Appendix for Weak-shot Fine-grained Classification via Similarity Transfer

Junjie Chen, Li Niu*, Liu Liu, Liqing Zhang*
MoE Key Lab of Artificial Intelligence,
Department of Computer Science and Engineering,
Shanghai Jiao Tong University
{chen.bys, ustcnewly, shirlley}@sjtu.edu.cn, zhang-lq@cs.sjtu.edu.cn

In this appendix, we first formulate the optimization of adversarial similarity net in Section 1. We clarify the dataset details and implementation details in Section 2 and 3. Then, we analyse the impact of hyper-parameters in Section 4. Afterwards, we report the results on ImageNet dataset in Section 5. We provide qualitative analysis for the sample weights and similarity matrix learnt by our method in Section 6. Afterwards, we investigate the impact of different noise ratios in Section 7 and explore the impact of different backbones in Section 8.

1 Adversarial Similarity Net

Here we introduce the optimization details of adversarial similarity net (SimNet) in Section 4.1 of the main paper. We denote the similarity between the i-th image $\mathbf{x}_i$ and the j-th image $\mathbf{x}_j$ as

$$s_{i,j} = P(G(\mathbf{x}_i, \mathbf{x}_j)) = P(\mathbf{r}_{i,j}), \tag{1}$$

where $G(\cdot)$ generates the relation feature $\mathbf{r}_{i,j} = G(\mathbf{x}_i, \mathbf{x}_j)$ given an image pair, and $P(\cdot)$ outputs the similarity score $s_{i,j}$ given the relation feature. The discriminator can be represented by

$$d_{i,j} = D(\mathbf{r}_{i,j}), \tag{2}$$

where $d_{i,j}$ is the discriminator score indicating that the relation feature $\mathbf{r}_{i,j}$ comes from base training set instead of novel training set. To avoid confusion, we use the super-scripts b and n to distinguish the variables for base training set and novel training set. In each training iteration, we construct mini-batch X^b and X^n with size M, and perform the following two optimizing steps in an alternating manner:

Firstly, we freeze the generator $G(\cdot)$ and update the discriminator $D(\cdot)$ by minimizing

$$L_D = \frac{1}{M^2} \sum_{i=1}^{M} \sum_{j=1}^{M} \left[\log(1 - D(\mathbf{r}^b_{i,j})) + \log D(\mathbf{r}^n_{i,j})\right]. \tag{3}$$

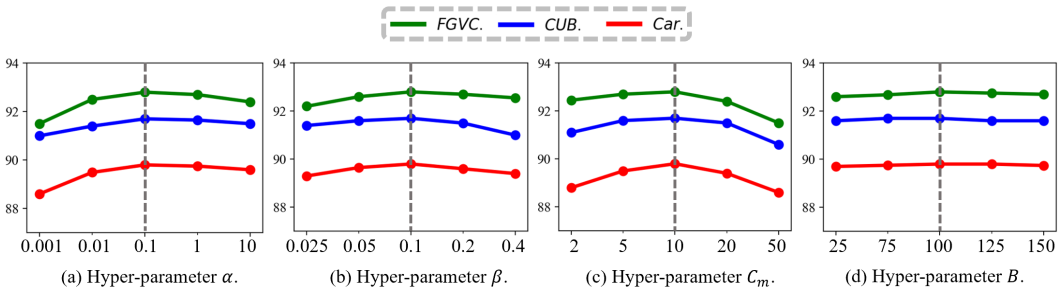
Secondly, we freeze the discriminator $D(\cdot)$ and update the whole SimNet by minimizing

$$L_G = -\beta L_D + \frac{1}{M^2} \sum_{i=1}^{M} \sum_{j=1}^{M} [-c_{i,j} \log P(\mathbf{r}^b_{i,j}) - (1 - c_{i,j}) \log(1 - P(\mathbf{r}^b_{i,j}))], \tag{4}$$

where $c_{i,j}$ is the binary relation category label (0 for "dissimilar pair" and 1 for "similar pair"), and β is a trade-off parameter set as 0.1 via cross-validation.

*Corresponding author

35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.



(a) Hyper-parameter α. (b) Hyper-parameter β. (c) Hyper-parameter C_m. (d) Hyper-parameter B.

Figure 1: The effects of varying the values of α (a) and β (b) on three datasets. The dashed vertical lines denote the default values used in our paper.

2 Datasets

We evaluate our method on three popular fine-grained datasets. 1) CompCars [12] (Car for short): we regard the 431 car models as fine-grained categories; 2) CUB [10]: the standard 200 bird species are regarded as fine-grained categories; 3) FGVC [5]: we regard the 100 aircraft variants as fine-grained categories. In terms of base/novel category split, for CUB, we follow [11, 6], which is commonly used for zero/few-shot learning. For Car and FGVC, we randomly split base and novel categories by $3:1$, with the same split ratio as in [11, 6].

Then, we need to construct web training set for novel categories. For Car, we use the released web images in WebCars [15], where the noise ratio is about 30% according to some sampled images from a few categories [15]. For CUB and FGVC, the datasets used in [3] only provide image URLs, most of which have expired, so we construct the web training images by ourselves. In particular, we use the category names as queries to obtain 1000 images from Google website after performing near-duplicate removal [14, 6].

3 Implementation Details

The proposed approach is implemented in Python 3.7 and Pytorch 1.0.0 [7], on Ubuntu 18.04 with 32 GB Intel 9700K CPU and two NVIDIA 2080ti GPUs. Standard data augmentation is applied for all methods on all datasets, including random rotation, random resized crop, and random horizontal flip. Analogous to [13], the backbone of similarity net is pre-trained on novel training set of the corresponding dataset. We choose the classification accuracy as the evaluation metric for the main classifier, following [6]. We choose precision rate, recall rate, and F1-score as the evaluation metrics for similarity net, following [13].

To select optimal hyper-parameters, we follow the validation strategy used in [6, 8]. Assume that there are C_b base categories and C_n novel categories, we choose the first C_v categories ($C_v = \lfloor \frac{C_n * C_b}{C_b + C_n} \rfloor$) based on the default category indices from C_b base categories as validation categories. As a consequence, we need to further collect web images for validation categories. In the validation stage, we regard C_v categories as novel categories and $C_b - C_v$ categories as base categories. Then, we determine the optimal hyper-parameters according to the validation performance via random search [6] within appropriate range. For learning similarity net (SimNet) on base training set, we adopt the SGD optimizer with learning rate 0.01 and batch size 100 to train 50 epochs for all datasets. For learning the main classifier on novel training set, we adopt the SGD optimizer with learning rate 0.005 and batch size 128. We train 50 epochs for CUB and FGVC datasets, while training 100 epochs for Car datasets. We set weight decay as 0.0001 and momentum as 0.9 for both SimNet and the main classifier on all datasets.

4 Hyper-parameter Analysis

In this section, we analyse the hyper-parameters: α for balancing graph regularization loss (see Eqn. (4) in the main paper), β for balancing adversarial loss (see Eqn. (4) in Appendix), C_m and B in Section 4.1 of main paper for training SimNet. The optimal values are determined via cross-validation

Table 1: Accuracies of different methods on ImageNet dataset in the weak-shot setting. The best results are highlighted in boldface.

Method	Accuracy (%)
Cls Loss	74.2
SOMNet+MetaBaseline	75.1
DivideMix+MetaBaseline	76.5
SimTrans	**78.7**

Table 2: Accuracies (%) on various levels of noise. The best results are highlighted in boldface.

Noise Ratio	10%	20%	30%	40%
Cls Loss	77.1	74.7	71.1	66.0
SOMNet+MetaBaseline	78.0	76.0	73.8	69.4
DivideMix+MetaBaseline	78.2	76.4	74.3	70.2
SimTrans	**79.4**	**78.3**	**76.4**	**73.6**

as discussed in Section 3 and kept the same on all datasets (The batch sizes B are with compatible learning rates). As shown in Figure 1, we vary each hyper-parameter in a range while fixing the other hyper-parameters, and plot the results obtained by our full-fledged method. The results reported in Figure 1 suggest that the performance of our method is robust to the hyper-parameters in an appropriate range.

5 Experiments on ImageNet dataset

Note that the focus of this work is fine-grained classification, so we mainly conduct experiments on fine-grained datasets. In this section, we explore the effectiveness of our setting and method on a large-scale dataset ImageNet [1], even though it is not fine-grained.

Following the random split in [9, 13], we have 882 categories and 118 categories for base set and novel set, respectively. The base training/test set and the novel test set are from the ImageNet dataset [1] while the novel training set is constructed using WebVision 1.0 [4] (about 20% noise ratio according to [4]). Specifically, each base category contains 1279 clean training images and 50 test images on average, while each novel category contains 2423 web training images and 50 test images on average.

We compare with the basic baseline *Cls Loss* as well as two combinations of webly supervised learning method and transfer learning method across categories, *i.e.*, *SOMNet+MetaBaseline* and *DivideMix+MetaBaseline*, considering their competitive performance reported in the main paper. The results are listed in Table 1, from which we can find that our method still dramatically improves the results, even on a coarse-grained dataset.

6 Qualitative Analysis for Sample Weight and Similarity Matrix

According to our hypothesis and formulation (Eqn. (1) in the main paper), in our weighted classification loss, higher weights are more likely to be assigned to non-outliers. Besides, the transferred similarities could reveal the semantic relations among web images. For qualitative analysis of assigned weights, by taking the "Evening Grosbeak" category in CUB as an example, we rank all web training images by assigned weights. According to the rank, we show the first 3 and the last 3 images as well as their similarity matrix in Figure 2. On the one hand, we could observe that the images with high weights are very similar to clean images, while the images with low weights are outliers. On the other hand, we observe that the transferred similarities accurately portray the semantic relations among web images. We have similar observations for the other categories and on the other datasets.

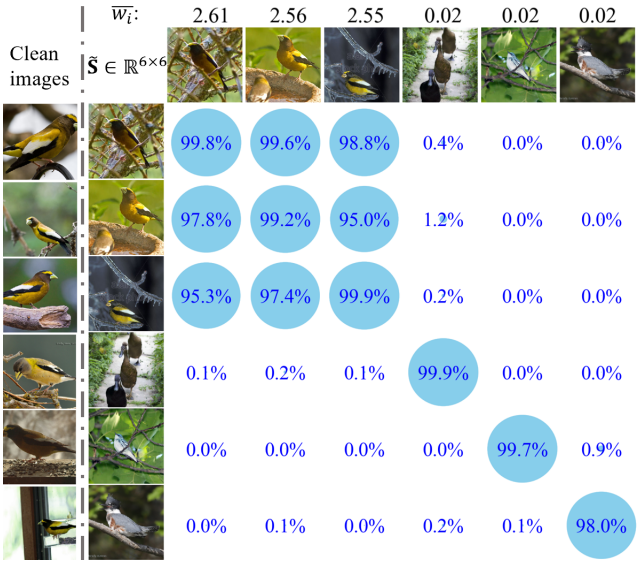


Figure 2: Visualization of sample weights and similarity matrix of 6 web images (3 with the highest weights and 3 with the lowest weights) from the category "Evening Grosbeak". The sample weights are shown on top of the web images. The right part shows the 6×6 similarity matrix. The left column shows some clean test images of the visualized category for reference.

Table 3: Accuracies (%) of different methods on four scales of backbone on CUB dataset. The best results are highlighted in boldface.

Method	ResNet18	ResNet34	ResNet50	ResNet101
Cls Loss	80.7	83.3	85.4	85.9
SOMNet+MetaBaseline	82.8	85.9	88.3	89.2
DivideMix+MetaBaseline	83.6	86.4	88.5	89.3
SimTrans	**88.3**	**90.4**	**91.7**	**92.1**

7 Experiments with Different Noise Ratios

In practice, we have no ground-truth labels for web data and expert knowledge is especially required to annotate or check labels for fine-grained categories, so it is hard to correct the labels and change the noise ratio of fine-grained web datasets. Therefore, to control the noise ratios, we can only conduct experiments on synthetic noisy data instead of real-world web data [6, 3]. Following [2], we synthesize noisy training data with different noise ratios using CIFAR-100. Specifically, as in [9, 13], we randomly split the classes into base set and novel set with the same split ratio stated in Section 2. For label noise, we circularly flip each class into the next within each super-class of CIFAR-100, following [2]. We summarize the results in Tab. 2, where we could see that our method is generally effective for different noise ratios.

8 Impact of Different Backbones.

We conduct experiments using different backbones on CUB dataset, including ResNet18, ResNet34, ResNet50, and ResNet101. We compare with the two most representative and competitive baselines, *i.e.*, "SOMNet+MetaBaseline" and "DivideMix+MetaBaseline". Are reported in Tab. 3, our method could achieve general improvement for different scales of architectures.

References

[1] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[2] Youngdong Kim, Junho Yim, Juseung Yun, and Junmo Kim. Nlnl: Negative learning for noisy labels. In *ICCV*, 2019.

[3] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *ECCV*, 2016.

[4] Wen Li, Limin Wang, Wei Li, Eirikur Agustsson, and Luc Van Gool. Webvision database: Visual learning and understanding from web data. *arXiv preprint arXiv:1708.02862*, 2017.

[5] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[6] Li Niu, Ashok Veeraraghavan, and Ashutosh Sabharwal. Webly supervised learning meets zero-shot learning: A hybrid approach for fine-grained classification. In *CVPR*, 2018.

[7] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[8] Seyed Mohsen Shojaee and Mahdieh Soleymani Baghshah. Semi-supervised zero-shot learning by a clustering-based approach. *arXiv preprint arXiv:1605.09016*, 2016.

[9] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *NeurIPS*, 2016.

[10] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[11] Yongqin Xian, Bernt Schiele, and Zeynep Akata. Zero-shot learning-the good, the bad and the ugly. In *CVPR*, 2017.

[12] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *CVPR*, 2015.

[13] Zhaoyang Lv Yen-Chang Hsu and Zsolt Kira. Learning to cluster in order to transfer across domains and tasks. In *ICLR*, 2018.

[14] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[15] Bohan Zhuang, Lingqiao Liu, Yao Li, Chunhua Shen, and Ian Reid. Attend in groups: a weakly-supervised deep learning framework for learning from web data. In *CVPR*, 2017.