Appendix:
ProtoDiff: Learning to Learn Prototypical Networks by Task-Guided Diffusion

Anonymous Author(s)
Affiliation
Address
`email`

Contents

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

1 Algorithms

We describe the detailed algorithms for meta-training and meta-test of ProtoDiff as following Algorithm 1 and 2, respectively:

Algorithm 1 Meta-training phase of ProtoDiff.
Input: $p(\mathcal{T})$: distribution over tasks; θ: diffusion parameters; ϕ: feature extractor parameters; T: diffusion steps.
Output: $\mathbf{z}_\theta$: diffusion network, f_ϕ: feature extractor.

```
repeat
    Sample batch of tasks $\mathcal{T}^i \sim p(\mathcal{T})$
    for all $\mathcal{T}^i$ do
        Sample support and query set $\{\mathcal{S}^i, \mathcal{Q}^i\}$ from $\mathcal{T}^i$
        Compute the vanilla prototype $\tilde{\mathbf{z}}^i = f(\mathcal{S}^i)$
        repeat
            Take some gradient step on $\nabla\mathcal{L}_{\mathrm{CE}}(\mathcal{S}^i, \mathcal{Q}^i)$, updating $\phi^i$
        until $\mathcal{L}_{\mathrm{CE}}(\mathcal{S}_i, \mathcal{Q}_i)$ converges
        Compute the overfitted prototype by using the updated $\phi^i$, $\mathbf{z}^{i,*} = f_{\phi^i}(\mathcal{S}_i)$
        $t \sim Uniform(1...T)$
        $\epsilon = \mathcal{N}(\mathbf{0}, \mathbf{1})$
        $\beta_t = \frac{10^{-4}(T-t)+10^{-2}(t-1)}{T-1}$,$\alpha_t = 1-\beta_t$, $\bar{\alpha}_t = \Pi_{k=0}^{k=t}\alpha_k$
        $\hat{\mathbf{z}}_t^i = \sqrt{\bar{\alpha}_t}(\mathbf{z}^{i,*} - \tilde{\mathbf{z}}^i) + \sqrt{1-\bar{\alpha}_t{}^2}\epsilon$
        Compute diffusion loss $\mathcal{L}_{\text{diff}}$ with Equation (9) and the final loss $\mathcal{L}_{\mathcal{T}^i}$ with Equation (10)
        Update $\{\phi, \theta\} \leftarrow \{\phi, \theta\} - \beta\nabla_{\{\phi,\theta\}} \sum_{\mathcal{T}^i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}^i}$ using query data of each task.
    end for
until $f_\phi$ and $\mathbf{z}_\theta$ converges
```

Algorithm 2 Meta-test phase of ProtoDiff.
Input: $\tau = \{\mathcal{S}, \mathcal{Q}\}$: meta-test task, $\mathbf{z}_\theta$: trained diffusion network parameters, f_ϕ: trained feature extractor network parameters, T: diffusion steps.

```
$\mathbf{z}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
Compute vanilla prototype $\tilde{\mathbf{z}}$ with support set $f(\mathcal{S})$
for t=$T, \cdots, 1$ do
    $\epsilon = N(\mathbf{0}, \mathbf{1})$
    $\beta_t = \frac{10^{-4}(T-t)+10^{-2}(t-1)}{T-1}$,$\alpha_t = 1-\beta_t$, $\bar{\alpha}_t = \Pi_{k=0}^{k=t}\alpha_k$
    $\mathbf{z}_{t-1} = \mathbf{z}_\theta(\mathbf{z}, \tilde{\mathbf{z}}, t)$
end for
Compute the final prediction $\mathbf{y}^q$ by with Equation (1) based on $\mathbf{z}_0 + \tilde{\mathbf{z}}$
```

2 Per-task prototype overfitting architecture

To enhance our comprehension of the Per-task prototype overfitting part, we propose a succinct architectural representation depicted in Figure 1. The initial step entails the computation of the conventional prototypes $\tilde{\mathbf{z}}$ for a meta-training task. Subsequently, Equation (1) is employed to calculate the predictions for the query sample. The backbone's parameters are subsequently updated through I iterations. Through the utilization of parameters from the final iteration, we ultimately obtain the prototypes $\mathbf{z}^*$ that exhibit overfitting characteristics.

3 Residual prototype learning architecture

In order to gain a more comprehensive understanding of our residual prototype learning, we have crafted a succinct architecture diagram illustrated in Figure 2. Our proposition involves the prediction of the prototype update, denoted as $\mathbf{z}^* - \tilde{\mathbf{z}}$, instead of directly predicting the overfitted prototype

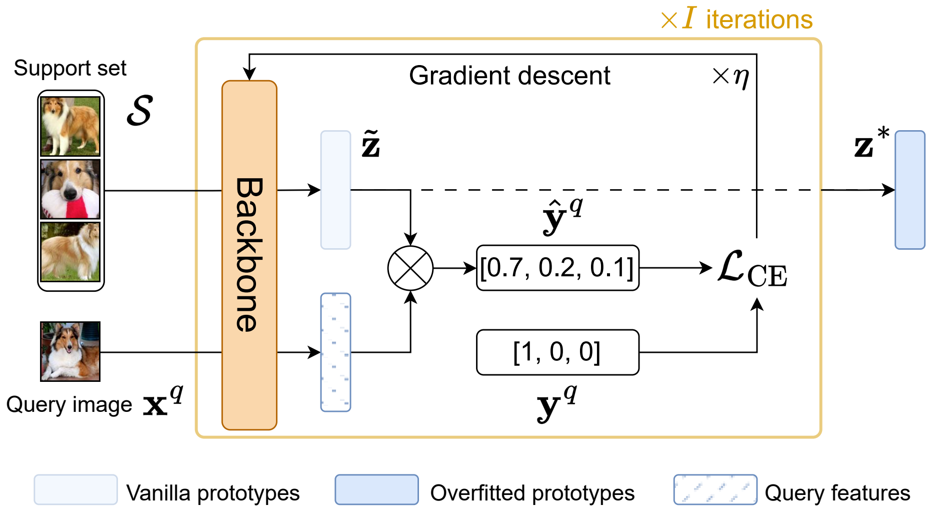

Figure 1: **Per-task prototype overfitting.**

$\mathbf{z}^*$. This distinctive approach also allows us to initialize ProtoDiff with the capability to perform the identity function, achieved by assigning zero weights to the decoder. Notably, we have discovered that the amalgamation of a global residual connection and the identity initialization substantially expedites the training process. By harnessing this mechanism, we have successfully enhanced the performance of ProtoDiff in the context of few-shot learning tasks.

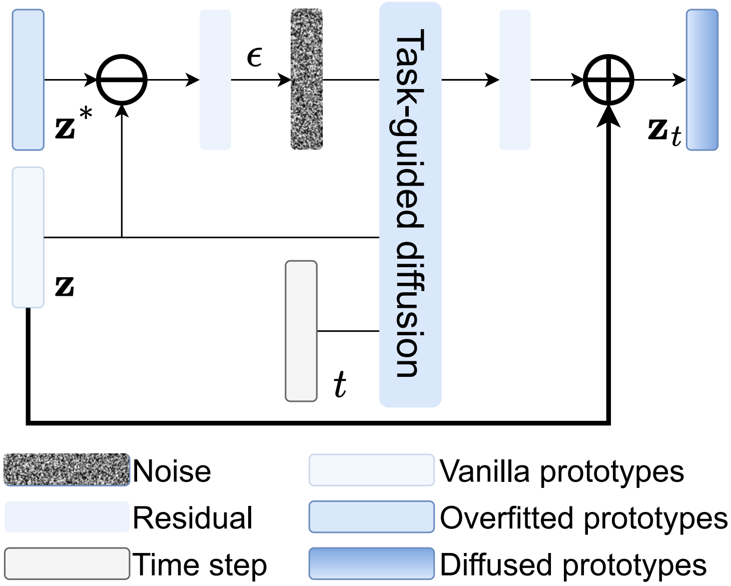

Figure 2: **Residual prototype learning.**

4 Datasets

Within-domain few-shot. For this setting we focus on 5-way 1-shot/5-shot tasks, which aligns with previous research [11]. The within-domain few-shot experiments are performed on three datasets: *mini*Imagenet [13] *tiered*Imagenet [8], and ImageNet-800 [2]. *mini*Imagenet consists of 100 randomly selected classes from ILSVRC-2012 [9], while *tiered*Imagenet is composed of 608 classes that are grouped into 34 high-level categories. We measure the accuracy of 600 tasks randomly sampled from the meta-test set to evaluate the performance. Following [2], we also evaluate our model on ImageNet-800, a dataset obtained by randomly dividing the 1,000 classes of ILSVRC-2012 into 800 base classes and 200 novel classes. The base classes consist of images from the original training set, while the novel classes comprise images from the original validation set.

Cross-domain few-shot. In the 5-way 5-shot cross-domain few-shot classification experiments, the training domain is *mini*Imagenet [13], and the testing is conducted on four different domains. These domains are CropDisease [7], which contains plant disease images; EuroSAT [5], a collection of satellite images; ISIC2018 [12], consisting of dermoscopic images of skin lesions, and ChestX [14], a dataset of X-ray images.

Few-task few-shot. Few-task few-shot learning [15] challenges the common meta-training assumption of having many tasks available. We conductt experiments on four few-task meta-learning challenges, namely *mini*Imagenet-S [13], ISIC [6], DermNet-S [3], and Tabular Murris [1]. To reduce the number of tasks and make it comparable to previous works, we followed [15] by limiting the number of meta-training classes to 12 for miniImagenet-S, with 20 meta-test classes. ISIC [6] involves classifying dermoscopic images across nine diagnostic categories, with 10,015 images available for training in eight different categories, of which we selected four as meta-training classes. DermNet [3], which contains over 23,000 images of skin diseases, was utilized to construct Dermnet-S by selecting 30 diseases as meta-training classes, following [15]. Finally, Tabular Murris [1], which deals with cell type classification across organs and includes nearly 100,000 cells from 20 organs and tissues, was utilized to select 57 base classes as meta-training classes, again following the same approach as [15].

5 Implementation details

In our within-domain experiments, we utilize a Conv-4 and ResNet-12 backbone for *mini*Imagenet and *tiered*Imagenet. A ResNet-50 is used for ImageNet-800. We follow the approach described in [2] to achieve better performance and initially train a feature extractor on all the meta-training data without episodic training. We use the SGD optimizer with a momentum of 0.9, a learning rate starting from 0.1, and a decay factor of 0.1. For *mini*Imagenet, we train for 100 epochs with a batch size of 128, where the learning rate decays at epoch 90. For tieredImageNet, we train for 120 epochs with a batch size of 512, where the learning rate decays at epochs 40 and 80. Lastly, for ImageNet-800, we train for 90 epochs with a batch size of 256, where the learning rate decays at epochs 30 and 60. The weight decay is 0.0005 for ResNet-12 and 0.0001 for ResNet-50. Standard data augmentation techniques, including random resized crop and horizontal flip, are applied. For episodic training, we use the SGD optimizer with a momentum of 0.9, a fixed learning rate of 0.001, and a batch size of 4, meaning each training batch consists of 4 few-shot tasks to calculate the average loss. For our cross-domain experiments, we use a ResNet-10 backbone to extract image features, which is a common choice for cross-domain few-shot classification [16, 4]. The training configuration for this experiment is the same as the within-domain *mini*Imagenet training. For few-task few-shot learning, we follow [15] using a network containing four convolutional blocks and a classifier layer. Each block comprises a 32-filter 3×3 convolution, a batch normalization layer, a ReLU nonlinearity, and a 2×2 max pooling layer. All experiments are performed on a single A100 GPU, each taking approximately 20 hours. We will release all our code.

6 Visualization of diffusion process

The ProtoDiff method utilizes a task-guided diffusion model to generate prototypes that provide efficient class representations, as discussed in the previous section. To better understand the effectiveness of our proposed approach, we provide a visualization by Grad-Cam [10] in Figure of the diffusion process, demonstrating how ProtoDiff gradually aggregates towards the desired class prototype during meta-training. The vanilla prototype is shown in the first row on the left, which does not exclusively focus on the *guitar*. In contrast, the overfitted prototype in the second row on the left provides the highest probability for the *guitar*. ProtoDiff, with the diffusion process, randomly selects certain areas to add noise and perform diffusion, resulting in a prototype that gradually moves towards the *guitar* with the highest probability at t=0. Moreover, ProtoDiff with residual learning produces a more precise prototype. The comparison between these different prototypes demonstrates

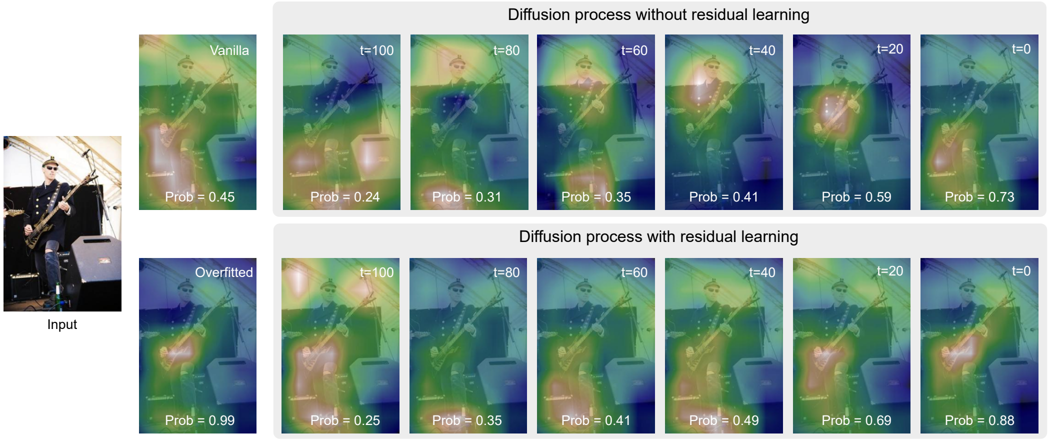

Figure 3: **Visualization of the diffusion process.** The first row on the right shows the vanilla prototype, which does not exclusively focus on the *guitar*. In contrast, the overfitted prototype in the second row on the right provides the highest probability for the *guitar*. ProtoDiff randomly selects certain areas to predict during the diffusion process, with the lowest probability at the beginning time step. As time progresses, the prototype gradually aggregates towards the *guitar*, with the highest probability at t=0. In comparison, ProtoDiff with residual learning produces a more precise prototype.

the effectiveness of the ProtoDiff diffusion process in generating a more accurate and informative prototype for few-shot learning tasks.

References

[1] K. Cao, M. Brbic, and J. Leskovec. Concept learners for few-shot learning. *ICLR*, 2020.

[2] Y. Chen, Z. Liu, H. Xu, T. Darrell, and X. Wang. Meta-baseline: Exploring simple meta-learning for few-shot learning. In *ICCV*, pages 9062–9071, 2021.

[3] Dermnet. Dermnet dataset. `https://dermnet.com/`. 2016.

[4] Y. Guo, N. C. Codella, L. Karlinsky, J. V. Codella, J. R. Smith, K. Saenko, T. Rosing, and R. Feris. A broader study of cross-domain few-shot learning. In *European Conference on Computer Vision*, 2020.

[5] P. Helber, B. Bischke, A. Dengel, and D. Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[6] M. A. A. Milton. Automated skin lesion classification using ensemble of deep neural networks in isic 2018: Skin lesion analysis towards melanoma detection challenge. *arXiv preprint arXiv:1901.10802*, 2019.

[7] S. P. Mohanty, D. P. Hughes, and M. Salathé. Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7:1419, 2016.

[8] M. Ren, E. Triantafillou, S. Ravi, J. Snell, K. Swersky, J. B. Tenenbaum, H. Larochelle, and R. S. Zemel. Meta-learning for semi-supervised few-shot classification. *arXiv preprint arXiv:1803.00676*, 2018.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[10] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017.

[11] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. In *NeurIPS*, pages 4077–4087, 2017.

[12] P. Tschandl, C. Rosendahl, and H. Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific data*, 5(1):1–9, 2018.

[13] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *NeurIPS*, pages 3630–3638, 2016.

[14] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers. Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In *CVPR*, pages 2097–2106, 2017.

[15] H. Yao, L. Zhang, and C. Finn. Meta-learning with fewer tasks through task interpolation. *arXiv preprint arXiv:2106.02695*, 2021.

[16] H. Ye, H. Hu, D. Zhan, and F. Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *CVPR*, 2020.