DEEPFIELD Winter School: Introduction to Few-Shot Learning

Mateusz Ochal

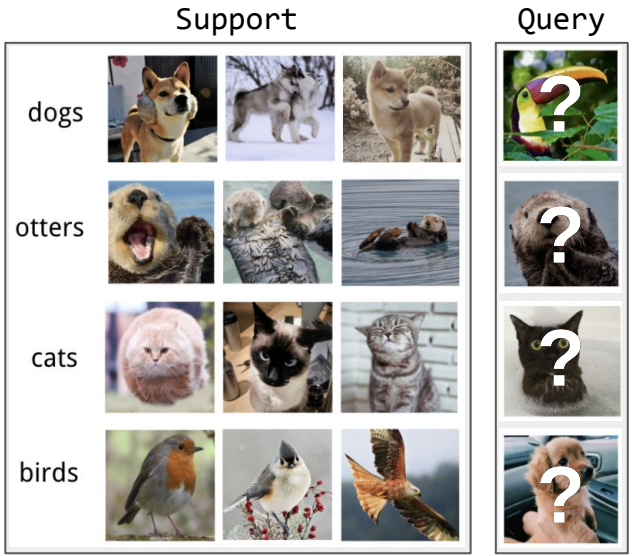
Funded by the European Commission under the H2020 EU Framework (H2020-WIDESPREAD-2018-03, 857339)

Today’s Agenda

- [09:30-10:30] Introduction to Few-Shot Learning
  - What is it?
  - Why use it?
  - How it works?
  - Approaches
  - Open Challenges
- [10:30-11:30] Applications to Underwater Imagery
  - Application to Optical Images
  - Application to Sonar Images
- [13:30-15:30] Hands-on Few-Shot Learning

What is Few-Shot Learning (FSL)?

- FSL is a task
  - Small training set (support set)
  - Small evaluation set (query set)
  - A task is denoted as “K-shot N-way”, i.e. the support set contains:
    - K samples per class (shot) for each one of N classes (way)
- Goal:
  - To leverage the support set and minimize some loss over the query set

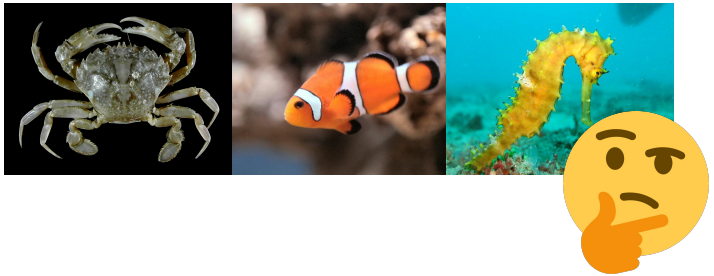


Why use it? The general problem...

- Deep Learning models are highly successful in data-rich applications
- However, their success is hindered when data sets are small
- Problem:

  Neural Networks optimize millions of parameters and easily overfit on small training datasets

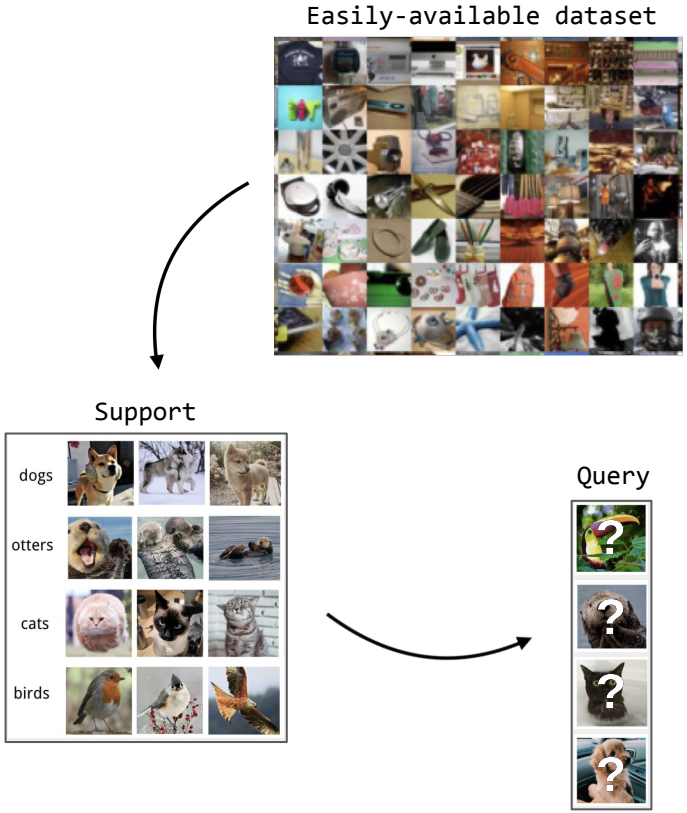
- Solution?
  - Few-Shot Learning!



How it works? The general approach...

Similar to Transfer Learning

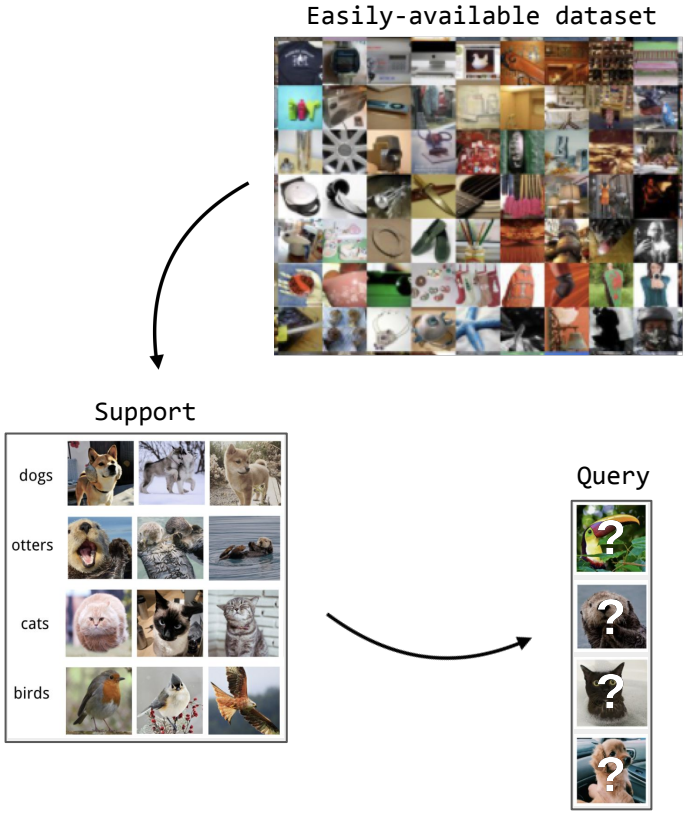
1. Pre-train a model on a large, easily-available dataset
   - e.g. ImageNet, MS-COCO, or some other data-rich dataset
2. Train / fine-tune the model on the support set
3. Evaluate the model on the query set



Which method is the best?

This is an open question in research...

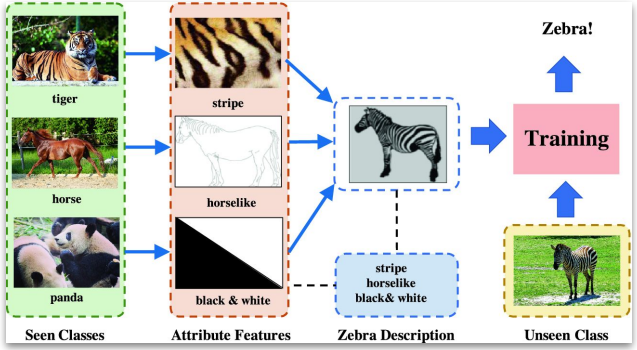
- How to pre-train the model?
  - e.g. Meta-Learning vs. Metric-Learning
- How to learn from the support set?
  - e.g. k-NN vs. SGD vs. meta-optimiser
- How to evaluate on the query set?
  - e.g. classical vs. transductive evaluation

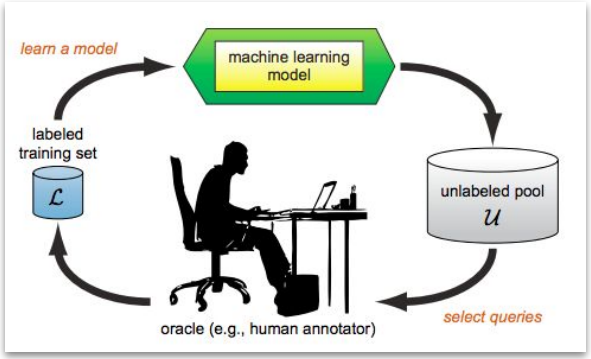


Related Work

Related Work

- Zero-Shot Learning
  - Concerned with inferring novel classes from text descriptions rather than actual images
- Semi-Supervised Learning
  - Concerned with learning from datasets that contain both labelled and unlabelled instances
- Active Learning
  - Studies how to obtain the most significant performance gains by labelling as fewest samples as possible
- Person/Vehicle ReID
  - Concerned with retrieving a person or vehicle of interest across multiple non-overlapping cameras

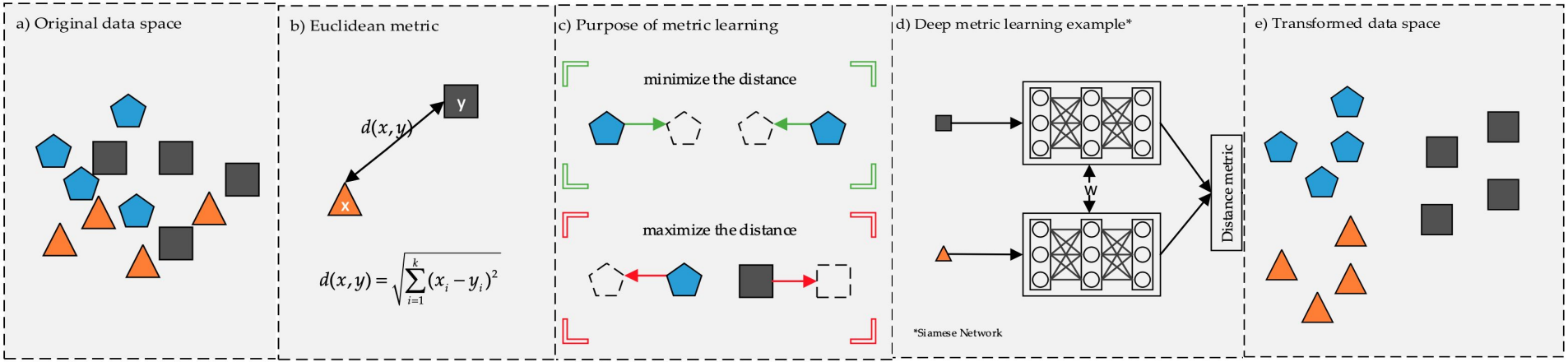




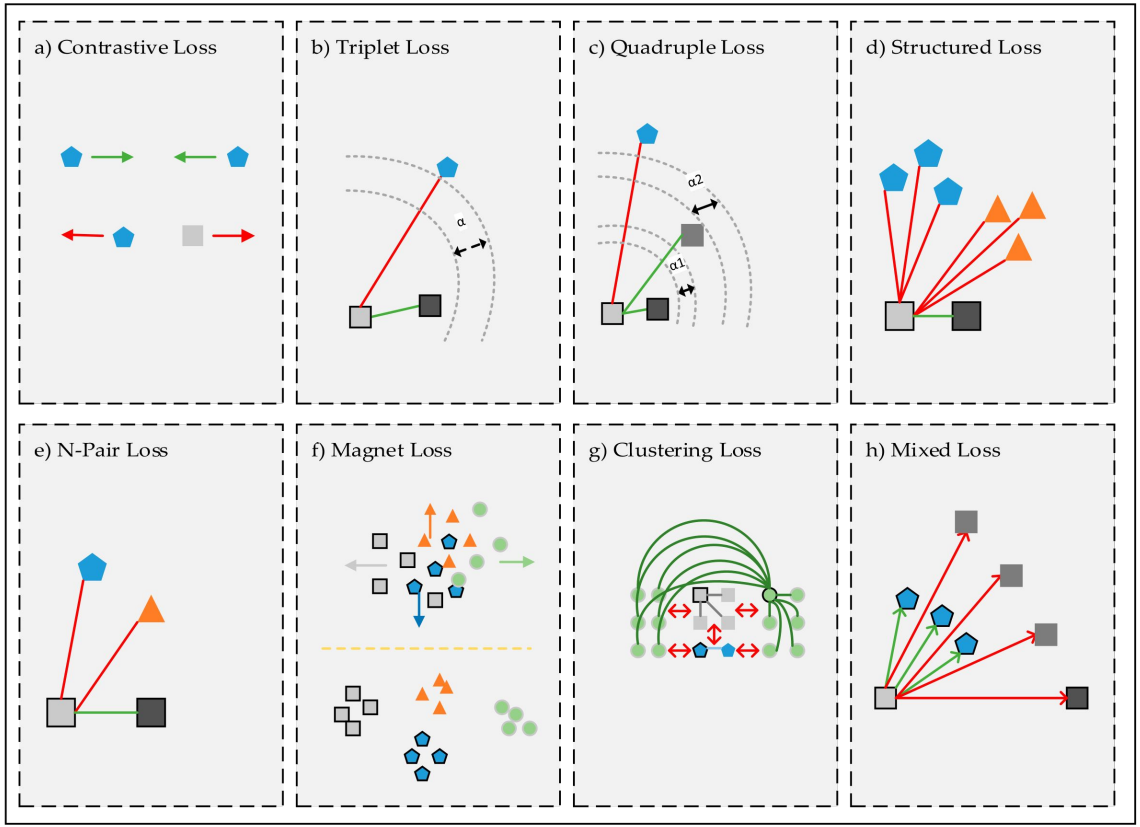
Metric-Learning

Metric-Learning: Overview

- Aims to learn a new metric:
  - To reduce the distances between samples from the same classes
  - To increase the distances between samples of different classes
- Requires a large dataset to learn a good metric
- Once learned a good metric, can be applied to Few-Shot Learning

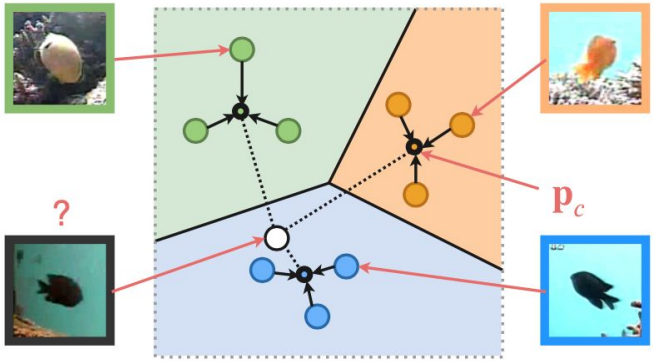


Metric-Learning: Examples



Metric-Learning: Prototypical Networks

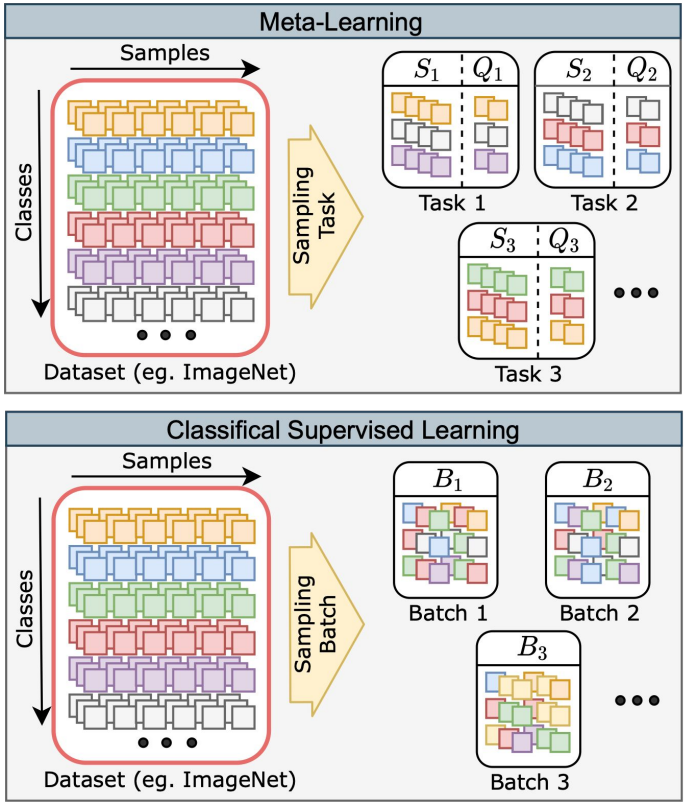
- Snell at el, 2017
- Intuition:
  - Learns a representation by minimising the distances between samples and their class centroids (prototypes)
- A class prototype is:
  - The mean of all features belonging to a single class



Meta-Learning

Meta-Learning: Overview

- Training models to “learn how to learn” on any task
  - Instead of training models to “learn” a specific task
- Using FSL task as example:
  - Meta-train the model on a distribution of FSL tasks, such that, the model generalises to unseen tasks during evaluation
- Meta-Learning is not limited to FSL, and could (in theory) applied to any problem



Meta-Learning vs Machine-Learning

- Consider conventional supervised machine-learning:
  - Given:
    - Dataset: $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$
    - Predictive function: $\hat{y} = f_\theta(x)$
    - Predefined knowledge 'how to learn': ω
      - This is often pre-selected and fixed
      - E.g.
        - optimizer: SGD
        - loss: CE
        - # of iter: 1000, etc…
  - Machine-Learning Goal:
    - Find optimal weights, θ^*: $\min_\theta \mathcal{L}(\mathcal{D}; \theta, \omega)$
  - Meta-Learning Goal:
    - Find optimal meta-knowledge, ω^*, across a set of tasks, $\mathcal{T} = \{\mathcal{D}, \mathcal{L}\}$: $\min_\omega \mathop{\mathbb{E}}_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}(\mathcal{T}; \omega)$

Meta-Learning: Inner vs Outer Optimization

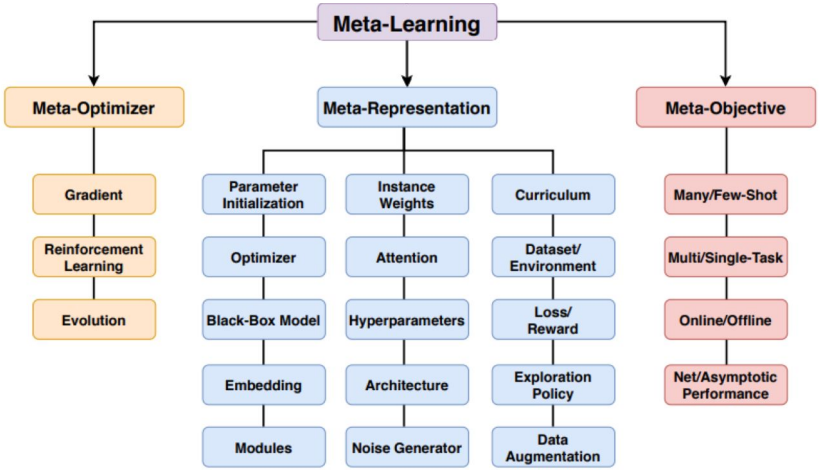
- Bilevel optimization view of meta-training:

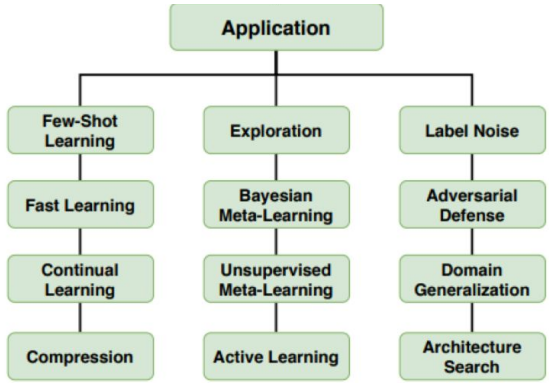
$$\omega^* = \arg\min_{\omega} \sum_{i=1}^{M} \mathcal{L}^{meta}(\theta^{*\,(i)}(\omega), \omega, \mathcal{D}_{source}^{val\,(i)})$$
$$\text{s.t. } \theta^{*(i)}(\omega) = \arg\min_{\theta} \mathcal{L}^{task}(\theta, \omega, \mathcal{D}_{source}^{train\,(i)})$$

  - where:
    - Inner objective: $\mathcal{L}^{task}$
    - Outer objective: $\mathcal{L}^{meta}$
    - Dataset of tasks: $\{(\mathcal{T}_{source})^{(i)}\}_{i=1}^{M} = \{(\mathcal{D}_{source}^{train}, \mathcal{D}_{source}^{val})^{(i)}\}_{i=1}^{M}$
- Meta-Evaluation on a *target* task:
  - Extract information from the training dataset: $\mathcal{D}_{target}^{train}$
  - Evaluation of performance on the test dataset: $\mathcal{D}_{target}^{val}$
- In FSL, $\mathcal{D}_{*}^{train}$ and $\mathcal{D}_{*}^{val}$ are the support and query sets

Meta-Learning: Taxonomy

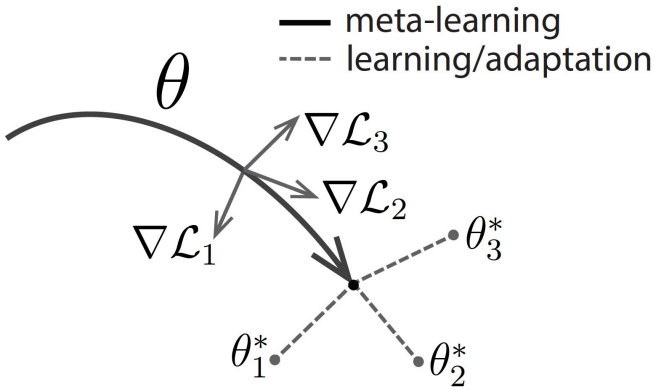
- Meta-Representation (What?):
  - Learn which meta-knowledge helps a model generalise across tasks
  - E.g. find which model weights should be learned or fixed during fine-tuning
- Meta-Optimizer (How?):
  - Aim to learn to choose outer optimization strategy for training model weights
- Meta-Objective (Why?):
  - Aim to learn to choose an appropriate objective (e.g. loss function)
  - Consider the design of task-distribution to learn from
  - And, consider the data flow between inner-loop and outer-loop optimization





Meta-Learning: MAML

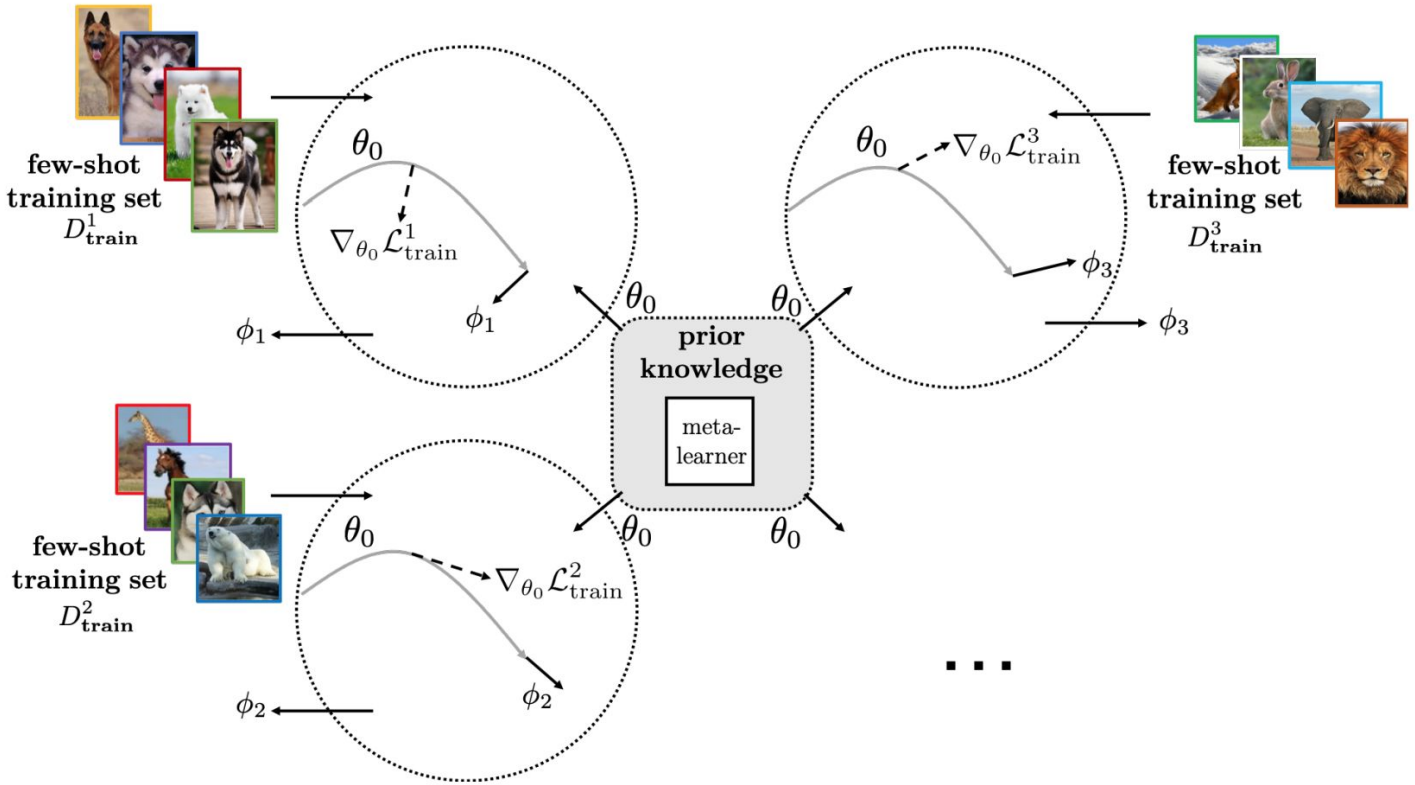
- Model
- Finn et al, 2017
- Intuition:
  - Meta-Learns a good initial condition of the model weights s.t. they can be easily fine-tuned on the support set of a target task
- Diagram parameters:
  - θ is the meta-learned initial starting position of for weights
  - $\nabla\mathcal{L}$ are the task gradients
  - θ^* are the task-optimal parameters
  - Expanded diagram on the next slide



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**

Meta-Learning: MAML



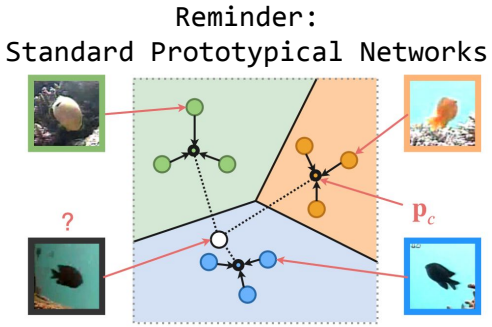
Generative Modelling, Transductive Evaluation

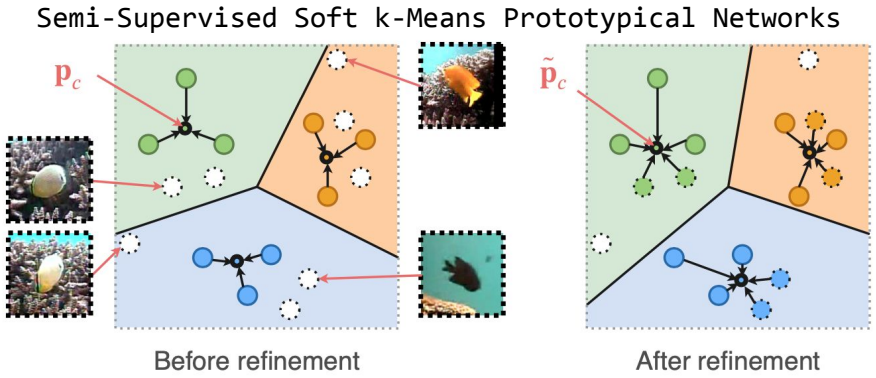
Generative Modelling

- Intuition:
  - Estimate the distribution over samples of a target task and generate more samples
- Can be achieved with:
  - Generative Adversarial Networks (GANs)
  - Strong data-augmentation
  - Image reconstruction methods

Transductive and Semi-Supervised Learning

- Semi-supervised learning:
  - Use an an labelled set in addition to the support set to refine task adaptation (inner-loop)
  - Example on the right
- Transductive Evaluation:
  - Use the query set as an additional unlabelled set during task adaptation (inner-loop)



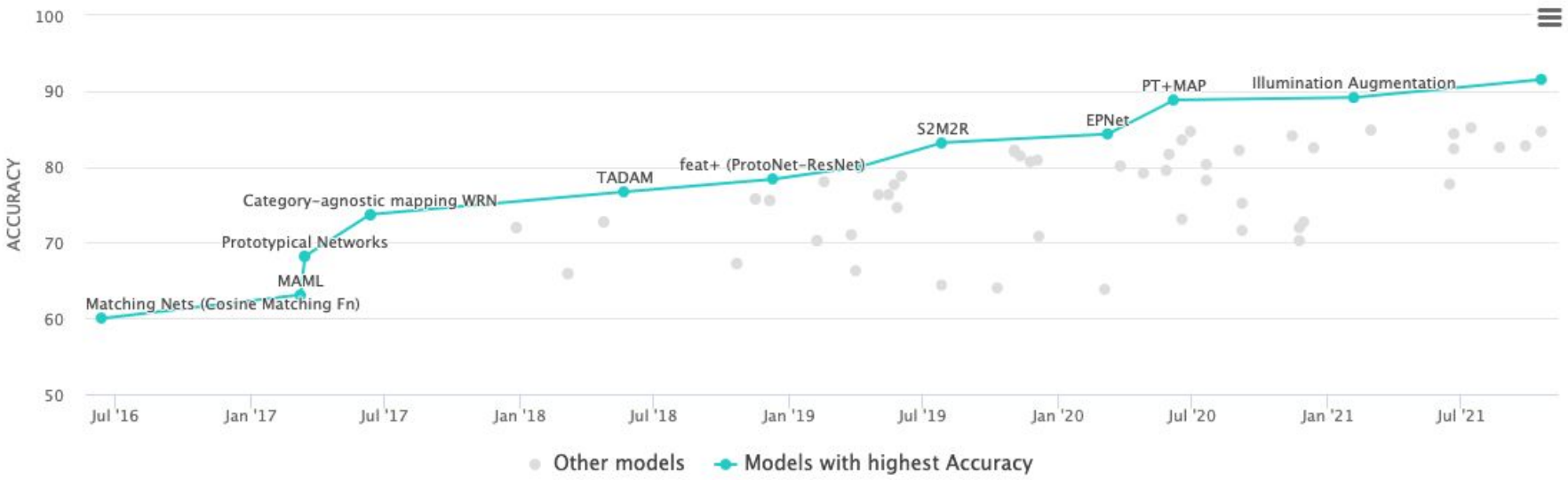


Questions?

5-min break

How good are Few-Shot models?

Leaderboard - 5-shot 5-way Mini-ImageNet

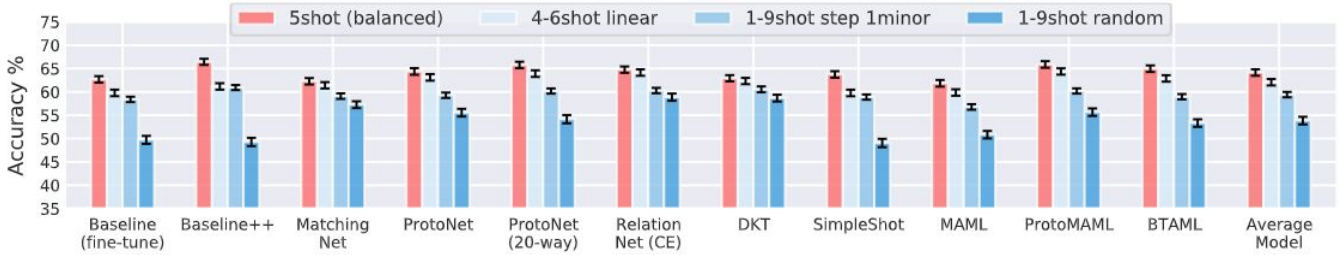


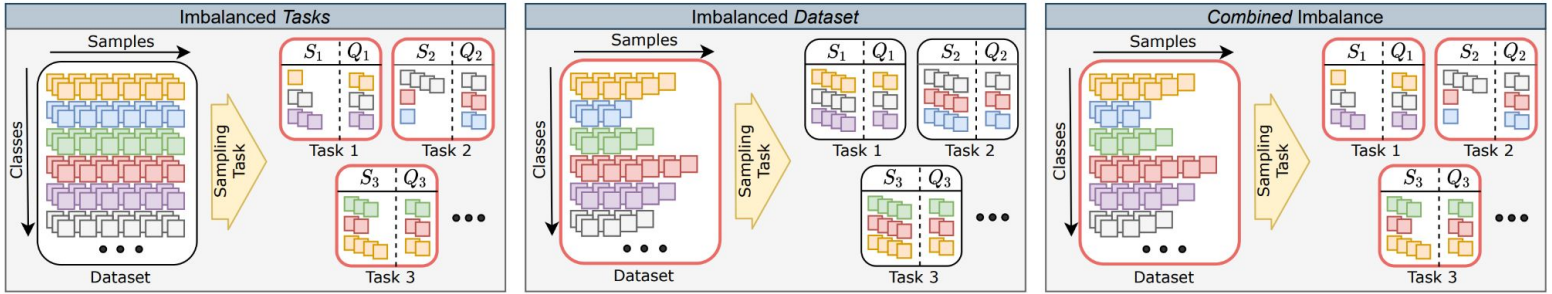
https://paperswithcode.com/sota/few-shot-image-classification-on-mini-3

Expanding FSL

Few-Shot Learning with Class Imbalance

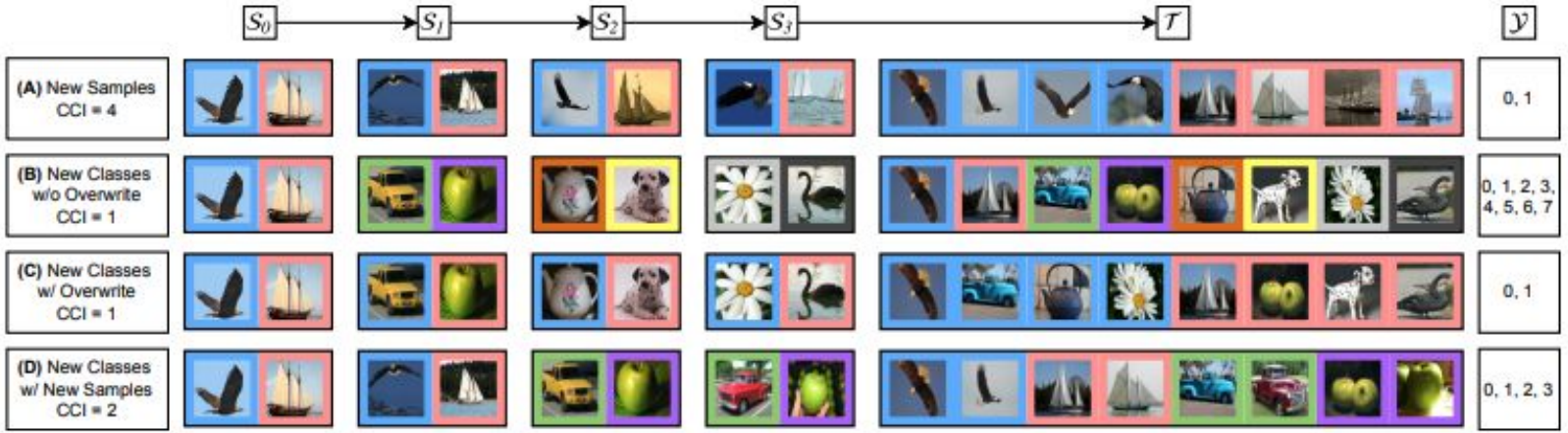
- Ochal et al., 2021
- Key takeaways:
  - Standard FSL task formulation assumes an idealised setting and ignores the fact that most real-world datasets are imbalanced
  - This work shows that some FSL methods perform more strongly in the imbalanced settings
  - Some rebalancing techniques from the supervised learning can apply in FSL





Continual Few-Shot Learning

- Antoniou et al., 2020
- Key takeaways:
  - The standard FSL setup ignores that some support samples could arrive dynamically overtime.
  - Many meta-learning algorithms still struggle from catastrophic forgetting



Beyond Few-Shot Classification

Few-Shot is not limited to classification…

- Few-Shot Detection
- Few-Shot Segmentation
- Few-Shot Domain Adaptation
- Few-Shot Reinforcement Learning

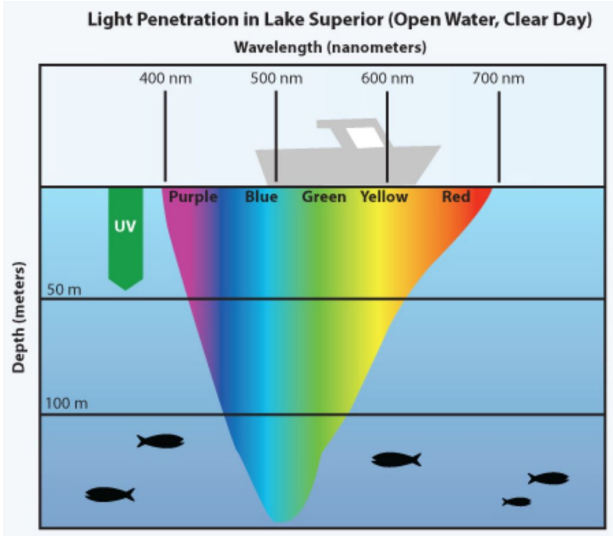
Resources:

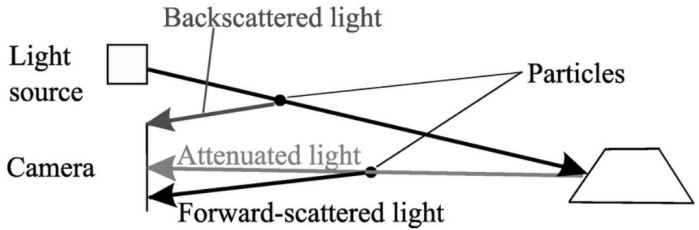
- Meta-Learning in Neural Networks: A Survey
  - https://arxiv.org/pdf/2004.05439.pdf
- Generalizing from a Few Examples: A Survey on Few-Shot Learning
  - https://arxiv.org/pdf/1904.05046.pdf
- Deep Metric Learning: A Survey
  - https://www.mdpi.com/2073-8994/11/9/1066

Application of Few-Shot Learning to Underwater Imagery

Optical Images: Challenges

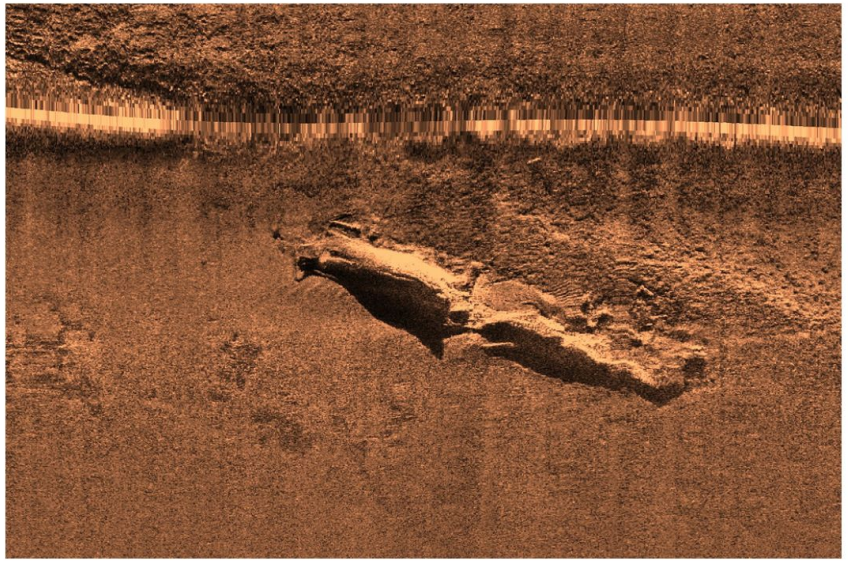
- Underwater vision using optical cameras is particularly challenging
  - Rapid absorption
  - Scattering of light by water molecules
  - Images are blurred and discoloured
- Visibility can depend on:
  - Salinity of water
  - Time of day and year
  - Conditions on the water surface





Sonar Images: Challenges

- Low-resolution
  - varies with depth
- No colour information
- Blind spots behind acoustic ‘shadows’

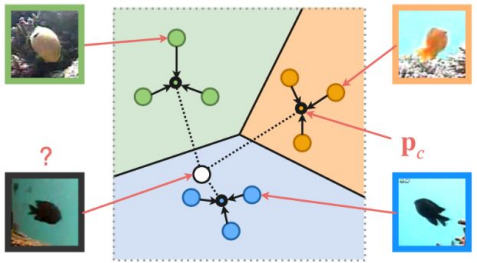


A Case Study

Methods - Supervised FSL

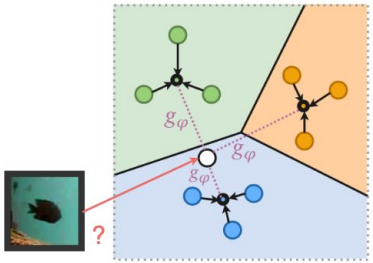
Prototypical Network

- (Snell et al. 2017)
- also called ProtoNet, or PN
- Map images into a feature space
- The mean of a class's support samples is called a prototype
- Target samples are classified based on distance to nearest prototype



Relation Networks

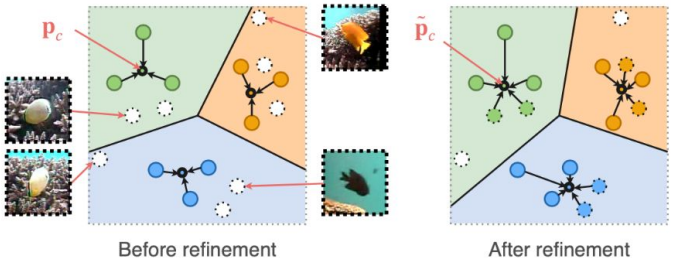
- (Sung et al. 2017)
- also called Relation Net, or RPN
- build on top of ordinary PN
- learn a relation module that scores similarily between target samples with prototypes



Methods - Semi-Supervised FSL

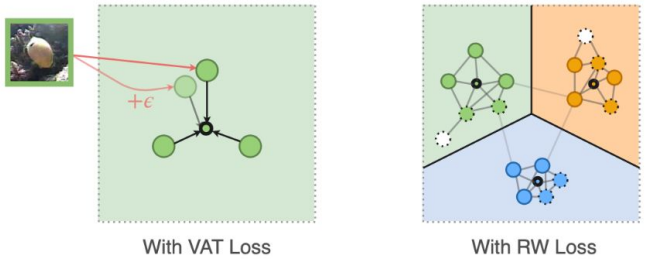
Soft K-Mean PN

- (Ren et al. 2018)
- with masking or clustering
- build on top of ordinary PN to add unlabeled support samples
- use a soft k-means refinement step for prototypes



Consistent ProtoNet

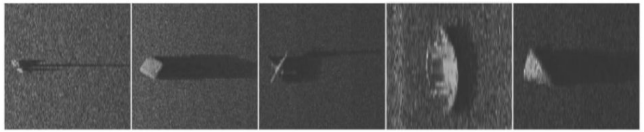
- (Ayyad et al. 2019)
- also called CPN
- build on top of Soft K-Mean PN, with added global and local consistency losses
- Virtual Adveserial Training (VAT)
- Random Walk (RW)



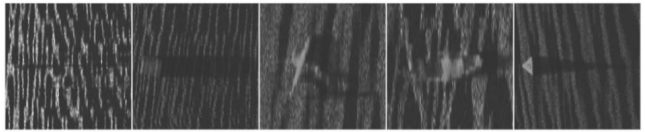
Datasets

(a) Fish Recognition(Fisher et al. 2016), showing five different fish species.

(b) Pipeline Features include an anode, grout bag, shell, fish and sea urchin.



(c) SSS (flat), showing an anchor, cube, plane, boat, and pyramid.



(d) SSS (rippled), showing an anchor, cube, plane, boat, and pyramid.

(e) Mini-ImageNet(Vinyals et al. 2016), showing a wolf, dog, lipstick, ant, and some fish.

Experiment Setup

- Three Scenarios:
  a. Meta-training on Mini-ImageNet only
  b. Meta-training on an underwater dataset only
  c. Meta-training on both:
     i. meta-training on Mini-ImageNet
     ii. then again on underwater dataset
- 5-shot 5-way tasks
  - 25 support samples in total
  - 5 targets/queries per class
- Network Architectures
  - All FSL models used a vanilla convolutional neural network consisting of 4 convolutional blocks (same as ConvNet).
- Fine-tune Baselines
  - ConvNet (from random weights)
  - ResNets (from random weights)
  - Initialized ResNet (pretrained on ImageNet)

Results: Optical Datasets

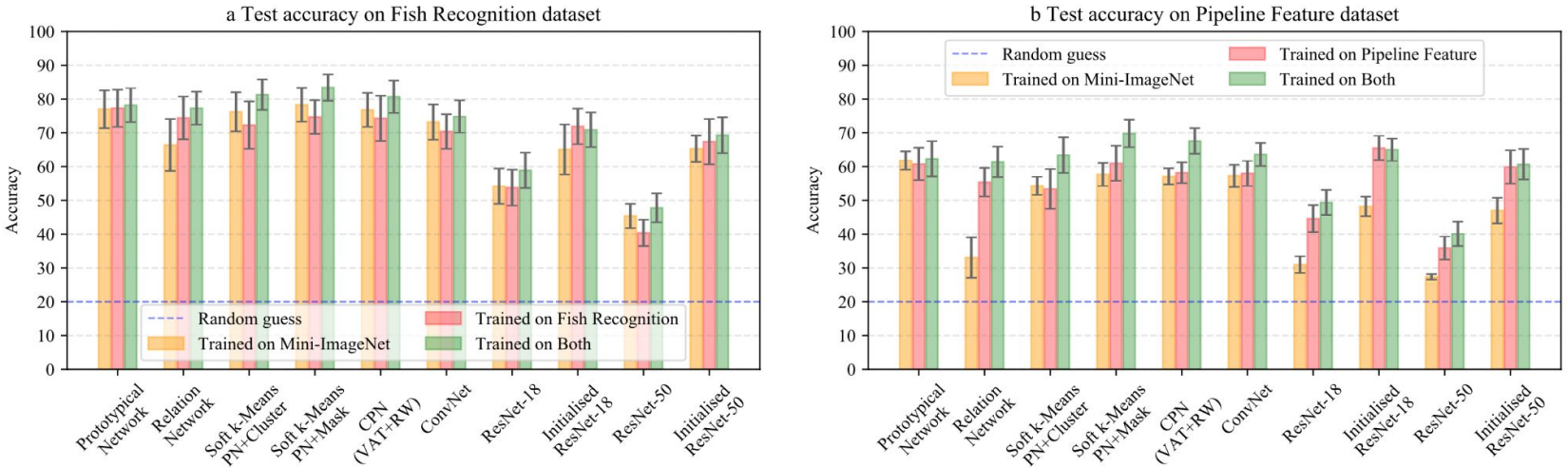


Figure: Test accuracy of models on the meta-testing split of underwater optical datasets after meta-/pre- training models on the meta-training split of Mini-ImageNet (yellow), the underwater dataset (red), and both datasets (green). The error bars show a 95% confidence interval.

Results: Sonar Datasets

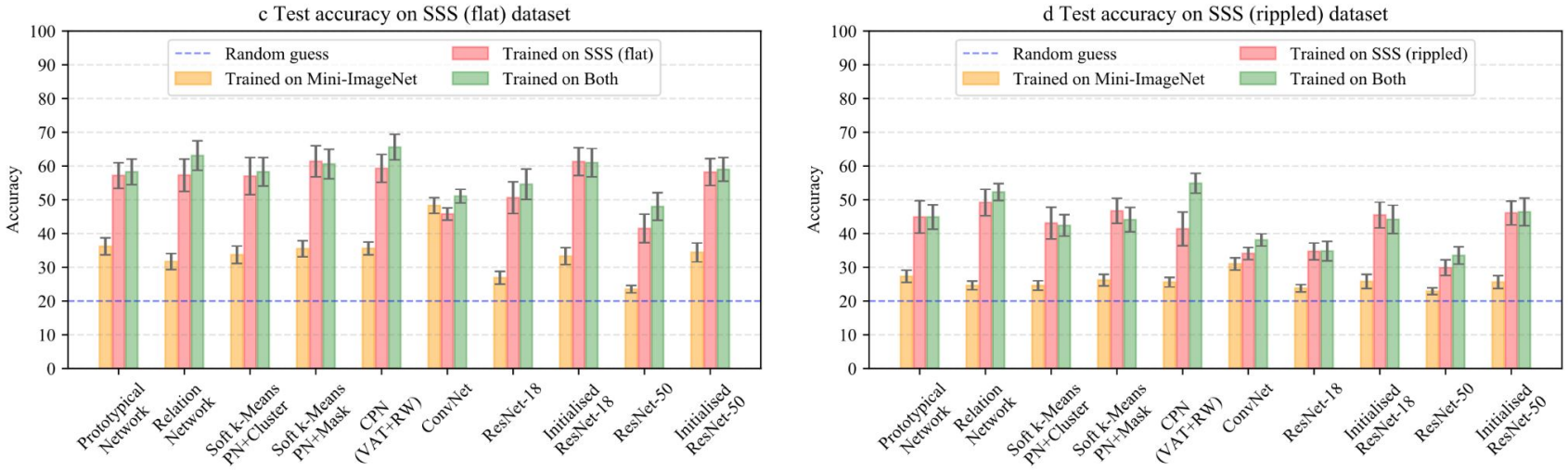


Figure: Test accuracy of models on the meta-testing split of simulated sonar datasets after meta-/pre- training models on the meta-training split of Mini-ImageNet (yellow), the underwater dataset (red), and both datasets (green). The error bars show a 95% confidence interval.

Key Findings

In this work, we found that:

- The more the underwater dataset’s style deviates from Mini-ImageNet, the worse the generalization
- FSL models had an average improvement of 7% accuracy over ConvNet fine-tuned baseline
  - up to 9% on optical and 17% on sonar
- Pre-meta-training produced an average improvement of 4% accuracy points over best of the other two meta-training scenarios
- Soft K-Means PN do best on optical images
  - 83.4% on fish and 69.8% on pipeline features
- CPN achieves the best performance on sonar
  - 65.6% on flat seabed and 54.9% on rippled seabed

Questions?

Hands-on Few-Shot Learning

- Goals
  - Getting to know low-level details of few-shot learning algorithms
  - Hands-on experience implementing FSL algorithms
  - Training and evaluating FSL models on underwater datasets

- Implementation of
  - FSL Task Sampler
  - Prototypical Networks
    - https://arxiv.org/pdf/1703.05175.pdf
  - MAML
    - https://arxiv.org/pdf/1703.03400.pdf
  - ProtoMAML
    - https://arxiv.org/pdf/1903.03096.pdf

Preliminaries:

- Code and Dataset
  - https://github.com/mattochal/fsl_tutorial_public
  - https://bit.ly/3qT61B8

- Conda:

```
conda create -n fsl_tutorial python=3.8
conda activate fsl_tutorial
conda install -y pytorch torchvision -c pytorch
conda install -y -c conda-forge tqdm
conda install -y -c anaconda pillow scikit-learn pytest
conda install -y -c conda-forge ipykernel
conda install -y -c conda-forge matplotlib
```

Back at 13:30

Hands-on Few-Shot Learning

Preliminaries:

- Code and Dataset
  - https://github.com/mattochal/fsl_tutorial_public
  - https://bit.ly/3qT61B8

- Conda:

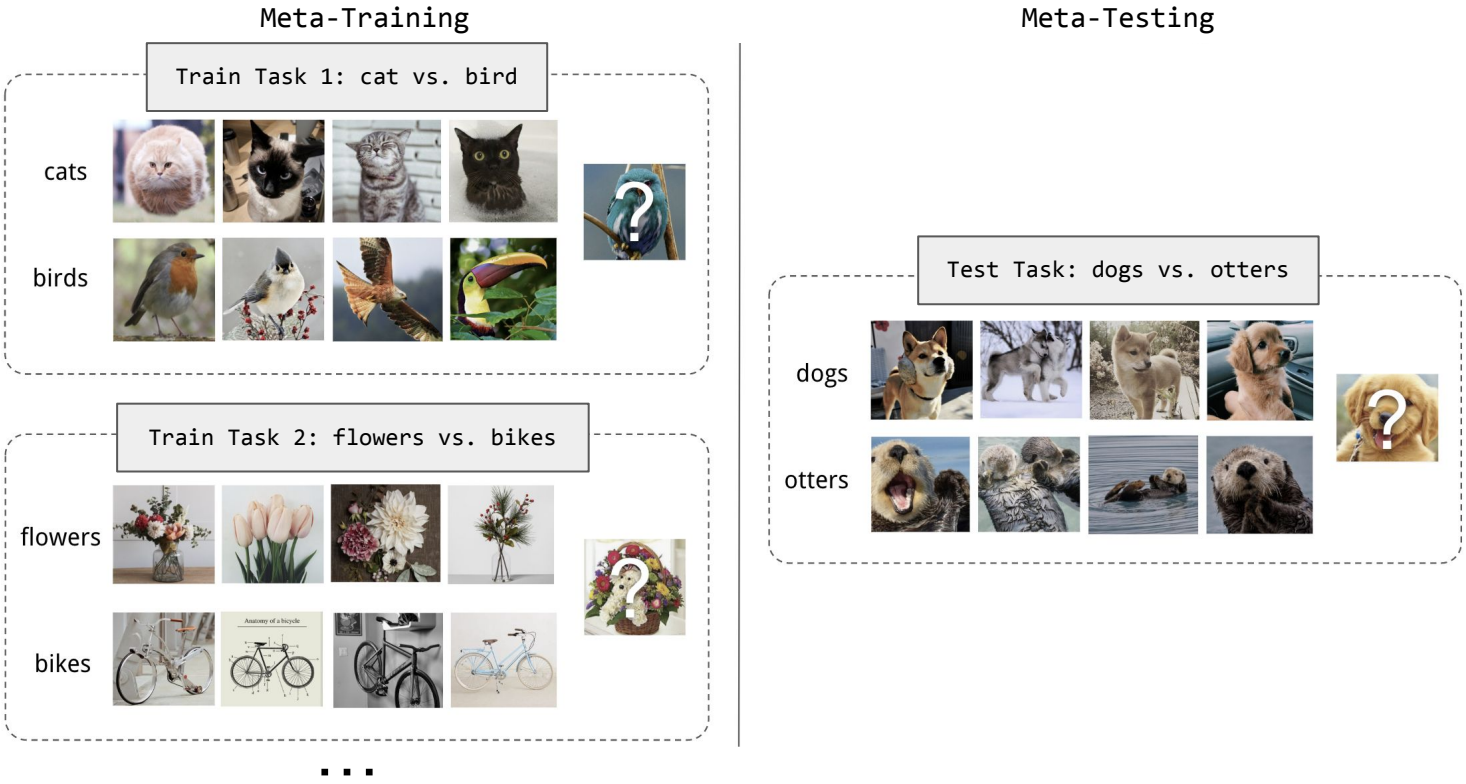
```
conda create -n fsl_tutorial python=3.8
conda activate fsl_tutorial
conda install -y pytorch torchvision -c pytorch
conda install -y -c conda-forge tqdm
conda install -y -c anaconda pillow scikit-learn pytest
conda install -y -c conda-forge ipykernel
conda install -y -c conda-forge matplotlib
```

Task

- What is provided:
  - Main code
  - Dataset
  - Backbones

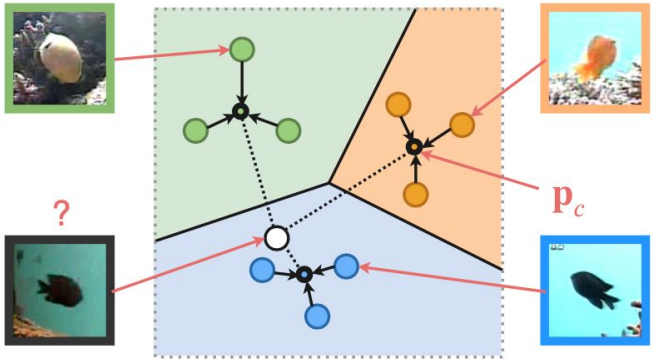
- What needs implementing:
  - Task Loader
  - Algorithms:
    - ProtoNet
    - MAML
    - ProtoMAML

Meta-Learning: Task Sampling



Metric-Learning: Prototypical Networks

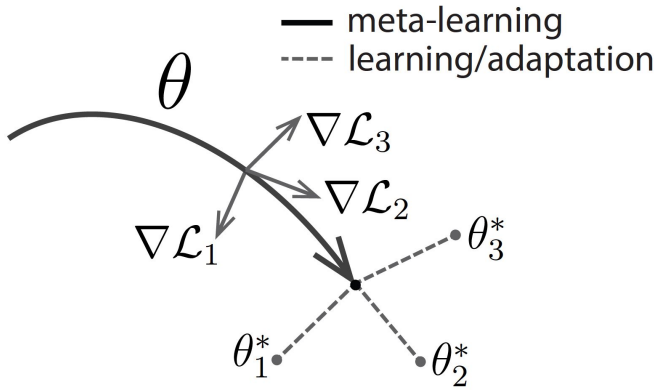
- Snell at el, 2017
  - https://arxiv.org/pdf/1703.05175.pdf
- Intuition:
  - Learns a representation by minimising the distances between samples and their class centroids (prototypes)
- A class prototype is:
  - The mean of all features belonging to a single class



$$p(y^* = k|\mathbf{x}^*, \mathcal{S}) = \frac{\exp(-||g(\mathbf{x}^*) - \mathbf{c}_k||_2^2)}{\sum_{k' \in \{1,\ldots,N\}} \exp(-||g(\mathbf{x}^*) - \mathbf{c}_{k'}||_2^2)}$$

Meta-Learning: MAML

- Model Agnostic Meta-Learning
- Finn et al, 2017
  - https://arxiv.org/pdf/1703.03400.pdf
- Intuition:
  - Meta-Learns a good initial condition of the model weights s.t. they can be easily fine-tuned on the support set of a target task
- Diagram parameters:
  - θ is the meta-learned initial starting position of for weights
  - $\nabla\mathcal{L}$ are the task gradients
  - θ^* are the task-optimal parameters
  - Expanded diagram on the next slide



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**

ProtoMAML

- Triantafillou et al., 2020
  - https://arxiv.org/pdf/1903.03096.pdf
- Combines ProtoNets and MAML together
  - Task adaptation mechanism is the same as MAML
  - Initializes the final FC weights with the class prototype
- Intuition:
  - Combines the best of both:
    - Class-separability of ProtoNets resulting in a good feature space
    - Task adaptation of MAML