Stochastic Prototype Embeddings

Tyler R. Scott [1 2] **Karl Ridgeway** [1] **Michael C. Mozer** [1 3]

Abstract

Supervised deep-embedding methods project inputs of a domain to a representational space where same-class instances lie near one another and different-class instances lie far apart. We propose a probabilistic method that treats embeddings as random variables. Based on a state-of-the-art deterministic method, Prototypical Networks (Snell et al., 2017), our approach supposes the existence of a class prototype around which class instances are Gaussian distributed. The prototype posterior is a product distribution over labeled instances, and query instances are classified by marginalizing relative prototype proximity over embedding uncertainty. We describe an efficient sampler for approximate inference that allows us to train the model at roughly the same space and time cost as its deterministic sibling. Incorporating uncertainty improves performance on few-shot learning and gracefully handles label noise and out-of-distribution inputs.

1. Introduction

Supervised deep-embedding methods map instances from an input space to a latent embedding space in which same-label pairs are near and different-label pairs are far. The embedding thus captures semantic relationships without discarding inter-class structure. In contrast, consider a standard neural network classifier with a softmax output layer trained with a cross-entropy loss. Although its penultimate layer might be treated as an embedding, the classifier's training objective attempts to orthogonalize all classes and thereby eliminate any information about inter-class structure.

Embedding methods are popular in the few-shot learning literature (Koch et al., 2015; Vinyals et al., 2016; Snell et al., 2017; Triantafillou et al., 2017; Finn et al., 2017; Edwards & Storkey, 2017; Scott et al., 2018; Ridgeway & Mozer, 2018; Mishra et al., 2018) where the goal is to classify query instances based on one or a small number of labeled exemplars of novel classes. These methods operate by embedding the queries and exemplars using a pre-trained network, and classifying each query according to its proximity to the exemplars. Embedding methods are also critical in open-set recognition domains such as face recognition and person re-identification.

Loss functions used to obtain embeddings can be characterized according to the number of instances used to specify a loss. To describe these losses, we will use the notation z_α for an embedding of class α. *Pairwise* losses attempt to minimize within-class distances, $||z_\alpha - z'_\alpha||$, and maximize between-class distances, $||z_\alpha - z_\beta||$ (Chopra et al., 2005; Yi et al., 2014; Hadsell et al., 2006; Oh et al., 2019). *Triplet* losses attempt to ensure within-class instances are closer than between-class instances, $||z_\alpha - z'_\alpha|| < ||z_\alpha - z_\beta||$ (Schroff et al., 2015; Song et al., 2016; Wang et al., 2017; Karaletsos et al., 2016). *Quadruplet* losses attempt to ensure every within-class pair is closer than every between-class pair, $||z_\alpha - z'_\alpha|| < ||z''_\alpha - z_\beta||$ (Ustinova & Lempitsky, 2016). Finally, *cluster-based losses* attempt to use all instances of a class (Snell et al., 2017; Song et al., 2017; Ridgeway & Mozer, 2018; Rippel et al., 2016); for example, *Prototypical Networks (PN)* compute the mean of a set of instances of a class, $\bar{z}_\alpha$, and ensure that additional instances of that class, z_α, satisfy a proximity constraint such as $||z_\alpha - \bar{z}_\alpha|| < ||z_\alpha - \bar{z}_\beta||$.

Nearly all methods previously proposed for deep embeddings are deterministic: an instance projects to a single point in the embedding space. Deterministic embeddings fail to capture uncertainty due either to out-of-distribution inputs (e.g., data corruption) or label ambiguity (e.g., overlapping classes). Representing uncertainty is important for many reasons, including robust classification and decision making, informing downstream models, interpreting representations, and detecting out-of-distribution samples. Our proposed method, the *Stochastic Prototype Embedding (SPE)*, is a variant of the PN (Snell et al., 2017), described above. As in the PN, our SPE assumes each class can be characterized by a prototype in the embedding space and an instance is classified based on its proximity to a prototype. In the case of the SPE, the embeddings and prototypes are Gaussian

[1]Department of Computer Science, University of Colorado Boulder [2]Sensory Inc. [3]Google Research. Correspondence to: Tyler R. Scott <tysc7237@colorado.edu>.

Presented at the ICML 2019 Workshop on Uncertainty and Robustness in Deep Learning. Copyright 2019 by the author(s).

random variables, each class instance is assumed to be a Gaussian perturbation of the prototype, and a query instance is classified by marginalizing out over the embedding uncertainty.

Using a synthetic data set, we demonstrate that the embedding uncertainty is related to both input and label noise. And on a few-shot learning task, we show that the SPE outperforms its state-of-the-art deterministic sibling, the PN.

2. The Model

The SPE assumes that the latent representation, z, is a Gaussian conditioned on the input, x:

$$p(z|x) = \mathcal{N}(z; \boldsymbol{\mu}_x, \boldsymbol{\sigma}_x^2 \mathbf{I}) \tag{1}$$

with mean, $\boldsymbol{\mu}_x$, and variance, $\boldsymbol{\sigma}_x^2$, computed by a deep neural network, similar to a Variational Autoencoder (Kingma & Welling, 2014). (We discuss the possibility of using a full covariance matrix in Appendix C.) The classification, y, in turn is conditioned on z, with $p(y|z)$ taking the same form as in the original PN (Snell et al., 2017), to be described shortly. Given an input, a class prediction is made by marginalizing over the embedding uncertainty:

$$p(y|x) = \int_z p(y|z)p(z|x)dz, \tag{2}$$

We train the SPE using the standard few-shot learning paradigm, consisting of a sequence of *episodes*, each with m instances of n classes. We split the $m \times n$ instances into $k \times n$ *support* examples, defining a set S, and $(m-k) \times n$ *query* examples. The support instances for each class c, $S_c \in S$, are used to determine the class prototype, ν_c, and the query instances are evaluated to predict class label (Equation 2).

2.1. Forming class prototypes

In the SPE, each class y has an associated prototype, ν_y, in the embedding space, and each instance i of class y, denoted x_i, projects to an embedding, z_i, in the neighborhood of ν_y:

$$\nu_y = z_i + \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0, \boldsymbol{\sigma}_\epsilon^2 \mathbf{I}). \tag{3}$$

We assume that the prototype is consistent with all support instances, allowing us to express the likelihood of ν_y as a product distribution:

$$p(\nu_y|S_y) = \frac{\prod_{i \in S_y} p(\nu_y|x_i)}{\int_\nu \prod_{i \in S_y} p(\nu_y|x_i) d\nu}. \tag{4}$$

Because $p(\nu_y|x_i)$ is Gaussian, the resulting product is too:

$$\nu_y|S_y \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\sigma}_y^2 \mathbf{I}) \text{ with}$$

$$\boldsymbol{\sigma}_y^2 = \left(\sum_{i \in S_y} \hat{\boldsymbol{\sigma}}_{x_i}^{-2}\right)^{-1} \text{ and } \boldsymbol{\mu}_y = \boldsymbol{\sigma}_y^2 \circ \left(\sum_{i \in S_y} \hat{\boldsymbol{\sigma}}_{x_i}^{-2} \circ \boldsymbol{\mu}_{x_i}\right),$$

where $\hat{\boldsymbol{\sigma}}_{x_i}^2 = \boldsymbol{\sigma}_{x_i}^2 + \boldsymbol{\sigma}_\epsilon^2$ and $\circ$ denotes the Hadamard product. Essentially, the prototype is a confidence-weighted average of the support instances.

2.2. Prediction and approximate inference

We assume a softmax prediction for a query embedding, z:

$$p(y|z, S) \propto \mathcal{N}(z; \boldsymbol{\mu}_y, \hat{\boldsymbol{\sigma}}_y^2 \mathbf{I}) \tag{5}$$

with $\hat{\boldsymbol{\sigma}}_y^2 = \boldsymbol{\sigma}_y^2 + \boldsymbol{\sigma}_\epsilon^2$ as before, yielding the class posterior for query x:

$$p(y|x, S) = \int_z \mathcal{N}(z; \boldsymbol{\mu}_x, \boldsymbol{\sigma}_x^2 \mathbf{I}) \frac{\mathcal{N}(z; \boldsymbol{\mu}_y, \hat{\boldsymbol{\sigma}}_y^2 \mathbf{I})}{\sum_c \mathcal{N}(z; \boldsymbol{\mu}_c, \hat{\boldsymbol{\sigma}}_c^2 \mathbf{I})} dz. \tag{6}$$

The class distribution is equivalent to that produced by the deterministic PN as $\boldsymbol{\sigma}_x^2 \to \mathbf{0}$ when $\boldsymbol{\sigma}_y^2 = \boldsymbol{\sigma}_{y'}^2$ for all class pairs (y, y'). However, in the general case, the integral has no closed form solution; thus, we must sample to approximate $p(y|x, S)$, both for training and evaluation. We employ two samplers, which we refer to as *naïve* and *intersection*.

2.2.1. Naïve sampling

A direct approach to approximating the class posterior is to express Equation 2 as an expectation, $\mathbb{E}_{z \sim p(z|x)}[p(y|z, S)]$, and to replace the expectation with the average over a set of samples. We utilize the reparameterization trick of Kingma & Welling (2014) to train the model. Although this is the simplest approach, it is sample-inefficient during training, and when the number of samples is reduced, performance is impacted.

2.2.2. Intersection sampling

In Equation 6, the product of Gaussian densities in the numerator can be rewritten:

$$\begin{aligned} &\mathcal{N}\left(z; \boldsymbol{\mu}_x, \boldsymbol{\sigma}_x^2 \mathbf{I}\right) \mathcal{N}\left(z; \boldsymbol{\mu}_y, \hat{\boldsymbol{\sigma}}_y^2 \mathbf{I}\right) = \\ &\quad \mathcal{N}\left(z; \boldsymbol{\mu}_{xy}, \boldsymbol{\sigma}_{xy}^2 \mathbf{I}\right) \mathcal{N}\left(\boldsymbol{\mu}_x; \boldsymbol{\mu}_y, (\boldsymbol{\sigma}_x^2 + \hat{\boldsymbol{\sigma}}_y^2) \mathbf{I}\right), \end{aligned} \tag{7}$$

where

$$\begin{aligned} \boldsymbol{\sigma}_{xy}^2 &= (\boldsymbol{\sigma}_x^{-2} + \hat{\boldsymbol{\sigma}}_y^{-2})^{-1} \text{ and} \\ \boldsymbol{\mu}_{xy} &= \boldsymbol{\sigma}_{xy}^2 \circ (\boldsymbol{\sigma}_x^{-2} \circ \boldsymbol{\mu}_x + \hat{\boldsymbol{\sigma}}_y^{-2} \circ \boldsymbol{\mu}_y). \end{aligned}$$

Substituting Equation 7 into Equation 6,

$$\begin{aligned} p(y|x, S) = &\ \mathcal{N}\left(\boldsymbol{\mu}_x; \boldsymbol{\mu}_y, (\boldsymbol{\sigma}_x^2 + \hat{\boldsymbol{\sigma}}_y^2) \mathbf{I}\right) \times \\ &\mathbb{E}_{z \sim \mathcal{N}\left(\boldsymbol{\mu}_{xy}, \boldsymbol{\sigma}_{xy}^2 \mathbf{I}\right)} \left[\sum_c \mathcal{N}(z; \boldsymbol{\mu}_c, \hat{\boldsymbol{\sigma}}_c^2 \mathbf{I})\right]^{-1}. \end{aligned} \tag{8}$$

Approximating the expectation with samples from $\mathcal{N}\left(\boldsymbol{\mu}_{xy}, \boldsymbol{\sigma}_{xy}^2 \mathbf{I}\right)$, we obtain a sampler that focuses on the

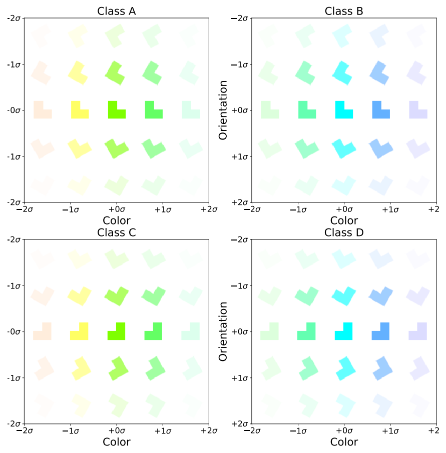

Figure 1. Samples from the four classes in our synthetic data set. In each plot, class means are shown at the center, along with samples spanning ± 2 standard deviations in both orientation and color. A sample's transparency is set according to its class-conditional likelihood. Both dimensions can be coded as directional variables. The class centroids on each dimension are 90° apart with standard deviation 22.5°.

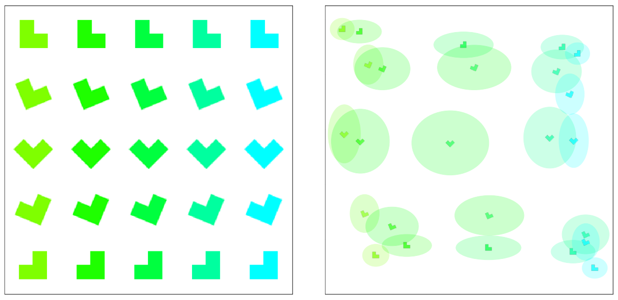

Figure 2. Synthetic data set: (left) A set of examples, with the four class centroids located in the corners and examples between formed via linear interpolation in the generative space. (right) The 2D stochastic prototype embedding for the examples. The shape is plotted at the mean of $p(z|x)$, and the outlines of the ovals represent equi-probability contours at 0.4 standard deviations.

intersection of the input distribution and a given class distribution. During training with a cross entropy loss, we need only sample for the known (target) class y. Compared to naïve sampling, we found this scheme to be more robust and significantly more sample efficient, requiring only a *single* sample to train effectively.

3. Experimental Results

We assess the SPE when trained on either the naïve or intersection sampler; in both cases, we evaluate using the naïve sampler with 200 samples. For details regarding network architectures and hyperparameters, see Appendix A, and for simulation details, including the choice of initialization for $\boldsymbol{\sigma}_\epsilon^2$, see Appendix B.

3.1. Synthetic color-orientation data set

Using a synthetic image data set, we demonstrate that the SPE can capture the generative structure of a domain and provide sensible estimates of uncertainty. The data set consists of 64×64 pixel images with a well-defined class structure. The four classes in our domain are distinguished by orientation, color, or both (Figure 1). Class-conditional generative parameters—orientation and color—are sampled from an isotropic Gaussian distribution. Class overlap on color and orientation dimensions is symmetric. We tuned the variance such that there would be significant overlap between classes in order to elicit embeddings with increased uncertainty in overlapping regions. Full details of the synthetic data set can be found in Appendix A.2.

We trained a two-dimensional, intersection-sampling SPE on samples from this domain, using two instances per class to form prototypes. Classification accuracy of held-out samples is approximately 86%. Accounting for class overlap, a Bayes optimal classifier has an accuracy of approximately 87%. For visualization, we selected 25 examples by interpolating between the class means. The examples and their embeddings are shown in the left and right panels of Figure 2. The network has captured the structure of the domain by disentangling the two factors of variation. To identify the input array and the embedding space, the input array must be mirrored along the horizontal axis. The embedding variance encodes label ambiguity. Instances half way between two classes on one dimension have maximal variance along that dimension. Label ambiguity is one type of uncertainty. An equally important source of uncertainty comes from noisy or out-of-distribution (OOD) inputs. We examined OOD inputs generated in two different ways. In the left panel of Figure 3, we show the consequence of adding pixel hue noise to the four class centroids. Only one of these centroids is shown along the abscissa, but all four are used to make the graph, with many samples per noise level. The grey and black bars in the graph indicate variance on the vertical and horizontal dimensions of the embedding space, respectively. As pixel hue noise increases, uncertainty in the color grows but uncertainty in the orientation does not. In the right panel of Figure 3, we show the consequence of shortening the length of the legs of the shape. Shortening the legs removes cues that can be used both for determining both color and orientation. As a result, the uncertainty grows on both dimensions.

3.2. Omniglot

The Omniglot data set contains images of labeled, handwritten characters from diverse alphabets. Omniglot is one of the standard data sets for comparing methods in the few-shot learning literature. The data set contains 1623 unique characters, each with 20 instances. Following Snell et al. (2017),

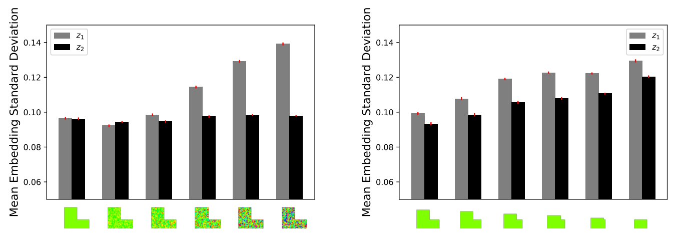

Figure 3. Synthetic data set: (left) uncertainty on the two embedding dimensions as the hue becomes more difficult to discern; (right) uncertainty on the two embedding dimensions as the orientation becomes more difficult to discern.

Table 1. Test classification accuracy (%) on Omniglot with both 2D and 64D embeddings comparing the PN and the SPE with intersection sampling (1 sample per trial). Performance for PN and SPE is a mean over 1000 random test episodes, showing ±1 standard error of the mean.

2D	1-shot, 5-class	5-shot, 5-class	1-shot, 20-class	5-shot, 20-class
PN	75.7 ± 0.4	$\mathbf{82.6 \pm 0.3}$	45.0 ± 0.2	$\mathbf{55.9 \pm 0.2}$
SPE	$\mathbf{76.9 \pm 0.4}$	$\mathbf{82.3 \pm 0.3}$	$\mathbf{49.7 \pm 0.2}$	55.3 ± 0.2
64D	1-shot, 5-class	5-shot, 5-class	1-shot, 20-class	5-shot, 20-class
PN	$\mathbf{98.46 \pm 0.09}$	$\mathbf{99.55 \pm 0.03}$	$\mathbf{94.87 \pm 0.08}$	$\mathbf{98.63 \pm 0.03}$
SPE	$\mathbf{98.53 \pm 0.08}$	$\mathbf{99.52 \pm 0.03}$	$\mathbf{94.93 \pm 0.08}$	98.56 ± 0.02

we augment the original classes with all 90° rotations, resulting in 6492 total classes, and we use the same train, validation, and test splits. Each grayscale image is resized from 32×32 to 28×28. We trained and evaluated deterministic Prototypical Networks, naïve-sampling SPEs, and intersection-sampling SPEs. Each is trained episodically, where a training episode contains 60 randomly sampled classes and 5 query instances per class, and test episodes contain 15 query instances per class.

We first compare the effectiveness of using naïve and intersection samplers during SPE training of Omniglot. We vary both the sampler and the number of samples drawn per training query, denoted by s. We evaluate in a 1-shot 20-class setting, where shot refers to the number of support examples used to compute a prototype, and class refers to the number of candidate test classes per episode. Figure 4 shows test classification accuracy as the number of samples drawn per training trial (s) increases. As we previously claimed, the intersection-sampling SPE is much more sample-efficient and systematically outperforms the naive-sampling SPE. Furthermore, the intersection-sampling SPE is more robust and less susceptible to poor initialization or uninformative samples.

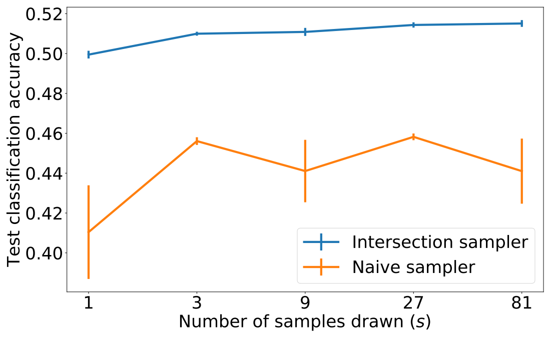

Figure 4. Test classification accuracy as a function of number of training samples per query instance for a naïve-sampling and intersection-sampling 2D SPE on a 1-shot, 20-class Omniglot task. Performance is a mean over 5 replications of running the model, showing ±1 standard error of the mean.

Table 1 presents results for simulations with Omniglot comparing the deterministic PN to the intersection-sampling SPE. For both 2- and 64-dimension embeddings, the SPE either matches or outperforms the PN, a state-of-the-art algorithm in few-shot learning. In the 64D embedding, both methods are close to ceiling. In the 2D embedding, the SPE particularly shines in the 1-shot tests, perhaps because, relative to the 5-shot tests, performance is further from ceiling. However, another possibility is that the role of uncertainty is greater when only one instance is used to form the prototype. When 5 instances are used, the uncertainty shrinks significantly, causing the SPE to behave more like its deterministic sibling. To emphasize, the improved performance of the SPE does not incur much of a computational cost in training. And by providing an estimate of uncertainty associated with embedded instances, the SPE offers the possibility of detecting OOD samples and informing downstream systems that operate on the embedding.

4. Conclusion

We propose a Stochastic Prototype Embedding (SPE) and demonstrate its effectiveness in representing uncertainty related to both input and label noise on a synthetic data set. Furthermore, we show that the SPE consistently outperforms its deterministic sibling, the Prototypical Network (PN), at approximately the same space and time cost.

References

Chopra, S., Hadsell, R., and LeCun, Y. Learning a Similarity Metric Discriminatively, with Application to Face Verification. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2005.

Edwards, H. and Storkey, A. Towards a Neural Statistician. In *International Conference on Learning Representations*, 2017.

Finn, C., Abbeel, P., and Levine, S. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1126–1135, 06–11 Aug 2017.

Hadsell, R., Chopra, S., and LeCun, Y. Dimensionality Reduction by Learning an Invariant Mapping. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pp. 1735–1742, 2006. doi: 10.1109/CVPR.2006.100.

Karaletsos, T., Belongie, S., and Rätsch, G. Bayesian Representation Learning with Oracle Constraints. In *International Conference on Learning Representations*, 2016.

Kingma, D. P. and Welling, M. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

Koch, G., Zemel, R., and Salakhutdinov, R. Siamese Neural Networks for One-Shot Image Recognition. In *ICML Deep Learning Workshop*, volume 2, 2015.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A Simple Neural Attentive Meta-Learner. In *International Conference on Learning Representations*, 2018.

Oh, S. J., Gallagher, A. C., Murphy, K. P., Schroff, F., Pan, J., and Roth, J. Modeling Uncertainty with Hedged Instance Embeddings. In *International Conference on Learning Representations*, 2019.

Ridgeway, K. and Mozer, M. C. Learning Deep Disentangled Embeddings With the F-Statistic Loss. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 185–194. Curran Associates, Inc., 2018.

Rippel, O., Paluri, M., Dollar, P., and Bourdev, L. Metric Learning with Adaptive Density Discrimination. In *International Conference on Learning Representations*, 2016.

Schroff, F., Kalenichenko, D., and Philbin, J. FaceNet: A Unified Embedding for Face Recognition and Clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Scott, T., Ridgeway, K., and Mozer, M. C. Adapted Deep Embeddings: A Synthesis of Methods for k-Shot Inductive Transfer Learning. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 31*, pp. 76–85. Curran Associates, Inc., 2018.

Snell, J., Swersky, K., and Zemel, R. Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems 31*, pp. 4077–4087, 2017.

Song, H. O., Xiang, Y., Jegelka, S., and Savarese, S. Deep Metric Learning via Lifted Structured Feature Embedding. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Song, H. O., Jegelka, S., Rathod, V., and Murphy, K. Deep Metric Learning via Facility Location. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2206–2214, July 2017. doi: 10.1109/CVPR.2017.237.

Triantafillou, E., Zemel, R., and Urtasun, R. Few-Shot Learning Through an Information Retrieval Lens. In *Advances in Neural Information Processing Systems 31*, pp. 2255–2265, 2017.

Ustinova, E. and Lempitsky, V. Learning Deep Embeddings with Histogram Loss. In *Advances in Neural Information Processing Systems 30*, pp. 4170–4178, 2016.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., and Wierstra, D. Matching Networks for One Shot Learning. In *Advances in Neural Information Processing Systems 30*, pp. 3630–3638, 2016.

Wang, J., Zhou, F., Wen, S., Liu, X., and Lin, Y. Deep Metric Learning with Angular Loss. In *IEEE International Conference on Computer Vision*, pp. 2612–2620, 2017. doi: 10.1109/ICCV.2017.283.

Yi, D., Lei, Z., Liao, S., and Li, S. Z. Deep Metric Learning for Person Re-identification. In *International Conference on Pattern Recognition*, pp. 34–39, 2014. doi: 10.1109/ICPR.2014.16.

A. Network Architectures and Hyperparameters

A.1. Omniglot

For all Omniglot experiments, the network consisted of four convolutional blocks. The first three blocks had a convolutional layer with 64 filters, a 3×3 kernel, zero-padding of length 1, and a stride of 1, followed by a batch normalization layer, ReLU activation, and 2×2 max-pooling. The fourth and final block had a convolutional layer with $2d$ filters, a 3×3 kernel, zero-padding of length 1, and a stride of 1, followed by 2×2 max-pooling, where d represents the dimensionality of the embedding space. The flattened output of the network is a vector of length $2d$, where the first d elements were considered the mean of the Gaussian distribution and the remaining d elements were the diagonal covariance entries. The weights were initialized using He initialization and the biases with the following uniform distribution: $\mathcal{U}(-\frac{1}{\sqrt{\text{fan in}}}, \frac{1}{\sqrt{\text{fan in}}})$.

All Omniglot models were trained with an initial learning rate of 0.001 which was cut in half every 50 epochs. The models were stopped early using a patience parameter when performance on the validation set no longer increased.

A.2. Synthetic data

The images in the synthetic data set are 64×64 pixels in size. For orientation, we chose class centers at $90°$ and $180°$, with a standard deviation of $30°$. For color, we manipulated the hue and kept value and saturation constant. Like orientation, hue is a circular quantity. If hue ranges from 0 to 360 degrees, we chose color class centers and standard deviation in the same way as orientation. Additionally, we add noise to a minority (15%) of the images used to train the model. For these, we add Gaussian noise to the hue of each pixel inside the shape. The standard deviation of the hue noise was chosen uniformly between $18°$ and $54°$. We also added noise to the leg lengths of the L shapes. The leg length was chosen uniformly between 10% and 98% of its original length. See Figure 3 for some examples.

The network followed an architecture similar to the one we used for Omniglot, except that we added two additional blocks of convolution, batch normalization, ReLU, and max-pooling because the images are larger. We used 2 instances per class to form prototypes and 8 samples per query instance during training. We used a learning rate of 0.0001 and the models were stopped early using a patience parameter when performance on the validation set no longer increased.

B. Simulation Details

For all SPE models,

$$\boldsymbol{\sigma}_{\epsilon}^{2} = \text{softplus}\left(\boldsymbol{\epsilon}\right),$$

where $\boldsymbol{\epsilon}$ is a trainable parameter. We initialize $\boldsymbol{\epsilon}$ using the following prescription:

$$\boldsymbol{\epsilon} = |S| \boldsymbol{\epsilon}_0^{2/d},$$

where $|S|$ is the number of support examples per episode during training and d is the dimensionality of the embedding. We chose this prescription for two reasons: (1) as the number of support examples increases, the variance of the prototype distribution approaches zero, so scaling linearly by $|S|$ tends to provide a stronger training signal early on, and (2) the amount of noise in the projection of an embedding should scale with the dimensionality of the embedding space as to maintain unit-volume. All models used $\boldsymbol{\epsilon}_0 = 0.01$.

It should also be noted that there is no constraint for $\boldsymbol{\sigma}_x^2 > 0$, since $\boldsymbol{\sigma}_x^2$ is a direct output of a deep network. To ensure a positive value, $\boldsymbol{\sigma}_x^2 \leftarrow \text{softplus}(\boldsymbol{\sigma}_x^2)$.

C. SPE Variants

We assumed only diagonal covariance matrices in this work. Switching to a full covariance matrix would require matrix inversion, which is ordinarily infeasible, but because one purpose of deep embeddings is visualization, there may be interesting cases involving 2D embeddings where the cost of inversion is trivial.