AUDIO ALBERT: A LITE BERT FOR SELF-SUPERVISED LEARNING OF AUDIO REPRESENTATION

Po-Han Chi[1], *Pei-Hung Chung*[1], *Tsung-Han Wu*[1], *Chun-Cheng Hsieh*[1], *Yen-Hao Chen*[1], *Shang-Wen Li*[2],* *Hung-yi Lee*[1]

[1]College of Electrical Engineering and Computer Science, National Taiwan University
[2]Amazon AI

ABSTRACT

Self-supervised speech models are powerful speech representation extractors for downstream applications. Recently, larger models have been utilized in acoustic model training to achieve better performance. We propose Audio ALBERT, a lite version of the self-supervised speech representation model. We apply the lightweight representation extractor to two downstream tasks, speaker classification and phoneme classification. We show that Audio ALBERT achieves performance comparable with massive pre-trained networks in the downstream tasks while having 91% fewer parameters. Moreover, we design probing models to measure how much the latent representations can encode the speaker's and phoneme's information. We find that the representations encoded in internal layers of Audio ALBERT contain more information for both phoneme and speaker than the last layer, which is generally used for downstream tasks. Our findings provide a new avenue for using self-supervised networks to achieve better performance and efficiency.

Index Terms— Self-supervised learning, Weight sharing, Network compression, transformer, Speech representation learning

1. INTRODUCTION

Recently, pre-trained models [1, 2, 3, 4], especially BERT, dominate Natural Language Processing (NLP) world. The models learn powerful and universal representation by utilizing self-supervised learning at the pre-training stage to encode the contextual information. The representation is beneficial to performance, especially when the data of the downstream task is limited. As of late, BERT-like models are also applied to the speech processing domain. The pre-trained model learns the robust speech representations for speech processing tasks, such as Automatic Speech Recognition (ASR) and speaker recognition, with the self-supervised learning. approaches [5, 6, 7, 8, 9].

However, since the size of these BERT-like pre-trained models is usually prohibitively large, these models require a significant amount of memory for computation, even at the fine-tuning stage. The requirement hinders the application of pre-trained models from different downstream tasks.

ALBERT [10] addresses the challenge of efficiency. ALBERT is a lite version of BERT for text by sharing one layer parameters across all layers and factorizing the embedding matrix to reduce most parameters. Although the number of parameters is reduced, the representations learned in ALBERT are still robust and task agnostic, such that ALBERT can achieve similar performance in the same downstream tasks comparing to BERT.

*work done before joining Amazon

In this paper, we first examine the knowledge encoding in each layer of Mockingjay, a pre-trained model utilizing BERT architecture to encode speech information. We found the learned parameters are redundant across layers. Thus, we bring the idea of sharing parameters from ALBERT to the speech processing domain and propose a novel self-supervised model, Audio ALBERT (AALBERT), for parameter-efficient representation learning.

We show that AALBERT yields comparable performance to other pre-trained models in downstream tasks, but with much smaller networks. To understand how to use the pre-trained networks properly in downstream tasks, we also analyze representations extracted from different layers of AALBERT. We use a simple classifier to probe each layer, and we find that the representations of the intermediate layers contain more phonetic and speaker information than that of the last layer. The finding indicates that the representations from the last layer fit the pre-training task too much, and the intermediate layers may be more suitable for adapting to downstream tasks. To our best knowledge, this is the first study to bring the idea of model compression in ALBERT to speech processing, to show the benefits in the efficiency of the novel architecture, AALBERT, for speech-related tasks, and to analyze learned latent representations for better usage of pre-trained networks in downstream tasks. The code will be available soon (`https://github.com/pohanchi/AALBERT`)

2. RELATED WORK

2.1. Self-supervised learning representation

In recent years, works related to self-supervised learning spring up in Computer Vision (CV), NLP, speech processing, etc. In CV, some works [11, 12] incorporate contrastive objective and self-supervised learning for learning visual representation. Self-supervised learning is also utilized to learn language representations for NLP tasks. ELMo [2] is the first work introducing the concept of contextualized embeddings and the weighted sum application. BERT [1] further presents the concept of Masked Language Model (MLM). Deep transformer encoder architecture is trained with MLM to reconstruct the masked input sequences in the pre-training stage. The resulting networks show substantial performance gain in downstream NLP tasks. XLNet [13], introduces the Permutation Language Model and outperforms both autoregressive models and MLM. However, Roberta [14], achieves performance comparable with XLNet by training with more data, larger batch size, and the better hyperparameter settings. For parameter efficiency, ALBERT [10] is proposed to reduce the model size without losing performances in NLP tasks compared to BERT. Self-supervised learning is also gaining attention in the speech field. Contrastive Predictive Cod-

ing (CPC) [15] incorporates contrastive objective in self-supervised learning to learn powerful representations for speech processing tasks. Autoregressive Predictive Coding (APC) [16] utilizes the idea of an autoregressive model from ELMo to learn stronger speech representations. Inspired by MLM, Mockingjay [5] masks frame from input acoustic feature and pre-trains the networks to reconstruct the corresponding linear spectrogram or mel spectrogram in the pre-training stage. Similarly, Masked Predictive Coding (MPC) [6] uses the idea of MLM to pre-train a model for speech recognition. Speech-XLNet [17] is the audio version of XLNet. vq-wav2vec [18] incorporates vector quantization and BERT to improve the performance on downstream tasks. Finally, DeCoAR [7], a model built with a deep LSTM module, adopts pre-training tasks similar to Mockingjay and MPC and yields significant performance gain in speech recognition. All of these pre-trained networks are large in model size and focus on improving performance with more parameters or pre-training data. To make the models more compact for training and deployment, we build a lite version of a pre-trained network that yields comparable performance with fewer parameters and memory footprint.

2.2. Weight sharing

The previous work [19] ties the input and output embeddings to reduce parameters without harming the performance. Tong Xiao et al. proposes a method [20], which reuses the attention weights of previous layers in the adjacent layers on the transformer model for faster inference and keeps performance in neural machine translation. Some works build compact transformer models [21, 10], which apply weight sharing mechanisms across layers to reduce parameters and achieve comparable performance in their tasks. Dehghani et al. proposed Universal Transformer [22], which utilizes the benefit of the transformer and recurrent neural network, and it also incorporates weight sharing across layers to reduce a great number of parameters but keep performance in the different tasks. In general, weight sharing mechanism across layers can be viewed as an RNN applied in the direction of the layer-axis. To sum up, weight sharing can not only bring faster inference and training speed but keep similar performance in previous works. It is a kind of network-compression mechanism to reduce parameters heavily in transformer models.

2.3. Probing task

Probing is a technique to measure whether the encoder embeds specific information in representation [23, 24, 25]. The probing is done by extracting representation to be examined, building a simple classifier based on the representation for a downstream probing task, and measuring the classifier's performance. Synthesizing audio from the ASR hidden state is also proposed [24], as another way of probing.

3. METHOD

3.1. Mockingjay

Mockingjay [5] is a pre-trained model that utilizes the architecture of BERT model. In pre-training, Mockingjay takes a masked spectrogram as the input to reconstruct the original one. There are three common model architectures for Mockingjay by adopting 3, 6, and 12 layers of transformer encoders (denoted as Mockingjay-3L, Mockingjay-6L, and Mockingjay-12L respectively). The previous study [5] showed that the representation of Mockingjay possesses both rich phonetic and speaker information.

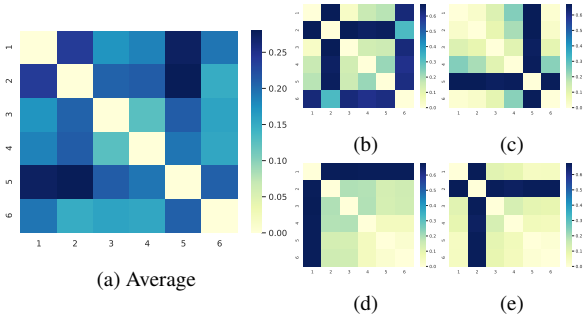


Fig. 1: The JS divergence of attention distribution between different layers in Mockingjay-6L. (a) is the average case, while (b)(c)(d)(e) represent different attention heads.

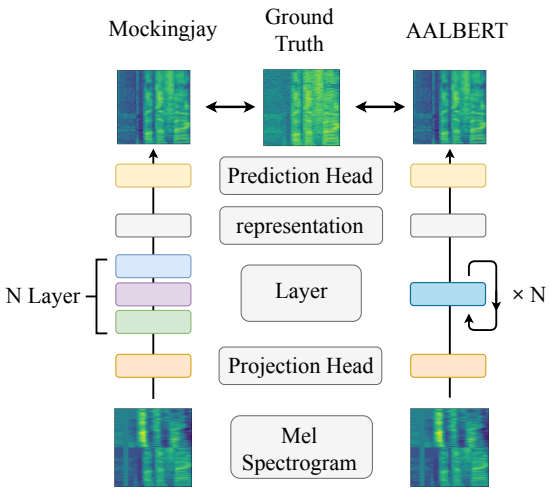


Fig. 2: Difference between Mockingjay and AALBERT

We further investigate the parameter usage in Mockingjay (due to space limitations, we only show results for the most commonly used architecture, Mockingjay-6L. Our findings here also apply to the other two variations). Inspired by Fast Transformer [20] Experiments, we use the Jensen-Shannon (JS) divergence to evaluate the difference between the attention distribution of each transformer encoder layer. For the multi-head attention, we calculate the JS divergence within each head, then average them to obtain the JS divergence between every layer. Figure 1a shows the JS divergence of attention distribution between layers in Mockingjay-6L. Here we can see that JS divergence between layers in Mockingjay-6L is significant. Furthermore, we randomly pick one attention head every layer and do the same experiment. We show them in Fig 1b, Fig 1c, Fig 1d, and Fig 1e. Although some attention heads are very different from those in other layers (dark blue cells), most layers are similar (light blue cells). The result shows that for a specific attention head in Mockingjay-6L, there is usually some similar attention distribution over different layers.

Although the parameters of each layer are different, they still generate similar attention distribution. This phenomenon indicates the existence of redundancy in parameter usage and the possibility to compress models via weight-sharing across layers without sacrificing model expressiveness.

Table 1: pre-trained Models

Model	Layer	Params	Param Sharing
AALBERT-12L	12	7.4M	True
AALBERT-6L	6	7.4M	True
AALBERT-3L	3	7.4M	True
Mockingjay-12L	12	84.3M	False
Mockingjay-6L	6	44.4M	False
Mockingjay-3L	3	21.6M	False

3.2. AALBERT

We propose AALBERT, or audio ALBERT, for a more compact pre-trained network. Similar to Mockingjay, AALBERT also takes mel-spectrogram as the input acoustic features. We mask the input features with zero and pre-train the network to reconstruct the corresponding log-linear spectrogram after applying Cepstral Mean and Variance (cmvn) from the masked input. We apply the masking to features of each utterance by first downsampling one out of every three frames and then randomly selecting 15% of the resulting frames for adding noises. The noises are introduced as the following. We zero out 80% of the selected frames, replace the frames with other frames randomly sampled from the same utterance with 10% probability, and keep the original frames for the remaining cases.

As compared to Mockingjay, we introduce weight tying for reducing parameters. As visualized in Fig 2, both Mockingjay and AALBERT are built with the architecture of the Transformer encoder, where each layer of the encoder consists of components including self-attention, feed-forward, and layer normalization. However, in AALBERT, parameters of each component are shared over all the layers. To illustrate the gain in efficiency, in Table 1, we compare the numbers of parameters for all pre-trained models studied in this paper. As we can see, AALBERT requires much fewer parameters than Mockingjay for the same depth of Transformer. All our experiments are modified from Mockingjay Large model, which is 12 layers and utilizes linear spectrogram reconstruction as a pre-training task. We do not use the Mockingjay base model because we want to compress the size of the original model, and the base model is only 3 number of layers, which limit the space of compression.

3.3. Application to downstream tasks

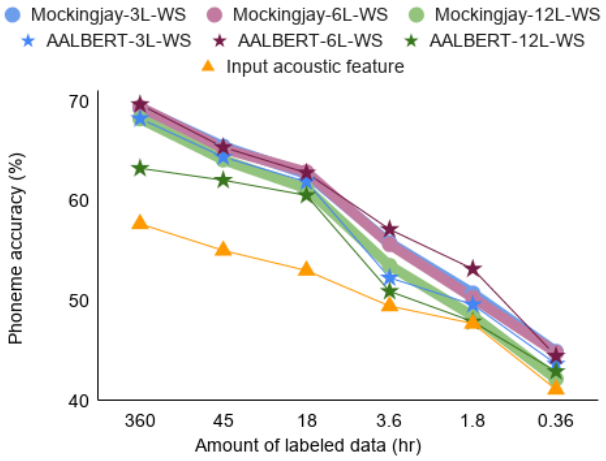
We investigate two popular ways of applying pre-trained networks to downstream tasks: feature extraction and fine-tuning.

3.3.1. Feature extraction

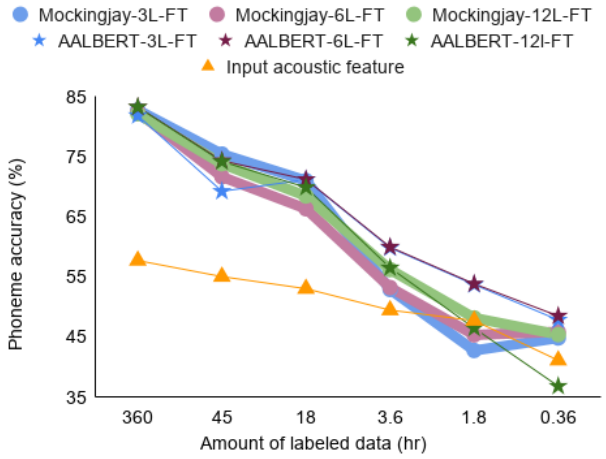
In feature extraction, all parameters in the pre-trained models are frozen when training on the downstream tasks. We utilize the representations extracted from the pre-trained model as fixed features and feed them into a simple, trainable layer. Following the typical setting, we use the representations of the last layer as the features. We also investigate a weighted sum approach proposed by ELMo [2] to fuse representations from various layers rather than the last one in the pre-trained networks and learn the weights along with the prediction layer from downstream tasks.

3.3.2. Fine-tuning

As for fine-tuning, we also build the classifier with a pre-trained network followed by a simple prediction layer. However, in fine-tuning, the parameters of the entire model are further trained on the downstream tasks. This technique boosts the classifier's performance, especially on difficult downstream tasks such as phoneme classification, but requires longer training time, more task-specific parameters, and a larger memory footprint.



(a) Feature-extraction case



(b) Fine-tuning case

Fig. 3: Phoneme classification accuracy vs amount of labeled data. 3L, 6L, 12L: the number of layers, FT: fine-tune, WS: weighted sum, Input acoustic feature: input acoustic feature as baseline.

3.4. Probing

We also propose probing tasks to understand how knowledge is encoded in pre-trained networks. For the probing tasks, we built classifiers with three different prediction layers: linear, one fully-connected, and two fully-connected layers. We explore several variations of the prediction layers to mitigate the possible bias introduced by network architectures. The prediction layers take representations from each layer of a pre-trained network as the input features and are trained on downstream tasks with the pre-trained model frozen. With the probing, we measure the information richness for each layer's representation based on the classifiers' performance.

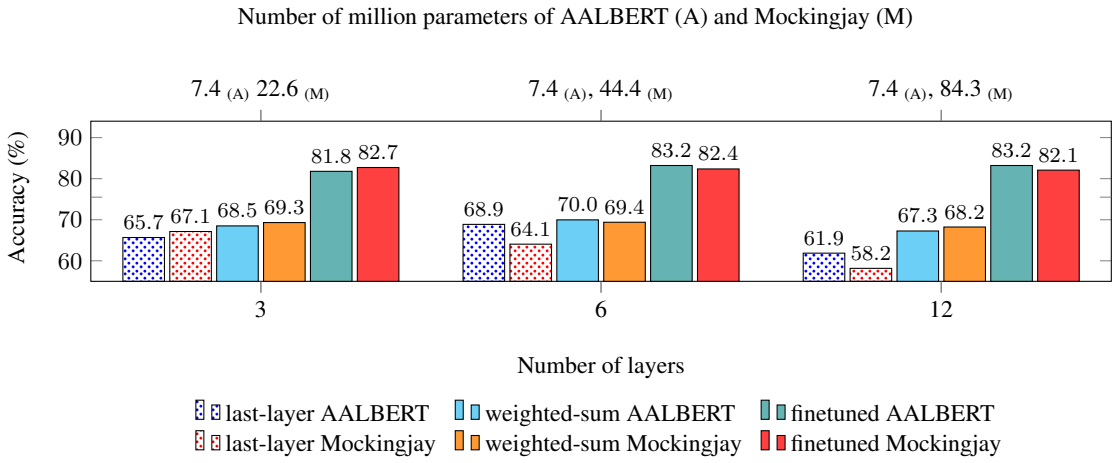


Fig. 4: Phoneme classification accuracy

Table 2: Hyperparameter for different downstream tasks, LR: Learning rate, SPK: speaker, PH: phoneme

Downstream	# PH / SPK	Detail	LR
Phoneme classification	72	weighted-sum	1e-3
	72	fine-tune	1e-4
Utterance-level speaker classification	921	weighted-sum	1e-3
	251	weighted-sum	1e-3

Such analysis allows us to interpret how information is encoded in the pre-trained networks and provides a new avenue to achieve better performance by using the networks more efficiently.

4. EXPERIMENT RESULTS AND DISCUSSIONS

4.1. Experimental setup

We evaluate the pre-trained networks with one phoneme classification task and three speaker classification tasks as the downstream tasks. In our experiment, we use a 160-dimension acoustic feature, i.e., an 80-dimension log mel-spectrogram and its delta and apply Cepstral Mean and Variance normalization (cmvn), as the input for the pre-trained networks. At the pre-training stage, we train our models with learning rate 5e-5, batch size 50, and AdamW optimizer [26] for 500k steps. The models are pre-trained on a single NVIDIA Tesla V100 32GB. We apply different hyperparameters to train classifiers for each downstream task and show the detailed settings in Table 2.

We utilize LibriSpeech [27] for our experiments. LibriSpeech contains three subsets with 500, 360, and 100 hours of speech (denoted as train-other-500, train-clean-360, and train-clean-100) and the transcription and speaker labels. We pre-train our networks on the train-clean-360 set without using any annotation. The train-clean-360 and train-clean-100 sets are used as downstream tasks for evaluating performance in phoneme and speaker classification. The two sets are further split into training, development, and test subset in the ration of 8:1:1 for our experiment. To obtain frame-level phoneme labels to benchmark phoneme classification results, we adopt Montreal Forced Aligner [28] to force align transcription to phoneme sequences containing 72 phoneme classes.

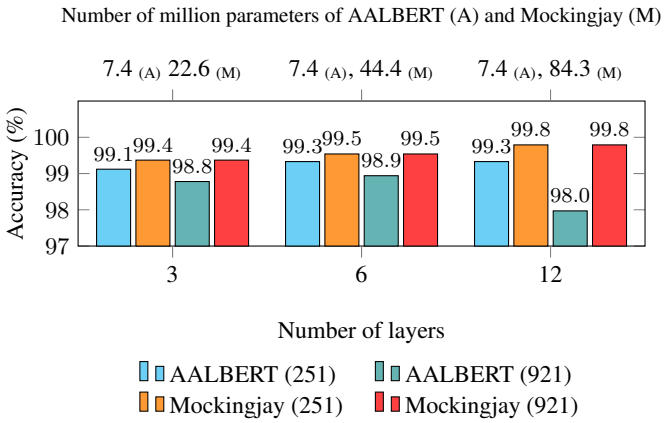


Fig. 5: Speaker classification accuracy on different models and settings with their model parameters. "AALBERT / Mockingjay (251)": settings of utterance-level speaker classification on 251 speaker, "AALBERT / Mockingjay (921)": settings of utterance-level speaker classification on 921 speaker

4.2. Phoneme classification

To measure pre-trained networks' performance on phoneme classification, we build phoneme classifiers with the pre-trained networks followed by two fully-connected layers for prediction. Two pre-trained networks, Mockingjay [5] and AALBERT, are investigated here, with the former as the baseline.

In Fig 4, we show the performance of our models with different layers and settings and compare them to the baseline model (Mockingjay). The vertical axis is the phoneme classification accuracy, while the horizontal axis is the number of network parameters. For both fine-tuning and weighted-sum case, AALBERT shows comparable classification accuracy compared to Mockingjay, but with much fewer network parameters. We also note that both 12-layer AALBERT and Mockingjay (denoted as AALBERT-12L and Mockingjay-12L) does not provide performance gain as compared

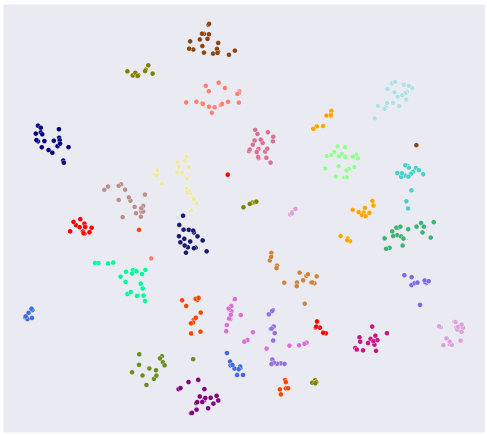
to the 3- and 6- layer counterpart. We conjecture that the saturation in performance is because we use a limited amount of pre-training data. The 6-layer AALBERT and Mockingjay are sufficient to encode knowledge in our 360 hours of pre-training data.

In Fig 3a and Fig 3b, we show the performance on phoneme classification tasks of both feature-extraction case and fine-tuning case versus different proportions of training data being used. Here are two observations. First of all, not only Mockingjay but AALBERT outperforms the input acoustic feature (shown in Fig 3a, Fig 3b). Secondly, these figures show that the representations extracted from Mockingjay and AALBERT have similar performance on phoneme classification tasks.

4.3. Speaker classification

Then, we evaluate the model performance with utterance-level speaker classification in train-clean-100 and train-clean-360 subsets. There are 251 and 921 speakers in the two subsets, respectively. We only use the weighted-sum representations in this part due to space limitation.

(a) mel and its delta (b) mel spectrogram



(c) AALBERT representation

Fig. 6: Visualization of 25 speakers representations (last layer) via t-SNE in train-clean-100 dataset. Different colors represent different speakers.

On top of the representation, we utilize a linear layer followed by a mean-pooling layer for the classification. In Fig 5, we visualize the performance of speaker classification using Mockingjay and AALBERT representations. The performance of the baseline using the input acoustic feature is not shown in the figure, where the accuracy is 0.6%. The results suggest that both AALBERT and Mockingjay encode more abundant speaker information than the raw input of the two pre-trained networks and yield nearly perfect classification results, while AALBERT leverages much fewer parameters than Mockingjay.

Furthermore, we use t-SNE [29] to visualize the utterance representations extracted from input acoustic feature and AALBERT in Fig 6a and Fig 6c. In the figures, each point represents an utterance with its embeddings generated by mean-pooling; we encoder each speaker with a different color. The utterance representations from AALBERT for each speaker are clustered together, while we cannot observe the same phenomenon from the input acoustic features. The result shows that AALBERT better encodes speaker information.

In conclusion, AALBERT shows comparable results on speaker classification tasks against Mockingjay, yet using 91% fewer parameters.

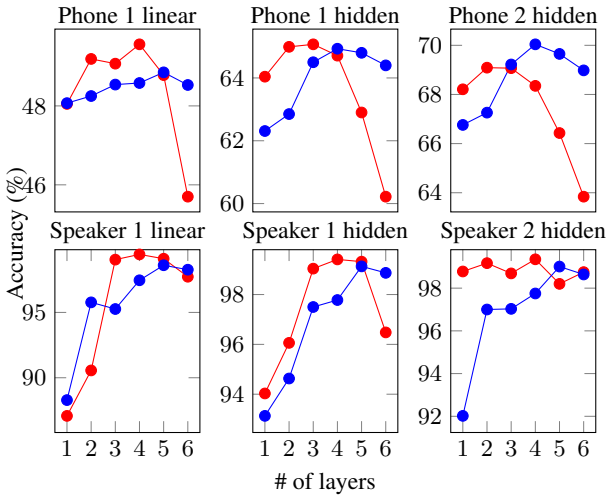


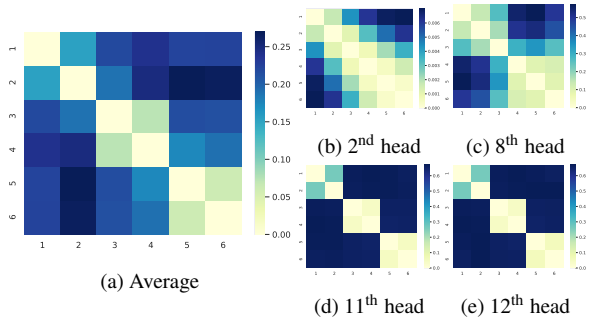
Fig. 7: Probing task, Blue Line: AALBERT-6L, Red line: Mockingjay-6L

4.4. Probing task

We utilize two probing tasks, phoneme classification and frame-level speaker classification[1], to examine how much phoneme and speaker information contain in the representations of each layer. In both tasks, we use train-clean-100 dataset, which is unseen at the pre-training stage. We probe AALBERT-6L and Mockingjay-6L since the average performances of them are the best. We utilize three different classifiers as the probing models, linear, one hidden layer, and two hidden layers, to probe each layer of the pre-trained models for the speaker information and the phoneme information. We use several probing models with different network architectures to mitigate the possible bias from the probing models.

Fig 7 shows the result of probing tasks. For the probing of phoneme information, the three different probing models show the same trends among the same pre-training model. In both pre-training models, as the depth increases, the phoneme information increases first and then decreases. Comparing the two pre-training models, the peak of the Mockingjay-6L is closer to the input layers than AALBERT-6L. On the other hand, when comparing the absolute performance of Mockingjay-6L and AALBERT-6L, the conclusion from different probing models would be different. Mockingjay-6L

[1]Since we want to analyze an individual representation instead of the whole utterance, we choose frame-level instead of utterance-level.



(a) Average (b) 2nd head (c) 8th head (d) 11th head (e) 12th head

Fig. 8: The JS divergence of attention distribution between different layers in AALBERT-6L. (a) is the average case, while (b)(c)(d)(e) represent different attention heads.

achieves better phoneme classification accuracy for the shallower probing model, whereas AALBERT-6L obtains better performance of the deeper probing model. For speaker information, the 5^{th} layer of AALBERT-6L contains the most speaker information, while the 4^{th} layer is the best for Mockingjay-6L.

The results in Fig 7 further indicate that the intermediate representations outperform the representations from the last layer in all four different probing tasks regardless of Mockingjay-6L or AALBERT-6L model. This might indicate that the last layer fits the pre-training tasks too much; therefore, the representations extracted from the intermediate layers may be more suitable for downstream tasks.

4.5. Attention distribution in AALBERT

Here we repeat section 3.1 experiments on AALBERT-6L. Fig 8a shows that the JS divergence of attention distribution is very small between layer 1, 2, layer 3, 4, and layer 5, 6. Fig 8b shows that the JS divergence are small in diagonal and its neighbor area. On the contrary, the first and the last layer differ a lot. In Fig 8c, the JS divergence between first three layers are small, and so do the last three ones. However, the JS divergence between these two parts are large. In Fig 8d, 8e, the JS divergence of layer 1,2, layer 3,4 and layer 5,6 are small, but the JS divergences between every two layers of them are large. These results show that the same parameters may still cause totally different attention distribution over different layers.

5. ACKNOWLEDGEMENT

We thank to National Center for High-performance Computing (NCHC) for providing computational and storage resources.

6. CONCLUSION

In this paper, we present a novel model, Audio ALBERT (AALBERT). AALBERT is a pre-trained model for extracting latent representations that encode the audio information. The model is learned by reconstructing the masked input acoustic features to the linear spectrogram. We show that AALBERT can achieve comparable performances against Mockingjay, a BERT-like pre-trained audio model, yet with much fewer parameters. Besides, we show promising results in encoding audio information with much smaller pre-trained models. For our future work, we will investigate various model architectures to improve further the efficiency of pre-trained models in computation and parameter usage.

7. REFERENCES

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[2] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer, "Deep contextualized word representations," in *Proceedings of NAACL-HLT*, 2018, pp. 2227–2237.

[3] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever, "Improving language understanding by generative pre-training," 2018.

[4] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, pp. 9, 2019.

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

[6] Dongwei Jiang, Xiaoning Lei, Wubo Li, Ne Luo, Yuxuan Hu, Wei Zou, and Xiangang Li, "Improving transformer-based speech recognition using unsupervised pre-training," *arXiv preprint arXiv:1910.09932*, 2019.

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

[8] Murali Karthick Baskar, Shinji Watanabe, Ramon Astudillo, Takaaki Hori, Lukáš Burget, and Jan Černocký, "Semi-Supervised Sequence-to-Sequence ASR Using Unpaired Speech and Text," in *Proc. Interspeech 2019*, 2019, pp. 3790–3794.

[9] Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli, "wav2vec: Unsupervised Pre-Training for Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 3465–3469.

[10] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut, "Albert: A lite bert for self-supervised learning of language representations," in *International Conference on Learning Representations*, 2019.

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton, "A simple framework for contrastive learning of visual representations," *arXiv preprint arXiv:2002.05709*, 2020.

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

[13] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in

Advances in neural information processing systems, 2019, pp. 5753–5763.

[14] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov, "Roberta: A robustly optimized bert pretraining approach," 2019.

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

[16] Yu-An Chung, Wei-Ning Hsu, Hao Tang, and James Glass, "An Unsupervised Autoregressive Model for Speech Representation Learning," in *Proc. Interspeech 2019*, 2019, pp. 146–150.

[17] Xingchen Song, Guangsen Wang, Zhiyong Wu, Yiheng Huang, Dan Su, Dong Yu, and Helen Meng, "Speech-XLNet: Unsupervised Acoustic Model Pretraining For Self-Attention Networks," *arXiv preprint arXiv:1910.10387*, 2019.

[18] Alexei Baevski, Steffen Schneider, and Michael Auli, "vq-wav2vec: Self-supervised learning of discrete speech representations," in *International Conference on Learning Representations*, 2019.

[19] Ofir Press and Lior Wolf, "Using the output embedding to improve language models," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 2017, pp. 157–163.

[20] Tong Xiao, Yinqiao Li, Jingbo Zhu, Zhengtao Yu, and Tongran Liu, "Sharing attention weights for fast transformer," *CoRR*, vol. abs/1906.11024, 2019.

[21] Raj Dabre and Atsushi Fujita, "Recurrent stacking of layers for compact neural machine translation models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, vol. 33, pp. 6292–6299.

[22] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob D Uszkoreit, and Lukasz Mieczyslaw Kaiser, "Universal transformers," Nov. 21 2019, US Patent App. 16/417,587.

[23] Ganesh Jawahar, Benoît Sagot, and Djamé Seddah, "What does bert learn about the structure of language?," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 3651–3657.

[24] Li, Chung-Yi and Yuan, Pei-Chieh and Lee, Hung-Yi, "What does a network layer hear? analyzing hidden representations of end-to-end asr through speech synthesis," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6434–6438.

[25] Yonatan Belinkov, Ahmed Ali, and James Glass, "Analyzing Phonetic and Graphemic Representations in End-to-End Automatic Speech Recognition," in *Proc. Interspeech 2019*, 2019, pp. 81–85.

[26] Ilya Loshchilov and Frank Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2018.

[27] Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 5206–5210.

[28] Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger, "Montreal forced aligner: Trainable text-speech alignment using kaldi," *Proc. Interspeech 2017*, pp. 498–502, 2017.

[29] Maaten, Laurens van der and Hinton, Geoffrey, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.