SCARF: Self-Supervised Contrastive Learning using Random Feature Corruption

Reminder

- Self-supervised learning – before solving the actual task, solve a task based on pseudo-labels to initialize network weights.
- Contrastive learning – type of self-supervised learning:
  - 1) uses data augmentations;
  - 2) compares the views of the original and augmented objects;
  - 3) the views of the objects derived from the same object must be similar; from different objects – different.

Main idea

- Contrastive learning is most often applied in CV:
    - crop, resize, color change, etc.
- How can we apply it to tabular data?

Main idea

- Contrastive learning is most often applied in CV:
  - crop, resize, color change, etc.
- How can we apply it to tabular data?
- We may corrupt data to create augmentations.

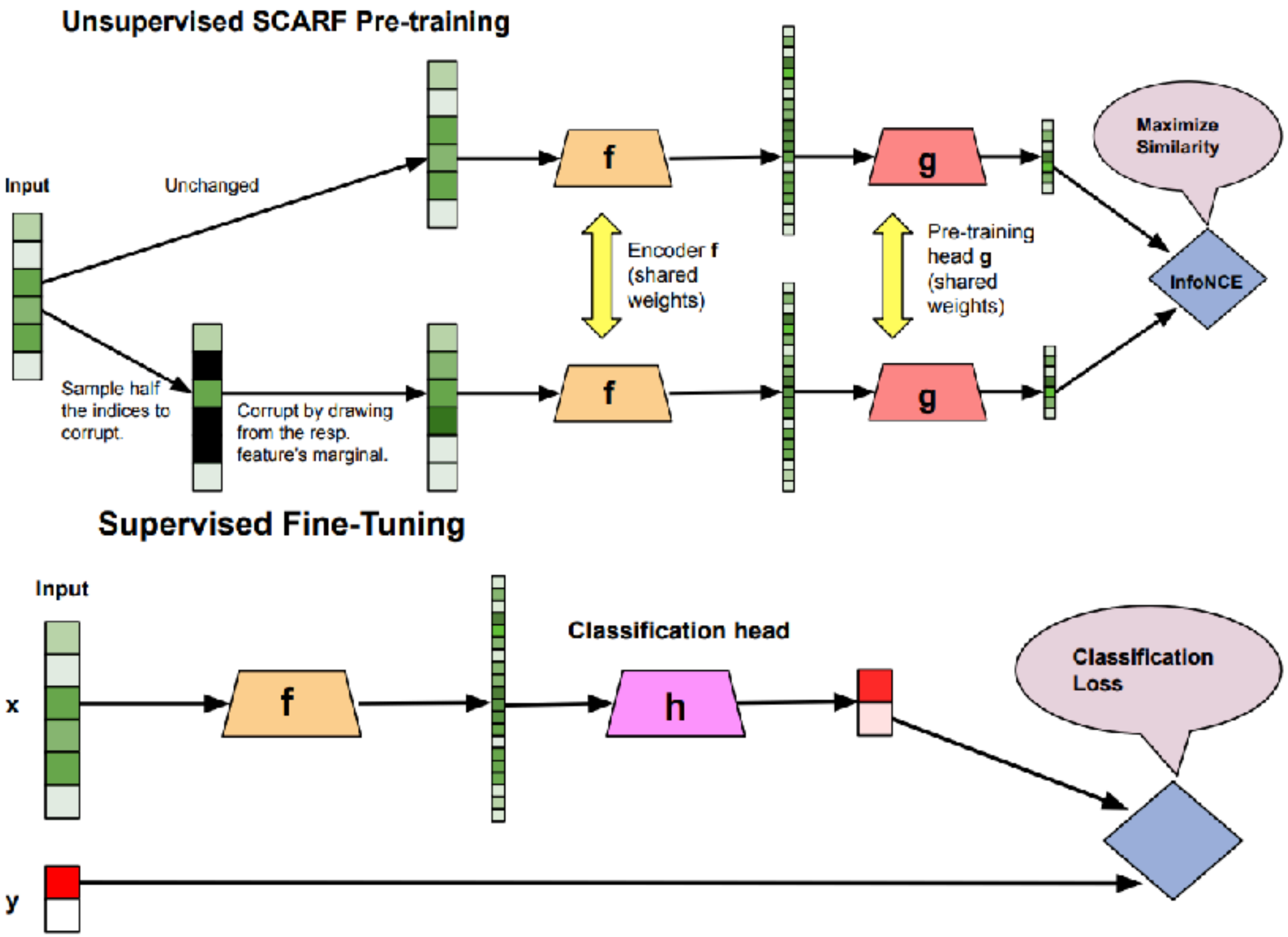
Data corruption

- Additive Gaussian noise
- Mean corruption: mean value
- Missing feature corruption: special learnable value
- Feature dropout: zero out

Data corruption

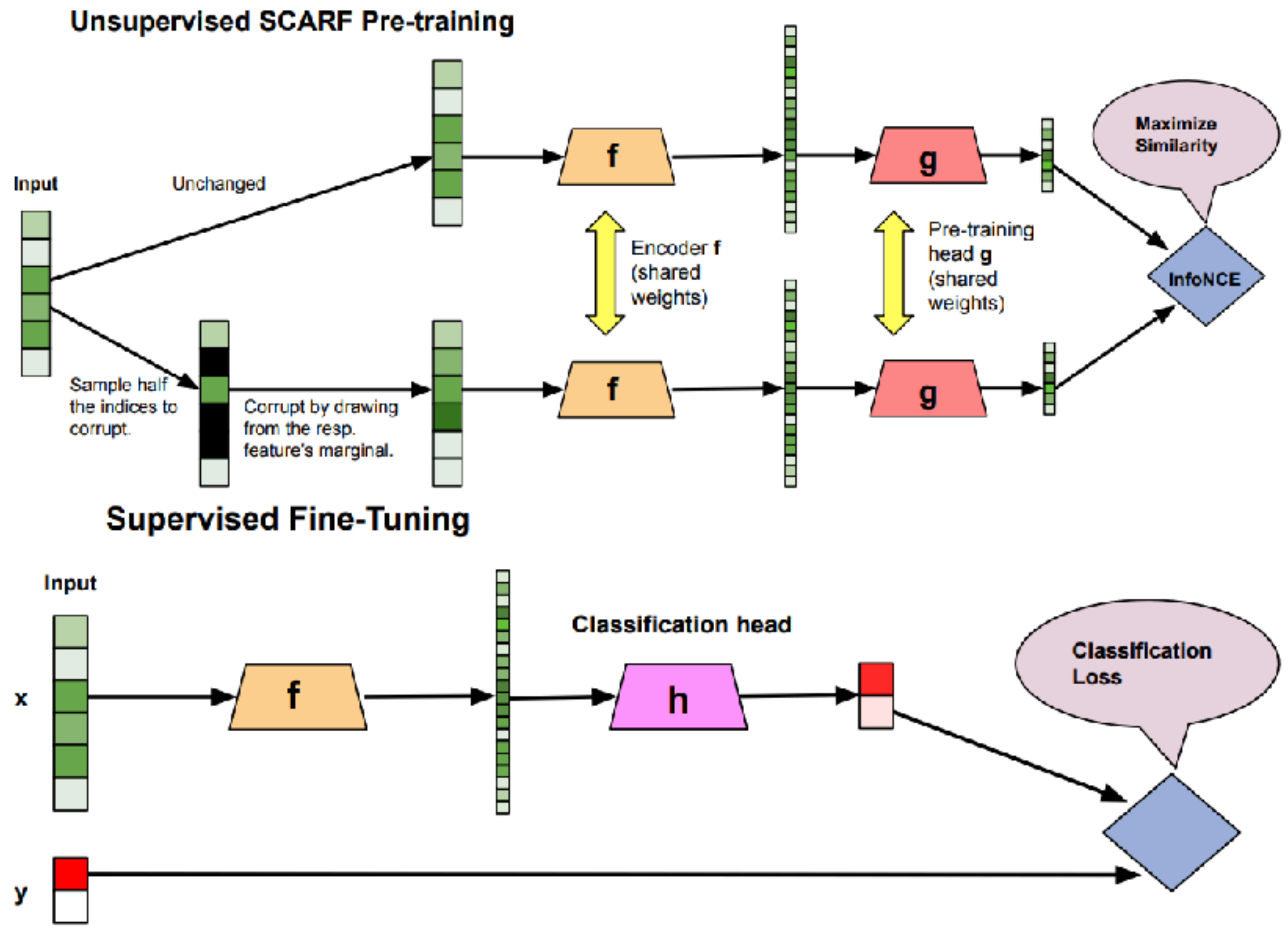
- Replace feature with random draws:
  - Joint sampling: from empirical joint distribution based on the whole train set.
  - Scarf: from empirical marginal uniform distribution over the feature's values.

Scarf: general model



From: https://arxiv.org/pdf/2106.15147.pdf

Scarf: general model



From: https://arxiv.org/pdf/2106.15147.pdf

- Note:
  - we continue training f during fine-tuning;
  - better choose features for each example separately;
  - a fixed number of features are chosen from the uniform distribution.

InfoNCE loss

$$\frac{1}{N}\sum_{i} - \log(\frac{exp(s_{i,i}/\tau)}{\frac{1}{N}\sum_{k} exp(s_{i,k}/\tau)})$$

- N – size of mini-batch
- $s_{i,j}$ – cosine similarity between views of original x_i and distorted $\tilde{x}_j$.
- τ – hyperparameter (actually not important and can be simply set to 1).

InfoNCE loss

$$\frac{1}{N}\sum_{i} -\log\left(\frac{exp(s_{i,i}/\tau)}{\frac{1}{N}\sum_k exp(s_{i,k}/\tau)}\right)$$

- Minimization with SGD.
- Best result is achieved when:
  - x_i is similar to $\tilde{x}_i$ ($s_{i,i} = 1$);
  - x_i is different from $\tilde{x}_j, j \neq i$ ($s_{i,j} = -1$).

Scarf advantages

- Scarf is not really sensitive to:
  - batch size (recommended = 128);
  - corruption rate (recommended = 60%);
  - temperature (recommended = 1).

Experiment results

- Win matrix:
  - $W_{i,j} = \frac{\sum_{d=1}^{69} \mathbb{1}[\text{method } i \text{ beats } j \text{ on dataset } d]}{\sum_{d=1}^{69} \mathbb{1}[\text{method } i \text{ beats } j \text{ on dataset } d] + \mathbb{1}[\text{method } i \text{ loses to } j \text{ on dataset } d]}.$
  - 0/69 is better than 0/1.

Other models

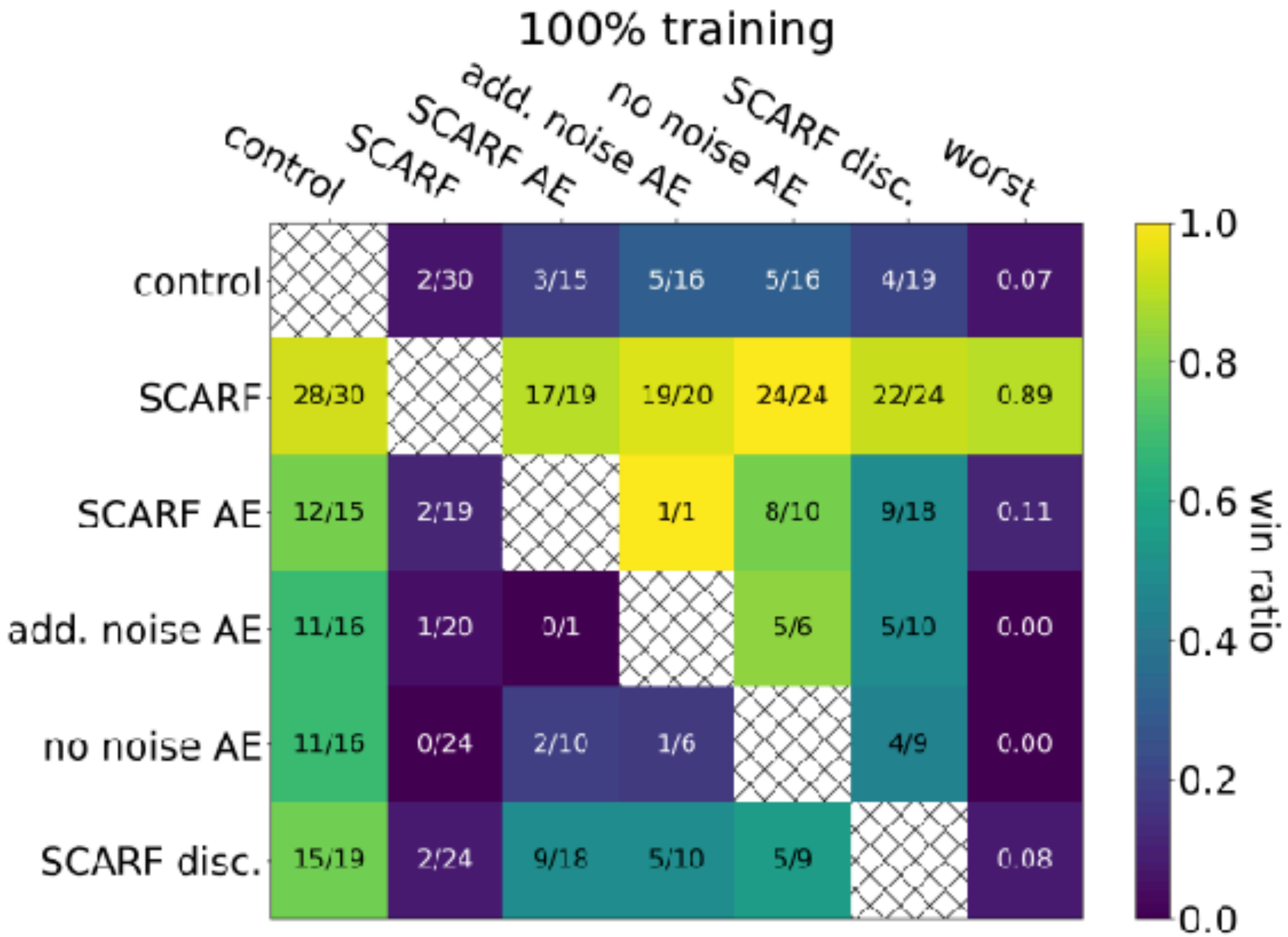
- Further, we compare Scarf with some other pretraining methods:
  - no noise AE (autoencoder);
  - add. noise AE (denoising autoencoder with input corrupted with Gaussian noise: output is compared with original);
  - Scarf AE (denoising autoencoder with input corrupted with method similar to Scarf: output is compared with original);
  - Scarf disc. (discriminative Scarf: classification: pre-training objective is to discriminatie between original input features and their corrupted counterparts).

Fine-tuning models

- In our 1st experiment all labels are known. Fine-tuning models are:
  - control (usual supervised learning);
  - dropout;
  - mixup (trains a neural network on convex combinations of pairs of examples and their labels: see source 2);
  - label smoothing (regularization method: see source 3);
  - distill (self-distillation: we train model on labeled data and then train again, adding previously unlabeled data with all labels as output of the 1st model);

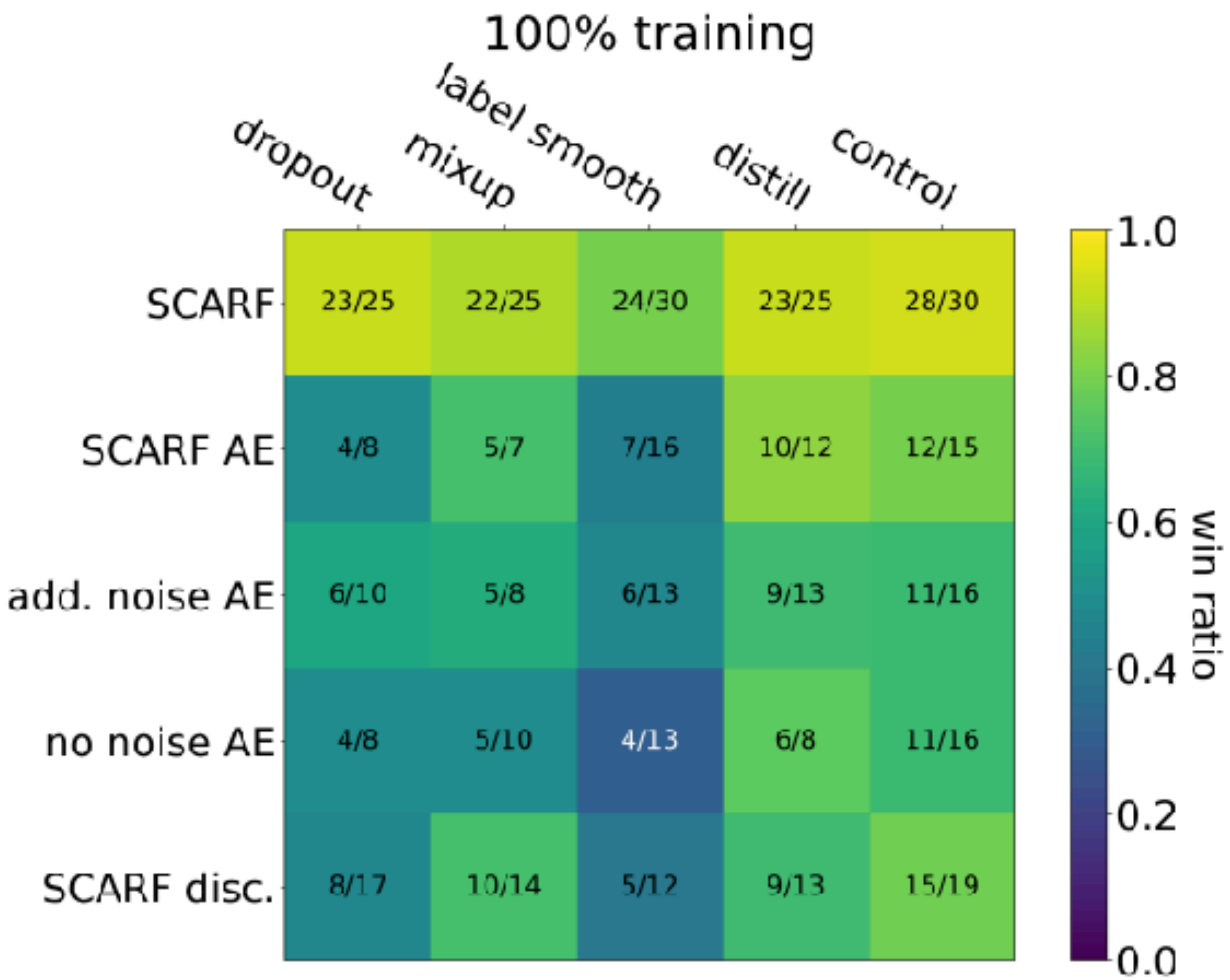
Experiment results: known labels

Just method



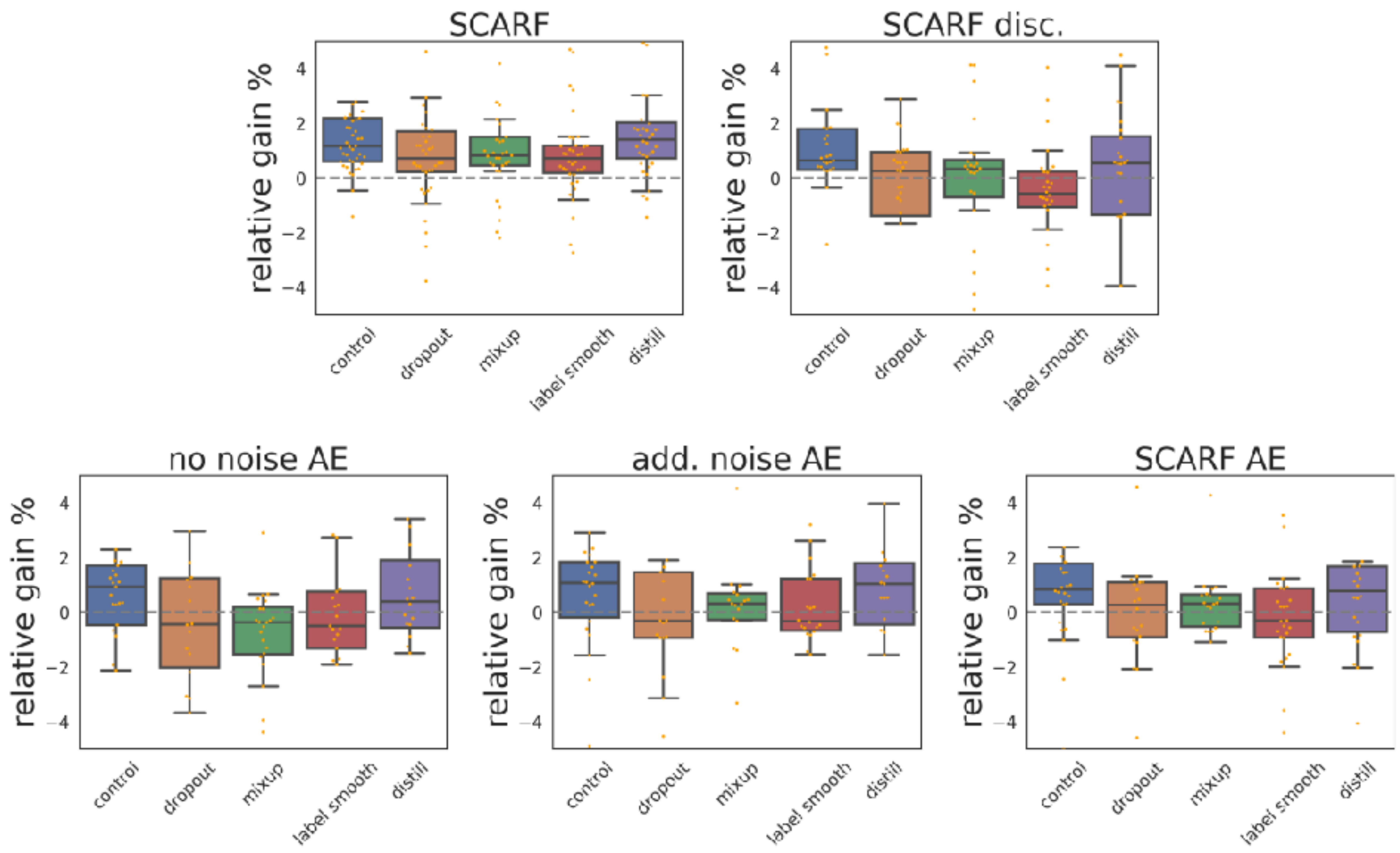
Here we compare pretraining methods;
in fine tuning we use usual supervised learning.

Method + sth:



Model we use in fine tuning.

Experiment results: known labels

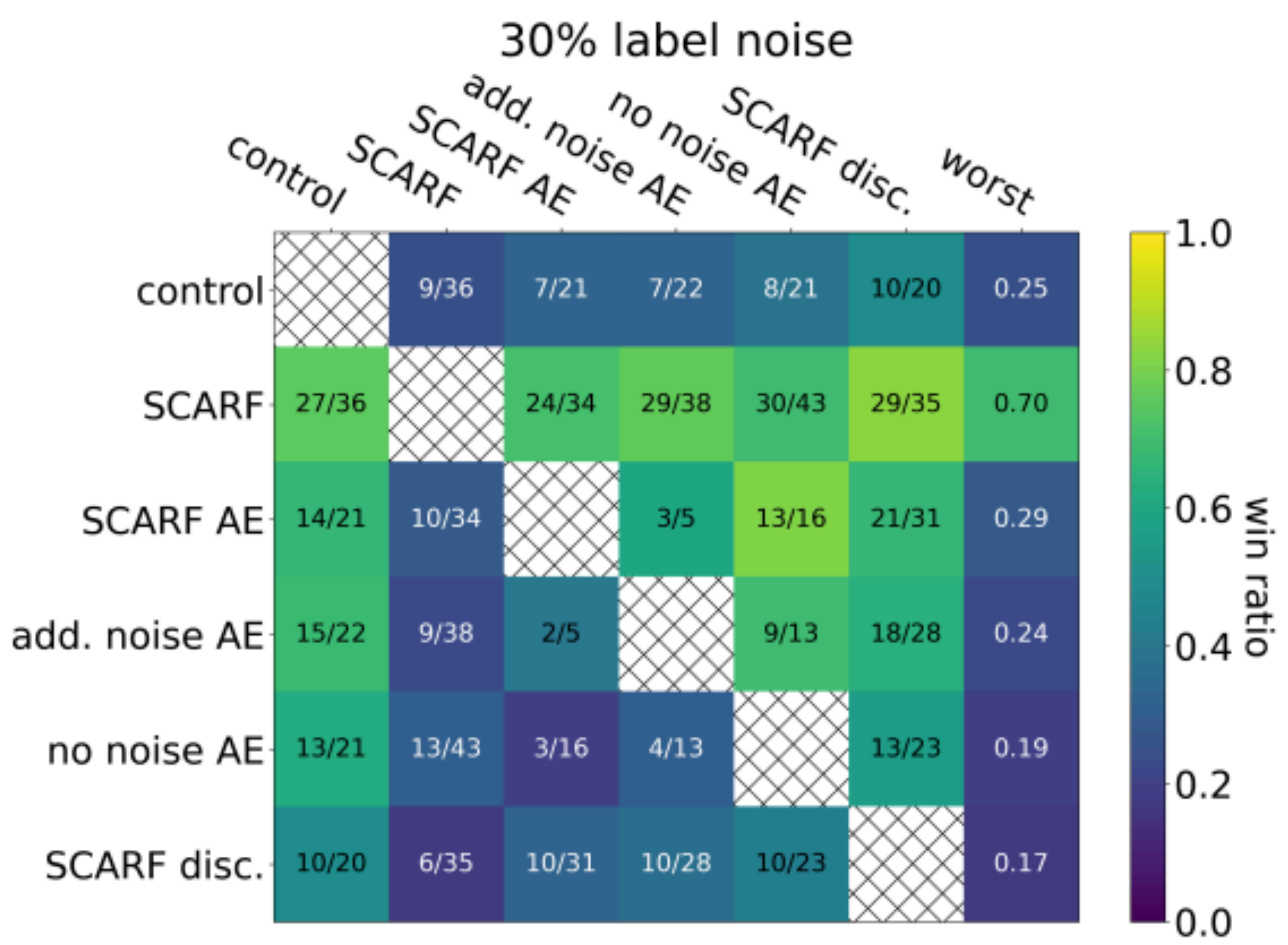


Fine-tuning models

- In our 2st experiment 30% labels are corrupted. Additional fine-tuning models (specifically created for this task) are:
  - bitempered (uses special loss function: see source 4);
  - deep kNN (method which combines kNN with neural network: see source 5).

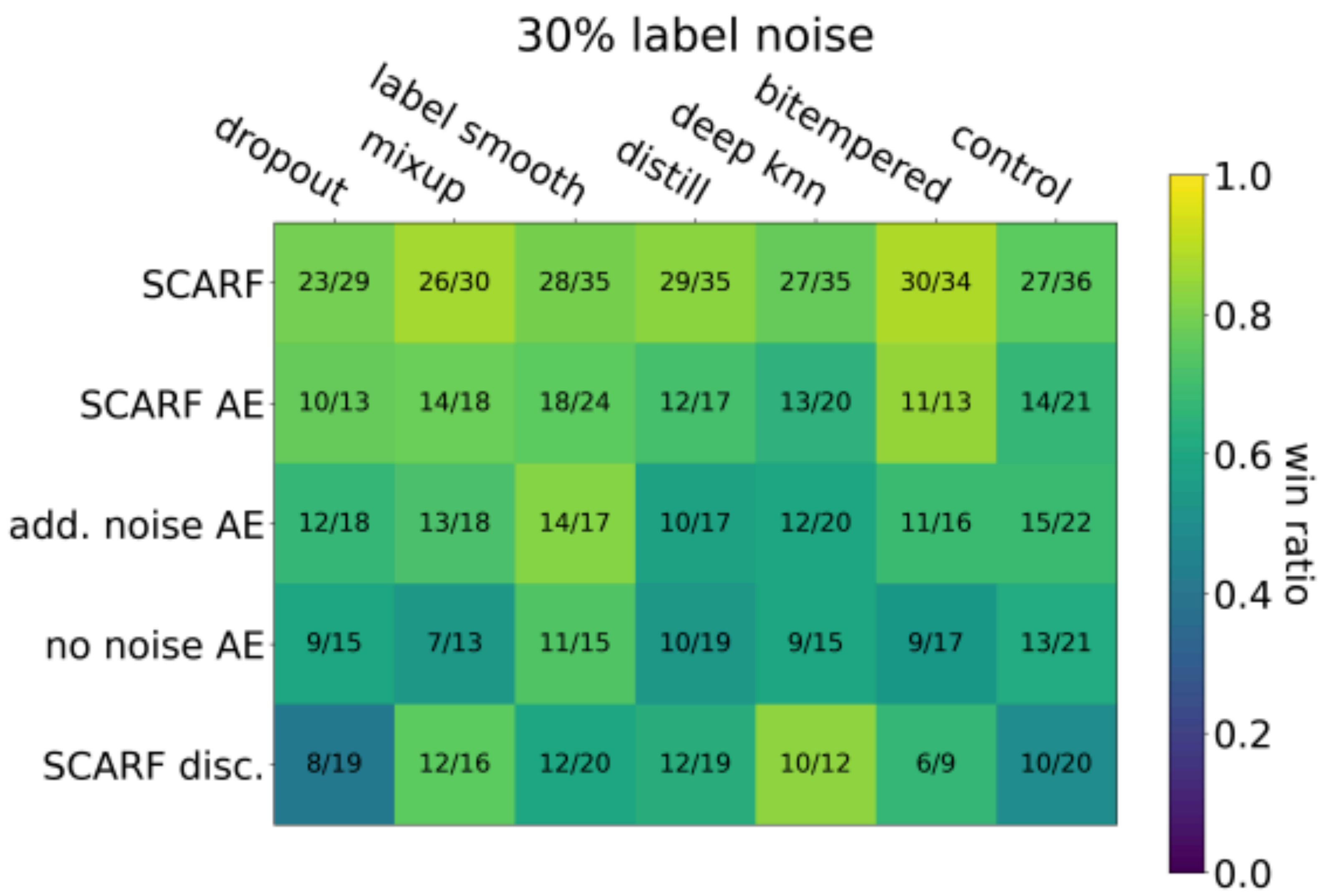
Experiment results: 30% labels corrupted

Just method



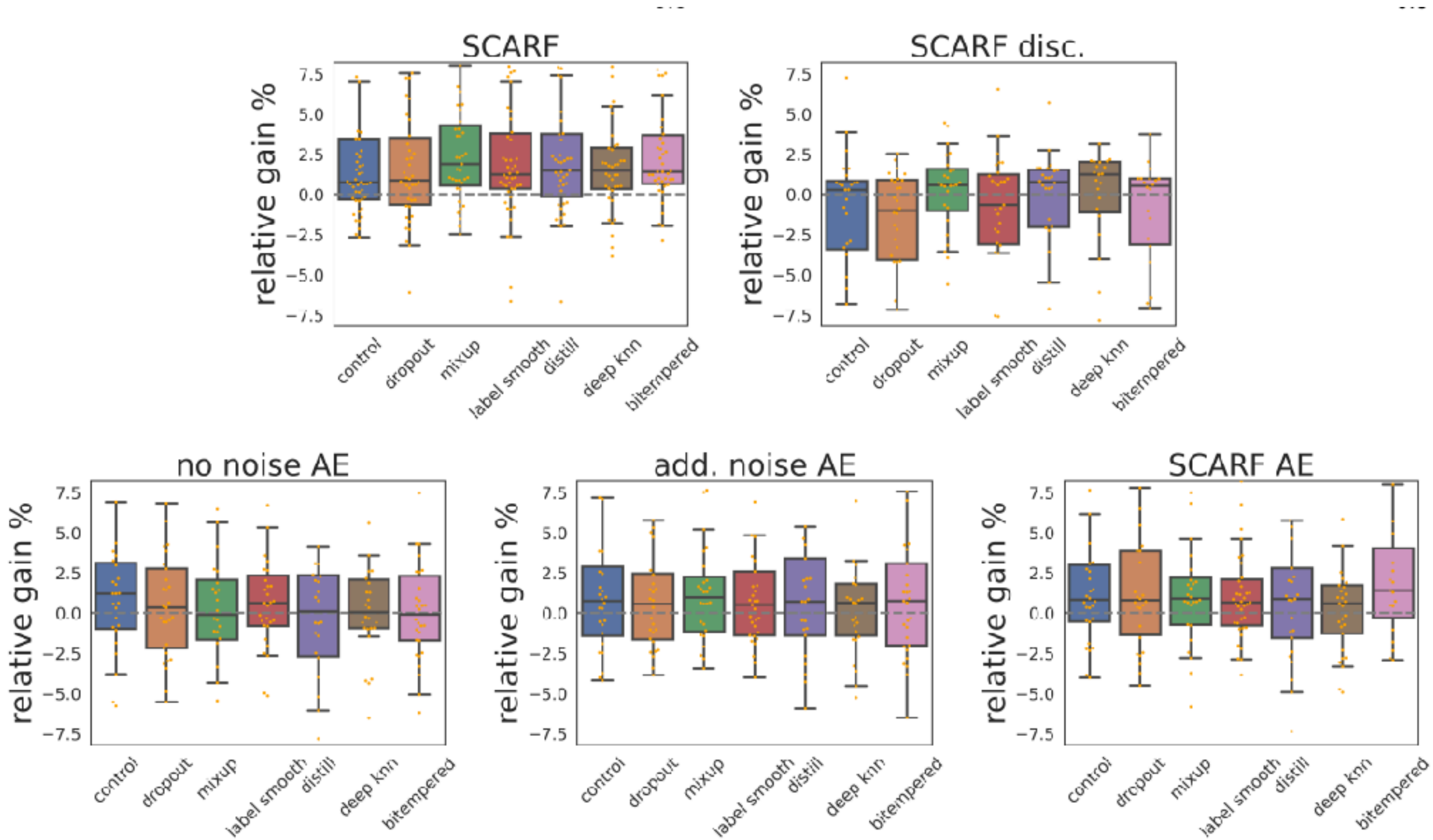
Again, here we compare pretraining methods;
in fine tuning we use common supervised learning.

Method + sth:



Again, model we use in fine tuning.

Experiment results: 30% labels corrupted



Fine-tuning models

- In our 3rd experiment only 25% labels in the training set are available. We don’t have win matrices or box plots for it.s
- Additional fine-tuning models (specifically created for this task) are:
    - self-training (each iteration, we train on pseudo-labeled data (initialized to be the original labeled dataset) and add highly confident predictions to the training set using the prediction as the label);
    - tri-training (like self-training, but using three models with different initial labeled data via bootstrap sampling; each iteration, every model’s training set is updated by adding only unlabeled points whose predictions made by the other two models agree).

100% labeled training	SCARF	SCARF AE	no noise AE	add. noise AE	SCARF disc.
control	**2.352**	2.244	1.107	1.559	0.574
dropout	**1.609**	1.196	0.623	1.228	-1.312
mixup	**1.72**	1.183	-0.377	0.971	-0.307
label smooth	**1.522**	0.711	-0.002	1.04	-0.894
distill	**2.392**	2.186	0.823	1.431	-0.394
25% labeled training					
control	**3.692**	1.702	0.777	1.662	0.233
dropout	**2.212**	1.848	2.013	1.155	-0.322
mixup	**2.809**	0.73	0.106	0.439	0.466
label smooth	**2.303**	0.705	-0.564	0.196	-0.206
distill	**3.609**	2.441	1.969	2.263	1.795
self-train	**3.839**	2.753	1.672	2.839	2.559
tri-train	**3.549**	2.706	1.455	2.526	1.92
30% label noise					
control	**2.261**	1.988	0.914	1.612	-1.408
dropout	2.004	**2.058**	0.9	1.471	-2.54
mixup	**2.739**	1.723	0.116	1.409	0.189
label smooth	**2.558**	1.474	0.703	1.395	-1.337
distill	**2.881**	2.296	-0.239	1.659	-0.226
deep knn	**2.001**	1.281	0.814	1.348	0.088
bitempered	2.68	**2.915**	0.435	1.387	-1.147

Shown is the average relative gain in accuracy when adding the pre-training methods (columns) to the reference methods (rows).

Sources

1. Original article: https://arxiv.org/pdf/2106.15147.pdf

2. Mixup: https://arxiv.org/pdf/1710.09412.pdf

3. Label smoothing: https://towardsdatascience.com/what-is-label-smoothing-108debd7ef06

4. Bitempered: https://arxiv.org/pdf/1906.03361.pdf

5. Deep kNN: https://arxiv.org/pdf/2004.12289.pdf