OpenLDN: Learning to Discover Novel Classes for Open-World Semi-Supervised Learning (Supplementary Material)

Mamshad Nayeem Rizve[1], Navid Kardan[1], Salman Khan[2], Fahad Shahbaz Khan[2], and Mubarak Shah[1]

[1] Center for Research in Computer Vision, UCF, USA
[2] Mohamed bin Zayed University of AI, UAE
{nayeemrizve, kardan}@knights.ucf.edu, {salman.khan, fahad.khan}@mbzuai.ac.ae, shah@crcv.ucf.edu

1 Overview

This supplementary document includes information about our training algorithm, experimental setup, and further evaluations. We provide our training algorithm in Section 2. Next, we cover additional implementation details in Section 3. Section 4 provides details for the baseline implementations. We conduct experiments within a more restricted environment with limited number of labeled data samples in Section 5. We include results for a moderately imbalanced dataset, i.e. SVHN, in Section 6. Furthermore, we provide results on three additional datasets (FGVC-Aircraft, Stanford-Cars, and Herbarium19) in Section 7. Finally, we discuss the effect of varying the frequency of iterative pseudo-labeling in our proposed OpenLDN approach in Section 8.

2 OpenLDN Training Algorithm

We provide OpenLDN training algorithm in Alg. 1. For OpenLDN training, we require a set of labeled data $\mathbb{S}_L$, and a set of unlabeled data $\mathbb{S}_U$. In addition to this, we need to set the number of maximum iterations for stage-1 (learning to discover novel classes) and stage-2 (closed-world SSL) training: t_1 and t_2 and also the frequency for iterative pseudo-labeling, m. The OpenLDN algorithm outputs trained feature extractor f_Θ, and classifier f_Φ.

For stage-1 of OpenLDN training, first, we initialize feature extractor, f_Θ, classifier, f_Φ, and similarity prediction network, f_Ω. Next, we sample a batch of labeled examples from known classes, $\mathbf{X}^l$, and their corresponding labels $\mathbf{Y}^l$. We also sample a batch of unlabeled samples from both known and novel classes $\mathbf{X}^u$. After that, to learn to recognise novel classes, we compute $\mathcal{L}_{nov}$ and update the parameters of feature extractor, f_Θ, and classifier, f_Φ, accordingly. Next, we compute a cross-entropy loss on the labeled examples with these updated parameters and update the parameters of similarity prediction network, f_Ω, based on this cross-entropy loss using the proposed bi-level optimization rule. We

continue this training procedure for t_1 iterations to train the feature extractor f_Θ, classifier f_Φ, and similarity prediction network, f_Ω. We use the trained feature extractor f_Θ and classifier f_Φ for generating pseudo-labels for the subsequent closed-world SSL training.

For stage-2 of OpenLDN training, first, we generate pseudo-labels, $\mathbb{S}_{PL}$, for novel classes using the trained feature extractor, f_Θ, and classifier, f_Φ. Next, we select top-k pseudo-labels from each class, $\mathbb{S}_{selected}$. After that, we complement the original set of labeled samples with the selected pseudo-labeled samples, $\tilde{\mathbb{S}}_L$ and also remove the selected pseudo-labeled samples from the unlabeled set to obtain $\tilde{\mathbb{S}}_U$. We re-initialize the feature extractor and classifier for the subsequent closed-world SSL training. Next, we perform closed-world SSL training by sampling a batch of labeled/pseudo-labeled samples $\mathbf{X}^l$, their corresponding labels/pseudo-labels $\mathbf{Y}^l$, and also a batch of unlabeled samples $\mathbf{X}^u$. We update the network parameters based on the appropriate closed-world SSL loss. We repeat the pseudo-label generation process every m iterations to mitigate the impact of noisy pseudo-labels. The stage-2 of OpenLDN ends after t_2 iterations and returns the trained feature extractor f_Θ, and classifier f_Φ.

Algorithm 1 OpenLDN training algorithm

Input: Set of labeled data $\mathbb{S}_L$, set of unlabeled data $\mathbb{S}_U$, maximum iterations t_1 and t_2, and frequency of iterative pseudo-labeling m
Output: Trained feature extractor f_Θ, and classifier f_Φ

Stage-1: Learning to Discover Novel Classes

Initialize feature extractor f_Θ, classifier f_Φ, and similarity prediction network f_Ω
for $t = 1...t_1$ **do**
  $\mathbf{X}^l, \mathbf{Y}^l \leftarrow \text{SampleBatch}(\mathbb{S}_L)$
  $\mathbf{X}^u \leftarrow \text{SampleBatch}(\mathbb{S}_U)$
  $\mathcal{L} \leftarrow \mathcal{L}_{nov}(\Theta^{(t)}, \Phi^{(t)}, \Omega^{(t)}, \mathbf{X}^l, \mathbf{X}^u, \mathbf{Y}^l)$ ▷ Eq. 1
  $(\Theta^{(t+1)}, \Phi^{(t+1)}) \leftarrow (\Theta^{(t)}, \Phi^{(t)}) - \alpha_{(\Theta,\Phi)} \nabla_{(\Theta,\Phi)} \mathcal{L}$
  $\mathcal{L} \leftarrow \mathcal{L}^l_{ce}(\Theta^{(t+1)}, \Phi^{(t+1)}, \mathbf{X}^l, \mathbf{Y}^l)$
  $\Omega^{(t+1)} \leftarrow \Omega^{(t)} - \alpha_\Omega \nabla_\Omega \mathcal{L}$

Stage-2: Closed-World SSL Training

$\mathbb{S}_{PL} \leftarrow$ Generate Pseudo-Labels ▷ Eq. 8
$\mathbb{S}_{selected} \leftarrow \text{TopK}(\mathbb{S}_{PL})$ ▷ Select top-k
$\tilde{\mathbb{S}}_L \leftarrow \mathbb{S}_L \cup \mathbb{S}_{selected}$
$\tilde{\mathbb{S}}_U \leftarrow \mathbb{S}_U \setminus \mathbb{S}_{selected}$
Initialize feature extractor f_Θ, and classifier f_Φ
for $t = 1...t_2$ **do**
  $\mathbf{X}^l, \mathbf{Y}^l \leftarrow \text{SampleBatch}(\tilde{\mathbb{S}}_L)$
  $\mathbf{X}^u \leftarrow \text{SampleBatch}(\tilde{\mathbb{S}}_U)$
  Update Θ and Φ using closed-world SSL loss
  if $t\%m = 0$ **then**
    Repeat steps 9 to 12
return f_Θ, f_Φ

3 Additional Implementation Details

In this section, we provide additional implementation details of our method. In OpenLDN algorithm, we use standard data augmentations which include random crop, and random horizontal flip for all datasets. To obtain the strongly augmented version of the image, $\mathbf{x}^s$, we use Randaugment [3]. We use the default parameters for Randaugment in all the datasets and set the value of N to 2 and M to 10. Here, N is the number of concurrent random augmentations and M is the magnitude of the selected augmentations in Randaugment. Following the prior works [6,2], we also modify the base feature extractor, ResNet-18 [8], for CIFAR-10, and CIFAR-100 datasets. To this end, we remove the first max-pooling layer; this helps in dealing with images of smaller resolution (32×32). We also change the first convolutional layer; we set the stride to 1 and the kernel size to 3×3. For Tiny ImageNet dataset experiment, we do not remove the first max-pooling layer. However, we do make the same change to the first convolutional layer as above. We do not modify the network architecture for ImageNet-100, and Oxford-IIIT Pet dataset experiments. To reduce the training time, we downsample the images (train and test) of Oxford-IIIT Pet dataset to 256×256 before applying data augmentation. We use the same downsampling operation for the baseline methods. We do not modify default parameters for Mixmatch [1] and UDA [15]. Finally, we apply Mixup [16] augmentation for the labeled and pseudo-labeled data in UDA training. We observe that this change helps the network to generate better pseudo-labels over time. For all the experiments, we report the results from the last epoch.

4 Baseline Implementation Details

For comparing our results on Tiny ImageNet and Oxford-IIIT Pet datasets, we modify three novel class discovery methods for the open-world SSL problem: DTC [7], RankStats [6], and UNO[4]. The details of these modifications are provided in this section. For DTC[7], we extend the unlabeled head to include both known and novel classes (for more details about the unlabeled head please refer to [7]). Following ORCA [2], we perform SimCLR pretraining for RankStats[6]. After that, similar to DTC, we also extend the unlabeled head of RankStats (for more details about the unlabeled head please refer to [6]). However, neither of these methods in their original formulation assume that the unlabeled data contain samples from known classes. Therefore, extending the unlabeled head to encompass both known and novel classes does not induce any ordering (predefined known class order) for the known classes. Hence, in our evaluation we use Hungarian algorithm [10] to match the known classes from the unlabeled head with the ground-truth labels. Finally, we calculate clustering accuracy on known classes for these two methods. UNO [4] is a novel class discovery method which assumes that the unlabeled data only contain samples from novel classes. Therefore, UNO generates pseudo-labels only for novel classes. To extend UNO to open-world SSL setup, we generate pseudo-labels for both known and novel

classes. For evaluation, we concatenate the labeled and unlabeled head predictions (for more details about the concatenation strategy please refer to [4]). Similar to OpenLDN, we calculate standard accuracy on known classes for UNO. For evaluating novel classes, and the joint task of classifying both known and novel classes we calculate clustering accuracy.

Method	Known	Novel	All
FixMatch[13]	64.3	49.4	47.3
DS^{3}L[5]	70.5	46.6	43.5
DTC[7]	42.7	31.8	32.4
RankStats[6]	71.4	63.9	66.7
UNO[4]	86.5	71.2	78.9
ORCA[2]	82.8	85.5	84.1
OpenLDN-MixMatch	**92.4**$_{\uparrow 9.6}$	**93.2**$_{\uparrow 7.7}$	**92.8**$_{\uparrow 8.7}$

Table 1: Average accuracy on **CIFAR-10** dataset with 10% labeled data. We set the first 50% classes as known and the remaining 50% classes as novel. The results are averaged over three independent runs.

5 Experiments with Limited Number of Labeled Data

In this section, we conduct additional experiments on CIFAR-10 and CIFAR-100 datasets with 10% labeled data. The results on CIFAR-10 dataset are provided in Tab. 1. We observe that, similar to the results with 50% labeled data, OpenLDN significantly outperforms the closed-world SSL method, FixMatch[13], safe SSL method, DS^{3}L[5], novel class discovery methods, DTC[7], RankStats[6], and UNO[4]. We also observe that OpenLDN outperforms ORCA[2] by 7.7% on novel classes and 8.7% on joint task of classifying known and novel classes. These results suggest that the performance gap between OpenLDN and other methods is even higher when working with less amount of annotated data.

Next, we report the results on CIFAR-100 dataset with 10% labeled data in Tab. 2. Once again we observe performance improvements similar to the ones observed in the CIFAR-10 experiment. On this dataset, OpenLDN outperforms ORCA[2] by 8.2% on novel classes and 9.1% on all classes. We draw two conclusions from these results on CIFAR-10 and CIFAR-100 datasts. First, OpenLDN shows larger improvement when the amount of labeled data is limited. This feature is particularly desirable since label efficiency is one of the crucial requirement of a SSL method. Second, we observe that the improvement is higher on CIFAR-100 dataset compared to CIFAR-10 dataset, which suggests that OpenLDN can scale up to challenging datasets (higher number of classes) more efficiently.

6 SVHN Experiment

We conduct additional experiment on SVHN[12] dataset. In this experiment, we set the first 5 classes as known and the remaining classes as novel. We consider

Method	Known	Novel	All
FixMatch[13]	30.9	18.5	15.3
DS^{3}L[5]	33.7	15.8	15.1
DTC[7]	22.1	10.5	13.7
RankStats[6]	20.4	16.7	17.8
UNO[4]	53.7	33.6	42.7
ORCA[2]	52.5	31.8	38.6
OpenLDN-MixMatch	**55.0**$_{\uparrow 2.5}$	**40.0**$_{\uparrow 8.2}$	**47.7**$_{\uparrow 9.1}$

Table 2: Average accuracy on **CIFAR-100** dataset with 10% labeled data. We set the first 50% classes as seen and the remaining 50% classes as novel. The results are averaged over three independent runs.

Method	Known	Novel	All
UNO[4]	85.4	74.3	79.0
OpenLDN-MixMatch	**95.7**$_{\uparrow 10.3}$	**87.2**$_{\uparrow 12.9}$	**92.6**$_{\uparrow 13.6}$

Table 3: Accuracy on **SVHN** dataset with 10% labeled data. We set the first 50% classes as known and the remaining 50% classes as novel.

10% data from known classes as labeled. As a baseline, we conduct the same experiment with UNO[4]. The results are provided in Tab. 3. From these results, we observe that OpenLDN outperforms UNO by a large margin. To be specific, OpenLDN improves over UNO by 10.3% on known classes; an even higher improvement (12.9%) is observed on novel classes. Finally, on the joint task of classifying both known and novel classes, OpenLDN outperforms UNO by 13.6%. Results on this dataset provide additional evidence of the effectiveness of OpenLDN which can consistently outperform other existing methods on multiple datasets. It is important to note that the SVHN dataset contains a moderate level of imbalance; the dataset suffers from an imbalance factor[17] of 2.98. Therefore, the results on this dataset also demonstrates that OpenLDN works reasonably well under moderate imbalance.

Method	FGVC-Aircraf	Stanford-Cars	Herbarium19
ORCA[2]	14.7	9.6	22.9
OpenLDN-UDA	**45.7**$_{\uparrow 31.0}$	**38.7**$_{\uparrow 29.1}$	**45.0**$_{\uparrow 22.1}$

Table 4: Accuracy on **FGVC-Aircraf**, **Stanford-Cars**, and **Herbarium19** datasets with 50% labeled data. We set the first 50% classes as known and the remaining 50% classes as novel.

7 Additional Results on FGVC-Aircraft, Stanford-Cars, and Herbarium19 Datasets

We conduct additional experiments on FGVC-Aircraft[11], Stanford-Cars[9], and Herbarium19 [14] dataset. In this experiment, we set the first 50% classes as known and the remaining classes as novel. We consider 50% data from known classes as labeled. We compare our results with ORCA. We use ResNet-18 for both ORCA and OpenLDN. The results are provided in Tab. 4. On all three datasets, OpenLDN substantially outperforms ORCA. However, we request readers to interpret the ORCA results with caution since it might be possible to obtain improved results for ORCA with better hyperparameter selection. Overall, these results (4) further validate OpenLDN's effectiveness on more challenging fine-grained and imbalanced datasets.

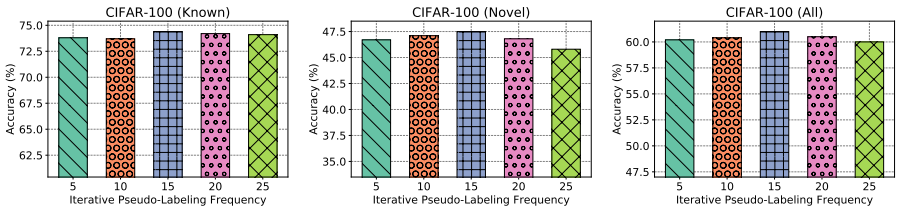

Fig. 1: The effect of changing the frequency of iterative pseudo-labeling on final accuracy. These graphs demonstrate classification accuracies on known/novel/all classes for **CIFAR-100** dataset.

8 Frequency of Iterative Pseudo-Labeling

Recall that in OpenLDN algorithm, during the second stage we generate pseudo-labels for novel classes after every 10 epochs. We introduce this iterative pseudo-labeling procedure to mitigate the negative impact of the noise present in the generated pseudo-labels after the novel class discovery phase. Here, we investigate the effect of changing the frequency of iterative pseudo-label generation process. To analyse this effect, we conduct experiments on CIFAR-100 (50% labeled data). The results are provided in Fig. 1. We observe that different frequencies for iterative pseudo-labeling lead to similar performance. This suggests that iterative pseudo-labeling is not sensitive to this hyperparameter and can improve the second stage closed-world SSL training irrespective of the frequency used. Besides, Fig. 1 demonstrates that applying a frequency of 15 leads to optimal performance. However, since we do not use any validation set to tune this hyperparameter, in our main experiments, instead, we apply our initial guess and use a frequency of 10.

References

1. Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., Raffel, C.A.: Mixmatch: A holistic approach to semi-supervised learning. In: Advances in Neural Information Processing Systems 32, pp. 5049–5059. Curran Associates, Inc. (2019) 3
2. Cao, K., Brbic, M., Leskovec, J.: Open-world semi-supervised learning. In: International Conference on Learning Representations (2022), https://openreview.net/forum?id=O-r8LOR-CCA 3, 4, 5
3. Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V.: Randaugment: Practical automated data augmentation with a reduced search space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops. pp. 702–703 (2020) 3
4. Fini, E., Sangineto, E., Lathuilière, S., Zhong, Z., Nabi, M., Ricci, E.: A unified objective for novel class discovery. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9284–9292 (2021) 3, 4, 5
5. Guo, L.Z., Zhang, Z.Y., Jiang, Y., Li, Y.F., Zhou, Z.H.: Safe deep semi-supervised learning for unseen-class unlabeled data. In: International Conference on Machine Learning. pp. 3897–3906. PMLR (2020) 4, 5
6. Han, K., Rebuffi, S.A., Ehrhardt, S., Vedaldi, A., Zisserman, A.: Automatically discovering and learning new visual categories with ranking statistics. In: International Conference on Learning Representations (2020) 3, 4, 5
7. Han, K., Vedaldi, A., Zisserman, A.: Learning to discover novel visual categories via deep transfer clustering. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8401–8409 (2019) 3, 4, 5
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) 3
9. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13). Sydney, Australia (2013) 6
10. Kuhn, H.W.: The hungarian method for the assignment problem. Naval research logistics quarterly **2**(1-2), 83–97 (1955) 3
11. Maji, S., Kannala, J., Rahtu, E., Blaschko, M., Vedaldi, A.: Fine-grained visual classification of aircraft. Tech. rep. (2013) 6
12. Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y.: Reading digits in natural images with unsupervised feature learning (2011) 4
13. Sohn, K., Berthelot, D., Li, C.L., Zhang, Z., Carlini, N., Cubuk, E.D., Kurakin, A., Zhang, H., Raffel, C.: Fixmatch: Simplifying semi-supervised learning with consistency and confidence. arXiv preprint arXiv:2001.07685 (2020) 4, 5
14. Tan, K.C., Liu, Y., Ambrose, B., Tulig, M., Belongie, S.: The herbarium challenge 2019 dataset. Workshop on Fine-Grained Visual Categorization (2019) 6
15. Xie, Q., Dai, Z., Hovy, E., Luong, M.T., Le, Q.V.: Unsupervised data augmentation for consistency training. arXiv preprint arXiv:1904.12848 (2019) 3
16. Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D.: mixup: Beyond empirical risk minimization. In: International Conference on Learning Representations (2018) 3
17. Zhou, B., Cui, Q., Wei, X.S., Chen, Z.M.: Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9719–9728 (2020) 5