Investigating Meta-Learning Algorithms for Low-Resource Natural Language Understanding Tasks

Zi-Yi Dou, Keyi Yu, Antonios Anastasopoulos
Language Technologies Institute, Carnegie Mellon University
{zdou, keyiy, aanastas}@cs.cmu.edu

Abstract

Learning general representations of text is a fundamental problem for many natural language understanding (NLU) tasks. Previously, researchers have proposed to use language model pre-training and multi-task learning to learn robust representations. However, these methods can achieve sub-optimal performance in low-resource scenarios. Inspired by the recent success of optimization-based meta-learning algorithms, in this paper, we explore the model-agnostic meta-learning algorithm (MAML) and its variants for low-resource NLU tasks. We validate our methods on the GLUE benchmark and show that our proposed models can outperform several strong baselines. We further empirically demonstrate that the learned representations can be adapted to new tasks efficiently and effectively.

1 Introduction

With the ability to learn rich distributed representations of data in an end-to-end fashion, deep neural networks have achieved the state of the arts in a variety of fields (He et al., 2017; Vaswani et al., 2017; Povey et al., 2018; Yu et al., 2018). For natural language understanding (NLU) tasks, robust and flexible language representations can be adapted to new tasks or domains efficiently. Aiming at learning representations that are not exclusively tailored to any specific tasks or domains, researchers have proposed several ways to learn general language representations.

Recently, there is a trend of learning universal language representations via language model pre-training (Dai and Le, 2015; Peters et al., 2018; Radford et al., 2018). In particular, Devlin et al. (2019) present the BERT model which is based on a bidirectional Transformer (Vaswani et al., 2017). BERT is pre-trained with both masked language model and next sentence prediction objectives and exhibits strong performance on several benchmarks, attracting huge attention from researchers. Another line of research tries to apply multi-task learning to representation learning (Liu et al., 2015; Luong et al., 2015). Multi-task learning allows the model to leverage supervision signals from related tasks and prevents the model from overfitting to a single task. By combining the strengths of both language model pre-training and multi-task learning, Liu et al. (2019b) improve the BERT model with multi-task learning and their proposed MT-DNN model successfully achieves state-of-the-art results on several NLU tasks.

Although multi-task learning can achieve promising performance, there still exist some potential problems. As shown in Figure 1, multi-task learning may favor tasks with significantly larger amounts of data than others. Liu et al. (2019b) alleviate this problem by adding an additional fine-tuning stage after multi-task learning. In this paper, we propose to apply meta-learning algorithms in general language represen-
tations learning. Meta-learning algorithms aim at learning good initializations that can be useful for fine-tuning on various tasks with minimal training data, which makes them appealing alternatives to multi-task learning. Specifically, we investigate the recently proposed model-agnostic meta-learning algorithm (MAML) (Finn et al., 2017) and its variants, namely first-order MAML and Reptile (Nichol et al., 2018), for NLU tasks.

We evaluate the effectiveness and generalization ability of the proposed approaches on the General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2019). Experimental results demonstrate that our approaches successfully outperform strong baseline models on the four low-resource tasks. In addition, we test generalization capacity of the models by fine-tuning them on a new task, and the results reveal that the representations learned by our models can be adapted to new tasks more effectively compared with baseline models.

2 Proposed Approaches

In this section, we first briefly introduce some key ideas of meta learning, and then illustrate how we apply meta-learning algorithms in language representations learning.

2.1 Background: Meta Learning

Meta-learning, or learning-to-learn, has recently attracted researchers’ interests in the machine learning community (Lake et al., 2015). The goal of meta-learning algorithms is to allow fast adaptation on new training data. In this paper, we mainly focus on optimization-based meta-learning algorithms, which achieve the goal by adjusting the optimization algorithm. Specifically, we investigate MAML, one of the most representative algorithms in this category, and its variants for NLU tasks.

MAML and its variants offer a way to learn from a distribution of tasks and adapt to target tasks using few samples. Formally, given a set of tasks \( \{ T_1, T_2, \cdots, T_k \} \), the process of learning model parameters \( \theta \) can be understood as (Gu et al., 2018):

\[
\theta^*_i = \text{Learn}(T_i; \text{MetaLearn}(T_1, \cdots, T_k)),
\]

where \( T_i \) is the target task.

Hopefully, by exposing models to a variety of tasks, the models can learn new tasks with few steps and minimal amounts of data.

2.2 General Framework

In this part, we introduce the general framework of the MAML approach and its variants, including first-order MAML and Reptile.

Algorithm 1 Training procedure.

Pre-train model parameters \( \theta \) with unlabeled datasets.

\[
\text{while not done do}
\]

Sample batch of tasks \( \{ T_i \} \sim p(T) \)

\[
\text{for all } T_i \text{ do}
\]

Compute \( \theta_i^{(k)} \) with Eqn. 1.

\[
\text{end for}
\]

Update \( \theta \) with Eqn. 2.

\[
\text{end while}
\]

Fine-tune \( \theta \) on the target task.

We first describe the meta-learning stage. Suppose we are given a model \( f_\theta \) with parameters \( \theta \) and a task distribution \( p(T) \) over a set of tasks \( \{ T_1, T_2, \cdots, T_k \} \), at each step during the meta-learning stage, we first sample a batch of tasks \( \{ T_i \} \sim p(T) \), and then update the model parameters by \( k \) \((k \geq 1)\) gradient descent steps for each task \( T_i \) according to the equation:

\[
\theta_i^{(k)} = \theta_i^{(k-1)} - \alpha \nabla_{\theta_i^{(k-1)}} L_i(f_{\theta_i^{(k-1)}}),
\]

where \( L_i \) is the loss function for \( T_i \) and \( \alpha \) is a hyper-parameter.

The model parameters \( \theta \) are then updated by:

\[
\theta = \text{MetaUpdate}(\theta; \{ \theta_i^{(k)} \}).
\]

We would illustrate the MetaUpdate step in the following part. It should be noted that the data used for the MetaUpdate step (Eqn. 2) is different from that used for the first \( k \) gradient descent steps (Eqn. 1).

The overall training procedure is shown in Algorithm 1. Basically, the algorithm consists of three stages: the pre-training stage as in BERT, the meta-learning stage and the fine-tuning stage.

2.3 The MetaUpdate Step

As demonstrated in the previous paragraph, MetaUpdate is an important step in the meta-learning stage. In this paper, we investigate three ways to perform MetaUpdate as described in the following parts.
MAML The vanilla MAML algorithm (Finn et al., 2017) updates the model with the meta-objective function:
\[
\min_\theta \sum_{T_i \sim p(T)} L_i(f_{\theta^{(k)}})
\]

Therefore, MAML would implement the MetaUpdate step by updating \( \theta \) according to:
\[
\theta = \theta - \beta \sum_{T_i \sim p(T)} \nabla_\theta L_i(f_{\theta^{(k)}}),
\]
where \( \beta \) is a hyper-parameter.

First-Order MAML Suppose \( \theta^{(k)} \) is obtained by performing \( k \) inner gradient steps starting from the initial parameter \( \theta^{(0)} \), we can deduce that:
\[
\nabla_{\theta^{(0)}} L(f_{\theta^{(k)}}) = \nabla_{\theta^{(k)}} L(f_{\theta^{(k)}}) \prod_{i=1}^k \nabla_{\theta^{(i-1)}} \theta^{(i)}
\]
\[
= \nabla_{\theta^{(k)}} L(f_{\theta^{(k)}}) \prod_{i=1}^k (I - \alpha \nabla_{\theta^{(i-1)}} L(f_{\theta^{(i-1)}})).
\]

Therefore, MAML requires calculating second derivatives, which can be both computationally and memory intensive. First-Order MAML (FOMAML) ignores the second derivative part and implement the MetaUpdate as:
\[
\theta = \theta - \beta \sum_{T_i \sim p(T)} \nabla_{\theta^{(k)}} L_i(f_{\theta^{(k)}}).\]

Reptile Reptile (Nichol et al., 2018) is another first-order gradient-based meta-learning algorithm that is similar to joint training, as it implements the MetaUpdate step as:
\[
\theta = \theta + \beta \frac{1}{|\{T_i\}|} \sum_{T_i \sim p(T)} (f_{\theta^{(k)}} - \theta).
\]

Basically, Reptile moves the model weights towards new parameters obtained by multiple gradient descent steps. Despite the simplicity of Reptile, it has been demonstrated to achieve competitive or superior performance compared to MAML.

2.4 Choosing the Task Distributions
We experiment with three different choices of the task distribution \( p(T) \). Specifically, we propose the following options:
- **Uniform**: sample tasks uniformly.
- **Probability Proportional to Size (PPS)**: the probability of selecting a task is proportional to the size of its dataset.
- **Mixed**: at each epoch, we first sample tasks uniformly and then exclusively select the target task.

3 Experiments
We conduct experiments on the GLUE dataset (Wang et al., 2019) and only on English. Following previous work (Devlin et al., 2019; Liu et al., 2019b) we do not train or test models on the WNLI dataset (Levesque et al., 2012). We treat the four high-resource tasks, namely SST-2 (Socher et al., 2013), QQP, MNLI (Williams et al., 2018), and QNLI (Rajpurkar et al., 2016), as auxiliary tasks. The other four tasks, namely CoLA (Warstadt et al., 2018), MRPC (Dolan and Brockett, 2005), STS-B (Cerasa et al., 2017), and RTE (Dagan et al., 2005) are our target tasks. We also evaluate the generalization ability of our approaches on the SciTail dataset (Khot et al., 2018). The details of all datasets are illustrated in Appendix A.

We compare our models with two strong baselines: the BERT model (Devlin et al., 2019) and the MT-DNN model (Liu et al., 2019b). While the former pre-trains the Transformer model on large amounts of unlabeled dataset, the latter further improves it with multi-task learning.

For BERT and MT-DNN, we use their publicly available code to obtain the final results. The setting of MT-DNN is slightly different from the setting of BERT in terms of optimizer choices. We implement our algorithms upon the BERT BASE

| Model       | Test Dataset |
|-------------|--------------|
| CoLA        | MRPC         | STS-B | RTE |
| BERT        | 52.1         | 88.9/84.8 | 87.1/85.8 | 66.4 |
| MT-DNN      | 51.7         | 89.9/86.3 | 87.6/86.8 | 75.4 |
| FOMAML      | -            | -      | -      | -    |
| Reptile     | -            | -      | -      | -    |
| MAML        | -            | -      | -      | -    |
| MT-DNN      | 53.4         | 89.5/85.8 | 88.0/87.3 | 76.4 |
| BERT        | 51.6         | 89.9/86.4 | 88.6/88.0 | 74.1 |
| Featurizer   | 53.2         | 90.2/86.7 | 88.7/88.1 | 77.0 |

Table 1: Results on GLUE test sets. Metrics differ per task (explained in Appendix A) but the best result is highlighted.

1. data.quora.com/First-Quora-DatasetRelease-Question-Pairs


| Model                | CoLA | MRPC | STS-B | RTE   |
|----------------------|------|------|-------|-------|
| Reptile-PPS          | 61.6 | 90.0 | 90.3  | 83.0  |
| Reptile-Uniform      | 61.5 | 84.0 | 90.3  | 75.7  |
| Reptile-Mixed 2:1    | 60.3 | 87.8 | 90.1  | 71.0  |
| Reptile-Mixed 5:1    | 61.6 | 85.8 | 90.1  | 71.0  |

Table 2: Effect of task distributions. We report the accuracy or Matthews correlation on development sets.

We use the Adam optimizer (Kingma and Ba, 2015) with a batch size of 32 and learning rates of 5e-5 to train the models for 5 epochs in the meta-learning stage. We set the update step $k$ to 5, the number of sampled tasks in each step to 8 and $\alpha$ to 1e-3.

3.1 Results

We first use the three meta-learning algorithms with PPS sampling and present in Table 1 the experimental results on the GLUE test set. Generally, the meta-learning algorithms achieve better performance than the strong baseline models, with Reptile performing the best.

Since the MT-DNN also uses PPS sampling, the improvements suggest meta-learning algorithms can indeed learn better representations compared with multi-task learning. Reptile outperforming MAML indicates that reptile is a more effective and efficient algorithm compared with MAML in our setting.

3.2 Ablation Studies

Effect of Task Distributions As we have mentioned above, we propose three different choices of the task distribution $p(T)$ in this paper. Here we train Reptile with these task distributions and test models’ performance on the development set as shown in Table 2.

For uniform sampling, we set the number of training steps equal to that of the PPS method. For mixed sampling, we try mix ratios of both 2:1 and 5:1. The results demonstrate that Reptile with PPS sampling achieves the best performance, which suggests that larger amounts of auxiliary task data can generally lead to better performance.

Effect of Hyperparameters for Meta-Gradients

In this part, we test the effect of the number of update steps $k$ and the learning rate in the inner learning loop. The experimental results on the development sets are shown in Table 3. We find that setting $k$ to 5 is the optimal strategy and more or fewer update steps may lead to worse performance.

Smaller $k$ would make the algorithms similar to joint training as joint training is an extreme case of Reptile where $k = 1$, and thus cause the model to lose the advantage of using meta-learning algorithms. Similarly, Larger $k$ can make the resulting gradients deviate from the normal ones and become uninformative.

We also vary the inner learning rate $\alpha$ and investigate its impact. The results are listed in Table 3. We can see that larger $\alpha$ may degrade the performance because the resulting gradients deviate a lot from normal ones. The above two ablations studies demonstrate the importance of making the meta-gradient informative.

3.3 Transferring to New Tasks

In this part, we test whether our learned representations can be adapted to new tasks efficiently. To this end, we perform transfer learning experiments on a new natural language inference dataset, namely SciTail.

We randomly sample 0.1%, 1%, 10% and 100% of the training data and test models’ performance on these datasets. Figure 2 reveals that our model consistently outperforms the strong MT-DNN.
DNN baseline across different settings, indicating the learned representations are more effective for transfer learning. In particular, the algorithm is more effective when less data are available, especially compared to BERT, suggesting the meta-learning algorithms can indeed be helpful for low-resource tasks.

4 Related Work

There is a long history of learning general language representations. Previous work on learning general language representations focus on learning word (Mikolov et al., 2013; Pennington et al., 2014) or sentence representations (Le and Mikolov, 2014; Kiros et al., 2015) that are helpful for downstream tasks. Recently, there is a trend of learning contextualized word embeddings (Dai and Le, 2015; McCann et al., 2017; Peters et al., 2018; Howard and Ruder, 2018). One representative approach is the BERT model (Devlin et al., 2019) which learns contextualized word embeddings via bidirectional Transformer models.

Another line of research on learning representations focus on multi-task learning (Collobert et al., 2011; Liu et al., 2015). In particular, Liu et al. (2019b) propose to combine multi-task learning with language model pre-training and demonstrate the two methods are complementary to each other.

Meta-learning algorithms have received lots of attention recently due to their effectiveness (Finn et al., 2017; Fan et al., 2018). However, the potential of applying meta-learning algorithms in NLU tasks have not been fully investigated yet. Gu et al. (2018) have tried to apply first-order MAML in machine translation and Qian and Yu (2019) propose to address the domain adaptation problem in dialogue generation by using MAML. To the best of our knowledge, the Reptile algorithm, which is simpler than MAML and potentially more useful, has been given less attention.

5 Conclusion

In this paper, we investigate three optimization-based meta-learning algorithms for low-resource NLU tasks. We demonstrate the effectiveness of these algorithms and perform a fair amount of ablation studies. We also show the learned representations can be adapted to new tasks effectively. Our study suggests promising applications of meta-learning algorithms in the field of NLU. Future directions include integrating more sophisticated training strategies of meta-learning algorithms as well as validating our algorithms on other datasets.

Acknowledgements

The authors are grateful to the anonymous reviewers for their constructive comments, and to Graham Neubig and Junxian He for helpful discussions. This material is based upon work generously supported partly by the National Science Foundation under grant 1761548.

References

Daniel Cera, Mona Diabb, Eneko Agirrec, Inigo Lopez-Gazpioc, Lucia Speciad, and Basque Country Donostia. 2017. Semeval-2017 task 1: Semantic textual similarity multilingual and cross-lingual focused evaluation. In 11th International Workshop on Semantic Evaluations.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. JMLR.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single \ &!#* vector: Probing sentence embeddings for linguistic properties. In ACL.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In Machine Learning Challenges Workshop, pages 177–190. Springer.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In NeurIPS.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In NAACL.

William B Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In Proceedings of the Third International Workshop on Paraphrasing (IWP2005).

Yang Fan, Fei Tian, Tao Qin, Xiang-Yang Li, and Tie-Yan Liu. 2018. Learning to teach. In ICLR.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In ICML.

Jiatao Gu, Yong Wang, Yun Chen, Victor OK Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In EMNLP.
The GLUE and SciTail Datasets

Basically, the GLUE dataset (Wang et al., 2019) consists of three types of tasks: single-sentence classification, similarity and paraphrase tasks, and inference tasks, as shown in Table 4.
| Corpus  | Task          | # Train | # Label | Metrics                      |
|---------|---------------|---------|---------|------------------------------|
| CoLA    | Acceptability | 8.5k    | 2       | Matthews correlation         |
| SST-2   | Sentiment     | 67k     | 2       | Accuracy                     |

**Similarity and Paraphrase Tasks**

| Task          | # Train | # Label | Metrics                          |
|---------------|---------|---------|----------------------------------|
| MRPC          | 3.7k    | 2       | F1/Accuracy                      |
| STS-B         | 7k      | 1       | Pearson/Spearman correlation     |
| QQP           | 364k    | 2       | F1/Accuracy                      |

**Inference Tasks**

| Task          | # Train | # Label | Metrics |
|---------------|---------|---------|---------|
| MNLI          | 393k    | 3       | Accuracy|
| QNLI          | 105k    | 2       | Accuracy|
| RTE           | 2.5k    | 2       | Accuracy|
| WNLI          | 634     | 2       | Accuracy|
| SciTail       | 23.5k   | 2       | Accuracy|

Table 4: Basic information and statistics of the GLUE and SciTail datasets (Williams et al., 2018).

**Single-Sentence Classification.** The model needs to make a prediction given a single sentence for this type of tasks. The goal of the CoLA task is to predict whether an English sentence is grammatically plausible and the goal of the SST-2 task is to determine whether the sentiment of a sentence is positive or negative.

**Similarity and Paraphrase Tasks.** For this type of tasks, the model needs to determine whether or to what extent two given sentences are semantically similar to each other. Both the MRPC and the QQP tasks are classification tasks that require the model to predict whether the sentences in a pair are semantically equivalent. The STS-B task, on the other hand, is a regression task and requires the model to output a real-value score representing the semantic similarity of the two sentences.

**Inference Tasks.** Both the RTE and the MNLI tasks aim at predicting whether a sentence is entailment, contradiction or neutral with respect to the other. QNLI is converted from a question answering dataset, and the task is to determine whether the context sentence contains the answer to the question. WNLI is to predict if the sentence with the pronoun substituted is entailed by the original sentence. Because the test set is imbalanced and the development set is adversarial, so far none of the proposed models could surpass the performance of the simple majority voting strategy. Therefore, we do not use the WNLI dataset in this paper.

SciTail is a textual entailment dataset that is derived from a science question answering dataset (Khot et al., 2018). Given a premise and a hypothesis, the model need to determine whether the premise entails the hypothesis. The dataset is fairly difficult as the sentences are linguistically challenging and the lexical similarity of premise and hypothesis is high.

**B Implementation Details**

Our implementation is based on the PyTorch implementation of BERT. We first load the pre-trained BERT\_BASE model. We use the Adam optimizer (Kingma and Ba, 2015) with a batch size of 32 for both meta-learning and fine-tuning. We set the maximum length to 80 to reduce GPU memory usages.

In the meta-learning stage, we use a learning rate of 5e-5 to train the models for 5 epochs. Both the dropout and the warm-up ratio are set to 0.1 and we do not use gradient clipping. We set the update step $k$ to 5, the number of sampled tasks in each step to 8 and $\alpha$ to 1e-3.

For fine-tuning, again the dropout and warm-up ratio are set to 0.1 and we do not use gradient clipping. The learning rate is selected from $\{5e-6, 1e-5, 2e-5, 5e-5\}$ and the number of epochs is selected from $\{3, 5, 10, 20\}$. We select hyperparameters that achieve the best performance on the development set.

3\https://github.com/huggingface/pytorch-pretrained-BERT
We do not use the stochastic answer network as in MT-DNN for efficiency.

C Linguistic Information

In this part, we use 10 probing tasks (Conneau et al., 2018) to study what linguistic information is captured by each layer of the models.

A probing task is a classification problem that requires the model to make predictions related to certain linguistic properties of sentences. The abbreviations for the 10 tasks are listed in Table 5. Basically, these tasks are set to test the model’s abilities to capture surface, syntactic or semantic information. We refer the reader to Conneau et al. (2018) for details. We freeze all the parameters of the models and only train the classification layer for the probing tasks.

First, we can see that the BERT model captures more surface, syntactic and semantic information than other models, suggesting it learns more general representations. MT-DNN and our models, on the other hand, learn representations that are more tailored to the GLUE tasks.

Second, our models perform better than MT-DNN on the probing tasks, indicating meta-learning algorithms may find a balance between general linguistic information and task-specific information. Among the three meta-learning algorithms, Reptile can capture more general linguistic information than others. Considering Reptile has outperformed the other two models on the GLUE dataset, these results further demonstrate Reptile may be more suitable for NLU tasks.

Third, we find that there may not always exist a monotonic trend on what linguistic information each layer captures. Also, contrary to the findings from Liu et al. (2019a) which suggest the middle layers of BERT are more transferable and contain more syntactic and semantic information, our experimental results demonstrate that this may not always be true. We conjecture this is because both syntactic and semantic information are broad concepts and the probing tasks in Liu et al. (2019a) may not cover all of them. For example, there exist a monotonic trend for SOMO while the middle layers of these models are better at tasks like Sub-Num.

Another interesting thing to note is that the lower layers of models perform rather poorly on the word content task, which tests whether the model can recover information about the original words in the sentence. We attribute this phenomenon to the use of subwords and position/token embeddings. In the higher layers, the model may gain more word-level information through the self-attention mechanism.
| Model                  | Surface | Syntactic | Semantic |
|-----------------------|---------|-----------|----------|
|                       | SentLen | Word      | TreeDep  | ToCo    | BShift  | Tense    | SubNum  | ObjNum  | SOMO  | Coln  |
| Majority Voting       | 16.67   | 0.10      | 17.88    | 5.00    | 50.00   | 50.00    | 50.00   | 50.00   | 50.13 | 50.00 |
| BERT-Layer 1          | 90.84   | 7.54      | 32.31    | 57.91   | 50.67   | 78.83    | 75.30   | 79.63   | 50.13 | 50.05 |
| BERT-Layer 6          | 69.87   | 1.06      | 31.96    | 76.97   | 79.66   | 86.19    | 84.33   | 77.58   | 57.73 | 63.43 |
| BERT-Layer 12         | 63.15   | 32.98     | 28.80    | 71.36   | 85.67   | 89.72    | 76.13   | 76.52   | 60.92 | 70.91 |
| MTDNN-Layer 1         | 92.43   | 25.84     | 33.57    | 58.64   | 50.00   | 78.00    | 80.70   | 79.83   | 51.26 | 51.37 |
| MTDNN-Layer 6         | 80.11   | 21.41     | 31.73    | 59.58   | 76.00   | 81.89    | 80.36   | 80.00   | 55.52 | 58.31 |
| MTDNN-Layer 12        | 58.15   | 23.49     | 28.03    | 56.93   | 75.58   | 85.47    | 76.94   | 72.76   | 58.16 | 66.09 |
| MAML-Layer 1          | 92.26   | 20.9      | 30.64    | 55.27   | 50.00   | 77.71    | 72.61   | 70.44   | 50.13 | 52.49 |
| MAML-Layer 6          | 76.26   | 32.13     | 28.24    | 67.45   | 68.43   | 87.88    | 80.79   | 80.07   | 55.40 | 59.38 |
| MAML-Layer 12         | 61.50   | 20.32     | 27.31    | 60.15   | 79.47   | 85.56    | 77.60   | 75.86   | 56.76 | 63.59 |
| FOMAML-Layer 1        | 88.39   | 3.22      | 30.91    | 51.01   | 49.97   | 79.56    | 74.53   | 71.28   | 50.13 | 50.00 |
| FOMAML-Layer 6        | 81.33   | 22.63     | 30.44    | 69.48   | 77.01   | 88.89    | 81.81   | 80.18   | 57.93 | 60.11 |
| FOMAML-Layer 12       | 62.93   | 30.84     | 28.33    | 59.15   | 79.96   | 87.60    | 79.33   | 77.98   | 58.05 | 64.58 |
| Reptile-Layer 1       | 87.97   | 3.26      | 30.00    | 52.48   | 50.74   | 80.48    | 74.32   | 70.90   | 50.13 | 50.00 |
| Reptile-Layer 6       | 77.55   | 24.52     | 30.74    | 69.18   | 75.20   | 88.42    | 82.11   | 81.03   | 58.52 | 61.39 |
| Reptile-Layer 12      | 60.02   | 29.07     | 27.78    | 59.00   | 82.95   | 87.34    | 77.75   | 75.21   | 59.23 | 67.60 |

Table 5: Accuracy numbers on the 10 probing tasks (Conneau et al., 2018).