Model Agnostic Meta-Learning (MAML) and First Order Model Agnostic Meta-Learning (FoMAML) on few-shots classification problem

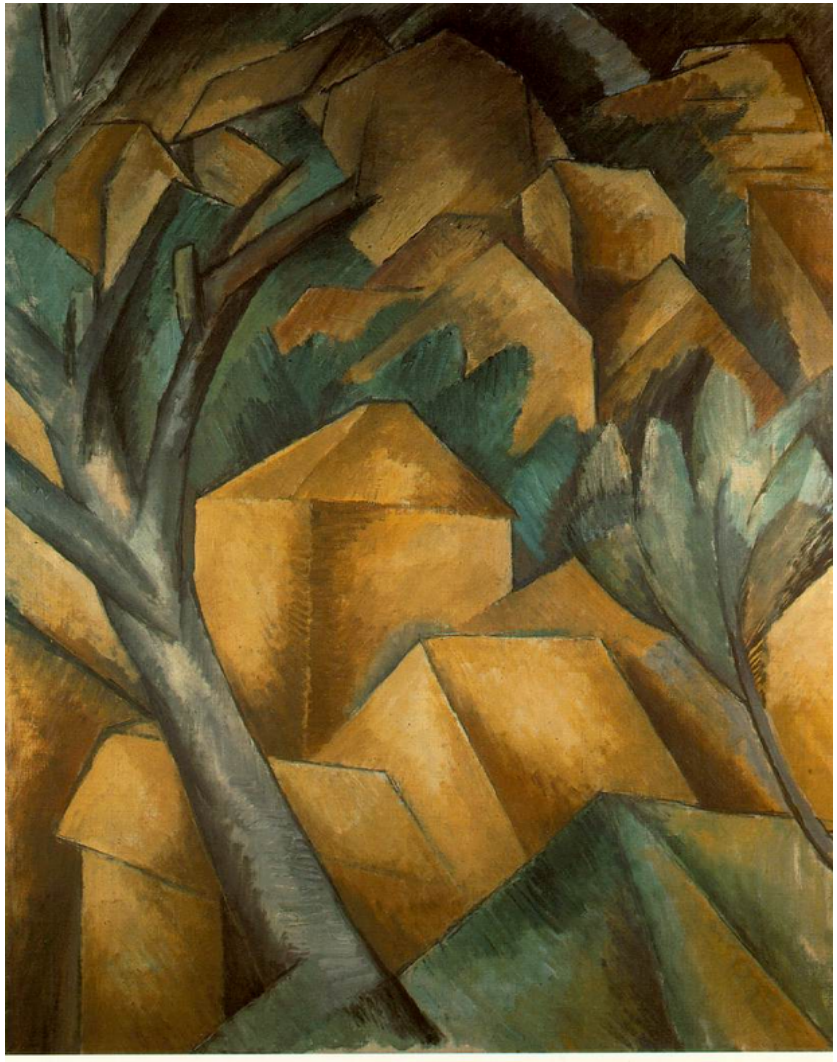
Saumya Vilas Roy *Student*
Deepak Mishra *Professor*

Indian Institute of Space Science and Technology
saumya@tutanota.com

Frontiers Symposium in Data Science 2024 (FS-DSC 2024)
February 10, 2024

Presentation Overview

Introduction to few-shots problem



(a) Braque - Houses at Estaque.

(b) Cézanne - Forest

Test Set

Figure: One shot learning problem

Test Set with labels

(a) Braque - Clarinet and Bottle of Rum on a Mantelpiece

(b) Cézanne - Mont Sainte-Victoire

Figure: One shot learning problem

Panting style analysis

(a) Braque - Bottle and Fishes

(b) Cézanne - Basket of Apples

As we can observe from the paintings above we find Cézanne to have a Post-Impressionism style subsequently Braque is following Analytical Cubism.

Few shots problem

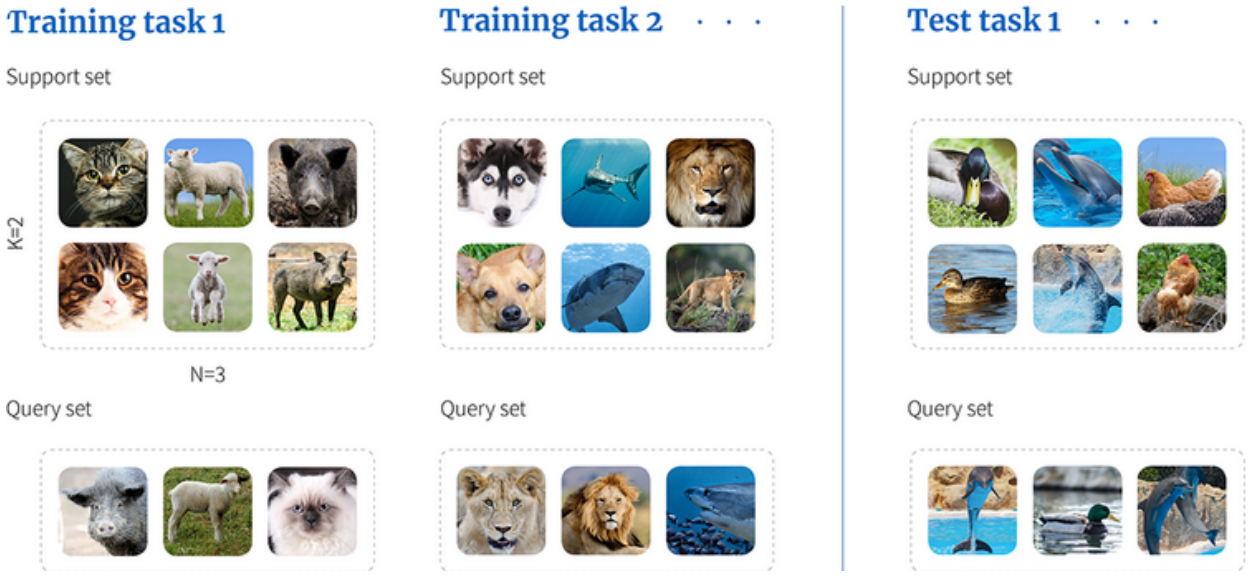


Figure: Few shot learning problems

Goal of a meta learning algorithm here is to find a set of hyper-parameters such that they can be tuned for the specific tasks over the least possible data and have a good accuracy.

Model-Agnostic Meta-learning (MAML)

- Goal:
  - Quickly adapt to new tasks on distribution with only small amount of data and with only a few gradient steps, even one gradient step.
- Learner:
  - Learn a new task based on previous tasks.
- Meta-Learner:
  - Learn a generalized parameter initialization of model.
- The MAML learner's **weights are updated using the gradient** rather than a learned rule.(No additional parameters required)
- **Model-agnostic** (Not dependent over the model)
  - Classification & regression with differentiable losses, even policy gradient RL.

Characteristics of MAML

- **Fast adaptability** though good parameter initialization
  - Explicitly optimizes to learn internal representations (i.e. suitable for many tasks).
  - Maximizes sensitivity of new task losses to the model parameters.
- **Task-agnostic** (Not selective over tasks)
  - Adopted all knowledge-transferable tasks.

No other assumptions required.

Intuition of MAML

- Some internal representations are **more transferable** than others.
- Desired model parameter set is Θ such that: Applying one (or very few) gradient steps to Θ on a new task will produce desirable results.
  - Finding Θ such that it decreases loss of each task after adaptation.

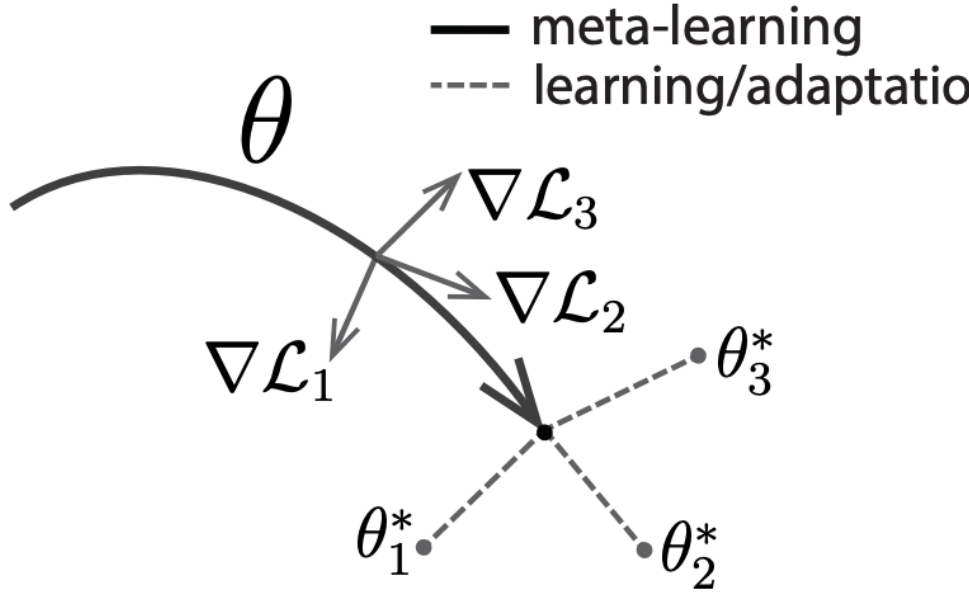


Figure: MAML parameter optimization.

MAML Algorithm

Pseudo-code

Algorithm 1 Model-Agnostic Meta-Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$
9: **end while**

Figure: MAML pseudo-code.

As we can observe in the above code MAML works by taking gradient of a gradient to find an appropriate Θ.

Theory

From line 8 in Pseudo-code,

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_{i \sim p(\mathcal{T})}} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

$\mathcal{L}$ is differentiable

$$\theta - \beta \sum_{\mathcal{T}_{i \sim p(\mathcal{T})}} \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

$$\theta - \beta \sum_{\mathcal{T}_{i \sim p(\mathcal{T})}} (\nabla_\theta \theta_i') \nabla_{\theta_i'} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

Recalling the gradient update rule:

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$$

$$\theta - \beta \sum_{\mathcal{T}_{i \sim p(\mathcal{T})}} (\mathcal{I} - \alpha \nabla_\theta^2 \mathcal{L}_{\mathcal{T}_i}(f_\theta)) \nabla_{\theta_i'} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$$

Here the highlighted portion entails calculation of the Hessian matrix and can be approximated by using 1^{st} order

1^{st} Order Approximation

- Update rule in MAML:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_{i \sim p(\mathcal{T})}} \mathcal{L}_{\mathcal{T}_i}(f_\theta - \alpha \nabla \mathcal{L}_{\mathcal{T}_i}(f_\theta))$$

  Here this makes MAML slow
- Update rule of MAML with 1st order approximation:

$$\delta \leftarrow \nabla \mathcal{L}_{\mathcal{T}_i}(f_\theta)$$

  Here δ is to be taken as constant
- After the approximation, the update rule is:

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_{i \sim p(\mathcal{T})}} \mathcal{L}_{\mathcal{T}_i}(f_\theta - \alpha \delta)$$

- This allows the Hessian calculation in the update rule to be just approximated to $\mathcal{I}$

Why 1^{st} order approximation works?

- The objective function of ReLU network is locally linear.
- So as the other piece-wise linear activations (maxout, leanky ReLU etc.) are used.
- Because , any composite function with linear or piece-wise linear function should be locally linear.

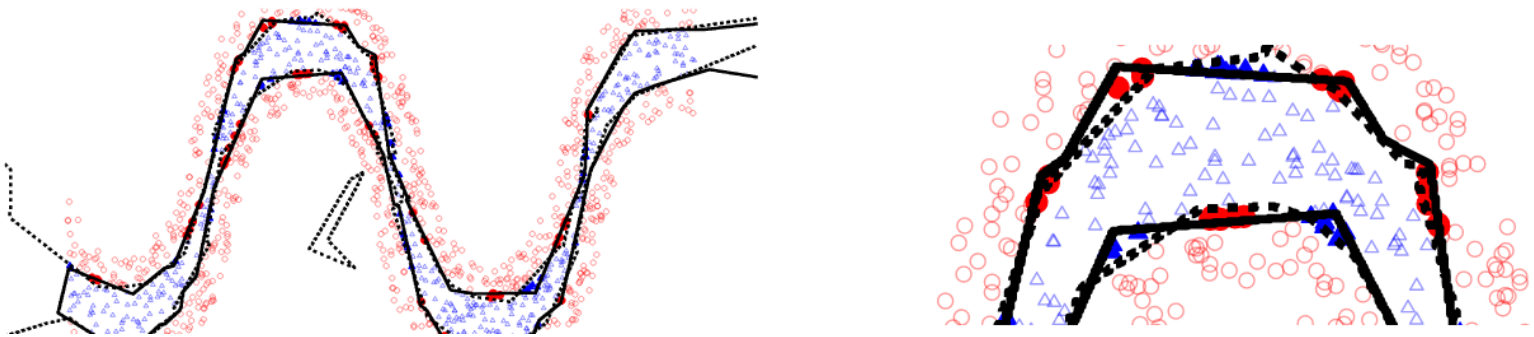


Figure: Continuity of objective functions of ReLU network.

- Hence, **the Hessian matrix is almost zero.** i.e. meaningless in our computations.

Data-sets used

- Omniglot
  - 50 different alphabets, 1623 characters.
  - 20 instances for each characters were drawn by 20 different people.
  - 1200 for training, 423 for test.
- Mini-imagenet
  - Classes for each set: train=64, validation=12, test=24

Results

- MAML outperforms sate-of-art algorithms

Omniglot (Lake et al., 2011)	5-way Accuracy		20-way Accuracy	
	1-shot	5-shot	1-shot	5-shot
MANN, no conv (Santoro et al., 2016)	82.8%	94.9%	–	–
MAML, no conv (ours)	**89.7 ± 1.1%**	**97.5 ± 0.6%**	–	–
Siamese nets (Koch, 2015)	97.3%	98.4%	88.2%	97.0%
matching nets (Vinyals et al., 2016)	98.1%	98.9%	93.8%	98.5%
neural statistician (Edwards & Storkey, 2017)	98.1%	99.5%	93.2%	98.1%
memory mod. (Kaiser et al., 2017)	98.4%	99.6%	95.0%	98.6%
MAML (ours)	**98.7 ± 0.4%**	**99.9 ± 0.1%**	**95.8 ± 0.3%**	**98.9 ± 0.2%**

Figure: MAML Results on Omniglot data-set.

- In 1^{st} order approximation, computation is roughly 33% better.
- Performances are similar.

MiniImagenet (Ravi & Larochelle, 2017)	5-way Accuracy	
	1-shot	5-shot
fine-tuning baseline	$28.86 \pm 0.54\%$	$49.79 \pm 0.79\%$
nearest neighbor baseline	$41.08 \pm 0.70\%$	$51.04 \pm 0.65\%$
matching nets (Vinyals et al., 2016)	$43.56 \pm 0.84\%$	$55.31 \pm 0.73\%$
meta-learner LSTM (Ravi & Larochelle, 2017)	$43.44 \pm 0.77\%$	$60.60 \pm 0.71\%$
MAML, first order approx. (ours)	$\mathbf{48.07 \pm 1.75\%}$	$\mathbf{63.15 \pm 0.91\%}$
MAML (ours)	$\mathbf{48.70 \pm 1.84\%}$	$\mathbf{63.11 \pm 0.92\%}$

Figure: FoMAML and MAML Results on MiniImagenet data-set.

Experimental setup and result analysis

With the parameters:

- ways=5 (No. of classes to be classified)
- shots=5 (No. of images/samples in each class)
- meta_lr=0.003 (Meta-learning gradient decent update learning rate)
- fast_lr=0.5 (Learning rate while using MAML)
- meta_batch_size=32 (Batch size for the MAML training)
- adaptation_steps=1 (No. of gradient steps for the fine tuning of the model parameter)
- num_iterations= 10000 (No. of Epochs for training)
- cuda=True (Setting the usage/availability of GPU)
- seed=42 (To generate same string of random numbers)

This training is done on NVIDIA GeForce GTX 1650 Mobile with Intel i7-9750H processor.
The running time has been sampled using the timeit library.
And the entire network has been built and deployed using

Result analysis

Metrics	FoMAML	MAML
Avg. Test Accuracy	0.9699	0.9587
Avg. Test Error	0.10566	0.12392
Execution Time (s)	23617.31	24875.94

- We can see FoMAML is producing higher accuracy for 10,000 epochs as due to the approximation it can converge much faster than MAML but if the number of epochs are increased the MAML is bound to have much higher accuracy as it would lead to complete convergence.
- We can also observe FoMAML to be faster by **5.05%** when comparing it with MAML as the total number of Epochs are only 10,000 it doesn't have much significance but for systems with multiple and higher number of epochs it is very significant.

References

- miscmontúfar2014number, title=On the Number of Linear Regions of Deep Neural Networks, author=Guido Montúfar and Razvan Pascanu and Kyunghyun Cho and Yoshua Bengio .
- miscfinn2017modelagnostic, title=Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks, author=Chelsea Finn and Pieter Abbeel and Sergey Levine.
- misc ndrychowicz, Marcin, Denil, Misha, Gomez, Sergio, Hoffman, Matthew W, Pfau, David, Schaul, Tom, and de Freitas, Nando. Learning to learn by gradient descent by gradient descent
- miscEdwards, Harrison and Storkey, Amos. Towards a neural statistician. International Conference on Learning Rep- resentations (ICLR), 2017
- misc Koch, Gregory. Siamese neural networks for one-shot im- age recognition. ICML Deep Learning Workshop, 2015
- misc Vinyals, Oriol, Blundell, Charles, Lillicrap, Tim, Wierstra, Daan, et al. Matching networks for one shot learning. In Neural Information Processing Systems (NIPS), 2016.

The End

Questions? Comments?