Parameter-efficient Dysarthric Speech Recognition Using Adapter Fusion and Householder Transformation

Jinzi Qi[1], *Hugo Van hamme*[1]

[1]Department Electrical Engineering-ESAT-PSI, KULeuven, Belgium

jqi@esat.kuleuven.be, hugo.vanhamme@kuleuven.be

Abstract

In dysarthric speech recognition, data scarcity and the vast diversity between dysarthric speakers pose significant challenges. While finetuning has been a popular solution, it can lead to overfitting and low parameter efficiency. Adapter modules offer a better solution, with their small size and easy applicability. Additionally, Adapter Fusion can facilitate knowledge transfer from multiple learned adapters, but may employ more parameters. In this work, we apply Adapter Fusion for target speaker adaptation and speech recognition, achieving acceptable accuracy with significantly fewer speaker-specific trainable parameters than classical finetuning methods. We further improve the parameter efficiency of the fusion layer by reducing the size of query and key layers and using Householder transformation to reparameterize the value linear layer. Our proposed fusion layer achieves comparable recognition results to the original method with only one third of the parameters.

Index Terms: dysarthric speech recognition, parameter efficiency, adapter fusion, Householder transformation

1. Introduction

The latest speech technologies, including Automatic Speech Recognition (ASR), have become increasingly popular and provide great convenience in everyday life. These technologies have traditionally focused on clear, canonical speech. However, in recent years, there has been growing interest in developing ASR models for dysarthric speech [1, 2, 3, 4], a neurological disease characterized by poor phoneme articulation. Dysarthric speakers face difficulties in daily communication, highlighting the importance of automatic dysarthric speech recognition.

State-of-the-art ASR models [5, 6, 7] typically use a Transformer architecture [8] which employs millions of trainable parameters and needs hundreds of hours data for model training. However, for dysarthric speech, data scarcity has been a constant issue, due to difficulties in recruitment, collection and labeling. Thus, compared to training an E2E ASR model from scratch, finetuning an E2E ASR model [9, 10, 11] pretrained on abundant canonical speech seems more feasible. However, finetuning the entire model containing a massive number of parameters with the limited available dysarthric data can lead to overfitting and parameter inefficiency. Moreover, dysarthric speech is influenced by mixed factors, like gender, pathogenesis (diagnosis) and severity level. The huge diversity between dysarthric speakers requires personalized model adaption [2, 3, 12, 13, 14]. Finetuning the entire model would require a personalized model to be stored for each user, which would occupy valuable on-device storage space or require a significant upgrade in server storage for a large number of users [15].

Adapters [16, 17], which only contain a limited number of parameters, can provide a solution for both data scarcity and limited storage size. An adapter is a bottleneck module that is injected between layers of a pretrained model (figure 1(a)). It is trained while the other parts of the pre-trained model are frozen. Previous studies [18, 19, 20, 21] have demonstrated the effectiveness of Adapters in parameter-efficient transfer learning. For dysarthric speech, personalized adapters are trained and tested on atypical speech data in [15], which has resulted in a similar Word Error Rate (WER) compared to finetuning. Furthermore, in [14], an auxiliary net that studies speaker information is added to boost the personalized adapter performance. In this work, we use a model pretrained on canonical speech and then train the inserted adapter with dysarthric speech.

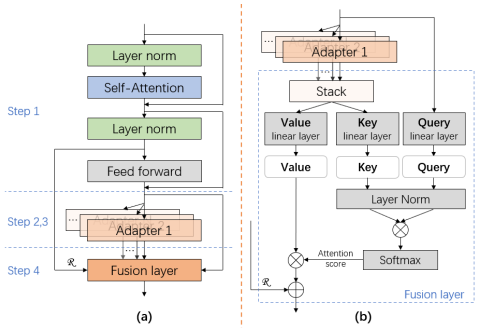


Figure 1: *(a) Transformer encoder layer with adapters and fusion layer, (b) details of the fusion layer.*

Due to the simplicity and the small size of adapter modules, multiple adapters can be easily deployed for different groups, individuals and tasks. When trained adapters from a source domain are available, adapting to a target domain could benefit from existing adapters. To maximize the transferred knowledge among different trained adapters, Adapter Fusion [22] (figure 1(b)) is promoted, which utilizes an attention mechanism, called fusion layer, to combine representations from source adapters and improve the performance of the target task. Specifically, the attention score is calculated and assigned to value linear layer output. For dysarthric speech, we could employ adapters trained on source speaker data and use the fusion layer to maximize the model's performance for target speaker adaptation. However, this may lead to an increase in the number of trainable parameters, which could contradict the goal of having a small storage size for personalized models.

In this work, we apply the Adapter Fusion method to dysarthric speaker adaptation and speech recognition and investigate the feasibility of improving its parameter efficiency. Firstly, we train personalized adapters using source dysarthric

speakers' data and then train the fusion layer using target speaker data for the dysarthric speech recognition task. Secondly, we trace the source of the performance improvement when using the fusion layer by ablating its two components: the attention score and the value linear layer. Furthermore, we inspect the influence of the rotation and scaling operation in the value linear layer by applying Singular Value Decomposition (SVD) to its weight matrix. Finally, we explore the possibility of improving the parameter efficiency of the fusion layer by reducing the key and query linear layer size and using Householder transformation [23] to reformulate the rotation operation in the value linear layer.

In section 2, we introduce the Adapter Fusion method and the application of Householder transformation. Section 3 describes the databases we used and the experimental settings. Results and analysis will be provided in section 4, and section 5 gives conclusions.

2. Methods

In this section, the method we use is introduced in detail. We use a encoder-decoder model with hybrid loss [24] as the base ASR model.It contains a transformer encoder and two decoders: A transformer decoder and a Connectionist Temporal Classification (CTC) [25] decoder. For simplicity, we only insert the Adapter module in the last encoder layer and thus the fusion layer is also used in the last encoder layer.

2.1. Adapter Fusion method

The transformer encoder pretrained on canonical speech $\mathcal{M}_O(\cdot)$ maps a target speaker's input sequence $\mathbf{X}$ of duration T, to $\mathbf{Y}_O = \mathcal{M}_O(\mathbf{X})$. Suppose we have N dysarthric source speakers, and the personalized adapter trained by each source speaker data is $\mathcal{M}_{a_n}(\cdot), n \in [1, N]$. Then the output of adapter n is $\mathbf{Y}_{a_n} = \mathcal{M}_{a_n}(\mathcal{M}_O(\mathbf{X}))$, the stacked adapter output is $\mathbf{Y}_A = [\mathbf{Y}_{a_1}, \mathbf{Y}_{a_2}, ..., \mathbf{Y}_{a_N}]$.

For the fusion layer, we name the whole layer by $\mathcal{M}_F(\cdot)$, weight matrix of value linear layer by $\mathbf{W}$ (no bias used in this layer), the key linear layer by $\mathbf{K}(\cdot)$ and query linear layer by $\mathbf{Q}(\cdot)$. Then in the original fusion layer [22], the fusion layer output is $\mathbf{Y}_F = \sum_{n=1}^{N} \alpha_n \mathbf{Y}_{a_n}\mathbf{W} + \mathcal{R}$, where α_n is the attention score for adapter n, $\alpha_n = softmax_{over\ n}(\mathbf{Q}(\mathbf{Y}_O) \times \mathbf{K}(\mathbf{Y}_{a_n})')$, $\mathcal{R}$ is a residual term (see figure 1(a)).

To avoid gradient vanishing, different from the original version, we add a layer-normalization layer $LN(\cdot)$ for key and query term, then the attention score could be written as $\alpha_n = Softmax_{over\ n}(LN(\mathbf{Q}(\mathbf{Y}_O)) \times LN(\mathbf{K}(\mathbf{Y}_{a_n}))')$. In the following experiments, when we ablate the attention score in the fusion layer, the fusion output becomes $\mathbf{Y}_F = \frac{1}{N}\sum_{n=1}^{N} \mathbf{Y}_{a_n}\mathbf{W} + \mathcal{R}$. When eliminating the value linear layer, the fusion output is $\mathbf{Y}_F = \sum_{n=1}^{N} \alpha_n \mathbf{Y}_{a_n} + \mathcal{R}$.

The weight matrix of value linear layer $\mathbf{W}$ is initialized with an all-one-diagonal and the rest with small weights $(1e{-}6)$ [22, 19]. To guarantee stable adapter outputs and avoid overtraining, $\mathbf{W}$ is regularized to the identity matrix by introducing an additional loss term:

$$L_{reg} = ||\mathbf{I_W} - \mathbf{W}||^2 \tag{1}$$

where $\mathbf{I_W}$ is an identity matrix of same size as $\mathbf{W}$. Then the total loss function during target speaker adaptation training is:

$$L = (1-\lambda_1) * L_{ASR,Trans} + \lambda_1 * L_{ASR,CTC} + \lambda_2 * L_{reg} \tag{2}$$

where $L_{ASR,Trans}$ is the loss from transformer decoder, $L_{ASR,CTC}$ is the loss from CTC decoder.

If we train the fusion layer on target speaker data, the performance could benefit from fusing learned knowledge from trained source adapters. However, compared to a single personalized adapter, the fusion layer may have no advantage in the number of trainable parameters. In this work, we explore methods to improve the parameter efficiency of the fusion layer. A natural choice is to reduce the size of $\mathbf{K}(\cdot)$ and $\mathbf{Q}(\cdot)$. Further on, in the next subsection, we will work on the value linear layer.

2.2. Adapter Fusion with Householder transformation

The weight matrix $\mathbf{W}$ of value linear layer acts on the adapter output. Through SVD, we obtain two orthogonal matrices $\mathbf{U}$ and $\mathbf{V}$ representing a rotation/reflection and a diagonal matrix $\mathbf{\Sigma}$ for scaling, where $\mathbf{W} = \mathbf{U\Sigma V}^T$. After training is complete, we can evaluate the effectiveness of the rotation and scaling operations on the trained $\mathbf{W}$ by forming a new matrix $\mathbf{W_{UV}} = \mathbf{UV}^T$ for rotation and another $\mathbf{W_\Sigma}$ for scaling. $\mathbf{W_\Sigma}$ is a diagonal matrix and its diagonal vector is calculated by $(\mathbf{\Sigma}_{max} - \mathbf{\Sigma}_{min})\frac{\mathbf{c}-max(\mathbf{c})}{max(\mathbf{c})-min(\mathbf{c})} + \mathbf{\Sigma}_{max}$, where $\mathbf{\Sigma}_{max}$ and $\mathbf{\Sigma}_{min}$ are the maximum and minimum of diagonal of $\mathbf{\Sigma}$, and $\mathbf{c}$ is a vector containing l-2 norm of the rows of $\mathbf{W}$.

Therefore, we can reformulate the matrix $\mathbf{W}$ as a scaling vector and a rotation matrix. Since the matrix is regularized to an identity matrix, there is redundancy in its parameters, and it should be full rank. To preserve the rank of the matrix while reducing the number of trainable parameters, we employ Householder transformation [23] to reparameterize the rotation part of $\mathbf{W}$.

The Householder transformation $\mathbf{P}$ describes a reflection about a hyperplane orthogonal to $\mathbf{v}$, a length-preserving orthogonal transform :

$$\mathbf{P} = \mathbf{I_W} - 2\mathbf{vv}^T \tag{3}$$

where $\mathbf{v}$ is column unit vector. Since the product of orthogonal matrices is again orthogonal, we can use it as a reparamerization of $\mathbf{UV}^T$. Suppose $\mathbf{W}$ has d dimensions and d^2 trainable elements. Then to reparameterize $\mathbf{W}$, the $\mathbf{v}$-vector should have d entries, and since it's a unit vector, it has $d-1$ degrees of freedom. If we employ d Householder matrices and use one scaling vector $\mathbf{s}$ (d-dimension), the total degrees of freedom $(d \times (d-1) + d)$ would be the same as $\mathbf{W}$. If the model obtains comparable performance using less than d Householder matrices, the fusion layer could achieve higher parameter efficiency.

To ensure the transform can be initialize close to the identity matrix, we use couples of $\mathbf{v}$-vectors:

$$\begin{aligned}\mathbf{P}_c = &(\mathbf{I_W} - 2\frac{(\mathbf{v}_{c,1} - \mathbf{v}_{c,2})(\mathbf{v}_{c,1} - \mathbf{v}_{c,2})^T}{||\mathbf{v}_{c,1} - \mathbf{v}_{c,2}||^2}) \times \\ &(\mathbf{I_W} - 2\frac{(\mathbf{v}_{c,1} + \mathbf{v}_{c,2})(\mathbf{v}_{c,1} + \mathbf{v}_{c,2})^T}{||\mathbf{v}_{c,1} + \mathbf{v}_{c,2}||^2})\end{aligned} \tag{4}$$

where $\mathbf{v}_{c,1}$ and $\mathbf{v}_{c,2}$ form the c-th couple. We initialize them with standard normally distributed values and rescale $\mathbf{v}_{c,1}$ to unit length and $\mathbf{v}_{c,2}$ to length $\frac{0.01}{\sqrt{d}}$. Then the final rotation matrix $\mathbf{P}_C$ using C $\mathbf{v}$-vector couples is written as:

$$\mathbf{P}_C = \prod_{c=1}^{C} \mathbf{P}_c \tag{5}$$

Finally, we can add a diagonal scaling matrix $\mathbf{\Sigma_s}$ using scaling vector $\mathbf{s}$ on its diagonal:

$$\mathbf{W}_C = \mathbf{\Sigma_s} * \mathbf{P}_C \quad (6)$$

Notice that this does not allow to build any square matrix, but seemed to suffice for obtaining good performance.

3. Experiments

3.1. Datasets

We use **CGN** dataset [26] (excluding the "a,c,d,e" components) as the canonical speech data to pretrain the ASR model. It contains more than 300-hour Dutch speech. For dysarthric speech, **Domotica** dataset [27] is used, which contains around 9-hour dysarthric Dutch speech from 17 speakers, in total 4173 utterances. The content is commands related to home automation, such as "turn on the light in the kitchen". The intelligibility scores of speakers are provided and we categorize speakers into 3 severity levels: high (score>80, 5 speakers), medium (80≥score>70, 7 speakers), low(70≥score, 5 speakers).

3.2. Training strategy

Training proceeds through the following steps (see figure 1(a)):

Step 1 Pretrain the transformer ASR model with canonical speech. Freeze the pretrained model.

Step 2 Insert one adapter into the last encoder layer and train it with $N-3$ source dysarthric speakers' data jointly.

Step 3 For each of N source dysarthric speakers, initialize the adapter from the one trained in **Step 2** and train each adapter using each source speaker's data only. Then freeze the N adapters.

Step 4 Insert the fusion layer after the N adapters and train with the target dysarthric speaker's data.

In the experiment, we divide 15 dysarthric speakers into 5 subsets and each subset includes 3 speakers with different severity levels. In each trial, we use four subsets as source speakers and one subset as the target speakers. The remaining two speakers (pp34, pp35) are consistently used as source speakers. Thus number of source speakers is $N = 14$. In **Step 2** we use 3 subsets of the source speakers and pp34, pp35 for training and 1 subset for validation. In **Step 3**, for each of the N source speakers, we use 90% data for training and 10% for validation. In **Step 4**, data of each target speaker is divided into five parts, and during fusion layer training, we utilize three parts (60%) for training, one for validation and one for testing in each fold. On average, 5% is around 1.5 minute per speaker and 60% is about 19 minutes. Our metric for speech recognition is Character Error Rate (CER) as it is more universal across tasks. 73 characters are used. The provided results are CER averaged over 15 speakers and five data-folds of each speaker.

3.3. Network and training setup

The speech features used in the model are 83-dimensional filter bank and pitch features. We implement the method based on the ESPnet toolkit [24]. The transformer encoder has 12 layers and the transformer decoder has 6 layers. For **Step 1**, we use a batch size of 64 and the "Noam" optimizer [8] with a learning rate of 10 and 25000 warm-up steps. The total number of training epochs is 230 and the final pretrained model is averaged over the 10 epochs with the highest validation accuracy. For **Step 2-4**, as well as in case of finetuning the model, we use the Adam optimizer with a learning rate of 0.001 and early stopping with a patience of 20. The batch size is 32. The final model is averaged over three checkpoints with highest validation accuracy. The dimension of the transformer encoder layer output is 256. The inner dimension (size of "Down projection") of the Adapter module is also chosen as 256 since this size gives the best performance in our preliminary experiments, meaning that the module is not a traditional bottleneck shape. $\mathbf{K}(\cdot)$ and $\mathbf{Q}(\cdot)$ in the fusion layer have an original size of 256 [22], and we use 64 as the reduced size. In the loss function, we set $\lambda_1 = 0.3$ and $\lambda_2 = 0.01$. Beam size 4 is employed for joint decoding.

4. Results

In this section, we provide the results of using Adapter Fusion with/without Householder transformation.

4.1. Adapter Fusion performance

We first compare the dysarthric speech recognition performance using different models:

Pretrain: Test the pretrained model obtained in **Step 1**.

FT-Enc: Finetune the pretrained model encoder with target speaker data.

FT-EncDec: Finetune the whole pretrained model with target speaker data.

Pretrain-Adpt: Test the model in **Step 2**.

Source-Adpt-avg: Test the model in **Step 3**, average outputs of all source adapters as the target speaker's output.

Target-Adpt: Finetune the pretrained adapter with target speaker data.

Fusion-256dAtt+W: Test the model in **Step 4**.

Fusion-64dAtt+W: Use fusion layer with reduced dimension 64 of $\mathbf{K}(\cdot)$ and $\mathbf{Q}(\cdot)$ in **Step 4** and test the model.

Fusion-64dAtt/W: Use fusion layer with eliminated value linear layer / attention score in **Step 4** and test the model.

Fusion-$\mathbf{W_{UV}}$/$\mathbf{W_{\Sigma}}$: Do SVD on matrix $\mathbf{W}$ in **Fusion-W**, replace it with $\mathbf{W_{UV}}$ / $\mathbf{W_{\Sigma}}$ and test model.

Table 1 provides the trainable parameter count and CER of these models. If the model is trained with target speaker data, the training data amounted to 60% of all data. Figure 2 shows the CER as a function of the amount of target training data.

To assess the effectiveness of the pretrained model in speech recognition, we test it on a new canonical speech dataset [28] and achieved a reasonable CER of 6.3%. When testing on dysarthric speech, we obtained a high CER due to domain mismatch. Finetuning methods yield the best recognition results when 60% data is used. However, as shown in figure 2, when only 5% of the data is used, the model is poorly trained with finetuning, resulting in a higher CER than other methods.

When we test the model with existing adapters, the pretrained adapter performs better than averaging source speaker adapters, as simple averaging might cause a performance drop due to the existence of individuality among adapters. Training a personalized adapter for the target speaker results in reasonable recognition results adding only 0.5% parameters, while a fusion layer produces an even better CER with more parameters than **Target-Adpt**. Both methods could not surpass the finetuning methods in our experiment setting when more than 5% data is used, except for the very-low-resource case (5%) due to the small parameter count. By reducing the size of the $\mathbf{K}(\cdot)$ and $\mathbf{Q}(\cdot)$ to 64, the CER is even further reduced while the layer has

fewer parameters. This might be because of overtraining of the larger model.

Table 1: *Number of target-specific trainable parameters and CER in % when using at most 60 % of data.*

Name	#para	CER
Pretrain	-	49.98
FT-Enc	17.7M	1.62
FT-EncDec	27.2M	1.39
Pretrain-Adpt	-	13.27
Source-Adpt-avg	-	14.08
Target-Adpt	131.5k	4.40
Fusion-256dAtt+W	197.6k	**2.85**
Fusion-64dAtt+W	98.6k	**2.61**
Fusion-64dAtt	33.0k	8.38
Fusion-W	65.5k	3.03
Fusion-$\mathbf{W_{UV}}$	-	6.62
Fusion-$\mathbf{W_{\Sigma}}$	-	13.34
Fusion-$\mathbf{P_{64}}$	32.8k	3.28
Fusion-$\mathbf{W_{64}}$	33.0k	3.19
Fusion-64dAtt+$\mathbf{W_{64}}$	66.1k	**2.79**

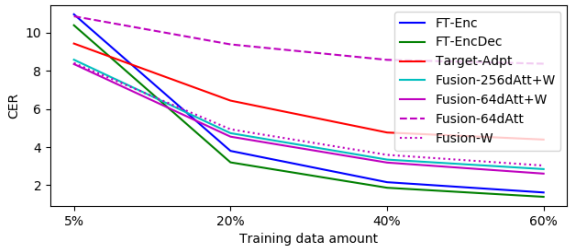


Figure 2: *CER results of different models when training on different amounts of target speaker data.*

To trace the performance improvement brought by the fusion layer, we ablate the value linear layer (**Fusion-64dAtt**) or the attention score (**Fusion-W**) in the fusion layer and then train it. The case **Fusion-W** gives a lower CER, indicating the greater importance of the value linear layer. This might be because the adaptation among dysarthric speakers has smaller differentiation than other tasks' adaptation [22, 19]. By performing SVD on the weight matrix $\mathbf{W}$ of the value linear layer, we separate it into rotation and scaling operations and evaluate the impact of each operation by testing the model **Fusion-W** with one single operation. Comparing **Fusion-$\mathbf{W_{UV}}$** and **Fusion-$\mathbf{W_{\Sigma}}$**, we find that the rotation plays a more important role in modifying the adapter outputs $\mathbf{Y}_A$. However, even using only the scaling vector still results in a performance improvement over no action (**Source-Adpt-avg**).

4.2. Adapter Fusion with Householder Transformation

To enhance parameter efficiency, we aim to reformulate the rotation operation in the matrix $\mathbf{W}$ using Householder transformation. Table 1 provides the number of trainable parameters and CER results using 64 $\mathbf{v}$-vector couples to form an orthogonal $\mathbf{P}_{64}$ matrix. In **Fusion-$\mathbf{P}_{64}$**, we use $\mathbf{P}_{64}$ only as the weight matrix, achieving 3.28% CER, while in **Fusion-$\mathbf{W_{64}}$**, we add the scaling as in equation (6), which further improves the CER. In **Fusion-64dAtt+$\mathbf{W_{64}}$**, we complete the model by adding the attention, yielding a CER of 2.79%, which is very close to the baseline model **Fusion-64dAtt+W** with two-thirds of its parameters and is even higher than baseline model **Fusion-256dAtt+W** with only one-third of its parameters.

Table 2 compares the CER for using different C values in model **Fusion-$\mathbf{W_C}$** with the baseline case **Fusion-W**. Our results show that using $C = 64$, the model **Fusion-$\mathbf{W_{64}}$** achieves similar CER as the baseline while using only half of the parameters. Table 2 also demonstrates that when we have sufficient training data (60%), increasing the number of $\mathbf{v}$-vectors (C value) always benefits performance, and it will reach an upper limit and doesn't outperform the baseline. When training data is limited (5%), increasing the C value will initially improve the model performance but will then suffer from a lack of training data as well. The results show that the Householder factorization is scalable way to trade off the target-specific model size for accuracy. Notice also that applying d Householder factors has a similar complexity as multiplication with a $d \times d$ matrix.

Table 2: *CER in % when using different C values in model **Fusion-$\mathbf{W_C}$**, compared with baseline case (**bl**) **Fusion-W**.*

Training data %	CER of **bl/Fusion-$\mathbf{W_C}$** when C =					
	bl	**1**	**2**	**8**	**64**	**128**
5%	8.42	10.90	10.48	9.62	8.90	9.20
60%	3.03	7.81	6.62	4.53	3.19	3.12

5. Conclusions

Dysarthric speech recognition and speaker adaptation face challenges due to data scarcity and huge diversity between dysarthric speakers. Finetuning, a common method of transferring knowledge from a rich resource domain (in our case canonical speech recognition), has drawbacks such as overfitting and high storage requirements for personalized use. Thanks to their small size and ease of use, Adapter modules offer a suitable solution. Adapter Fusion can boost knowledge transfer between learned source speaker adapters, but it may increase the number of parameters used.

In this study, we apply Adapter Fusion to target speaker adaptation for speech recognition, achieving acceptable CER results with significantly fewer trainable parameters than classical finetuning methods. We also analyze the performance improvement brought by the fusion layer and identify the critical role played by the rotation operation of the value linear layer weight matrix $\mathbf{W}$. Finally, we improve the parameter efficiency of the fusion layer by reducing the size of the query and key linear layer and reformulating $\mathbf{W}$ using Householder transformation. The proposed fusion layer achieves comparable recognition results as our starting point with only one third of the parameters.

In the future, we plan to further validate the generality of the proposed methods on additional datasets and different tasks. Additionally, we will explore solutions for the zero-shot case [29] of dysarthric speech recognition, taking practical scenarios into consideration where the model has no access to the target speakers during training.

6. Acknowledgements

The research was supported by KU Leuven Special Research Fund grant C24M/22/025 and the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

7. References

[1] D. Wang, J. Yu, X. Wu, L. Sun, X. Liu, and H. Meng, "Improved end-to-end dysarthric speech recognition via meta-learning based model re-initialization," in *2021 12th International Symposium on Chinese Spoken Language Processing (ISCSLP)*. IEEE, 2021, pp. 1–5.

[2] M. Geng, X. Xie, Z. Ye, T. Wang, G. Li, S. Hu, X. Liu, and H. Meng, "Speaker adaptation using spectro-temporal deep features for dysarthric and elderly speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 2597–2611, 2022.

[3] J. Tobin and K. Tomanek, "Personalized automatic speech recognition trained on small disordered speech datasets," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6637–6641.

[4] Z. Yue, E. Loweimi, Z. Cvetkovic, H. Christensen, and J. Barker, "Multi-modal acoustic-articulatory feature fusion for dysarthric speech recognition," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7372–7376.

[5] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and rnn-t loss," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7829–7833.

[6] A. Gulati, J. Qin, C.-C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu *et al.*, "Conformer: Convolution-augmented transformer for speech recognition," *Proc. Interspeech 2020*, pp. 5036–5040, 2020.

[7] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A framework for self-supervised learning of speech representations," *Advances in neural information processing systems*, vol. 33, pp. 12 449–12 460, 2020.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[9] J. Shor, D. Emanuel, O. Lang, O. Tuval, M. Brenner, J. Cattiau, F. Vieira, M. McNally, T. Charbonneau, M. Nollstadt *et al.*, "Personalizing asr for dysarthric and accented speech with limited data," *Proc. Interspeech 2019*, pp. 784–788, 2019.

[10] Y. Takashima, R. Takashima, T. Takiguchi, and Y. Ariki, "Knowledge transferability between the speech data of persons with dysarthria speaking different languages for dysarthric speech recognition," *IEEE Access*, vol. 7, pp. 164 320–164 326, 2019.

[11] P. Wang, B. BabaAli, and H. Van hamme, "A study into pre-training strategies for spoken language understanding on dysarthric speech," *Proc. Interspeech 2021*, 2021.

[12] R. Takashima, T. Takiguchi, and Y. Ariki, "Two-step acoustic model adaptation for dysarthric speech recognition," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6104–6108.

[13] R. Turrisi and L. Badino, "Interpretable dysarthric speaker adaptation based on optimal-transport," *arXiv preprint arXiv:2203.07143*, 2022.

[14] M. K. Baskar, T. Herzig, D. Nguyen, M. Diez, T. Polzehl, L. Burget, J. Černockỳ *et al.*, "Speaker adaptation for wav2vec2 based dysarthric asr," *Proc. Interspeech 2022*, pp. 3403–3407, 2022.

[15] K. Tomanek, V. Zayats, D. Padfield, K. Vaillancourt, and F. Biadsy, "Residual adapters for parameter-efficient asr adaptation to atypical and accented speech," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 6751–6760.

[16] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," *Advances in neural information processing systems*, vol. 30, 2017.

[17] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly, "Parameter-efficient transfer learning for nlp," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2790–2799.

[18] A. Kannan, A. Datta, T. N. Sainath, E. Weinstein, B. Ramabhadran, Y. Wu, A. Bapna, Z. Chen, and S. Lee, "Large-scale multilingual speech recognition with a streaming end-to-end model," *Proc. Interspeech 2019*, pp. 2130–2134, 2019.

[19] W. Hou, H. Zhu, Y. Wang, J. Wang, T. Qin, R. Xu, and T. Shinozaki, "Exploiting adapters for cross-lingual low-resource speech recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 30, pp. 317–329, 2021.

[20] R. Karimi Mahabadi, J. Henderson, and S. Ruder, "Compacter: Efficient low-rank hypercomplex adapter layers," *Advances in Neural Information Processing Systems*, vol. 34, pp. 1022–1035, 2021.

[21] R. Wang, D. Tang, N. Duan, Z. Wei, X.-J. Huang, J. Ji, G. Cao, D. Jiang, and M. Zhou, "K-adapter: Infusing knowledge into pre-trained models with adapters," in *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 2021, pp. 1405–1418.

[22] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych, "Adapterfusion: Non-destructive task composition for transfer learning," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 2021, pp. 487–503.

[23] A. S. Householder, "Unitary triangularization of a nonsymmetric matrix," *Journal of the ACM (JACM)*, vol. 5, no. 4, pp. 339–342, 1958.

[24] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Enrique Yalta Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "Espnet: End-to-end speech processing toolkit," *Proc. Interspeech 2018*, pp. 2207–2211, 2018.

[25] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 369–376.

[26] N. Oostdijk, W. Goedertier, F. Van Eynde, L. Boves, J.-P. Martens, M. Moortgat, and R. H. Baayen, "Experiences from the spoken dutch corpus project," in *LREC 2002*. European Language Resources Association, 2002, pp. 340–347.

[27] B. Ons, J. F. Gemmeke, and H. Van hamme, "The self-taught vocal interface," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2014, no. 1, pp. 1–16, 2014.

[28] L. Bollens, B. Accou, H. Van hamme, and T. Francart, "A large auditory eeg decoding dataset," 2023. [Online]. Available: https://doi.org/10.48804/K3VSND

[29] Q. Xu, A. Baevski, and M. Auli, "Simple and effective zero-shot cross-lingual phoneme recognition," *arXiv preprint arXiv:2109.11680*, 2021.