Prototypical Bregman Networks

Kubra Cilingir [1] **Brian Kulis** [1]

Abstract

In this work, we approach one-shot and few-shot learning problems as methods for finding good prototypes for each class, where these prototypes are generalizable to new data samples and classes. We propose a metric learner that learns a Bregman divergence by learning its underlying convex function. Bregman divergences are a good candidate for this framework given they are the only class of divergences with the property that the best representative of a set of points is given by its mean. We propose a flexible extension to prototypical networks to enable joint learning of the embedding and the divergence, while preserving computational efficiency. Our preliminary results are comparable with the prior work on the Omniglot and Mini-imagenet datasets, two standard benchmarks for one-shot and few-shot learning. We argue that our model can be used for other tasks that involve metric learning or tasks that require approximate convexity such as structured prediction and data completion.

1. Introduction

Deep learning methods have shown tremendous performance on many tasks involving large-scale data. However, collecting large amounts of data is costly or even infeasible for many applications (Kaiser et al., 2017; Altae-Tran et al., 2017). The few-shot learning problem aims to achieve good performance on adapting to novel classes where only a small number of examples per novel class are available. In scenarios with few examples, classical classification, fine-tuning, or retraining methods fail due to severe overfitting, catastrophic forgetting, or inflexibility to adapt to new samples and categories (Fink, 2005).

This problem has been of increasing interest to researchers, some of whom have been inspired by humans' ability to recognize novel classes very successfully with very few examples. The most recent approaches to solve the few-shot learning problem involve meta learning, which attempts to learn transferable knowledge between classes and tasks at training time, in order to help generalization and adaptivity at test time. Information is stored either in the initialization of the weights (Finn et al., 2017), in a recurrent memory unit (Santoro et al., 2016), in the optimization strategy (Ravi & Larochelle, 2016), or in an embedded space (Snell et al., 2017). In this work, we focus on the last approach due to its simplicity and compelling results, whereas the other methods require complex training mechanisms, complex inference, or the gathering of many similar tasks.

In particular, we based our approach on prototype networks (Snell et al., 2017), which learn an embedding of the input data, and then construct prototypes for classes via averages or weighted averages over points in each class. A single vector representation per class is assumed to be sufficient to contain class-specific features (Rosch, 1973). In (Snell et al., 2017), the Euclidean distance is used to measure distance between a query point and a class prototype. In contrast to existing work, we treat the problem as a joint embedding and metric learning problem. Because prototypes are typically represented by means of points, for the metric learning function we choose to learn a *Bregman divergence* as the underlying divergence. This class of divergences has the key property that the best representative of a set of points (in terms of the sum of divergences between the points and the representative) is given by the mean, which we argue makes it appropriate for constructing prototypes of classes for our problem.

Compared to existing methods such as relation networks (Sung et al., 2018), ours is a more flexible approach since we focus on Bregman divergences, of which squared Euclidean distances are a special case. We may favor Bregman divergences over Euclidean distances since symmetry and the triangle inequality may not be necessary for data in a few-shot learning problem. In Figure 1 we show a possible scenario to demonstrate this. Suppose image A and image B have the same shape, and image B and image C have the same color. Image A and image C need not be similar, but the triangle inequality forces a resemblance between A and C. Similarly, representations and similarity measures of class members' may not be desired to be symmetric. Proto-

[1]Boston University, MA, USA. Correspondence to: Kubra Cilingir <kubra@bu.edu>, Brian Kulis <bkulis@bu.edu>.

Proceedings of the 1st Adaptive & Multitask Learning Workshop, Long Beach, California, 2019.

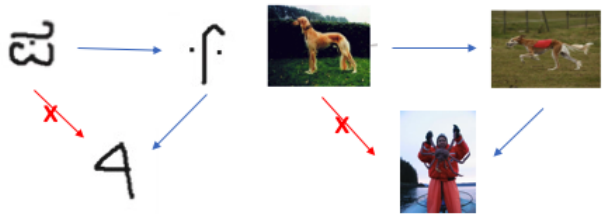

Figure 1. The triangle inequality may fail to represent relations between samples. Images are from 3 different categories from Omniglot and mini Imagenet. See text for details.

types share the abstract representative features with the data points, but each point has idiosyncratic features, that may break symmetry when interpreting the embedding space.

Each Bregman divergence is parametrized by a convex function; furthermore, this relationship is a surjection. We design the metric learning function of our deep learning model with a convexity constraint with respect to the embedding space. This convex function is used to calculate the Bregman divergence as a learned metric. Our formulation also provides flexibility for the architectural design of the convex function, with a regularization term to improve generalizability. We empirically measure a convexity score by drawing random points from the convex hull of the data samples to verify our claim.

Overall, we propose a model that has a learnable embedding and a learnable Bregman divergence that can be trained simultaneously. Compared with the state-of-art, our initial results are promising. Other than improving the results with the current model and testing on other datasets, our preliminary work has two clear future directions:

(i) Taking our convex framework to other sets of problems such as semi-supervised learning, similarity learning, structured prediction, etc.

(ii) Following different approaches to satisfy and measure convexity such as modifying the constraint or the optimization algorithm itself.

2. Related Work

Few shot learning methods have received increasing interest given the recent success of discriminative models. Many of these effective methods fall under meta learning, where these approaches store transferable information in different ways to remedy overfitting issues and provide adaptivity for new samples and classes.

Meta learning by initialization: The most well-known method under this category is MAML (Finn et al., 2017). These methods attempt to learn an initialization over the weights, such that a similar few shot learning problem can be adapted by fine-tuning (Gidaris & Komodakis, 2018), (Mishra et al., 2017). Many related target tasks are employed in order to train a model which can later be fine-tuned for each task. The need for fine-tuning and many tasks limits the efficiency of these methods.

Meta learning by a recurrent memory: In these methods, such as MANN (Santoro et al., 2016), Meta Nets (Munkhdalai & Yu, 2017), and Memory Matching Networks (Cai et al., 2018), the useful knowledge required to solve the tasks are stored in a recurrent manner using a memory unit. Existing information in the memory and new information is compared to update the model and perform the task. However, this type of algorithm suffers from inherent issues in RNNs such as instability and difficulty in properly storing long-term dependency.

Meta learning by optimizer: This category of methods aims at training an optimizer to provide gradients and to be used in fine tuning. The LSTM-based optimizer (Ravi & Larochelle, 2016) is an example of this approach. These methods also require fine-tuning while currently-proposed optimizers add unnecessary complexity to the training phase compared to the benefit for their performance.

Meta learning by embedding: These approaches are based on metric learning methods that aim to learn an embedded space to store the relations between the samples and classes. A comprehensive overview on metric learning can be found in (Kulis, 2013). The task is then performed with a classifier that uses the embedding and a fixed metric. This type of method is free from the various complexity issues that the other approaches have. Siamese Networks (Koch et al., 2015), Matching Networks (Vinyals et al., 2016), Relation Nets (Sung et al., 2018), and Prototypical Nets (Snell et al., 2017) are examples from this class which efficiently represent each class by its mean. Our work resembles this setting, with the distinction that our work jointly learns the embedding and the metric from a set of distance families called Bregman divergences.

Data Augmentation methods: Another approach for few-shot learning is to leverage additional data to eliminate overfitting issues (Antoniou & Storkey, 2019), (Wang et al., 2018). Learned transformations are applied or new data is generated to enrich prior information. We will not consider this approach separately since they can be combined with the above methods.

3. Proposed Method

We now define our problem setup and notations. Assume we are given a set of examples coming from N classes. The training data is split into a training set, and a separate validation set V. Both sets are divided into a support set

S with K samples per class and a query set Q with the remaining samples for each class. The tasks are defined as N-way K-shot learning, which corresponds to having K samples for each one of N classes in S. We consider the cases where K is 1 or 5 and N is 5 for our experiments.

We refer to the embedding function of our model as f_{w_f}, and the subsequent convexified layers as ϕ_{w_ϕ} where w_f and w_ϕ are the trainable weights for these functions, respectively. For simplicity we drop the weight variables from the notation, and continue with f and ϕ. We use x' and x'' to represent a random pair from the space of interest.

3.1. Bregman Divergence

Before diving into details of our model, we briefly review Bregman divergences and their properties to better clarify our choice for the class of divergences. Bregman divergences are derived from a strictly convex and differentiable function, denoted as ϕ. The Bregman divergence between two points x' and x'' is defined as:

$$D_\phi(x', x'') = \phi(x') - \phi(x'') - \nabla\phi(x'')^T(x' - x'') \quad (1)$$

Bregman divergences are not metrics, but they satisfy sufficient properties for our problem, namely non-negativity and having a unique 0. We previously discussed why symmetry and the triangle inequality may not be desired for our setting. Since we are exploring the case where we represent classes as their means in the embedded space, our choice of distance should have the mean as the minimizer for the distances within a class.

Mean Minimization Property: Assume we have an n-dimensional random variable X defined on a convex set $\Omega \in R^n$. $d : R^n \times R^n \to R$ is a continuous function and $d(X, X) \geq 0$ with continuous derivatives. It is known that $E[d(X, \bar{x})] \leq E[d(X, x')]$ for all $x' \in \Omega$ where $\bar{x} = E[X]$, if, and only if $d \in D_\phi$ where ϕ is the corresponding strictly convex function (Frigyik et al., 2008).

For the cases where all samples are not available, or if the distribution is discrete, the expectation can simply be replaced by the sample averages. Assume we have M observed samples $\{x_i\}_{i=1}^M$. The inequality can be rewritten as the following:

$\sum_i d(x_i, \bar{x}) \leq \sum_{x\in\Omega} d(x_i, x')$ for all $x' \in \Omega$ where $\bar{x} = \frac{1}{M}\sum_i x_i$, if and only if $d \in D_\phi$.

Thus, given a set of points, the best representative is given by the mean, under any Bregman divergence. Each Bregman divergence is identified by a ϕ such as Euclidean distance is a Bregman divergence identified with $\phi = x^2$.

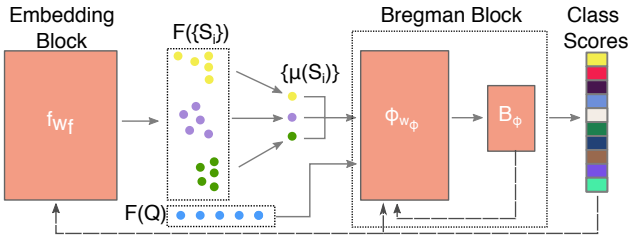

Figure 2. Our model architecture for few shot learning. $F(\{S_i\})$ are the class embedded clusters and $F(Q)$ are the query points. $\{\mu(S_i)\}$ are the calculated class means in the embedded space. ϕ_{w_ϕ} is a convex function and B_ϕ represents Bregman loss.

3.2. Model

We overview our model in Figure 2. Our model consists of two learnable functions: (i) the embedding function f, and (ii) the metric learning function ϕ. The embedding function f brings in non-linearity to the framework, which provides flexibility when integrating with the metric learning function. The metric learning function is a neural network ϕ that is trained via a convexity constraint to output a convex function with respect to the embedded features $f(x_i)$, followed by Bregman divergence using the learned ϕ. We impose convexity by using midpoint convexity characterization with the continuity of f, which is equivalent to the standard convexity definition.

Midpoint Convexity: ϕ is midpoint convex if and only if $C_\phi(x', x'') := \phi(x') + \phi(x'') - 2\phi\left(\frac{x'+x''}{2}\right) \geq 0$ for all x',$x'' \in \Omega$. If ϕ is continuous and satisfies midpoint convexity, ϕ is convex (Jensen, 1905).

At a high level the problem can be expressed as:

$$\begin{aligned} &\min L_p(D_\phi(f(x), \mu_n^f), y) \\ &\text{subject to} \\ &\textstyle\int_{x',x''\in\Omega}[1_{C_\phi(f(x'),f(x''))\leq 0}]d\Omega = 0, \end{aligned} \quad (2)$$

where μ_n^f represents the n^{th} class's mean on the embedded space induced by f. x and y represent the sample and target pairs coming from T. L_p represents a classification loss, e.g., cross entropy loss or mean squared error loss.

Midpoint convexity implies that for any input pairs, the midpoint value of the function is not greater than the function value of the midpoint of the pairs. The integral turns into a summation since we have a finite number of samples. This definition naturally integrates to our framework without additional significant computation since pairs are already used to determine the similarities. We obtain an approximately convex network by feeding a sufficient number of samples. We reformulate the indicator function with a clamping function to impose the convexity inequality, which penalizes any pair of points that violates the midpoint convexity constraint.

This formulation also gives flexibility in architectural design for the convex function.

We use a regularization term in order to further control overfitting and convexity in hard tasks by controlling the gradient change, i.e, the Lipschitz constant of the convex function.

$$L_r = \frac{||\nabla_{f(x)}\phi(f(x'')) - \nabla_{f(x)}\phi(f(x'))||}{||x'' - x'||} \text{ for all } x', x'' \in \Omega \tag{3}$$

We combine L_p loss and L_r loss with a weighting term, and relax the LHS term in the constraint inequality in 2, denoted as L_c, to train our model. We follow a joint training approach; however, it is of value to note that alternating training between the embedding function and the metric function would be another option suitable for our setting.

4. Experiments

We applied our model on two commonly used datasets for our experiments: Omniglot and mini-Imagenet.

The Omniglot dataset consists of 1,623 handwritten letters coming from 60 different alphabets. Each letter image has dimensions 28x28 and has 20 samples. We applied rotations to increase the number of classes. The data is then divided into training, validation and test sets with 4112, 688 and 423 classes similar to a previous approach (Snell et al., 2017). The Mini-Imagenet dataset is derived from ILSVRC12 (Russakovsky et al., 2015). It contains 1000 classes with 600 84x84 images each. We split the data into training, validation and test cases by using a standard method proposed in (Ravi & Larochelle, 2016).

We choose our model architecture to be comparable with existing methods. We use 3 convnet blocks for the embedding function, where each block consists of a 3x3 convolutional layer followed by a batch normalization, ReLU and pooling layers. Our embedding layer is 128 dimensional for the Omniglot and 512 dimensional for the mini Imagenet. Our convex network also contains 2 fully connected layers with sigmoid activations. The fully connected layers followed by a linear layer output a scalar for each input pair. Then the Bregman divergence term is calculated to classify samples according to their distance to the class means. It is worth mentioning that different kinds of layers other than fully-connected layers for the metric function are adaptable to our scheme.

We test our model and compare with existing algorithms for 1-way 5-shot and 5-way 5-shot problems. We use meta-validation set to determine the best model to use in the test case. The accuracies for each task are given in Tables 1 and 2. Despite the fact that these are our preliminary results, they are comparable with the previous models.

We define a way to measure the convexity we achieve for ϕ.

Table 1. Accuracy results for Omniglot Dataset

MODELS	5-WAY 1-SHOT	5-WAY 5-SHOT
SIAMESE NETS	96.7	98.4
MATCHING NETS	98.1	98.9
META NETS	99.0	-
MAML	98.7	99.9
PROTOTYPICAL NETS	98.5	99.6
RELATION NETS	99.6	99.8
PROTOTYPICAL-BREGMAN NETS	99.0	99.7

Table 2. Accuracy results for Mini-ImageNet Dataset

MODELS	5-WAY 1-SHOT	5-WAY 5-SHOT
MATCHING NETS	43.5	55.3
META NETS	49.2	-
MAML	48.7	63.11
PROTOTYPICAL NETS	49.4	68.2
RELATION NETS	50.4	65.3
PROTOTYPICAL-BREGMAN NETS	49.8	58.9

We select random pairs and divide the line connecting the pairs into 100 segments with 101 points. We then re-sample new pairs from these points and record how much L_c deviates from the convexity constraint. We take the overall average deviation of all pairs. Our results for Omniglot dataset are: (i) For training, we obtain $8,4x10^{-8} \pm 2x10^{-8}$. (ii) For testing, we obtain $6,7x10^{-3} \pm 5x10^{-1}$. We will run more tests for the convexity measure and analyze its behavior under different problem configurations and various modifications to the network, e.g., sensitivity to hyperparameters and different architectures.

5. Conclusion and Future Work

In this paper we proposed an alternative method for few shot learning. Our method is based on two jointly learnable functions: a nonlinear embedding function followed by a metric learning function. Our metric function learns a suitable Bregman divergence via approximating a convex function, with a motivation that using Bregman divergences allows the mean to be the most representative point for the relevant class. We achieve comparable preliminary results sufficient to validate their potential.

We plan to further investigate our model and do more extensive parameter and architecture search to improve our results. We can utilize different constraints or optimization methods to satisfy convexity, or apply an alternating training between embedding and metric function. We can also carry our method to other problems that contains convexity such as semi-supervised learning and structured prediction.

References

Altae-Tran, H., Ramsundar, B., Pappu, A. S., and Pande, V. Low data drug discovery with one-shot learning. *ACS central science*, 3(4):283–293, 2017.

Antoniou, A. and Storkey, A. Assume, augment and learn: Unsupervised few-shot meta-learning via random labels and data augmentation. *arXiv preprint arXiv:1902.09884*, 2019.

Cai, Q., Pan, Y., Yao, T., Yan, C., and Mei, T. Memory matching networks for one-shot image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4080–4088, 2018.

Fink, M. Object classification from a single example utilizing class relevance metrics. In *Advances in neural information processing systems*, pp. 449–456, 2005.

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

Frigyik, B. A., Srivastava, S., and Gupta, M. R. Functional bregman divergence and bayesian estimation of distributions. *IEEE Transactions on Information Theory*, 54(11): 5130–5139, 2008.

Gidaris, S. and Komodakis, N. Dynamic few-shot visual learning without forgetting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4367–4375, 2018.

Jensen, J. L. W. V. Om konvekse funktioner og uligheder imellem middelvaerdier. *Nyt tidsskrift for matematik*, 16: 49–68, 1905.

Kaiser, Ł., Nachum, O., Roy, A., and Bengio, S. Learning to remember rare events. *arXiv preprint arXiv:1703.03129*, 2017.

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

Kulis, B. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013.

Mishra, N., Rohaninejad, M., Chen, X., and Abbeel, P. A simple neural attentive meta-learner. *arXiv preprint arXiv:1707.03141*, 2017.

Munkhdalai, T. and Yu, H. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2554–2563. JMLR. org, 2017.

Ravi, S. and Larochelle, H. Optimization as a model for few-shot learning. 2016.

Rosch, E. H. Natural categories. *Cognitive psychology*, 4 (3):328–350, 1973.

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

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., and Lillicrap, T. One-shot learning with memory-augmented neural networks. *arXiv preprint arXiv:1605.06065*, 2016.

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

Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P. H., and Hospedales, T. M. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1199–1208, 2018.

Vinyals, O., Blundell, C., Lillicrap, T., and Wierstra, D. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Wang, Y.-X., Girshick, R., Hebert, M., and Hariharan, B. Low-shot learning from imaginary data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7278–7286, 2018.