Assessing the Efficacy of Adapters in Cross-Language Transfer Learning For Low-Resource Automatic Speech Recognition

Yan Meng and Péter Mihajlik

Abstract—**In recent years, the application of adapter modules in large language models proved to be successful in reducing computing and memory costs during fine-tuning. In our paper, we apply adapters to the field of automatic speech recognition. Specifically, we add adapters to different pre-trained speech recognition models to evaluate their efficiency in cross-language transfer learning. In this study, the evaluations are extended to GPU memory consumption, training duration, and recognition accuracy. By comparing the effects of adapters added to different models, we further explore the impact of whether the foundational model was (pre-) trained in the target language.**

Index Terms—**Adapters, Whisper, Conformer, Fast Conformer, Cross-lingual transfer learning, speech recognition**

I. Introduction

With the implementation of neural networks in speech recognition, significant improvements have been achieved in neural speech-to-text (STT) models. When transformer [1] was initially proposed, it made significant progress mainly in machine translation tasks. Conformer [2] is an improvement and extension of the transformer model, used primarily in speech recognition tasks. Based on the LibriSpeech benchmark [3], Gulati et al. conducted experiments based on three different sizes of conformer models (small, medium, and large) [2]. The experimental results are improved by increasing the model parameters. The large-size Conformer model achieved excellent results with a word error rate (WER) of 2.1%/4.3% without using a language model and 1.9%/3.9% with an external language model on test/testother dataset. Fast Conformer [4] is a redesigned Conformer model with a novel downsampling schema, which is 2.8 times faster than the Conformer model. Fast Conformer is a newly proposed model, that has not been widely used in the training of low-resource data sets. Whisper [5] is a weakly-supervised model that can target multiple languages and tasks. The Whisper model has been trained on multiple language datasets, including Hungarian. Therefore, in automatic speech recognition tasks, high-quality results on specific distributions can be obtained without fine-tuning the Whisper model based on a Hungarian dataset.

Based on the English dataset, all the above models can achieve a lower word error rate (WER). For relatively low-resource languages such as Hungarian, direct training of ASR neural models from scratch is both time-consuming and may not result in optimal performances. Therefore, cross-lingual transfer learning methods are often applied for speech recognition tasks, especially for low-resource datasets ([6], [7], [8], [9], [10]). It means that when fine-tuning the model parameters, some pre-trained models are used to help models adapt to the distributional features of the language faster, e.g., fine-tuning the pre-trained English model based on the Hungarian dataset to obtain better WER results. However, as the number of model parameters increases, the cost required to fine-tune and train the model increases accordingly. Therefore, we need an approach to achieve better recognition performance of the model in a shorter time and with lower memory consumption.

To address these issues, especially for large-scale models, the Parameter Efficient Fine-Tuning (PEFT) method [11] has been proposed. Adapter is one of the core fine-tuning methods of the PEFT technique, which is a new module added between layers of a pre-trained network [11]. Adapter modules have two main features: a small number of parameters, and a near-identity initialization [12]. In every training step, the parameters of the original foundational model remain frozen, only all the parameters in the adaptors module are tuned. However their number is relatively small as compared to the foundational model parameters. This method effectively reduces the consumption of GPU memory in the training process. An increasing number of adapter types have been proposed for different types of basic models in ASR [13], [14]. In [15], Hou, et al. proposed a new adapter algorithm-based transformer structure for cross-lingual transfer learning, SimAdapter, and MetaAdapter, which was applied to parameter efficient cross-lingual speech adaptation. To explore the effect of adapters applied to the basic model, in [16], adapters were applied to both Transformer and Conformer architectures to comprehensively evaluate the adapter performance within the context of children's ASR. Huang, et al. [17] demonstrated that integrating adapters into End-to-End model can effectively mitigate catastrophic forgetting (CF), which is a common drawback of improving models through fine-tuning.

In this paper, we focus on studying the effect of adapters applied to the foundational model in cross-lingual transfer

Yan Meng is with the department of Telecommunications and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary (e-mail: yan.meng@edu.bme.hu).

Péter Mihajlik is with the department of Telecommunications and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary and the HUN-REN Hungarian Research Centre for Linguistics, Budapest, Hungary (e-mail: mihajlik@tmit.bme.hu).

DOI: 10.36244/ICJ.2024.4.1

TABLE I
CHARACTERISTICS OF DATABASE (BEA-BASE + COMMONVOICE).

			BEA-Base			CV12
	train-114	**dev-repet**	**dev-spont**	**eval-repet**	**eval-spont**	**test**
Length [hours]	71.2	0.65	4.02	0.95	4.91	6.80
Num of speakers	114	10	10	16	16	251
Num of segments	76 881	568	4 893	858	5 69	4 871
Num of characters	3.1M	28 467	154 994	43 448	197 738	250 709
Num of words	0.56M	4 110	27 939	6 229	35 178	35 485

learning based on the Hungarian dataset BEA-Base [18]. Meanwhile, by comparing different foundational models, we explore whether a foundational model pre-trained in the target language affects the effectiveness of adapter addition. Foundational models used in this paper are mainly Conformer-CTC, Fast Conformer [4], and Whisper [5]. All experiments are divided into two main aspects: first, the adapter module is added directly to the pre-trained model. Secondly, the adapter module is added to the model that has been fully fine-tuned. With these two types of experiments and using three different foundational models, we conclude that adapters significantly reduce GPU memory consumption. The application of adapter modules provided better results than full fine-tuning only in the Whisper model, suggesting that adding the adapter to models (pre-) trained already on the target language can perform better.

II. DATABASE AND BASELINE

A. Database

Throughout the experiment, the training and test dataset used is the Hungarian benchmark dataset BEA-Base [18]. The BEA-Base dataset contains both spontaneous speech and conversations with repeated elements, making it suitable for ASR research. The detailed dataset information of BEA-Base is shown in Table I, including training set train-114, verification set dev-spont, and test set eval-spont. In addition, the datasets include the repeated speech datasets dev-repet and eval-repet, as shown in Table I. To assess the generality of the model, an additional test dataset CommonVoice (CV) Hungarian v12.0 [19] was used in the study, which differs from BEA-Base in terms of recording conditions and speaker/speech diversity.

B. Fine-tuning baseline

The experimental baseline model comes from the paper [20], and the specific parameters can be referred to as the parameter setting in the paper. The baseline model is Conformer-CTC ([21], [22]) model from NeMo toolkit v1.62[1]. Conformer-CTC is a speech recognition model that combines Conformer architecture [2] and Connectionist Temporal Classification (CTC) [21] technology. The sizes of the Conformer-CTC model cited in this article are medium and large. The pre-trained model is in English, and the fine-tuning process uses the Hungarian dataset train-114 from BEA-Base for training. The results are shown in the first row of the two Conformer models in Table III.

The experiments were conducted on two distinct server configurations. The first configuration employed dual A6000 NVIDIA graphics cards, specifically designed for large-scale model experiments. Each card featured a substantial 48GB of memory and consumed 300 watts of power. In contrast, the second configuration utilized two NVIDIA Ge-Force GTX 1070 graphics cards tailored for lighter computational workloads. These cards were equipped with 8GB of memory, making them suitable for less memory-demanding tasks.

III. INCORPORATING ADAPTERS INTO ENGLISH PRE-TRAINED MODELS

In this section, Conformer-CTC model and adapter module are selected, provided from NVIDIA's NeMo toolkit v1.15.0[2]. There are two types of adapters used in experiments: Linear Adapter [11] and Multi-Head Attention Adapter [23]. In the subsequent exposition, the attention adapter is denoted as a tiny-attention adapter. The name of the adapter module is taken from the type of adapter function [12]. During fine-tuning experiments with adapters, only the parameters of the adapter module are updated, while the parameters of the original language model remain frozen. The linear adapter is a simple bottleneck structure feed-forward module [11]. Multi-Head attention adapter is an adapter model that combines multi-head self-attention [24] mechanism. The multi-head self-attention mechanism allows the model to focus on different parts of the input sequence separately under different contexts and positions.

Fast Conformer (FC) [4] was also selected for this experiment as the foundational model. But for the Fast Conformer experiment, all pre-trained models are provided by NVIDIA's NeMo toolkit v1.22.0[3].

A. Pre-trained Model with Adapters

For the Conformer experiments, we used three sizes of English pre-trained model provided by NVIDIA NeMo toolkit (STT En Conformer-CTC XLarge[4], STT En Conformer-CTC Large[5], STT En Conformer-CTC Medium[6]). During the ex-

[1] https://github.com/NVIDIA/NeMo/tree/v1.6.2

[2] https://github.com/NVIDIA/NeMo/tree/v1.15.0

[3] https://github.com/NVIDIA/NeMo/tree/v1.22.0

[4] https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_conformer_ctc_xlarge

[5] https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_conformer_ctc_large

[6] https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_conformer_ctc_medium

TABLE II
CER(%) / WER(%) RESULTS BASED ON FINE-TUNING THE ENGLISH PRE-TRAINED CONFORMER AND FAST CONFORMER MODEL WITH LINEAR ADAPTER ADDED.

Foundational model	**Total num params**	**Trainable params**		**BEA-Base**			**CV12**
			dev-repet	**dev-spont**	**eval-repet**	**eval-spont**	**test**
FastConformer-Large	116M	0.52%	11.30 / 28.18	16.73 / 33.61	13.21 / 31.14	17.95 / 35.87	17.90 / 39.48
FastConformer-XLarge	610M	0.26%	8.84 / 23.50	14.40 / 31.11	10.56 / 26.70	15.14 / 32.25	16.13 / 37.28
Conformer-Medium	31.1M	0.98%	8.71 / 23.35	11.41 / 26.25	9.20 / 23.97	12.20 / 27.93	14.75 / 35.18
Conformer-Large	122M	0.50%	5.72 / 17.22	8.79 / 21.27	6.32 / 18.51	9.40 / 22.55	10.58 / 27.27
Conformer-XLarge	637M	0.25%	**4.19 / 12.21**	**8.29 / 20.03**	**4.77 / 13.28**	**8.73 / 21.08**	**9.53 / 25.22**

TABLE III
CER(%) / WER(%) RESULTS BASED ON TWO ROUNDS OF FINE-TUNING EXPERIMENTS ON ENGLISH PRE-TRAINED CONFORMER AND FAST CONFORMER MODELS OF DIFFERENT SIZES.

Foundational model	**Total num params**	**Trainable params**	**Adapter type**		**BEA-Base**			**CV12**
				dev-repet	**dev-spont**	**eval-repet**	**eval-spont**	**test**
FastConformer-Large	115M	100.00%	-	1.25 / 5.46	5.84 / 17.23	1.31 / 5.06	6.30 / 18.20	11.56 / 39.13
	116M	0.52%	Linear	1.25 / 5.43	5.83 / 17.18	1.30 / 5.02	6.32 / 18.19	11.55 / 39.02
	115M	0.06%	tiny-attention	1.23 / 5.36	5.82 / 17.20	1.30 / 5.05	6.28 / 18.19	11.54 / 39.00
FastConformer-XLarge	608M	100.00%	-	1.45 / 6.08	5.80 / 17.63	1.68 / 6.39	6.03 / 18.32	9.76 / 35.80
	610M	0.26%	Linear	1.46 / 6.11	5.79 / 17.60	1.69 / 6.44	6.07 / 18.38	9.81 / 35.87
	608M	0.03%	tiny-attention	1.47 / 6.11	5.79 / 17.59	1.70 / 6.45	6.06 / 18.37	9.80 / 35.88
Conformer-Medium	30.5M	100.00%	-	1.72 / 8.56	5.58 / 18.44	2.15 / 9.68	5.82 / 19.60	8.37 / 35.57
	30.8M	0.99%	Linear	1.72 / 8.54	5.58 / 18.44	2.15 / 9.65	5.83 / 19.61	8.36 / 35.57
	30.7M	0.12%	tiny-attention	1.72 / 8.56	5.58 / 18.44	2.15 / 9.68	5.82 / 19.60	8.37 / 35.57
Conformer-Large	121M	100.00%	-	**1.13 / 5.45**	**5.09 / 16.45**	**1.27 / 5.28**	**5.28 / 17.23**	**8.78 / 34.85**
	122M	0.50%	Linear	1.13 / 5.47	5.10 / 16.45	1.26 / 5.25	5.28 / 17.22	8.77 / 34.80
	121M	0.06%	tiny-attention	1.14 / 5.47	5.12 / 16.42	1.28 / 5.30	5.30 / 17.34	8.75 / 34.79

periments, we only use linear adapter modules combined with the English pre-trained model and then fine-tune the model for experiments based on the Hungarian BEA-Base train-114 dataset. For data augmentation, SpecAugment [25] and speed perturbation were applied, using the same configuration as [20]. Throughout the fine-tuning experimental process, for each experiment, we set the batch size to 16, and the learning rate to 0.001, and ran it on a GPU of the A6000 server for 100 epochs.

For the Fast Conformer experiments, we mainly used extra large and large-sized pre-trained English Fast Conformer models as foundational models (STT En Fast Conformer-CTC XLarge[7], STT En Fast Conformer-CTC Large[8]) and linear type adapter module. We use the same linear adapter module as the Conformer model experiment above. During the experiments, the batch size was set to 16, the learning rate was set to 0.001, and 100 training epochs were performed on one GPU of the A6000 server. Otherwise, the other settings were the same as the Conformer experiments described above. The experimental outcomes of the Conformer and Fast Conformer model are presented in Table II.

B. Fine-tuned Model with Adapters

This section focuses on adding adapters to the original model that has been fully fine-tuned based on the BEA-Base dataset to further explore the impact of adding adapters on the fully fine-tuned model. The whole experimental process consists of two rounds of fine-tuning experiments. First, the pre-trained English model was fully fine-tuned, and then the adapter module was added to the fully fine-tuned model to fine-tune it again. For the Conformer model, in the first round of the fine-tuning phase, we used the same parameters as [20], batch size of 32, 200 training epochs, etc. In the second fine-tuning phase, two types of adapter modules were used, linear and tiny-attention. During the fine-tuning process, we set the learning rate to 0.01, training epochs to 100, and other parameters consistent with the pre-trained model experiments in the previous subsection, batch size of 16. The experimental results are shown in Table III. For the Fast Conformer model, in the first fine-tuning phase, the batch size was 96, the learning rate was 0.01, and the experiments were trained in 150 epochs. In the second fine-tuning phase, two types of adapters were used, with a batch size of 96, a learning rate of 0.02, and 50 epochs training. The results of this experiment are displayed in Table III.

C. Result Analysis

Figure 1 shows the memory consumption of the two fine-tuning methods in the Conformer model within a single epoch. The blue line represents a large-sized Conformer model, and the orange line represents a medium-sized Conformer model. The dotted line represents the experimental results of full fine-tuning of the original pre-trained model, while the solid line represents the experimental results of fine-tuning the original

[7] https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_fastconformer_ctc_xlarge

[8] https://catalog.ngc.nvidia.com/orgs/nvidia/teams/nemo/models/stt_en_fastconformer_ctc_large

pre-trained model with the addition of a linear adapter to it. Throughout the experiment, the batch size of training was set to 16, and GPU memory utilization was logged at intervals of 5 seconds. The results show that, regardless of model size, training with a linear adapter takes significantly less memory than direct full fine-tuning the Conformer model. In addition, when calculating the training duration of one epoch, it is clear that the training duration is relatively short for the linear adapter experiments.

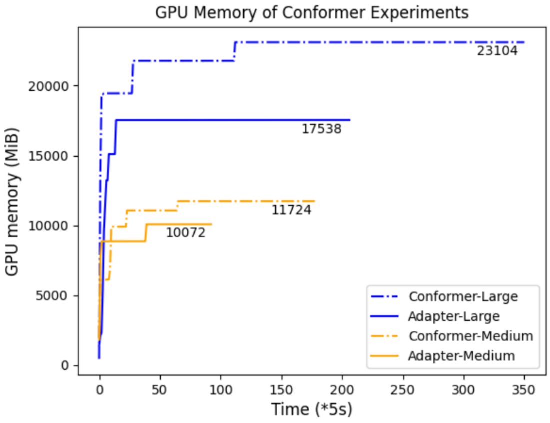

Fig. 1. GPU memory consumption from fine-tuning experiments on medium and large-sized English pre-trained Conformer models. Direct full fine-tuning the English pre-trained model is denoted by the dotted line, whereas incorporating adapters into the pre-trained model for fine-tuning is illustrated with the solid line.

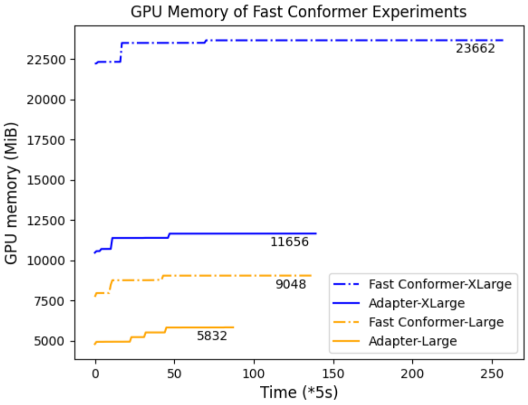

Fig. 2. GPU memory consumption from fine-tuning experiments on large-sized English pre-trained Fast Conformer models. Direct full fine-tuning the English pre-trained model is shown by the dotted line, whereas incorporating adapters into the pre-trained model for fine-tuning is illustrated by the solid line.

Figure 2 shows the GPU memory recorded every 5 seconds in two fine-tuning experiments using the English pre-trained extra large and large-sized Fast Conformer model. The dotted line indicates fully fine-tuning the pre-trained model directly, and the solid line indicates that the linear adapter is added to the pre-trained model for fine-tuning. The training batch size is also set as 16.

TABLE IV
Results show the maximum GPU memory consumed in the fine-tuning experiments for Conformer and Fast Conformer models. Corresponds to the values in Figure 1 and 2.

Foundational model	Adapter	GPU memory (MiB)
Conformer-Large	-	23 104
	Linear	17 538
Conformer-Medium	-	11 724
	Linear	10 072
FastConformer-XLarge	-	23 662
	Linear	11 656
FastConformer-Large	-	9 048
	Linear	**5 832**

As can be observed from Figure 2, for fine-tuning linear adapter experiments, both the training duration and memory usage required were significantly reduced compared to the full fine-tuning experiment. Comparing the results of Figure 1 and Figure 2, it can be clearly seen that the training time of the Fast Conformer model is much shorter than that of the traditional Conformer model.

Observing Table II, When fine-tuning a cross-language pre-trained model with an adapter added, lower word error rates can also be achieved. However, for Fast Conformer model experiments, the results obtained are significantly worse than those of the Conformer model. We infer that the addition of the adapter to the Fast Conformer resulted in a poorer convergence of the overall model. By comparing the results of full fine-tuning experiments in Table III, the Fast Conformer results in a lower WER in a shorter training duration relative to the Conformer model, demonstrating the superiority of the Fast Conformer model. The overall analysis of Table III shows that for adding separately two types of adapter modules into the Fast Conformer and Conformer model, the experimental results have a relatively limited improvement on both CV and the BEA-Base datasets.

Comparing the results in the two tables, we found that adding adapters to the Conformer model and training for cross-lingual transfer learning did not show superior performance over the full fine-tuning results on the BEA-Base dataset. But it is worth noting that in the experiment of adding an adapter into the pre-trained Conformer model, the Xlarge-sized and large-sized models achieved lower character/word error rates (CER/WER) on the CommonVoice dataset, 9.53%/25.22% and 10.58%/27.27% respectively. By comparing the results in Table II (27.27%) and Table III (34.85%), the WER of the large-sized model on the CommonVoice dataset relatively decreased by about 21.7% when incorporating adapter modules. Comparing the results of the Fast Conformer model in the two tables, adding the adapter directly to the English pre-trained Fast Conformer model does not provide better results based on the BEA-Base dataset. Nevertheless, on the CommonVoice dataset, we still obtained similar WER results (39.48%) compared to the full fine-tuning results (39.13%). We infer that adding adapters to the foundational model for cross-lingual transfer learning did not significantly improve the word error rate on the original dataset. However, the model performs consistently on multiple datasets with wider

TABLE V
CER(%) / WER(%) ZERO-SHOT RESULTS OF DIFFERENT SIZES WHISPER MODEL ON HUNGARIAN DATASET.

Model	**Total num**		**BEA-Base**			**CV12**
	params	**dev-repet**	**dev-spont**	**eval-repet**	**eval-spont**	**test**
whisper-small	242M	6.71 / 32.99	19.67 / 41.25	7.47 / 35.21	20.22 / 41.80	9.83 / 41.05
whisper-medium	764M	4.82 / 21.92	17.97 / 37.18	5.18 / 22.33	19.46 / 38.67	6.91 / 27.61
whisper-large-v2	1.54B	**3.74 / 17.54**	**17.06 / 33.17**	**3.99 / 18.04**	**17.06 / 32.76**	**5.27 / 20.41**

applicability and robustness. In contrast, fine-tuning tends to overfit the current dataset, resulting in poorer performance on external datasets. In addition, by comparing the results in two figures, we concluded that the adapter approach significantly reduces training time and GPU memory consumption.

IV. INCORPORATING ADAPTERS INTO MULTILINGUAL PRE-TRAINED MODELS

In the study of adding the adapter module into multilingual pre-trained models, we chose the typical weakly supervised model Whisper [5] as the foundational model. Different from the Conformer model experiments, the Whisper model has been trained on multiple languages, including Hungarian, and thus fine-tuning is a multilingual to monolingual transfer learning process.

This section carries out related experiments by using three Whisper models of different sizes. First, this study directly evaluate the speech recognition accuracy of Whisper models on Hungarian datasets. The CER and WER in the experimental results in Table V are normalized results. The models and codes used are from the Speech Brain [26], [27]. Among the zero-shot results, the experimental results of Whisper-large-V2 and Whisper-medium were quoted from paper [20]. Second, fine-tuning the Whisper model on the Hungarian dataset. Last, two types of Parameter-Efficient Fine-Tuning (PEFT) methods [11], namely Low-Rank Adaptation (LoRA) [28] and Adaptive Low-Rank Adaptation (AdaLoRA) [29] were used for the experiment. The implementation of the PEFT method aims to minimize the number of parameters while maintaining the performance of the model, thus improving the parameter efficiency. The code we used for train and evaluating the Whisper experiments in this section is available at https://github.com/MengYan0901/Whisper-Experiments.

This section focuses on the effect of integrating the Whisper model with two PEFT methods on the accuracy of speech recognition in Hungarian. Compared with the experimental group of the Conformer model, the effect of adding adapters to the foundational model trained by the target language in transfer learning is further discussed. Different sizes of Whisper models (Large-V2[9], Medium[10], Small[11]) were used and trained based on the BEA-Base train-114 dataset. In the testing phase, the CV test set and the BEA-Base test set were used to evaluate the performance of the model. Tests on multiple datasets aim to more comprehensively examine the experimental results of model training.

A. Fine-tuning Whisper model

In this experiment, Whisper models are fine-tuned across three distinct scales. Specifically, for the Whisper-large-v2 model, the approach adopted in Paper [20] was used, where the encoder part was frozen and only the decoder part was fine-tuned. Conversely, for the Whisper-medium and Whisper-small models, fine-tune the parameters of the entire model. The outcomes of these experiments are shown in Table VI. The results of Whisper-V2 and Whisper-Medium are cited from the paper [20]. Regarding the experiments of the Whisper-Small model, the learning rate is set to 3e-4, the batch size to 16, and training is conducted for 5 epochs on one GPU of the A6000 server. The results distinctly demonstrate a notable decrease in word error rate (WER) as the number of model parameters increased, showing significant improvement, particularly within the CommonVoice dataset.

B. Incorporating Adapters into Whisper Model

Low-Rank Adaptation (LoRA) [28] is a popular Parameter-Efficient Fine-Tuning (PEFT) method [11]. When LoRA is used for downstream fine-tuning tasks, the parameters of the foundational model remain frozen throughout the training process, while the trainable rank decomposition matrices are integrated into each layer of the model transformer architecture [28]. This strategy significantly reduces the trainable parameters of the model. As indicated in the ”Trainable params” column of Table VII, trainable parameters are only approximately 1% of the total model parameters, thereby reducing training duration and GPU memory usage. Adaptive Low-Rank Adaptation (AdaLoRA) [29] is a derivative of LoRA that manages the count of parameters introduced by LoRA. Throughout training, AdaLoRA will allocate parameters to different weight matrices based on the degree of adaptation to the task. The weight matrix that is more adaptable to the task will be assigned more parameters for training. As shown in the Trainable params column of Table VIII, trainable parameters only account for about 0.5% of all model’s parameters.

Our training procedure follows the same configuration as the last fine-tuning section, but we change the learning rate to 1e-3 for both LoRA and AdaLoRA experiments, and the batch size remains unchanged at 16, still running on one GPU of the A6000 server for 5 epochs. By integrating two types of adapters (namely LoRA and AdaLoRA) into the Whisper model, the experimental results given in Tables VII and VIII show the impact of their application.

[9] https://huggingface.co/openai/whisper-large-v2
[10] https://huggingface.co/openai/whisper-medium
[11] https://huggingface.co/openai/whisper-small

TABLE VI
CER(%) / WER(%). RESULTS BASED ON DIRECT FINE-TUNING THE WHISPER MODEL ON THE HUNGARIAN DATASET.

Model	Total num params	Trainable params	BEA-Base				CV12
			dev-repet	dev-spont	eval-repet	eval-spont	test
whisper-small	242M	100%	3.15 / 12.21	9.70 / 26.44	4.17 / 14.35	10.56 / 28.63	20.59 / 58.23
whisper-medium	764M	100%	1.31 / 5.38	7.96 / 18.83	1.50 / 4.90	9.33 / 20.60	7.83 / 27.93
whisper-large-v2	1.54B	58.75%	**1.01 / 4.45**	**7.10 / 16.96**	**1.23 / 4.37**	**8.46 / 18.69**	**6.19 / 23.69**

TABLE VII
CER(%) / WER(%). RESULTS BASED ON INCORPORATING LOW-RANK ADAPTATION (LORA) FOR FINE-TUNING ON THE HUNGARIAN DATASET, UTILIZING WHISPER AS THE FOUNDATIONAL MODEL.

Founditional model	Total num params	Trainable params	BEA-Base				CV12
			dev-repet	dev-spont	eval-repet	eval-spont	test
whisper-small	245M	1.44%	3.37 / 15.47	7.48 / 23.07	3.71 / 15.40	8.75 / 25.59	10.25 / 40.21
whisper-medium	773M	1.22%	2.12 / 10.02	5.80 / 17.91	2.61 / 10.71	6.36 / 19.37	7.66 / 31.29
whisper-large-v2	1.56B	1.01%	**1.65 / 7.13**	**4.74 / 14.89**	**1.68 / 6.66**	**5.08 / 15.56**	**5.91 / 24.22**

TABLE VIII
CER(%) / WER(%). RESULTS BASED ON INCORPORATING ADAPTIVE LOW-RANK ADAPTATION (ADALORA) FOR FINE-TUNING ON THE HUNGARIAN DATASET, UTILIZING WHISPER AS THE FOUNDATIONAL MODEL.

Founditional model	Total num params	Trainable params	BEA-Base				CV12
			dev-repet	dev-spont	eval-repet	eval-spont	test
whisper-small	243M	0.55%	4.32 / 19.49	9.21 / 26.72	4.84 / 20.23	9.35 / 27.46	10.71 / 41.68
whisper-medium	767M	0.46%	2.40 / 10.68	5.72 / 17.67	2.55 / 10.80	6.24 / 18.97	6.62 / 27.49
whisper-large-v2	1.55B	0.38%	**1.73 / 8.15**	**4.86 / 15.17**	**1.81 / 7.53**	**5.21 / 15.77**	**5.47 / 22.64**

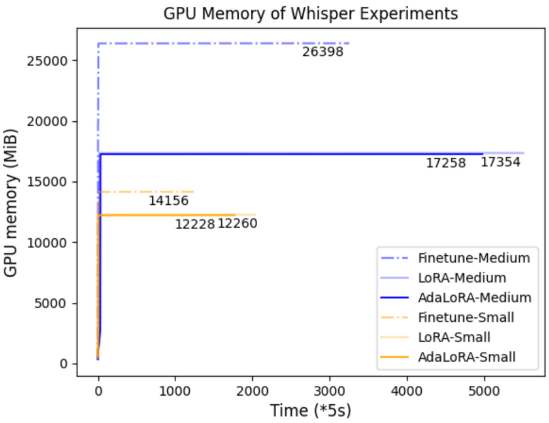

Fig. 3. GPU memory consumption from fine-tuning experiments on medium-sized and large-sized Whisper models in one epoch. Direct fine-tuning is indicated by the dotted line, incorporating LoRA into the model for fine-tuning is illustrated by the solid light blue line, and incorporating AdaLoRA into the model for fine-tuning is shown by the solid blue line.

TABLE IX
RESULTS SHOWS THE MAXIMUM GPU MEMORY CONSUMED IN THE FINE-TUNING EXPERIMENTS FOR WHISPER MODELS (CORRESPONDS TO THE VALUE IN THE FIGURE 3).

Foundational model	Adapter	GPU memory (MiB)
whisper-medium	-	26 398
	LoRA	17 354
	AdaLoRA	17 258
whisper-small	-	14 156
	LoRA	12 260
	AdaLoRA	**12 228**

C. Result Analysis

In Tables VI, VII and VIII, the data in bold font represent the best results obtained in the three experiments. It is clear that the performance of the model improves with increasing the number of model parameters whether LoRA or AdaLoRA is used. Comparing results in three tables, particularly in spontaneous speech tasks, both LoRA and AdaLoRA demonstrate superior performance, with the lowest WER 14.85%/15.16% on the dev-spont/eval-spont datasets of the BEA-Base dataset.

Figure 3 shows the memory occupation when fine-tuning the Whisper model in three different ways. Since the experiments with the whisper-large-V2 model were only trained on the decoder part of the model's parameters, it is not suitable for memory occupation comparisons. The results in Table IX correspond to the maximum memory occupied by each experiment in Figure 3. It can be observed from Figure 3 and Table IX that memory occupation can be significantly reduced when fine-tuning the Whisper model with LoRA or AdaLoRA. Especially for the medium-sized model, the memory occupation can be reduced by approximately 1/3, and the effect of reducing memory occupation is more significant as the model parameters increase. By comparing the CER/WER results of three tables, it can be concluded that for the Whisper model, the addition of an adapter module achieves a reduction in memory occupation and an improvement in speech recognition accuracy.

V. Conclusion

In this paper, we added adapter modules to different foundational models for automatic speech recognition tasks. When adding the adapter module to the foundational model that has not been trained in the target language, the test accuracy on the BEA-Base dataset was decreased while a significant improvement could be obtained on the CV dataset. We inferred that adapters can preserve the model's generalization ability without over-fitting the model to a specific training dataset, thus helping to maintain model excellence across multiple datasets. For a multilingual Whisper model (pre-) trained in Hungarian, adding the adapter module significantly reduced the word error rate on both datasets. The comparison of these two models shows that the addition of adapters can benefit from the foundational model (pre-) trained on the target language and achieve higher recognition accuracy. Furthermore, the addition of the adapter module showed a significant reduction in GPU memory consumption for all models during fine-tuning. Future research will further explore the adapter's efficacy for adding to other ASR models in cross-language transfer learning.

Acknowledgment

This research benefited greatly from the support provided by the Hungarian Linguistic Research Center in the development of the BEA-Base dataset. This work was supported partially by NKFIH-828- 2/2021(MILab), by the NVIDIA Academic Hardware Grant, and by the NKFIH K143075 and K135038 projects of the NRDI Fund. Thanks are also extended to the Budapest University of Technology and Economics and NVIDIA Academic Hardware Grant for their vital contribution including but not limited to hardware support.

References

[1] 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, **doi**: 10.48550/arXiv.1706.03762.

[2] 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," *arXiv preprint arXiv:2005.08100*, 2020, **doi**: 10.48550/arXiv.2005.08100.

[3] V. Panayotov, G. Chen, D. Povey, and S. 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, **doi**: 10.1109/ICASSP.2015.7178964.

[4] D. Rekesh, N. R. Koluguri, S. Kriman, S. Majumdar, V. Noroozi, H. Huang, O. Hrinchuk, K. Puvvada, A. Kumar, J. Balam et al., "Fast conformer with linearly scalable attention for efficient speech recognition," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2023, pp. 1–8, **doi**: 10.1109/ASRU57492.2023.1234567.

[5] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. Mcleavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proceedings of the 40th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, Eds., vol. 202. PMLR, 23–29 Jul 2023, pp. 28 492–28 518, **doi**: 10.48550/arXiv.2212.04356.

[6] S. Bansal, H. Kamper, K. Livescu, A. Lopez, and S. Goldwater, "Pre-training on high-resource speech recognition improves low-resource speech-to-text translation," *arXiv preprint arXiv:1809.01431*, 2018, **doi**: 10.48550/arXiv.1809.01431.

[7] C. Yu, Y. Chen, Y. Li, M. Kang, S. Xu, and X. Liu, "Cross-language end-to-end speech recognition research based on transfer learning for the low-resource tujia language," *Symmetry*, vol. 11, no. 2, p. 179, 2019, **doi**: 10.3390/sym11020179.

[8] J. Huang, O. Kuchaiev, P. O'Neill, V. Lavrukhin, J. Li, A. Flores, G. Kucsko, and B. Ginsburg, "Cross-language transfer learning, continuous learning, and domain adaptation for end-to-end automatic speech recognition," *arXiv preprint arXiv:2005.04290*, 2020, **doi**: 10.48550/arXiv.2005.04290.

[9] J. Luo, J. Wang, N. Cheng, E. Xiao, J. Xiao, G. Kucsko, P. O'Neill, J. Balam, S. Deng, A. Flores et al., "Cross-language transfer learning and domain adaptation for end-to-end automatic speech recognition," in *2021 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 2021, pp. 1–6, **doi**: 10.1109/ICME51207.2021.9428334.

[10] J. Zhao and W.-Q. Zhang, "Improving automatic speech recognition performance for low-resource languages with self-supervised models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 16, no. 6, pp. 1227–1241, 2022, **doi**: 10.1109/JSTSP.2022.3184480.

[11] 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, **doi**: 10.48550/arXiv.1902.00751.

[12] J. He, C. Zhou, X. Ma, T. Berg-Kirkpatrick, and G. Neubig, "Towards a unified view of parameter-efficient transfer learning," *arXiv preprint arXiv:2110.04366*, 2021, **doi**: 10.48550/arXiv.2110.04366.

[13] Y. Yu, C.-H. H. Yang, J. Kolehmainen, P. G. Shivakumar, Y. Gu, S. R. R. Ren, Q. Luo, A. Gourav, I.-F. Chen, Y.-C. Liu et al., "Low-rank adaptation of large language model rescoring for parameter-efficient speech recognition," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*. IEEE, 2023, pp. 1–8, **doi**: 10.1109/ASRU57964.2023.10389632.

[14] N. Inoue, S. Otake, T. Hirose, M. Ohi, and R. Kawakami, "Elp-adapters: Parameter efficient adapter tuning for various speech processing tasks," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 2024, **doi**: 10.1109/TASLP.2024.3434445.

[15] 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, **doi**: 10.1109/TASLP.2021.3138674.

[16] T. Rolland and A. Abad, "Exploring adapters with conformers for children's automatic speech recognition," in *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2024, pp. 12 747–12 751, **doi**: 10.1109/ICASSP48485.2024.10447091.

[17] S. Vander Eeckt and H. Van Hamme, "Using adapters to overcome catastrophic forgetting in end-to-end automatic speech recognition," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5, **doi**: 10.1109/ICASSP49357.2023.10095837.

[18] P. Mihajlik, A. Balog, T. E. Graczi, A. Kohari, B. Tarján, and K. Mady, "BEA-base: A benchmark for ASR of spontaneous Hungarian," in *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, Jun. 2022, pp. 1970–1977, **doi**: 10.48550/arXiv.2202.00601.

[19] R. Ardila, M. Branson, K. Davis, M. Henretty, M. Kohler, J. Meyer, R. Morais, L. Saunders, F. M. Tyers, and G. Weber, "Common voice: A massively-multilingual speech corpus," *arXiv preprint arXiv:1912.06670*, 2019, **doi**: 10.48550/arXiv.1912.06670.

[20] P. Mihajlik, M. S. Kádár, G. Dobsinszki, Y. Meng, M. Kedalai, J. Linke, T. Fegyó, and K. Mády, "What kind of multi-or cross-lingual pre-training is the most effective for a spontaneous, less-resourced asr task?" in *2nd Annual Meeting of the Special Interest Group on Under-resourced Languages: SIGUL 2023*, 2023, **doi**: 10.21437/SIGUL.2023-13.

[21] 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, DOI: 10.1145/1143844.1143891.

[22] 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,” *arXiv preprint arXiv:2005.08100*, 2020, DOI: 10.48550/arXiv.2005.08100.

[23] H. Zhao, H. Tan, and H. Mei, “Tiny-attention adapter: Contexts are more important than the number of parameters,” *arXiv preprint arXiv:2211.01979*, 2022, DOI: 10.48550/arXiv.2211.01979.

[24] E. Voita, D. Talbot, F. Moiseev, R. Sennrich, and I. Titov, “Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the 7 rest can be pruned,” *arXiv preprint arXiv:1905.09418*, 2019, DOI: 10.48550/arXiv.1905.09418e.

[25] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, “Specaugment: A simple data augmentation method for automatic speech recognition,” *arXiv preprint arXiv:1904.08779*, 2019, DOI: 10.48550/arXiv.1904.08779.

[26] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, “SpeechBrain: A general-purpose speech toolkit,” 2021, DOI: 10.48550/arXiv.2106.04624.

[27] M. Ravanelli, T. Parcollet, A. Moumen, S. de Langen, C. Subakan, P. Plantinga, Y. Wang, P. Mousavi, L. D. Libera, A. Ploujnikov, F. Paissan, D. Borra, S. Zaiem, Z. Zhao, S. Zhang, G. Karakasidis, S.-L. Yeh, P. Champion, A. Rouhe, R. Braun, F. Mai, J. Zuluaga-Gomez, S. M. Mousavi, A. Nautsch, X. Liu, S. Sagar, J. Duret, S. Mdhaffar, G. Laperriere, M. Rouvier, R. D. Mori, and Y. Esteve, “Open-source conversational ai with SpeechBrain 1.0,” 2024, DOI: 10.48550/arXiv.2407.00463.

[28] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “Lora: Low-rank adaptation of large language models,” *arXiv preprint arXiv:2106.09685*, 2021, DOI: 10.48550/arXiv.2106.09685.

[29] Q. Zhang, M. Chen, A. Bukharin, N. Karampatziakis, P. He, Y. Cheng, W. Chen, and T. Zhao, “Adalora: Adaptive budget allocation for parameter-efficient fine-tuning,” *arXiv preprint arXiv:2303.10512*, 2023, DOI: 10.48550/arXiv.2303.10512.

Yan Meng is a Ph.D. student at Department of Telecommunications and Artificial Intelligence, Budapest University of Technology and Economics, Hungary. She received her management bachelor’s degree in information management and information systems at Beijing Technology and Business University (2020) and master of science in computer science engineering at Budapest University of Technology and Economics (2022). Her current research interests include automatic speech recognition based on low-resource datasets, spontaneous speech recognition, Deep Learning, signal processing and acoustic models.

Péter Mihajlik PhD, is a senior research fellow at the Dept. of Telecommunications and Artificial Intelligence (TMIT), Faculty of Electrical Engineering and Informatics (VIK), Budapest University of Technology and Economics (BME). He also affiliated with the Hungarian Research Centre for Linguistics, Hungarian Research Network (HUN-REN) as part-time senior researcher. He got the MSc in Electrical Engineering in 1999 at BME and obtained the PhD in Automatic Speech Recognition in 2011. He is the head of Speech Recognition Group in SmartLabs/BME-TMIT.