ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-6327-7/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICASSP49357.2023.10096308

HUBERT-AGG: AGGREGATED REPRESENTATION DISTILLATION OF HIDDEN-UNIT BERT FOR ROBUST SPEECH RECOGNITION

Wei Wang, Yanmin Qian†

MoE Key Lab of Artificial Intelligence, AI Institute
X-LANCE Lab, Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, China

ABSTRACT

Self-supervised learning (SSL) has attracted widespread research interest since many successful SSL approaches such as wav2vec 2.0 and Hidden-unit BERT (HuBERT) have achieved promising results on speech-related tasks such as automatic speech recognition (ASR). However, few works have been conducted to improve the noise robustness of SSL models. In this paper, we propose HuBERT-AGG, a novel method that learns noise-invariant SSL representations for robust speech recognition by distilling aggregated layer-wise representations. Specifically, we learn an aggregator that computes the weighted sum of all hidden states of a pretrained vanilla HuBERT by fine-tuning it on a small portion of labeled data. Then a noise-robust HuBERT is trained on the simulated noisy speech by distilling from the aggregated representations and layer-wise hidden states produced by a pretrained vanilla HuBERT with parallel original speech as input. Experiments on LIBRISPEECH simulated noisy test sets show 13.1%-17.0% relative word error rate (WER) reduction with very slight degradation on the original test sets. On CHiME-4 1-channel real speech test sets, we have surpassed the best results achieved by all published fully supervised ASR models as well as other SSL approaches adopting the same data usage as ours.

Index Terms— robust speech recognition, self supervised learning, knowledge distillation, HuBERT

1. INTRODUCTION

The performance of automatic speech recognition (ASR) systems has been greatly boosted since the advance of deep neural networks and end-to-end (E2E) ASR architectures. Previous works have shown that training a neural ASR system with large-scale speech-text paired data is very effective [1, 2]. However, neural ASR systems, especially E2E models, are susceptible to the overfitting problem when the training data is limited.

Recently, a great many self-supervised learning (SSL) methods [3, 4, 5, 6, 7] have been proposed to tackle this problem by leveraging the abundant unlabeled speech to learn contextualized speech representations that benefit downstream tasks like ASR. For example, wav2vec 2.0 [3] and w2v-BERT [4] apply contrastive learning to representations in intermediate layers. Speech SimCLR [5], SimSiam [8] and c-siam [6] learn to match higher-level representations between different versions of speech input using siamese networks. HuBERT [7] iteratively optimizes the model's ability to classify randomly masked frames in speech to pseudo labels obtained by clustering Mel-frequency cepstral coefficients (MFCC) or latent embeddings from another model.

† corresponding author

Although these SSL methods have shown promising results on various ASR benchmarks, there is little work investigating their noise robustness. To improve the robustness of conventional supervised ASR models, previous works often integrate a speech enhancement (SE) module as a pre-processing front-end to suppress the noise from noisy speech [9, 10, 11, 12, 13]. However, it has been observed that the enhanced signals with the better auditory quality might not always yield better results for ASR systems [14, 15, 16] partially due to over-suppression of speech signals. Although a cascaded framework [17] was proposed to optimize SE module and ASR module with ASR objectives, such paradigms increase the complexity of noise-robust ASR systems. To combat the vulnerability of SSL models to background noise for the ASR task, wav2vec-Switch [18] feeds the wav2vec2.0 network with original-noisy speech pair and predicts the quantized representations of each other. Wav2vec-C [19] and [20] incorporate a reconstruction loss with the contrastive wav2vec2.0 loss. PASE+ [21] reconstructs distorted speech to various versions of transformed acoustic features.

In this work, we propose a novel method that builds noise-robust SSL models for the ASR task based on the HuBERT framework. Unlike previous works that train a robust SSL model from scratch, we start with a pretrained vanilla HuBERT and boost its noise robustness with limited training steps. Specifically, we first fine-tune a well-trained vanilla HuBERT with the connectionist temporal classification (CTC) criterion following the scheme applied in SUPERB [22]. The fine-tuning scheme learns a vector of weights that computes the weighted sum of multiple hidden states from the pretrained HuBERT as the final representation for the ASR task. The learned vector of weights is referred to as an aggregator. Then we train a noise-robust HuBERT on original-noisy speech pairs via knowledge distillation guided by the aggregator. Specifically, we feed the robust HuBERT with simulated noisy speech and feed a pretrained vanilla HuBERT with the corresponding original speech. The last layer of the robust HuBERT is supervised by the aggregated representation produced by the vanilla HuBERT. Meanwhile, other layers of the robust HuBERT receive supervision from the original representations produced by the corresponding layers of the pretrained HuBERT (i.e., representations produced by the last layer of the pretrained HuBERT are ignored). By doing this, the SSL representations learned by the robust HuBERT is not only noise-invariant, but also more suitable for the ASR task since the aggregator is trained on the ASR Task. Besides, the robust HuBERT is initialized with the pretrained vanilla HuBERT and enjoys fast convergence.

Our contributions can be summarized as follows: (1) we incorporate noise-invariant training into HuBERT SSL framework with knowledge distillation. (2) We guide the distillation with aggregated representations that are optimized for the ASR task. (3) We vali-

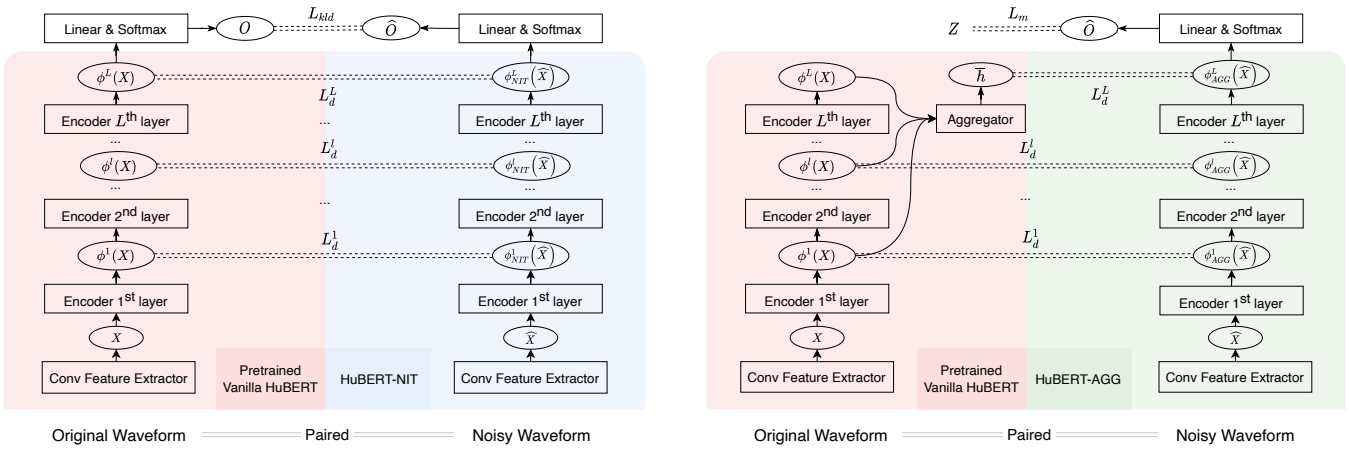

Fig. 1: **Left:** HuBERT-NIT incorporates noise-invariant training with HuBERT in a straightforward way. Distance between latent encodings as well as output distribution produced from origin-noisy speech pairs is regularized in a layer-wise manner. **Right:** Different from HuBERT-NIT, the last layer of the HuBERT-AGG encoder is supervised by the aggregated original representations which are trained to benefit the ASR task. Note that we do not apply masks to the encoded original waveforms.

date through experiments that the proposed HuBERT-AGG framework improves the noise robustness of learned representations and yields significant WER improvement for LIBRISPEECH simulated noisy speech. (4) Our approach surpassed the published best results achieved by the fully supervised ASR systems on CHiME-4 real noisy speech as well as other SSL approaches adopting the same data usage as ours.

2. METHODOLOGY

2.1. HuBERT

We first revisit HuBERT on which our method is based. HuBERT is an SSL approach that exploits an offline clustering step to provide aligned target labels for a BERT-like prediction loss [23]. HuBERT applies the prediction loss only over the masked regions, forcing the model to learn a combined acoustic and language model over the continuous inputs. The backbone used in HuBERT is a convolutional waveform encoder followed by a BERT mask predictor comprised of many identical Transformer blocks [24]. Given a speech embedding sequence encoded by the convolutional encoder $X = [x_1, \ldots, x_T]$, the mask predictor of HuBERT takes as input its masked version $\tilde{X}$ and predicts a distribution over the target codewords at each timestep t. The distribution over codewords is parameterized with

$$p(c \mid \tilde{X}, t) = \frac{\exp(\text{sim}(\phi(\tilde{X})_t \mathbf{W}, \mathbf{e}_c)/\tau)}{\sum_{c'=1}^{C} \exp(\text{sim}(\phi(\tilde{X})_t \mathbf{W}, \mathbf{e}_{c'})/\tau)} \tag{1}$$

where C is the total number of codewords and $\mathbf{e}_c$ is the embedding for codeword c. $\mathbf{W}$ is a projection matrix. $\phi(\tilde{X})_t$ denotes the output feature sequence at step t. $\text{sim}(\cdot, \cdot)$ computes the cosine similarity and τ scales the logit.

Denoting discrete target sequence for X as $Z = [z_1, z_2, \ldots, z_T]$, where $z_t \in [C]$ is a C-class categorical variable, the prediction loss for masked regions in HuBERT is formulated as

$$L_m(\tilde{X}) = \sum_{t \in M} \log p\left(z_t \mid \tilde{X}, t\right) \tag{2}$$

where $M \subset \{1 \ldots T\}$ denotes the masked timesteps for X.

HuBERT refines the assignment of clustered codewords iteratively. That is, in the first iteration, the codewords are assigned by clustering the MFCC features of the training data; In subsequent iterations, new codewords are assigned by clustering the latent representations produced by the model trained in the previous iteration.

2.2. HuBERT-NIT

We first introduce a straightforward way to incorporate noise-invariant training into HuBERT (HuBERT-NIT) with original-noisy speech pairs as illustrated in the left part of Fig 1. Denote the encoded original speech as X, the masked version of the corresponding encoded noisy speech as $\hat{X}$. We adopt a regularization term proposed in [25] that penalizes L_2 and the cosine distance between the encodings produced by the pretrained vanilla HuBERT using original speech encodings $\phi^l(X)$ and HuBERT-NIT using masked noisy speech encodings $\phi^l_{NIT}(\hat{X})$ at layer l. The regularization term is applied in a layer-wise manner:

$$\begin{aligned} L_d\left(\phi^l(X), \phi^l_{\text{NIT}}(\hat{X})\right) = &\left\|\phi^l(X) - \phi^l_{\text{NIT}}(\hat{X})\right\|^2 \\ &- \frac{\phi^l(X) \cdot \phi^l_{\text{NIT}}(X)}{\|\phi^l(X)\| \cdot \left\|\phi^l_{\text{NIT}}\left(\hat{X}\right)\right\|} \end{aligned} \tag{3}$$

$$L_{d-NIT} = \sum_{l=1}^{L} L_d\left(\phi^l(X), \phi^l_{\text{NIT}}(\hat{X})\right) \tag{4}$$

where L is the number of encoder layers. Note that the parameters of the pretrained vanilla HuBERT are frozen when training $\text{HuBERT}_{\text{NIT}}$. The output distribution from $\text{HuBERT}_{\text{NIT}}$ $\hat{O}$ is also supervised by that from the pretrained vanilla HuBERT O via Kullback–Leibler Divergence (KLD):

$$L_{\text{KLD}} = \sum_{t=1}^{T} O_t\left(\log O_t - \log \hat{O}_t\right) \tag{5}$$

The final loss for training $\text{HuBERT}_{\text{NIT}}$ is a weighted sum of L_{d-NIT} and L_{kld}:

$$L_{NIT} = \lambda_1 L_{d-NIT} + \lambda_2 L_{\text{KLD}} \tag{6}$$

2.3. HuBERT-AGG

Compared to HuBERT-NIT, HuBERT-AGG improves the learned noise-invariant representations by distilling aggregated representations that are optimized exclusively for the ASR task as illustrated in the right part of Fig 1, where the aggregator is effectively a frozen well-trained vector used to compute the weighted-sum representations of all encoder layers in the pretrained vanilla HuBERT. Specifically, the aggregator used for distillation has been trained on labelled speech data with CTC criterion following the fine-tuning scheme used in [22] that feeds the aggregated representations to a linear projection layer of which the input is mapped to output tokens (characters). In this way, the trained aggregator is capable of extracting aggregated representations that are beneficial for the ASR task. Denote the representations produced by L layers of the pretrained vanilla HuBERT as $\mathbf{H} = [\mathbf{h_1}, \mathbf{h_2}, \ldots, \mathbf{h_L}]$. Denote the aggregator as a vector $\mathbf{a} = [a_1, a_2, \ldots, a_L]$ where a_l is the weight corresponding to $\mathbf{h_l}$ for weighted-sum. The aggregated representation can be formulated as $\bar{\mathbf{h}} = \mathbf{a}\mathbf{H}^T$. We modify L_{d-NIT} and use $\bar{\mathbf{h}}$ as the target for the last encoder layer of HuBERT-AGG:

$$L_{d-AGG} = \sum_{l=1}^{L-1} L_d\left(\phi^l(X), \phi^l_{\text{AGG}}(\hat{X})\right) + \left\|\bar{\mathbf{h}} - \phi^l_{\text{AGG}}(\hat{X})\right\|^2 - \frac{\bar{\mathbf{h}} \cdot \phi^L_{\text{AGG}}(X)}{\|\bar{\mathbf{h}}\| \cdot \left\|\phi^L_{\text{AGG}}\left(\hat{X}\right)\right\|} \tag{7}$$

Besides, we replace the L_{kld} with the original HuBERT loss in Eq (2) since the last layer of HuBERT-AGG is supervised by aggregated representations rather than the encodings from the last layer of the pretrained vanilla HuBERT. The final loss for training HuBERT-AGG is a weighted sum of L_{d-AGG} and L_m:

$$L_{AGG} = \lambda_1 L_{d-AGG} + \lambda_2 L_m \tag{8}$$

3. EXPERIMENTAL SETUP

3.1. Datasets

We validate the effectiveness of our proposed method with both simulated and real-world noisy data. The data preparation involves 4 well-known corpora in ASR: LIBRILIGHT [26], LibriSpeech [27], MUSAN [28] and the 1-channel track of CHiME-4 [29]. The training procedure is comprised of 3 stages, and we introduce data usage for each stage: (1) Labelled data used to train the representation aggregator is denoted as D_{AGG} (For HuBERT-AGG only). (2) Unlabelled speech data used to pretrain HuBERT-NIT and HuBERT-AGG is denoted as D_P. (3) Labelled data used to fine-tune HuBERT-NIT and HuBERT-AGG is denoted as D_F . We investigate the impact of using different D_{AGG} and specify the choice for D_{AGG} later in each experiment. D_P is synthesized by mixing full 960 hours of LIBRISPEECH with noise randomly sampled from *music* and *noise* categories of MUSAN [28] and SNRs uniformly sampled between 5 to 10 dB. *speech* category in MUSAN is not used during training to avoid confusion with actual speech when learning speech representations. We fine-tune models on different D_F for testing on simulated noisy data and real noisy data.

For testing on simulated noisy data, the original LIBRISPEECH `train-clean-100` partition is used for D_F. We prepare original `test-clean` and `test-other` for testing. Besides, following [18], by mixing the original test sets with different categories in MUSAN we synthesize several simulated noisy test sets. The SNRs of all simulated test sets are uniformly sampled from 5 to 10 dB.

For testing on real noisy data, all CHiME-4 data are used for D_F excluding the second channel due to its low quality. We adopt the official CHiME-4 1-channel real dev and eval sets for testing.

3.2. Aggregator Training

We train the aggregator by fine-tuning the released official checkpoint for HuBERT BASE [1], which was pretrained on the full 960-hour data in LIBRISPEECH. 5 different partitions for D_{AGG} are considered: Libri-light (LL) 10-minute, 1-hour, 10-hour splits, LibriSpeech (LS) 100-hour (`train-clean-100`) and CHiME-4 clean speech. For the first 5 splits, we follow the fine-tuning setups in [3]. For CHiME-4 clean speech, we use the same setup as LS 100-hour split since they are comparable in size. All encoder layers are frozen during fine-tuning. Layerdrop is set to zero to make sure all layers can produce speech representations for aggregation.

3.3. Model Pre-training

We carry out model pretraining with the FAIRSEQ [30] toolkit. We adopt the same architecture as specified in [7]: 12 transformer blocks with hidden dimensions 768 and 8 heads. For faster convergence, all models are initialized with the same released HuBERT checkpoint as used in aggregator training. We further apply k-means clustering with 500 clusters on the latent features extracted from the 9th layer official HuBERT BASE model since it has the highest phone purity as shown in [7]. The same configuration for training the second iteration of HuBERT BASE is adopted except that all models are pretrained for only 50k steps unless specified otherwise. We disable layerdrop, dropout and masking for pretrained vanilla HuBERT in Fig 1 to produce original speech representation with better quality for distillation. Note that the target sequence Z for the noisy speech input is obtained with codewords from the corresponding original speech. λ_1 and λ_2 in Eq (6) is set to 1 and 10. λ_1 and λ_2 in Eq (8) is set to 1 and 1000. The values of these hyper-parameters are determined by the performance on original LIBRISPEECH development sets after fine-tuning on the Libri-light 10-hour split.

3.4. Model Fine-tuning

Since the CHiME-4 training set (92.28 hours) and the LIBRISPEECH 100-hour split are comparable in size, we follow the `base_100h` setup in the wav2vec 2.0 for both experiments.

3.5. Decoding and Language Models

We use the official LIBRISPEECH 4-gram language model[2] and follow the configuration introduced in [3] for decoding on LIBRISPEECH test sets. For decoding on CHiME-4 test sets, an LSTM-based word-level language model with a vocabulary size of 65,000 is trained on the text portion of the WSJ [31] corpus using the Espresso [32] recipe, and then the same decoding strategy was performed. Note that the hyper-parameters for decoding are separately tuned for LIBRISPEECH and CHiME-4 test sets.

4. RESULTS AND ANALYSIS

4.1. Results on Simulated Noisy Speech

Table 1 presents the word error rate (WER, %) results on original and simulated test sets based on LIBRISPEECH. The *music* + *noise* test sets are synthesized by mixing original test sets with *music* and *noise* categories in MUSAN, which are in a matched condition with the pretraining stage. Likewise, results on *speech* test sets show

[1] https://dl.fbaipublicfiles.com/hubert/hubert_base_ls960.pt

[2] https://www.openslr.org/resources/11/

Table 1: WER(%) comparison of different pretrained models on LIBRISPEECH original and artificial test sets.

System	original clean	original other	*music* + *noise* clean	*music* + *noise* other	*speech* clean	*speech* other
HuBERT	**3.4**	**8.2**	8.7	21.2	32.9	51.6
+ MUSAN	4.1	9.0	5.4	14.1	15.3	36.1
HuBERT-NIT	3.7	8.7	4.6	11.7	13.3	30.8
HuBERT-AGG	3.5	8.5	**4.3**	**11.6**	**12.4**	**30.3**

mismatched testing conditions. In the first row, it is not surprising that the HuBERT pretrained on the original LIBRISPEECH data shows severe degradation on noisy test sets. Besides, degradation on *speech* test sets is significantly more severe than on *music* + *noise* test tests. This can be ascribed to the fact that background speech is more likely to be confused with the target speech than music and noise. In the second row, the HuBERT is pretrained with augmented data D_P as described in section 3.1. Performance on noisy test sets is improved at the cost of degradation on original test sets. In the third row, by simply penalizing the distance between representations produced by pretrained vanilla HuBERT and HuBERT-NIT, we see consistent improvements on all test sets. The relative WER reduction on *speech* mismatched testing condition (13.1% / 14.7%) is slightly smaller than on *music* + *noise* matched testing conditions (i.e. *music* + *noise*: 14.8% / 17.0%). In the last row, we train the aggregator on the Libri-light 10-hour partition as described in section 2.3. The aggregated speech representations are more beneficial to the ASR task and yield further improvement on all test sets.

4.2. Investigating Choices for D_{AGG}

Table 2: WER(%) comparison of proposed HuBERT-AGG on artificial noisy test sets. Different D_{AGG} are investigated.

System	Aggregator	*music* + *noise* clean	*music* + *noise* other	*speech* clean	*speech* other
HuBERT-AGG	LS 100 hr	**4.3**	**11.5**	12.5	30.4
	LL 10 hr	**4.3**	11.6	12.4	30.3
	LL 1 hr	**4.3**	11.7	12.4	30.2
	LL 10 min	4.4	11.8	**12.3**	**30.1**
	CHiME-4	4.5	12.0	12.6	30.2

In Table 2 we investigate different choices for D_{AGG}. 5 different partitions are considered as described in Section 3.2. The LS and LL partitions are in the same domain as the test sets and the CHiME-4 partition is in a different domain. Surprisingly, the amount of data used to train the aggregator does not have an apparent impact on the final ASR performance. Training the aggregator with data from a different domain (the last row) yields slight performance degradation on most test sets. We believe that the benefits of introducing the aggregator are brought by the better correlation between the aggregated speech representations and the downstream ASR task.

4.3. Results on Real-World Noisy Speech

In this section, our proposed methods are evaluated on the CHiME-4 1-channel real noisy test sets as shown in Table 3. We include the recent best results achieved by supervised learning with [33] and without [34] an enhancement front-end. We also include recently published SSL results [18, 20], which are constructed with the same data usage as in our experiments (i.e. unlabeled LIBRISPEECH 960-hour data and labeled CHiME-4 1-channel data). To perform a fair comparison with [18] and [20] which do not use labeled data in LIBRISPEECH, the aggregator for HuBERT-AGG in these experiments is trained with CHiME-4 clean data in the simulated partition.

By training on simulated noisy data synthesized with LIBRISPEECH and MUSAN (the sixth row), the results on CHiME-4 test sets surpass the results obtained with the HuBERT pretrained on original data (the fifth row) by relatively 18.2% / 23.2% WER reduction. These results show that the HuBERT pretrained with simulated noisy data generalizes well to real-world noisy data. No further improvement is observed with HuBERT-NIT, which is probably caused by the gap between LIBRISPEECH and CHiME-4 data.

In previous experiments, we trained HuBERT-AGG for 50k steps due to limited computational resources. We now investigate how the number of training steps affects the performance of HuBERT-AGG. As shown in the last 4 rows in Table 3, increasing training steps from 25k to 150k yields continuous improvement. With 150k training steps, the proposed HuBERT-AGG achieves obviously better results than all fully supervised models as well as the other SSL models with the same data usage. Note that [35] (2022) achieves the state-of-the-art results on CHiME-4 1-channel real data test sets (2.0% / 3.9%) with an SSL model, but it is pretrained on a significantly larger corpus (94k hours) and thus not compared here.

Table 3: WER(%) of different systems on CHiME-4 real test sets, including published best fully supervised ASR and SSL models.

System	Train Steps	CHiME-4 REAL dev	CHiME-4 REAL eval
Wang et al. [33] (2020)	NA	3.5	6.8
Yang et al. [34] (2022)		3.4	6.3
wav2vec-switch [18] (2022)		3.5	6.6
wav2vec (recons) [20] (2022)		5.0	9.0
HuBERT	50k	4.4	8.6
+ MUSAN	50k	3.6	6.6
HuBERT-NIT	50k	3.5	6.8
HuBERT-AGG (CHiME-4 Aggregator)	25k	3.8	7.2
	50k	3.3	6.1
	100k	3.3	5.9
	150k	**3.2**	**5.8**

5. CONCLUSIONS

In this paper, we introduce HuBERT-AGG, a novel method for improving the noise robustness of the popular HuBERT SSL approach for ASR task. We apply noise-invariant training based on encoding distance regularization between original-noisy speech pairs to boost the noise robustness. Moreover, we introduce a representation aggregator that computes the weighted-sum representations optimized for the ASR task and use them to teach the last layer of HuBERT-AGG. By doing this, the HuBERT-AGG model learns noise-invariant representations that also fit the ASR task well. Experiments on simulated noisy LIBRISPEECH test sets show 13.1%-17.0% relative WER reduction compared to a straightforward data augmentation SSL method, and also still preserve similar performance on the original test sets. On CHiME-4 real-world noisy tests sets, the proposed HuBERT-AGG surpasses previous best results achieved by fully supervised learning methods as well as other SSL approaches with the same data usage as ours.

6. ACKNOWLEDGEMENTS

This work was supported in part by China NSFC projects under Grants 62122050 and 62071288, in part by Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0102, and in part by Shanghai Science and Technology Committee under Grant No. 21511101100.

7. REFERENCES

[1] T. Likhomanenko, Q. Xu *et al.*, "Rethinking evaluation in ASR: Are our models robust enough?" in *INTERSPEECH*, 2021, pp. 311–315.

[2] W. Chan, D. S. Park *et al.*, "SpeechStew: Simply mix all available speech recognition data to train one large neural network," in *Workshop on Machine Learning in Speech and Language Processing (MLSLP)*, 2021.

[3] A. Baevski, Y. Zhou *et al.*, "wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Advances in Neural Information Processing Systems*, vol. 33, 2020, pp. 12 449–12 460.

[4] Y.-A. Chung, Y. Zhang *et al.*, "W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training," in *IEEE ASRU*, 2021, pp. 244–250.

[5] D. Jiang, W. Li *et al.*, "Speech SimCLR: Combining contrastive and reconstruction objective for self-supervised speech representation learning," in *INTERSPEECH*, 2021, pp. 1544–1548.

[6] S. Khorram, J. Kim *et al.*, "Contrastive siamese network for semi-supervised speech recognition," in *IEEE ICASSP*, 2022, pp. 7207–7211.

[7] W.-N. Hsu, B. Bolte *et al.*, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM TASLP*, vol. 29, pp. 3451–3460, 2021.

[8] T. Chen, S. Kornblith *et al.*, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning (ICML)*, 2020, pp. 1597–1607.

[9] Y. Xu, J. Du *et al.*, "An experimental study on speech enhancement based on deep neural networks," *IEEE Signal Processing Letters*, vol. 21, no. 1, pp. 65–68, 2014.

[10] S. Pascual, A. Bonafonte, and J. Serrà, "SEGAN: Speech enhancement generative adversarial network," in *INTERSPEECH*, 2017.

[11] K. Tan and D. Wang, "A convolutional recurrent neural network for real-time speech enhancement," in *INTERSPEECH*, 2018.

[12] Y. Hu, Y. Liu *et al.*, "DCCRN: Deep complex convolution recurrent network for phase-aware speech enhancement," in *INTERSPEECH*, 2020, pp. 2472–2476.

[13] D. Yin, C. Luo *et al.*, "PHASEN: A phase-and-harmonics-aware speech enhancement network," in *AAAI*, 2020.

[14] P. C. Loizou and G. Kim, "Reasons why current speech-enhancement algorithms do not improve speech intelligibility and suggested solutions," *IEEE TASLP*, vol. 19, no. 1, pp. 47–56, 2011.

[15] P. Wang, K. Tan *et al.*, "Bridging the gap between monaural speech enhancement and recognition with distortion-independent acoustic modeling," *IEEE/ACM TASLP*, vol. 28, pp. 39–48, 2019.

[16] K. Iwamoto, T. Ochiai *et al.*, "How bad are artifacts?: Analyzing the impact of speech enhancement errors on ASR," in *INTERSPEECH*, 2022, pp. 5418–5422.

[17] A. S. Subramanian, X. Wang *et al.*, "Speech enhancement using end-to-end speech recognition objectives," in *IEEE WASPAA*, 2019, pp. 234–238.

[18] Y. Wang, J. Li *et al.*, "Wav2vec-Switch: Contrastive learning from original-noisy speech pairs for robust speech recognition," in *IEEE ICASSP*, 2022, pp. 7097–7101.

[19] S. Sadhu, D. He *et al.*, "wav2vec-C: A self-supervised model for speech representation learning," in *INTERSPEECH*, 2021, pp. 711–715.

[20] H. Wang, Y. Qian *et al.*, "Improving noise robustness of contrastive speech representation learning with speech reconstruction," in *IEEE ICASSP*, 2022, pp. 6062–6066.

[21] M. Ravanelli, J. Zhong *et al.*, "Multi-task self-supervised learning for robust speech recognition," in *IEEE ICASSP*, 2020, pp. 6989–6993.

[22] S. wen Yang, P.-H. Chi *et al.*, "SUPERB: Speech processing universal performance benchmark," in *INTERSPEECH*, 2021, pp. 1194–1198.

[23] J. Devlin, M.-W. Chang *et al.*, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[24] A. Vaswani, N. Shazeer *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[25] D. Liang, Z. Huang, and Z. C. Lipton, "Learning noise-invariant representations for robust speech recognition," in *IEEE SLT*, 2018, pp. 56–63.

[26] J. Kahn, M. Rivière *et al.*, "Libri-light: A benchmark for ASR with limited or no supervision," in *IEEE ICASSP*, 2020, pp. 7669–7673.

[27] V. Panayotov, G. Chen *et al.*, "Librispeech: An ASR corpus based on public domain audio books," in *IEEE ICASSP*, 2015, pp. 5206–5210.

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

[29] E. Vincent, S. Watanabe *et al.*, "An analysis of environment, microphone and data simulation mismatches in robust speech recognition," *Computer Speech and Language*, vol. 46, pp. 535–557, 2017.

[30] M. Ott, S. Edunov *et al.*, "fairseq: A fast, extensible toolkit for sequence modeling," in *Proc. NAACL-HLT: Demonstrations*, 2019, pp. 48–53.

[31] D. B. Paul and J. M. Baker, "The design for the Wall Street Journal-based CSR corpus," in *Speech and Natural Language: Proceedings of a Workshop Held at Harriman, New York*, 1992.

[32] Y. Wang, T. Chen *et al.*, "Espresso: A fast end-to-end neural speech recognition toolkit," in *IEEE ASRU*, 2019, pp. 136–143.

[33] Z.-Q. Wang, P. Wang, and D. Wang, "Complex spectral mapping for single- and multi-channel speech enhancement and robust ASR," *IEEE/ACM TASLP*, vol. 28, pp. 1778–1787, 2020.

[34] Y. Yang, P. Wang, and D. Wang, "A conformer based acoustic model for robust automatic speech recognition," *arXiv preprint arXiv:2203.00725*, 2022.

[35] X. Chang, T. Maekaku *et al.*, "End-to-end integration of speech recognition, speech enhancement, and self-supervised learning representation," in *INTERSPEECH*, 2022, pp. 3819–3823.