PARAMETER AVERAGING IS ALL YOU NEED TO PREVENT FORGETTING

Peter Plantinga[2*], *Jaekwon Yoo*[1], *Abenezer Girma*[1], *Chandra Dhir*[1]

[1]Machine Learning Center of Excellence, JPMorganChase
[2]McGill University

peter.plantinga@mcgill.ca, jaekwon.yoo@jpmchase.com, abenezer.girma@jpmchase.com, chandra.dhir@jpmchase.com

ABSTRACT

Continual learning for end-to-end automatic speech recognition has to contend with a number of difficulties. Fine-tuning strategies tend to lose performance on data that has been previously trained on, a phenomenon known as catastrophic forgetting. Adapters can help by allowing easy switching between fine-tuned models, but adapted models lose performance on data from other domains, which is a problem if you don't know what domain your input data comes from. We propose a solution that reduces forgetting to only 3.4% while exceeding the average performance of solutions fine-tuned on all available data, which even with LoRA has a forgetting rate of 49%. Our experiments on diverse datasets and models show that a linear interpolation of several models' parameters, each fine-tuned from the same generalist model, results in a unified model that performs well on all tested data. In addition, the same model can be fine-tuned and averaged multiple times while maintaining low rates of forgetting.

***Index Terms*—** speech recognition, continual learning, catastrophic forgetting, parameter averaging, adapters

I. INTRODUCTION

Modern end-to-end automatic speech recognition (E2E-ASR) systems have achieved impressive results across a variety of data by training on massive datasets up to millions of hours [1]. While these generalist models often perform surprisingly well on domains they have never seen in a zero-shot manner, they can still benefit from fine-tuning on data in a target domain for specific applications.

A typical strategy for fine-tuning E2E-ASR systems involves standard gradient descent updates to model parameters using data from the target domain [2]. However, this strategy usually suffers reduced performance on data from the original domain, a phenomenon known as catastrophic forgetting [3]. While it may be possible for some applications to maintain different parameters for different domains, this approach has the downside of adding complexity and consuming storage space, particularly for large models. In addition, it may not be clear for all cases which domain a target sample falls into, nor is it obvious which parameters to use in new domains.

Several attempts have been made to address this challenge by changing the rate that some parameters are updated. This has been achieved either by freezing some parameters [4] or by introducing loss regularization designed to mitigate forgetting [5]. Other similar approaches include layer-wise learning rate decay (LLRD) [6] and slanted triangular learning rates (STLR) [7]. These techniques have had mixed success in reducing forgetting; serial fine-tuning across multiple new domains often still results in substantially reduced performance on data from domains seen during training. Our proposed method can be combined with these approaches, and demonstrates a significantly reduced rate of forgetting on top of them.

Some have addressed forgetting by freezing the entirety of the pre-trained generalist model and introducing domain-specific parameters [8]. A popular technique in this vein is adapters [9], which involves freezing the original model parameters and updating only small modules inserted between layers, with a starting configuration that preserves the behavior of the original model. One primary example is LoRA [10]. In principle, when performing inference on a sample, the appropriate adapters for the given sample's domain can be loaded and utilized. However, determining the domain of a sample may not always be straightforward, or the sample may originate from a new domain. In addition to making it easier to train, store, and switch between models, adapters may also mitigate forgetting on original domain data. But there is still room for improvement. Our approach reduces forgetting drastically while maintaining a single model, thereby removing complexity and eliminating the need to determine the domain from which a sample originates.

One final technique involves replaying data from the original domain [11]. This approach can work well when the original data is available. However, it is not always feasible, especially for pretrained models where the original data is not publicly available. This is the case for most of the best-performing publicly-available end-to-end ASR models.

In summary, this work proposes a reformulation of the continual learning paradigm. Instead of training a model

*Work performed at JPMorganChase

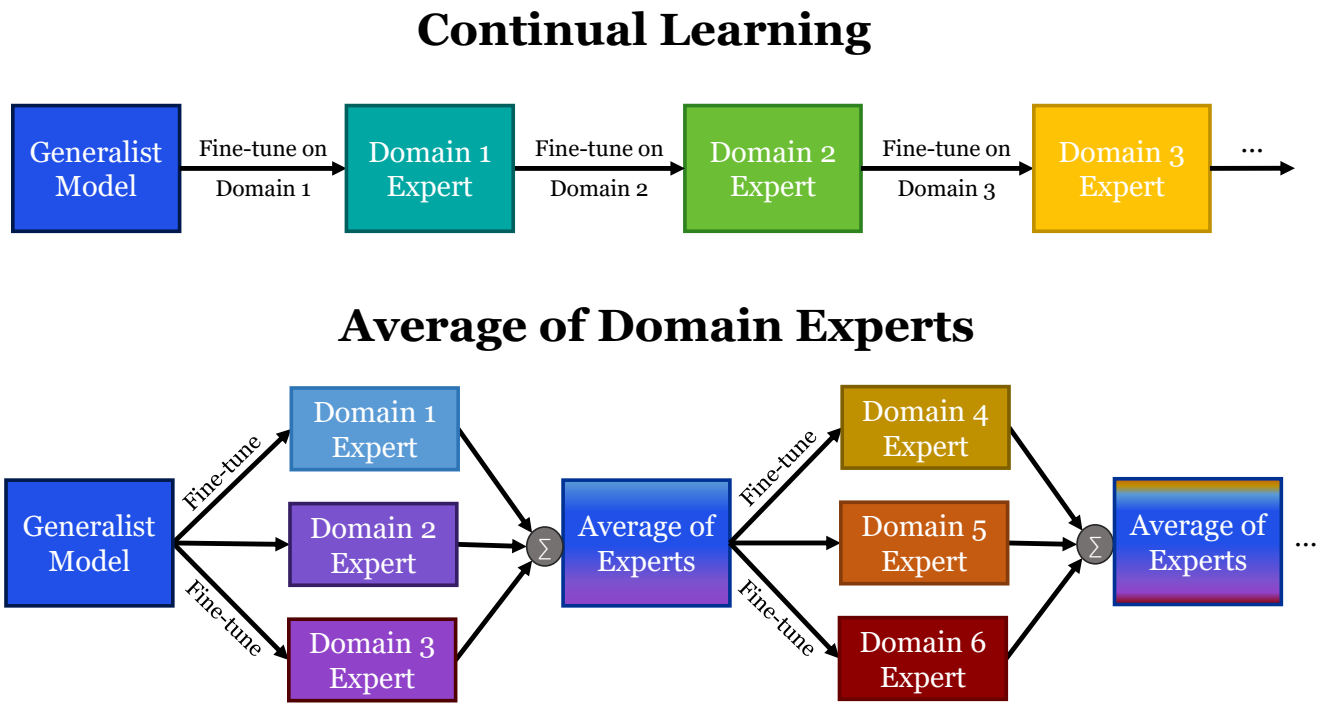

Fig. 1. Our proposed update to the continual learning paradigm: instead of training sequentially on a variety of domains, fine-tune on each domain in parallel and then combine the results to get an average of domain experts model with minimal forgetting. This process can be done multiple times in sequence to improve performance without forgetting.

serially on different datasets, which can lead to forgetting, we minimize this issue by training in parallel on several datasets and combining the resulting expert models through parameter averaging. This process can be repeated, further fine-tuning the averaged model on the same or new data, with little to no increase in the degree of forgetting. We call this proposed paradigm Average of Domain Experts (AoDE).

II. RELATED WORK

Model parameter averaging appears in a number of contexts but appears only rarely in the context of continual learning, given the emphasis of the field on serial fine-tuning on a sequence of new domains. We relate a few of these contexts:

For distributed model training with limited connectivity, called federated learning, some researchers have found that averaging models can achieve a similar performance as training a single model on all data [12], [13]. This strategy has the advantages of reducing communication overhead, as well as preserving data privacy by sharing only model parameters and not data samples. Dynamic approaches share parameters more or less frequently based on how rapidly performance deteriorates on out-of-domain data.

Another context in which model averaging appears is in improving the generalization of trained models. One example is Stochastic Weight Averaging [14] which finds better optima by collecting checkpoints throughout the training process and averaging them.

Other researchers have used model averaging to improve semi-supervised learning using a teacher model [15]. The authors found that better teacher models could be created by averaging the parameters of several teacher models, created by adding noise to the internal representations of student models. These averaged teacher models produce better training targets during semi-supervised training.

One last example is that model averaging is used to understand the dynamics of loss basins for neural networks [16]. In order to understand how it is that different random initializations of ResNets end up achieving very similar performance after training, the authors suggest permuting the parameters in an isomorphic way so as to merge the distinct loss basins into a single loss basin. This is done by rearranging the parameters in one model to best match the parameters in an original model, and then merging the models.

All of this related work shows that model parameter averaging is a powerful technique that is under-used for the purpose of continual learning.

III. METHODS

Nearly all SGD-based fine-tuning procedures that sequentially access data will inevitably lose some performance on the original domain. In addition, the choice of which

Training Procedure	CORAAL	% Forgetting
Pretrained Model	18.8	-
FT CORAAL, LLRD=1.0	14.4	62%
FT CORAAL, LLRD=0.9	12.4	18%
FT CORAAL, LLRD=0.8	13.2	6.7%

Table I. WER comparison for Whisper small.en fine-tuning with several values of LLRD. Forgetting is defined as percent increase in error rates on LibriSpeech.

Procedure	SPGI	CORAAL	% Forgetting
Pretrained Whisper	4.94	18.8	-
Frozen Dec., lr=3e-5	4.17	19.0	10.5%
LLRD=0.9, lr=1e-5	3.14	18.7	9.3%
LLRD=0.9, lr=3e-5	2.87	22.6	52.3%

Table II. WER performance comparison between freezing all layers except encoder multi-head attention layers as proposed by [4] and the LLRD approach, using Whisper small.en model.

Procedure	VCTK	CORAAL	Avg.	% Forgetting
FT - VCTK	1.62	33.2	12.5	120%
LoRA - VCTK	1.78	31.3	11.2	85%
FT - COR.	3.64	15.1	7.09	38%
LoRA - COR.	2.65	15.2	6.52	22%
Combined	3.62	17.5	7.05	120%
Comb. + LoRA	3.38	18.5	7.47	49%
AoDE	2.57	15.3	5.51	8.9%
AoDE + LoRA	2.07	19.7	6.17	8.6%

Table III. WER comparison between different fine-tuning procedures and their LoRA equivalents.

Aggregation	VCTK	CORAAL	Avg.	% Forgetting
Average	2.57	15.3	5.51	9%
Most extreme	1.99	18.1	6.60	23%
Least extreme	11.4	26.7	12.4	219%

Table IV. A comparison of WER scores for different model parameter aggregation methods.

order to fine-tune on domains can have a significant effect on the outcome. To address all of these limitations and produce a single good-performing generalist model with no loss of performance, we propose a new continual learning paradigm that fine-tunes on each domain in parallel, and then averages the resulting expert models. See Figure 1 for a graphical depiction of the proposed Average of Domain Experts approach.

III-A. LLRD and STLR

We begin by reproducing state-of-the-art continual learning techniques such as layer-wise learning rate decay (LLRD) [6] and slanted triangular learning rates (STLR) [7]. These techniques help with better learning and less forgetting, as shown in Table III-A. LLRD is applied by assigning the highest learning rate to the highest encoder layer and decaying the learning rate of each lower layer by a constant factor, usually 0.9. The learning rate of the lowest encoder layer is applied to any layers not in the encoder (decoder, output, embedding, etc.), from the inspiration of [4]. Our learning rate schedule, STLR, peaks at roughly 10-20% of the total training time.

Our experiments show that freezing non-encoder layers, as suggested by [4] is roughly equivalent to reducing the overall learning rate in the proposed scheme, as shown in Table II. If one compares the frozen layers row with the following row, they achieve very similar results across the board. We also found no benefit to LLRD by adding a loss against the predictions of the original model, a technique called Learning without Forgetting (LwF) [5].

III-B. LoRA

LoRA (Low-Rank Adaption) [10] is an approach to fine-tune models performance by employing a low-rank matrix decomposition technique to adapt the pre-trained model's weight matrices. In contrast to traditional fine-tuning methods, LoRA minimizes the number of trainable parameters by keeping the pre-trained model weights frozen and introducing trainable rank decomposition matrices to the layers. This approach fine-tunes a smaller set of parameters while capturing domain-specific information. The two decomposition matrices of rank (r) will be multiplied together to achieve the dimension of the original weights matrices (d_{model}) in the pre-trained model, $r \leq d_{model}$. A low-rank indicates a more compact representation of the matrix, leading a fewer wights to be learned during fine-tuning. Hence the choice of the rank in LoRA affects the balance between computational cost and fine-tuning performance. To enhance the fine-tuning effectiveness of LoRA, we not only applied it to the transformer layers but also applied it into the depth-wise separable convolution layer of the Parakeet model. We conducted experiments using various LoRA rank values to balance between the model's performance and the phenomenon of forgetting during sequential fine-tuning.

Our experiments using LoRA seen in Table III, we see a consistent pattern of slightly worse performance on the training dataset, but better performance on non-training datasets and less forgetting. All experiments were conducted with either rank=16 or rank=32, or around 2.5-5% of the parameters of the model, which were experimentally found to give the lowest WERs. Scale parameter was set to $2 \times r$. For AoDE, we first merge the LoRA weights back into the model before averaging, avoiding any dimension mismatches.

III-C. AoDE

Once the fine-tuning process is done producing domain expert models, we compute the average of experts in a straightforward way: a linear interpolation of corresponding model parameters with equal weighting on every model.

Dataset	Hours	Domain	Condition
SPGISpeech	5000	Finance	Read Speech
CORAAL	140	Everyday	Conversational
DiPCo	6	Everyday	Dinner Party
Noisy VCTK	20	News	Added Noise
Google NE	6	News	Accented
JL Corpus	1	Script	Emotional

Table V. A comparison of datasets used for our experiments.

This is sufficient for good results, and works well with other techniques for reduced forgetting, such as LLRD. We experimented with other aggregation techniques and found similar average WER from taking the most extreme values for each parameter, but greatly increased WER from the least extreme values. This provides additional experimental evidence for "dead" neurons with values close to zero [17]. We measure forgetting as the reduction in performance on LibriSpeech test-clean and test-other sets combined, and parameter averaging performs best in terms of forgetting.

This process can be taken further, by taking the averaged model and fine-tuning it once more on the datasets to take a new average. Our experiments show that this process can be repeated multiple times with little additional forgetting, and improves the average performance across datasets. It is worth noting that this is only possible in the scenario where the data do not have a short retention period, unlike the first average in which the experts can be trained asynchronously. This makes the iterative retraining and averaging method more comparable to fine-tuning using a combined dataset of all available data.

IV. EXPERIMENTS

We demonstrate the generality of our results by using unrelated pre-trained models with differing architecture, training loss, data, sponsoring organization, etc. The two pretrained models we used in our experiments are the NeMo Parakeet CTC model [18] and the OpenAI Whisper small.en model [19].

IV-A. Datasets

For our experiments we used multiple public datasets with a variety of qualities. A comparison can be found in table V. The number of hours ranges from just a couple to over 5000 hours, and the datasets are from a variety of domains and have various difficult conditions. Here is a list of the datasets and some information on them.

SPGISpeech [20] consists of 5000 hours of high-quality recordings of earnings calls. These recordings are well-transcribed and are difficult for a generalist model only on account of a large vocabulary of financial terms that are unlikely to appear elsewhere. To speed up evaluation, we use a random subset of 2000 samples (roughly same size as LibriSpeech test sets) for a test set, and find the resulting performance differs by less than 3% relative in all measured cases. In addition, while training the Whisper Small.en model we found that our techniques were still not sufficient to prevent some catastrophic forgetting when the entire training set was used. Instead, we select a random subset of about 1000 hours of the data for training.

Procedure	SPGI	CORAAL	DiPCo	Avg.	% Forgetting
Pretrained	4.94	18.8	48.5	16.6	-
FT SPGI	2.87	22.6	50.1	18.4	52.3%
FT COR.	4.58	12.4	44.3	14.8	17.7%
FT DiPCo	4.31	18.2	44.0	15.5	-0.1%
FT S,D,C	4.35	13.1	43.3	14.5	6.2%
AoDE	3.41	15.7	43.0	14.6	2.1%

Table VI. Whisper small.en performance under different fine-tuning procedures. The first three are directly fine-tuned on a single dataset, and the fourth result is fine-tuned on the three datasets in sequence. The final result is the proposed average of domain experts.

CORAAL dataset [21], a conversational dataset between folks whose primary sociolect is African American Vernacular English (AAVE). The data was recorded in six separate locations and over the course of ten years (with one exception). In total, there are more than 150 interviews, surpassing 140 hours of audio. We split the data by separating 5 speakers for each of the validation and test sets, amounting to roughly 5 hours each. Generalist models have difficulty with the conversational nature of the data and the different grammars of the sociolect. We divided the audio into segments based on the provided timings in the transcript, ensuring that the total length does not exceed 30 seconds, in order to match the expected input length for Whisper models.

DiPCo dataset [22] is a small dataset of conversation in a dinner party scenario. These data were the most challenging, involving the most speakers and difficult acoustic conditions. The length of the audio available is 2.7 hours for development and 3.4 hours for test. In a process similar to the one used on CORAAL data, we divided the audio into segments not exceeding 30s in length. The conversations were recorded from a number of devices, for the sake of simplicity we take the sum of close-talking microphones as the audio signal.

Noisy VCTK dataset [23], also sometimes called the Voicebank + DEMAND dataset, is a standard dataset for training and evaluating models for speech enhancement, but is also sometimes used for evaluating robust ASR performance [24]. The DEMAND dataset has various real recorded noisy conditions, such as metro, car, cafe, street, and so on. These are artificially added to the Voicebank dataset which has recorded speakers with a variety of different UK accents. The dataset totals 19 hours of training data, 1h for test.

Google NE [25] is a subset of a crowd-sourced dataset targeting low-resource languages and dialects. In particular, we take the subset of recordings from the Nigerian dialect

of English, totaling 5.8 hours of audio. We select roughly 10% of the speakers for both test and validation, resulting in 4.2 hours for training, 1.1 hours for validation, and 0.5 for test.

JL Corpus [26] is read by voice actors from New Zealand playing different emotions for improving human-computer interactions. This emotional speech provides a different dimension of diversity for the sets of data used in our experiments. However the targets can be memorized because there is only a small pool of different phrases used and said in different emotional ways.

IV-B. Whisper & Cross-Entropy

The Whisper small.en model [19] is trained using standard sequence-to-sequence cross-entropy loss and consists of two major sub-models, an encoder and a decoder. The encoder consists of a small downsampling layer followed by 11 self-attention blocks and the decoder consists of 11 multi-headed attention blocks and an output layer. The total number of parameters is 241M.

This model uses an English-only tokenizer with the same 50k-token vocabulary as GPT-2. The set of characters present in these tokens is much larger than for NeMo Conformer, including upper and lower case as well as punctuation. The training data consists of roughly 500,000 hours of English-only data present in the OpenAI speech data.

IV-C. Parakeet & CTC

The Parakeet CTC model has 1.1 billion parameters and is a top-performing model on Huggingface Open ASR Leaderboard [27]. The architecture is similar to NeMo Conformer but uses Fast Conformer [28] blocks instead, speeding up training and inference. The tokenizer vocabulary includes 1024 subword tokens; all tokens include only lowercase letters, apostrophes, and spaces. The model is trained on NeMo ASRset which currently consists of roughly 36,000 hours of audio from a variety of sources.

V. RESULTS

The results for Whisper small.en are in Table VI. Although we do not know whether Whisper was trained on Librispeech, we still see significantly reduced performance from finetuning on SPGI, indicating that forgetting is occurring. The table shows that the proposed approach (AoDE) demonstrates an improvement over the generalist baseline across all datasets with very little forgetting. Although the degree of forgetting is correlated with the size of the datasets in this table, we show later that the more consistent correlation is with performance. Training on "easier" datasets seems to be more correlated with forgetting. The sequentially fine-tuned model performs the best on the last seen dataset but shows little improvement on the first seen dataset over the pre-trained version.

The results for Parakeet CTC are in Table VII. The proposed approach results in drastically less forgetting compared to other potential approaches, even when techniques such as LLRD and SLTR are used to minimize the forgetting.

- **Scenario 1**: We first compare results for models fine-tuned on a single dataset. Unlike in our Whisper experiment, we do not see that larger datasets are more prone to forgetting, but we do see a consistent pattern that "easier" datasets (as measured by lower error rates) are more prone to forgetting. For example, JL Corpus has repeated phrases in different emotions, which may allow for memorization – leading to very low error rates and a high rate of forgetting.

- **Scenario 2**: We compare methods that are possible where the data have a short retention period, and are not available synchronously. In this scenario, since only one dataset is available at a time, we are forced to do sequential fine-tuning or to train a model on one subset at a time. The averaged model has much better overall performance, though the sequentially-fine-tuned model sometimes performs better on data from a domain it has seen recently.

- **Scenario 3**: In this scenario the data are available long term and can be combined or reviewed. For the first rows, we combine all the data into one fine-tuning set, including a reweighing process aimed at mitigating the data imbalance. For the rest of the rows, we show that we are able to iteratively train and then average on the datasets, leading to better overall performance while maintaining reduced forgetting. During iterative training and averaging we do not include the pre-trained model in the averages. In the last row of this section, we include the original pre-trained Parakeet model in the average, and it reduces the forgetting further to only 5.3% while maintaining the performance of the averaged model without the original.

- **Scenario 4**: Finally, we include results where an oracle provides the domain of the incoming data, and the corresponding trained LoRA system is loaded and used for inference. This has no forgetting – we can use the original model for data from the original domain – but may not always be possible in practice, as new domains are added it may not be clear which LoRA system to use. Additionally, in industrial deployments, managing and refining multiple LoRAs can make the maintenance process exceedingly challenging and time-consuming.

In summary for Table VII, we note that the proposed system achieves the best average performance across the domains, while maintaining a low rate of forgetting of close to 5%.

In Table VIII, we show that it is possible to re-weight the parameters of the component models to favor one domain

Procedure	VCTK	CORAAL	JL Corpus	Google NE	LS t-clean	LS t-other	Average	% Forgetting
Pretrained Parakeet	2.26	20.7	4.85	6.66	1.96	3.65	6.68	0%
Scenario 1								
Fine-tuned - VCTK	**1.62**	33.2	16.8	10.9	4.50	7.78	12.5	120%
Fine-tuned - CORAAL	3.67	**15.1**	8.03	8.38	**2.55**	**5.21**	**6.79**	**38%**
Fine-tuned - JL Corpus	11.4	43.7	**0.28**	15.9	7.26	12.9	15.3	260%
Fine-tuned - Google NE	5.58	25.0	9.89	**5.30**	3.24	6.82	9.30	79%
Scenario 2								
Sequential FT - V,C,J,G	5.64	18.4	**1.11**	6.24	4.90	8.64	7.49	140%
Sequential FT - G,J,C,V	**1.49**	26.7	5.93	9.09	4.76	8.57	9.41	140%
Proposed AoDE	2.57	**15.3**	3.12	5.93	1.98	4.13	**5.51**	8.9%
AoDE with orig.	2.00	18.4	3.03	**5.78**	**1.93**	**3.87**	5.83	**3.4%**
Scenario 3								
Combined Datasets	3.60	17.5	0.49	8.49	4.07	8.19	7.09	120%
Resampled Datasets	2.54	16.7	**0.12**	6.24	3.71	7.16	6.07	94%
AoDE - 2nd iteration	1.87	**15.7**	1.42	5.67	2.02	4.17	5.14	10%
AoDE - 3rd iteration	1.68	15.8	1.24	**5.42**	2.03	4.24	5.07	12%
AoDE - 3rd iter w/ orig.	**1.60**	15.8	1.30	5.59	**1.93**	**3.98**	**5.03**	**5.3%**
Scenario 4								
Oracle Domain LoRA	1.78	15.2	2.47	6.04	1.96	3.65	5.19	0%

Table VII. WER of proposed method on all available datasets against all baselines. After the pretrained baseline, the sections are as follows: **Scenario 1**: Fine-tuned on a single dataset, **Scenario 2**: Datasets have short retention periods, **Scenario 3**: Datasets are available long-term, **Scenario 4**: Oracle provides the sample domain.

Procedure	VCTK	CORAAL	JL Corpus	Google NE	LS t-clean	LS t-other	Average	% Forgetting
AoDE - 3rd it. w/ orig.	1.60	15.8	1.30	5.59	1.93	3.98	5.03	5.3%
AoDE - VCTK $\times$ 4	**1.42**	17.5	1.39	6.12	2.14	4.30	5.47	15%
AoDE - CORAAL $\times$ 4	1.78	**14.7**	1.73	5.53	1.96	4.05	**4.96**	7.1%
AoDE - JL Corpus $\times$ 4	2.00	16.4	**1.21**	5.84	2.02	4.25	5.28	12%
AoDE - Google NE $\times$ 4	1.70	16.2	1.39	**5.25**	2.01	4.11	5.10	9.1%
AoDE - original $\times$ 4	1.75	17.1	2.35	5.81	**1.86**	**3.77**	5.44	**0.4%**

Table VIII. WER comparison between AoDE models with different weightings of component models.

over other domains. We weight each different component model as half of the total weight (4/8) and weight the rest equally (1/8). The results show that it is possible to improve performance on a given domain by up to 10% at the cost of increased forgetting. When the original model is weighted as half, we see our lowest level of forgetting at only 0.4%. This model actually shows *5% better* performance on LibriSpeech test-clean without having seen any additional data from this domain.

VI. CONCLUSIONS

We were able to show that updating the continual learning paradigm with parameter averaging can dramatically reduce catastrophic forgetting in well-trained generalist end-to-end speech recognition models. This is achieved by averaging the parameters of well-trained expert models, proving a flexible and tunable approach to model development.

In the future, we hope to improve on this work with more sophisticated averaging techniques, taking into account permutation invariances, with techniques such as Git Re-Basin [16] or Federated Matched Averaging (FedMA) [13]. In addition, large language model merging algorithms such as [29], [30] could help preserve more performance from each domain while averaging.

VII. REFERENCES

[1] Yu Zhang, Wei Han, James Qin, Yongqiang Wang, Ankur Bapna, Zhehuai Chen, Nanxin Chen, Bo Li, Vera Axelrod, Gary Wang, et al., "Google USM: Scaling automatic speech recognition beyond 100 languages," *arXiv preprint arXiv:2303.01037*, 2023.

[2] Seyedmahdad Mirsamadi and John HL Hansen, "On multi-domain training and adaptation of end-to-end RNN acoustic models for distant speech recognition.," in *INTERSPEECH*, 2017, pp. 404–408.

[3] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter, "Continual lifelong learning with neural networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[4] Yuki Takashima, Shota Horiguchi, Shinji Watanabe, Paola García, and Yohei Kawaguchi, "Updating only

encoders prevents catastrophic forgetting of end-to-end ASR models," in *INTERSPEECH*, 2022.

[5] Zhizhong Li and Derek Hoiem, "Learning without forgetting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 2935–2947, 2016.

[6] Tianyi Zhang, Felix Wu, Arzoo Katiyar, Kilian Q. Weinberger, and Yoav Artzi, "Revisiting few-sample bert fine-tuning," *ArXiv*, vol. abs/2006.05987, 2020.

[7] Jeremy Howard and Sebastian Ruder, "Universal language model fine-tuning for text classification," in *Annual Meeting of the Association for Computational Linguistics*, 2018.

[8] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell, "Progressive neural networks," *ArXiv*, vol. abs/1606.04671, 2016.

[9] Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou, "K-Adapter: Infusing knowledge into pre-trained models with adapters," in *Findings*, 2020.

[10] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen, "Lora: Low-rank adaptation of large language models," *arXiv preprint arXiv:2106.09685*, 2021.

[11] David Isele and Akansel Cosgun, "Selective experience replay for lifelong learning," in *AAAI Conference on Artificial Intelligence*, 2018.

[12] Michael Kamp, Linara Adilova, Joachim Sicking, Fabian Hüger, Peter Schlicht, Tim Wirtz, and Stefan Wrobel, "Efficient decentralized deep learning by dynamic model averaging," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18*. Springer, 2019, pp. 393–409.

[13] Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni, "Federated learning with matched averaging," in *International Conference on Learning Representations (ICLR)*, 2020.

[14] Pavel Izmailov, Dmitrii Podoprikhin, T. Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson, "Averaging weights leads to wider optima and better generalization," *ArXiv*, vol. abs/1803.05407, 2018.

[15] Antti Tarvainen and Harri Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," *Advances in neural information processing systems*, vol. 30, 2017.

[16] Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa, "Git Re-Basin: Merging models modulo permutation symmetries," *arXiv e-prints*, 2022.

[17] Scott C. Douglas and Jiutian Yu, "Why relu units sometimes die: Analysis of single-unit error backpropagation in neural networks," *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, pp. 864–868, 2018.

[18] Oleksii Kuchaiev, Jason Li, Huyen Nguyen, Oleksii Hrinchuk, Ryan Leary, Boris Ginsburg, Samuel Kriman, Stanislav Beliaev, Vitaly Lavrukhin, Jack Cook, et al., "NeMo: a toolkit for building AI applications using neural modules," *arXiv preprint arXiv:1909.09577*, 2019.

[19] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever, "Robust speech recognition via large-scale weak supervision," *arXiv preprint arXiv:2212.04356*, 2022.

[20] Patrick K O'Neill, Vitaly Lavrukhin, Somshubra Majumdar, Vahid Noroozi, Yuekai Zhang, Oleksii Kuchaiev, Jagadeesh Balam, Yuliya Dovzhenko, Keenan Freyberg, Michael D Shulman, et al., "SPGISpeech: 5,000 hours of transcribed financial audio for fully formatted end-to-end speech recognition," *arXiv preprint arXiv:2104.02014*, 2021.

[21] Tyler Kendall and Charlie Farrington, "The corpus of regional african american language," *The Online Resources for African American Language Project*, 2021.

[22] Maarten Van Segbroeck, Zaid Ahmed, Ksenia Kutsenko, Cirenia Huerta, Tinh Nguyen, Bjorn Hoffmeister, Jan Trmal, Maurizio Omologo, and Roland Maas, "DiPCo - dinner party corpus," in *INTERSPEECH*, 2020.

[23] Cassia Valentini-Botinhao, "Noisy speech database for training speech enhancement algorithms and TTS models," *Edinburgh DataShare*, 2017.

[24] Peter Plantinga, Deblin Bagchi, and Eric Fosler-Lussier, "Perceptual loss with recognition model for single-channel enhancement and robust asr," *arXiv preprint arXiv:2112.06068*, 2021.

[25] Alena Butryna, Shan-Hui Cathy Chu, Isin Demirsahin, Alexander Gutkin, Linne Ha, Fei He, Martin Jansche, Cibu Johny, Anna Katanova, Oddur Kjartansson, et al., "Google crowdsourced speech corpora and related open-source resources for low-resource languages and dialects: an overview," *arXiv preprint arXiv:2010.06778*, 2020.

[26] Jesin James, Li Tian, and Catherine Watson, "An open source emotional speech corpus for human robot interaction applications," *Interspeech 2018*, 2018.

[27] Vaibhav Srivastav, Somshubra Majumdar, Nithin Koluguri, Adel Moumen, Sanchit Gandhi, et al., "Open automatic speech recognition leaderboard," https://huggingface.co/spaces/hf-audio/open_asr_leaderboard, 2023.

[28] Dima Rekesh, Nithin Rao Koluguri, Samuel Kriman, Somshubra Majumdar, Vahid Noroozi, He Huang,

Oleksii Hrinchuk, Krishna Puvvada, Ankur Kumar, Jagadeesh Balam, and Boris Ginsburg, "Fast conformer with linearly scalable attention for efficient speech recognition," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–8.

[29] Prateek Yadav, Derek Tam, Leshem Choshen, Colin A Raffel, and Mohit Bansal, "Ties-merging: Resolving interference when merging models," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[30] Le Yu, Bowen Yu, Haiyang Yu, Fei Huang, and Yongbin Li, "Language models are super mario: Absorbing abilities from homologous models as a free lunch," in *Forty-first International Conference on Machine Learning*, 2024.