Supplementary Material of "Glocal Energy-based Learning for Few-Shot Open-Set Recognition"

Haoyu Wang[1]* Guansong Pang[2]* Peng Wang[3]* Lei Zhang[1] Wei Wei[1] Yanning Zhang[1]†
[1]Northwestern Polytechnical University
[2]Singapore Management University [3]University of Wollonong

1. Dataset Details

MiniImageNet [10] is designed to build a lightweight but challenging dataset. It consists of 600 RGB images per class from 100 different classes from ILSVRC-12 [9]. And the classes are randomly sampled from ImageNet, which has 1000 classes. As in previous work [2–4, 6], we adopt the same split of Ravi & Larochelle [7], who resized all images to 84×84 and use 64 classes for training, 16 for validation and 20 for testing.

TieredImageNet [8] is also a subset of ILSVRC-12. Different from miniImageNet, tieredImageNet is not randomly sampled. It extracts 34 categories form ILSVRC-12, each of which contains $10 \sim 30$ different classes, with a total of 608 classes. Each classes has a varying number of images, with a total of 779,165 images. Unlike miniImageNet, tieredImageNet considers ImageNet's category hierarchy. The data is divided according to categories, in which 20 categories (351 classes, 448,695 images) are used as training sets, 6 categories (97 classes, 124,261 images) are used as validation sets, and 8 categories (160 classes, 206,209 images) are used as test sets. We use the same split as [8] and resized all images to 84×84 as the same as previous work [2–4].

CIFAR-FS [1] (CIFAR100 few shots) is randomly sampled from CIFAR100 [5] by using the same criteria with miniImageNet. We use the same split as [1]. But it is much more lightweight. It also consists of 600 RGB images per class from 100 different classes and use 64 classes for training, 16 for validation and 20 for testing. But the resolution of the image is only 32×32.

2. Method Details

In order to illustrate our proposed method more clearly, as shown in Algorithms 1 and 2, we give the PyTorch like pseudo code of our proposed pixel-wise similarity module and energy-based module, respectively.

*H. Wang, G. Pang and P. Wang contributed equally in this work.
†Corresponding author.

Algorithm 1: Pixel-wise similarity, PyTorch-like

```
# f_query: Tensor [2NQ, 1, m^2, dim/2, 1]
#           is the feature map of all query samples
# f_proto: Tensor [1, N, 1, dim/2, m^2]
#           is the feature map of all classes
# N: N-way
# Q: Q-query
# m: spatial dimension of feature map
# dim: channel dimension of feature map
# top_k: the hyperparameter used to select topk
# temp: temperature coefficient hyperparameter
class PixelSimilarity(Module):
    def __init__(args):
        self.top_k = args.top_k
        self.temperature = args.temp
        self.cos = nn.CosineSimilarity(dim=3)

    def forward(f_query, f_proto):
        # cosine similarity
        sim = self.cos(f_query, f_proto)

        # topk
        sim = sim.topk(self.topk, dim=3).values
        sim = sim.sum(dim=[2, 3])
        sim = sim / self.temperature
        return sim
```

3. Experiments Details

3.1. Detailed Task Sampling

For a N-way K-shot Q-query FSOR task, its sampling is independent of the dataset and training or testing stage. Specifically, it will randomly select 2N classes from the dataset, half of which will be regarded as known classes and the other half as unknown classes. Q samples will be sampled for each classes as query, while additional K samples will be sampled for known classes as support.

Algorithm 2: Energy-based Module, PyTorch like

```
# e_sim: Tensor [2NQ, N] is class-wise similarity
# f_sim: Tensor [2NQ, N] is pixel-wise similarity
# N: N-way
# Q: Q-query
# m_k: margin for closed-set samples
# m_u: margin for open-set samples
class EnergyLoss(Module):
    def __init__(args):
        self.m_k = args.m_k
        self.m_u = args.m_u

    def forward(e_sim, f_sim):
        # energy score
        e_egy = -torch.logsumexp(e_sim, dim=1)
        f_egy = -torch.logsumexp(f_sim, dim=1)
        energy = e_egy + f_egy

        # energy loss
        k_egy, u_egy = torch.split(energy, NQ)
        l_k = pow(F.relu(k_egy - self.m_k), 2)
        l_u = pow(F.relu(self.m_u - u_egy), 2)
        l_energy = l_k.mean() + l_u.mean()
        return l_energy
```

3.2. Detailed Ablation Study

We provide the detailed ablation results on miniImageNet, tieredImageNet and CIFAR-FS under 5-way 1-shot and 5-shot setting.

As show in Table 1, we give the ablation study results of three modules proposed by us on all benchmarks. On miniImageNet and tirerdImageNet dataset, our three modules can improve the performance of open-set recognition while maintaining the closed-set classification performance. On the CIFAR-FS dataset, our pixel-wise similarity branch may not work due to the decrease in original image resolution. The spatial dimension of the feature map is only 2×2, the pixel-wise similarity branch is difficult to get discriminative local information. But using only the energy loss, our open-set recognition performance can significantly exceed the baseline.

We also give the results of glocal energy score combination of the class-wise branch and pixel-wise on all benchmarks. It can be seen in Table 2 that *Ahead Combine* can always improve open-set recognition performance compared to *Delay Combine*.

For the ablation study of combination coefficients between class-wise energy and pixel-wise energy, we give detailed results in Table 3. Compared to *Learnable* and *Task-adaptive*, *Fixed value* achieves better open-set recognition performance on all benchmarks.

References

[1] Luca Bertinetto, Joao F Henriques, Philip HS Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. *arXiv preprint arXiv:1805.08136*, 2018. 1

[2] Shule Deng, Jin-Gang Yu, Zihao Wu, Hongxia Gao, Yansheng Li, and Yang Yang. Learning relative feature displacement for few-shot open-set recognition. *IEEE Transactions on Multimedia*, 2022. 1

[3] Shiyuan Huang, Jiawei Ma, Guangxing Han, and Shih-Fu Chang. Task-adaptive negative envision for few-shot open-set recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7171–7180, 2022. 1

[4] Minki Jeong, Seokeon Choi, and Changick Kim. Few-shot open-set recognition by transformation consistency. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12566–12575, 2021. 1

[5] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5(4):1, 2010. 1

[6] Bo Liu, Hao Kang, Haoxiang Li, Gang Hua, and Nuno Vasconcelos. Few-shot open-set recognition using meta-learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, 2020. 1

[7] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016. 1

[8] 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. 1

[9] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015. 1

[10] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016. 1

Benchmark	Method	ACC↑	AUROC↑	F1 Score↑	FPR95↓	AUPR↑
miniImageNet 5-way 1-shot	Baseline	68.53 ± 0.78	69.85 ± 0.87	65.06 ± 0.69	79.35 ± 1.45	70.53 ± 0.83
	+ Energy Loss	68.14 ± 0.78	71.97 ± 0.83	65.92 ± 0.69	77.88 ± 1.34	71.96 ± 0.83
	+ Pixel wise	68.40 ± 0.86	72.49 ± 0.76	67.38 ± 0.69	74.99 ± 1.35	73.28 ± 0.85
	+ Combine score	68.26 ± 0.85	73.70 ± 0.82	67.72 ± 0.70	74.10 ± 1.38	73.80 ± 0.87
miniImageNet 5-way 5-shot	Baseline	83.66 ± 0.55	77.46 ± 0.78	71.36 ± 0.68	70.95 ± 1.35	78.63 ± 0.69
	+ Energy Loss	82.88 ± 0.55	79.25 ± 0.69	72.24 ± 0.63	69.02 ± 1.54	79.97 ± 0.70
	+ Pixel wise	83.57 ± 0.53	80.47 ± 0.66	73.32 ± 0.61	66.18 ± 1.50	80.98 ± 0.66
	+ Combine score	83.05 ± 0.55	82.29 ± 0.60	74.75 ± 0.56	61.52 ± 1.44	82.40 ± 0.63
tieredImageNet 5-way 1-shot	Baseline	70.55 ± 0.93	71.66 ± 0.82	67.02 ± 0.63	74.66 ± 1.40	71.01 ± 0.83
	+ Energy Loss	70.60 ± 0.94	73.57 ± 0.84	67.90 ± 0.61	72.60 ± 1.46	73.48 ± 0.86
	+ Pixel wise	70.39 ± 0.96	74.06 ± 0.73	67.90 ± 0.61	71.72 ± 1.32	73.10 ± 0.75
	+ Combine score	70.50 ± 0.93	75.86 ± 0.81	69.44 ± 0.68	70.22 ± 1.47	75.78 ± 0.80
tieredImageNet 5-way 5-shot	Baseline	85.38 ± 0.66	76.32 ± 0.71	70.42 ± 0.62	68.91 ± 1.60	76.82 ± 0.70
	+ Energy Loss	84.24 ± 0.69	79.54 ± 0.67	72.38 ± 0.61	66.40 ± 1.55	79.86 ± 0.66
	+ Pixel wise	85.06 ± 0.66	80.22 ± 0.72	73.08 ± 0.61	63.29 ± 1.63	80.14 ± 0.74
	+ Combine score	84.60 ± 0.65	81.95 ± 0.72	74.39 ± 0.63	59.77 ± 1.56	82.17 ± 0.68
CIFAR-FS 5-way 1-shot	Baseline	76.85 ± 0.88	78.51 ± 0.75	71.70 ± 0.72	67.02 ± 1.72	79.11 ± 0.80
	+ Energy Loss	76.67 ± 0.90	79.43 ± 0.72	72.44 ± 0.63	65.59 ± 1.72	79.70 ± 0.74
	+ Pixel wise	76.60 ± 0.87	78.33 ± 0.81	71.65 ± 0.71	67.12 ± 1.72	78.73 ± 0.80
	+ Combine score	76.77 ± 0.88	78.67 ± 0.80	71.77 ± 0.72	66.93 ± 1.72	79.51 ± 0.81
CIFAR-FS 5-way 5-shot	Baseline	87.98 ± 0.61	84.93 ± 0.64	77.12 ± 0.59	57.04 ± 1.95	84.95 ± 0.56
	+ Energy Loss	87.63 ± 0.62	86.84 ± 0.58	79.51 ± 0.58	54.47 ± 1.98	87.81 ± 0.52
	+ Pixel wise	86.94 ± 0.65	85.55 ± 0.59	78.10 ± 0.59	55.68 ± 1.93	86.17 ± 0.56
	+ Combine score	86.74 ± 0.66	86.56 ± 0.59	79.15 ± 0.59	55.33 ± 2.03	87.71 ± 0.52

Table 1. Ablation study of three modules proposed in GEL. We report the 5-way 1-shot and 5-shot results on miniImageNet, tieredImageNet and CIFAR-FS to demonstrate the effectiveness of our modules.

		5-way 1-shot		5-way 5-shot	
Dataset	Method	ACC	AUROC	ACC	AUROC
miniImageNet	Delay Combine	68.39 ± 0.86	73.17 ± 0.83	83.00 ± 0.55	80.67 ± 0.65
	Ahead Combine	68.26 ± 0.85	73.70 ± 0.82	83.05 ± 0.55	82.29 ± 0.60
tieredImageNet	Delay Combine	70.72 ± 0.99	74.00 ± 0.79	84.44 ± 0.66	80.57 ± 0.69
	Ahead Combine	70.50 ± 0.93	75.86 ± 0.81	84.60 ± 0.65	81.95 ± 0.72
CIFAR-FS	Delay Combine	76.58 ± 0.86	78.29 ± 0.76	87.75 ± 0.64	86.01 ± 0.63
	Ahead Combine	76.77 ± 0.88	78.67 ± 0.80	86.74 ± 0.66	86.56 ± 0.59

Table 2. Ablation study of glocal energy score combination of the class-wise branch and pixel-wise branch on miniImageNet, tieredImageNet and CIFAR-FS under 5-way 1-shot and 5-shot setting.

Dataset	Method	5-way 1-shot ACC	5-way 1-shot AUROC	5-way 5-shot ACC	5-way 5-shot AUROC
miniImageNet	Fixed value	68.26 ± 0.85	73.70 ± 0.82	83.05 ± 0.55	82.29 ± 0.60
	Learnable	68.34 ± 0.85	73.40 ± 0.82	83.08 ± 0.56	81.37 ± 0.64
	Task-adaptive	68.34 ± 0.85	73.71 ± 0.72	83.00 ± 0.56	81.34 ± 0.62
tieredImageNet	Fixed value	70.50 ± 0.93	75.86 ± 0.81	84.60 ± 0.65	81.95 ± 0.72
	Learnable	70.99 ± 0.93	75.59 ± 0.80	84.84 ± 0.65	80.86 ± 0.70
	Task-adaptive	70.68 ± 0.92	74.93 ± 0.83	84.79 ± 0.66	80.49 ± 0.72
CIFAR-FS	Fixed value	76.77 ± 0.88	78.67 ± 0.80	86.74 ± 0.66	86.56 ± 0.59
	Learnable	76.60 ± 0.87	78.62 ± 0.80	86.94 ± 0.65	87.29 ± 0.57
	Task-adaptive	76.70 ± 0.87	77.99 ± 0.78	87.33 ± 0.60	85.67 ± 0.60

Table 3. Ablation study of combination coefficients between class-wise energy and pixel-wise energy on miniImageNet, tieredImageNet and CIFAR-FS under 5-way 1-shot and 5-shot setting.

Dataset	Method	5-way 1-shot ACC	5-way 1-shot AUROC	5-way 5-shot ACC	5-way 5-shot AUROC
MiniImageNet	RFDNet	66.40 ± 0.82	71.91 ± 0.78	81.91 ± 0.60	$\mathbf{81.29 \pm 0.56}$
MiniImageNet	Ours	$\mathbf{67.40 \pm 0.84}$	$\mathbf{72.35 \pm 0.80}$	$\mathbf{82.32 \pm 0.56}$	80.89 ± 0.65
TieredImageNet	RFDNet	70.33 ± 0.90	75.12 ± 0.74	$\mathbf{84.88 \pm 0.62}$	$\mathbf{82.19 \pm 0.66}$
TieredImageNet	Ours	$\mathbf{70.45 \pm 0.92}$	$\mathbf{75.69 \pm 0.83}$	84.21 ± 0.67	81.47 ± 0.76
CIFAR-FS	RFDNet	71.34 ± 0.90	76.74 ± 0.75	84.61 ± 0.62	85.17 ± 0.57
CIFAR-FS	Ours	$\mathbf{73.36 \pm 0.89}$	$\mathbf{77.69 \pm 0.82}$	$\mathbf{85.46 \pm 0.63}$	$\mathbf{85.68 \pm 0.64}$

Table 4. Results of our method with ResNet-18 backbone. We reimplement our method using ResNet-18 as RFDNet, and report their comparison in the table. Our method still outperforms RFDNet in most cases.

Figure 1. Four additional examples.