y. Liu, N. Goyal et al., “BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension,” in ACL, 2020.
[24] C. Raffel, N. Shazeer, A. Roberts et al., “Exploring the limits of transfer learning with a unified text-to-text transformer,” JMLR, 2019.
[25] A. Baevski, S. Schneider, and M. Auli, “vq-wav2vec: Self-supervised learning of discrete speech representations,” in ICLR, 2020.
[26] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in NAACL-HLT, 2019.
[27] J. Ao, R. Wang, L. Zhou et al., “SpeechT5: Unified-modal encoder-decoder pre-training for spoken language processing,” in ACL, 2022.
[28] K. Lakhota, E. Kharitonov, W.-N. Hsu et al., “Generative spoken language modeling from raw audio,” arXiv:2102.01192, 2021.
[29] P. Gage, “A new algorithm for data compression,” C Users Journal, 1994.
[30] J. Kahn, M. Rivi`ere, W. Zheng et al., “Libri-light: A benchmark for asr with limited or no supervision,” in ICASSP, 2020.
[31] M. Ott, S. Edunov, A. Baevski et al., “fairseq: A fast, extensible toolkit for sequence modeling,” in NAACL-HLT, 2019.
[32] D. Sculley, “Web-scale k-means clustering,” in WWW, 2010.
[33] D. Arthur and S. Vassilvitskii, “K-means++: The advantages of careful seeding,” in SODA, 2007.
[34] F. Pedregosa, G. Varoquaux, A. Gramfort et al., “Scikit-learn: Machine learning in Python,” JMLR, 2011.
[35] F. Wu, K. Kim, J. Pan et al., “Performance-efficiency trade-offs in unsupervised pre-training for speech recognition,” in ICASSP, 2022.
[36] H. Inan, K. Khosravi, and R. Socher, “Tying word vectors and word classifiers: A loss framework for language modeling,” in ICLR, 2017.
[37] O. Press and L. Wolf, “Using the output embedding to improve language models,” in EACL, 2017.
[38] S. Ghannay, A. Caubrière, Y. Estève et al., “End-to-end named entity and semantic concept extraction from speech,” in SLT, 2018.
[39] H. Yadav, S. Ghosh, Y. Yu, and R. R. Shah, “End-to-end named entity recognition from English speech,” in Interspeech, 2020.
[40] A. Pasad et al., “On the use of external data for spoken named entity recognition,” in NAACL, 2022.
[41] A. Conneau, A. Baevski, R. Collobert et al., “Unsupervised cross-lingual representation learning for speech recognition,” Interspeech, 2021.
[42] C. Wang, M. Rivière, A. Lee et al., “Voxpopuli: A large-scale multilingual speech corpus for representation learning, semi-supervised learning and interpretation,” in ACL, 2021.
[43] X. Li, C. Wang, Y. Tang et al., “Multilingual speech translation with efficient finetuning of pretrained models,” in ACL/IJCNLP, 2021.
[44] Y. Liu, J. Gu, N. Goyal et al., “Multilingual denoising pre-training for neural machine translation,” TACL, 2020.