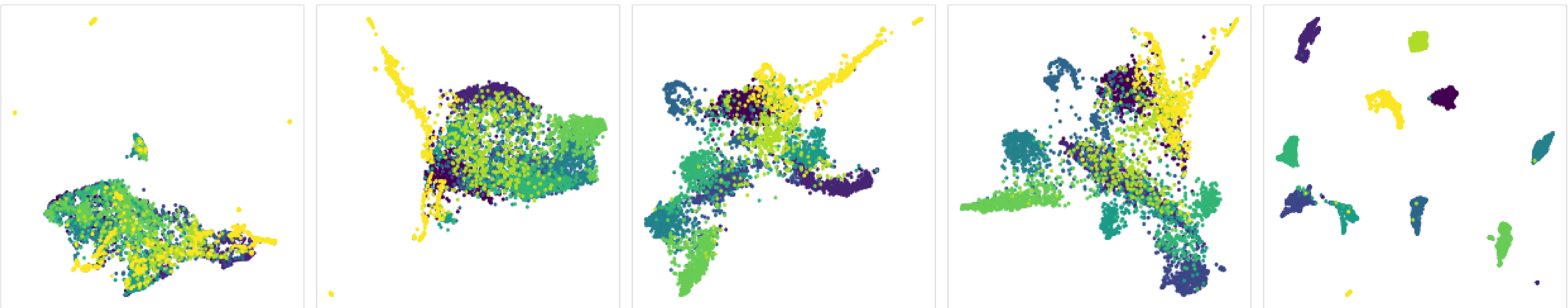
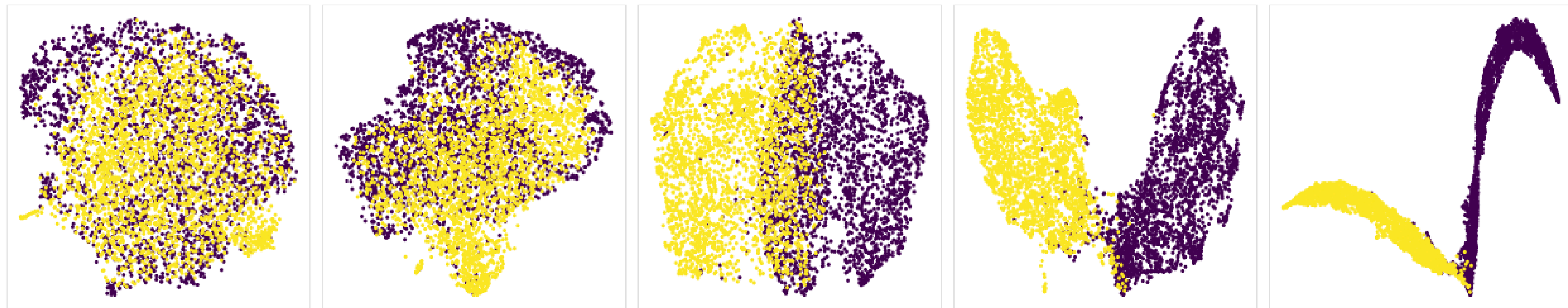
Improving the Fine-Tuning of Image Classifiers with Latent Cluster Correction

Cédric HT

RIKEN



EuroSAT samples passed through VGG11



Cat and dogs image samples passed through AlexNet

TL;DR

- Neural networks take input samples and transform them into **latent representations**
- Semantically similar samples tend to aggregate into **latent clusters**
- We develop **Latent Cluster Correction** to improve them

`https://github.com/altaris/lcc`

Latent spaces

In its simplest form, a neural network is a sequence of elementary function (aka layers) $F(x) = f_n(f_{n-1}(\cdots f_1(x)\cdots))$. A **latent representation** is an embedding of the form

$$z = f_m(\cdots f_1(x)\cdots)$$

for some intermediate value $1 \leq m \leq n$.

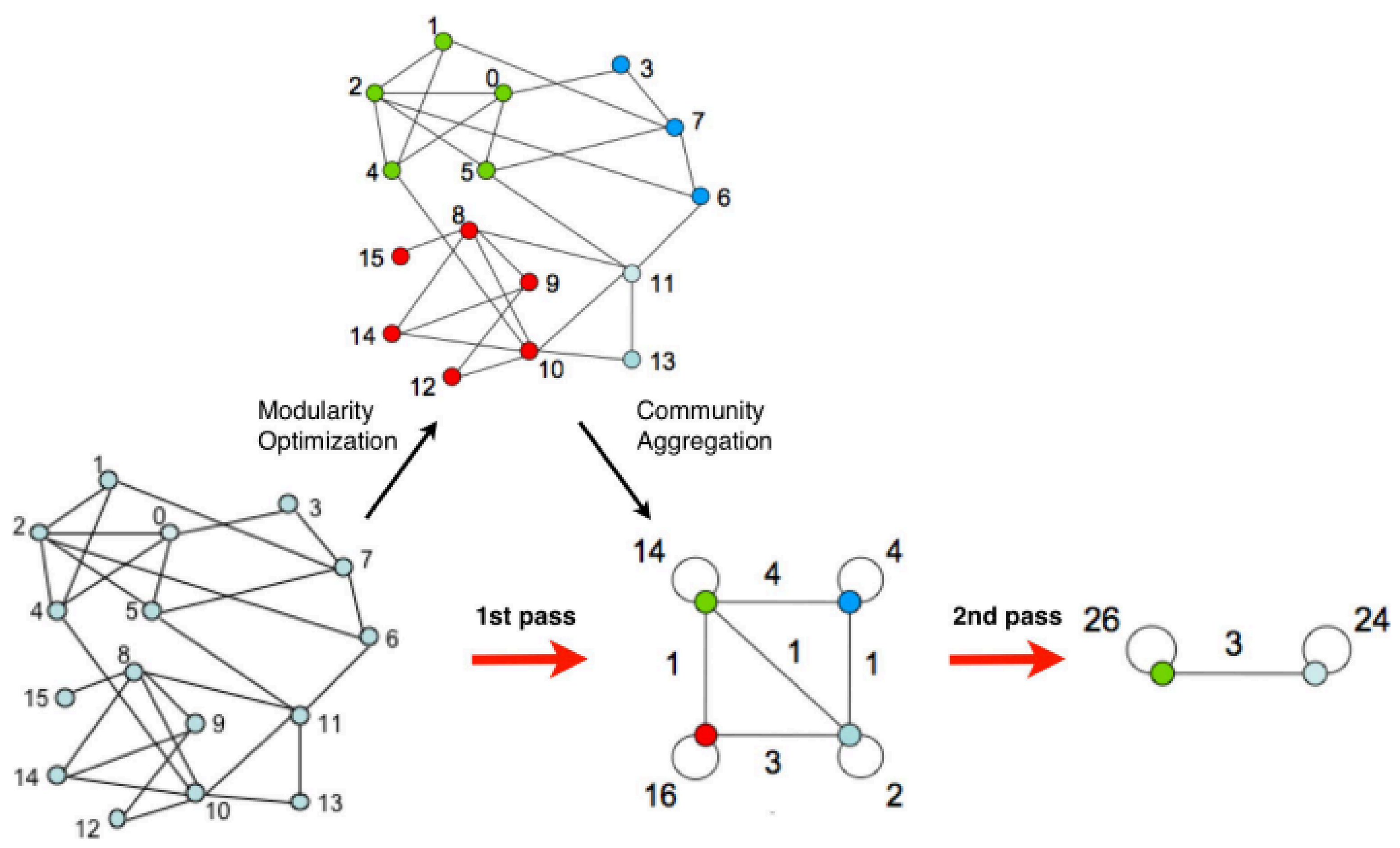
- For small m's (i.e. shallow latent spaces), $Z = f_m(\cdots f_1(X)\cdots)$ is pretty scrambled.
- For large m's (i.e. deep latent spaces), Z is more structured and tends to form clusters of semantically similar samples.

Goal

1. Choose a deep **latent** space. Find the clusters in $Z = f_m(\cdots f_1(X)\cdots)$.
2. Find latent representations that are not in the **cluster** they should be in.
3. "**Correct**" them by penalizing the distance to their closest acceptable cluster.

Step 1: Detect latent clusters

We use the k**NN-Louvain** method, which computes the symmetric and unweighted kNN graph of Z, and applies the **Louvain-Leiden community detection algorithm** to it.



Blondel et. al., Fast unfolding of communities in large networks

Step 2: Match clusters to true classes

How do we compare true classes with latent clusters obtained in step 1? We find a **matching** $M : \{\text{Latent clusters}\} \to \{\text{True classes}\}$ that maximizes

$$\sum_{\ell \text{ true class}} \left| \{z_n \mid y_n = \ell\} \cap \{z_n \mid M(c_n) = \ell\} \right|$$
$$= \sum_{\ell \text{ true class}} \left| \{\text{Smpls. in } \ell\} \cap \{\text{Smpls. in clsts. assigned to } \ell\} \right|$$

where y_n is the true class of z_n and c_n is the cluster of z_n.

There are two types of samples:

- **Correctly clustered**: $M(c_n) = y_n$, i.e. z_n is in a cluster assigned to its class $\to$ these are OK!
- **Missclustered**: $M(c_n) \neq y_n$, i.e. z_n is in a cluster assigned to another class $\to$ we need to correct them by **pushing them towads a cluster that belongs to their class**.

Step 3: Apply the Louvain loss

The **Louvain loss** is defined as

$$\mathcal{L}_{\text{Louvain}} = \frac{1}{N_{\text{miss}}\sqrt{d}} \sum_{z \text{ miss}} \left\| z - T(z) \right\|^2$$

where $T(z) = (z_1 + \cdots + z_k)/k$, and $z_1, \ldots, z_k$ are the kNNs of z **in the same class as** z **and that are correctly clustered**. The training loss function is then $\mathcal{L} = \mathcal{L}_{\text{CE}} + w\mathcal{L}_{\text{Louvain}}$

Step 4: Profit

Models pretrained on ImageNet and fined-tuned on CIFAR100 using LCC with $w = 0.01$.

Model	Layer	k	Wmp.	Acc. (top 1)	Gain (top 1)	Acc. (top 5)	Gain (top 5)
AlexNet	Baseline			71.49%	/	92.08%	/
	Head	5	1	71.6%	+0.11%	91.73%	−0.35%
	2^{nd} to last	5	1	71.36%	−0.13%	91.72%	−0.36%
	Head	5	10	71.61%	+0.12%	91.56%	−0.52%
	2^{nd} to last	5	10	71.35%	−0.14%	91.91%	−0.17%
	Head	50	1	71.82%	+0.33%	91.68%	−0.4%
	2^{nd} to last	50	1	71.54%	+0.05%	91.71%	−0.37%
	Head	50	10	71.62%	+0.13%	91.74%	−0.34%
	2^{nd} to last	50	10	71.29%	−0.2%	91.7%	−0.38%
ResNet18	Baseline			78.51%	/	96.06%	/
	Head	5	1	79.28%	+0.77%	95.62%	−0.44%
	2^{nd} to last	5	1	79.02%	+0.51%	95.71%	−0.35%
	Head	5	10	79.23%	+0.72%	95.62%	−0.44%
	2^{nd} to last	5	10	79.24%	+0.73%	95.85%	−0.21%
	Head	50	1	79.24%	+0.73%	95.31%	−0.75%
	2^{nd} to last	50	1	79.47%	+0.96%	95.84%	−0.22%
	Head	50	10	79.06%	+0.55%	95.5%	−0.56%
	2^{nd} to last	50	10	79.12%	+0.61%	95.76%	−0.3%