Improved Few-Shot Visual Classification

Peyman Bateni[1], Raghav Goyal[1,3], Vaden Masrani[1], Frank Wood[1,2,4], Leonid Sigal[1,3,4]
[1]University of British Columbia, [2]MILA, [3]Vector Institute, [4]CIFAR AI Chair
{pbateni, rgoyal14, vadmas, fwood, lsigal}@cs.ubc.ca

Abstract

Few-shot learning carries the promise of alleviating the need for exhaustively labeled data. In this paper, we present a classifier based on the Mahalanobis distance adopted into a state of the art few-shot learning approach (CNAPS [20]). Our method allows for useful high dimensional feature covariance estimates from surprisingly few samples, which results in a new "Simple CNAPS" architecture which performs up to 6.1% better than CNAPS despite up to 9.2% fewer trainable parameters. Note that this paper is a workshop adapted shorter version of the longer 8 page paper that was accepted into the main conference at CVPR 2020.

1. Introduction and Related Work

Deep learning successes have led to major computer vision advances [8, 25, 6]. However, most such methods operate in fully-supervised, high data availability regimes. Few-shot learning aims to adapt models such that they work well on instances from classes not seen at training time, given only a few labelled "support" examples for each new class.

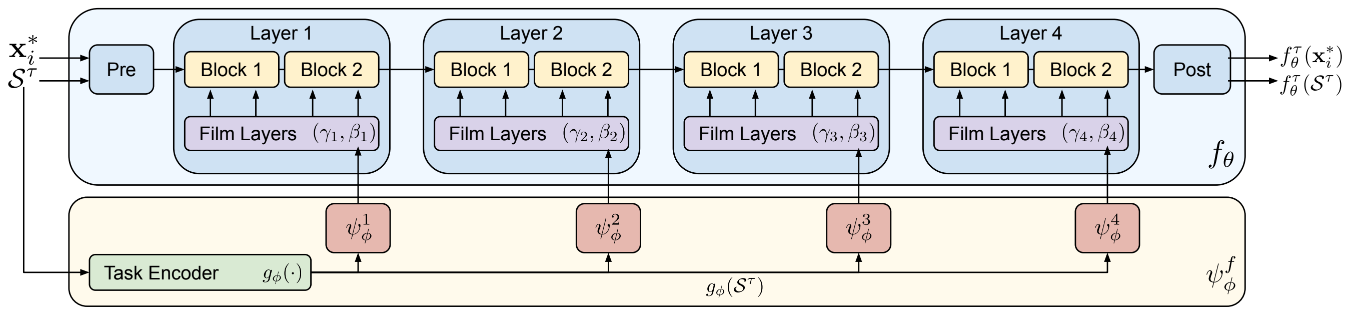
Few-shot learning approaches typically take one of two forms: 1) nearest neighbor approaches [27] that effectively apply nearest-neighbor classification on samples themselves, either in a feature [10, 11, 22] or a semantic space [3]; or 2) embedding methods that effectively distill examples to a single class prototype that is either learned [4, 20] or implicitly derived from the samples [24]. The prototypes are often defined in feature or semantic space (*e.g.* word2vec [29]). Most research in this domain has focused on learning non-linear mappings (often expressed as neural nets) from images to the embedding space subject to a pre-defined metric in the embedding space used for final nearest class classification (usually cosine similarity between query image embedding and class embedding). Most recently CNAPS [20] achieved state of the art (SoTA) few-shot visual image classification by utilizing sparse FiLM [19] layers within the context of episodic training to get around problems that arise from trying to adapt the entire embedding neural network using few support samples.

Overall, much less attention has been given to the metric used to compute distances for classification in the embedding space. In [24] the authors analyze the underlying distance function used in order to justify the use of sample means as prototypes. They argue that Bregman divergences [1] are the theoretically sound family of metrics to use in this setting, but only utilize a single instance within this class — squared Euclidean distance, which they find to perform better than the more traditional cosine metric. However, the choice of Euclidean metric involves making two flawed assumptions: 1) that feature dimensions are uncorrelated and 2) that they have uniform variance. Further it is insensitive to the distribution of within-class samples with respect to their prototype and recent results [18, 24] suggest that this is problematic. Modeling this distribution (in the case of [1] using extreme value theory) is, as we find, a key to better performance.

Our Contributions: 1. A new "Simple CNAPS" architecture that achieves 6.1% improvement over SoTA (CNAPS [20]) while removing 788,485 parameters (3.2%-9.2% of the total) from original CNAPS architecture, replacing them with fixed, not-learned, deterministic covariance estimation and Mahalanobis distance computations. 2. The surprising finding that we are able to estimate such a metric even in the few shot classification setting, where the number of available support examples per class is far too few in theory to estimate the required class-specific covariances.

2. Formal Problem Definition

We frame few-shot image classification as an amortized classification task. Assume that we have a large labelled dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$ of images $\mathbf{x}_i$ and labels y_i. From this dataset we can construct a very large number of classification *tasks* $\mathcal{D}^\tau \subseteq \mathcal{D}$ by repeatedly sampling without replacement from $\mathcal{D}$. Let $\tau \in \mathbb{Z}_+$ uniquely identify a classification task. We define the *support set* of a task to be $\mathcal{S}^\tau = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N^\tau}$ and the *query set* $Q^\tau = \{(\mathbf{x}_i^*, y_i^*)\}_{i=1}^{N^{*\tau}}$ where $\mathcal{D}^\tau = \mathcal{S}^\tau \cup Q^\tau$ where $\mathbf{x}_i, \mathbf{x}_i^* \in \mathbb{R}^D$ are vectorized images and $y_i, y_i^* \in \{1, ..., K\}$ are class labels. Our objective is to find parameters θ of a classifier f_θ that maximizes



$\mathbf{x}_i^*$
$\mathcal{S}^\tau$
Pre
Layer 1
Block 1
Block 2
Film Layers (γ_1, β_1)
Layer 2
Film Layers (γ_2, β_2)
Layer 3
Film Layers (γ_3, β_3)
Layer 4
Film Layers (γ_4, β_4)
Post
$f_\theta^\tau(\mathbf{x}_i^*)$
$f_\theta^\tau(\mathcal{S}^\tau)$
f_θ
ψ_ϕ^1
ψ_ϕ^2
ψ_ϕ^3
ψ_ϕ^4
Task Encoder $g_\phi(\cdot)$
$g_\phi(\mathcal{S}^\tau)$
ψ_ϕ^f

Figure 1: **Overview of the feature extractor adaptation methodology in CNAPS:** task encoder $g_\phi(\cdot)$ provides the adaptation network ψ_ϕ^i at each block i with the task representations $(g_\phi(\mathcal{S}^\tau)$ to produce FiLM parameters (γ_j, β_j).

$\mathbb{E}_\tau[\prod_{Q^\tau} p(y_i^*|f_\theta(\mathbf{x}_i^*, \mathcal{S}^\tau)]$. In practice $\mathcal{D}$ is constructed by concatenating large image classification datasets and the set of classification tasks $\{\mathcal{D}^\tau\}_{\tau=1}^{\cdot}$ is sampled in a more complex way than simply without replacement.

3. Method

Conditional Neural Adapative Processes (CNAPS) consist of: a feature extractor and a classifier, both of which are task-adapted. Our method also employs the feature extractor of CNAPS, which is shown in Figure 1. It consists of a ResNet18 [5] network pre-trained on ImageNet [21] which also has been augmented with FiLM layers [19]. The parameters $\{\gamma_j, \beta_j\}_{j=1}^4$ of the FiLM layers can scale and shift the extracted features at each layer of the ResNet18, allowing the feature extractor to focus and disregard different features on a task-by-task basis. A feature adaptation module ψ_ϕ^f is trained to produce $\{\boldsymbol{\gamma}_j, \boldsymbol{\beta}_j\}_{j=1}^4$ based on the support examples $\mathcal{S}^\tau$ provided for the task. The feature extractor adaptation module ψ_ϕ^f consists of two stages: support set encoding followed by film layer parameter production. The set encoder $g_\phi(\cdot)$, parameterized by a deep neural network, produces a permutation invariant task representation $g_\phi(\mathcal{S}^\tau)$ based on the support images $\mathcal{S}^\tau$. This task representation is then passed to ψ_ϕ^j which then produces the FiLM parameters $\{\boldsymbol{\gamma}_j, \boldsymbol{\beta}_j\}$ for each block j in the ResNet. Once the FiLM parameters have been set, the feature extractor has been adapted to the task. We use f_θ^τ to denote the feature extractor adapted to task τ. The CNAPS paper [20] also proposes an auto-regressive adaptation method which conditions each adaptor ψ_ϕ^j on the output of the previous adapter ψ_ϕ^{j-1}. We refer to this variation as AR-CNAPS but for conciseness we omit the details of this architecture here, and instead refer the interested reader to [20].

Classification in CNAPS is performed by a task-adapted linear classifier where the class probabilities for a query image $\mathbf{x}_i^*$ are computed as softmax$(\mathbf{W} f_\theta^\tau(\mathbf{x}_i^*) + \mathbf{b})$. The classification weights $\mathbf{W}$ and biases $\mathbf{b}$ are produced by the classifier adaptation network ψ_ϕ^c forming $[\mathbf{W}, \mathbf{b}] = [\psi_\phi^c(\boldsymbol{\mu}_1)\ \psi_\phi^c(\boldsymbol{\mu}_2)\ \ldots\ \psi_\phi^c(\boldsymbol{\mu}_K)]^T$ where for each class k in the task, the corresponding row of classification weights is produced by ψ_ϕ^c from the class mean $\boldsymbol{\mu}_k$. The class mean $\boldsymbol{\mu}_k$ is obtained by mean-pooling the feature vectors of the support examples for class k extracted by the adapted feature extractor f_θ^τ.

3.1. Simple CNAPS

In Simple CNAPS, we also use the same pre-trained ResNet18 for feature extraction with the same adaptation module ψ_ϕ^f, although, because of the classifier architecture we use, it becomes trained to do something different than it does in CNAPS. This choice, like for CNAPS, allows for a task-specific adaptation of the feature extractor. Unlike CNAPS, we directly compute

$$p(y_i^* = k|f_\theta^\tau(\mathbf{x}_i^*), \mathcal{S}^\tau) = \text{softmax}(-d_k(f_\theta^\tau(\mathbf{x}_i^*), \boldsymbol{\mu}_k)) \quad (1)$$

using a deterministic, fixed d_k

$$d_k(\mathbf{x}, \mathbf{y}) = \frac{1}{2}(\mathbf{x} - \mathbf{y})^T (\mathbf{Q}_k^\tau)^{-1} (\mathbf{x} - \mathbf{y}). \quad (2)$$

Here $\mathbf{Q}_k^\tau$ is a covariance matrix specific to the task and class.

As we cannot know the value of $\mathbf{Q}_k^\tau$ ahead of time, it must be estimated from the feature embeddings of the task-specific support set. As the number of examples in any particular support set is likely to be much smaller than the dimension of the feature space, we use a regularized estimator

$$\mathbf{Q}_k^\tau = \lambda_k^\tau \boldsymbol{\Sigma}_k^\tau + (1 - \lambda_k^\tau)\boldsymbol{\Sigma}^\tau + \beta I. \quad (3)$$

formed from a convex combination of the class-within-task and all-classes-in-task covariance matrices $\boldsymbol{\Sigma}_k^\tau$ and $\boldsymbol{\Sigma}^\tau$ respectively.

We estimate the class-within-task covariance matrix $\boldsymbol{\Sigma}_k^\tau$ using the feature embeddings $f_\theta^\tau(\mathbf{x}_i)$ of all $\mathbf{x}_i \in \mathcal{S}_k^\tau$ where $\mathcal{S}_k^\tau$ is the set of examples in $\mathcal{S}^\tau$ with class label k.

$$\boldsymbol{\Sigma}_k^\tau = \frac{1}{|\mathcal{S}_k^\tau| - 1} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{S}_k^\tau} (f_\theta^\tau(\mathbf{x}_i) - \boldsymbol{\mu}_k)(f_\theta^\tau(\mathbf{x}_i) - \boldsymbol{\mu}_k)^T.$$

If the number of support instance of that class is one, i.e. $|\mathcal{S}_k^\tau| = 1$, then we define $\mathbf{\Sigma}_k^\tau$ to be the zero matrix of the appropriate size. The all-classes-in-task covariance $\mathbf{\Sigma}^\tau$ is estimated in the same way as the class-within-task except that it uses *all* the support set examples $\mathbf{x}_i \in \mathcal{S}^\tau$ regardless of their class. We choose a particular, deterministic scheme for computing the weighting of class and task specific covariance estimates, $\lambda_k^\tau = |\mathcal{S}_k^\tau|/(|\mathcal{S}_k^\tau|+1)$. This choice means that in the case of a single labeled instance for class in the support set, a single "shot," $\mathbf{Q}_k^\tau = 0.5\mathbf{\Sigma}^\tau + \beta\mathbf{I}$. This can be viewed as increasing the strength of the regularization parameter β relative to the task covariance $\mathbf{\Sigma}^\tau$. When $|\mathcal{S}_k^\tau|= 2$, λ_k^τ becomes $2/3$ and $\mathbf{Q}_k^\tau$ only partially favors the class-level covariance over the all-class-level covariance. In a high-shot setting, λ_k^τ tends to 1 and $\mathbf{Q}_k^\tau$ mainly consists of the class-level covariance. The intuition behind this formula for λ_k^τ is that the higher the number of shots, the better the class-within-task covariance estimate gets, and the more $\mathbf{Q}_k^\tau$ starts to look like $\mathbf{\Sigma}_k^\tau$. We considered other ratios and making λ_k^τ's learnable parameters, but found that out of all the considered alternatives the simple deterministic ratio above produced the best results.

4. Results

Dataset Meta-Dataset [26] is a benchmark for few-shot learning and image classification. It is the union of 10 labeled image datasets: ILSVRC-2012 (ImageNet) [21], Omniglot [13], FGVC-Aircraft (Aircraft) [16], CUB-200-2011 (Birds) [28], Describable Textures (DTD) [2], QuickDraw [9], FGVCx Fungi (Fungi) [23], VGG Flower (Flower) [17], Traffic Signs (Signs) [7] and MSCOCO [15]. In keeping with prior art we report results using the first 8 datasets for training, reserving Traffic Signs and MSCOCO for "out-of-domain" performance evaluation. Following [20], we extend the out-of-domain evaluation with 3 more datasets: MNIST [14], CIFAR10 [12] and CIFAR100 [12].

4.1. Results

As shown in Table 1, in-domain performance gains are considerable in the few-shot learning domain with 2-6% margins. Simple CNAPS achieves an average 73.8% accuracy on in-domain few-shot classification, a 4.2% gain over CNAPS, while Simple AR-CNAPS achieves 73.5% accuracy, a 3.8% gain over AR-CNAPS. Furthermore, both models produce substantial gains in classification accuracy on out-of-domain datasets, each exceeding the SoTA baselines. With an average out-of-domain accuracy of 69.7% and 67.6%, Simple CNAPS and Simple AR-CNAPS outperform SoTA by 8.2% and 7.8%. This means that Simple CNAPS/AR-CNAPS generalizes to out-of-domain datasets better than baseline models. Overall, Simple CNAPS achieves the best classification accuracy at 72.2% with Simple AR-CNAPS trailing very closely at 71.2%. Figure 2 shows how the overall classification accuracy varies as a function of the average number of support examples per class (shots) over all tasks. We compare Simple CNAPS, original CNAPS, and the squared Euclidean variant of our method. As expected, the average number of support examples per class is highly correlated with the performance. All methods perform better with more labeled examples per support class, with Simple CNAPS performing substantially better as the number of shots increases. The surprising discovery is that Simple CNAPS is effective even when the number of labeled instances is as low as four, suggesting both that even poor estimates of the task and class specific covariance matrices are helpful and that the regularization scheme we have introduced works remarkably well.

	Average Accuracy (%)		
Model	In-Domain	Out-Domain	Overall
MAML	50.4±1.0	29.2±1.2	46.2±1.1
RelationNet	58.6±0.9	32.5±0.9	53.3±0.9
k-NN	59.2±0.9	34.9±1.1	54.3±0.9
MatchingNet	59.8±0.9	42.2±1.1	56.3±1.0
Finetune	60.7±1.1	51.0±1.2	58.8±1.1
ProtoNet	64.9±1.0	56.4±0.9	61.6±0.9
ProtoMAML	67.5±0.9	46.8±1.1	63.4±0.9
CNAPS	69.6±0.8	59.8±0.8	65.9±0.8
AR-CNAPS	69.7±0.8	61.5±0.8	66.5±0.8
Simple AR-CNAPS	**73.5±0.8**	67.6±0.8	**71.2±0.8**
Simple CNAPS	**73.8±0.8**	**69.7±0.8**	**72.2±0.8**

Table 1: In-domain, out-of-domain and overall mean classification accuracies compared to the baselines on Meta-Dataset [26]. Error bars represent 95% confidence intervals.

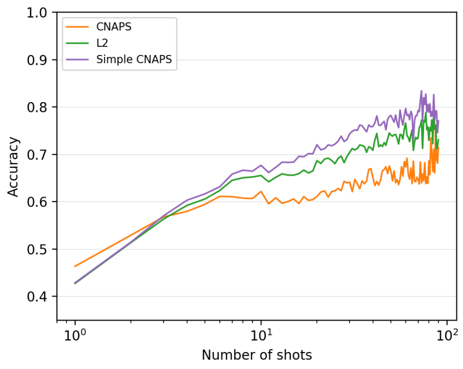


Figure 2: **Accuracy vs. Shots:** Average number of support examples (in log scale) per class v/s accuracy.

5. Discussion

Few shot learning is a fundamental task in modern AI research. In this paper we have introduced a new method for amortized few shot image classification which establishes a new SoTA performance benchmark by making a simplification to the current SoTA architecture. Our specific architectural choice, that of deterministically estimating and using Mahalanobis distances for classification of task-adjusted class-specific feature vectors seems to produce, via training, embeddings that generally allow for useful covariance estimates, even when the number of labeled instances per task and class is small. In the future, exploration of other Bregman divergences can be an avenue of potentially fruitful research.

References

[1] A. Banerjee, S. Merugu, I. S. Dhillon, and J. Ghosh. Clustering with bregman divergences. *Journal of machine learning research*, 6(Oct):1705–1749, 2005. 1

[2] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, 2014. 3

[3] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. A. Ranzato, and T. Mikolov. Devise: A deep visual-semantic embedding model. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2121–2129. Curran Associates, Inc., 2013. 1

[4] S. Gidaris and N. Komodakis. Generating classification weights with gnn denoising autoencoders for few-shot learning. *arXiv preprint arXiv:1905.01102*, 2019. 1

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 2

[6] M. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga. A comprehensive survey of deep learning for image captioning. *ACM Comput. Surv.*, 51(6):118:1–118:36, Feb. 2019. 1

[7] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel. Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In *The 2013 international joint conference on neural networks (IJCNN)*, pages 1–8. IEEE, 2013. 3

[8] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu. A survey of deep learning-based object detection. *CoRR*, abs/1907.09408, 2019. 1

[9] J. Jongejan, H. Rowley, T. Kawashima, J. Kim, and N. Fox-Gieg. The quick, draw!-ai experiment.(2016), 2016. 3

[10] J. Kim, T. Kim, S. Kim, and C. D. Yoo. Edge-labeling graph neural network for few-shot learning. *CoRR*, abs/1905.01436, 2019. 1

[11] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, 2015. 1

[12] A. Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009. 3

[13] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015. 3

[14] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010. 3

[15] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 3

[16] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151*, 2013. 3

[17] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008. 3

[18] B. Oreshkin, P. Rodríguez López, and A. Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 721–731. Curran Associates, Inc., 2018. 1

[19] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 1, 2

[20] J. Requeima, J. Gordon, J. Bronskill, S. Nowozin, and R. E. Turner. Fast and flexible multi-task classification using conditional neural adaptive processes. *arXiv preprint arXiv:1906.07697*, 2019. 1, 2, 3

[21] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 2, 3

[22] V. G. Satorras and J. B. Estrach. Few-shot learning with graph neural networks. In *International Conference on Learning Representations*, 2018. 1

[23] B. Schroeder and Y. Cui. Fgvcx fungi classification challenge 2018. `https://github.com/visipedia/fgvcx_fungi_comp`, 2018. 3

[24] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017. 1

[25] M. Sornam, K. Muthusubash, and V. Vanitha. A survey on image classification and activity recognition using deep convolutional neural network architecture. In *2017 Ninth International Conference on Advanced Computing (ICoAC)*, pages 121–126, Dec 2017. 1

[26] E. Triantafillou, T. Zhu, V. Dumoulin, P. Lamblin, K. Xu, R. Goroshin, C. Gelada, K. Swersky, P.-A. Manzagol, and H. Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. *arXiv preprint arXiv:1903.03096*, 2019. 3

[27] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016. 1

[28] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 3

[29] C. Xing, N. Rostamzadeh, B. N. Oreshkin, and P. O. Pinheiro. Adaptive cross-modal few-shot learning. *CoRR*, abs/1902.07104, 2019. 1