Supplementary Material of "Revisiting Unsupervised Domain Adaptation Models: a Smoothness Perspective"

Xiaodong Wang[1], Junbao Zhuo[2⋆], Mengru Zhang[3], Shuhui Wang[2], and Yuejian Fang[1⋆]

[1] School of Software & Microelectronics, Peking University
[2] Key Lab of Intell. Info. Process., Inst. of Comput. Tech., CAS
[3] MEGVII Technology
{wangxiaodong21s@stu,fangyj@ss}.pku.edu.cn,
{junbao.zhuo@vipl.,wangshuhui@}ict.ac.cn, zhangmengru@megvii.com

A appendix

A.1 Variance Calculation

In this paper, the smoothness is defined as the reciprocal of intra-class variance, because the intra-class variance is easy to be calculated. Also, the inter-class variance could represent the discriminability of target features. So, we can use the intra-class and iter-class variance to clarify the smoothness. In the introduction of our paper, we show the intra-class variance and inter-class variance of methods including CDAN, MCC, and the combinations with the LeCo.

Firstly, we randomly selected the 10k target samples from VisDA-C, and we obtained the classification responses through these different models. The VisDA-C dataset contains images from 12 categories.

Intra-class variance: For each UDA model, firstly, by using the ground-truth labels, we can calculate the standard deviation of classification responses within each class, and the shape of the standard deviation for each class is $(1, 12)$, then we take the average across 12 classes. Then, We take the average of 12 standard deviations as the final intra-class variance of this model.

Inter-class variance: For each UDA model, we first obtained the class centers of classification responses by using the ground-truth labels. Then we calculated the standard variance across these cluster centers, and the shape of standard variance is $(1, 12)$, so we take the average on the class dimension as the inter-class variance of this model.

Obviously, this calculation way can be applied to the various methods and different datasets. The codes can be found in our codebase.

⋆ Corresponding author

A.2 Dataset

Office-31 [1] is a classic benchmark with 31 categories and three domains: *Amazon* (A) with 2,817 images, *Dslr* (D) 498 images, and *Webcam* (W) with 795 images, containing 6 transfer tasks.

Office-Home [2] is a more difficult benchmark that consists of images from four domains: *Art* (Ar), *Clipart* (Cl), *Product* (Pr), and *Real-world* (Rw), totally around 15,500 images from 65 different categories. All 12 transfer tasks are selected for evaluation.

VisDA-C [3] is a large-scale benchmark that contains the images from 12 categories of two very distinct domains: *synthetic* domain and *real-world* domain. The synthetic domain contains 152,397 images and the latter contains 55,388 images, and we focus on the synthetic-to-real transfer task.

DomainNet [4] is the largest and hardest benchmark for UDA by far. It contains approximately 0.6 million images with 345 categories and 6 domains: *Clipart* (Clp), *Infograph* (Inf), *Painting* (Pnt), *Quickdraw* (Qdr), *Real* (Rel), and *Sketch* (Skt). Here we focus on the tasks between Clp, Pnt, Rel, and Skt, following the settings in [5], which contains 12 relatively challenging transfer tasks.

A.3 Algorithm

We provide the algorithm pseudocode for LeCo in PyTorch style, as shown in Alg.1. Most UDA methods can be considered as the baseline in our algorithm. We release our codebase using in our experiments, the details are clear in README.md in it.

References

1. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: European conference on computer vision, Springer (2010) 213–226
2. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2017) 5018–5027
3. Peng, X., Usman, B., Kaushik, N., Hoffman, J., Wang, D., Saenko, K.: Visda: The visual domain adaptation challenge. arXiv preprint arXiv:1710.06924 (2017)
4. Peng, X., Bai, Q., Xia, X., Huang, Z., Saenko, K., Wang, B.: Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. (2019) 1406–1415
5. Jiang, J., Chen, B., Fu, B., Long, M.: Transfer-learning-library. https://github.com/thuml/Transfer-Learning-Library (2020)

Algorithm 1 LeCo PyTorch pseudocode.

Input: Source and Target data sets $\mathcal{D}_s$ and $\mathcal{D}_t$

Init: parameters $\theta = \{\theta_\psi, \theta_f\}$, batch size: B, tradeoff parameter: λ, warm-up iterations k, total iterations N.

for $i \leftarrow 1$ **to** N **do**

    X^t_w = weak_aug(X^t) # a mini-batch weak views

    X^t_{str} = strong_aug(X^t) # a mini-batch strong views

    baselines:

    Compute classification loss $\mathcal{L}_s$

    Compute baseline loss $\mathcal{L}_{dom}/\mathcal{L}_{reg}$

    LeCo:

    $\hat{Y}^t_w = f(\psi(X^t_w))$ # classification response (B, K)

    $\hat{Y}^t_{str} = f(\psi(X^t_{str}))$ # classification response (B, K)

    Update model parameters:

    if $i < k$ **then**

        $\min\limits_{\theta} \mathcal{L}_s + \mathcal{L}_{dom}/\mathcal{L}_{reg}$ # warm-up

    else

        Compute *Instance Class Confusion* vector $\boldsymbol{I}$

        $\tau = \boldsymbol{I}$.mean() # $\boldsymbol{I}$: (B, 1)

        $\boldsymbol{I} = (\boldsymbol{I} < \tau)$.detach() # no gradient to $\boldsymbol{I}$

        L2_dis = (($\hat{Y}^t_w - \hat{Y}^t_{str}$)**2).mean(dim=1)

        $\mathcal{L}_{leco}$ =(L2_dis.view(-1, 1)*$\boldsymbol{I}$.view(-1, 1)).sum()

        $\min\limits_{\theta} \mathcal{L}_s + \mathcal{L}_{dom}/\mathcal{L}_{reg} + \lambda\mathcal{L}_{leco}$

    end if

end for

Output: θ