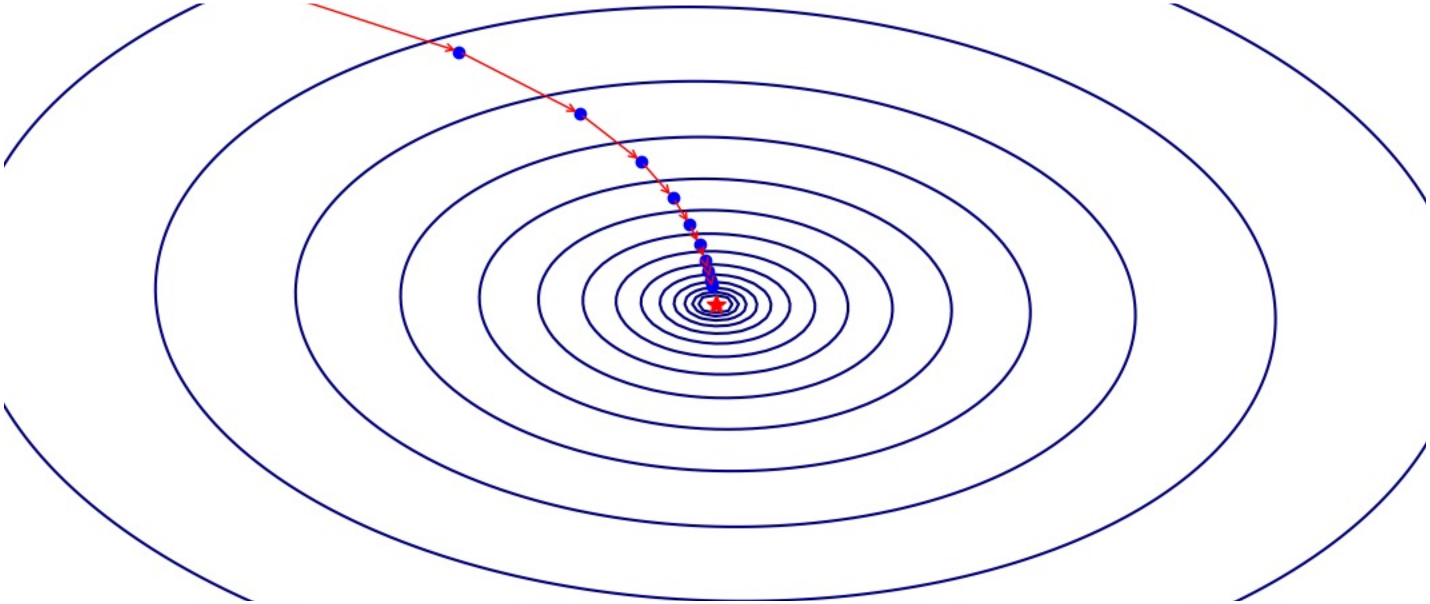


Advanced Topics in Deep Learning

Summer Semester 2024

8. Few-shot Learning

01.07.2024

Prof. Dr. Vasileios Belagiannis

Chair of Multimedia Communications and Signal Processing

Friedrich-Alexander-Universität
Technische Fakultät

Course Topics

1. Interpretability.
2. Attention and Transformers.
3. Self-supervised Learning I.
4. Self-supervised Learning II.
5. Similarity Learning.
6. Generative Models.
7. Model Compression.
8. **Few-shot learning.**
9. Uncertainty Estimation.
10. Geometric Deep Learning.
11. Recap and Q&A.

- The exam will be written.
- We will have an exam preparation test.

Acknowledgements

- Special thanks Arij Bouazizi, Julia Hornauer, Julian Wiederer, Adrian Holzbock and Youssef Dawoud for contributing to the lecture preparation.

Recap

- Compression categories.
- Parameter quantitation.
- Parameter pruning.
- Data-free approaches.

Today's Agenda and Objectives

- Few-shot learning.
- Meta learning.
- Task-based learning.
- Model Agnostic Meta Learning.
- Reptile.

Large-Scale Annotation

- If there are enough annotated samples, supervised learning has demonstrated promising results using deep neural networks.
- We already discussed that:
  - The sample annotation is a costly process.
  - The samples available for specific applications may be limited.
- A pre-trained model may not work well for the target test set because it was trained on a <u>source</u> domain dataset that differs to some extent from the <u>target</u> domain (test set).
- When moving to a new domain and it is possible to perform large-scale annotation, there is still the cost of training from the beginning.
- One approach discussed to address domain shift is the use of transfer learning.

Transfer Learning Recap (Step 1)

- Pre-training is normally happening with a <u>large-scale</u> dataset.

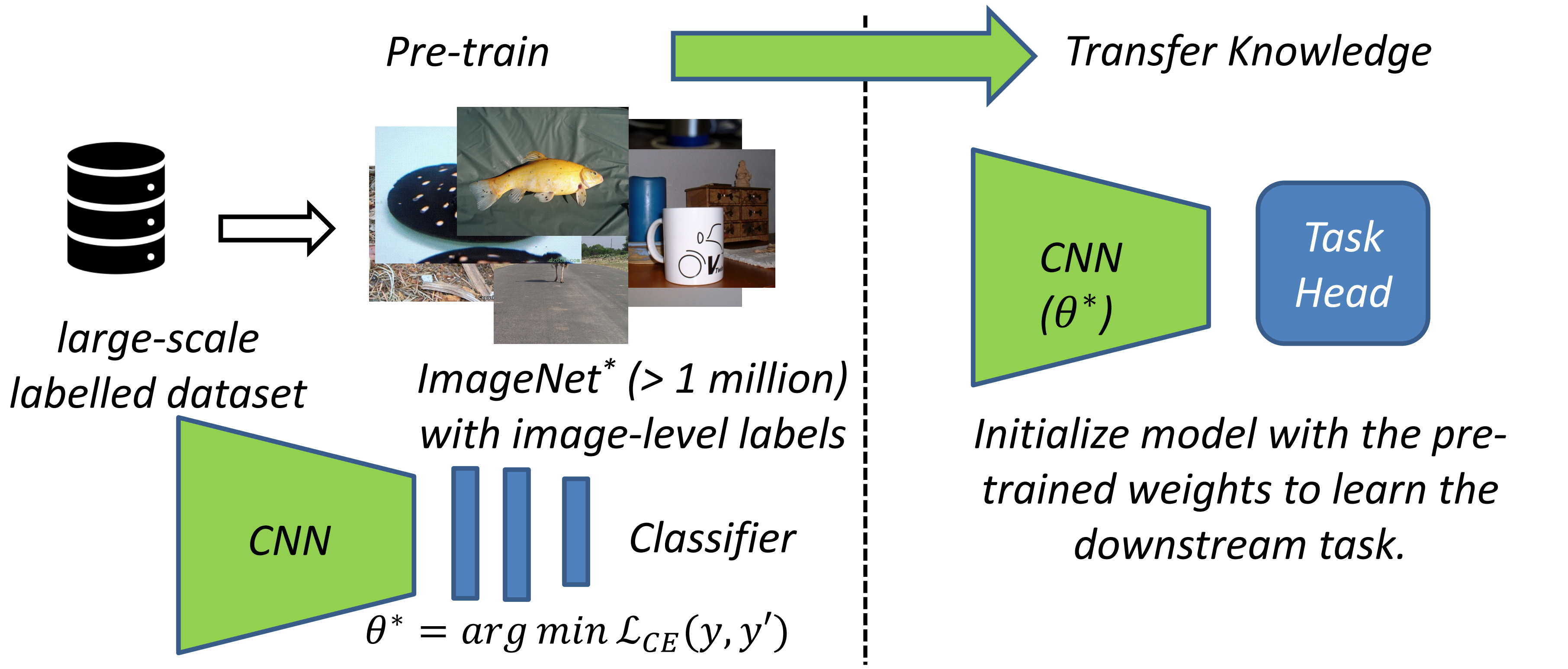


*Image Source: https://www.image-net.org

Transfer Learning Recap (Step 2)

Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)

- The deep neural network is entirely or partially, in terms of network parameters, trained on the dataset and task of interest using the optimized weights from pre-training as prior knowledge.

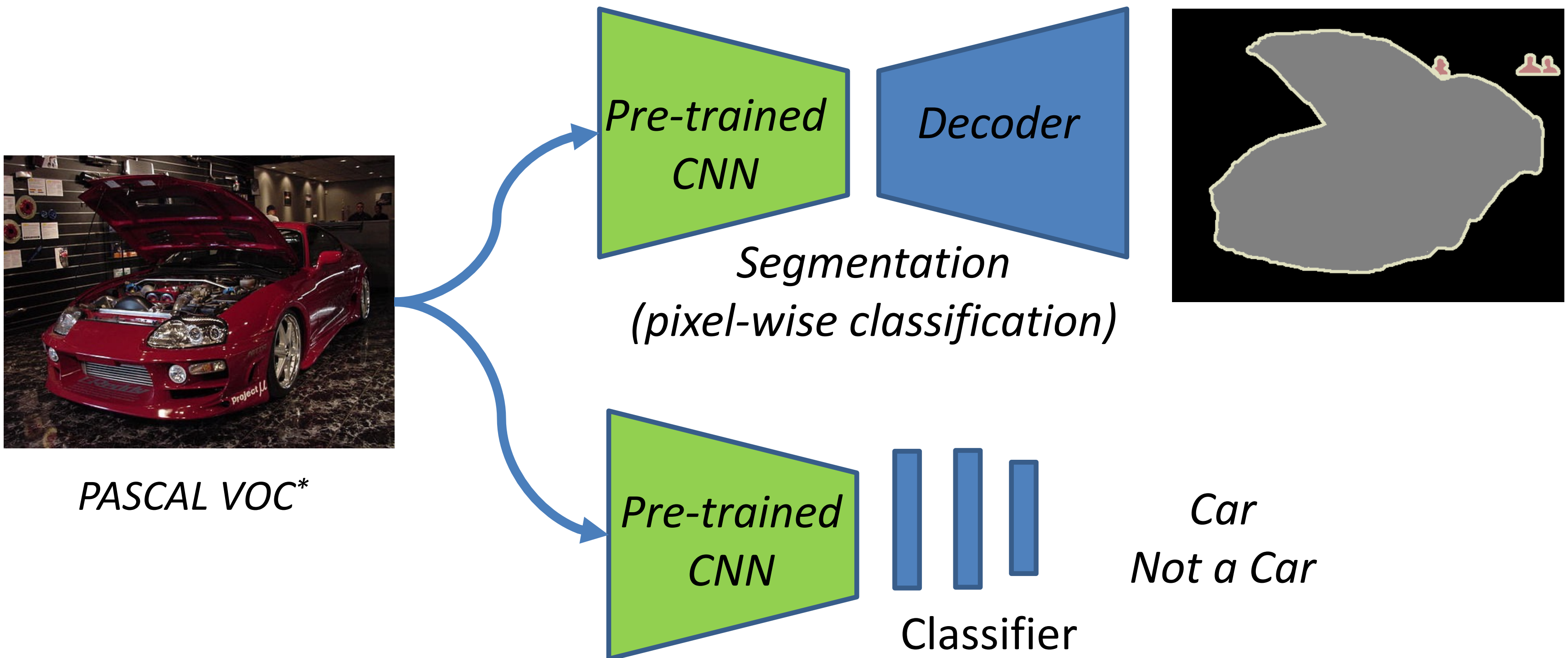


*Image source: http://host.robots.ox.ac.uk/pascal/VOC/

Transfer Learning Limitation

- Transfer learning is the approach of adapting the pre-trained model to the new task using the new dataset.
- In practice, we use the pre-trained model weights as initialisation for the new (downstream) task.
- Although transfer learning can be a useful approach to cover domain shift, it requires a considerable number of annotated samples.
- *What if the available number of annotated samples is not more than 1, 5 or 10?*

Few-shot Learning (FSL)

- Few-shot learning addresses the ability of an already trained deep neural network to <u>generalise</u> to unseen samples, using only a few annotated samples.
  - The unseen samples comes from the <u>target</u> domain, which was not encountered during pre-training.
  - The unseen samples may have <u>semantic</u> overlap with the data used for the original model training, but in a different context, e.g., real dog images and animated dog images where the class dog is the same.
  - The unseen samples may belong to different object categories, e.g., wolfs instead of dogs used during the original model training.
- Like transfer learning, it is also assumed access to a large-scale annotated during the original model training, which is commonly referred to as *Source* (D), and another unseen dataset containing the task of interest referred to as *Target* (D_T).
  - The original model training can be also called pretraining.

FSL Terminology

- The main difference is that D_T contains a small data set with only few-annotated samples for adapting the pre-trained model. We refer to it as the <u>support set</u> (S). Moreover, there is another set for testing referred to as <u>query set</u> (Q).
- Support set: It consists of a few annotated examples of the existing or new class category (or categories). It is used to train the pre-trained model.
- Query set: It consists of the samples from the target domain to be tested. They can be from a new or existing class category (or categories).
- *How many novel (new) categories are available?*
- *How many samples are available per category?*

Few-Shot Classification Example

- Also known as N-way, K-shot few-shot classification task, where N denotes the number of novel class categories and K denotes the number of annotated samples per class (normally $K \leq 10$).
  - In total of NK samples are available to trained the pre-trained model.
  - The NK data samples form the support set.
- We have the support set $S = \{S_n\}_{n=1}^{N}$, where $S_n = \{(\mathbf{x}, y^n)_i\}_{i=1}^{K}, y^n \in \{1, \dots, N\}$ and the query set $Q = \{(\mathbf{x}_j)\}_{j=1}^{L}$.
  - Note that S and Q share the same label space.

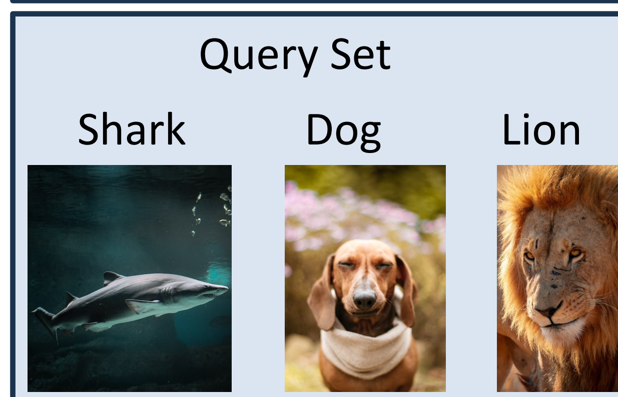


Image source: https://www.pexels.com/

Few-Shot Classification Example (Cont.)

- Note that the categories in D_T and D may overlap (partially or totally).
  - Hence, we could encounter new unseen examples of same class category, as in the source set.
- The categories in D_T and D may be mutually exclusive.
  - Hence, we encounter new unseen samples of novel categories than in source data.

Support Set

Shark Dog Lion

Query Set

Shark Dog Lion

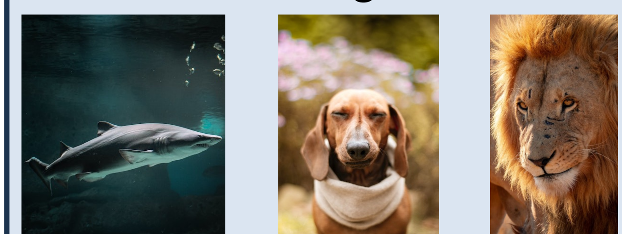


Image source: https://www.pexels.com/

Few-Shot Segmentation Example

- Like few-shot classifying, we deal with the N-way K-shot problem, but now on the pixel level.
- D_T consists of the support set $S = \{(\mathbf{x}, \mathbf{y})_i\}_{i=1}^{K}$ with $\mathbf{y} \in \mathbb{R}^{H\times W}$ is a pixel-wise segmentation map and a query set $Q = \{(\mathbf{x}_j)\}_{j=1}^{L}$.
- For example, consider the problem of learning to segment cells from certain types of images. Then the training set has different types of cells.

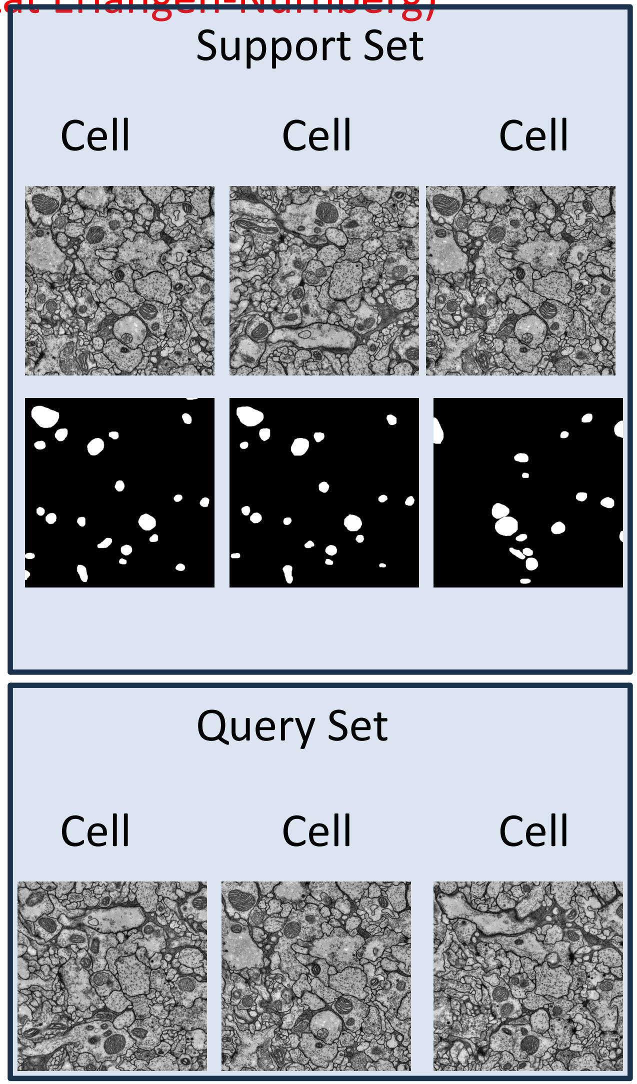


Image source: https://paperswithcode.com/dataset/sstem

Types of Approaches

- In the standard supervised learning training protocol, there is a large number of sample in the mini-batch to perform backpropagation and gradient descent.
  - Few-shot learning cannot work under this setting due to the lack of plethora of training data.
  - Using standard supervised learning can easily lead to overfitting.
- In few-shot learning, there are mainly four type of approaches:
  - Data-based approaches: One tries to increase the limited dataset, for instance, with augmentation.
  - Parameter-based approaches: The goal is to constrain the way the parameters change, for example, with meta-learning.
  - Metric-based approaches: These approaches aim to learn a distance function between the available samples. For example, a similarity function between pairs or triplets of samples.
  - Gradient-based meta-learning approaches: These approaches are based on a base-learner and a meta-learner that cooperate to learn from tasks.

Meta Learning

- Meta learning refers to learning the learning algorithm itself. It is also known as learning to learn. The main type of meta learning algorithm are:
  - Metric-based meta learning[1].
  - Model-based meta learning[2].
  - Optimization-based meta learning.
- A common learning approach for meta learning algorithms is using tasks.
  - We apply <u>task-based</u> (episodic) learning where the model is optimized using tasks τ sampled from D. This is a different approach from the mini-batch training of the standard supervised learning.
- *How do we form a task for few-shot learning?*

[1]Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." Advances in neural information processing systems 30 (2017).
[2]Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., & Lillicrap, T. (2016, June). Meta-learning with memory-augmented neural networks. In International conference on machine learning (pp. 1842-1850). PMLR.

Task Formation

- Consider N-way, K-shot classification. Recall a few-shot classification task consists of support and query sets.
- Randomly select N classes from the entire set of classes C. For example, if C = [dog, cat, bird, lion,…,] and assume we select randomly 2 classes from C i.e., N=2 , then we have the classes [dog, lion].
- Then select all samples from D that are classified as dog or lion, let this subset be $D^{dog,lion} = \{D^{dog}, D^{lion}\}$.
- Next, select at random K-shots (samples) per class from $D^{dog,lion}$.
- From $D^{dog,lion}$ we obtain the support set $S^{dog,lion}$.
- Finally, we remove all samples from $D^{dog,lion}$ to get the query set i.e., $Q^{dog,lion} = D^{dog,lion} \setminus S^{dog,lion}$.

Optimization-based Meta Learning

- We have two models the meta-learner and the base learner which are going to be differently updated.
- Both learners have the same neural network architecture, but the base learner is optimized using a single task, while the meta learner is optimized using a set of tasks, i.e., it distils the information learned from several base learners to update its parameters.
- The main objective is to find the optimal parameters θ that allow learning using a few-shots only. Recall that each task is composed of a few-shot samples.
  - Ultimately, this process would allow for efficient fine-tuning to the target task.
- Based on these scheme, there have been developed several optimization-based algorithms. The standard meta learning approaches are:
  - Model Agnostic Meta Learning (MAML).
  - Reptile.

Model Agnostic Meta Learning (MAML)

- MAML[1] is a model as well as task-agnostic approach to learn adapting to a task using only a few gradient steps.
- Pseudo algorithm:
  - Randomly initialize θ.
  - Initialize base model with meta-learner parameters θ.
  - Outer loop: Sample a batch of tasks, each task $\tau_i = \{S_i, Q_i\}$.
  - Inner loop: Train the base model f_θ using S_i with gradient descent (or any other optimizer) for a few iterations to get updated parameters θ_i'.
    - $\theta_i' \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}_{\tau_i}(f_\theta)$,
    - where α is the base model learning rate.
    - $\mathcal{L}_{\tau_i}$ is the task loss e.g., cross-entropy for classification.
  - Back to outer loop: Update the meta-learner parameters using:
    - $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\tau_i} \mathcal{L}_{\tau_i}(f_{\theta_i'})$,
    - where β is the meta-learner's learning rate.
    - $\mathcal{L}_{\tau_i}(f_{\theta_i'})$ is calculated using the <u>query</u> sets of the sampled tasks Q_i.
    - Moreover, $\mathcal{L}_{\tau_i}(f_{\theta_i'})$ is a function in the optimized base learner parameters θ_i' using S_i. Think of it as calculating the loss with respect to a validation set.

[1]Finn, C., Abbeel, P., and Levine, S., "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks", arXiv e-prints, 2017. doi:10.48550/arXiv.1703.03400.

MAML

- Updating the meta-learner is equivalent to aggregating the information learned by individual base models on different tasks.
- Computing the gradient $\nabla_\theta \sum_{\tau_i} \mathcal{L}_{\tau_i}(f_{\theta_i'})$ of the outer loop is a second-order derivative. This is an expensive operation.
- Consider the inner loop where a base learner is trained for k iterations, $k > 1$ on task τ_1 i.e., $i = 1$.We start with $\theta_{1,0} = \theta$:
  - $\theta_{1,1} = \theta_{1,0} - \alpha\nabla_\theta \mathcal{L}_{\tau_1}(f_{\theta_{1,0}})$
  - $\theta_{1,2} = \theta_{1,1} - \alpha\nabla_\theta \mathcal{L}_{\tau_1}(f_{\theta_{1,1}})$
  - …
  - $\theta_{1,k} = \theta_{1,k} - \alpha\nabla_\theta \mathcal{L}_{\tau_1}(f_{\theta_{1,k-1}})$
- This is the inner loop parameter update.

MAML (Cont.)

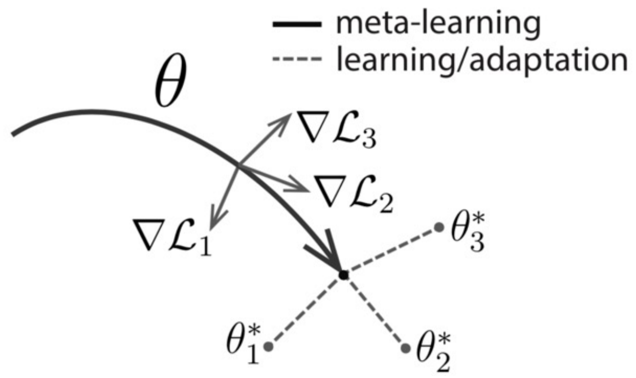
- Afterwards, we update the meta-parameters θ in the outer loop:
  - $\theta \leftarrow \theta - \beta g_{MAML}$,
  - where $g_{MAML} = \nabla_\theta \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right)$,
  - $\nabla_\theta \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right) = \nabla_{\theta_{1,k}} \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right).\left(\nabla_{\theta_{k-1}}\theta_k\right) \ldots \left(\nabla_{\theta_0}\theta_1\right).\left(\nabla_\theta \theta_0\right)$
  - This is the chain rule for finding the gradients with respect to θ i.e., the meta-learner parameters.
  - We can factorize it to:
  - $g_{MAML} = \nabla_{\theta_{1,k}} \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right).\left(\nabla_{\theta_{k-1}}\theta_k\right) \ldots \left(\nabla_{\theta_0}\theta_1\right).\left(\nabla_\theta \theta_0\right)$
  - $= \nabla_{\theta_{1,k}} \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right).\left(\prod_{l=1}^{k} \nabla_{l-1}\theta_l\right).I$
  - $= \nabla_{\theta_{1,k}} \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right).\prod_{l=1}^{k} \nabla_{l-1}\left(\theta_{l-1} - \alpha \nabla_\theta \mathcal{L}\left(f_{\theta_{1,l-1}}\right)\right)$
  - $= \nabla_{\theta_{1,k}} \mathcal{L}_{\tau_1}\left(f_{\theta_{1,k}}\right).\prod_{l=1}^{k}\left(I - \alpha \nabla_{\theta_{l-1}}\left(\nabla_\theta \mathcal{L}\left(f_{\theta_{1,l-1}}\right)\right)\right)$
  - The second order derivatives make the computations complex.

MAML Second order derivatives.

- To relax the computations, MAML has been formulated as a first-order derivatives approach.
- This variant is known as First-order MAML (FOMAML), hence, g_{MAML} is replaced with g_{FOMAML} as:
  - $g_{FOMAML} = \nabla_{\theta_{1,k}} \mathcal{L}_{\tau_1}(f_{\theta_{1,k}})$.
- The gradient calculation simply omits the chain rule.
- *What can be the performance drop by treating the method as first-order approach?*

MAML Second order derivatives.

- MAML[1] results indicate the the performance drop is not significant with first order derivatives.
- Note that "fine-tuning baseline" is the standard transfer learning procedure.



MAML

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)	$\mathbf{48.70 \pm 1.84\%}$	$\mathbf{63.11 \pm 0.92\%}$

Source: Finn, C., Abbeel, P., and Levine, S., "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks", arXiv e-prints, 2017

Reptile

- Reptile[1] is yet another optimization-based meta learning algorithm quite like MAML.
- It mainly differs only in the meta-update rule because it does not require second-order derivatives; hence, it is computationally more <u>efficient</u>.
- Reptile a <u>first</u> order approach.
- It depends on learning from tasks, while the idea of base- and meta-learner is like with MAML.

[1]Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

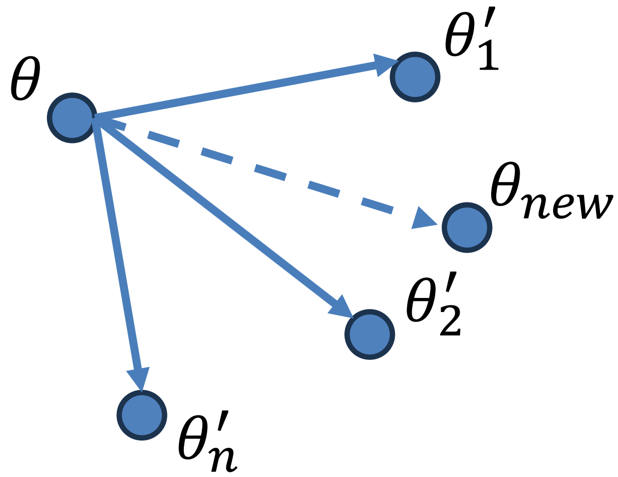
Reptile Algorithm

- The algorithm works as follows:
  - Randomly initialize θ.
  - Initialize base model with meta-learner parameters θ.
  - Outer Loop: Sample a batch of tasks, each task $\tau_i = \{S_i\}$. Note Reptile does not require a query set to update meta-parameters.
  - Inner Loop: Train base model f_θ using the support sets S_i with gradient descent (or any other optimizer) for a few iterations to get updated parameters θ_i'.
  - Back Outer Loop: Update meta-parameters using the following update rule:
  - $\theta \leftarrow \theta + \beta \frac{1}{n} \sum_{i=1}^{n} (\theta_i' - \theta)$,
  - where β is the meta-learner learning rate, n denotes the number of sampled tasks.

[1]Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Reptile Outer Loop

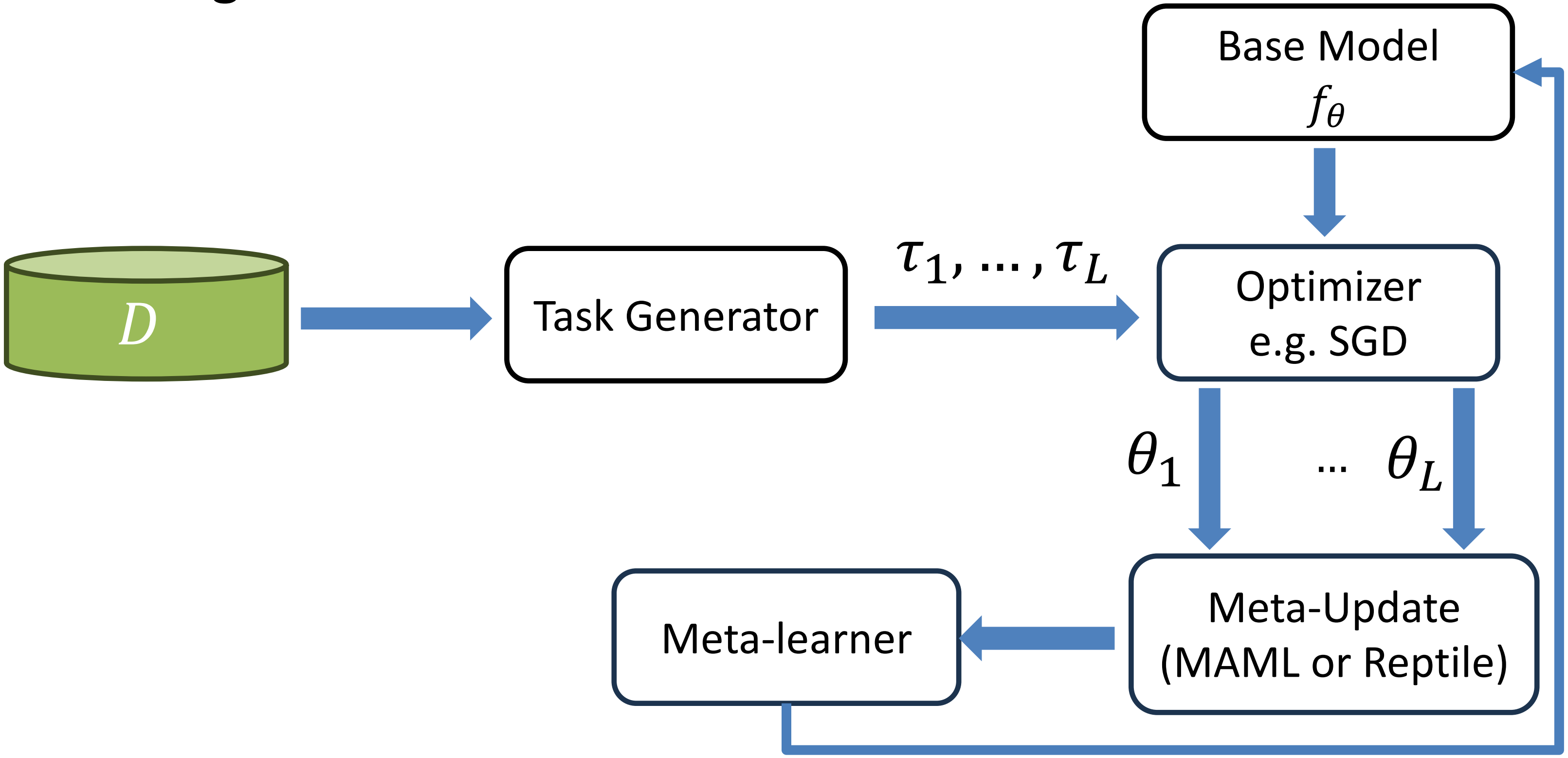
- The meta-parameters are updated using the following update rule: $\theta \leftarrow \theta + \beta \frac{1}{n} \sum_{i=1}^{n} (\theta_i' - \theta)$.
- This can be though as averaging the gradient of different tasks.
- It is related to SGD but is more like averaging gradients.



[1]Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Optimisation-based Approaches

- Both Reptile and MAML share the same idea for few-shot learning.



Initialize base model with Meta-learner parameters

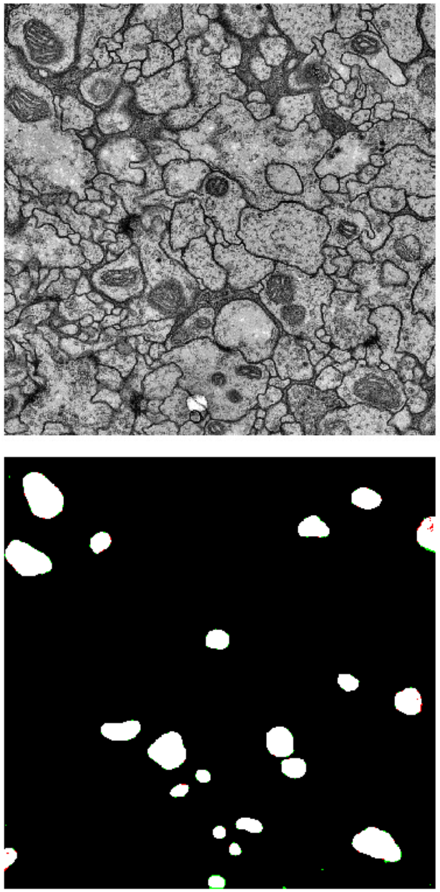
MAML vs Reptile

- Both algorithms aim to find a set of parameters that would allow the meta-learner to efficiently adapt to a new task from the target samples.
- Both algorithms are task- and model-agnostic.
- Reptile is by default a first-order approach and thus lighter.
- <u>Importantly</u> Reptile does not require a query set to update the meta-parameters, unlike MAML.
- Both algorithms are sensitive to hyperparameters and need to be tuned carefully.

[1]Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Few-shot Learning Use Case

- Consider the problem of few-shot microscopy image cell segmentation (FS-MICS).
  - Imagine that experts must annotate hundreds/thousands of microscopy data (pixel level) to train neural networks using standard supervised learning.
  - For each new microscopy dataset, the process needs to start from the beginning.
- Few-shot learning for binary cell segmentation is a practical solution to reduce the annotation time and required resources.



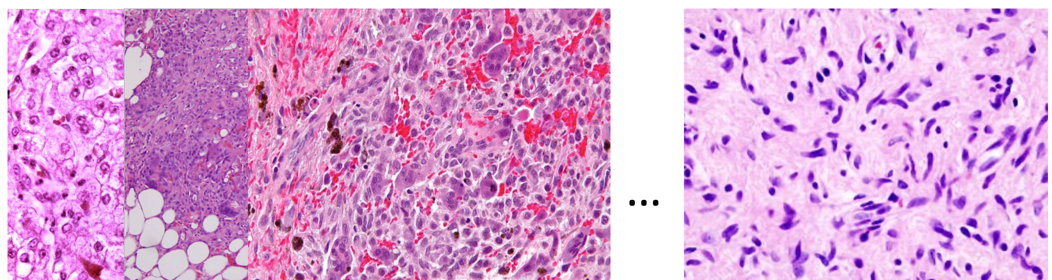
Cell Segmentation

Dawoud, Y., Hornauer, J., Carneiro, G., & Belagiannis, V. (2021). Few-shot microscopy image cell segmentation. In Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track: European Conference, ECML PKDD 2020.

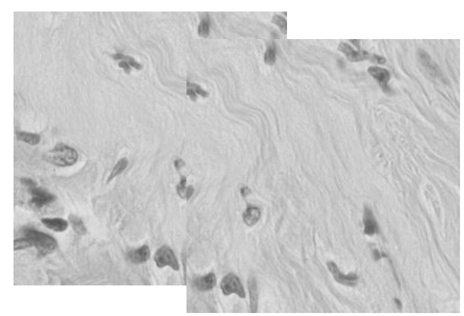
FS-MICS Problem

Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)

- Few-shot problem formulation:
  – Consider a several source of microscopy datasets with available annotation for for binary cell segmentation.
  – Also consider the target domain with a few annotate images.



Access to multiple different microscopy databases (sources).

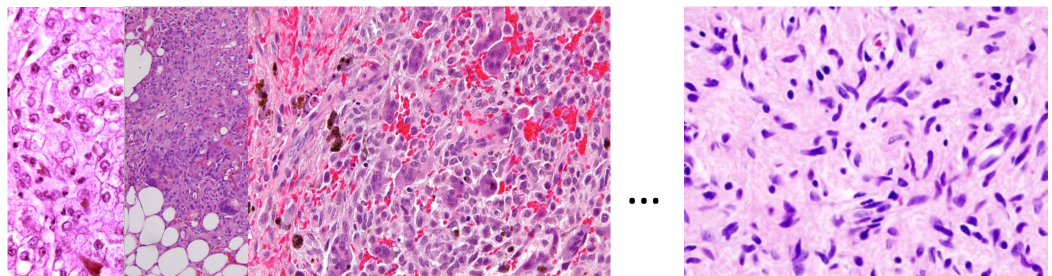


Few annotated samples from the target database (no overlap with sources)

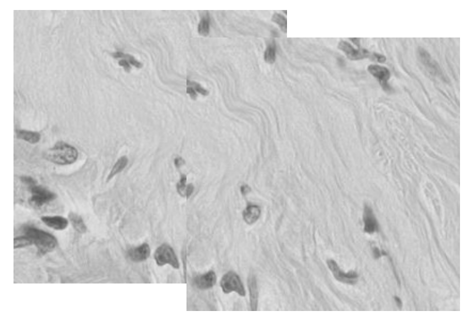
FS-MICS Goal

Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)

- Objective:
  - Learn a generic and easily adaptable model using first-order gradient-based few-shot learning.
- We will rely on the reptile algorithm for that reason.



Access to multiple different microscopy databases (sources).

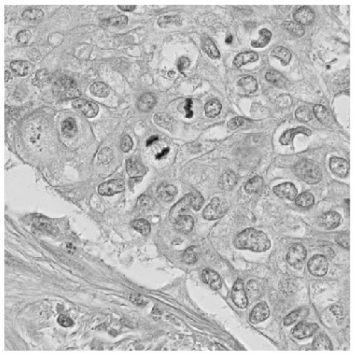


Few annotated samples from the target database (no overlap with sources)

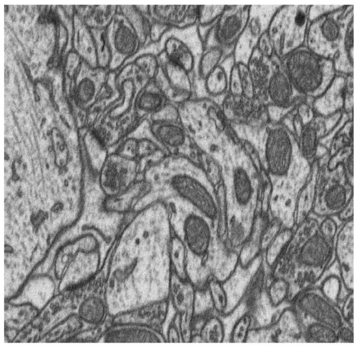
Friedrich-Alexander-Universität
Technische Fakultät

FS-MICS Training Set

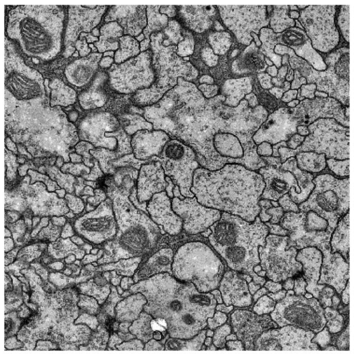
- We utilize 5 microscopy datasets. These are:



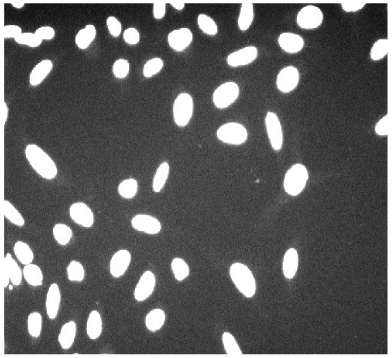
TNBC



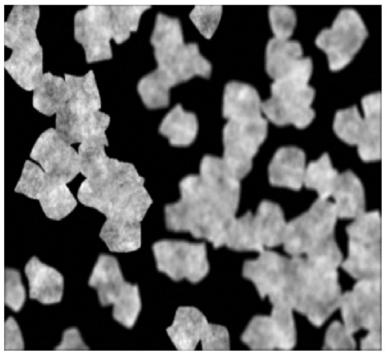
EM



ssTEM



B39



B5

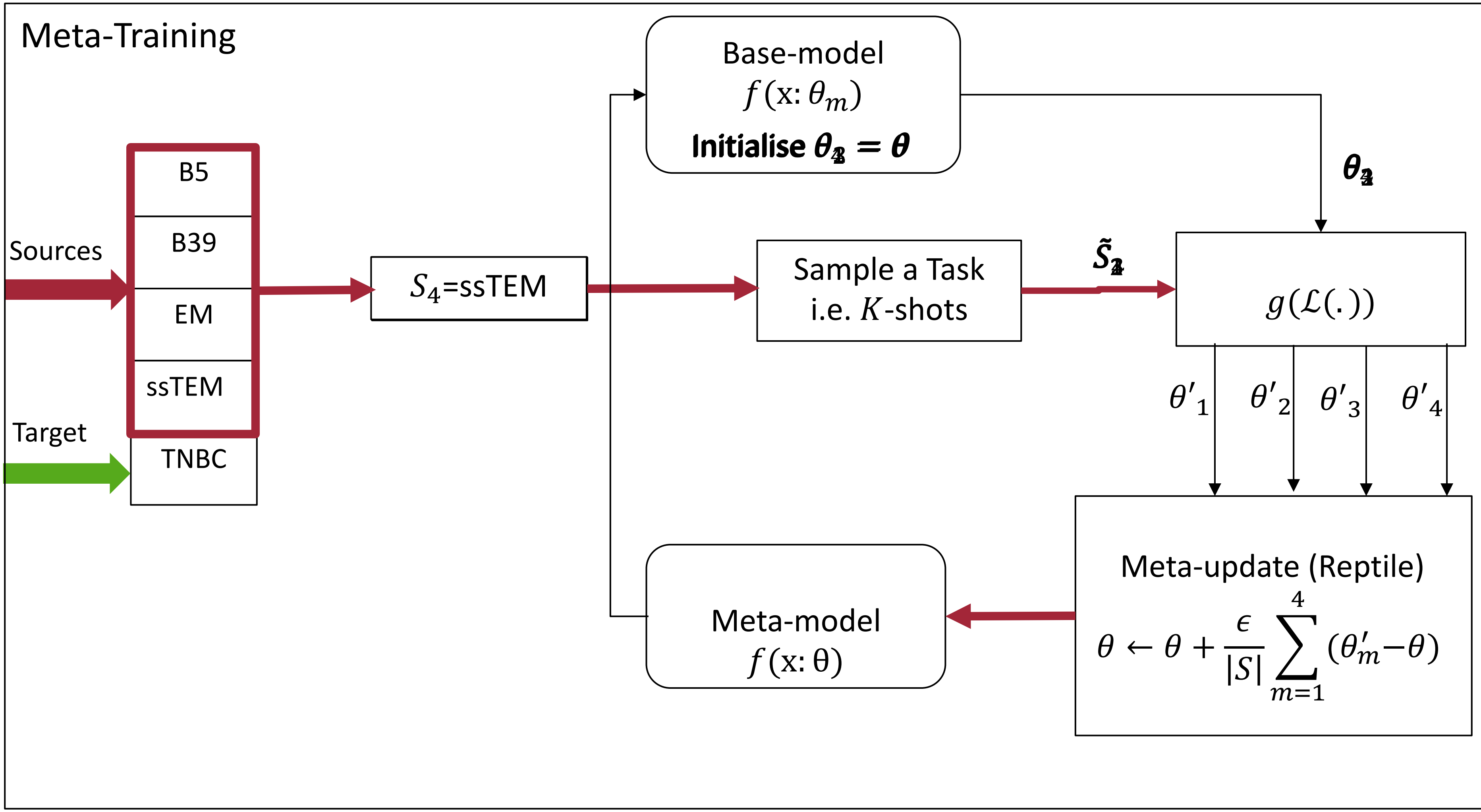
Data set	B5	B39	ssTEM	EM	TNBC
Cell Type	Synthetic stain	nuclei	mitochondria	mitochondria	nuclei
Resolution	696 x 520	626 x 520	1024 x 1024	768 x 1024	512 x 512
# of Samples	1200	200	20	165	50

FS-MICS Training

Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)

- We use 4 of them as the source data for pre-training on cell segmentation using Reptile algorithm.
- Given that we have 4 datasets in the source, hence, we can sample in total 4 tasks, where we set each task to contain 5-shots.
- Formally, we have:
  - $D = \{D_1, D_2, \dots, D_{|D|}\}$ a collection of microscopy data sets.
  - $D_m = \{(\mathrm{x}, \mathrm{y})_k\}_{k=1}^{|D_m|}$. → Single dataset
    - $\mathrm{x}: \Omega \to \mathbb{R}$. → Input Image
    - $\mathrm{y}: \Omega \to \{0,1\}$. → binary segmentation mask
    - A task is a number of samples from D_m.
  - Each data set corresponds to a different task.
    - D is the source data.
    - Target support set $S = \{(\mathrm{x}, \mathrm{y})_k\}_{k=1}^{|S|}$. → K-shots from D_T for fine-tuning.

Training Algorithm (Reptile)



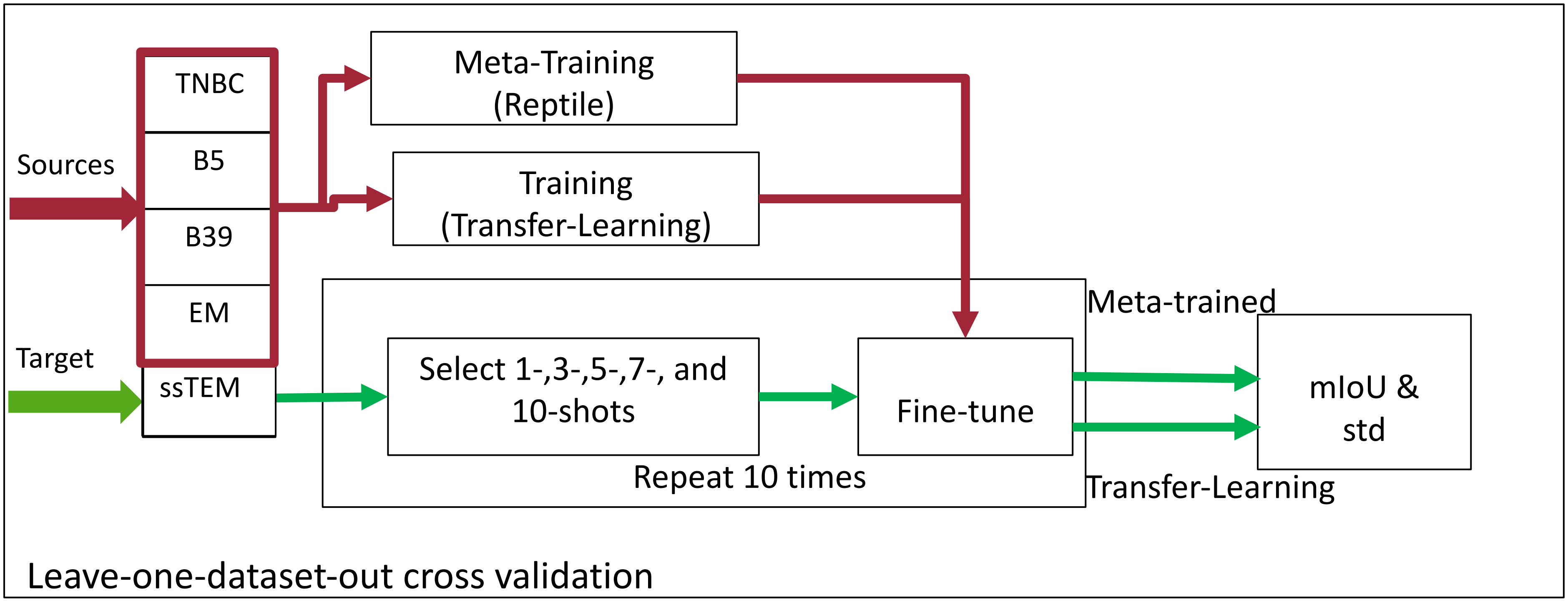
Friedrich-Alexander-Universität
Technische Fakultät

Fine-tuning Algorithm

- After meta-training the model using Reptile algorithm, we fine-tune it on the target data using a randomly support set.
- We analyse the performance under different K-shots i.e., K = 1-, 3-, 5-, 7- and 10-shots.
- Finally, we test it on other samples from the target. Note that we exclude the support set from the target data.
- We repeat the process of fine-tuning and testing 10 times and report the average results.
- In segmentation, there are different evaluation metrics to quantify the performance of the segmentation model.
- Mean Intersection over union (mIoU) is mostly used. It measures the overlap between a prediction $\mathbf{y}'$ and the ground truth segmentation $\mathbf{y}$.

$$\text{mIoU} = \frac{\mathbf{y}' \cap \mathbf{y}}{\mathbf{y} \cup \mathbf{y}}$$

Fine-tuning Algorithm (Cont.)



FS-MICS Results

TNBC Database

ssTEM Database

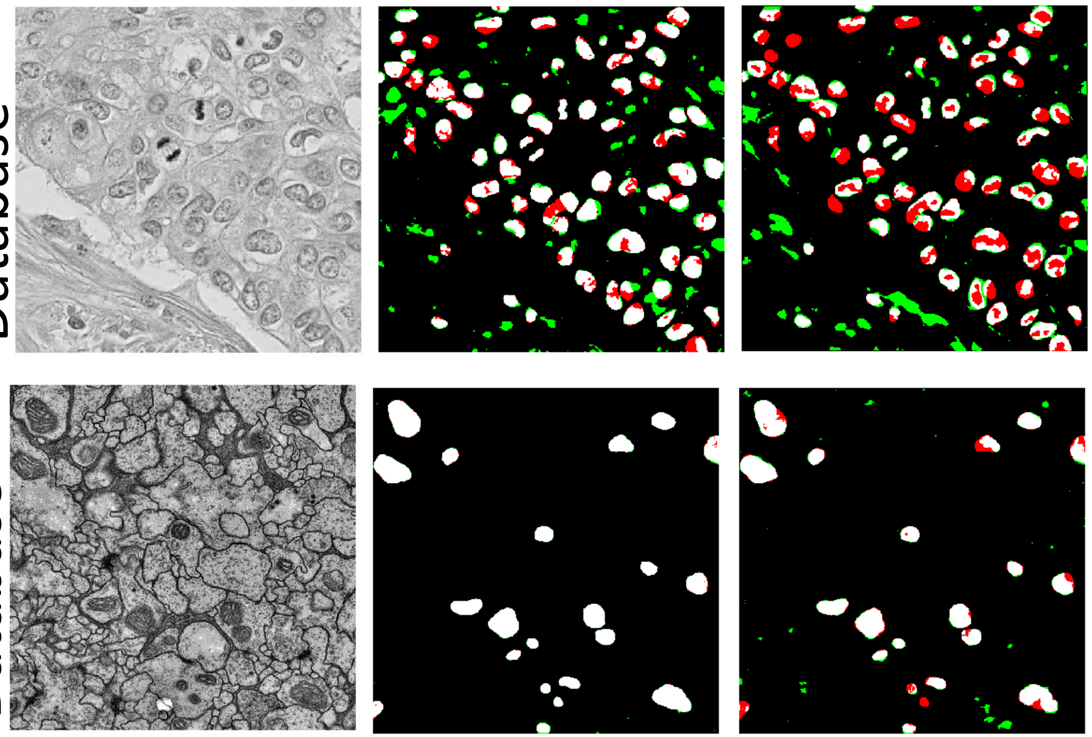


Image.

Meta-Learning

Transfer-Learning

Study Material

- *Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999.*
- *Finn, C., Abbeel, P., and Levine, S., "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks", arXiv e-prints, 2017.*
- *Dawoud, Y., Hornauer, J., Carneiro, G., & Belagiannis, V. (2021). Few-shot microscopy image cell segmentation. In Machine Learning and Knowledge Discovery in Databases. Applied Data Science and Demo Track: European Conference, ECML PKDD 2020.*
- *Meta-Learning: Learning to Learn fast https://lilianweng.github.io/posts/2018-11-30-meta-learning/*
- *Meta-Learning in Neural Networks: A survey https://arxiv.org/pdf/2004.05439.pdf*
- *Learning from Few Examples: A Summary of Approaches To Few-Shot Learning, https://arxiv.org/pdf/2203.04291.pdf*

Next Lecture

Uncertainty Estimation