- Appendices -
Dataset Similarity to Assess Semi-supervised Learning Under Distribution Mismatch Between the Labelled and Unlabelled Datasets

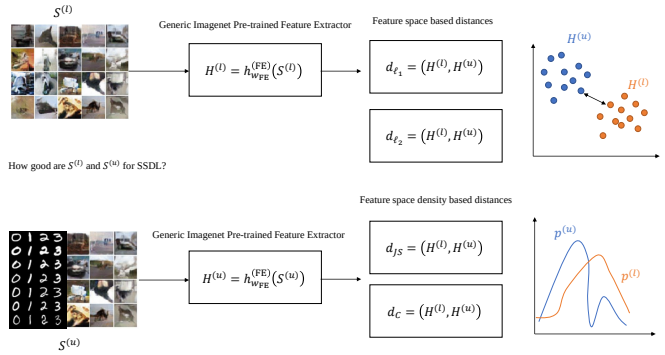

Fig. 1. Summary of the proposed and tested dataset comparison measurements.

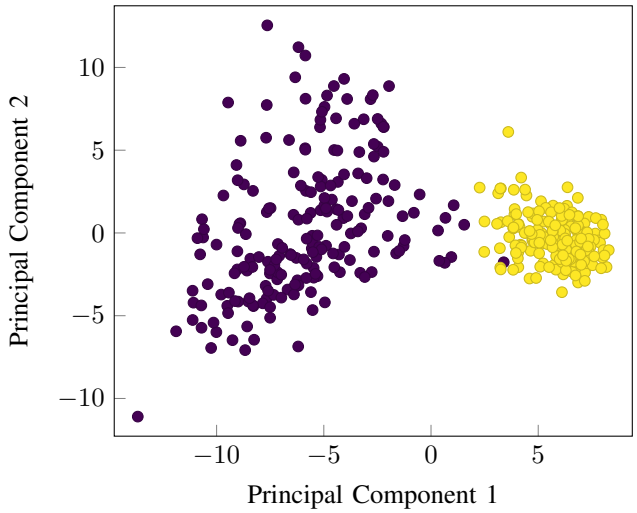

Fig. 2. Example of a 2D projection of CIFAR-10 and ImageNet datasets to the feature space of an AlexNet model. The projections to the feature space of two different datasets, result in different data distributions. To visualize the data, the two principal components are used. Using a dataset $S^{(l)}$ and $S^{(u)}$ with very different distributions might harm SSDL performance.

Appendix A
MixMatch: Detailed description of the SSL algorithm used in this paper

In MixMatch, the consistency loss term minimizes the distance of the pseudo-labels and the model predictions over the unlabelled dataset X_u. Pseudo-label $\widehat{\boldsymbol{y}}_j$ estimation is performed with the average model output of a transformed input x_j, with K number of different transformations. $K = 2$ is advised in [1]. The estimated pseudo-labels $\widehat{\boldsymbol{y}}$ might be too *unconfident*. To tackle this, pseudo-label sharpening is performed with a temperature ρ. The dataset with the estimated and sharpened pseudo-labels was defined as $\widetilde{S}_u = \left(X_u, \widetilde{Y}\right)$, with $\widetilde{Y} = \left\{\widetilde{\boldsymbol{y}}_1, \widetilde{\boldsymbol{y}}_2, \ldots, \widetilde{\boldsymbol{y}}_{n_u}\right\}$.

Data augmentation is a key aspect in semi-supervised deep learning as found in [1]. To further augment data using both labelled and unlabelled samples, they implemented the Mix Up algorithm developed in [2]. Linear interpolation of a mix labelled observations and unlabelled (with its corresponding pseudo-labels) observations.

$$\left(S_l', \widetilde{S}_u'\right) = \Psi_{\text{MixUp}}\left(S_l, \widehat{S}_u, \alpha\right) \tag{1}$$

The Mix Up algorithm creates new observations from a linear interpolation of a mix of unlabelled (with its corresponding pseudo-labels) and labelled data. More specifically, it takes two labelled (or pseudo labelled) data pairs $(\boldsymbol{x}_a, y_a)$ and $(\boldsymbol{x}_b, y_b)$. The Mix Up method generates a new observation and its label $(\boldsymbol{x}', y')$ by following these steps:

1) Sample the Mix Up parameter λ from a Beta distribution $\lambda \sim \text{Beta}(\alpha, \alpha)$.
2) Ensure that $\lambda > 0.5$ by making $\lambda' = \max(\lambda, 1-\lambda)$.
3) Create a new observation with a lineal interpolation of both observations: $\boldsymbol{x}' = \lambda' \boldsymbol{x}_a + (1-\lambda')\,\boldsymbol{x}_b$.

With the augmented datasets $\left(S_l', \widetilde{S}_u'\right)$, the MixMatch training can be summarized as:

$$f_{\boldsymbol{w}} = T_{\text{MixMatch}}\left(S_l, S_u, \alpha, \gamma, \lambda\right) = \underset{\boldsymbol{w}}{\text{argmin}}\mathcal{L}\left(S, \boldsymbol{w}\right) \tag{2}$$

$$\mathcal{L}\left(S, \boldsymbol{w}\right) = \sum_{(\boldsymbol{x}_i, \boldsymbol{y}_i) \in S_l'} \mathcal{L}_l\left(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{y}_i\right) + \text{r(t)}\ \gamma \sum_{(\boldsymbol{x}_j, \widetilde{\boldsymbol{y}}_j) \in \widetilde{S}_u'} \mathcal{L}_u\left(\boldsymbol{w}, \boldsymbol{x}_j, \widetilde{\boldsymbol{y}}_j\right) \tag{3}$$

For supervised and semi-supervised loss functions, the cross-entropy $\mathcal{L}_l\left(\boldsymbol{w}, \boldsymbol{x}_i, \boldsymbol{y}_i\right) = \delta_{\text{cross-entropy}}\left(\boldsymbol{y}_i, f_{\boldsymbol{w}}\left(\boldsymbol{x}_i\right)\right)$ and the Euclidean distance $\mathcal{L}_u\left(\boldsymbol{w}, \boldsymbol{x}_j, \widetilde{\boldsymbol{y}}_j\right) = \left\|\widetilde{\boldsymbol{y}}_j - f_{\boldsymbol{w}}\left(\boldsymbol{x}_j\right)\right\|$, are usually implemented, respectively. The regularization γ controls the direct influence on unlabelled data. Since in the first epochs, unlabelled data based predictions are unreliable, the function $r(t) = t/\rho$ increases the unsupervised term contribution as the number of epochs progress. The coefficient ρ is referred to as the rampup coefficient. Unlabelled data also influences the labelled data term $\mathcal{L}_l$, since unlabelled data is also used to artificially augment the dataset using the Mix Up algorithm. This loss term is used at training time, for testing, a regular cross entropy loss is implemented.

APPENDIX B
DEDIMS PSEUDOCODE DESCRIPTION

Algorithm 1 DeDiMs for unlabelled dataset selection

Input: A list of unlabelled datasets $S_{u_1}, S_{u_2}, ..., S_{u_k}$

For each unlabelled dataset S_{u_i} do:

1) Randomly sub-sample each one of the datasets S_l and S_{u_i}, with a sample size of τ, creating the sampled datasets $S_{l,\tau}$ and $S_{u_i,\tau}$.
2) Transform all input observations in the two samples $\boldsymbol{x}_j \in S_i$, with $\boldsymbol{x}_j \in \mathbb{R}^n$, with n the dimensionality of the input space, using the feature extractor f, yielding $\boldsymbol{h}_j = f(\boldsymbol{x}_j)$. Where $\boldsymbol{h}_i \in \mathbb{R}^{n'}$ is the feature vector of n' dimensions, with $n' < n$. For instance, the implemented feature extractor uses the Wide-ResNet architecture, extracting $n' = 512$ features. This yields the two feature sets $H_{l,\tau}$ and $H_{u_i,\tau}$
3) For each dimension $r = 1, ..., n'$ in the feature space, compute the normalized histogram $p_{r,l}$, in the sample $H_{l,\tau}$. Similarly, we compute the set of density functions $p_{u_i,b}$ for $r = 1, ..., n'$, using the observations in the sample $H_{u_i,\tau}$.
4) Compute the sum of the distances between the density functions $p_{r,l}$ and p_{r,u_i}, to yield the distance approximation for the sample j: $\widehat{d}_j = \sum_{r=1}^{n'} \delta_g(p_{r,a}, p_{r,b})$. We do this for all the $\mathcal{C}$ samples, yielding the list of inter-dataset distances: $\widehat{d}_1, \widehat{d}_2, ..., \widehat{d}_{\mathcal{C}}$. To lower the computational burden, we assume that the dimensions are statistically independent.
5) Compute the intra-dataset distances for the dataset S_l, in this context the labelled dataset S_l, to obtain the list of reference distances $\check{d}_1, \check{d}_2, ..., \check{d}_{\mathcal{C}}$.
6) Compute the p significance value with a Wilcoxon test to verify that the inter- and intra-dataset distance difference $d_c = \left|\widehat{d}_c - \check{d}_c\right|$ are statistically significant. The distance computation yields the sample mean distance $\overline{d}_{u,i}$ and its confidence value $p_{u,i}$.

Pick the unlabelled dataset $S_{u_{\text{best}}}$ with the lowest distance $\overline{d}_{u_{\text{lowest}}}$.

Result: $S_{u_{\text{best}}}$ the unlabelled dataset to yield the best accuracy for MixMatch

Note that based on our empirical results we recommend the use of density based deep dissimilarity measures, in particular cosine distance, as these displayed the best correlation with MixMatch accuracy.

Figure 1 shows a graphical summary of the proposed method. To further visualize the feature densities comparison, Figure 3 shows an extended comparison of the feature histograms of the two pairs of datasets compared in the manuscript. Moreover, Figure 2 shows the principal components of the feature space of two different datasets. Both figures illustrates how two different datsets are compared.

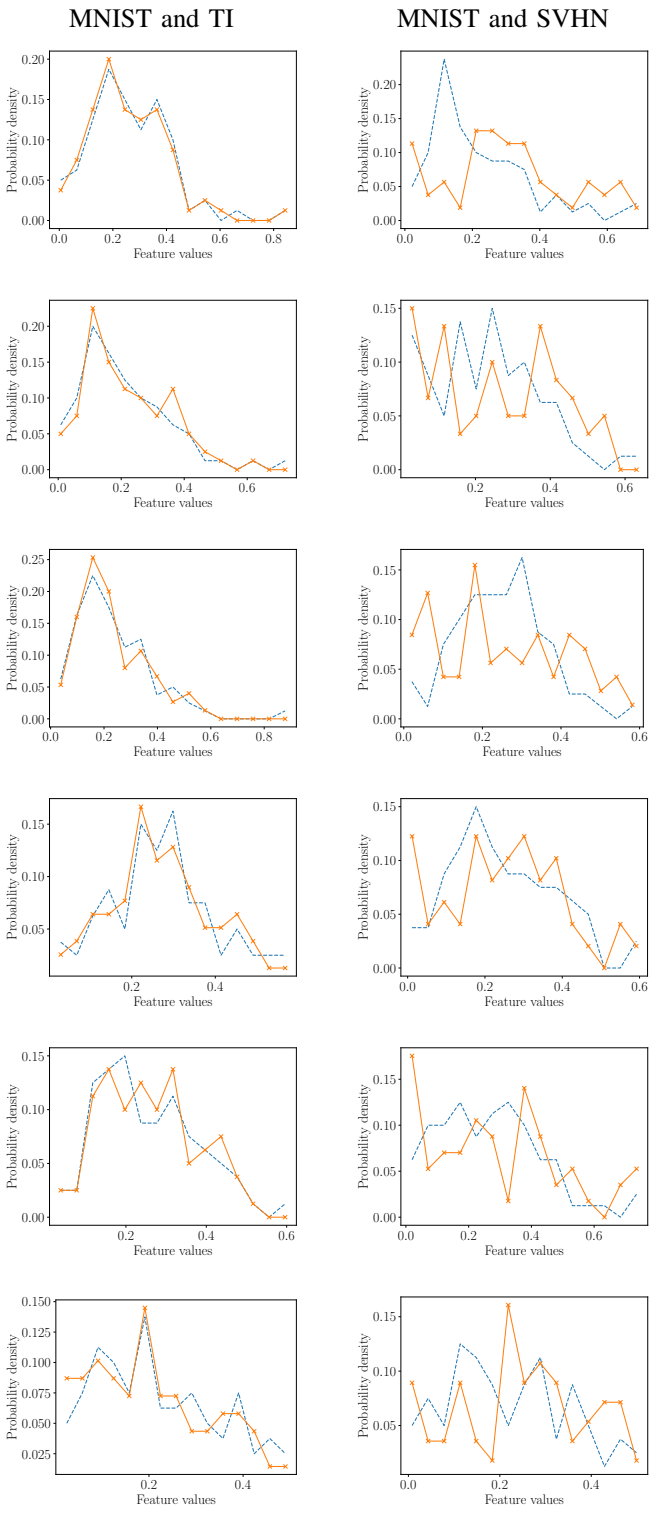

Fig. 3. Feature distribution for a model trained with MNIST labelled data (continous orange line in both plots), and TI and SVHN unlabelled data (left and right column, respectively, blue dashed line in both). For each plot a different dataset partition was used. From top to bottom, for each row: feature 372, 159, 7, 420, 82 and 491.

TABLE I
GLOBAL HYPERPARAMETERS WHICH ARE KEPT CONSTANT THROUGHOUT ALL EXPERIMENTS. THIS WAS DONE IN ORDER TO ISOLATE THE EFFECTS OF THE CHANGING OOD DATA CONFIGURATIONS.

Description	Value
Model architecture used in all tasks	wide_resnet
Number of training epochs	50
Batch size	16
Learning rate	0.0002
Weight decay	0.0001
Rampup coefficient	3000
Optimizer	Adam with 1-cycle policy [3]
K, Number of augmentations	2
T, Sharpening temperature	0.5
α, Parameter for the Beta distribution	0.75
γ, Gamma for the unsupervised loss weight	25

TABLE II
INFORMATION ON THE DATASETS USED IN THE EXPERIMENTS. **FORMAT** SPECIFIES THE FORMAT THE IMAGE FILES WERE PROVIDED IN, $\mathbf{d}$ SPECIFIES THE SIZE OF THE IMAGES, $\mathbf{N}$ SPECIFIES THE NUMBER OF SAMPLES IN THE DATASET, $|\mathcal{Y}|$ SPECIFIES THE NUMBER OF CLASSES IN THE DATASET, **RELATIVE CLASS DISTRIBUTION** SPECIFIES THE RELATIVE CLASS DISTRIBUTION IN THE DATASET.

Dataset	Format	d	N	$\lvert\mathcal{Y}\rvert$	Relative class distribution
MNIST[4]	.jpg	$28 \times 28 \times 1$	42,000	10	Uniform
SVHN[5]	.png	$32 \times 32 \times 3$	73,557	10	0.07/0.19/0.14/0.12/0.1/ 0.09/0.08/0.07/0.07/0.07
Tiny ImageNet[6]	.jpg	$64 \times 64 \times 3$	100,000	200	Uniform
CIFAR-10[7]	.jpg	$32 \times 32 \times 3$	50,000	10	Uniform
FashionMNIST[8]	.png	$28 \times 28 \times 1$	60,000	10	Uniform
Fashion Product[9]	.jpg	$60 \times 80 \times 3$	44,441	5	0.48/0.25/0.21/0.05/0.01
Gaussian	.png	$224 \times 224 \times 1$	20,000	NA	NA
Salt and Pepper	.png	$224 \times 224 \times 1$	20,000	NA	NA

APPENDIX C
EXPERIMENT HYPERPARAMETERS

Table I shows the hyperparameters used across all the experiments in this work.

APPENDIX D
DATASET DESCRIPTIONS

An overview of the different datasets used in this work can be found in Table II. Note that we used the training split of each dataset as the basis to construct our own training and test splits for each experimental run.

The Gaussian and Salt and Pepper datasets were created with the following parameters: a variance of 10 and mean 0 for the Gaussian noise, and an equal Bernoulli probability for 0 and 255 pixels, in the case of the Salt and Pepper noise.

A. Preprocessing

Each data point was preprocessed in the following way. After a subset of labelled and unlabelled data for an experimental run had been constructed the means and standard deviations (one pair for labelled data, one pair for unlabelled data) were calculated for this specific subset. Then, the labelled and unlabelled inputs were standardized by subtracting the respective mean and dividing it by the respective standard deviation.

In addition, in situations when the size of the unlabelled images differed from the size of the labelled images up- or downsampling was used to align the unlabelled image size.

TABLE III
OOD TEST BENCHMARKS FOR DIFFERENT TECHNIQUES. DATASETS WITH * WERE RANDOMLY CUT IN HALF FOR IN-DISTRIBUTION TRAINING LABELLED DATA AND THE OTHER HALF WAS USED AS OOD UNLABELLED DATA. THE TABLE REVEALS HOW ARBITRARY DIFFERENT TESTBEDS HAVE BEEN USED BY THE BENCHMARKING OOD DETECTION ALGORITHMS. IOD-OOD DATASET PAIRS ARE INDICATED BY NUMBER PAIRS IN THE TABLE.

Method name	IOD data	OOD data
Max. value of Softmax layer [10]	CIFAR-10 [1] CIFAR-100 [2] MNIST [3]	SUN[1,2] Gaussian [1,2] Omniglot [3] notMNIST[3] Uniform noise[3]
Inhibited Softmax[11]	CIFAR-10[1] MNIST[2]	SVHN[1] LFW-A[1] notMNIST[2] Omniglot[2]
ODIN [12]	CIFAR-10[1] CIFAR-100[2]	TinyImageNet[1,2] LSUN[1,2] iSUN[1,2] Uniform[1,2] Gaussian[1,2]
Epistemic Uncertainty Estimation [13]	CIFAR *[1] FashionMNIST*[2] SVHN*[3] MNIST*[4]	CIFAR*[1] FashionMNIST*[2] SVHN*[3] MNIST*[4]
Mahalanobis latent distance [14]	CIFAR-10[1] CIFAR-100[2] SVHN[3]	SVHN[1,2] CIFAR-10[3] TinyImageNet[1,2,3] LSUN[1,2,3]

APPENDIX E
CODE ARCHIVE AND COMPUTING INFRASTRUCTURE

All training and evaluation code can be found in https://github.com/luisoala/non-iid-ssdl. Software dependencies are specified in the requirements.txt file in the same archive. We use the mlflow framework for experiment management and reproduction. After experiments have been completed you can extract results from all runs using the analysis scripts provided in the archive.

Experiments were run on three machines. Machine 1 has one 12GB NVIDIA TITAN X GPU, 24 Intel(R) Xeon(R) CPU E5-2687W v4 @ 3.00GHz and 32GB RAM. Machine 2 has four 16GB NVIDIA T4 GPUs, 44 CPUs from the Intel Xeon Skylake family and 150GB RAM. Machine 3 has one 12GB NVIDIA TITAN V GPU, 24 Intel(R) Xeon(R) E5-2620 0 @ 2.00GHz CPU and 32GB RAM.

Experimental runs were parallelized using the ampersand option in bash executing 10 runs in parallel on a single GPU. With the current code base this requires up to 10 CPUs per GPU as well as approximately 25GB RAM per GPU. With this setup a single training epoch of 10 parallel experimental runs should last between 2 and 4 minutes per GPU, depending on which type of GPU is used.

APPENDIX F
EXISTING OOD DETECTION METHODS AND IOD-OOD DATA PAIRINGS

To expand the state of the art study, we include Table III with a comparison of the dataset settings used in previous work.

REFERENCES

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *Advances in Neural Information Processing Systems*, pages 5050–5060, 2019.

[2] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[3] Leslie N Smith. A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay. *arXiv preprint arXiv:1803.09820*, 2018.

[4] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Ha. Gradient-based learning applied to document recognition. page 46.

[5] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[7] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[8] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

[9] Param Aggarwal. Fashion product images (small). https://www.kaggle.com/paramaggarwal/fashion-product-images-small.

[10] Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016.

[11] Marcin Możejko, Mateusz Susik, and Rafał Karczewski. Inhibited softmax for uncertainty estimation in neural networks. *arXiv preprint arXiv:1810.01861*, 2018.

[12] Shiyu Liang, Yixuan Li, and Rayadurgam Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *6th International Conference on Learning Representations, ICLR 2018*, 2018.

[13] Natasa Tagasovska and David Lopez-Paz. Single-model uncertainties for deep learning. In *Advances in Neural Information Processing Systems*, pages 6414–6425, 2019.

[14] Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *Advances in Neural Information Processing Systems*, pages 7167–7177, 2018.