Appendix for Learning Flexible Classifiers with Shot-CONditional Episodic (SCONE) Training

Anonymous Author(s)
Affiliation
Address
`email`

A Appendix

Additional Background: Prototypical Networks

Prototypical Networks Prototypical Networks (Snell et al., 2017) is a simple but effective episodic model which constructs a prototype ϕ_c for each class c in an episode as

$$\phi_c = \frac{1}{|\mathcal{S}_c|} \sum_{x \in \mathcal{S}_c} f_\theta(x), \tag{1}$$

where f is an embedding function parametrized by θ and $\mathcal{S}_c$ represents the set of support examples belonging to class c, and classifies a given query example as

$$p(y^* = c \mid x^*, \mathcal{S}) = \frac{\exp(-||x^* - \phi_c||_2^2)}{\sum_{c'} \exp(-||x^* - \phi_{c'}||_2^2)}. \tag{2}$$

Related Work

Few-shot classification A plethora of models have been recently proposed for few-shot classification, and we refer the reader to (Hospedales et al., 2020) for a broad survey. Before episodic training was introduced, few-shot classifiers often relied on metric learning (Koch et al., 2015; Triantafillou et al., 2017). This theme persisted in early episodic models like Matching Networks (Vinyals et al., 2016) and Prototypical Networks (Snell et al., 2017) where classification is made via nearest-neighbour comparisons in the embedding space. Matching Networks apply a soft k-NN algorithm where the label of a query example is predicted to be the weighted average of the (one-hot) support labels with the weights determined by the similarity of that query to each support example.

Gradient-based episodic models are another popular family of approaches following the influential MAML paper (Finn et al., 2017). To create a classifier for each given episode, this approach fine-tunes the embedding weights along with a linear classifier head using gradient descent on the support set. Intuitively, this results in learning an embedding space that serves as a useful starting point from which a few steps of gradient descent suffice to adapt the model to each episode's classification task. Proto-MAML (Triantafillou et al., 2020) is a simple extension that initializes the linear classifier for each episode from the prototypes of the classes appearing in that episode.

Recently, the field has shifted towards studying few-shot classification in more realistic environments like *tiered*-ImageNet (Ren et al., 2018) and Meta-Dataset (Triantafillou et al., 2020), which has encouraged research into newly-introduced challenges, such as accounting for multiple diverse datasets. Along these lines, Requeima et al. (2019); Bateni et al. (2019) proposed novel task conditioning approaches, Saikia et al. (2020) introduced an improved hyperparameter tuning approach, and Dvornik et al. (2020) proposed a method for selecting an appropriate set of features for each test episode out of a universal feature representation.

Understanding episodic learning Our work inscribes itself in a recent line of work attempting to understand the differences between episodic and non-episodic learning. Goldblum et al. (2020) attempts to understand episodic learning from the perspective of how classes cluster in feature-space (for models that learn a final classification layer on top of a feature extractor) as well as from the perspective of local minima clusters (for gradient-based meta-learners). Huang et al. (2020); Chao et al. (2020) draw parallels between learning episodes and supervised learning examples, Bronskill et al. (2020) discusses batch normalization in episodic learning, drawing parallels from its use in non-episodic learning and Chen et al. (2020) contrasts episodic and non-episodic learning in their ability to generalize to new examples of previously seen classes or new examples of *unseen* classes. Finally, Cao et al. (2020) theoretically investigates the role of the shot in Prototypical Networks to explain the observed performance drop when there is a mismatch between the shots at training and test time. Instead, we empirically study the effect of the shot chosen during episodic fine-tuning of a pre-trained solution, in a larger-scale and more diverse environment.

Feature-wise conditioning Feature-wise transformations such as FiLM (Perez et al., 2018) are used as a conditioning mechanism in a variety of problem settings; see Dumoulin et al. (2018) for a survey on the topic. In the few-shot classification setting, FiLM is used as a way to condition metric learners' backbones on the support set (Oreshkin et al., 2018; Requeima et al., 2019; Bateni et al., 2019), as well as a way to represent many pre-trained classifiers using a shared parametrization (Dvornik et al., 2020). Notably, TADAM (Oreshkin et al., 2018), CNAPs (Requeima et al., 2019) and Simple-CNAPs (Bateni et al., 2019) also use task conditioning, but they use the mean of the support set for this and thus the 'shot' information is discarded. The purpose of our conditioning mechanism is instead to make the backbone shot-aware. The idea of shot-conditional learners is inspired by recent work that investigates loss-conditional training using feature-wise transformations (Dosovitskiy and Djolonga, 2020; Babaeizadeh and Ghiasi, 2020).

Meta-Dataset

Meta-Dataset is comprised of ten distinct image datasets, including natural images, handwritten characters and sketches. It also defines a generative process for episodes that varies the way and shot across episodes, and within a particular episode varies the shot for different classes, introducing imbalance. The range of shots induced by this episode generator is also larger than what we considered in the previous section. It is a long-tailed distribution under which small and medium shots are more likely but it is possible to also encounter very large shots (e.g. >400), though this would happen very infrequently. We include histograms of the shot distributions of Meta-Dataset's training, validation and test episodes in the Appendix.

SCONE 's training algorithm in more detail

For clarity, we provide pseudocode for SCONE 's training algorithm including our procedure for shot smoothing in Algorithm 1. We will also release our code upon publication for reproducibility.

Visualizing the FiLM parameters that SCONE learns

Finally, as a sanity check, we perform a UMAP projection (McInnes et al., 2018) of the learned FiLM parameters for each shot setting (Figure 1). As expected, similar shot settings tend to learn similar sets of FiLM parameters, which is reflective of the fact that they rely on similar features for classification.

Example smoothed shot distribution

To gain an intuition on the effect of our smoothing procedure, we illustrate in Figure 2 the result of smoothing an example shot distribution using $m = 1 - 1e - 06$, which is the value of the smoothing hyperparameter that we used for our Prototypical Network experiments on Meta-Dataset. For this, we consider a hypothetical 4-way episode where the shots for the four classes are: 1, 10, 23, and 103. We observe that the largest peak is in the range of small values, due to the first three shots of the

Algorithm 1 SCONE training

```
Require: Distributions of training episodes $P_{train}$, pre-trained embedding weights $\theta$, pre-trained
    batch norm weights $\gamma$ and $\beta$, embedding function $f$, learning rate $\epsilon$ (a float), smoothing co-efficient
    $m$ (a float in the range $[0, 1]$) and maximum supported shot MAX-SHOT (an int).
Ensure: Finetuned embedding weights $\theta'$ and FiLM parameters $\mathcal{F} = \{\gamma', \beta'\}$.

  procedure SMOOTH-SHOT($s, m,$ MAX-SHOT)
      if $s >$ MAX-SHOT then
          $s \leftarrow$ MAX-SHOT                          ▷ Cap $s$ to the max supported shot
      end if
      $s \leftarrow s - 1$                                  ▷ So that $s$ is in the range $[0, \text{MAX-SHOT} - 1]$
      $\tilde{s} \leftarrow$ ONE-HOT($s$, DEPTH=MAX-SHOT)          ▷ Init the smoothed shot
      for $0 \leq j \leq$ MAX-SHOT do
          $l \leftarrow s - j - 1$                          ▷ The index $j$ slots to the left of $s$
          $l \leftarrow$ ONE-HOT($l$, DEPTH=MAX-SHOT) $* m$      ▷ Outputs the zero vector if $l < 0$
          $r \leftarrow s + j + 1$                          ▷ The index $j$ slots to the right of $s$
          $r \leftarrow$ ONE-HOT($r$, DEPTH=MAX-SHOT) $* m$      ▷ Outputs the zero vector if $r < 0$
          $\tilde{s} \leftarrow \tilde{s} + l + r$
          $m \leftarrow m^2$                                ▷ Adjust the next iteration's smoothing
      end for
  end procedure

  $\theta' \leftarrow \theta$                    ▷ Init the embedding weights from the pre-trained embeddings
  for $1 \leq k \leq$ MAX-SHOT do          ▷ Init the FiLM params from the pre-trained batch norm params
      $\gamma'(k) \leftarrow \gamma$
      $\beta'(k) \leftarrow \beta$
  end for
  while validation accuracy is improving do
      Sample a training episode with support set $\mathcal{S}$ and query set $\mathcal{Q}$
      Let $k_1, \ldots k_N$ be the shots of the episode's classes.
      $s \leftarrow$ ZEROS(MAX-SHOT)                     ▷ Init the (unnormalized) shot distribution
      for each class $i$ do
          $s_i \leftarrow$ ONE-HOT($k_i$, DEPTH = MAX-SHOT)
          $s_i \leftarrow$ SMOOTH-SHOT($s_i, m,$ MAX-SHOT)        ▷ Smooth the one-hot shot of class $i$
          $s \leftarrow s + s_i$
      end for
      $s \leftarrow s \div \text{SUM}(s)$                ▷ Normalize to get the episode's shot distribution
      $\gamma'_s \leftarrow s^T \gamma'$                  ▷ Select the FiLM params for the episode
      $\beta'_s \leftarrow s^T \beta'$
      Let $\mathcal{S}^H = \{f(x; \theta', \gamma'_s, \beta'_s), y\}_{(x,y) \in \mathcal{S}}$        ▷ The embedded support set
      Let $\mathcal{Q}^H = \{f(x; \theta', \gamma'_s, \beta'_s), y\}_{(x,y) \in \mathcal{Q}}$        ▷ The embedded query set
      $\mathcal{L} \leftarrow \frac{1}{|\mathcal{Q}^H|} \sum_{(h^*, y^*) \in \mathcal{Q}^H} - \log p(y^* \mid h^*, \mathcal{S}^H)$      ▷ Compute the episode's loss
      $\theta' \leftarrow \theta' - \epsilon \frac{\partial \mathcal{L}}{\partial \theta'}$          ▷ Update the model via gradient descent
      $\gamma' \leftarrow \gamma' - \epsilon \frac{\partial \mathcal{L}}{\partial \gamma'}$
      $\beta' \leftarrow \beta' - \epsilon \frac{\partial \mathcal{L}}{\partial \beta'}$
  end while
```

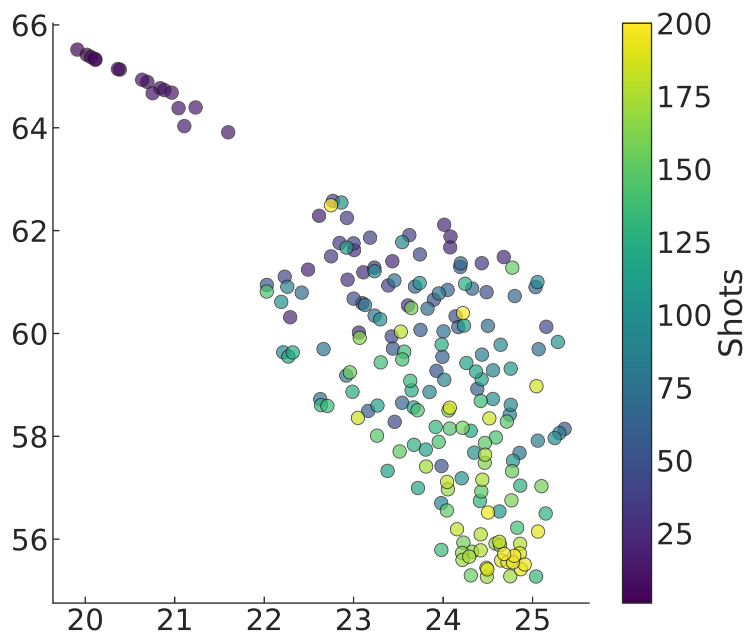


Figure 1: UMAP projection of the learned FiLM parameters for each "shot" setting, color-coded by shots.

episode, with the fourth shot causing a second peak around the value 103. As a reminder, this shot distribution defines the weights of the convex combination of FiLM parameters that will be used for the episode. In practice therefore, we are activating 'blocks' of FiLM parameters that are relevant for each episode, instead of strictly activating only the FiLM parameters of the observed shots.

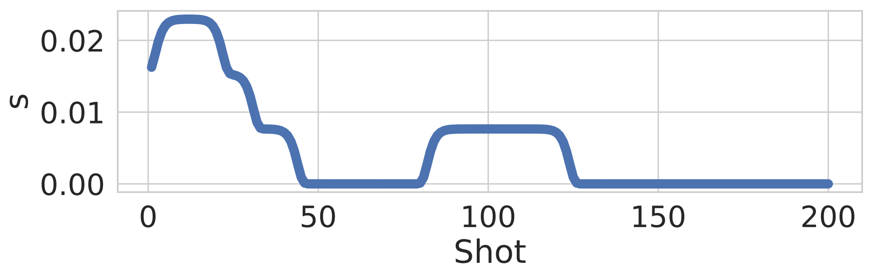


Figure 2: The shot distribution s produced according to our smoothing procedure for a hypothetical 4-way episode where the shots for the four classes are: 1, 10, 23, and 103.

Experimental details

We plan to open source our code upon publication, including all experimental details. In the meantime, we outline these details below for completeness.

Architecture We use ResNet-18 as the feature extractor for all of our experiments, following the implementation in (Triantafillou et al., 2020). For the SCONE variants, we add FiLM to all of the batch normalization layers throughout the network.

Image processing For all experiments, we use Meta-Dataset's input pipeline to obtain images, and we follow the image processing performed in (Chen et al., 2020) which yields images of size 128 x 128. We apply standard data augmentation consisting of horizontal flipping and random cropping followed by standardization using a commonly-used mean and standard deviation as in (Chen et al., 2020). For episodic models, data augmentation is applied in both the support and query sets. No data augmentation is used at validation nor test time.

Optimization We use ADAM with exponential learning rate decay and weight decay of $1e-8$ to optimize all models in this work. We tune the initial learning rate, the decay rate, and the number of updates between each learning rate decay separately for each model presented in the paper. The initial learning rate values we considered are 0.0005 and 0.001, with a decay factor of 0.8 or 0.9 applied every 1000, 2000, 3000 steps. We ran a variant for every combination of those values. We also tune the weight decay applied to the FiLM parameters (for SCONE variants) or the batch normalization parameters (for non-SCONE variants). We tried the values: $1e-8$, $1e-6$, $1e-4$.

SCONE hyperparameters For the SCONE variants, aside from the above hyperparameters, we additionally tune the smoothing parameter m described in the main paper that is used for training and for evaluation. We did not tune the MAX-SHOT hyperparameter mentioned in the main paper as we found that our initial choices worked reasonably. Specifically, we set it to 40 for the smaller-scale experiments where the maximum shot was 40, and to 200 for the large-scale experiments. The latter choice was performed heuristically since shots much larger than 200 are unlikely under the shot distribution induced by Meta-Dataset's episode generator. For more information on that shot distribution, we refer the reader to the next section.

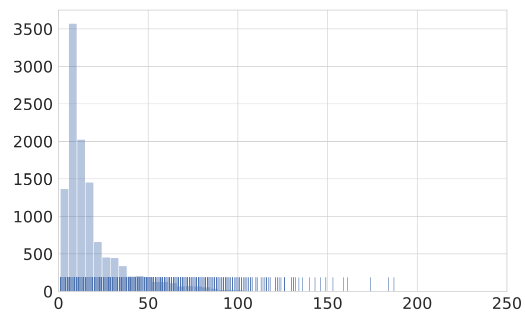
SCONE smoothing hyperparameter We tuned the value of this hyperparameter that will be used both at training and at evaluation. At training time, we considered values in the range $0, 0.2, 0.4, 0.6, 0.9$ for Prototypical Network experiments, and we picked the variant that worked best according to the validation performance that was computed without smoothing. Once the model was trained and all of the remaining hyperparamters were tuned, we performed a final validation round to tune the evaluation-time smoothing that will be used in the chosen model. We found it beneficial to use larger values here, picking the value of $1-1e-06$ for example for the Prototypical Network on ImageNet. In the Meta-Baseline codebase, we trained with larger values of smoothing (the best we found was $1-1e-10$) and didn't find it beneficial to additionally smooth at evaluation time.

Model selection For each experiment, we perform early stopping according to the performance on the validation set. For the models that train on a single shot k in the smaller-scale experiments, the validation performance that we monitor for early stopping is the average query set accuracy on k-shot 5-way episodes drawn from the validation set. For the models in the small-scale experiments that train on a distribution of shots, we use the average validation performance over 5-way episodes whose shot is sampled according to the same distribution used for training the respective model. For the larger-scale Meta-Dataset experiments, we draw validation episodes only from the validation set of ImageNet for the experiments that train on ImageNet only, or from the validation sets of all datasets for the experiments that train on all datasets. In both cases, the validation episodes are drawn using Meta-Dataset's episode generator that yields episodes of variable ways and variable shots with class imbalance. In all cases, the average validation performance is computed over 600 validation episodes and is monitored every 2K training updates. We apply exponential smoothing to the resulting validation "curve" (using the default value of 0.6 in TensorBoard). Then, we choose the update step at which the highest peak of that curve is found and we use the checkpoint corresponding to that update step for testing.

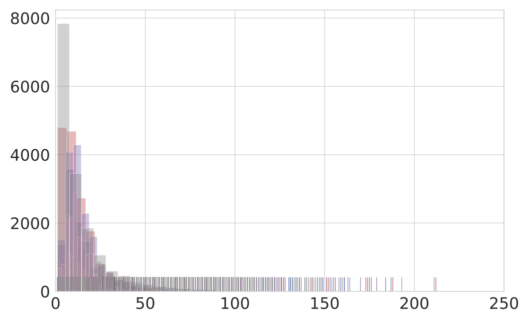
Hypothesis testing We follow the same procedure as in (Triantafillou et al., 2020) to determine which entries to bold in our tables. Specifically, we perform a 95% confidence interval statistical test on the difference between the mean accuracies of the two entries of that row. If we are not able to reject the null hypothesis that the difference between their means is 0, we bold both entries. If we are able to reject that hypothesis, we bold whichever entry has the largest mean accuracy.

Distribution of shots in Meta-Dataset episodes

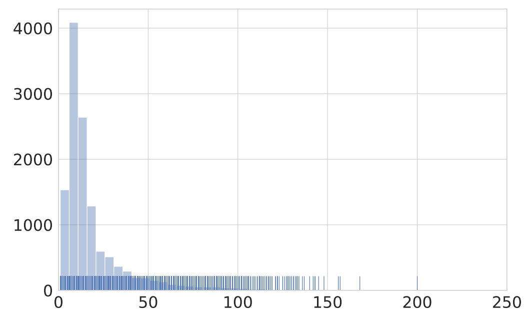
For reference, Figure 3 displays histograms of the number of shots produced by Meta-Dataset's episode sampling algorithm. These are computed by sampling 600 episodes per dataset for each of the training, validation and test splits of Meta-Dataset.



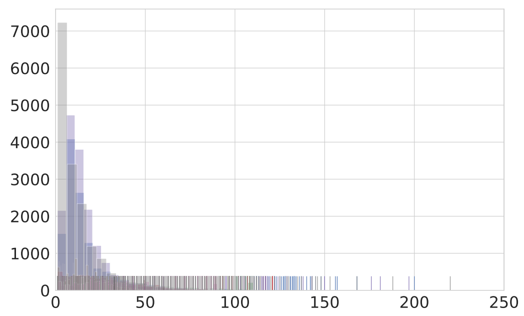
(a) Training episodes from ImageNet



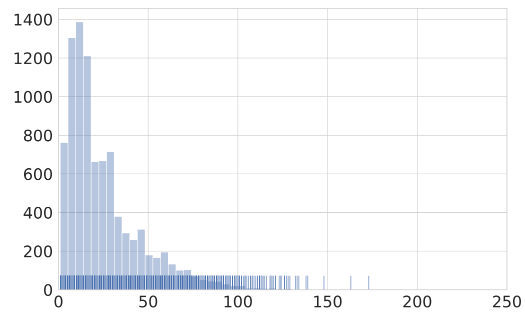
(b) Training episodes from all datasets



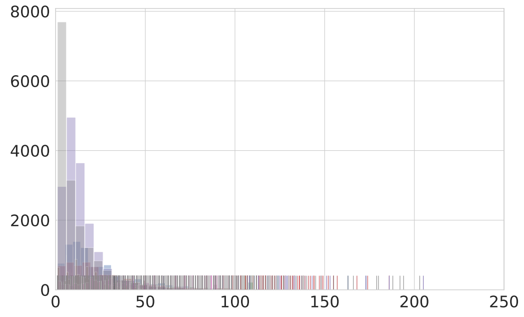
(c) Validation episodes from ImageNet



(d) Validation episodes from all datasets



(e) Test episodes from ImageNet



(f) Test episodes from all datasets

Figure 3: Histogram of shots appearing in episodes generated using Meta-Dataset's sampling algorithm for the different splits.

References

Mohammad Babaeizadeh and Golnaz Ghiasi. Adjustable real-time style transfer. In *Proceedings of the International Conference on Learning Representations*, 2020.

Peyman Bateni, Raghav Goyal, Vaden Masrani, Frank Wood, and Leonid Sigal. Improved few-shot visual classification. *arXiv preprint arXiv:1912.03432*, 2019.

John Bronskill, Jonathan Gordon, James Requeima, Sebastian Nowozin, and Richard E Turner. Tasknorm: Rethinking batch normalization for meta-learning. *arXiv preprint arXiv:2003.03284*, 2020.

Tianshi Cao, Marc Law, and Sanja Fidler. A theoretical analysis of the number of shots in few-shot learning. In *Proceedings of the International Conference on Learning Representations*, 2020.

Wei-Lun Chao, Han-Jia Ye, De-Chuan Zhan, Mark Campbell, and Kilian Q Weinberger. Revisiting meta-learning as supervised learning. *arXiv preprint arXiv:2002.00573*, 2020.

Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning. *arXiv preprint arXiv:2003.04390*, 2020.

Alexey Dosovitskiy and Josip Djolonga. You only train once: Loss-conditional training of deep networks. In *Proceedings of the International Conference on Learning Representations*, 2020.

Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. Feature-wise transformations. *Distill*, 2018.

Nikita Dvornik, Cordelia Schmid, and Julien Mairal. Selecting relevant features from a universal representation for few-shot classification. *arXiv preprint arXiv:2003.09338*, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

Micah Goldblum, Steven Reich, Liam Fowl, Renkun Ni, Valeriia Cherepanova, and Tom Goldstein. Unraveling meta-learning: Understanding feature representations for few-shot tasks. In *Proceedings of the International Conference on Machine Learning*, 2020.

Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

Meiyu Huang, Xueshuang Xiang, and Yao Xu. Training few-shot classification via the perspective of minibatch and pretraining. *arXiv preprint arXiv:2004.05910*, 2020.

Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2. Lille, 2015.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *Advances in Neural Information Processing Systems*, pages 721–731, 2018.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua B Tenenbaum, Hugo Larochelle, and Richard S Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

James Requeima, Jonathan Gordon, John Bronskill, Sebastian Nowozin, and Richard E Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. In *Advances in Neural Information Processing Systems*, pages 7957–7968, 2019.

Tonmoy Saikia, Thomas Brox, and Cordelia Schmid. Optimized generic feature learning for few-shot classification across domains. *arXiv preprint arXiv:2001.07926*, 2020.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in neural information processing systems*, pages 4077–4087, 2017.

Eleni Triantafillou, Richard Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens. In *Advances in Neural Information Processing Systems*, pages 2255–2265, 2017.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, et al. Meta-Dataset: A dataset of datasets for learning to learn from few examples. In *Proceedings of the International Conference on Learning Representations*, 2020.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.