Semi-supervised learning made simple with self-supervised clustering
Supplementary material

Enrico Fini*1 Pietro Astolfi*1,2 Karteek Alahari2 Xavier Alameda-Pineda2 Julien Mairal2 Moin Nabi3 Elisa Ricci1,4
1 University of Trento 2 Inria† 3 SAP AI Research 4 Fondazione Bruno Kessler

1. More implementation details

1.1. ImageNet-1k

Pre-training. On ImageNet-1k, we train Suave with ResNet-50 backbone a projection head with hidden and output dimensions 2048 and 128, respectively. The number of prototypes is set to 1000 as the number of classes. We train with mini-batches composed of 256 unlabeled and 128 labeled images. The training lasts for 100 epochs; each epoch consumes all the unlabeled images once. We optimize using LARS with a learning rate of linearly increased from 0 to 0.4 throughout 5 epochs and then decreased to 0.001 with a cosine scheduler. The cross-entropy loss is regularized with a weight decay of $10^{-6}$. Also, the ground-truth labels are smoothed with a factor of 0.01. The pseudo-labels, instead, are computed via three iterations of the Sinkhorn-Knopp algorithm [4] applied to the detached logits (output of the network) extended with a queue of 3840 embeddings buffered from previous mini-batches. The logits used for pseudo labeling are first peaked using a temperature (ϵ parameter) of 0.05, while the logits used as predictions are peaked with a temperature of 0.1 before computing the loss. On the unlabeled images, we use multi-crop [2] with two large crops (random crop range (0.14, 1)) of size 2242 and eight small crops (random crop range (0.08, 0.14)) of size 962. We extend each batch with mixed images generated from MixUp [8, 9] with probability 1.0, applying either MixUp or CutMix with probability 0.5 and degree of mixing (known as lambda) drawn from Beta(1, 1). The augmentation recipe of unlabeled images is the exact same as SwAV [2] (color jittering with intensity 0.8 and probability 0.8, random grayscale with probability 0.2, and Gaussian blurring with probability 0.5). For labeled images, the Inception-style [7] augmentations adopted consist of random cropping with range (0.08, 1), horizontal flip with probability 0.5, color jittering with intensity 0.4 and probability 0.8, and grayscale with probability 0.2.

Fine-tuning. The fine-tuning runs for 3/5 epochs when using 1%/10% of the labels with the same semi-supervised setting of the pre-training. Note that the hyper-parameters are kept the same unless specified in the following. The network is fully initialized with the pre-trained weights, except for the prototypes layer, which is randomly initialized. We adopt a smaller learning rate, 0.02, with no linear warm-up and a final value of 0.0002 after cosine decreasing. Also, we reduce the intensity of the augmentations; on the labeled images, we reduce color jittering intensity to 0.1 (keeping probability 0.8) and disable grayscale; on the unlabeled, we turn off multi-crops, generating only two crops per image with crop range of (0.08, 1), and drop off the color distortions and the blurring.

Simplified training recipe for Daino. For Daino experiments with ViT-S/16 backbone [5] we adopt the default DINO [3] pre-training recipe1 for most hyper-parameters except for a few modification that we report in the following. We perform semi-supervised pre-training for 60 epochs, initializing the ViT backbone weights with the DINO pre-trained ones (800 epochs checkpoint); the teacher momentum is set to 0.990; the teacher temperature is raised from 0.04 to 0.07 during the first 10 epochs; the student temperature is fixed to 0.1; we do not freeze the last layer because the labeled loss help to avoid clustering collapse; we set the learning rate to 0.00024 and warm it up linearly for the first 4 epochs; each mini-batch is composed of 1024 unlabeled and 512 labeled images; the labeled images are extended using MixUp [8, 9] with probability 1.0 as in the Suave recipe; we augment the unlabeled images with multi-crop obtaining two large crops (crop range (0.25, 1)) and eight small crops (crop range (0.05, 0.25)); other data augmentations are maintained as in the original DINO. Note that no fine-tuning is explored for Daino.

---

1Enrico Fini and Pietro Astolfi contributed equally.
2Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, 38000 Grenoble, France.

1see https://dl.fbaipublicfiles.com/dino/dino_deitsmall16_pretrain/args.txt
1.2. CIFAR100

For CIFAR100 we use a slightly different recipe with respect to ImageNet. First, we do not perform fine-tuning (neither supervised nor semi-supervised), as we found that it does not improve performance. Semi-supervised training is performed with unlabeled batch size 128 and labeled batch size 100 for 200 epochs. For both Suave and Daino, the backbone is initialized using weights obtained by unsupervised training of SwAV for 500 epochs on the same dataset. In addition, we use multi-crop with 4 local crops of size (0.1, 0.6) and 2 global crops of size (0.6, 1.0). Similarly to ImageNet, we use label smoothing with coefficient 0.01. The learning rate for LARS is set to 2.8 and a weight decay of $3 \times 10^{-6}$ is applied. The $\epsilon$ coefficients are set to 0.086 and 0.07 for Suave and Daino respectively. For both methods we also use a momentum encoder with momentum 0.99. We apply image mixing techniques as data augmentation as for ImageNet, with the only difference that we also mix local crops on CIFAR100. All the other hyperparameters are kept the same as described before.

2. Additional results

We present further comparisons with the state-of-the-art in Sec. 2.1 and show additional visualizations in Sec. 2.2.

2.1. Pre-training results

In Tab. 1 we report results on ImageNet-1k after semi-supervised pretraining (without fine-tuning) using the same classifier as the one that was trained during pre-training (PAWS uses a nearest-neighbor classifier, we use a linear classifier). The results clearly show that, despite a much smaller batch size, Suave is able to match or outperform PAWS, even without fine-tuning.

2.2. Latent representations

In Fig. 1, we report the real-data counterpart of Figure 1 of the main paper, computed with UMAP [6]. The latent vectors are taken from the bottleneck layer (output of the projection head) of the models trained with 1% of the labels. All the models are initialized with SwAV pre-trained at 800 epochs. We observe a neat difference between (a), where classes are less isolated/separable, and (b-c), where, instead, classes are well separated. Moreover, by visually comparing (b) and (c), we notice a slightly better class separation obtained by Suave (c). However, we remark that the random classes shown may not highlight the difference of the models at best, as Suave outperforms SwAV-finetuned of $\sim 9\%$.
References


