Mitigating Catastrophic Forgetting for Few-Shot Spoken Word Classification Through Meta-Learning

Ruan van der Merwe[1] and Herman Kamper[2]

[1]ByteFuse
[2]E&E Engineering, Stellenbosch University, South Africa
`ruan@bytefuse.ai, kamperh@sun.ac.za`

Abstract

We consider the problem of few-shot spoken word classification in a setting where a model is incrementally introduced to new word classes. This would occur in a user-defined keyword system where new words can be added as the system is used. In such a continual learning scenario, a model might start to misclassify earlier words as newer classes are added, i.e. catastrophic forgetting. To address this, we propose an extension to model-agnostic meta-learning (MAML). In our new approach, each inner learning loop—where a model "learns how to learn" new classes—ends with a single gradient update using stored templates from all the classes that the model has already seen (one template per class). We compare this method to OML (another extension of MAML) in few-shot isolated-word classification experiments on Google Commands and FACC. Our method consistently outperforms OML in experiments where the number of shots and the final number of classes are varied.

Index Terms: continual learning, few-shot learning, spoken word classification, meta-learning.

1. Introduction

Imagine a speech system that a user can teach new commands by providing it with just a few examples per word class. To start out with, the user might provide the system with examples of the words "sing", "open" and "close", and with just a handful of support examples, the system should be able to correctly classify new test inputs. (This should work irrespective of the language of the user.) In contrast to conventional speech recognition systems that are trained on thousands of hours of examples, such a system would be *few-shot*. Inspired by the observation that humans can learn new words from very few examples, a number of studies in machine learning have started to look at this problem of few-shot word classification [1, 2, 3].

But now imagine that, as the user is using the system, they want to add more words to the system, e.g. "turn" and "give". As more and more words are added, the system might start to misclassify words that it learned earlier—the problem of catastrophic forgetting [4, 5]. The combination of dynamic environments, limited support examples used for training, and continual learning make this task a major challenge. While other studies have look at the few-shot problem [1, 6], the proposed methods do not deal with the continual learning problem. In this paper we propose a new approach for few-shot continual learning and evaluate it specifically for isolated word classification.

Outside of speech processing, there has been several studies on continual learning, e.g. [7]. Many of these studies try to explicitly address the problem of catastrophic forgetting [8, 9]. Within speech research, there has been some limited attempts to address the continual learning problem, specifically in au-

tomatic speech recognition (ASR) [10] and keyword spotting applications [11]. However, these studies do not consider the few-shot learning setting, but rather on adding new vocabulary words to supervised models trained on substantial amounts of labelled data. Within the signal processing community, there has been some studies looking at both few-shot learning and continual updating [12], but this was for general audio and not spoken word classification.

In this paper we specifically look at addressing few-shot continual learning by utilising meta-learning techniques, where algorithms learn automatically how to solve the continual learning task [13, 14, 15]. We specifically extend model-agnostic meta-learning (MAML) [16], which is a meta-learning technique that optimises an initial set of model weights such that they can be quickly updated to a new task. MAML has been used before within speech research for speaker adaptive training [17], and data-efficient ASR [18, 19], but not for few-shot continual word learning.

We propose a new approach: MAML for **con**tinual learning (MAMLCon). This extension over MAML is very simple, but it leads to consistent improvements in few-shot word classification. MAMLCon specifically extends MAML by explicitly doing meta-learning of an increasing number of new classes in the inner loop of the algorithm. At the end of the inner loop, MAMLCon also performs a single update using templates stored for all the classes seen up to that point. Since MAMLCon has learned how to learn continually, it is able to do so efficiently at test time on classes that are completely unseen during meta-learning.

We compare MAMLCon to another continual learning extension of MAML called OML [13]. We perform experiments where we vary the number of shots, the number of steps where classes are added, and the final number of word classes. In all cases the simple MAMLCon extension outperforms OML in isolated word few-shot classification.

2. MAMLCon

2.1. Background on MAML

Model-agnostic meta-learning (MAML) [16] is an algorithm that aims to learn an initial set of weights that can be rapidly adapted to new tasks using just a few examples from the target task. Consider the example of one-shot speech classification. We want a model that can learn to classify new words based on a single training example per word class. E.g. we give the model a *support set*[1] with a single example for "sing", "open", "close" and then want the model to accurately classify test inputs from

---

[1]In few-shot classification, the *support set* is the small set of training examples that we get for the target task.
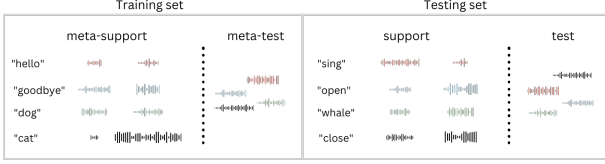
Figure 1: *During training, MAML samples meta-support and -test sets from labelled data. At test time, it is then presented with a support set containing classes never seen during training, and asked to classify test items from these classes.*

Figure 2: *The MAMLCon training process. We construct the continual learning setup directly as a meta-task, where the algorithm is tasked to learn how to perform well in continual learning setup while being allowed to observe one already seen example from previously learned word groups and update its weights with one update step.*

one of these classes. A naive approach would be to start with a randomly initialised model and then simply update its weights through gradient descent directly on this support set. The idea behind MAML is to instead learn good initial weights which can then subsequently be fine-tuned. MAML does this by using a large labelled dataset and then simulating many few-shot classification tasks. Continuing with our example, let's say we have a very large training set of isolated words with their labels (no examples from our few-shot classes). From this training dataset we can sample a meta-support set and a meta-test set, e.g. "hello", "drop", "greetings". In the so-called *inner loop* of the MAML algorithm, we then update the model weights using a few gradient descent steps on the support set. Instead of storing the resulting weights from these inner-loop updates, MAML optimises the initial weights θ on top of which the inner-loop updates are performed. I.e., the *outer loop* of MAML tries to find a good initialisation for doing a few gradient steps on a handful of examples. The result is weights θ^* that are optimised so that they work well when a few gradient steps are applied on top of them using a small set of support examples.

More formally, in the inner loop, the model's current weights at step j, θ_0^j, are optimised for a given task $\mathcal{T}_i$, resulting in updated weights θ_T^j, where T is the total number of inner-loop update steps. In the outer loop, the performance of the fine-tuned model θ_T^j is evaluated on a meta-test set, and the initial weights θ_0^j are then updated through:

$$\theta_0^{j+1} \leftarrow \theta_0^j - \beta \nabla_{\theta_0^j} \sum_{\mathcal{T}_i} \mathcal{L}_{\mathcal{T}_i} \left(X_i^{\text{TEST}}, Y_i^{\text{TEST}}, \theta_T^j \right) \quad (1)$$

Here, X_i and Y_i are data points from task $\mathcal{T}_i$, and β is the outer learning rate, with the inner-loop update steps having an inner learning rate α. Updating θ_0^j in this manner leads to optimised weights θ^* which can be fine-tuned to new tasks in only a few steps. When the inner loop is constrained to only a few examples per class, the algorithm can learn to accomplish the task with a limited number of examples, thus resulting in a few-shot classification model.

To test a model after training it using MAML, we can sample multiple groups of words from our few-shot classes and construct multiple scenarios where you train on a support set and measure on a held-out test set. The optimised model θ^* is copied to each distinct scenario for training. An example of how these meta-training and -testing scenarios are constructed is shown in Figure 1, where we show just one task in both the training and testing stages. For further reading on meta-learning and MAML, please refer to [20].

2.2. MAMLCon: Learning to Continually Learn

Consider the following example for word classification in a continual learning setting. Let's say at test time a model has re-
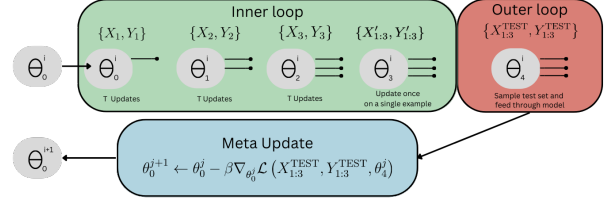
ceived a support set for the words "sing", "open" and "close". We used MAML and updated the model on this support set and it achieves reasonable performance. But now we want the model to additionally be able to classify the words "turn" and "give". We give the model a few more support examples for these new words and update its weights through further fine-tuning. Later on, we want to add even more words by just giving a few examples. The problem is that as we add more and more words, the model would start to fail on words that it learned earlier. This is called catastrophic forgetting.

To address this, we propose a new extension of MAML: **m**odel-**a**gnostic **m**eta-**l**earning for **con**tinual learning (MAML-Con). MAMLCon extends MAML in two ways. First, it formulates the continual learning problem itself as a meta-learning task. Secondly, it utilises a single update step on previously acquired knowledge. The motivation for this step is to optimise the model such that one can use use the smallest possible dataset (one example per class) to maintain performance on previously learned words.

The training process of MAMLCon is shown in Figure 2. As an example, let's say that during training we sample a meta-support set consisting of five examples each for "hello", "drop", "greetings". In MAML we would just fine-tune on all the examples together. Instead, in the inner-loop training phase of MAMLCon, the model is first trained for T steps on the "hello" examples, followed by T steps of training on "drop" and then T steps on "greetings". Once the model has been trained on all examples in the meta-support set, a single batched weight update step is performed using a single stored example for each of the "hello", "drop", "greetings" classes. In the outer loop, the meta-test set, which contains samples for all words in the meta-support set, is used to evaluate the performance of the model, and the original weights are updated to obtain an optimal set of weights θ^*. Because with MAMLCon the model has seen incremental learning during training, these weights are optimised to facilitate few-shot continual learning. This means we can update the model further on "turn" and "give" and the model would still perform well on "hello", "drop" and "greetings".

To state this formally, in the inner loop, the model's weights, θ_0^j, are updated through sequential training on new classes in the meta-support set. The inner-loop optimisation is performed through the calculation of gradients with respect to θ_i^j based on the loss computed on a per-class (or per-group of classes) basis from the meta-support set, leading to the updated weights θ_{i+1}^j. At the end of the inner loop, a single weight update is performed on a previously seen template from each class, enabling the model to leverage its prior knowledge. In Figure 2,

this set of templates is denoted with a dash, $\{X'_{1:3}, Y'_{1:3}\}$. The outer loop computes the loss on the meta-test set and applies the meta-update step to the original weights to obtain θ_0^{j+1}. The update is performed based on the gradient of the test loss with respect to θ_0^j, as in Equation 1.

At test time, MAMLCon is used by just following the inner loop. Every time that classes are added, T update steps are followed with one update step on a set of templates for all classes learned up to that point. This means that in a real-life use case, we will just have to store a single example per class to act as templates in future updates.

Our method is most similar to online aware meta-learning (OML) [13]. The OML classifier consists of a feature extractor with weights θ_{FE} that feeds into a prediction network with weights θ_{PN}. In OML's the inner loop, they sample N classes to train on sequentially but only update θ_{PN}, leading to θ_{PN}^*. After training these N classes, they sample a random batch of data and measure the meta-test loss on this batch. They then back-propogate through this entire process to update θ_{FE} and θ_{PN}. Our method differs from OML in several ways. Firstly, in the inner loop, we update the entire network and not just the prediction network. Secondly, we allow the model to access a single example of a previously seen class during the inner-loop training phase. Finally, in contrast to OML, we do not perform the meta-test on a random sample of classes, but instead on all classes seen up to that point.

3. Experimental Setup

Data. We perform word classification experiments using the Flickr 8k Audio Caption Corpus (FACC) [21] and the Google Commands v2 dataset [22]. For the experiments on FACC, utterances are segmented into isolated words using forced alignments, and words with the same stem are grouped into a single class. Both the FACC and Google Commands datasets are split so that words with the same stem will not appear in the training and test sets. For FACC, this results in approximately 100 unique stems that can be sampled for continual learning, while there are 10 unique stems for Google Commands. We divide these stems randomly into our test and train splits. Between epochs in meta-learning, the same word class will be assigned a different integer label so that the model is not able memorise a particular word in the meta-learned weights.

Models. All words are parameterised as mel-frequency cepstral coefficients (MFCCs) with delta and delta-delta features. Input items are zero-padded to a consistent length. A simple 3-layer 2D convolutional neural network is applied to extract features from the MFCCs, which are then fed into a single fully connected layer that is trained to classify the given words. We use the same architecture for OML. The Adam optimiser [23] is used for both inner and outer loop updates, with a learning rate of 0.001 for the inner loop and 0.0001 for the outer loop.

In all the experiments below we start with a set of initial words, and then incrementally add more word classes. For the initial set of words being learned by the model, we perform $T = 30$ weight updates to ensure saturation of the model to simulate the scenario in the real world of having a well-trained model and subsequently updating it. After this, for each new group of classes added to the model, $T = 5$ update steps are performed. In the quick adaptation step on the templates at the end of the inner loop, a single example per class is sampled from the support set and a single update is performed. We use the first-order MAML algorithm [16], which ignores the

meta-learning process's second-order derivatives; this doesn't affect performance while speeding up computation and reducing memory requirements [16, 24]. We adapt the Learn2Learn software package [25] for training both OML and MAMLCon.[2]

Evaluation. We consider different continual learning scenarios. All start with an initial set of few-shot learned word classes: this number of initial classes are denoted as **CS**. We then incrementally introduce a number of additional word types (**CA**) at every update step. The final number of word types is denoted as N. An experiment can then be summarised using a succinct notation: e.g. $N$50:**CS**5:**CA**5 would represent a a scenario in which the model ends with a total of 50 word classes, with each iteration incorporating five new words after initially training on five words.

4. Experiments

We compare MAMLCon to OML for few-shot word classification in a range of continual learning experiments. We do not evaluate MAML in isolation, as it has been surpassed in performance by OML and other recent advancements [13, 14].

4.1. Frequent vs Infrequent Updates

A good continual learning algorithm should perform well in scenarios where we add many words at every update step (therefore requiring fewer updates to reach the final number of types N) as well as scenarios where a small number of words are added at every update (requiring more frequent updates to reach N). We compare MAMLCon to OML in both these scenarios, referred to, respectively, as infrequent and frequent updates. For infrequent updates we consider these setups: $N$5:**CS**1:**CA**3, $N$10:**CS**2:**CA**5 and $N$50:**CS**5:**CA**20. For frequent updates we consider $N$5:**CS**1:**CA**1, $N$10:**CS**2:**CA**1 and $N$50:**CS**5:**CA**5. All setups here use $K = 5$ shots (we vary this in the section below).

The results are shown in Table 1, where N is used to identify the particular learning scenario. By looking at the infrequent update scenario, we observe that MAMLCon achieves high accuracies in both smaller ($N = \{5, 10\}$) and larger class scenarios ($N = 50$). In contrast, OML struggles particularly when more classes need to be learned: this can be seen when looking at the sharp drop in accuracy between the results for the FACC dataset in the $N = 10$ and $N = 50$ cases, and on the Google Commands dataset when going from $N = 5$

---

[2]Source code: https://github.com/ByteFuse/MAMLCon

Table 1: *Few-shot classification accuracy (%) over all N classes for continual learning settings where a small number of classes are added frequently, or a large number of classes are added infrequently. N is the final number of classes after continual learning.*

N final classes	Google Commands		FACC	
	5	10	10	50
Infrequent updates:				
OML	62.1	49.9	72.8	32.3
MAMLCon	**85.2**	**73.6**	**86.7**	**74.5**
Frequent updates:				
OML	61.8	36.5	75.1	51.8
MAMLCon	**82.7**	**72.9**	**76.8**	**71.7**

Figure 3 bar chart values: 56.90, 55.64, 71.72, 72.91, 72.03, 54.14, 62.96, 42.61, 49.04, 42.00

Y-axis: Final Accuracy (0.0–0.7). X-axis: K-Shot (1, 5, 20, 50, 200)

Legend: FACC: N=50, CA=5, CS=5 / Google Commands: N=10, CA=2, CS=2
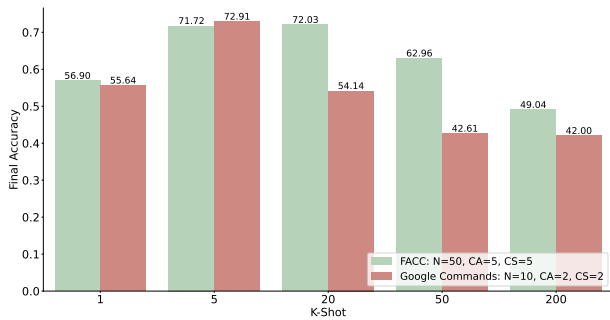
Figure 3: Few-shot classification accuracy (%) of MAMLCon as the number of shots K per class is varied.

to $N = 10$. A similar pattern emerges in the frequent update scenario, where we see that OML shows large drops in accuracy when learning more classes: a particularly large drop is observed on Google Commands when going from $N = 5$ to $N = 10$. Overall, the results demonstrate the superior performance of MAMLCon over OML in both frequent and infrequent update scenarios.

4.2. Few-shot Capabilities

The number of support examples a model can use for learning a new word would depend on the specific practical setting: in some cases we would have only one example per class, while in other cases we could get substantially more. Here we assess the performance of MAMLCon as the number K of support examples (the number of "shots") are varied. We investigate how well MAMLCon operates under these different conditions to gain a better understanding of its capabilities.

Concretely, we present the performance for continual learning setups of $N50$:**CS**5:**CA**5 when evaluating on the FACC dataset and $N10$:**CS**2:**CA**1 when evaluating on the Google Commands dataset over different values of K. These setups were chosen as they represent the most challenging scenarios, requiring multiple weight updates between the initial and final classes.

As seen in Figure 3, when focusing solely on the results for the FACC dataset, as K increases from 1 to 20, the overall performance improves as expected, with only a small increase in performance between $K = 5$ and $K = 20$. However, as K continues to increase, performance decreases. This pattern is also evident in the Google Commands results.

It is encouraging that MAMLCon still performs well with a small number of shots, but it is also somewhat surprising and concerning that as K increases, performance starts to deteriorate. This relationship between accuracy and the number of training examples in Figure 3 can be explained by the trade-off between sample complexity and catastrophic forgetting. We speculate that a moderate value of K, in the range of 20, is sufficient to acquire a robust representation of the task at hand, which is to learn a new word. However, as K increases beyond this point, the weight updates for the new classes may become excessive, resulting in the model forgetting previously learned information.

4.3. Retention of Knowledge

In the preceding sections we looked at performance across all words after a few-shot system has been trained in a continual learning setting. But how does performance differ between

Table 2: *Evaluation of knowledge retention capabilities in continual learning models on the FACC dataset. We measure the accuracy for each label group as it was trained, as well as at the end of training after all words have been learned by the model. We then show the the difference (Δ) between the start (S) and end (E) accuracies. The final accuracy when taking all labels into account is also shown.*

Labels	MAMLCon		OML		No Pre-Training	
	S/E	Δ	S/E	Δ	S/E	Δ
1-5	95/95	0	100/35	-65	90/20	-70
6-10	100/95	-5	85/5	-80	85/50	-35
11-15	90/85	-5	100/70	-30	90/25	-65
16-20	95/70	-25	90/75	-25	100/40	-60
21-25	95/80	-15	75/65	-10	60/10	-50
26-30	95/70	-25	100/95	-5	100/40	-60
31-35	95/50	-45	80/55	-25	85/20	-65
36-40	80/70	-10	75/80	5	90/0	-90
41-45	85/60	-25	90/75	-15	75/60	-15
46-50	-/95	-	-/90	-	-/65	-
Accuracy	77.0		64.5		33.0	

words that are learned earlier relative to words added later in the continual learning cycle? To answer this, we look at the performance of individual words. This allows us to determine how well the model performs on previous classes and how well it retains the knowledge about those words after being trained on new words.

Table 2 shows the results of MAMLCon, OML and a model which was not pre-trainedon the FACC dataset. We use a $N50$:**CS**5:**CA**5 setup, with $K = 20$. This means that there will be ten update steps, with five word classes being added each time. The performance for the words learned in the very first group are given in the row with the 1-5 label, while the words learned in the very last update are given in the 46-50 row. The accuracy after initial training (S) and the final training (E) for each label group is displayed, along with the difference (Δ) between these two accuracy scores.

MAMLCon again outperforms OML in terms of overall accuracy, achieving 77.0% accuracy versus the 64.5% of OML. Looking at individual words, MAMLCon is effective in retaining its knowledge of early label groups (1-30) while struggling more to maintain its accuracy over the later label groups. Conversely, OML performs better in retaining knowledge over later label groups, but shows low accuracy for the early groups.

5. Conclusion

We proposed a novel few-shot continual learning algorithm: model-agnostic meta-learning for continual learning (MAML-Con). It is an extension of MAML that formulates the few-shot continual learning task as a meta-task, allowing the weights to be updated only once by a previously seen word example upon completion of training on new words. We compared MAML-Con to OML, a previous meta-learning algorithm for continual learning. The findings show that MAMLCon outperforms OML in overall accuracy across two datasets and label distribution sizes under both infrequent and frequent update scenarios. Furthermore, our results indicate that MAMLCon effectively maintains knowledge of early label groups while showing more difficulty retaining knowledge of later groups. Nonetheless, it achieves a higher overall accuracy.

6. References

[1] B. Lake, C.-y. Lee, J. Glass, and J. Tenenbaum, "One-shot learning of generative speech concepts," in *Proceedings of the Annual Meeting of the Cognitive Science Society*, vol. 36, 2014.

[2] R. Eloff, H. A. Engelbrecht, and H. Kamper, "Multimodal one-shot learning of speech and images," in *International Conference on Acoustics, Speech and Signal Processing*, 2019.

[3] W.-T. Kao, Y.-K. Wu, C.-P. Chen, Z.-S. Chen, Y.-P. Tsai, and H.-Y. Lee, "On the efficiency of integrating self-supervised learning and meta-learning for user-defined few-shot keyword spotting," in *Spoken Language Technology Workshop*, 2023.

[4] M. McCloskey and N. J. Cohen, "Catastrophic interference in connectionist networks: The sequential learning problem," in *Psychology of learning and motivation*, 1989, vol. 24.

[5] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio, "An empirical investigation of catastrophic forgetting in gradient-based neural networks," *arXiv preprint arXiv:1312.6211*, 2013.

[6] C. Heggan, S. Budgett, T. Hospedales, and M. Yaghoobi, "Metaaudio: A few-shot audio classification benchmark," in *International Conference on Artificial Neural Networks*, vol. 13529, 2022.

[7] Z. Mai, R. Li, J. Jeong, D. Quispe, H. Kim, and S. Sanner, "Online continual learning in image classification: An empirical survey," *Neurocomputing*, vol. 469, 2022.

[8] Z. Li and D. Hoiem, "Learning without forgetting," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, 2017.

[9] D. Lopez-Paz and M. Ranzato, "Gradient episodic memory for continual learning," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[10] S. Sadhu and H. Hermansky, "Continual learning in automatic speech recognition," in *Interspeech*, 2020.

[11] Y. Huang, N. Hou, and N. F. Chen, "Progressive continual learning for spoken keyword spotting," in *International Conference on Acoustics, Speech and Signal Processing*, 2022.

[12] Y. Wang, N. J. Bryan, M. Cartwright, J. P. Bello, and J. Salamon, "Few-shot continual learning for audio classification," in *International Conference on Acoustics, Speech and Signal Processing*, 2021.

[13] K. Javed and M. White, "Meta-learning representations for continual learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[14] G. Gupta, K. Yadav, and L. Paull, "Look-ahead meta learning for continual learning," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[15] S. Beaulieu, L. Frati, T. Miconi, J. Lehman, K. O. Stanley, J. Clune, and N. Cheney, "Learning to continually learn," in *European Conference on Artificial Intelligence*, 2020, vol. 325.

[16] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*, 2017.

[17] O. Klejch, J. Fainberg, P. Bell, and S. Renals, "Speaker adaptive training using model agnostic meta-learning," in *Automatic Speech Recognition and Understanding Workshop*, 2019.

[18] S. Indurthi, H. Han, N. K. Lakumarapu, B. Lee, I. Chung, S. Kim, and C. Kim, "Data efficient direct speech-to-text translation with modality agnostic meta-learning," *arXiv preprint arXiv:1911.04283*, 2019.

[19] G. I. Winata, S. Cahyawijaya, Z. Liu, Z. Lin, A. Madotto, P. Xu, and P. Fung, "Learning fast adaptation on cross-accented speech recognition," in *Interspeech*, 2020.

[20] Y. Tian, X. Zhao, and W. Huang, "Meta-learning approaches for learning-to-learn in deep learning: A survey," *Neurocomputing*, vol. 494, 2022.

[21] D. Harwath and J. Glass, "Deep multimodal semantic embeddings for speech and images," in *Workshop on Automatic Speech Recognition and Understanding*, 2015.

[22] P. Warden, "Speech commands: A dataset for limited-vocabulary speech recognition," *arXiv preprint arXiv:1804.03209*, 2018.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[24] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[25] S. M. Arnold, P. Mahajan, D. Datta, I. Bunner, and K. S. Zarkias, "learn2learn: A library for meta-learning research," *arXiv preprint arXiv:2008.12284*, 2020.