Personality-aware Training based Speaker Adaptation for End-to-end Speech Recognition

Yue Gu[1]*, Zhihao Du, Shiliang Zhang, Qian Chen, Jiqing Han*[1]

[1]School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

yuegu@stu.hit.edu.cn, jqhan@hit.edu.cn

Abstract

Speaker adaptation has been widely studied to solve the mismatch between training and test conditions for end-to-end automatic speech recognition (ASR). A key challenge of speaker adaptation is lack of sufficient annotated target-speaker data. Considering the training set is always a large-scale one and contains various speakers, it is likely that utterances in the training set can have similar voice characters with the target speaker, and naturally those similar utterances can be treated as a supplement for target speaker data in the adaptation process. Therefore, we propose personality-aware training (PAT) framework to adapt a pre-trained ASR to the target speaker. In PAT, the small-scale target speaker data is viewed as anchors, and the losses of training samples are re-weighted according to the voice character similarity between the anchors and training samples, where the voice character similarity is derived from the speaker or prosody embedding extractor. Experiments on KeSpeech and MagicData corpora show that, compared with the unadapted system, the proposed method achieves 6.35% and 11.86% relative reduction on character error rate with only 10-minute pseudo-label and true-label adaptation data, respectively[1].

Index Terms: Personality-aware, speaker adaptation, speech recognition, personalization

1. Introduction

Recently, end-to-end automatic speech recognition (E2E ASR) systems achieve promising performance with thousands of training data recorded by many speakers [1–3]. However, the performance of E2E systems degrade rapidly, when there is a huge mismatch on voice characters between the training and test conditions. Speaker adaptation algorithms attempt to alleviate the above mismatch by adapting the ASR model to the target speaker, which is also called personalization.

Recent studies on speaker adaptation for ASR systems can be devided into three categories. The first one is embedding-based speaker-aware training (SAT), in which auxiliary speaker embeddings, such as i-vector [4] and x-vector [5], are fed into an ASR model along with speech features. In this way, the model is facilitated to normalize the speaker variation [6, 7]. Subsequently, the attention mechanism is introduced to SAT. Typically, a speaker-aware attention module is incorporated to the transformer based ASR model [8], the attention-over-attention mechanism is introduced to aggregate frame-level speaker embeddings and generate an utterance-level embedding [9], and the M-vector is generated by involving a memory block and weighting relevant i-vectors through attention [10]. The second category is based on data-augmentation, in which the personalized speech synthesis model is used to generate additional data for specific speaker adaptation [11, 12]. The third one is a model-based approach, which can be divided into two types according to whether additional parameters are imported into the adapted model or not. Finetuning is the most straightforward model-based approach. Representatively, the speaker signature (SS) is registered into the parameters by finetuning with adaptation data and a hierarchical decomposition study is performed to evaluate the effect of finetuning on each module [12]. Another model-based approach is involving extra speaker-dependent parameters to represent speaker variability, such as learning hidden unit contributions [13–16] and scaling and shifting factors [17]. To alleviate the overfitting problem caused by the limited adaptation data, the L2 norm [18], Kullback–Leibler divergence [19–21] and adversarial training [22] are introduced to regularize the adapted model.

Current adaptation approaches always need sufficient adaptation data to guarantee the performance on the target speaker. However, it is challenging to collect sufficient annotated data from the target speaker due to the cost and privacy protection. Considering the training set of E2E ASR is always large-scale, there may be utterances having the similar voice characters as the target speaker. Thus, it is reasonable to utilize the similar utterances to be a supplement for target speaker data in the adaptation process, which has not been well-studied. Therefore, we propose a novel model-based approach, personality-aware training (PAT) framework, in which the small amount of target speaker data is treated as anchors, and the losses of training samples are re-weighted according to the similarities to the anchors, where training samples come from original training set and the target speaker data. In order to estimate the voice character similarity, the speaker or prosody extractor is trained to extract utterance-level representations. The contrast between training samples and anchors plays a role of filtering the utterances, so the model can be adapted with filtered utterances whose personality is similar to the target speaker.

We evaluate the effectiveness of the PAT framework on the open-source KeSpeech and MagicData corpora. Experiments show that PAT can achieve 6.35% and 11.86% relative reduction on character error rate (CER) with only 10-minute pseudo-label and true-label adaptation data. With extremely scarce 5-minute and 3-minute true-label adaptation data, PAT can still provide 8.24% and 5.77% relative improvements. In addition, by using training and adaptation data at the same time, PAT not only improves the performance on specific speaker, but also alleviates the overfitting problem.

2. Personality-aware Training Framework

In this section, we describe the proposed personality-aware training framework and its practical implementation with speaker and prosody embedding extractors. As shown in Fig-

[1]Code will be at github:shibeiing/Personality-aware-Training-PAT

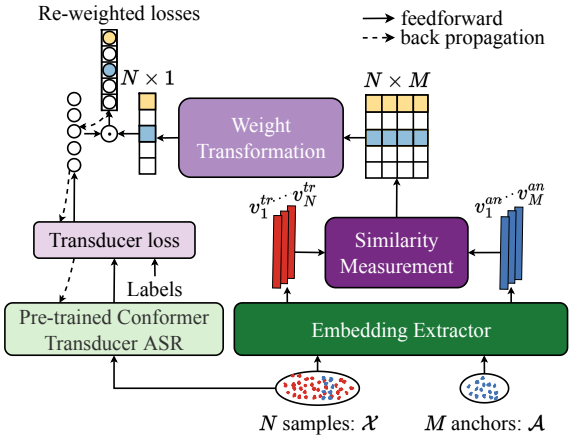


Figure 1: *The overview of our PAT framework.*

ure 1, PAT framework comprises a shared embedding extractor to obtain utterance-level representations, and the representations are then consumed by a similarity measurement module to calculate the Cartesian product of similarity on training samples and anchors. After the weight transformation, similarities are converted to the importance weights of training samples. Finally, by re-weighting ASR losses and back-propagation, the end-to-end ASR model is adapted to the target speaker.

2.1. PAT framework

In each mini-batch, there are N training samples $\mathcal{X}$: $\{X_1, X_2, \ldots, X_N\}$ and M anchors $\mathcal{A}$: $\{A_1, A_2, \ldots, A_M\}$, where $\mathcal{X}$ are randomly selected from the union set of original training set and the target speaker data. The training samples $\mathcal{X}$ and anchors $\mathcal{A}$ are fed into a shared embedding extractor to obtain utterance-level representations v_n^{tr} and v_m^{an}, respectively:

$$v_n^{tr} = \text{EmbExtractor}(X_n) \quad (1)$$
$$v_m^{an} = \text{EmbExtractor}(A_m) \quad (2)$$

where "EmbExtractor" can be implemented by any utterance-level embedding models. In this paper, we employ the speaker and prosody embedding extractors, and more details are provided in section 2.2 and 2.3, respectively.

Given the representations of training and anchor samples, we calculate the Cartesian product of their cosine similarities to obtain the similarity matrix $S = \{s_{n,m}\}$:

$$s_{n,m} = \frac{< v_n^{tr}, v_m^{an} >}{||v_n^{tr}|| \cdot ||v_m^{an}||} \quad (3)$$

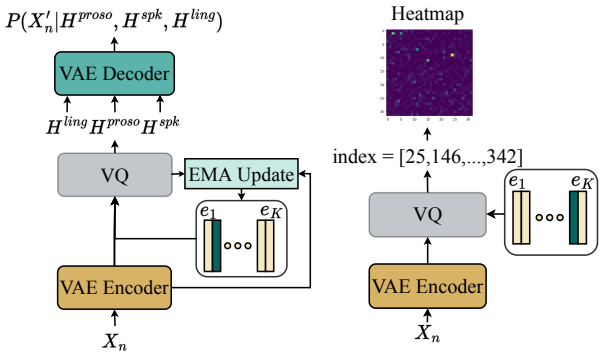
where $< \cdot, \cdot >$ represents the dot product of two vectors, and $||\cdot||$ means the L2 norm of a vector. To evaluate the importance weight w_n of a training sample X_n, we aggregate the similarities between X_n and all anchors $\mathcal{A}$ in a mini-batch:

$$w_n = \frac{1}{M} \sum_{m=1}^{M} s_{n,m} \quad (4)$$

Since cosine similarities belong to $[-1, 1]$, simply using the raw weight w_n to re-weight the training losses may slow down the optimization process. Therefore, we clip and re-scale the raw weights to match the batch size N:

$$\bar{w}_n = \frac{N \cdot \max(w_n, 0)}{\sum_{i=1}^{N} \max(w_i, 0) + \epsilon} \quad (5)$$

where ϵ is a small number for numeric stability. Finally, we use the re-scaled value $\bar{w}_n$ to represent the importance of training sample X_n and re-weight the ASR losses:



(a) *Training stage* (b) *Inference stage*

Figure 2: *The illustration of (a) training and (b) inference stages for prosody embedding extractor.*

$$\mathcal{L} = \frac{1}{N} \sum_{n=1}^{N} \bar{w}_n \cdot L(\boldsymbol{y}_n, \mathbf{ASR}_\theta(X_n)) \quad (6)$$

where $\boldsymbol{y}_n$ is the text label of training sample X_n, and L means the original loss function of ASR model. $\mathbf{ASR}_\theta$ represents a pre-trained ASR model with the parameter of θ. Through back-propagation, the gradients of samples with similar voice characters are enhanced and the pre-trained model can be adapted to the target speaker:

$$\theta = \theta - \frac{\eta}{N} \sum_{n=1}^{N} \bar{w}_n \frac{\partial L(\boldsymbol{y}_n, \mathbf{ASR}_\theta(X_n))}{\partial \theta} \quad (7)$$

where η means the learning rate. Theoretically, PAT can be applied to all popular end-to-end ASR models, such as CTC, attention and transducer based models.

2.2. Speaker Embedding Extractor

The speaker embedding extractor is based on the commonly-used ResNet101 network architecture [23], which comprises four residual blocks with the layer number of $3, 4, 23, 3$ and the channel number of $32, 64, 128, 256$. As in x-vector [5], a statistics pooling layer is employed to aggregate the speaker information over the whole utterance (i.e. mean and standard deviation are calculated across the frames). After the pooling layer, we flatten the utterance-level representation and feed it to a linear transformation. As as result, a 256-dimensional speaker embedding is obtained. More details and pre-trained models can be found in [24].

2.3. Prosody Embedding Extractor

The prosody extractor is based on vector quantized variational autoencoder (VQ-VAE) [25] which combines VAE with vector quantization. VQ-VAE learns discrete tokens from the continuous latent space by using the latent representations to look up a 1-of-K embedding vector, which is used to reconstruct the targets. As shown in Figure 2(a), the prosody extractor mainly comprises three parts at the training stage: an encoder to map log-mel filterbank (Fbank) $X = (x_1, \ldots, x_T)$[2] into latent representations $H^{enc} = (h_1^{enc}, \ldots, h_T^{enc})$, a vector quantization module to discretize latent representations and form an embedding codebook $E = (e_1, \ldots, e_K)$, and a decoder to reconstruct the spectrogram $X' = (x_1', \ldots, x_T')$ with the discretized latent representations $H^{proso} = (h_1^{proso}, \ldots, h_T^{proso})$, the linguistic embeddings $H^{ling} = (h_1^{ling}, \ldots, h_T^{ling})$ and the

[2]For notational simplicity, we omit the subscript n of samples.

speaker embedding H^{spk}. While H^{spk} is extracted with the speaker embedding extractor in section 2.2, H^{ling} is the phonetic posteriorgram predicted by a phoneme recognition model [26, 27]. Similar to [28], H^{spk} and H^{ling} represent the timbre and content in speech, respectively, so that H^{proso} contains much more speaker-and-content-independent prosody information. The discretized latent representation is calculated by the nearest neighbour look-up with the shared codebook E:

$$h_t^{proso} = e_k, \text{where } k = \arg\min_j ||h_t^{enc} - e_j||_2 \quad (8)$$

The codebook is updated in a manner of exponential moving average (EMA), in which the weighting factors for older latent representations decrease exponentially:

$$e_j^{(u)} = \alpha \cdot h_t^{enc} + (1-\alpha) \cdot e_j^{(u-1)} \quad (9)$$

where $\alpha \in [0, 1]$ is the smoothing factor, and u is the index of update steps. To get a phone-aligned prosody embedding, we calculate the autocorrelation matrices (Corr) of H^{ling} and H^{enc} and minimize the mean square error (MSE) of $Corr(H^{ling})$ and $Corr(H^{enc})$. At the same time, the output of prosody encoder is forced to be closed to the nearest neighbour e_k. The final loss function of prosody extractor is:

$$\begin{aligned} L_{proso} = MSE(\hat{X}', X) + \beta MSE(H^{proso}, H^{enc}) \\ + \gamma MSE(Corr(H^{ling}), Corr(H^{enc})) \end{aligned} \quad (10)$$

where β and γ are the hyperparameters. $\hat{X}'$ denotes the prediction of X'. Note that the input X only contains low frequency bands (1-24), and the expected output X' only contains middle frequency bands (25-48). In this way, the prosody embedding h^{proso} learns the common formant trend across different frequency bins. Since a proper initialization of embedding codebook benefits the prosody extractor, we train the VAE without vector quantization in the first 50 epochs, and apply k-means clustering on the H^{enc} of all training samples to obtain K cluster centroids. After initialization, we add the vector quantization module as the prosody bottleneck in the last 50 epochs.

The concatenation of prosody encoder and vector quantization module is viewed as the prosody embedding extractor. We first use the prosody embedding extractor to transform the utterance into a sequence of discrete embeddings, then the frequency vector of every embedding in the codebook is used to represent the utterance. Figure 2(b) shows the extraction process of frequency vector, in which the frequency vector is folded to form a two-dimensional heatmap.

3. Experiment Settings

3.1. Datasets

Experiments are mainly conducted on the KeSpeech corpus, an open source dataset of Mandarin and its eight subdialects including Zhongyuan, Southwestern, Ji-Lu, Jiang-Huai, Lan-Yin, Jiao-Liao, Northeastern and Beijing [29]. KeSpeech is split into two phases according to the recording time, and the phase-1 is used to pre-train the ASR model in this paper, which contains 895 hours data recorded by 27,044 speakers in 34 cities of China. The train and development sets contain 860 hours (24,945 speakers) and 3.4 hours (100 speakers) respectively. The test set is used to evaluate the source-domain performance, which contains 31 hours (1,999 speakers).

We evaluate our method in a mismatch condition where training and test data are different on voice characters, accent distributions, recording devices, etc. Two open-source datasets from the MagicData organization are employed as the target-domain test sets, i.e., the Sichuan[3] and Zhengzhou[4] corpora. The Sichuan corpus is a part of Southwestern Mandarin, and the Zhengzhou corpus belongs to Zhongyuan Mandarin. The target-domain test set consists of 20 speakers with the duration of about 30 minutes for each speaker. We use 10 minutes for adapting (also used as anchors) and 20 minutes for test.

Table 1: *The CERs(%) and parameters numbers of two Conformer Transducer with different model sizes.*

Model	Target domain	Parameters (M)
Conformer-T(L)	15.81	112.8
Conformer-T(S)	19.06	9.4

3.2. Training details

We evaluate the effectiveness of our PAT framework on conformer transducer (Conformer-T) based ASR model, which achieves the state-of-the-art performance on several benchmarks. The 80-dimensional Fbank is used as the feature with the window size of 25ms and the shift size of 10ms. Since personalized ASR models are probably adapted on the cloud and pushed back to devices for inference [30], the computational complexity and model size at the inference stage should be carefully considered. Thus, we train two ASR models with different sizes, i.e., Conformer-T(S) and Conformer-T(L), where the network configurations are the same as [1]. Table 1 shows their performance and parameter numbers. Although the large model achieves better performance than the small one, it is impractical to deploy it on smart devices. The personalized ASR is expected to have a small footprint and the closing performance to the larger one. We employ the Conformer-T(S) as baseline in the following experiments and fill the performance gap. We use the ESPnet toolkit [31] to implement and evaluate baselines and adaptation methods. Following the default settings in ESPnet, we train the model 100 epochs with the batch size of 80.

For the speaker embedding extractor, a pre-trained model [24] is employed to extract embeddings for similarity measurement and the calculation of H^{spk} (seen in Section 2.2). The prosody embedding extractor comprises one convolutional subsampling layer with a factor of four on the time axis, which is followed by four transformer layers and a bottleneck linear layer. In the first three transformer layers, a multi-head attention with four heads and a feedforward layer with 256 output units are stacked. The last transformer layer has a single-head attention to calculate the autocorrelation matrix $Corr(H^{enc})$, then the dimension of its output is reduced to 16 by a linear layer. In vector quantization, the codebook size is set to 1,024 and the vector dimension is 16. The hyperparameter α, β and γ are set to 0.001, 0.25 and 1.0, respectively. We train the prosody embedding extractor on four GPU with the batch size of 48 on each GPU. We employ the Noam optimizer [32] to train the extractor with the learning rate of 1.5 and 25000 warmup steps.

4. Experimental Results

4.1. Comparison of PAT and other adaptation methods

Table 2 compares the proposed PAT framework with other adaptation methods in terms of CER on source and target domains. In the accent adaptation, the location of target speaker is available and the utterances(75h∼84h) collected from the same area

[3]https://magichub.com/datasets/sichuan-dialect-scripted-speech-corpus-daily-use-sentence/

[4]https://magichub.com/datasets/zhengzhou-dialect-scripted-speech-corpus-daily-use-sentence/

Table 2: *Evaluation of different speaker adaptation methods on source and target domain test sets. The "Supervised" and "Unsupervised" mean whether the 10-minute adaptation data is annotated by human or not. CER.R means the relative CER reduction on baseline.*

Method	Source domain		Target domain	
	CER	**CER.R**	CER	**CER.R**
Baseline	12.69	NA	19.06	NA
Unsupervised				
Accent adaptation	13.14	-3.54	18.71	1.84
SAT [7]	12.79	-0.79	18.35	3.57
Speaker sign. (SS) [12]	13.11	-3.31	18.21	4.46
PAT (speaker)	12.82	-1.02	17.90	6.09
PAT (prosody)	**12.65**	**0.32**	**17.85**	**6.35**
Supervised				
SAT+SS	14.79	-16.55	18.09	5.09
Speaker sign. (SS) [12]	13.14	-3.54	17.82	6.51
PAT (speaker)	12.82	-1.02	**16.80**	**11.86**
PAT (prosody)	**12.63**	**0.47**	16.87	11.49

are selected to fine-tune the pre-trained ASR. In speaker signature, the pre-trained ASR model is directly fine-tuned with the scarce target speaker adaptation data, which is a commonly-used effective method. In speaker-aware training (SAT), the speaker embedding of the same utterance is employed as the auxiliary input for ASR model. SAT can be applied to various ASR models, and it is usually considered as a comparative method in recent studies. In SAT+SS, a well-trained SAT model is further fine-tuned with the annotated adaptation data. "Unsupervised" means the adaptation data is not annotated by human, instead, audios are fed into the baseline to obtain the 1-best results of beam search, which is viewed as the pseudo-label. In both unsupervised and supervised conditions, all adaptation methods can improve the performance on the target speaker. Compared with the baseline, our PAT framework can achieve the relative CER reductions of 6.09% and 6.35% without any annotated adaptation data. When the ground-truth label is involved, PAT achieves the highest relative improvements of 11.86% and 11.49%. While the performance of other methods degrades on the source domain to some extent, our PAT maintains or even slightly improves the performance on the source domain. This indicates that the generalization ability of PAT is better than other adaptation methods. It should be noted that, at the inference stage, PAT does not involve any extra parameters but fills 69% performance gap between Conformer-T(S) and Conformer-T(L) which is ten times larger than the former.

4.2. Ablation study and analysis

We perform ablation study to evaluate the impact of each module in PAT. Table 3 shows the performance of different combinations of adaptation data and PAT modules. In EXP1 and EXP6, we add 10-minute pseudo-label and true-label adaptation data into the training set. From the table, we can see that the performance improvement is fairly limited, which is because the adaptation data is very scarce (10 minutes) compared with the training set (860 hours). In EXP2 and EXP3, the adaptation data is excluded, and the recognition performance can be improved. This indicates that our PAT framework can select similar utterances and adapt model to the target speaker. When the adaptation data is involved, PAT provides a significant improvement no matter the data is annotated or not, which indicates that PAT can make fully use of adaptation data. Comparing the prosody

Table 3: *The ablation studies on adaptation data and PAT.*

Experiment	Source domain	Adapt. data	PAT (spk)	PAT (proso)	CER	**CER.R**
Baseline	✓				19.06	NA
Unsupervised						
EXP1	✓	✓			18.82	1.26
EXP2	✓		✓		18.99	0.37
EXP3	✓			✓	18.85	1.10
EXP4	✓	✓	✓		17.90	6.09
EXP5	✓	✓		✓	**17.85**	**6.35**
Supervised						
EXP6	✓	✓			18.60	2.41
EXP7	✓	✓	✓		**16.80**	**11.86**
EXP8	✓	✓		✓	16.87	11.49

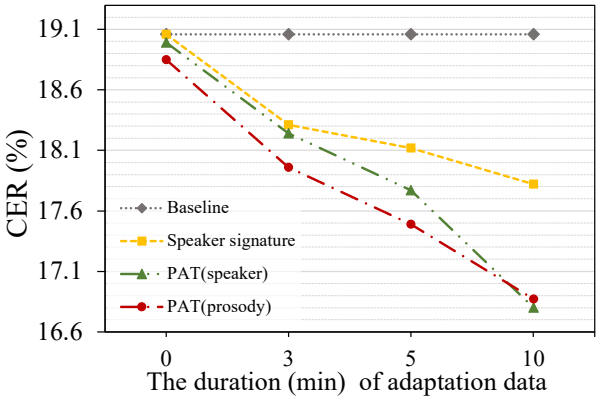


Figure 3: *Comparison of different methods on the duration of adaptation data.*

and speaker embedding extractors, PAT(prosody) achieves a better performance than PAT(speaker) when the pseudo-label is used for adaptation. In contrast, when the true-label is available, PAT(speaker) performs slightly better.

4.3. Robustness to the duration of adaptation data

Figure 3 shows the CERs of adapted ASR models using various duration of true-labeled adaptation data. As expected, the more adaptation data is used, the more improvement can be achieved. Compared with the speaker signature, our PAT can achieve a similar performance with only 3-minute adaptation data, while SS needs 10 minutes. This indicates that PAT utilizes the data more efficiently and alleviates the cost of collocting the adaptation data. We find that PAT(prosody) outperforms PAT(speaker), when the adaptation data is extremely scarce ($\leq$ 5 minutes). This indicates prosody based similarity measurement is more robust to the duration of adaptation data.

5. Conclusion

In this paper, we propose a personality-aware training framework and it is implemented with two embedding extractors. Experiments on the open-source KeSpeech and MagicData datasets show that the proposed PAT framework can make fully use of the limited adaptation data. Compared with the unadapted Conformer transducer model, our PAT framework achieves up to 6.35% and 11.86% relative CER reduction with 10-minute pseudo-label and true-label adaptation data, respectively. Moreover, the PAT framework not only improves the performance on target speaker but also alleviates the overfitting problem caused by finetuning with limited data.

6. References

[1] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *INTERSPEECH*, 2020, pp. 5036–5040.

[2] F. Boyer, Y. Shinohara, T. Ishii, H. Inaguma, and S. Watanabe, "A study of transducer based end-to-end ASR with espnet: Architecture, auxiliary loss and decoding strategies," in *ASRU*, 2021, pp. 16–23.

[3] Z. Tüske, G. Saon, and B. Kingsbury, "On the limit of english conversational speech recognition," in *INTERSPEECH*, 2021, pp. 2062–2066.

[4] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *TASLP*, vol. 19, no. 4, pp. 788–798, 2010.

[5] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *ICASSP*, 2018, pp. 5329–5333.

[6] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *ASRU*, 2013, pp. 55–59.

[7] A. Senior and I. Lopez-Moreno, "Improving dnn speaker independence with i-vector inputs," in *ICASSP*, 2014, pp. 225–229.

[8] Y. Zhao, C. Ni, C.-C. Leung, S. R. Joty, E. S. Chng, and B. Ma, "Speech transformer with speaker aware persistent memory," in *INTERSPEECH*, 2020, pp. 1261–1265.

[9] G. Wan, J. Pan, Q. Wang, J. Gao, and Z. Ye, "Speaker adaptive training for speech recognition based on attention-over-attention mechanism," in *INTERSPEECH*, 2020, pp. 1251–1255.

[10] L. Sarı, N. Moritz, T. Hori, and J. Le Roux, "Unsupervised speaker adaptation using attention-based speaker memory for end-to-end asr," in *ICASSP*, 2020, pp. 7384–7388.

[11] Y. Huang, L. He, W. Wei, W. Gale, J. Li, and Y. Gong, "Using personalized speech synthesis and neural language generator for rapid speaker adaptation," in *ICASSP*, 2020, pp. 7399–7403.

[12] Y. Huang, G. Ye, J. Li, and Y. Gong, "Rapid speaker adaptation for conformer transducer: Attention and bias are all you need," in *INTERSPEECH*, 2021, pp. 1309–1313.

[13] P. Swietojanski and S. Renals, "Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models," in *SLT*, 2014, pp. 171–176.

[14] P. Swietojanski, J. Li, and S. Renals, "Learning hidden unit contributions for unsupervised acoustic model adaptation," *TASLP*, vol. 24, no. 8, pp. 1450–1463, 2016.

[15] X. Xie, X. Liu, T. Lee, and L. Wang, "Bayesian learning for deep neural network adaptation," *TASLP*, vol. 29, pp. 2096–2110, 2021.

[16] J. Deng, X. Xie, T. Wang, M. Cui, B. Xue, Z. Jin, M. Geng, G. Li, X. Liu, and H. Meng, "Confidence score based conformer speaker adaptation for speech recognition," in *INTERSPEECH*, 2022, pp. 2623–2627.

[17] Z.-Q. Wang and D. Wang, "Unsupervised speaker adaptation of batch normalized acoustic models for robust asr," in *ICASSP*, 2017, pp. 4890–4894.

[18] H. Liao, "Speaker adaptation of context dependent deep neural networks," in *ICASSP*, 2013, pp. 7947–7951.

[19] D. Yu, K. Yao, H. Su, G. Li, and F. Seide, "Kl-divergence regularized deep neural network adaptation for improved large vocabulary speech recognition," in *ICASSP*, 2013, pp. 7893–7897.

[20] Z. Meng, Y. Gaur, J. Li, and Y. Gong, "Speaker adaptation for attention-based end-to-end speech recognition," in *INTERSPEECH*, 2019, pp. 241–245.

[21] F. Weninger, J. Andrés-Ferrer, X. Li, and P. Zhan, "Listen, attend, spell and adapt: Speaker adapted sequence-to-sequence ASR," in *INTERSPEECH*, 2019, pp. 3805–3809.

[22] Z. Meng, J. Li, and Y. Gong, "Adversarial speaker adaptation," in *ICASSP*, 2019, pp. 5721–5725.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[24] F. Landini, J. Profant, M. Díez, and L. Burget, "Bayesian HMM clustering of x-vector sequences (vbx) in speaker diarization: Theory, implementation and analysis on standard tasks," *Comput. Speech Lang.*, vol. 71, p. 101254, 2022.

[25] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural discrete representation learning," in *NeurIPS*, 2017, pp. 6306–6315.

[26] Z. Du, M. Lei, J. Han, and S. Zhang, "Pan: Phoneme-aware network for monaural speech enhancement," in *ICASSP*, 2020, pp. 6634–6638.

[27] L. Sun, K. Li, H. Wang, S. Kang, and H. M. Meng, "Phonetic posteriorgrams for many-to-one voice conversion without parallel data training," in *ICME*, 2016, pp. 1–6.

[28] Y. Ren, M. Lei, Z. Huang, S. Zhang, Q. Chen, Z. Yan, and Z. Zhao, "Prosospeech: Enhancing prosody with quantized vector pre-training in text-to-speech," in *ICASSP*, 2022, pp. 7577–7581.

[29] Z. Tang, D. Wang, Y. Xu, J. Sun, and et.al., "Kespeech: An open source speech dataset of mandarin and its eight subdialects," in *NeurIPS Datasets and Benchmarks*, 2021.

[30] P. Bell, J. Fainberg, O. Klejch, J. Li, S. Renals, and P. Swietojanski, "Adaptation algorithms for neural network-based speech recognition: An overview," *IEEE Open Journal of Signal Processing*, vol. 2, pp. 33–66, 2020.

[31] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. E. Y. Soplin, J. Heymann, M. Wiesner, N. Chen, A. Renduchintala, and T. Ochiai, "Espnet: End-to-end speech processing toolkit," in *INTERSPEECH*, 2018, pp. 2207–2211.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017, pp. 5998–6008.