Meta-Visualization: Investigating Rapid Learning and Feature Reuse

Rumen Dangovski [*1] **Kristian Georgiev** [*1] **Rogerio Junior** [*1]

Abstract

Gradient based meta-learning has established itself as a promising research direction for problems in *few-shot learning*: data-constrained training on an ensemble of tasks with quick adaptation on a task, unseen during training. Until recently, little has been known about the success of *model-agnostic meta-learners* (MAMLs), which are particularly useful to few shot learning. We shed light on the phenomenon of feature reuse in MAMLs through empirical visualizations of the loss landscapes of a variety of tasks. We develop *meta-visualization*: an efficient framework for visualization of any gradient based meta learning algorithm that can be used to generate hypotheses and guide model designs. The contributions of our work vary from augmenting research in the field of meta learning to explaining the success of the method through geometrical lens. Our code is available at https://github.com/kristian-georgiev/867-Project.

1. Introduction

MAMLs. Model-agnostic meta-learning has been a flourishing field of research (Finn et al., 2017; Finn, 2018). Despite not yielding the state of the art on few-shot learning tasks, the universality of MAMLs makes them an appealing choice to a plethora of applications: from continual and few-shot classification, through regression, to reinforcement learning. MAML falls under the category of gradient based meta-learning (Lopez-Paz & Ranzato, 2017; Finn et al., 2017) and consists of two optimization loops. An outer loop finds a good initialization, named *meta-initialization*, and an inner loop can efficiently learn the new task.

Feature reuse. The question about the effectiveness of MAML compares two hypotheses: (*i*) *rapid learning*, the meta-initialization allows for quick adaptation to new tasks; (*ii*) *feature reuse*, the meta-initialization is already transferable enough, so the feature extracting part of the network does not evolve significantly. Recently, empirical findings suggest that feature reuse is the dominating hypothesis (Raghu et al., 2019). To validate the findings, Raghu et al. provide a new algorithm which does not optimize the feature extracting part of the neural network in the inner loop. Revealing the tension between rapid learning and feature reuse for a general set of meta-learning algorithms and few-shot classification tasks would be of seminal contribution to the field, as it would provide an understanding of how MAML works.

Main problem. Similarly, in attempt to understand how MAML works, we now ask: How does the trajectory of the weights on the loss landscape behave during meta-learning across all tasks? Can we observe the phenomena of rapid learning and feature reuse in the geometry of the trajectory on the loss landscapes? Here we attempt to answer these questions.

Approach. Inspired by visual analysis of loss landscapes of neural networks (Li et al., 2018), we approach our main problem by developing a visualization technique for meta-learing algorithms, which we name *meta-initialization*. The objective of meta-initialization is plot the *outer loss*, i.e. the loss on the query set after adaptation on the support set.

Our contributions can be summarized as follows:

1. We develop a framework for visualizing the loss landscape of meta-learning algorithms;
2. Through our visualizations we validate the empirical gains of MAML vs standard joint training;
3. We analyze our visualizations from meta-learning and propose methods of probing the phenomena of rapid learning and feature reuse;
4. We publish our code in order to aid research on meta-learning.

The rest of this paper is organized as follows: In Section 2 we discuss related work. In Section 3 we explain the experimental setup in this paper. In Section 4 we elaborate on our main contribution of meta-visualization. In Section 5 we present our main results. In Section 6 we discuss interpretation of our results. In Section 7 we conclude and suggest promising directions of future work.

2. Related work

2.1. How to optimize MAMLs

Antoniou et al. (2019) provides good practices for training MAML models and in particular suggests that special care needs to be taken with batch normalization (Ioffe & Szegedy, 2015). In our work, to isolate the effect of inner and outer loop optimization, we *do not use* batch normalization in our architectures. Albeit this architecture choice reduces the accuracy of our models it provides a cleaner analysis. Fallah et al. (2019) prove convergence guarantees for MAML and first order MAML in nonconvex settings and also give insights on how to construct optimal MAMLs. Rajeswaran et al. (2019) suggest optimizing MAMLs in memory and time complexity through efficient methods of computing implicit meta-gradients. In contrast, our work takes a more empirical approach, by visualizing the performance of MAMLs on the loss landscape and inferring efficient optimizations from the visualziations.

2.2. How to understand MAMLs

Nichol et al. (2018) provides a Taylor series analysis to demonstrate that at first order, MAML optimizes not only for a join training objective, but also for an objective that maximizes the inner product between gradient updates across tasks: the latter objective suggesting a setting where transfer across tasks is achieved. Riemer et al. (2019) extend the applications of that analysis. Raghu et al. (2019) demonstrates empirically that feature reuse seems to be dominating rapid learning. Neither of these works have explored the geometry of loss landscapes of MAMLs empirically, which is our main objective in this paper.

2.3. Visualization of neural networks' loss landscapes.

Our approach is most similar to (Li et al., 2018) who provide an empirical study of visualization of loss landscapes, building on insights obtained from a history of works on optimizing the loss landscape of neural networks (Blum & Rivest, 1988; Zhang et al., 2017). The authors propose a visualization technique that allows them to study the structure of neural loss functions and from there to infer generalization properties. In a similar fashion, we visualize the neural loss landscape with the objective to understand the success of MAMLs. However, visualizing the neural loss function of MAMLs is not straightforward, thus we have to solve a series of challenges with we discuss in Section 4.

3. Experimental setup

3.1. Preliminaries

With mathcal notation we denote labeled datasets of the form $\mathcal{D} = \{(x, y)\}$, where x is an input tensor and y is a label. With superscripts we denote the task and with subscript we denote the index of a batch, e.g. $\mathcal{S}_i^{(t)}$ is the i-th batch for the support set of task t.

3.2. Few-shot learning.

Our experiments specify the framework of *few-shot learning* (Vinyals et al., 2016b; Santoro et al., 2016; Ravi & Larochelle, 2017). We focus on N-way/ K-shot classification, meaning that the model sees only K examples from a task, consisting of N randomly samples classes, before making a prediction. In our work we focus on $N = 5$ and $K = 5$, i.e. all of our experiments are 5-way/ 5-shot classification.

Support set Following our discussion above, for a task t, its *support* set $\mathcal{S}_t$ consists of NK examples, i.e. 5-way/ 5-shot gives 25 examples. The few-shot task is to train on $\mathcal{S}_t$ and evaluate the accuracy of a query set as follows.

Query set Analogously to $\mathcal{S}_t$ we denote the *query* set as $\mathcal{Q}_t$. The number of examples in $\mathcal{Q}_t$ is $N\tilde{K}$, where $\tilde{K}$ is the number of examples of a given class on which we want to test. We set $\tilde{K} = 15$ across all experiments, i.e. 5-way/ 5-shot gives 75 examples in $\mathcal{Q}_t$.

3.3. Algorithms for optimization

We analyze three natural and well-established algorithms for few-shot learning.

1. **SGD**. Suppose we want to train jointly on many tasks. A natural setting would be to follow this proceduce: starting from weights θ, sample a minibatch $\mathcal{D}_t$ for a task t. Compute the loss on $\mathcal{D}_t$ and update the parameters θ. Then sample again and repeat the process. We call this *single loop* optimization SGD since

2. **MAML** (Finn et al., 2017) offers an alternative to the single loop by breaking the optimization process into two loops as follows. We define a *meta-batch* to be a collection $\mathcal{M} := \{(\mathcal{S}_t, \mathcal{Q}_t)\}_t$ of pairs of support and query sets for tasks t. Then, starting from weights θ, which we call *slow weights*, we enter:

   Inner loop Separately, for each task t represented in $\mathcal{M}$ perform N gradient updates on θ using $\mathcal{S}_t$, thereby adapting on the support set, and arrive at weights ϕ_t, which we call *fast weights*.

   Outer loop Update the slow weights θ using the average of the losses of the query sets $\mathcal{Q}_t$ at ϕ_t.

Figure 1. Examples from the class "clock" in Quickdraw.

3.4. Datasets

We base our experiments on three standard datasets for few shot classification.

Omniglot. This dataset is a standard benchmark for few-shot learning. The dataset includes 1623 handwritten characters from 50 different alphabets. Each character has 20 examples in the datset. Each example is a grayscale 28x28 image. Following (Finn et al., 2017) we use the 1200 classes for training and 423 classes for testing.

Improvements on 5-way/ 5-shot Omniglot have been saturated, all achieving almost perfect accuracy. This necessitates more challenging benchmark datasets.

QuickDraw. This dataset has recently been introduced as a novel benchmark for meta-learning (Triantafillou et al., 2019). It consists of 50 million drawings consisting of 345 classes. We select 300 classes for training and 45 classes for testing.

Note that this dataset is significantly more challenging than Omniglot. Figure 1 demonstrates that there is a lot of variability within the class "clock," a phenomenon, characteristic for across all classes in Quickdraw. Therefore, we expect lower accuracy for Quickdraw than for Omniglot.

For Quickdraw the neural networks suffers more acutely from the covariance shift problem, because of the high variability within classes.

Constructing the tasks. In order to construct the tasks for the 5-way/ 5-shot experiments we sample 5 classes at random respecitvely from the train and test sets and for the 5 chosen classes we sample 20 random examples. Then we split the 20 random examples in 5 examples for the inner loop and 15 for the outer loop optimization. Thus we obtain an inner loop batch of 25 examples an outer loop batch of 75 examples for each tasks. Each meta-batch consists of 32 tasks, and we have 37 meta-batches in total for each dataset.

3.5. Models, additional hyperparameters and implementation

Model. Our model is a neural network NN that is a composition of a feature extractor F and a classification head H, i.e. $\mathsf{NN} = \mathsf{H} \circ \mathsf{F}$. The feature extractor $\mathcal{F}$ consist of three layers of 2D convolutions with zero padding, stride one, and kernel size three, and [input, output] channels as follows: [1, 64], [64, 64], [64, 64]. After each convolution we use 2D batch normalization with momentum 1.0 (Ioffe & Szegedy, 2015), then ReLU activations, followed by 2D max pooling with kernel size two. Since we start with 28x28 images the resulting 64 neurons from the feature extractor are inputs to the head $\mathcal{H}$, which is a linear layers with $N = 5$ output neurons and a $\mathsf{softmax}$ activation to model the predictions of NN.

Hyperparameters. We train each model for 50 epochs. We use Adam (Kingma & Ba, 2015) optimizer with learning rate 1e-3 for the outer loop and a Stochastic Gradient Descent optimizer with learning rate 1e-1 for the inner loop. We perform $N = 5$ adapting steps in the inner loop.

Implementation. Our code is unique because it combines the Higher library[1] with differentiable optimizers, data loaders from Learn2Learn[2] (which we hope to integrate with Meta-Dataset[3]) and our visualization tools. It has the potential to become a useful toolkit for researchers in meta-learning.

4. Meta-visualization

The main objective of this work is to analyze the evolution of the weights of the neural network. Since the claim of MAML is that the weights of the model are a good meta-initialization, a natural way to validate this statement is to plot the loss landscape of the model NN for a variety of tasks t and compare the results.

We describe this idea more formally now. Suppose NN is a neural network parametrized by weights $\theta \in \mathbb{R}^d$, each weight being a column vector. Suppose we initalize the network with weights θ_0 and then train for n epochs, saving the snapshots of the weights $\theta_1, \ldots, \theta_n$ along the way. Let us call the matrix

$$\mathbf{T} := [\theta_0, \ldots, \theta_n]^T \tag{1}$$

the *trajectory* of the weights, which at every row has a snapshot of the weights of the NN during training. Our goal is to find an affine subspace $V \subset \mathbb{R}^d$, such that V is isomorphic to the plane $\mathbb{R}^2$, that would enable us to visualize $\mathbf{T}$ efficiently, and be able to attach meaningful loss values to the plotted region around our trajectory. To address our goal we answer two questions. The first question we answer is: What is the correct loss to visualize?

[1] https://github.com/facebookresearch/higher

[2] https://github.com/learnables/learn2learn

[3] https://github.com/google-research/meta-dataset

4.1. Loss motivation from the optimization objective

The standard join training, applied to few-shot learning, optimizes the following objective

$$\theta^* := \arg\min_{\theta} \mathbb{E}_{t\in\text{tasks}}[f_t(\theta; \mathcal{S}_t)], \tag{2}$$

where f_t is the loss for task t given the support $\mathcal{S}_t$ (which in this case coincides with the query $\mathcal{Q}_t$, meaning that there is a single loop, i.e. no inner and outer loop distinction). In contrast MAML optimizes the following objective

$$\theta^* := \arg\min_{\theta} \mathbb{E}_{t\in\text{tasks}}[f_t(\theta - \eta\nabla f_t(\theta; \mathcal{S}_t); \mathcal{Q}_t)], \tag{3}$$

where for simplicity we show only one gradient step for adaptation. Therefore, when we plot the loss landscape it does not suffice to simply evaluate at a given query $\mathcal{Q}_t$ from a fixed task t, as one might naïvely follow the approach of (Li et al., 2018), because the inductive bias of θ is to be a good *initialization*, not optimal weights for task t. Thus, we have to adapt the weights on the support $\mathcal{S}_t$ before evaluating the loss on $\mathcal{Q}_t$. This observation necessitates a new setting for visualization, which to our knowledge has not been studied in the literature: visualization of meta-learning optimization algorithms, which we dub *meta-visualization.*

In order to define meta-visualization rigorously, we answer a second question: How to efficiently choose the affine subspace V? We would want to capture the most of the variation of the trajectory $\mathbf{T}$, and keep the distances between the original and projected weights small. To address the former property, we use *principle component analysis* (PCA) (Pearson, 1901; Li et al., 2018).

4.2. Principle Component Analysis

PCA is a good fit to our goal, since it attains dimensionality reduction by capturing most of the variance of the data. We perform the classical PCA algorithm on the matrix $\mathbf{T}$ to obtain the top-k principal component vectors $\delta_1, \ldots, \delta_k$. In practice, we use the first two principal components δ_1 and δ_2, which span a plane P given as follows

$$P = \{\alpha_1\delta_1 + \alpha_2\delta_2 \mid \alpha_1, \alpha_2 \in \mathbb{R}\}. \tag{4}$$

We would like to obtain the affine plane that reconstructs our trajectory with the smallest reconstruction (mean squared) error. Naturally, this means that we offset P by the center of mass of the trajectory, which is $\tilde{\theta} := 1/(n+1)\sum_{i=0}^{n}\cdot\theta_i$. Thus the affine space becomes

$$V = \{\tilde{\theta} + w : w \in P\}, \tag{5}$$

as desired.

Now, let us denote the projection of θ_i onto V by $\hat{\theta}_i$. Let $C \subset V$ be a compact region such that $\hat{\theta}_i \in C$ for all θ_i (such a region exists since we have a finite number of points). For example, C can simply be the smallest rectangle in which all the weights projections fit. We evaluate a test loss of the network parametrized with weights on a grid of points in C and then plot the results.

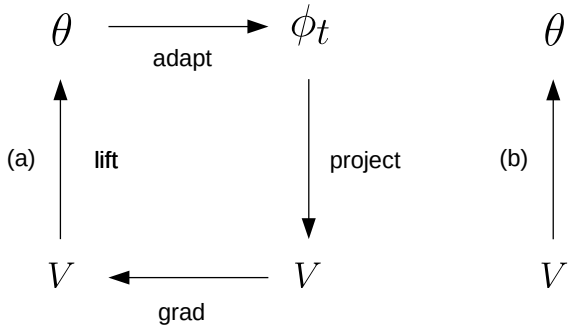

Figure 2. Visualizing the loss landscape of MAMLs requires adaptation and offers a measure of rapid learning. (*a*) Our approach: lifting, adapting, projecting and directing for a given task *t*; (*b*) The standard approach in (Li et al., 2018) that consist only of lifting and projecting.

4.3. Constructing vector fields

We demonstrate our proposition for meta-visualization on figure 2 for any task t. In (a) we start from the principles components $\delta_1, \ldots, \delta_k$ and the center of mass $\tilde{\theta}$ to *lift* to from V to a point θ. Then we set θ as a meta-initialization for the model NN, and adapt it to ϕ_t on the support $\mathcal{S}_t$ using MAML's inner loop proceduce. After this we can project ϕ_t back to V and use the projection to evaluate the loss on the query set $\mathcal{Q}_t$. Finally, we can use the original point and the adapted point to define the difference between the two vectors. In this way in our plot we generate a *vector field* where each vector at a point measures the vector of adaptation that θ undergoes during the inner loop. In (b) we compare our meta-visualization in (a) to the work by (Li et al., 2018), which concerns only lifting from V. We speculate that via observing the vector fields we might be able to define a notion of rapid learning, and leave that for future work.

5. Results

We begin our discussion with a simple toy task.

5.1. A simple toy study

In Figure 3 we present a simple proof of concept visualization of how MAML is supposed to work, compared to SGD. In this example we work with a neural network with only 2 neurons. We observe that SGD training on the three green tasks leaves the initialization at a place that is fur-

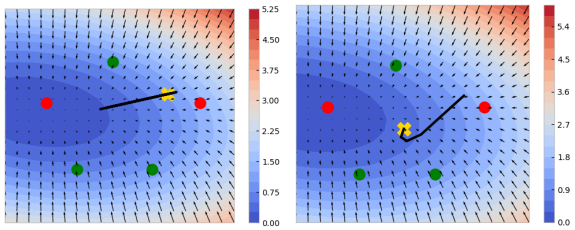

Figure 3. SGD (left) and MAML (right) loss landscapes for a toy task with a neural network with only two neurons. In green are training points and in red are the test points. Loss landscape is for one of the red points.

ther away from the minimum, compared to the "centroid" meta-initialization that we get from MAML. In this case, we clearly see that MAML yields a good initialization, and this visualization we would like to probe for more realistic settings.

We now continue in realistic settings and conduct the following set of studies: (*i*) varying the learning rate during adaptation in meta-visualization; (*ii*) varying the number of adaptation steps during adaptation in meta-visualization; (*iii*) observing the loss landscape of MAML on Omniglot and Quickdraw; (*iv*)) observing the loss landscape of SGD on Omniglot and Quickdraw; (*v*) Holding the feature extractor F fixed and varying the head H only in MAML; (*vi*) Holding the head H fixed and varying the feature extractor F only in MAML. First we train MAML and SGD on Omniglot and Quickdraw and report the results.

5.2. Training performance

Figure 4 addresses the training curves of our models, along with their performance on the test split. The accuracy that we plot is the accuracy on the query $\mathcal{Q}_t$ after fine-tuning on the support $\mathcal{S}_t$. We observe that MAML learns meaningful meta-initializations that allow it yield high accuracy, while SGD does not and thus essentially performs random guessing. Moreover, we observe that Quickdraw is a more challenging task than Omniglot, as expected from observing Figure 1.

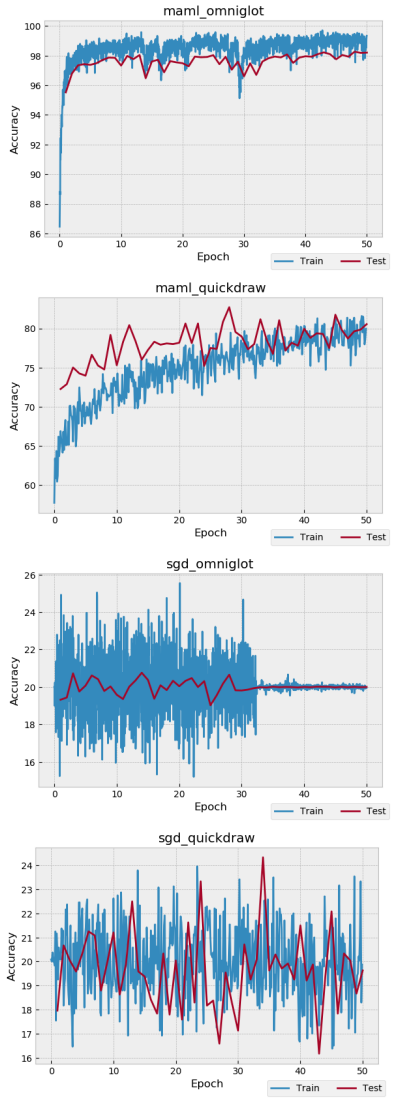

Figure 4. Training performance. We see that Quickdraw is a more challenging task than Omniglot. We also see that SGD is simply doing random guessing.

5.3. Varying the learning rate

We fix MAML and a particular task from Omniglot. We present our results in Figure 5. We observe that decreasing the learning rate from the default training learning rate (1e-1) increases the loss of the region, since the MAML adapts too slowly in the inner loop.

5.4. Varying the number of adaptation steps

We use the same MAML and task from Omniglot as in the previous subsection. We present our results in Figure 5. We observe that the best loss landscape is for $N = 5$, which is exactly the number of adaptation steps that we use for the inner loop during training. In contrast, when there are zero steps (and thus no vector fields, because of the lack of adaptation) there the loss suggests that the model is essentially doing random guessing. Moreover by observing the drastic change in the loss landscape for $N = 10$, we see that over-adapting on the support set does not help. We conjecture that the model might begin over-fitting on the support set and will study this in future work.

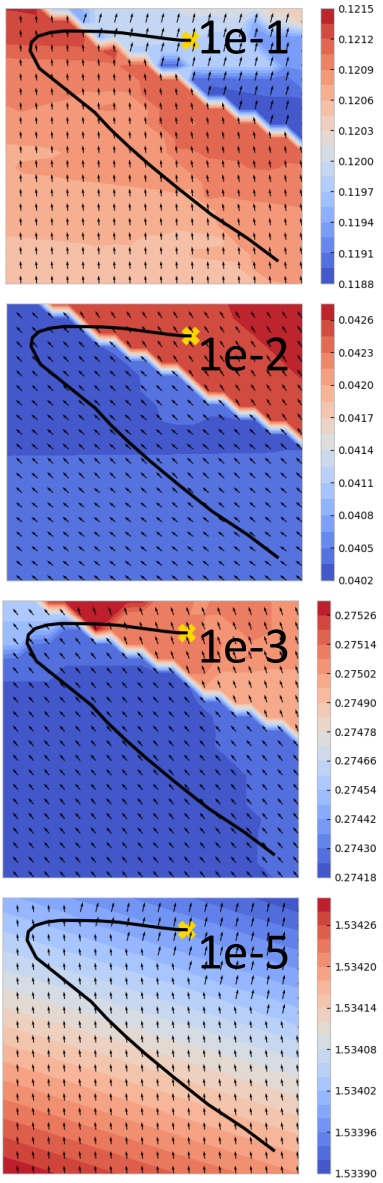

Figure 5. Finding the best learning rate for meta-visualization needs tuning. We see that here 1e-2 is the best choice, even though the models has been trained with 1e-1 in the inner loop. Final point labeled by cross. The last 20 points of the trajectory are plotted.

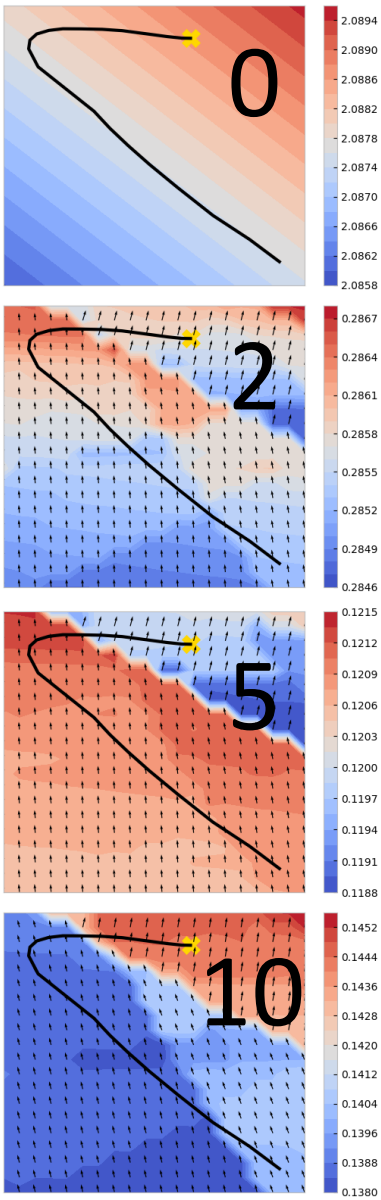

Figure 6. We see that adaptation is very important for meta-visualization. Namely, at zero steps of adaptation, meta-visualization of MAML is similar to that of SGD, suggesting that MAML with zero adaptation steps is simply random guessing. We see that the best performance is at $N = 5$ steps, which might be due to the fact that the model has been trained with this setting. Final point labeled by cross. The last 20 points of the trajectory are plotted.

5.5. Loss landscapes for MAML and SGD

We present our results in Figure 7. We observe that MAML always brings the outer loss to a low level, thus enabling high accuracy (dark green). In contrast SDG fails to bring the outer loss to a low level, thus it accuracy (light green) suggests that SGD is performing random guessing. Note that for each task there is a little variability both for the loss and the accuracy.

5.6. Fixed feature extractor and fixed head

We present our results in Figure 8 and Figure 9 respectively. We observe uninformative plots which are hard to interpret, possibly because we are plotting the situation only for a single task. Moreover, fixing one component of the NN and

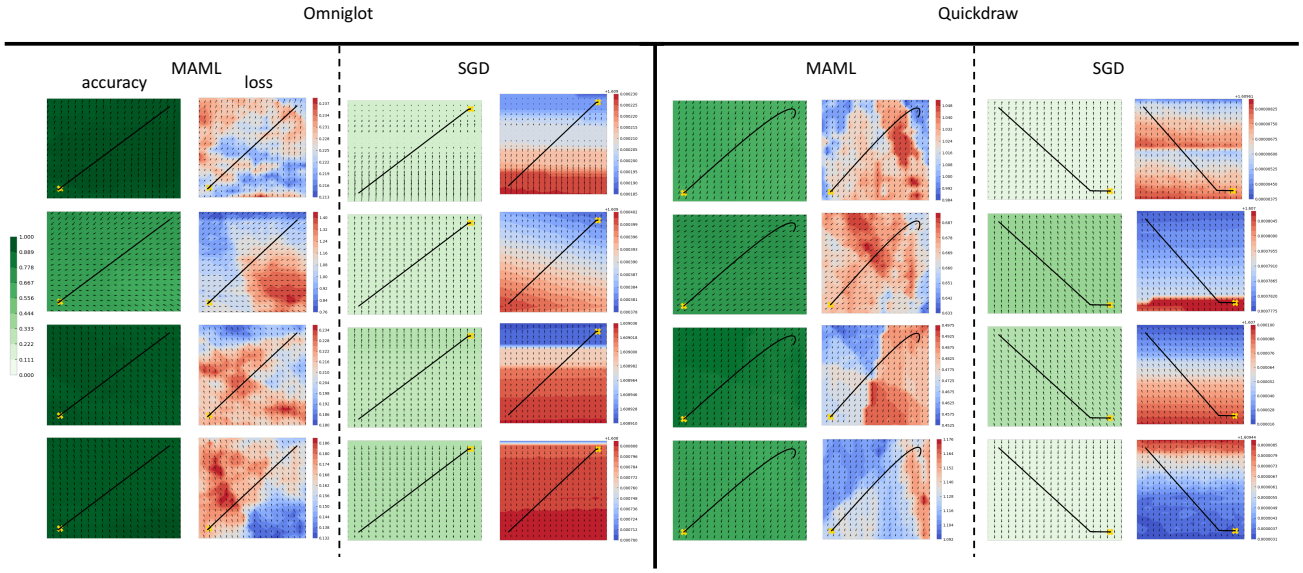

Figure 7. Comparing the landscapes of MAML and SGD we see that MAML ends at a point that is reasonable across tasks while SGD strikes at random and ends up doing random guessing (about 20% accuracy). Each new line is a new task for the respective dataset. Final point labeled by cross. The full trajectory is plotted. Best viewed in color.

varying the other component might require rethinking, since the weights of both components are coupled throughout training. Therefore, we leave futher analysis in this direction for future work.

6. Discussion

Note that all plots in the previous section exhibit sharp edges. We conjecture that this phenomenon might be due to the fact that we plot each loss landscape for a fixed task. Instead, we argue that averaging our meta-visualizations across a batch of tasks might smooth out the edges and yield a more informative plot. We test the smoothing in the case of varying the learning rate and report our results in Figure 10. Indeed, we observe that the loss landscapes are smoother and the trajectories keep on improving by following lower loss. We leave the smoothing of rest of our plots in the previous section for future work.

7. Conclusion and future work

In this work we presented meta-visualization. We observed some properties of feature reuse and rapid learning. Immediate future work suggests that we can vary the size of the models, the size of the datasets, the models, test other meta learning algorithms, and also test non-optimization based methods, i.e. Matching Networks (Vinyals et al., 2016a), etc.

Outlook. In this paper with meta-visualization we evaluate the outer loss (on the query set) after adapting on the support set. As we saw, this gave us a good measure of the success of MAML against SGD, but promising alternatives for measuring feature reuse and rapid learning exist. For example, supposet that we remove the head H from the model and consider a nearest neighbor loss L_t for task t which finds the nearest neighbor of the representation from the feature extractor F to prototypical examples' representations. A promising direction is to computer the *inner loss* for T tasks, given as follows

$$\frac{1}{T}\sum_{t=1}^{T} L_t(\theta; \mathcal{S}_t). \tag{6}$$

If this loss is small then we might have evidence of feature reuse, conditioned on low outer loss. Furthermore, if the loss 6 is high, while the outer loss is low, then we might observe evidence of rapid learning. We would attempt testing that conjecture on the collection of tasks (Triantafillou et al., 2019). Finally, we will attempt to use the meta-visualization tool to recognize similarity between tasks, which has the potential to yield novel testbeds for meta-learning.

References

Antoniou, A., Edwards, H., and Storkey, A. How to train your MAML. In *Proceedings of the 7th International Conference on Learning Representations*, ICLR '19, New Orleans, LA, USA, 2019.

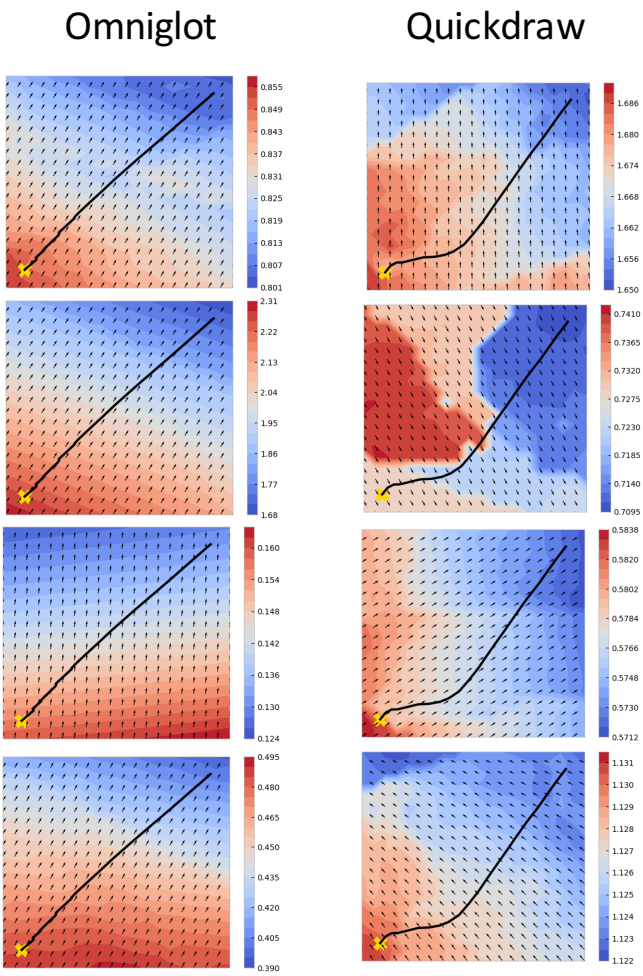

Figure 8. Meta-visualization of MAML when the extractor F. Final point labeled by cross. The last 20 points of the trajectory are plotted.

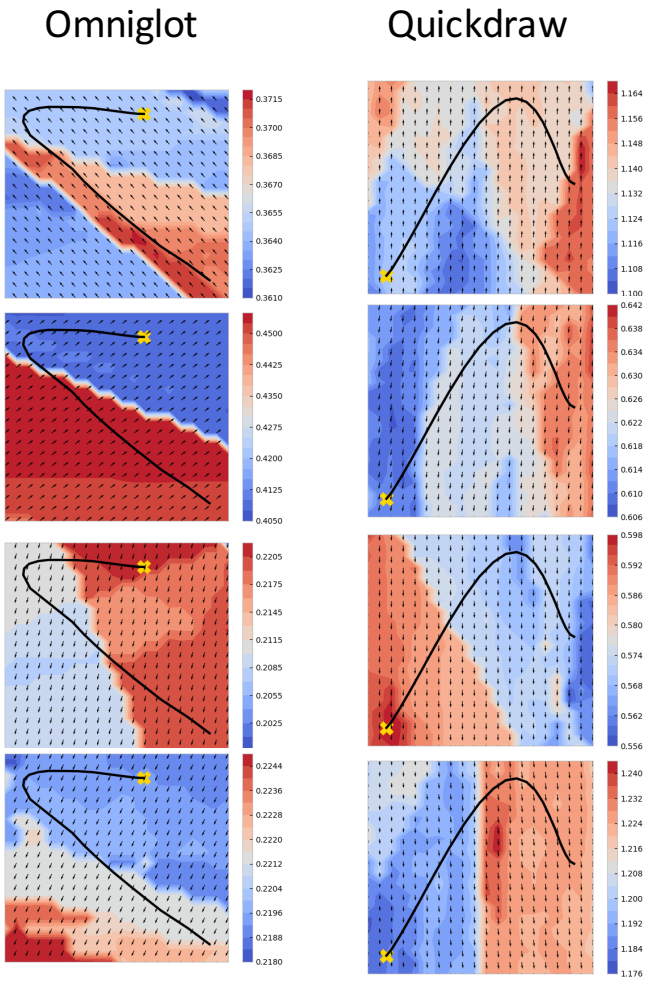

Figure 9. Fixed head H. Final point labeled by cross. The last 20 points of the trajectory are plotted.

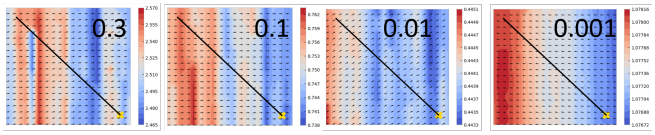

Figure 10. Averaging meta-visualizations across tasks in a batch yields smoother loss landscapes.

Blum, A. and Rivest, R. L. Training a 3-node neural network is np-complete. In *Proceedings of the First Annual Workshop on Computational Learning Theory*, COLT '88, pp. 9–18, San Francisco, CA, USA, 1988. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=93025.93033.

Fallah, A., Mokhtari, A., and Ozdaglar, A. On the convergence theory of gradient-basedmodel-agnostic meta-learning algorithms. *arXiv preprint arXiv:1908.10400*, 2019.

Finn, C. *Learning to Learn with Gradients*. PhD thesis, EECS Department, University of California, Berkeley, Aug 2018. URL http://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-105.html.

Finn, C., Abbeel, P., and Levine, S. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, ICML '17, pp. 1126–1135, Sydney, Australia, 2017.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate

shift. pp. 448–456, 2015. URL http://jmlr.org/proceedings/papers/v37/ioffe15.pdf.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, ICLR '15, San Diego, CA, USA, 2015.

Li, H., Xu, Z., Taylor, G., Studer, C., and Goldstein, T. Visualizing the loss landscape of neural nets. In *Advances in Neural Information Processing Systems*, pp. 6389–6399, 2018.

Lopez-Paz, D. and Ranzato, M. Gradient episodic memory for continuum learning. *NeurIPS*, 2017.

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

Pearson, K. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572, 1901.

Raghu, A., Raghu, M., Bengio, S., and Vinyals, O. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. *arXiv preprint arXiv:1909.09157*, 2019.

Rajeswaran, A., Finn, C., Kakade, S., and Levine, S. Meta-learning with implicit gradients. *arXiv preprint arXiv:1909.04630*, 2019.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, New Orleans, LA, USA, 2017.

Riemer, M., Cases, I., Ajemian, R., Liu, M., Rish, I., Tu, Y., and Tesauro, G. Learning to learn without forgetting by maximizing transfer and minimizing interference. In *Proceedings of the 7th International Conference on Learning Representations*, ICLR '19, New Orleans, LA, USA, 2019.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. Meta-learning with memory-augmented neural networks. In *Proceedings of the 33rd International Conference on Machine Learning (ICML 2000)*, pp. 1842–1850, 2016.

Triantafillou, E., Zhu, T., Dumoulin, V., Lamblin, P., Evci, U., Xu, K., Goroshin, R., Gelada, C., Swersky, K., Manzagol, P.-A., and Larochelle, H. Meta-dataset: A dataset of datasets for learning to learn from few examples. *arXiv preprint arXiv:1903.03096*, 2019.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. Matching networks for one shot learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pp. 3637–3645, USA, 2016a. Curran Associates Inc. ISBN 978-1-5108-3881-9. URL http://dl.acm.org/citation.cfm?id=3157382.3157504.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. Matching networks for one shot learning. In *Proceedings of the Annual Conference on Neural Information Processing Systems: Advances in Neural Information Processing Systems 30*, NeurIPS '16, Barcelona, Spain, 2016b.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. In *Proceedings of the 5th International Conference on Learning Representations*, ICLR '17, Toulon, France, 2017.