Appendix: Dissecting Sample Hardness: A Fine-Grained Analysis of Hardness Characterization Methods for Data-Centric AI

Table of Contents

A Additional background and details

This appendix provides additional details and background on the different HCMs.

A.1 Description of HCMs

We now provide a detailed description of the HCMs supported by H-CAT grouped by their class. We briefly describe how they characterize data and the modalities to which the HCMs have been previously applied.

We group the HCMs into broad classes determined based on the core metric or approach each method uses to characterize example hardness, namely

- Learning dynamics-based: These HCMs rely on metrics computed during the training process itself to characterize example hardness. As we describe in the paper, we further divide these into sub-categories based on the specific training dynamic metric used: (i) Margin, (ii) Uncertainty, (iii) Loss, (iv) Gradient, (v) Forgetting.
- Distance-based: These HCMs characterize hardness based on the distance or similarity of examples in an embedding space.
- Statistical-based: These methods use statistical metrics computed over the data to characterize example hardness.

Recall HCMs generally follow the following input-output paradigm:

■ **Inputs:** (i) Dataset $\mathcal{D} = \{(x_i, y_i)\}$ drawn from both $\mathcal{P}_{XY}$. (ii) Learning algorithm f_θ.
■ **Outputs:** Assign a score s_i to sample (x_i, y_i). Apply threshold τ to assign a hardness label $g \in \mathcal{G}$, where $\mathcal{G} = \{Easy, Hard\}$, which is then used to partition $\mathcal{D} = \mathcal{D}_e \cup \mathcal{D}_h$.

Consequently, we define the scoring function S as the general notation applicable to each HCM. Note that whether a high score or low score indicates a "hard" sample is method-dependent. We highlight HCMs included in H-CAT with a * and formalize those specifically with notation.

We provide a Table below to clarify whether high or low scores mean "hard" for specific HCMs.

Table 2: Overview of HCMs and the meaning of their scoring functions.

Method	Description	Score
Area Under the Margin (AUM)	Characterizes data examples based on the margin of a classifier – i.e., the difference between the logit values of the correct class and the next class.	Hard - low scores.
Confident Learning	Confident Learning estimates the joint distribution of noisy and true labels — characterizing data as easy and hard for mislabeling.	Hard - low scores
Conf Agree	Agreement measures the agreement of predictions on the same example.	Hard - low scores
Data IQ	Data-IQ computes the aleatoric uncertainty and confidence to characterize the data into easy, ambiguous, and hard examples.	Hard - low confidence scores. Low Aleatoric Uncertainty scores
Data Maps	Data Maps focuses on measuring variability (epistemic uncertainty) and confidence to characterize the data into easy, ambiguous, and hard examples.	Hard - low confidence scores. Low Epistemic Uncertainty scores
Gradient Norm (GraNd)	GraNd measures the gradient norm to characterize data.	Hard - high scores
Error L2-Norm (EL2N)	EL2N calculates the L2 norm of error over training in order to characterize data for computational purposes.	Hard - high scores
Forgetting	Forgetting scores analyze example transitions through training. i.e., the time a sample correctly learned at one epoch is then forgotten.	Hard - high scores
Large Loss	Large Loss characterizes data based on sample-level loss magnitudes.	Hard - high scores
Prototypicalilty	Prototypicality calculates the latent space clustering distance of the sample to the class centroid as the metric to characterize data.	Hard - high scores
Variance of Gradients (VOG)	VoG (Variance of gradients) estimates the variance of gradients for each sample over training	Hard - high scores
ALLSH	ALLSH computes the KL divergence of softmax outputs between original and augmented samples to characterize data.	Hard - high scores

We now outline different HCMs. Specifically, we formalize the HCMs included in H-CAT in a unified notation of $S(x, y)$.

A.1.1 LEARNING-BASED (MARGIN)

AUM (Pleiss et al., 2020)* AUM (Area under the margin) characterizes data examples based on the margin of a classifier – i.e. the difference between the logit values of the correct class and the next class. It has been applied to mislabeling for computer vision (CV).

$$M^{(t)}(x, y) = \overbrace{z_y^{(t)}(x)}^{\text{assigned logit}} - \overbrace{\max_{i \neq y} z_i^{(t)}(x)}^{\text{largest other logit}}. \tag{1}$$

Over all epochs the AUM is then computed as the average of these margin calculations. We define it in terms of S.

$$S(x, y) = AUM(x, y) = \tfrac{1}{T} \textstyle\sum_{t=1}^{T} M^{(t)}(x, y), \tag{2}$$

$$S(x, y) = \tfrac{1}{T} \textstyle\sum_{t=1}^{T} (\overbrace{z_y^{(t)}(x)}^{\text{assigned logit}} - \overbrace{\max_{i \neq y} z_i^{(t)}(x))}^{\text{largest other logit}} \tag{3}$$

A.1.2 LEARNING-BASED (UNCERTAINTY)

Data-IQ (Seedat et al., 2022a)* Data-IQ computes the aleatoric uncertainty and confidence to characterize the data into easy, ambiguous, and hard examples and has been studied for the tabular domain.

The aleatoric uncertainty is computed as $v_{\text{al}}(x) = \frac{1}{T} \sum_{t=1}^{T} \mathcal{P}(x, \theta_t)(1 - \mathcal{P}(x, \theta_t))$ and confidence over epochs as $\mathcal{P}(x, \theta_t)$.

For hard samples, the score for an input (x,y) is computed primarily on the confidence with uncertainty used to delineate ambiguous:

$$S(x, y) = \frac{1}{T} \sum_{t=1}^{T} \mathcal{P}(x, \theta_t) \tag{4}$$

$$C(x, y) = \frac{1}{T} \sum_{t=1}^{T} \mathcal{P}(x, \theta_t) \tag{5}$$

Data Maps (Swayamdipta et al., 2020)* Data Maps focuses on measuring variability (epistemic uncertainty) and confidence to characterize the data into easy, ambiguous, and hard examples and has been studied for natural language processing (NLP) tasks.

The epistemic uncertainty is computed as $v_{\text{ep}}(x) = \frac{1}{T} \sum_{t=1}^{T} \left[\mathcal{P}(x, \theta_t) - \bar{\mathcal{P}}(x)\right]^2$ and confidence over epochs as $\mathcal{P}(x, \theta_t)$.

For hard samples, the score for an input (x,y) is computed primarily on the confidence with uncertainty used to delineate ambiguous:

$$S(x, y) = \mathcal{P}(x, \theta_e) \tag{6}$$

A.1.3 LEARNING-BASED (LOSS)

Small Loss (Xia et al., 2021)* Small Loss characterized data based on sample-level loss magnitudes and has been studied for computer vision (CV) tasks.

The small loss for an input x, y is computed as:

$$S(x, y) = \frac{1}{t} \sum_{i=1}^{t} l_i, \tag{7}$$

where $\quad l_i = \ell(f(w; x), y)$.

RHO-Loss (Mindermann et al., 2022) RHO-Loss estimates the sample loss on a holdout set and has been applied in CV tasks. We do not include it in H-CAT as it requires training an irreducible loss model.

A.1.4 LEARNING-BASED (GRADIENT)

GraNd (Paul et al., 2021)* GraNd measures the gradient norm to characterize data, particularly for computational reasons in computer vision (CV) tasks.

The gradient norm at epoch t for an input x, y is computed as :

$$S(x,y) = \mathbb{E}\left\|\sum_{k=1}^{K} \boldsymbol{\nabla}_{f^{(k)}}\ell\left(f_t(x),y\right)^T \psi_t^{(k)}(x)\right\|_2 \quad where \psi_t^{(k)}(x) = \boldsymbol{\nabla}_{\mathbf{w}_t} f_t^{(k)}(x). \tag{8}$$

VoG (Agarwal et al., 2022)* VoG (Variance of gradients) estimates the variance of gradients for each sample over training, particularly in computer vision (CV) tasks.

The VoG at epoch t for an input x, y is computed as :

$$S(x,y) = \frac{1}{N}\sum_{t=1}^{N}(VoG_p(x,y)), \tag{9}$$

$$VoG_p(x,y) = \sqrt{\frac{1}{K}\sum_{t=1}^{K}(M_t(x,y)-\mu)^2}. \tag{3}$$

where $\mu = \frac{1}{K}\sum_{t=1}^{K} M_t$

Note M_t is the gradient matrix.

A.1.5 LEARNING-BASED (FORGETTING)

Forgetting scores (Toneva et al., 2019)* Forgetting scores analyze example transitions through training. i.e., the time a sample correctly learned at one epoch is then forgotten. It is commonly used in CV tasks.

The Forgetting score an input x, y is computed as :

$$S(x,y) \leftarrow S(x,y) + \mathbf{1}(\text{prev\_acc}_i > \text{acc}(x,y)) \tag{10}$$

SSFT (Maini et al., 2022) SSFT (Second Split Forgetting Time) measures the time of forgetting when fine-tuning models on a second split. It has been applied in CV tasks. We do not include it in H-CAT as it requires fine-tuning on a validation dataset,

A.1.6 LEARNING-BASED (STATISTICS)

EL2N (Paul et al., 2021)* EL2N (L2 norm of error over training) calculates the L2 norm of error over training in order to characterize data for computational purposes. It is predominantly applied in CV tasks.

The EL2N at epoch t for an input x, y is computed as :

$$S(x,y) = \mathbb{E}\|p(w_t,x) - y\|_2. \tag{11}$$

Noise detector (Jia et al., 2022)* Noise detector utilizes a trained detector model to predict noisy samples. The detector is applied to training dynamics and has been used in CV tasks.

Noise detector uses a trained detector g to score x,y as:

$$S(x_i,y_i) = g(L_{y_i}(i)) \tag{12}$$

where $L(i,t) = f^{(t)}(x_i)$[training dynamic]

A.1.7 DISTANCE-BASED

Prototypicality (Sorscher et al., 2022)* Prototypicality calculates the latent space clustering distance to the centroid as the metric to characterize data. It has been applied to CV tasks.

Prototypicality score for x,y is computed as:

$$S(x, y) = \|\phi(x) - \mu_y\|_2 \tag{13}$$

Here, $\phi(x)$ represents the mapping of x to the latent space, and μ_y is the centroid of the class y in the latent space. The norm $\| \cdot \|_2$ denotes the Euclidean distance.

A.1.8 STATISTICAL-BASED

Cleanlab (Northcutt et al., 2021a)* Cleanlab estimates the joint distribution of noisy and true labels — characterizing data as easy and hard for mislabeling. It can be applied to any modality.

The Cleanlab score for an input x, y is computed as :

$$S(x, y) = \{x \in X \mid y = y^* \neq j\} \text{ for } x \text{ from the off-diagonals of } C_{y_j,y^*}. \tag{14}$$

ALLSH (Zhang et al., 2022)* ALLSH computes the KL divergence of softmax outputs between original and augmented samples to characterize data.

The Allsh score for an input x, y is computed as :

$$S(x, y) = D(p_\theta(\cdot|x), p_\theta(\cdot|x')) \tag{15}$$

where x' is a augmented x and D a distance measure (e.g KL-Divergence).

Agreement (Carlini et al., 2019)* Agreement measures the agreement of predictions on the same example. It is commonly used in CV tasks.

The Agreement for an input x, y is computed as :

$$S(x, y) = \frac{1}{N} \sum_{i=1}^{N} \max f_i(x) \tag{16}$$

Data Shapley (Ghorbani & Zou, 2019) Data Shapley computes the Shapley value to characterize the value of including a data point. It is commonly used in CV tasks. We also do not consider Data Shapley in H-CAT and variants (e.g. Beta Shapley (Kwon & Zou, 2022)), which have been shown to be computationally unfeasible with numerical instabilities for higher dimensional data such as MNIST and CIFAR-10 with > 1000 samples (Wang & Jia, 2023).

A.2 PRACTICAL APPLICATIONS OF HCMS.

To illustrate the practical value and need for HCMs in creating high-quality datasets, we highlight the following application domains:

- The Cleanlab HCM was used to audit ML benchmark datasets for mislabeling (Northcutt et al., 2021b).
- The Data Maps HCM was used to audit and create high-quality RNA data in biology (Nabi et al., 2023).
- The Data-IQ HCM was used to audit and curate high-quality sleep staging data (Heremans et al., 2024).
- The Data-IQ and Cleanlab HCMs have been used to guide synthetic data generation (Hansen et al., 2023).
- The Data Maps and Data-IQ HCMs where re-purposed to assess samples for data-centric off-policy evaluation (Sun et al., 2023a).

A.3 Hardness taxonomy for tabular data.

In the main paper, we primarily discussed our manifestation of the hardness taxonomy with respect to images. We describe below how this isn't confined to images and how these hardness types could manifest, for instance, in tabular data.

Mislabeling: Mislabeling in tabular data can occur when the label attributed to a particular row (representing a sample) is incorrect. For instance, in a medical dataset, this could occur when a patient with a certain condition is wrongly labeled as not having the condition (or vice versa). Note mislabeling is no different from the image domain as the perturbation is applied on y.

Hence, the various sub-types of mislabeling can still apply. Uniform mislabeling means any class of disease can be mislabeled as any other with equal probability. Asymmetric mislabeling would mean certain classes of disease are more likely to be confused with others, for instance, certain diseases might be more commonly misdiagnosed as each other. Consequently, in implementation uniform and asymmetric do not change from images. Adjacent mislabeling (special case of asymmetric) could refer to situations where labels are ordinal and a higher value might be mislabeled as a lower one (or vice versa). Instance-specific mislabeling refers to instances where certain individual observations are more likely to be mislabeled based on their specific attributes. We could still use "human knowledge" of similar classes for tabular data. We demonstrate the use of PCA to embed the data and then find the nearest class for a specific label in embedding space as the instance-wise mislabeling.

Near-OoD: Near-OoD in tabular data could refer to rows of data where the feature values are still within the same domain but differ in some way from typical observations. For example, in a financial dataset, a Near OoD sample might be an account with an unusually high number of transactions, which, although not completely unusual, stands apart from typical transaction behavior. This could represent a Covariate Shift where the distribution of the predictors changes, or Domain Shift where the rules of data generation change slightly.

Alternatively, in a dataset of regular health screenings, a Near OoD instance might be a health record of a patient who shows high values in their lab results, which although still plausible, stand apart from typical health metrics. For instance, a patient with very high cholesterol levels would be a Near OoD example in this context, indicating a Covariate Shift where the distribution of predictors changes. In terms of implementation, the continuous variables of our tabular data might be noisy or covariate shifted data. For example, Gaussian noise to continuous features.

Far-OoD: Far Out-of-Distribution in tabular data refers to rows that are dramatically different from typical observations in a dataset, or might even represent entirely new categories of data. Using the financial dataset example, a Far OoD sample might be transactions in a new foreign currency not previously seen in the dataset. This type of data is distinctly different and unrelated to the rest of the dataset.

In a medical dataset, Far-OoD might refer to patient records that are substantially different from typical observations or that fall into entirely new categories. For instance, imagine a dataset mainly consisting of adult patients, and suddenly you get data from pediatric patients. Children have different medical characteristics and range for many medical metrics (like heart rate, blood pressure, etc.), so pediatric patient data would be Far OoD in this context. Another example could be the inclusion of a new disease class not previously part of the dataset.

Practically, for a given tabular dataset, this could be data from a different distribution (e.g. shuffled data) or feature relationships that don't match — e.g. older people having a specific feature linked to a disease, but these relationships are now broken. For example, within the framework this could be implemented as follows: for binary covariates, we can swap the features (i.e. flip male to female etc.) — to break the feature relationships such that they are drawn from a completely different distribution. For categorical variables, we can flip the category of a subset of variables in order to break the feature relationships.

Atypical: Atypical data in a tabular dataset can refer to observations that, while not incorrect or out of distribution, deviate from common patterns in the data, i.e. they are rare but valid rows of the table (e.g. outliers, extremes — samples from tails of marginal). For example, in a dataset of real estate properties, an atypical sample could be a property with an unusually large number of rooms or an unusually low price for its size. While such a property could exist (so it's not necessarily mislabeled), it deviates from the common pattern in the data and might be found in the long tail of the distribution.

In a medical dataset, it might refer to patient records that, while not incorrect or out of distribution, deviate from common patterns in the data. For instance, consider a dataset of prostate cancer patients, where naturally, most patients are older. An atypical example might be a young adult with prostate cancer, which would represent an unusual instance, as they fall outside the common pattern of older patients with prostate cancer. From a framework implementation perspective, we could sample and replace with values from the tails of the marginals for the most predictive long-tailed feature (via feature-label correlation).

B EXPERIMENTAL DETAILS

This appendix describes our experimental details, computational resources, HCMs, and datasets below.

B.1 COMPUTATIONAL RESOURCES

Experiments are parallelized to run different setups using three NVIDIA RTX A4000s, with an 18-Core Intel Core i9 and 16 gb of RAM. Each experimental setup can be run on a single GPU.

B.2 DATASETS

We run our experiments on MNIST and CIFAR-10 datasets. However, these experiments can be run on alternative datasets, as shown in Appendix C.

MNIST (LeCun et al., 2010) : consists of 70,000 small square 28x28 pixel grayscale images of digits from 0 to 9. Each image is a handwritten digit, with 60,000 of the images dedicated for training data and the remaining 10,000 for testing data.

CIFAR-10 (Krizhevsky et al.) : It consists of 60,000 32x32 color images distributed among 10 different classes such as airplanes, dogs, cats, and trucks. The dataset is divided into 50,000 training images and 10,000 testing images. Each class has an equal representation in the dataset with 6,000 images per class.

Note on selection of MNIST & CIFAR-10. We chose the MNIST and CIFAR-10 datasets as benchmarks because they are well-established datasets in the literature. Notably, many HCMs employ MNIST and CIFAR-10 within the context of their limited quantitative evaluation. Further, the noisy label literature frequently uses them for controlled experiments. This prevalence underscores their appropriateness for controlled benchmarking.

Let’s compare potential alternatives.

- Completely synthetic data: This would be 100% clean, but too simple and toy-like.
- Large and highly complicated datasets like ImageNet: These contain significant levels of mislabeling (over 5%) (Northcutt et al., 2021b), hence we cannot do controlled experiments.
- MNIST and CIFAR-10: These are real image datasets. But they have been shown to contain almost no/little mislabeling (under 0.5%) (Northcutt et al., 2021b),.

Tabular data : We conducted analogous experiments to the image experiments on tabular datasets, specifically, the OpenML (Vanschoren et al., 2014) datasets ’cover’ and ’diabetes130US’. Note these datasets are included in a recent NeurIPS benchmark for tabular models (Grinsztajn et al., 2022). Additionally, this allows us to flexibly use the H-CAT framework with *any* tabular dataset as long as it conforms to the OpenML structure.

B.3 BACKBONE MODELS

We use the following backbone models: LeNet (7-layer convolutional neural network) and ResNet-18 (18-layer deep convolutional neural network with "skip" or "residual" connections). We train the backbones with a cross-entropy loss using the Adam optimizer. Our learning rate of 1e-3 and batch size of 32 are kept fixed as they showed good convergence. As discussed, we explore different training epochs of 10 and 20 and show consistency irrespective, in Appendix D. This is reasonable as has been shown by (Seedat et al., 2022a) that the learning captured by HCMs (especially in training dynamics) happens in the early epochs. Note, we fix the backbone models in line with principles of Data-Centric AI of systematically assessing the data for a fixed ML model (Liang et al., 2022; Seedat et al., 2023b; Whang et al., 2023; Ng et al., 2021; Jarrahi et al., 2022).

B.4 HCMs

We outline the implementation of the 13 HCMs supported by H-CAT below. We have described the HCMs in Appendix A.

1. Learning-based (Margin)

AUM (Pleiss et al., 2020) : Our implementation is based on [3].

2. Learning-based (Uncertainty)

Data-IQ (Seedat et al., 2022a) : Our implementation is based on [4].

Data Maps (Swayamdipta et al., 2020) : Our implementation is based on [5].

3. Learning-based (Loss)

Large Loss (Xia et al., 2021) : Our implementation computes the loss per sample individually.

4. Learning-based (Gradient)

GraNd (Paul et al., 2021) : Our implementation is based on the Pytorch implementation from [6] which is based on the original JAX [7].

VoG (Agarwal et al., 2022) : Our implementation is based on [8].

5.Learning-based (Forgetting)

Forgetting scores (Toneva et al., 2019) : Our implementation based on the original paper tracks the learning and forgetting times per sample in order to compute the forgetting score.

6. Learning-based (Statistics)

EL2N (Paul et al., 2021) : Our implementation is based on the Pytorch implementation from [9] which is based on the original JAX [10].

Noise detector (Jia et al., 2022) : Our implementation is based on [11]. We use the noise detector training scripts in the repo.

7. Distance-based

Prototypicality (Sorscher et al., 2022) : Our implementation follows the original paper and computes prototypicality as follows. We embed the image as a low-dimensional embedding using the backbone network. We then compute the Cosine distances to the mean embedding of the specific label for that sample. The cosine distances being smaller are more prototypical.

8. Statistical-based

[3]https://github.com/asappresearch/aum
[4]https://github.com/seedatnabeel/Data-IQ
[5]https://github.com/seedatnabeel/Data-IQ
[6]https://github.com/BlackHC/pytorch_datadiet
[7]https://github.com/mansheej/data_diet
[8]https://github.com/chirag126/VOG
[9]https://github.com/BlackHC/pytorch_datadiet
[10]https://github.com/mansheej/data_diet
[11]https://github.com/Christophe-Jia/mislabel-detection

Cleanlab (Northcutt et al., 2021a) : Our implementation is based on [12]

ALLSH (Zhang et al., 2022) : Our implementation follows the original paper. We compute the KL divergence in the model predictive probabilities for the pairs containing: (1) original input and (2) augmentation of the input. We perform the augmentation using AugLY [13]. The augmentations are selected from Horizontal flip, Random brightness, Random Noise and Random Pixelization.

Agreement (Carlini et al., 2019) : Our implementation of confidence agreement produces multiple predictions using MC-Dropout to create 10 pseudo-ensembles. We then compute the mean of the confidences as a measure of agreement, as per the original paper.

B.5 Implementation details

We outline the experimental implementation details next. Details on the H-CAT and its usage can be found in Appendix C.

Hardness detection & Ranking. In this experiment, we set a different random seed for each of the three runs. For each run, we resample the dataset and change the samples that we perturb (i.e. which samples are hard). We also repeat the experiments for multiple proportions p.

We execute all experiments with ‘p’ values of [0.1, 0.2, 0.3, 0.4, 0.5], except for Far-OoD and Atypical experiments, where ‘p’ values are restricted to [0.05, 0.1, 0.15, 0.2, 0.25]. The rationale behind this distinction is that, in reality, Far-OoD and Atypical conditions do not typically occur in high proportions. For instance, a sample can’t be considered Atypical if it constitutes 50% of occurrences. Therefore, for these conditions, we limit ‘p’ to a maximum of 0.25.

Stability/Consistency In this experiment, we maintain a consistent random data split and the set of perturbed samples, which represent the difficult examples, is also kept constant across different runs. The element of randomness is brought in solely through the training of the backbone model and by extension, the characterization of the HCM. i.e. we have a different seed for the model. This approach is taken because our aim is to evaluate the Stability/Consistency of the HCM across multiple runs when presented with the same data.

[12]https://github.com/cleanlab/cleanlab

[13]https://github.com/facebookresearch/AugLy

C Using H-CAT

This appendix discusses how to use the H-CAT framework. We link to the codebase, provide a tutorial on using H-CAT (see Sec. C.2), and also show how H-CAT can be extended to new HCMs and datasets (see Sec. C.3).

C.1 H-CAT Benchmarking Framework Repository

To foster ease of use and engagement, we release the H-CAT Benchmarking Framework's codebase with Apache 2.0 License. It can be found at:
https://github.com/seedatnabeel/H-CAT

C.2 Tutorial on using H-CAT

Below, we illustrate a tutorial demonstrating how to use H-CAT, highlighting the composable nature and the object-oriented interface making it easy to use. We also provide this in the form of a Jupyter notebook called *tutorial.ipynb*.

```
from src.trainer import PyTorchTrainer
from src.dataloader import MultiFormatDataLoader
from src.models import *
from src.evaluator import Evaluator
```

Listing 1: Step 1: Necessary imports

```
hardness = "uniform"
p=0.1
dataset = "mnist"
model_name = "lenet"
epochs = 20
seed = 0

# Defined by prior or domain knowledge
if hardness =="instance":
    if dataset == "mnist":
        rule_matrix = {
                    1: [7],
                    2: [7],
                    3: [8],
                    4: [4],
                    5: [6],
                    6: [5],
                    7: [1, 2],
                    8: [3],
                    9: [7],
                    0: [0]
                }
    if dataset == "cifar":

        rule_matrix = {
                    0: [2],    # airplane (unchanged)
                    1: [9],    # automobile -> truck
                    2: [9],    # bird (unchanged)
                    3: [5],    # cat -> automobile
                    4: [5,7],    # deer (unchanged)
                    5: [3, 4],    # dog -> cat
                    6: [6],    # frog (unchanged)
                    7: [5],    # horse -> dog
                    8: [7],    # ship (unchanged)
                    9: [9],    # truck -> horse
                }

else:
    rule_matrix = None
```

Listing 2: Step 2: Define experimental parameters

```
characterization_methods =  [
            "aum",
            "data_uncert", # for both Data-IQ and Data-Maps
            "el2n",
            "grand",
            "cleanlab",
            "forgetting",
            "vog",
            "prototypicality",
            "allsh",
            "loss",
            "conf_agree",
            "detector"
        ],
```

Listing 3: Step 3: Define HCMs to evaluate — if excluded we evaluate all

```
import torch
import torch.nn as nn
import torch.optim as optim
from torchvision import datasets, transforms

if dataset == 'cifar':
    # Define transforms for the dataset
    transform = transforms.Compose([
        transforms.ToTensor(),
        transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
    ])

    # Load the CIFAR-10 dataset
    train_dataset = datasets.CIFAR10(root="./data", train=True, download=
    True, transform=transform)
    test_dataset = datasets.CIFAR10(root="./data", train=False, download=
    True, transform=transform)

elif dataset =='mnist':
    # Define transforms for the dataset
    transform = transforms.Compose(
        [transforms.ToTensor(),
        transforms.Normalize((0.5,), (0.5,))])


    # Load the MNIST dataset
    train_dataset = datasets.MNIST(root="./data", train=True, download=
    True, transform=transform)
    test_dataset = datasets.MNIST(root="./data", train=False, download=
    True, transform=transform)


total_samples = len(train_dataset)
num_classes = 10

# Set device to use
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

Listing 4: Step 4: Load datasets

```python
# Allows importing data in multiple formats

dataloader_class = MultiFormatDataLoader(data=train_dataset,
                                        target_column=None,
                                        data_type='torch_dataset',
                                        batch_size=64,
                                        shuffle=True,
                                        num_workers=0,
                                        transform=None,
                                        image_transform=None,
                                        perturbation_method=hardness,
                                        p=p,
                                        rule_matrix=rule_matrix
        )


dataloader, dataloader_unshuffled = dataloader_class.get_dataloader()
flag_ids = dataloader_class.get_flag_ids()
```

Listing 5: Step 5: Call H-CAT Dataloader Module (with Hardness module parameterized)

```python
# Instantiate the neural network and optimizer
if dataset == 'cifar':
    if model_name == 'LeNet':
        model = LeNet(num_classes=10).to(device)
    if model_name == 'ResNet':
        model = ResNet18().to(device)
elif dataset == 'mnist':
    if model_name == 'LeNet':
        model = LeNetMNIST(num_classes=10).to(device)
    if model_name == 'ResNet':
        model = ResNet18MNIST().to(device)

criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(model.parameters(), lr=0.001)


# Instantiate the PyTorchTrainer class
trainer = PyTorchTrainer(model=model,
                            criterion=criterion,
                            optimizer=optimizer,
                            lr=0.001,
                            epochs=epochs,
                            total_samples=total_samples,
                            num_classes=num_classes,
                            characterization_methods=
    characterization_methods,
                            device=device)

# Train the model
trainer.fit(dataloader, dataloader_unshuffled)

hardness_dict = trainer.get_hardness_methods()
```

Listing 6: Step 6: Call H-CAT Trainer Module (with HCM module parameterized)

```python
eval = Evaluator(hardness_dict=hardness_dict, flag_ids=flag_ids, p=p)

eval_dict, raw_scores_dict = eval.compute_results()

print(eval_dict)
```

Listing 7: Step 7: Call H-CAT Evaluator Module

C.3 Extending H-CAT

C.3.1 Adding a new HCM

New HCMs can easily be added to H-CAT. We describe the process below.

We take an object-oriented approach and hence new HCMs can be included by inheriting from our base class Hardness_Base defined below.

```python
# Base class for HCMs
class Hardness_Base:
    def __init__(self, name):
        self.name = name
        self._scores = None

    def updates(self):
        pass

    def compute_scores(self):
        pass

    @property
    def scores(self):
        if self._scores is None:
            self.compute_scores()
        return self._scores
```

Listing 8: HCM Base class

To create a new HCM, you would create a new class that inherits from Hardness_Base. In this new subclass, override the updates and compute_scores methods with the specific logic for this new hardness calculation method. We show an example below.

```python
class MyNewHCM(Hardness_Base):
    def __init__(self, name):
        super().__init__(name)

    def updates(self):
        # logic to perform updates goes here
        pass

    def compute_scores(self):
        # logic to compute scores goes here
        self._scores = "some computed value"  # Replace this with your
    computation
```

Listing 9: Example New HCM

The HCM then needs to be added to *trainer.py* in the correct part of the training loop and can be used as below

```python
hcm = MyNewHCM('some name')
hcm.updates()
print(hcm.scores)
```

Listing 10: Usage of the new defined HCM

Note, one needs to think for the HCM if it gets updated every epoch or at the end of training to define placement in the training loop.

C.3.2 Using a different dataset

We can use different datasets with H-CAT quite simply.

Before delving further, we show below where the new dataset needs to be passed to the *Dataloader module*. i.e. We need to pass *train_dataset*.

```python
# Allows importing data in multiple formats

dataloader_class = MultiFormatDataLoader(data=train_dataset,
                                        target_column=None,
                                        data_type='torch_dataset',
                                        batch_size=64,
                                        shuffle=True,
                                        num_workers=0,
                                        transform=None,
                                        image_transform=None,
                                        perturbation_method=hardness,
                                        p=p,
                                        rule_matrix=rule_matrix
        )
```

Listing 11: Example of passing data to the Dataloader module

The easiest way is to define your dataset as a standard torch dataset outside of H-CAT and simply to pass that into the *MultiFormatDataLoader* and specify a *target_column=None*.

If one has the data as *NumPy arrays* or *torch tensors*, one would need to pass a tuple as *data = (X,y)* and specify a *target_column*.

Other formats are also possible with H-CAT. For instance, if you have images in folders. One could pass data as a CSV file, JSON, or Python dictionary — wherein there is a column with paths to each sample and a column with the target. One can then pass the path to the CSV file or JSON file or pass the Python dict itself. In addition, one should specify the *target_column*. The rest is handled internally within H-CAT to convert the data to an appropriate Torch dataset for training.

That said, ideally if the dataset can be converted to a Torch dataset before passing it to H-CAT that is preferred.

D Additional Experiments

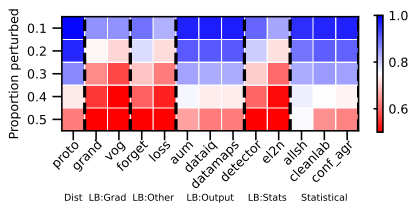
This appendix presents additional experiments and full results across all setups using H-CAT to evaluate different HCMs.

D.1 Multiple hardness types simultaneously

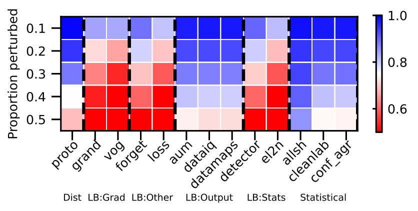
Overview. As discussed in the main paper, we primarily focussed on single-source hardness. We might also expect cases where multiple hardness types manifest simultaneously. For instance, if an image is Near-OoD (e.g. blurry or different texture), we might be more likely also to mislabel it. As a demonstration of the usage of H-CAT for this purpose, we hence assess the case where both Near-OoD (Covariate and Domain) are paired with Instance-specific mislabeling.

We want to understand the effect of this on HCM capabilities.

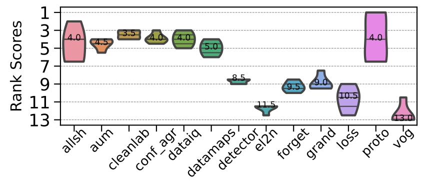
Results. We show results for both Near-OoD (Covariate) + Instance and Near-OoD (Domain) + Instance below in Figure 8.



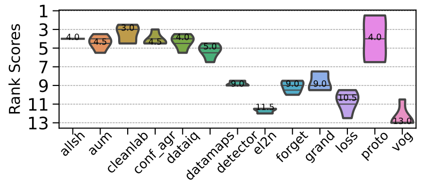
(a) Near-OoD (Covariate) + Instance



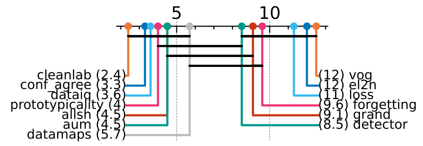
(b) Near-OoD (Domain) + Instance



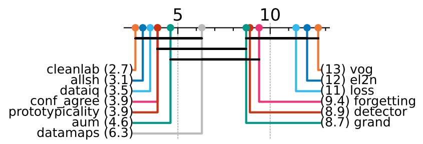
(c) Near-OoD (Covariate) + Instance



(d) Near-OoD (Domain) + Instance



(e) Near-OoD (Covariate) + Instance



(f) Near-OoD (Domain) + Instance

Figure 8: Results using H-CAT for multiple hardness types manifesting simultaneously.

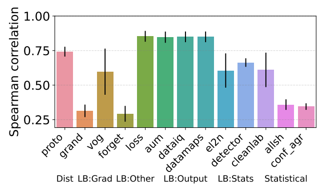
Takeaway. Our results indicate that H-CAT can be used to assess multiple hardness types, which manifest simultaneously. We find that HCMs which perform well individually on both types of hardness — still perform well in the simultaneous setting.

D.2 Stability/Consistency: Backbone model type

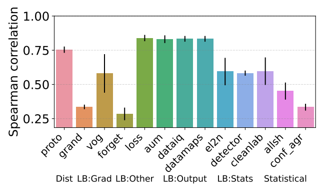
Overview. We assessed HCMs using two different backbone models - LeNet and ResNet-18. We wish to assess the stability/consistency of the backbone model. As discussed in the main paper, we can quantify this based on the Spearman rank correlation — i.e., are the HCMs ordering the samples consistently in a stable manner?

In the main paper, we assessed, within model stability/consistency, the randomness of a run. However, here we assess the rank correlation of models head-to-head for the same hardness type.

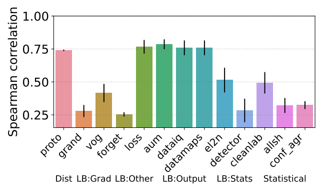
Results. Figure 9 shows the results across multiple hardness types.



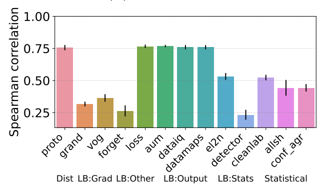
(a) Uniform



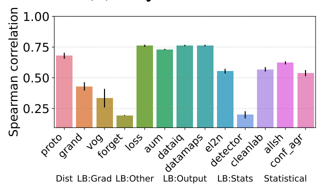
(b) Asymmetric



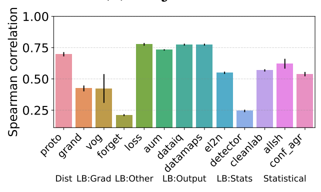
(c) Adjacent



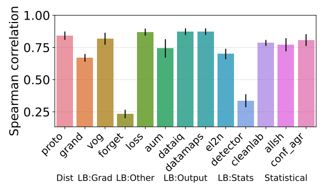
(d) Instance



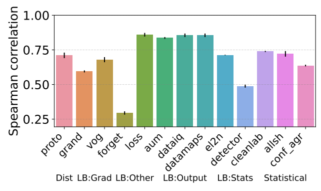
(e) Near-OoD (Covariate)



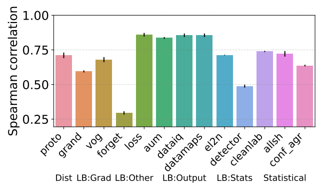
(f) Near-OoD (Domain)



(g) Far-OoD



(h) Atypical (Shift)



(i) Atypical (Zoom)

Figure 9: Stability/consistency to model types: higher Spearman rank correlations indicate greater consistency and stability.

Takeaway. Of course, the LeNet and ResNet-18 are significantly different in model type and size. However, we find that compared to the results in the main paper, we see minimal degradation in the Spearman rank correlation. This indicates that the backbone model to do the hardness characterization is minimally impacted by the backbone model. i.e. those HCMs which were the most stable and consistent remain the most stable and consistent.

D.3 Stability/Consistency: Parameterizations

Overview. We can train the backbone models differently, with different parameterizations. For instance, with different numbers of epochs, 10 or 20. We wish to assess stability/consistency with respect to different training parameterizations when performing the HCM evaluation. As discussed in the main paper, we can quantify this based on the Spearman rank correlation — i.e. are the HCMs ordering the samples consistently in a stable manner?

We make a head-to-head comparison of scores obtained from the HCMs applied to backbone models trained for 10 and 20 epochs. Ideally, we still would want a high Spearman correlation indicating sample order is maintained.

Results. Figure 10 shows the results across multiple hardness types.

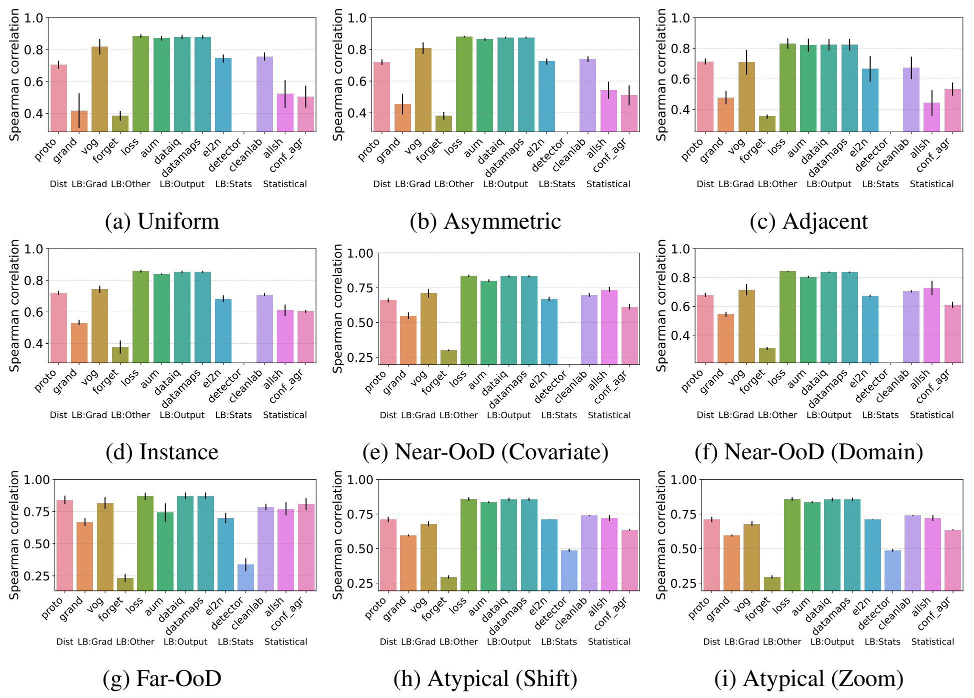


Figure 10: Stability/consistency to parameterizations: higher Spearman rank correlations indicate greater consistency and stability.

Takeaway. We investigated the stability/consistency of the model parameterization in terms of the number of epochs used to train the backbone models. We find the Spearman correlations are not as affected by the epoch number as potentially expected. We see the magnitudes of the Spearman correlation for the different HCMs are similar to that of the main paper. Moreover, the HCMs, which were the most stable and consistent, remain the most stable and consistent.

This result is understandable, as it was shown in (Seedat et al., 2022a) that the majority of variation comes from the early epochs (before 10 epochs), hence after the training dynamics on which most of the metrics are computed stabilize. This means that it won't affect our HCM characterization whether we continue training the backbone till 20 epochs. This result simply mirrors this result from the literature and highlights for benchmarking purposes that we don't need to train the backbone model for too many epochs in order to have a good assessment of the HCMs' capabilities.

D.4 ADDITIONAL AGGREGATED RESULTS

Overview. In the main paper, we presented aggregate results for 6 out of 9 hardness manifestations or types. We present the results for the remaining three Adjacent mislabeling, Near-OOD (Domain) and Atypical (Zoom), below in Figures 11-14.

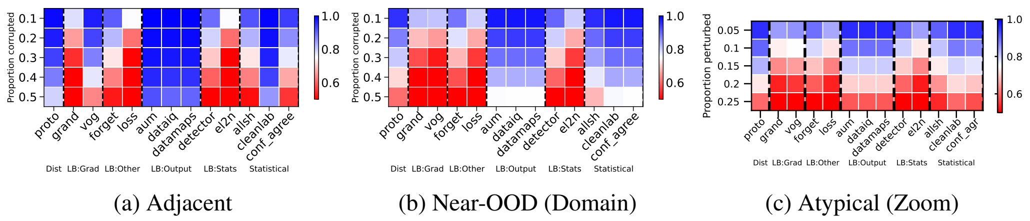

(a) Adjacent (b) Near-OOD (Domain) (c) Atypical (Zoom)

Figure 11: AUPRC for different HCMs for different hardness types aggregated across setups. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse. We see the variability of HCM capabilities across hardness types and proportions.

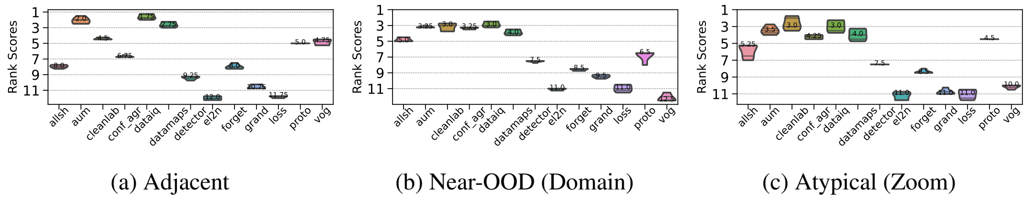

(a) Adjacent (b) Near-OOD (Domain) (c) Atypical (Zoom)

Figure 12: Performance rankings of HCMs vary depending on the hardness type.

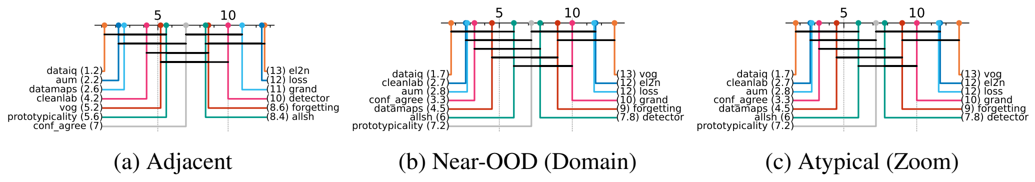

(a) Adjacent (b) Near-OOD (Domain) (c) Atypical (Zoom)

Figure 13: Critical difference diagrams highlight that similar categories/classes of HCMs *do not* have a statistically significant difference in their performance, indicated by the horizontal black lines linking HCMs which are not statistically different. The numbers in brackets denote mean rank.

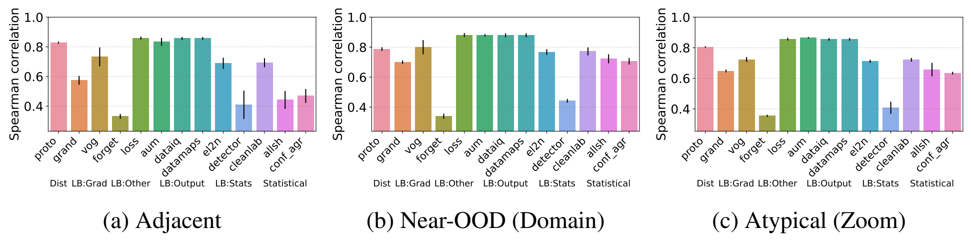
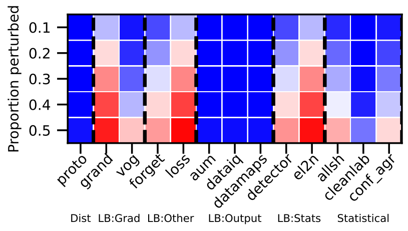

(a) Adjacent (b) Near-OOD (Domain) (c) Atypical (Zoom)

Figure 14: Certain classes of HCMs are more stable and consistent than others (retaining rank order), with higher Spearman rank correlations for multiple runs of the HCM on the same data

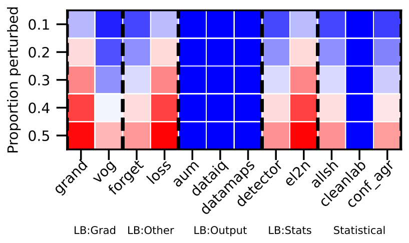
D.5 Individual results - Images

D.5.1 Mislabeling: Adjacent

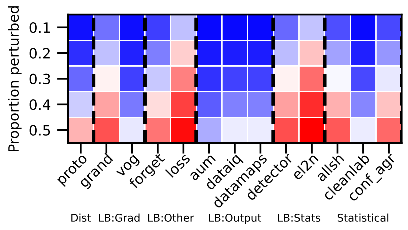
Heatmaps.



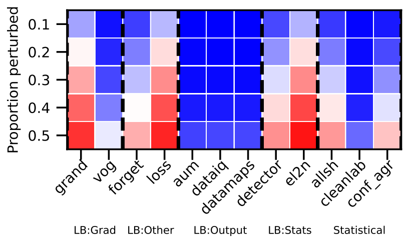
(a) MNIST - LeNet



(b) MNIST - ResNet

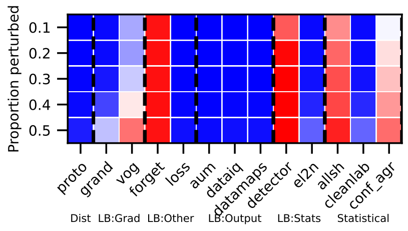


(c) CIFAR - LeNet

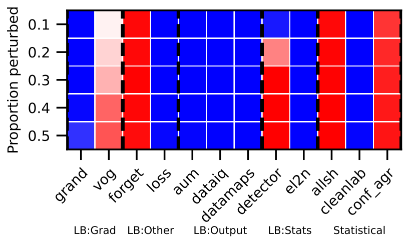


(d) CIFAR - ResNet

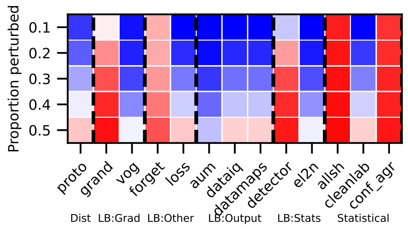
Figure 15: Adjacent mislabeling – AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.



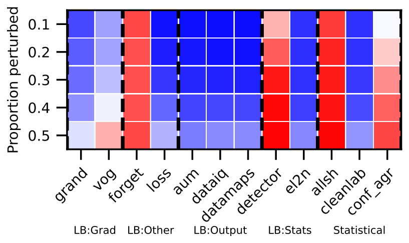
(a) MNIST - LeNet



(b) MNIST - ResNet



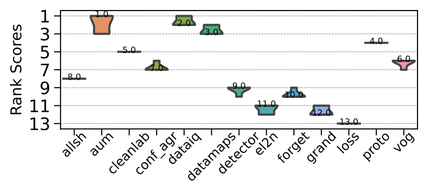
(c) CIFAR - LeNet



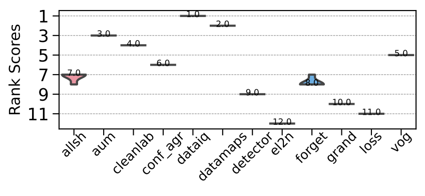
(d) CIFAR - ResNet

Figure 16: Adjacent mislabeling – AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

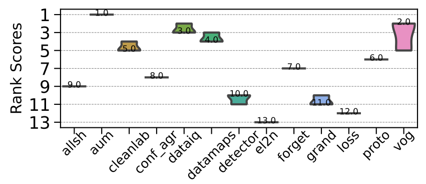
Rankings.



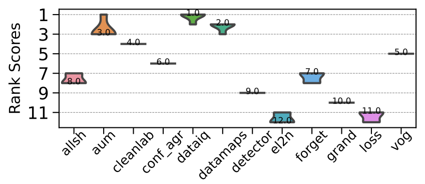
(a) MNIST - LeNet



(b) MNIST - ResNet

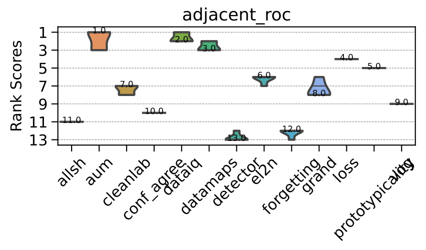


(c) CIFAR - LeNet

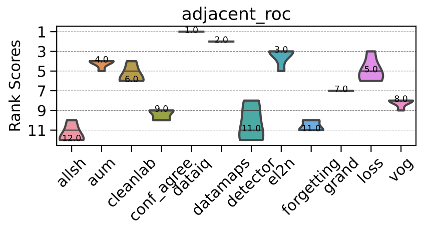


(d) CIFAR - ResNet

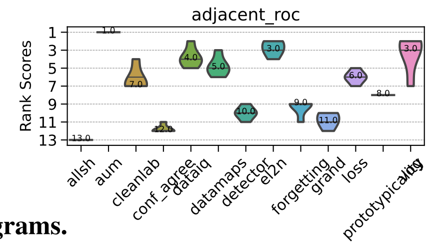
Figure 17: Adjacent mislabeling – AUPRC. Ranking of HCMs.

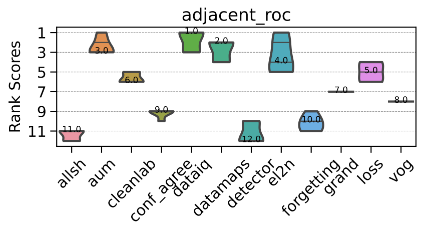


(a) MNIST - LeNet

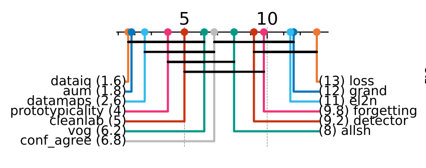


(b) MNIST - ResNet

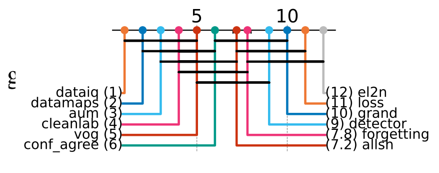




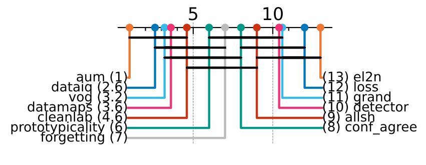
CD diagrams.



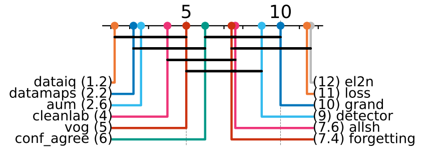
(a) MNIST - LeNet



(b) MNIST - ResNet

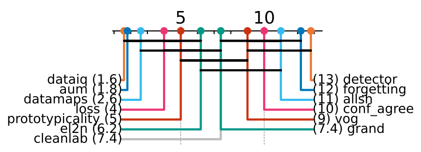


(c) CIFAR - LeNet

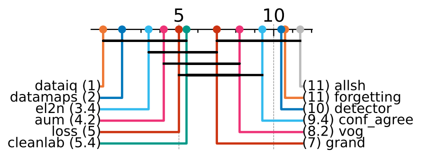


(d) CIFAR - ResNet

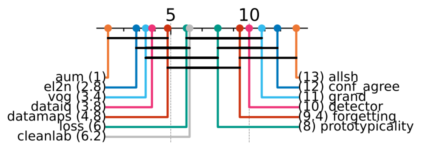
Figure 19: Adjacent mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



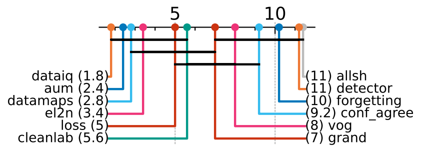
(a) MNIST - LeNet



(b) MNIST - ResNet



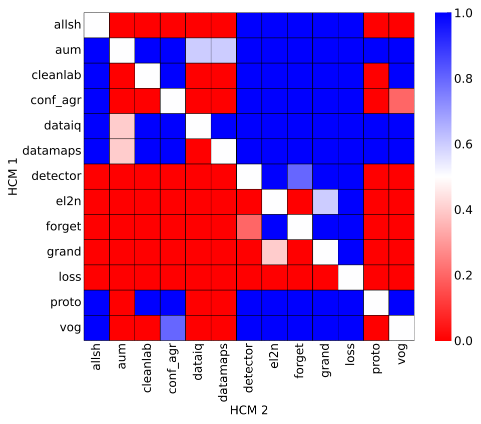
(c) CIFAR - LeNet



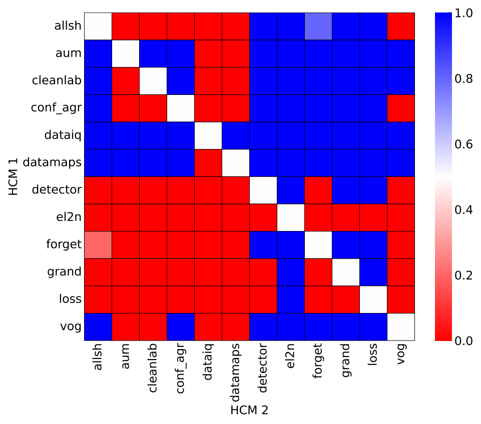
(d) CIFAR - ResNet

Figure 20: Adjacent mislabeling - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

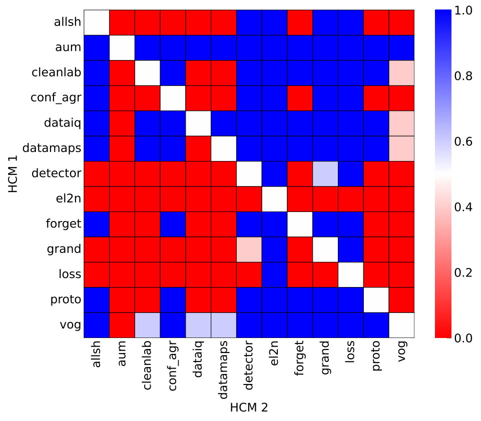
Matrix compare



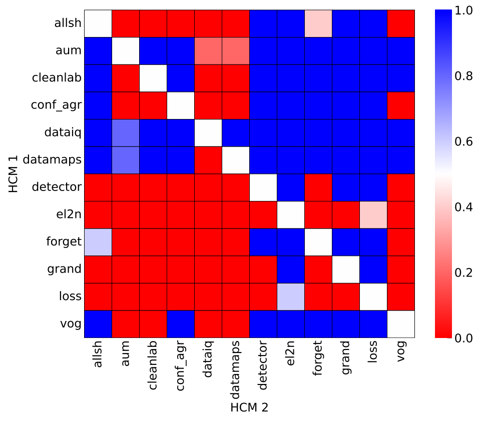
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet

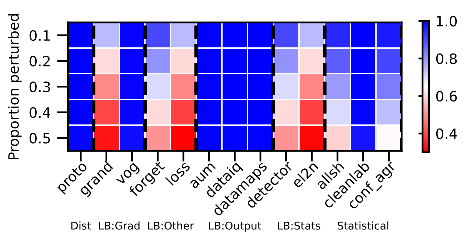


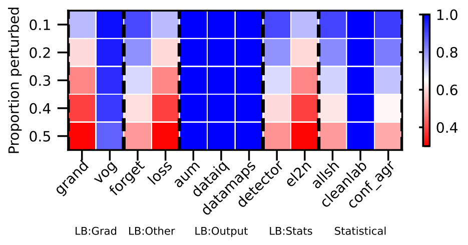
(d) CIFAR - ResNet

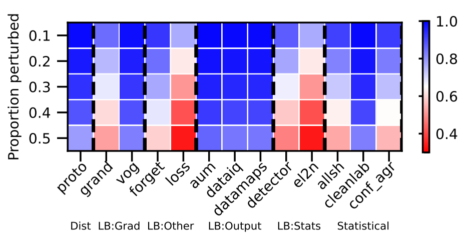
Figure 21: Adjacent mislabeling. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.2 Mislabeling: Asymmetric

Heatmaps.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

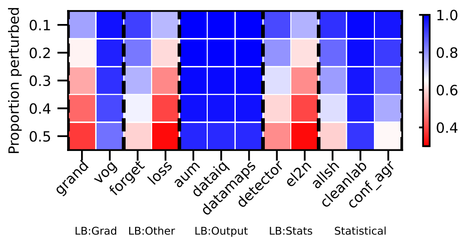
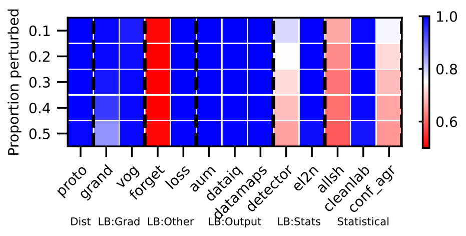
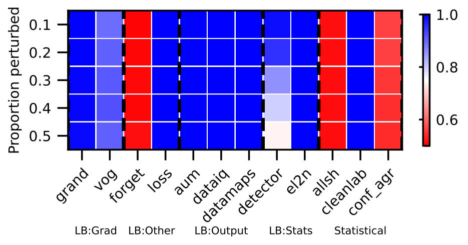
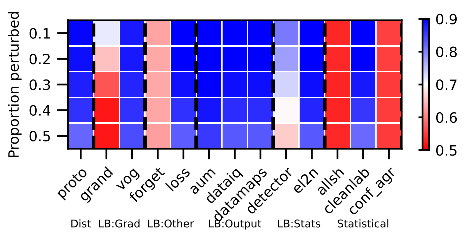

(d) CIFAR - ResNet

Figure 22: Asymmetric mislabeling - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

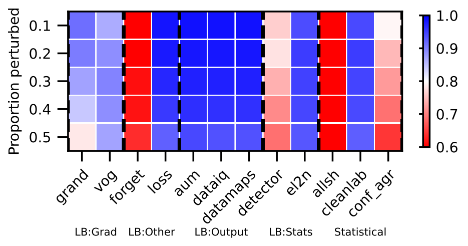
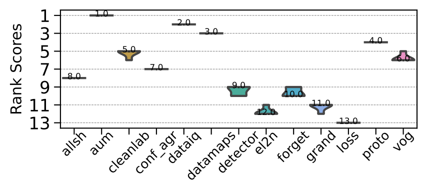
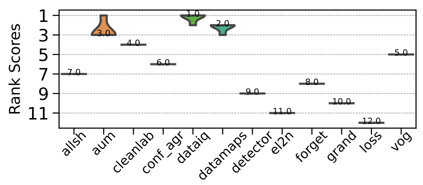

(d) CIFAR - ResNet

Figure 23: Asymmetric mislabeling - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

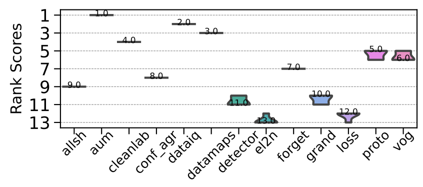
Rankings.



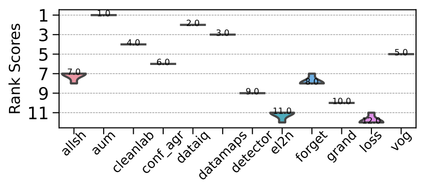
(a) MNIST - LeNet



(b) MNIST - ResNet

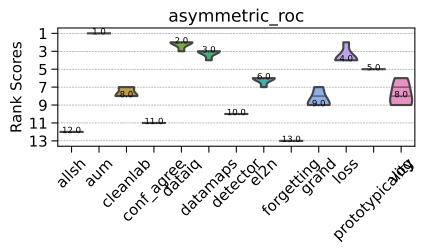


(c) CIFAR - LeNet

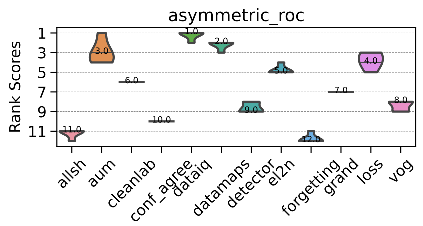


(d) CIFAR - ResNet

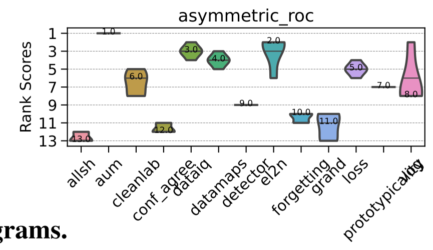
Figure 24: Asymmetric mislabeling - AUPRC. Ranking of HCMs.

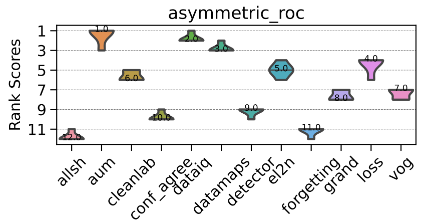


(a) MNIST - LeNet

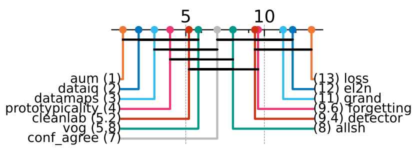


(b) MNIST - ResNet

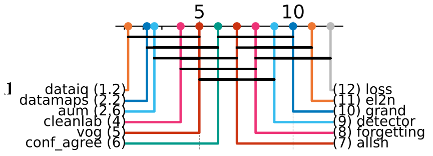




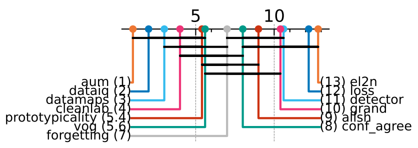
CD diagrams.



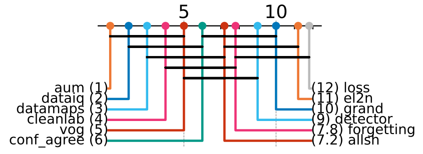
(a) MNIST - LeNet



(b) MNIST - ResNet

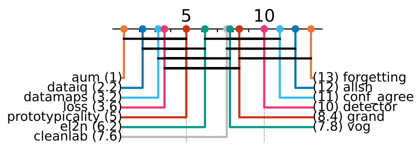


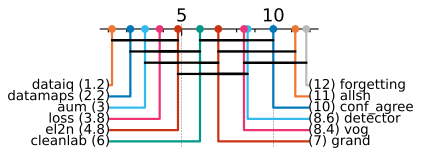
(c) CIFAR - LeNet

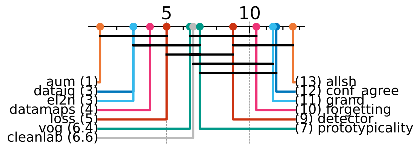


(d) CIFAR - ResNet

Figure 26: Asymmetric mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

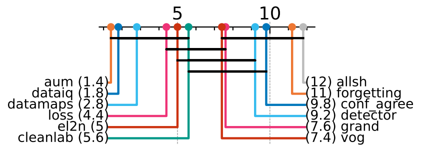
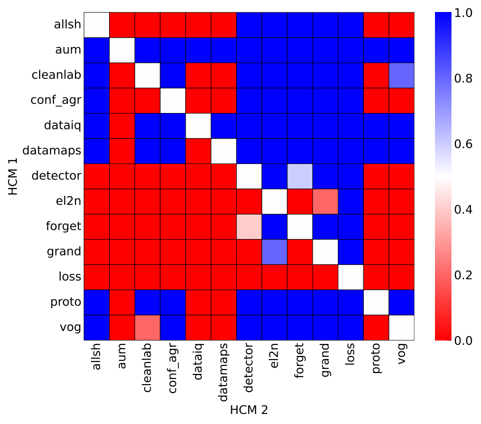
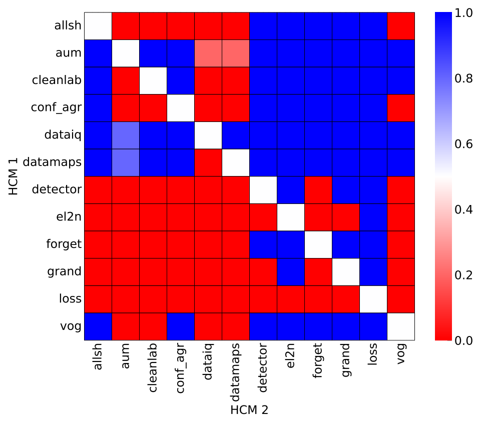
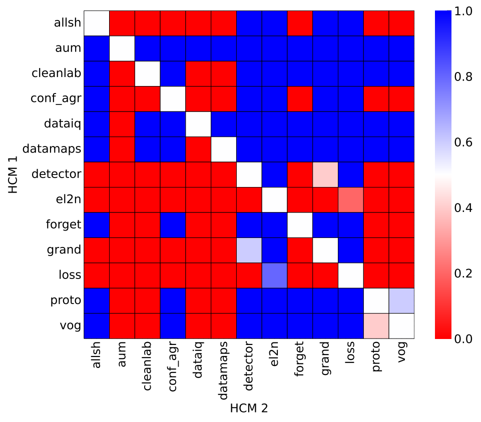

(d) CIFAR - ResNet

Figure 27: Asymmetric mislabeling - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

Matrix compare


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

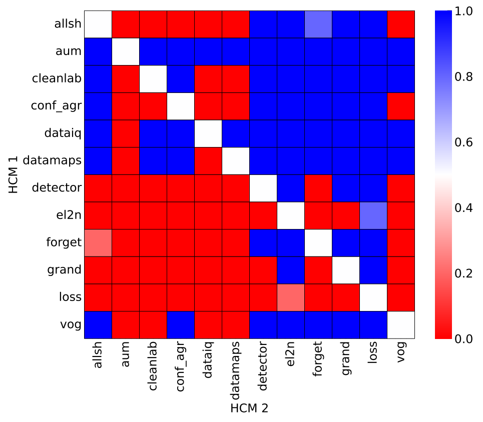
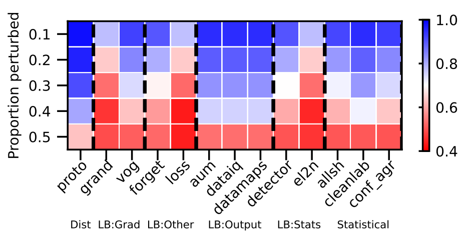

(d) CIFAR - ResNet

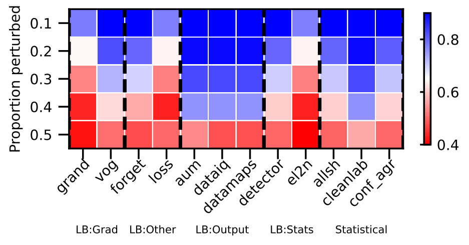
Figure 28: Asymmetric mislabeling. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.3 MISLABELING: INSTANCE

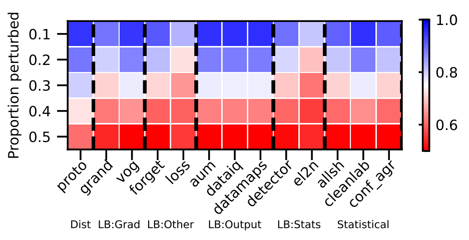
Heatmaps.



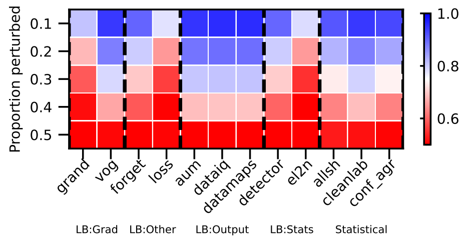
(a) MNIST - LeNet



(b) MNIST - ResNet

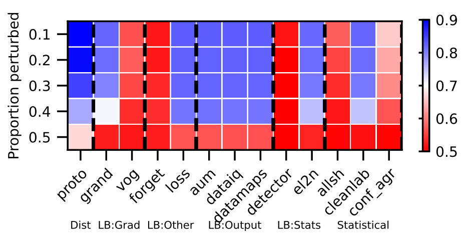


(c) CIFAR - LeNet

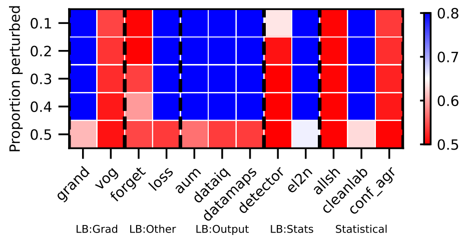


(d) CIFAR - ResNet

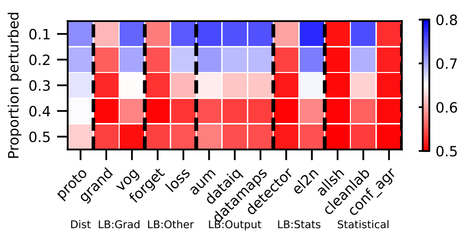
Figure 29: Instance mislabeling - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.



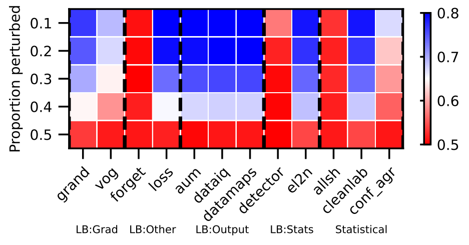
(a) MNIST - LeNet



(b) MNIST - ResNet



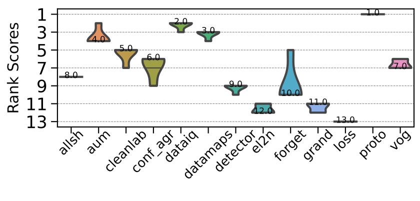
(c) CIFAR - LeNet



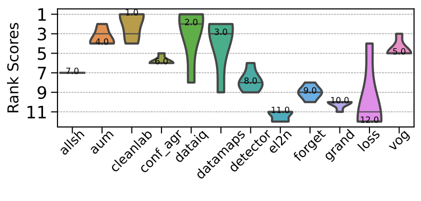
(d) CIFAR - ResNet

Figure 30: Instance mislabeling - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

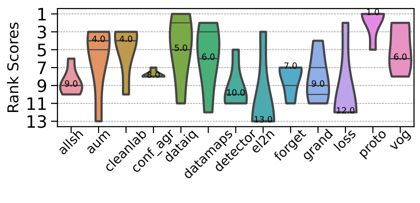
Rankings.



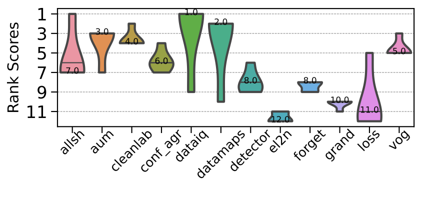
(a) MNIST - LeNet



(b) MNIST - ResNet

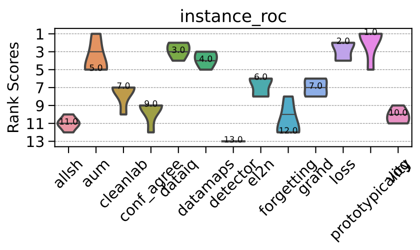


(c) CIFAR - LeNet

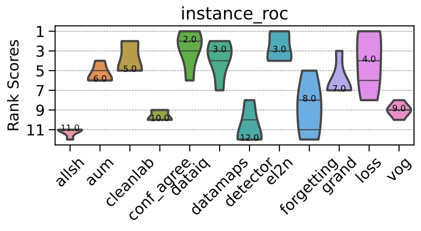


(d) CIFAR - ResNet

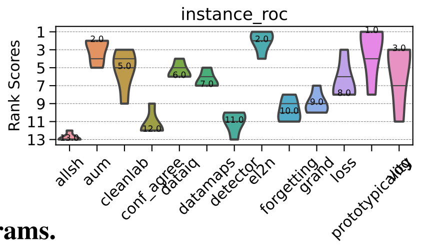
Figure 31: Instance mislabeling - AUPRC. Ranking of HCMs.

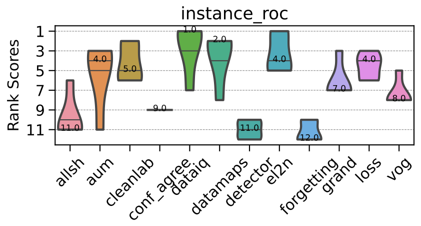


(a) MNIST - LeNet

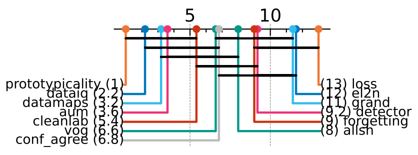


(b) MNIST - ResNet

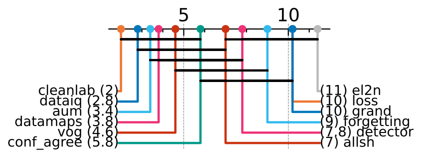




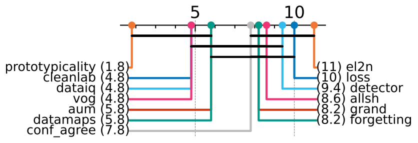
CD diagrams.



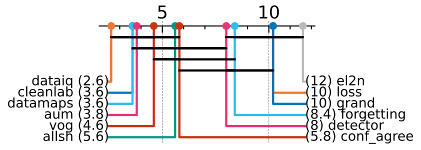
(a) MNIST - LeNet



(b) MNIST - ResNet

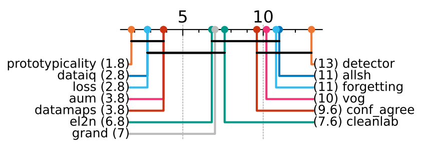


(c) CIFAR - LeNet

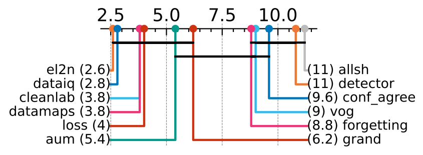


(d) CIFAR - ResNet

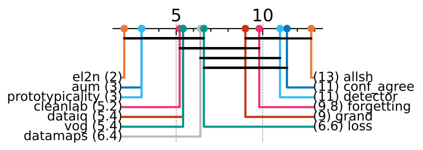
Figure 33: Instance mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



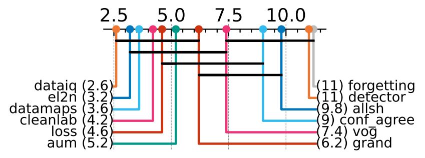
(a) MNIST - LeNet



(b) MNIST - ResNet



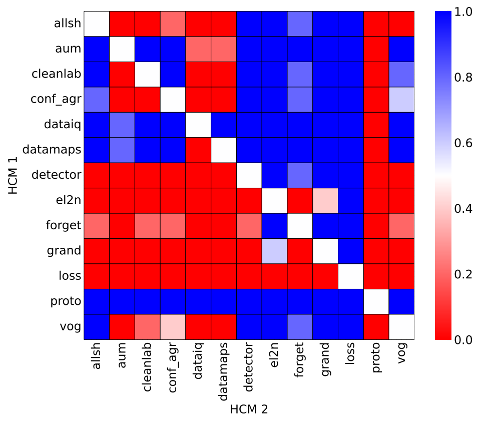
(c) CIFAR - LeNet



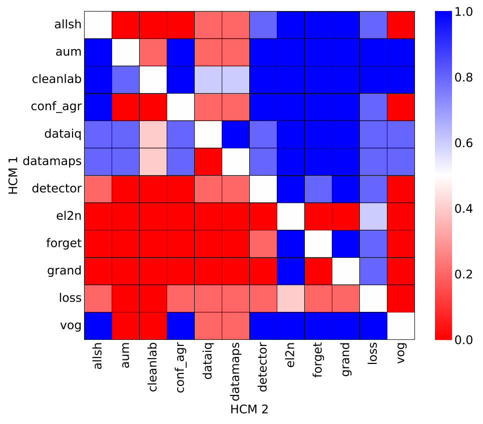
(d) CIFAR - ResNet

Figure 34: Instance mislabeling - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

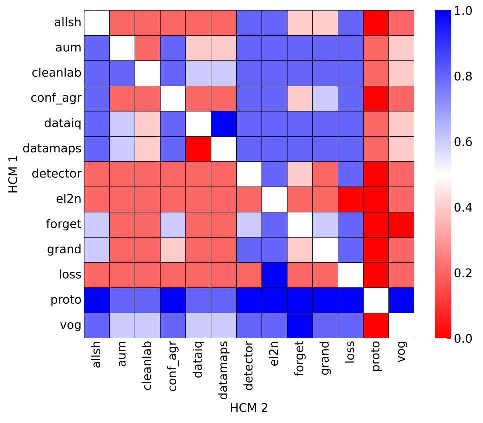
Matrix compare



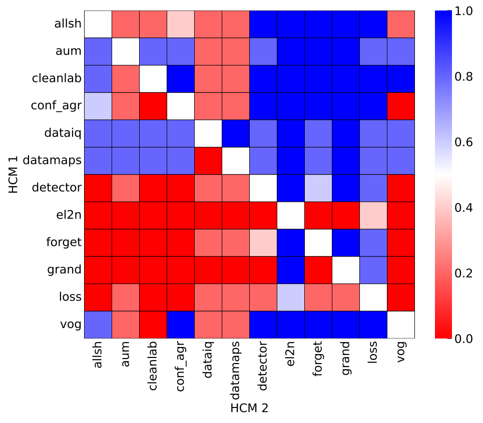
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet

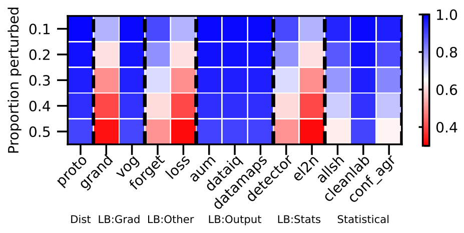


(d) CIFAR - ResNet

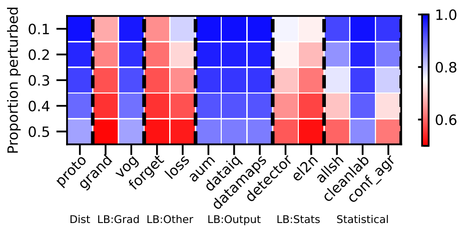
Figure 35: Instance mislabeling. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.4 Mislabeling: Uniform

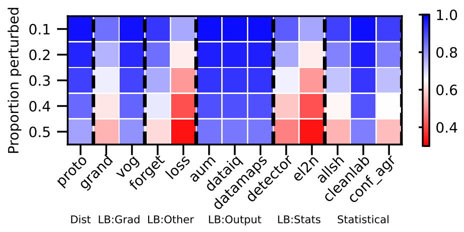
Heatmaps.



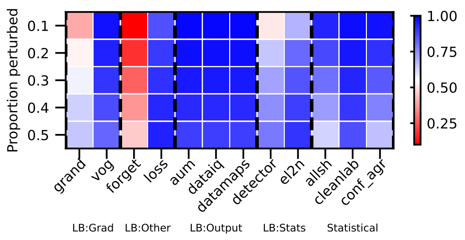
(a) MNIST - LeNet



(b) MNIST - ResNet

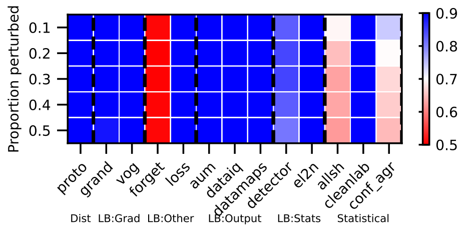


(c) CIFAR - LeNet

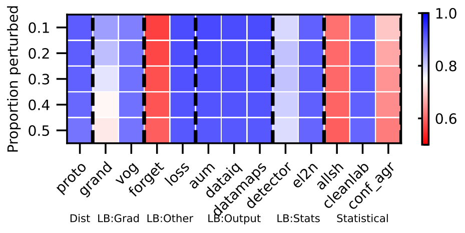


(d) CIFAR - ResNet

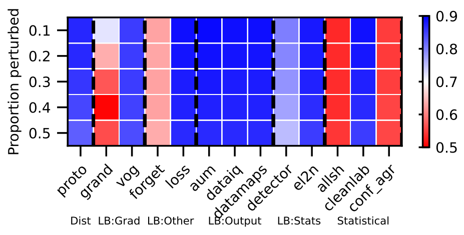
Figure 36: Uniform mislabeling - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.



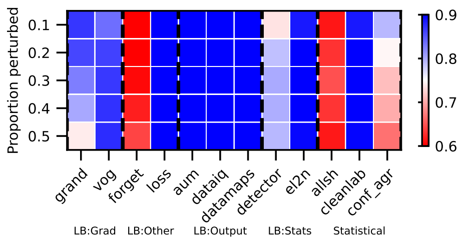
(a) MNIST - LeNet



(b) MNIST - ResNet



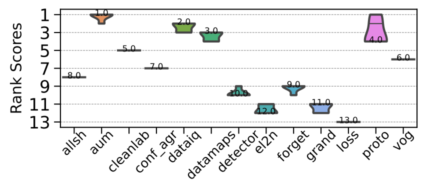
(c) CIFAR - LeNet



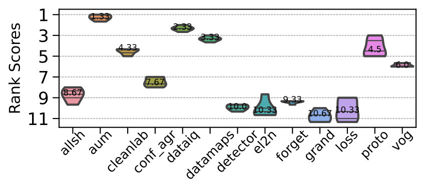
(d) CIFAR - ResNet

Figure 37: Uniform mislabeling - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

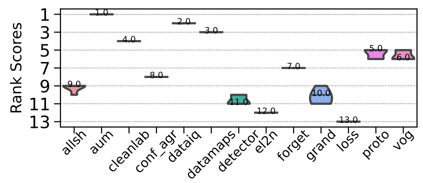
Rankings.



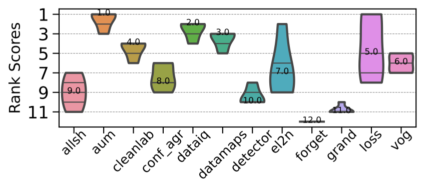
(a) MNIST - LeNet



(b) MNIST - ResNet

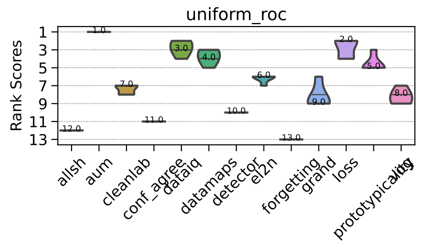


(c) CIFAR - LeNet

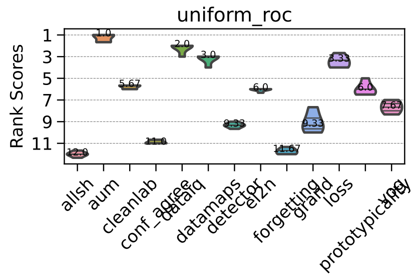


(d) CIFAR - ResNet

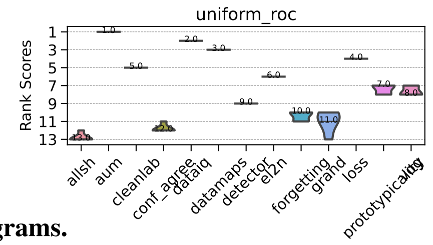
Figure 38: Uniform mislabeling - AUPRC. Ranking of HCMs.

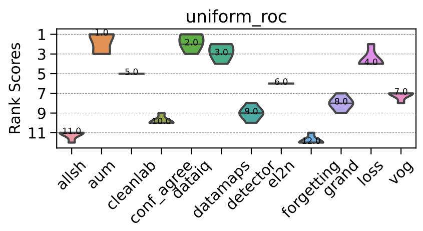


(a) MNIST - LeNet

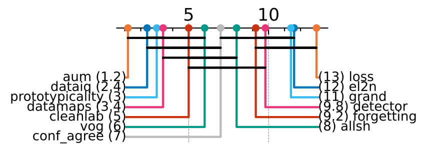


(b) MNIST - ResNet

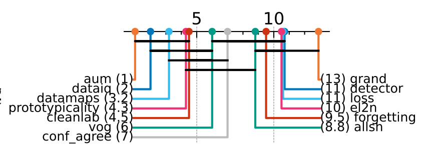




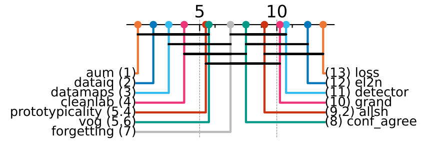
CD diagrams.



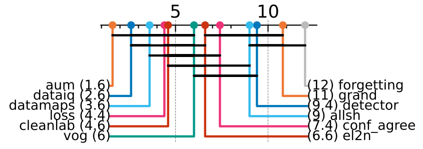
(a) MNIST - LeNet



(b) MNIST - ResNet

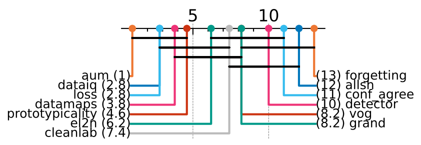


(c) CIFAR - LeNet

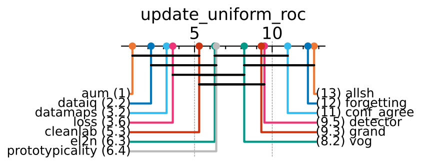


(d) CIFAR - ResNet

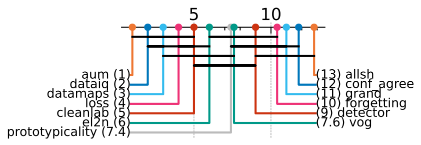
Figure 40: Uniform mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



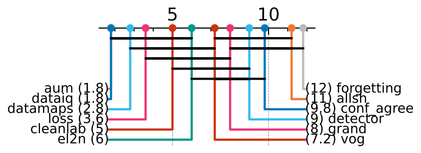
(a) MNIST - LeNet



(b) MNIST - ResNet



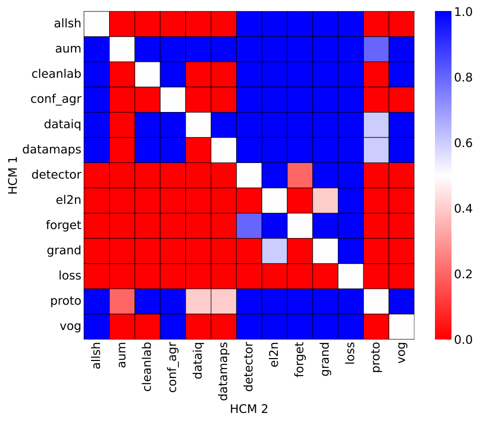
(c) CIFAR - LeNet



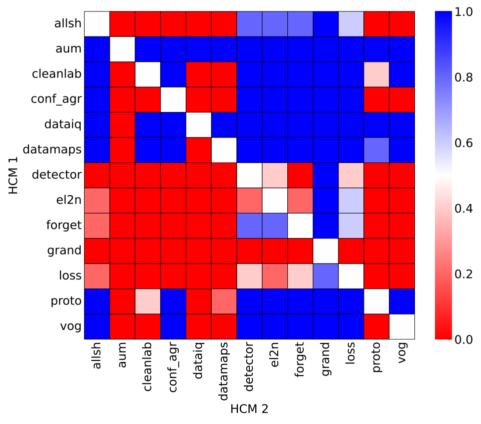
(d) CIFAR - ResNet

Figure 41: Uniform mislabeling - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

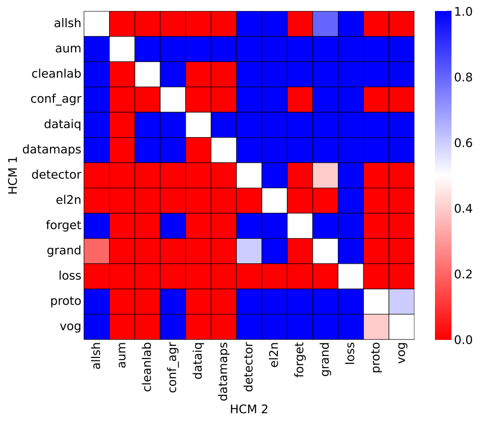
Matrix compare



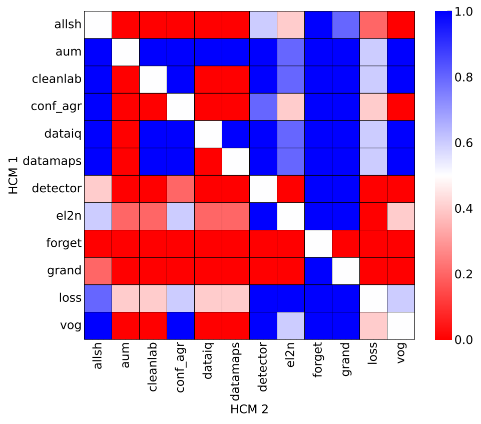
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet

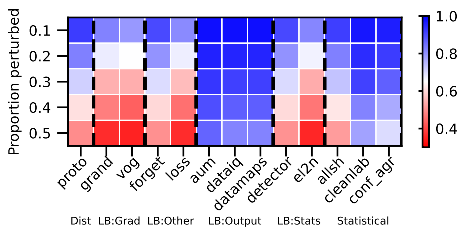


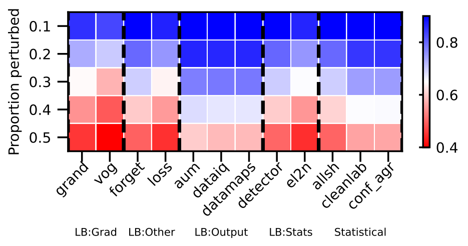
(d) CIFAR - ResNet

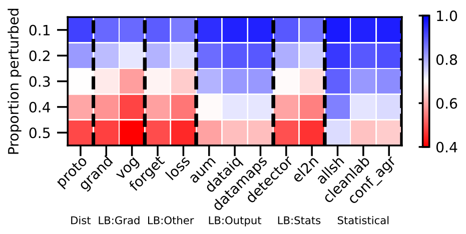
Figure 42: Uniform mislabeling. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.5 Near OOD: Covariate

Heatmaps.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

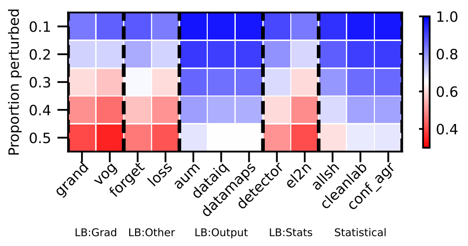
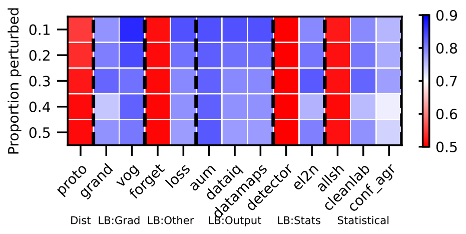
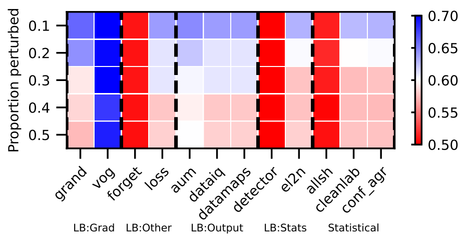
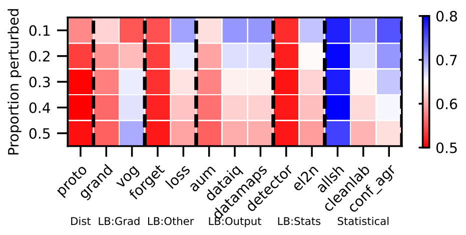

(d) CIFAR - ResNet

Figure 43: Near OOD: Covariate - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

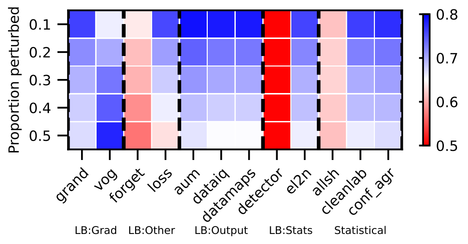
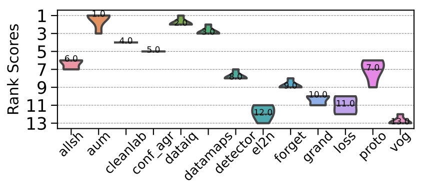
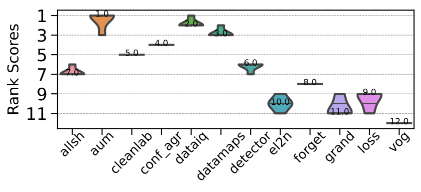

(d) CIFAR - ResNet

Figure 44: Near OOD: Covariate - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

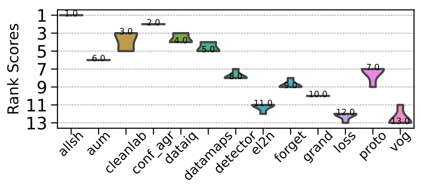
Rankings.



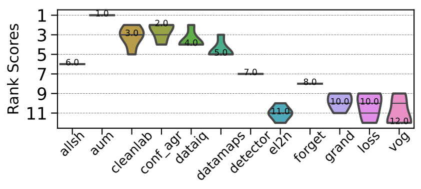
(a) MNIST - LeNet



(b) MNIST - ResNet

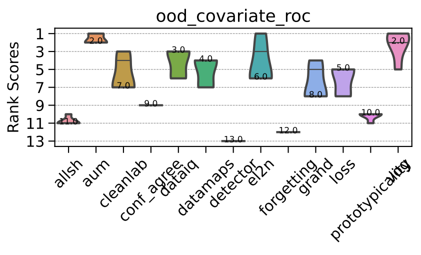


(c) CIFAR - LeNet

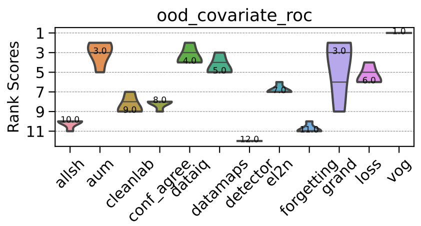


(d) CIFAR - ResNet

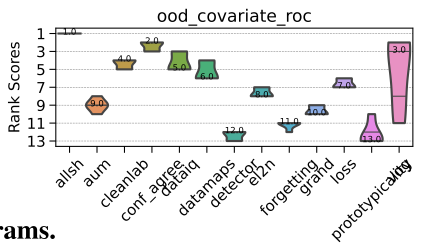
Figure 45: Near OOD: Covariate - AUPRC. Ranking of HCMs.



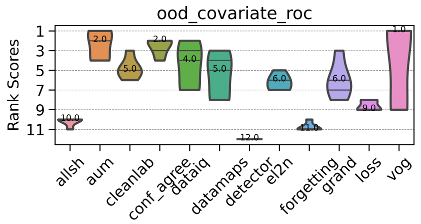
(a) MNIST - LeNet

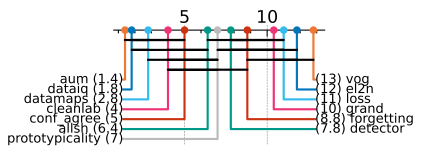


(b) MNIST - ResNet

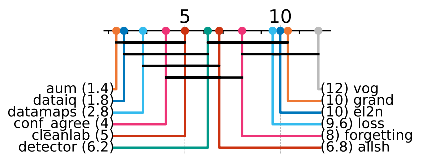


CD diagrams.

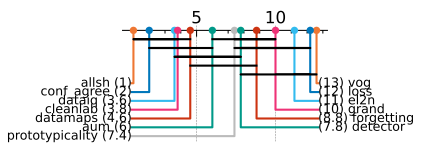




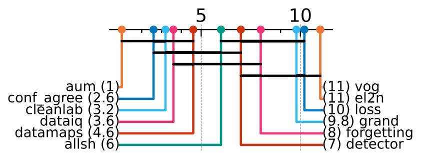
(a) MNIST - LeNet



(b) MNIST - ResNet

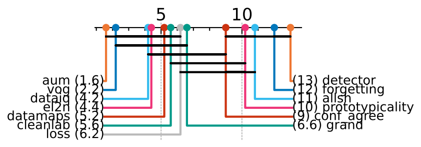


(c) CIFAR - LeNet

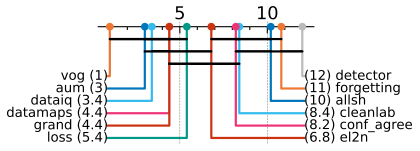


(d) CIFAR - ResNet

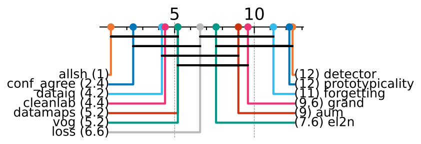
Figure 47: Near OOD: Covariate - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



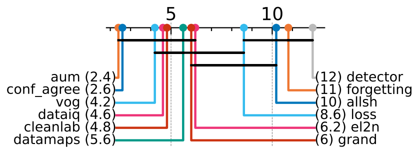
(a) MNIST - LeNet



(b) MNIST - ResNet



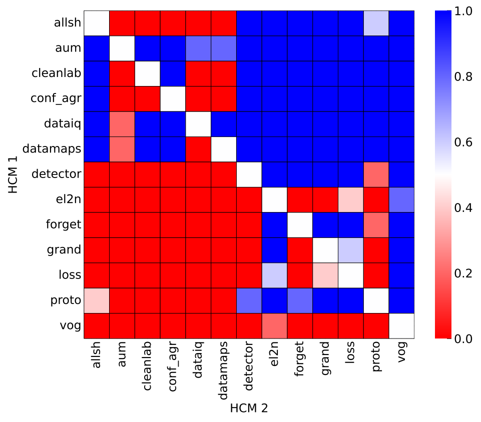
(c) CIFAR - LeNet



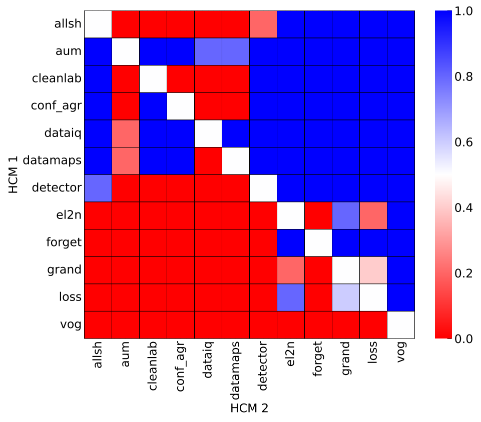
(d) CIFAR - ResNet

Figure 48: Near OOD: Covariate - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

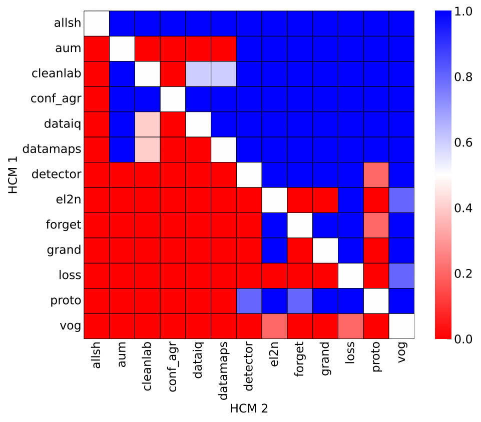
Matrix compare



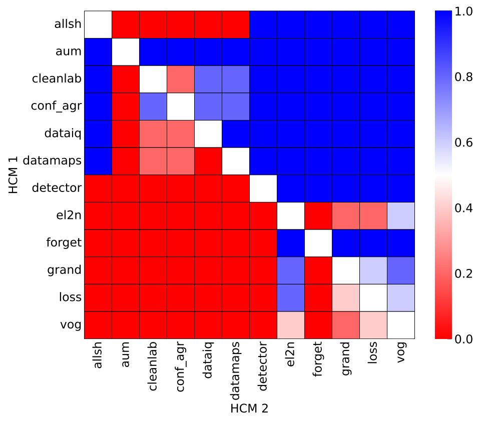
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet

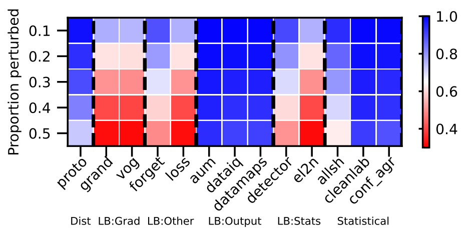


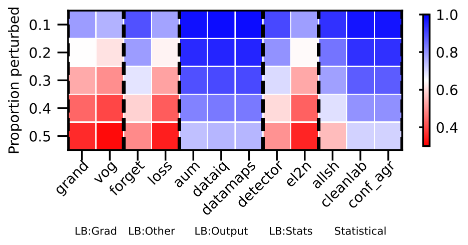
(d) CIFAR - ResNet

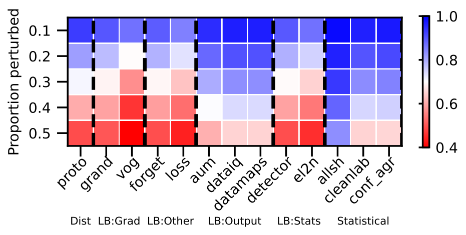
Figure 49: Near OOD: Covariate. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.6 Near OOD: Domain

Heatmaps.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

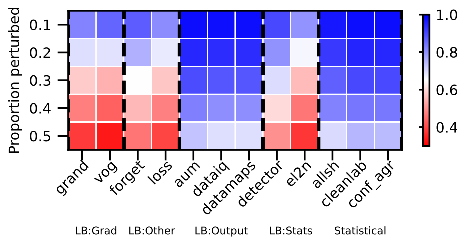
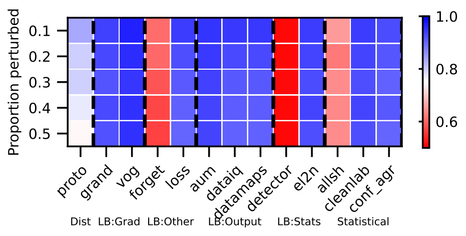
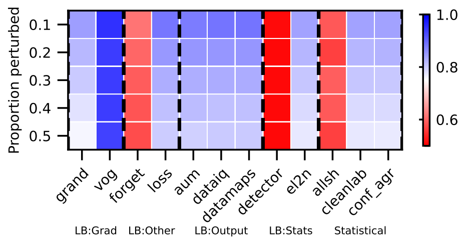
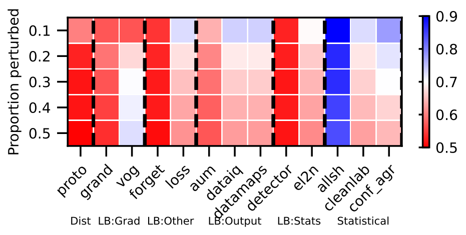

(d) CIFAR - ResNet

Figure 50: Near OOD: Domain - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

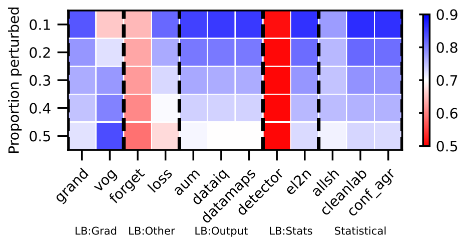
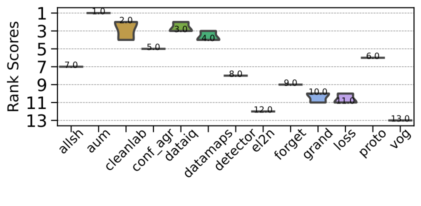
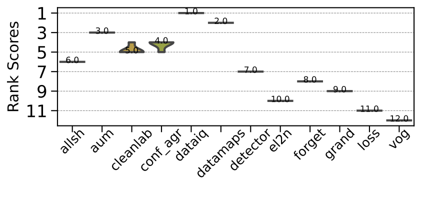

(d) CIFAR - ResNet

Figure 51: Near OOD: Domain - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

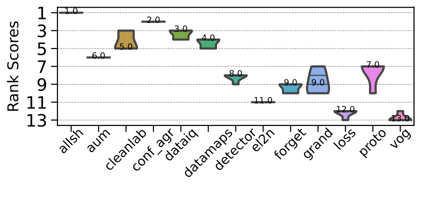
Rankings.



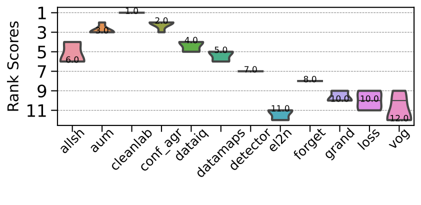
(a) MNIST - LeNet



(b) MNIST - ResNet

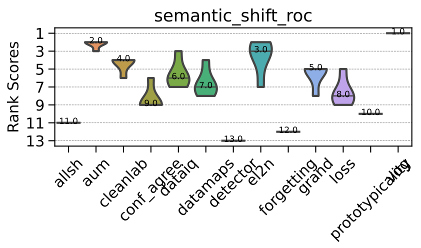


(c) CIFAR - LeNet

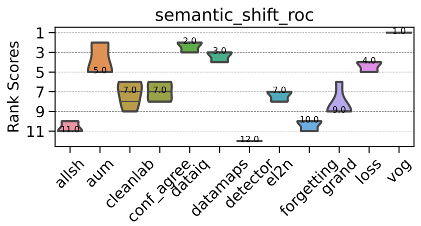


(d) CIFAR - ResNet

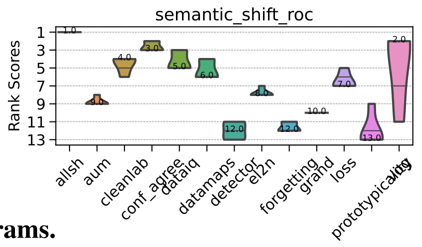
Figure 52: Near OOD: Domain - AUPRC. Ranking of HCMs.



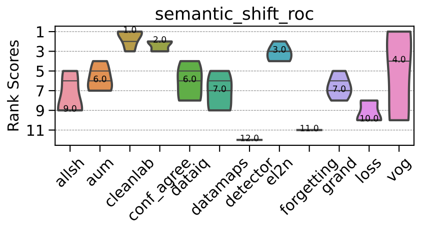
(a) MNIST - LeNet

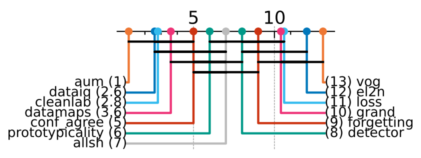


(b) MNIST - ResNet

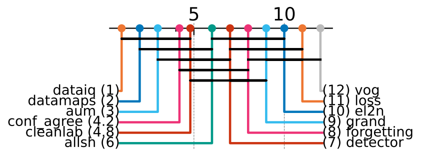


CD diagrams.

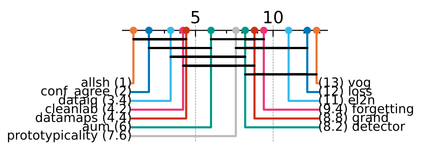




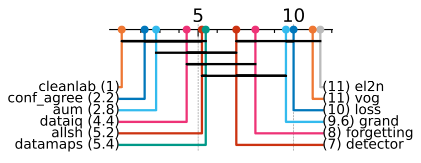
(a) MNIST - LeNet



(b) MNIST - ResNet

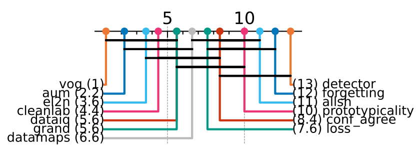


(c) CIFAR - LeNet

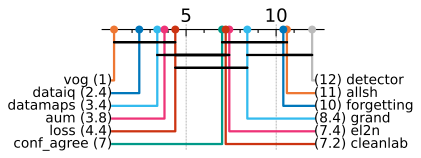


(d) CIFAR - ResNet

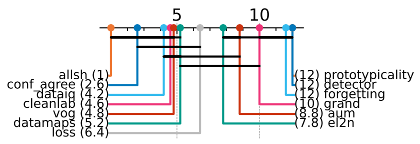
Figure 54: Near OOD: Domain - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



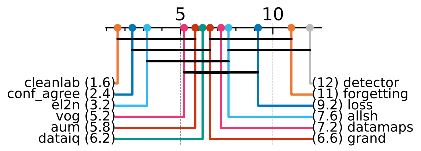
(a) MNIST - LeNet



(b) MNIST - ResNet



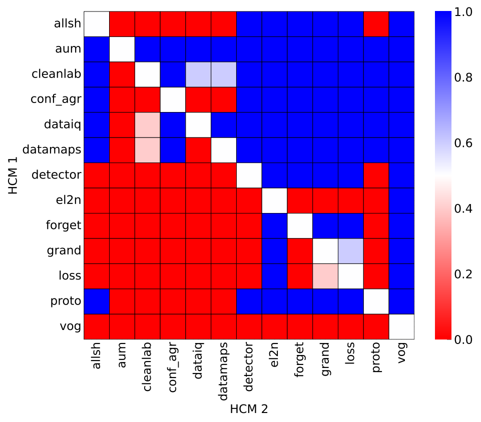
(c) CIFAR - LeNet



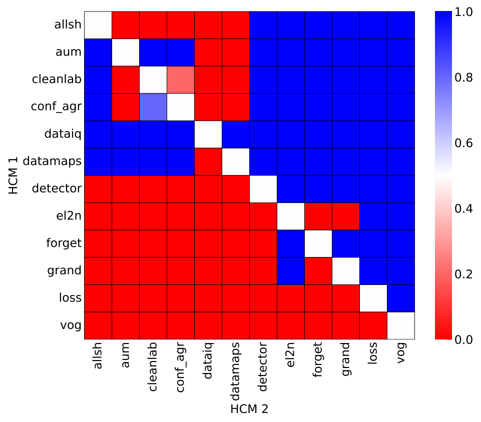
(d) CIFAR - ResNet

Figure 55: Near OOD: Domain - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

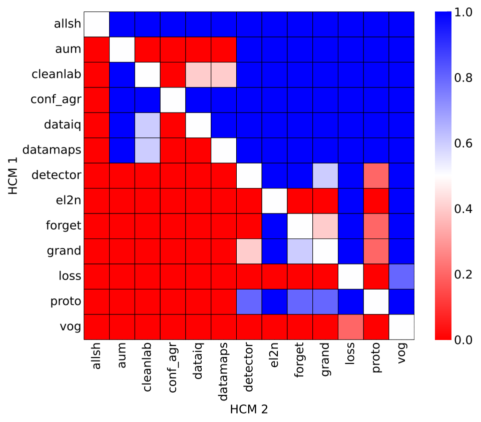
Matrix compare



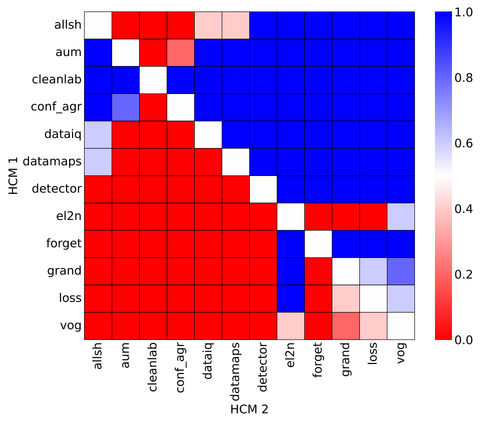
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet

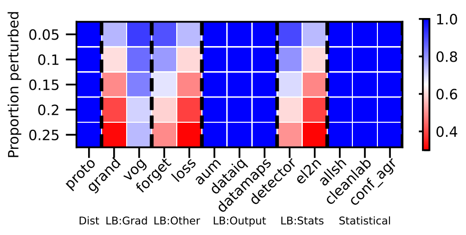


(d) CIFAR - ResNet

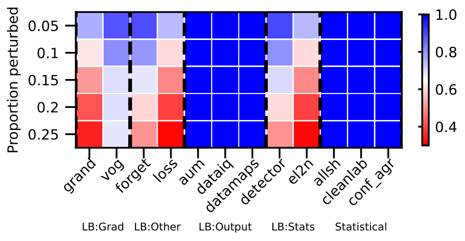
Figure 56: Near OOD: Domain. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.7 Far OoD

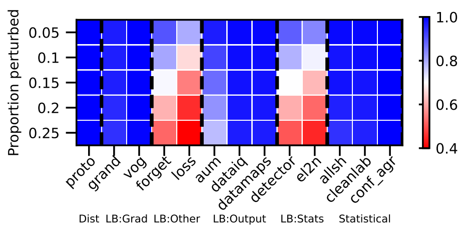
Heatmaps.



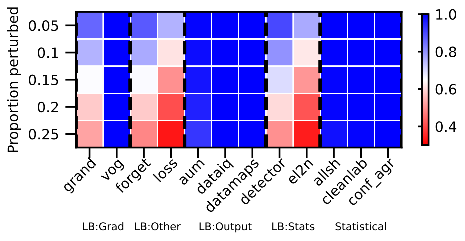
(a) MNIST - LeNet



(b) MNIST - ResNet

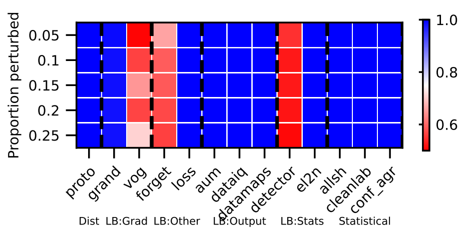


(c) CIFAR - LeNet

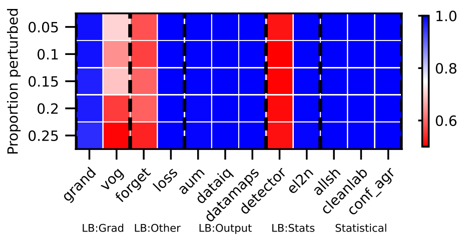


(d) CIFAR - ResNet

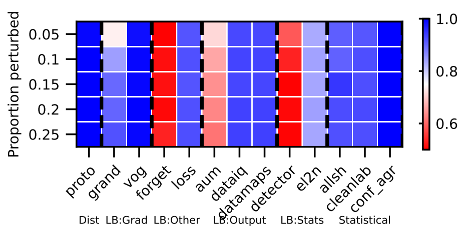
Figure 57: Far OoD - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.



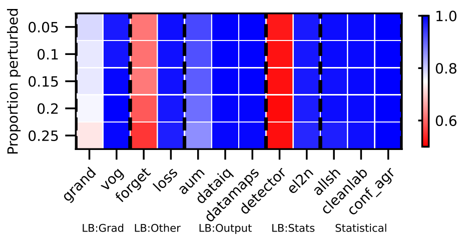
(a) MNIST - LeNet



(b) MNIST - ResNet



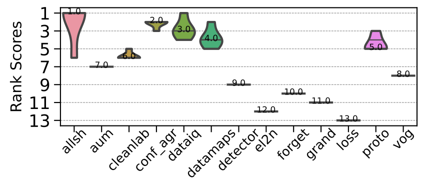
(c) CIFAR - LeNet



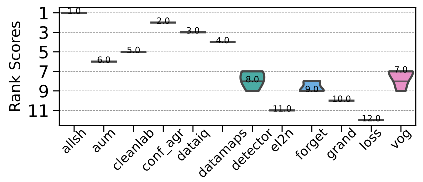
(d) CIFAR - ResNet

Figure 58: Far OoD - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

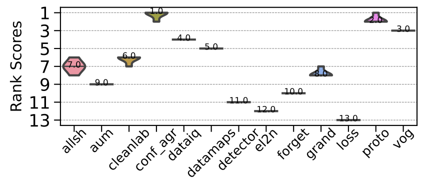
Rankings.



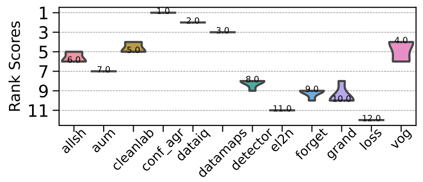
(a) MNIST - LeNet



(b) MNIST - ResNet

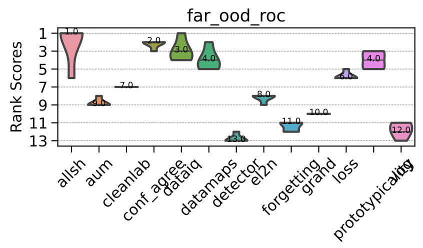


(c) CIFAR - LeNet

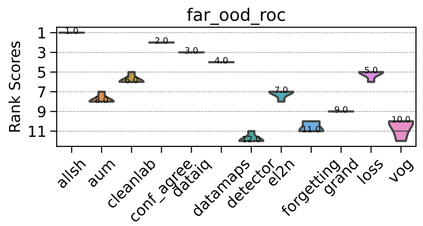


(d) CIFAR - ResNet

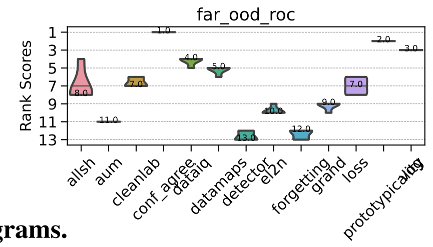
Figure 59: Far OoD - AUPRC. Ranking of HCMs.

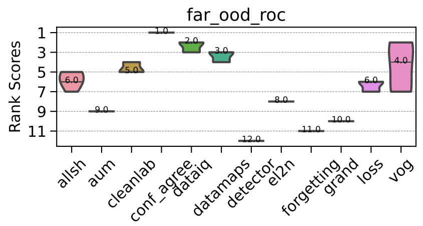


(a) MNIST - LeNet

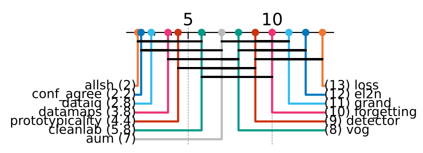


(b) MNIST - ResNet

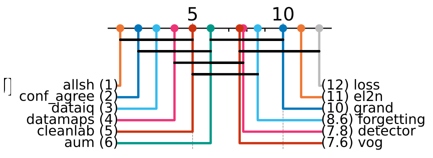




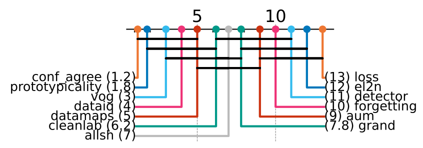
CD diagrams.



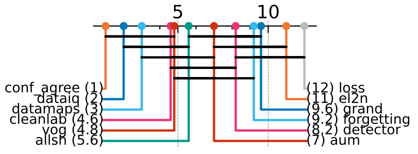
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet



(d) CIFAR - ResNet

Figure 61: Far OoD - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

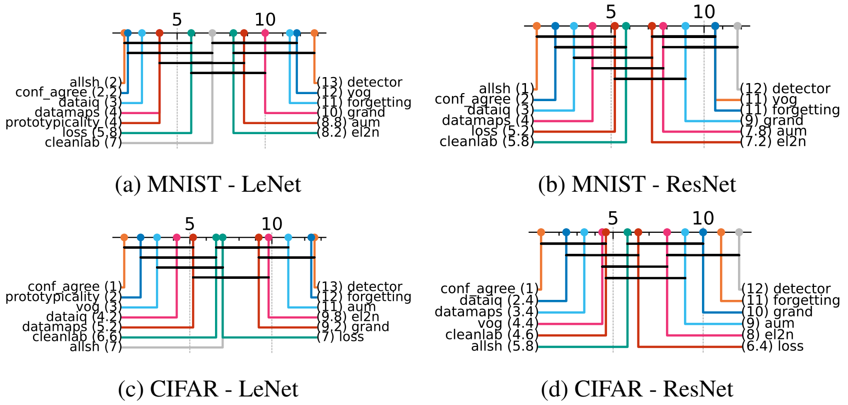


Figure 62: Far OoD - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

Matrix compare

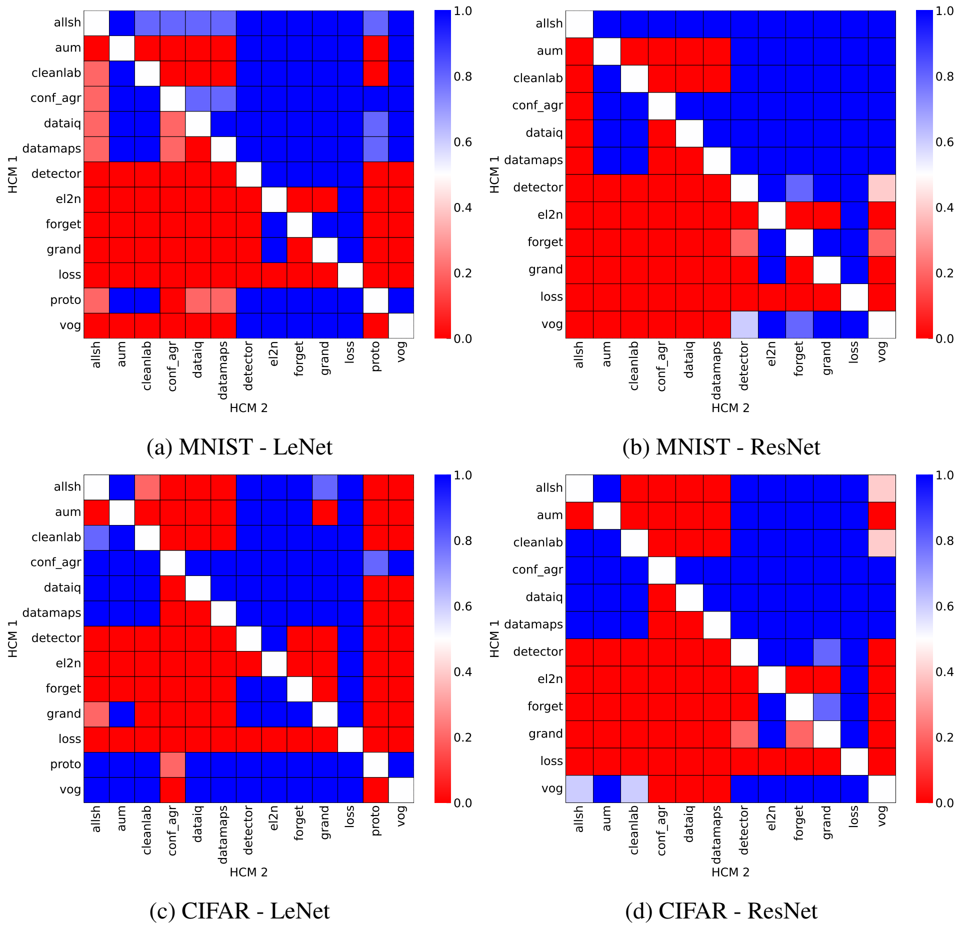
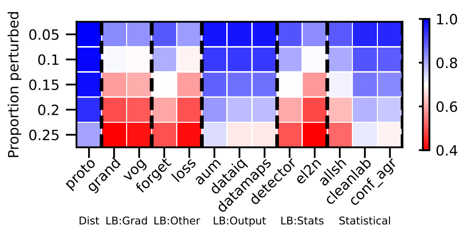
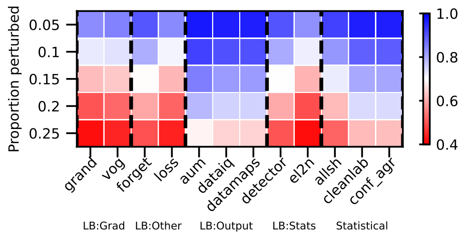


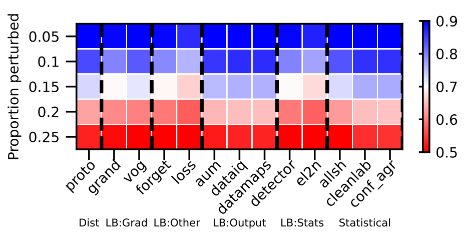
Figure 63: Far OoD. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.8 ATYPICAL: CROP SHIFT

Heatmaps.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

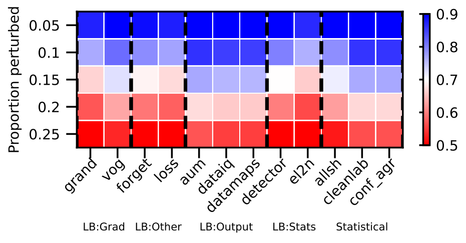
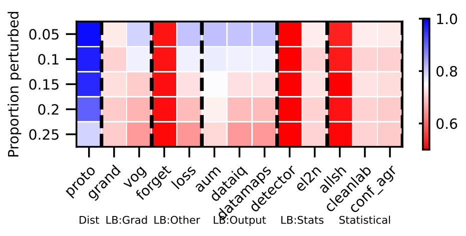
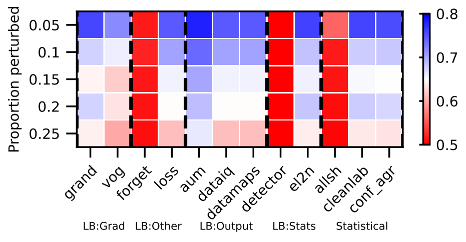
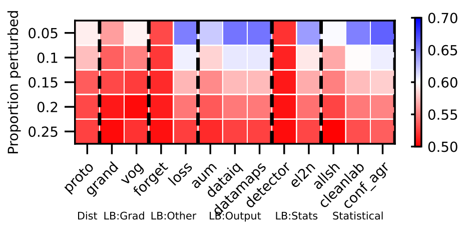

(d) CIFAR - ResNet

Figure 64: Atypical: Crop Shift - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

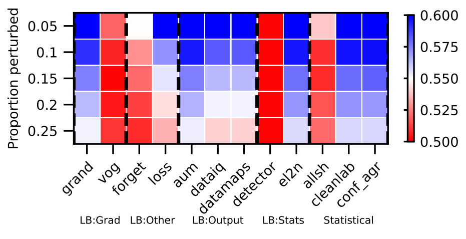
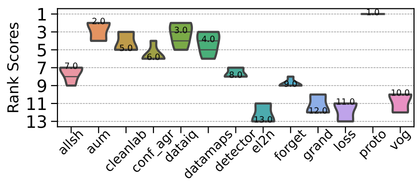
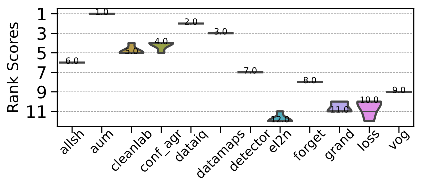

(d) CIFAR - ResNet

Figure 65: Atypical: Crop Shift - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

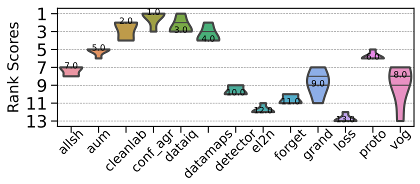
Rankings.



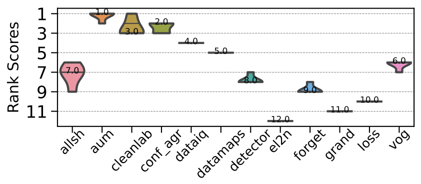
(a) MNIST - LeNet



(b) MNIST - ResNet

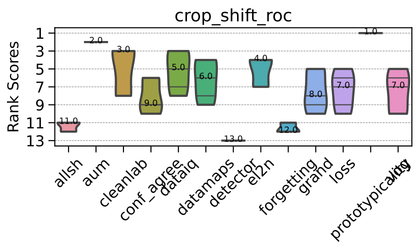


(c) CIFAR - LeNet

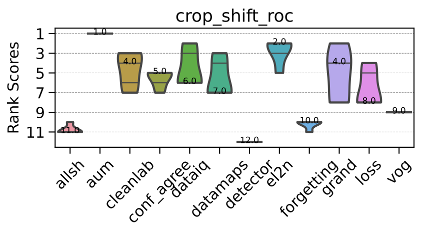


(d) CIFAR - ResNet

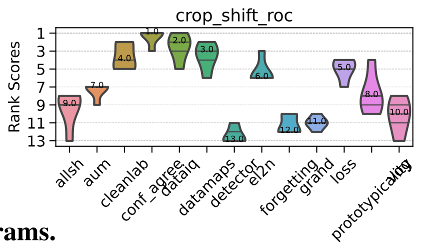
Figure 66: Atypical: Crop Shift - AUPRC. Ranking of HCMs.

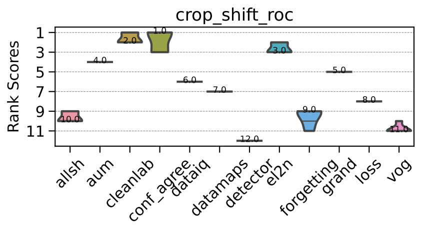


(a) MNIST - LeNet

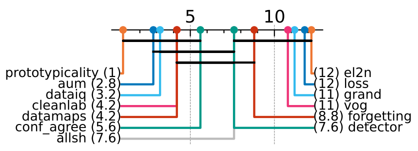


(b) MNIST - ResNet

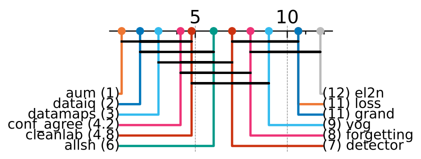




CD diagrams.

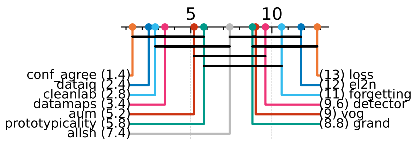


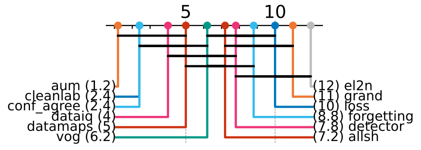
t



(a) MNIST - LeNet

(b) MNIST - ResNet

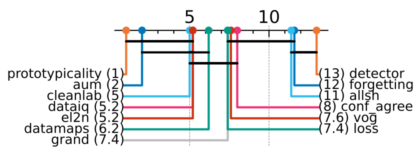




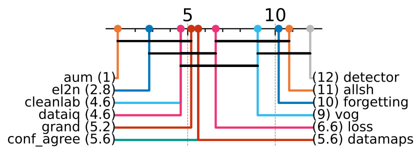
(c) CIFAR - LeNet

(d) CIFAR - ResNet

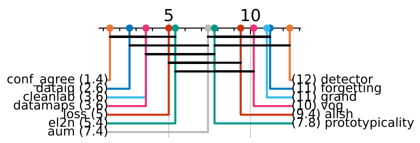
Figure 68: Atypical: Crop Shift - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



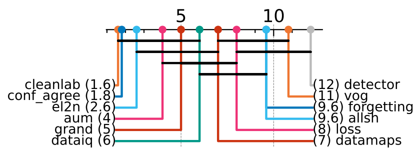
(a) MNIST - LeNet



(b) MNIST - ResNet



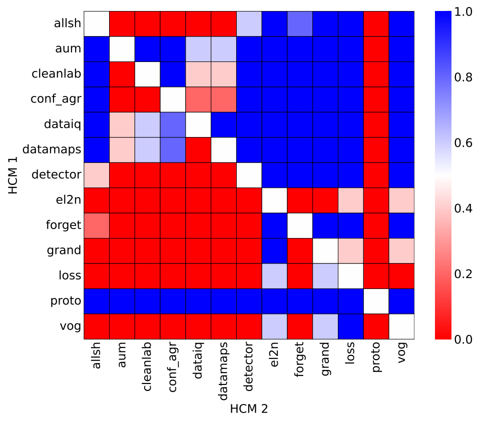
(c) CIFAR - LeNet



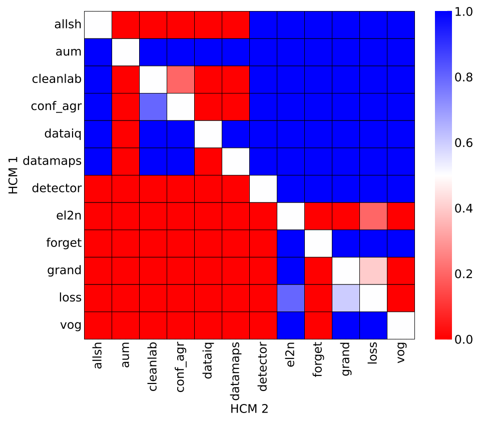
(d) CIFAR - ResNet

Figure 69: Atypical: Crop Shift - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

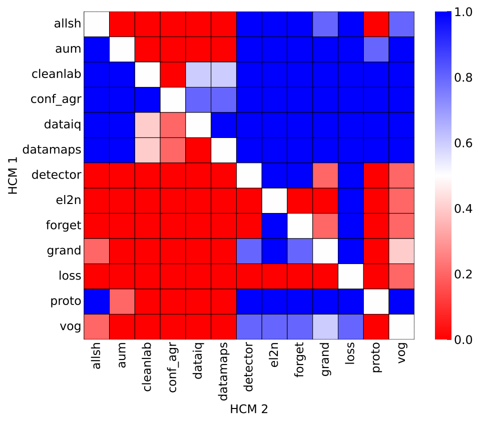
Matrix compare



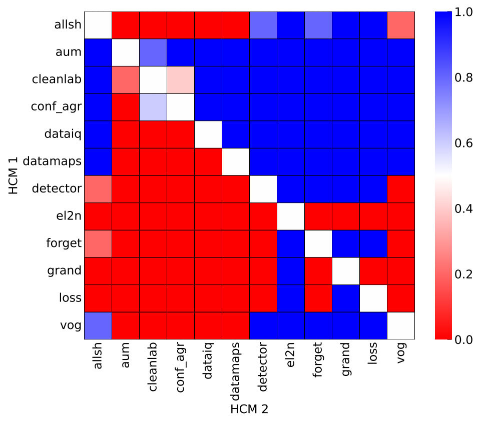
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet

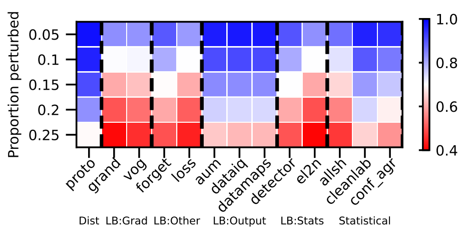


(d) CIFAR - ResNet

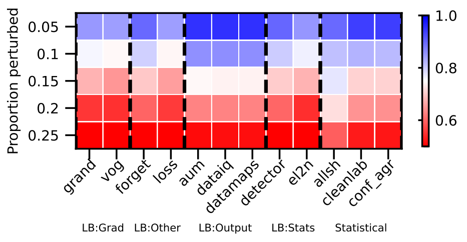
Figure 70: Atypical: Crop Shift. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

D.5.9 ATYPICAL: ZOOM SHIFT

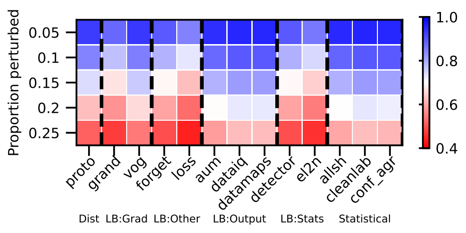
Heatmaps.



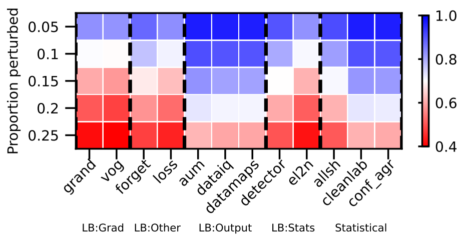
(a) MNIST - LeNet



(b) MNIST - ResNet

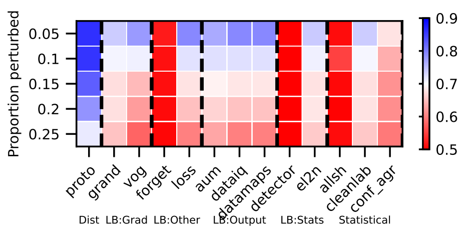


(c) CIFAR - LeNet

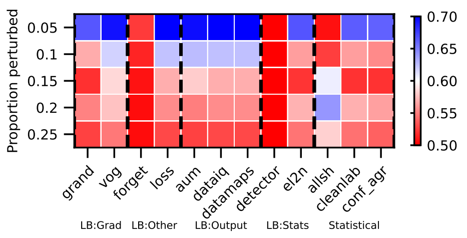


(d) CIFAR - ResNet

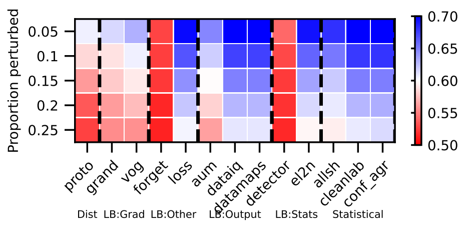
Figure 71: Atypical: Zoom Shift - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.



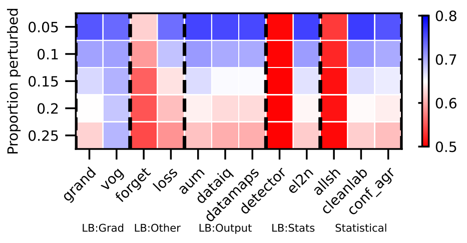
(a) MNIST - LeNet



(b) MNIST - ResNet



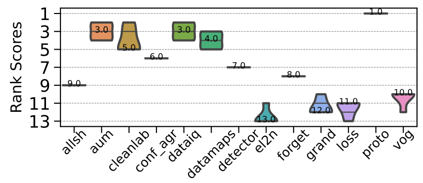
(c) CIFAR - LeNet

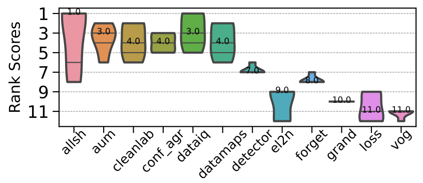


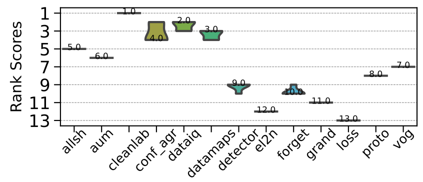
(d) CIFAR - ResNet

Figure 72: Atypical: Zoom Shift - AUROC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.


(a) MNIST - LeNet


(b) MNIST - ResNet


(c) CIFAR - LeNet

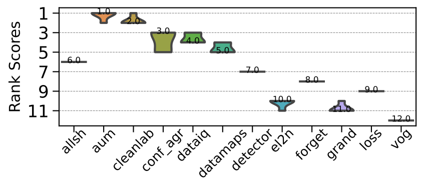
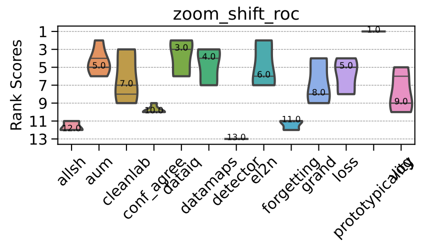
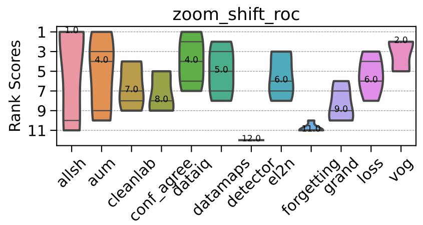

(d) CIFAR - ResNet

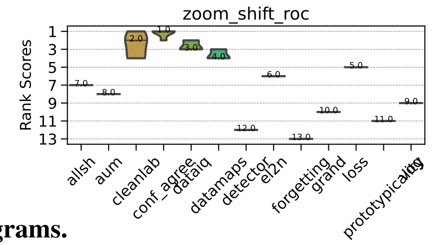
Figure 73: Atypical: Zoom Shift - AUPRC. Ranking of HCMs.

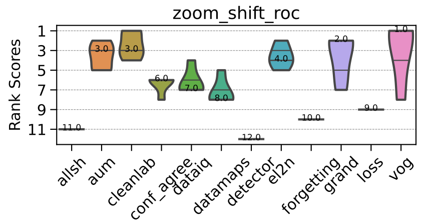


(a) MNIST - LeNet

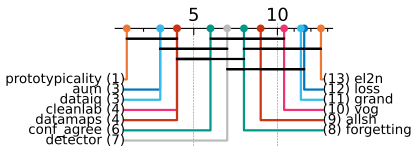


(b) MNIST - ResNet

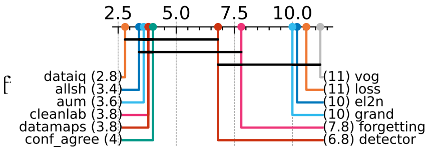




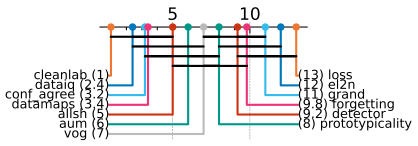
CD diagrams.



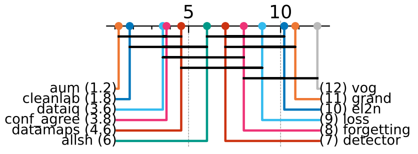
(a) MNIST - LeNet



(b) MNIST - ResNet

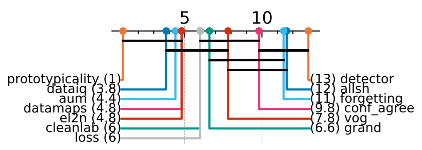


(c) CIFAR - LeNet

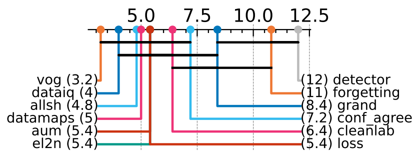


(d) CIFAR - ResNet

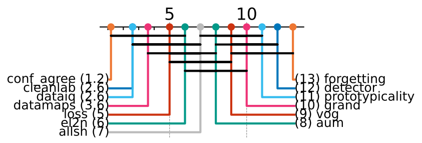
Figure 75: Atypical: Zoom Shift - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.



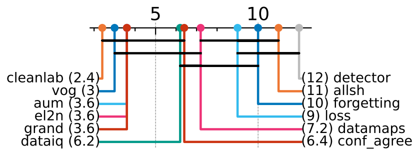
(a) MNIST - LeNet



(b) MNIST - ResNet



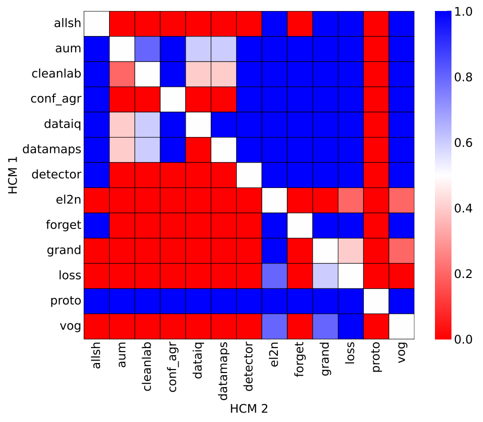
(c) CIFAR - LeNet



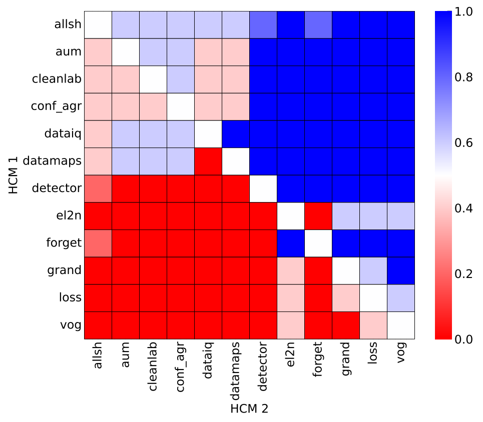
(d) CIFAR - ResNet

Figure 76: Atypical: Zoom Shift - AUROC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

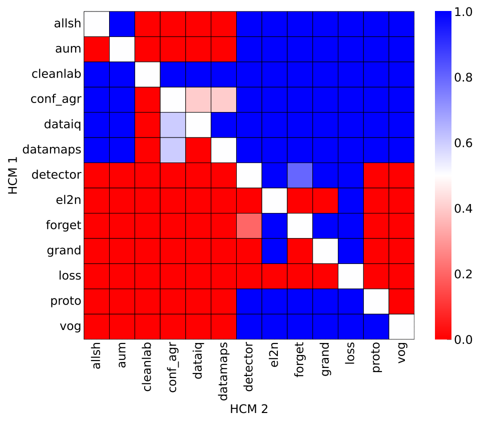
Matrix compare



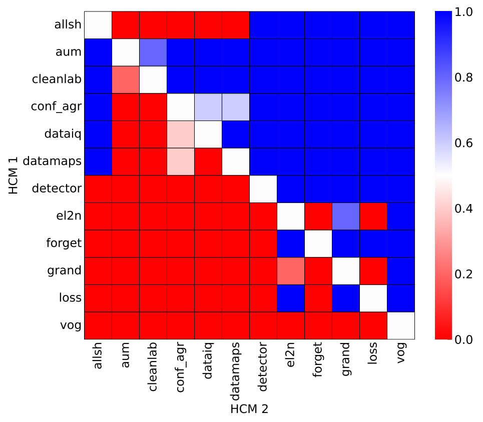
(a) MNIST - LeNet



(b) MNIST - ResNet



(c) CIFAR - LeNet



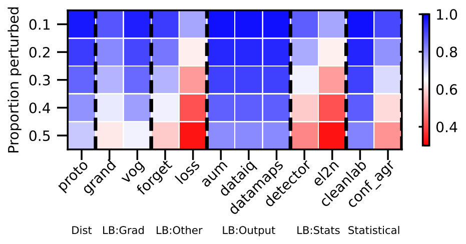
(d) CIFAR - ResNet

Figure 77: Atypical: Zoom Shift. Head-to-head comparison matrix assessing for runs when a certain HCM outperforms another. i.e. when y-axis (HCM 1) beats x-axis competitor (HCM 2).

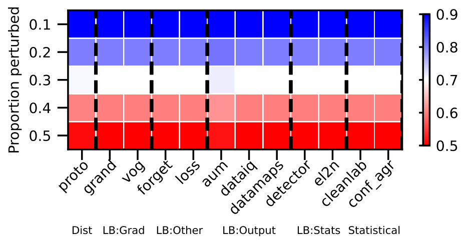
D.6 Individual results - Tabular

D.6.1 Mislabeling: Uniform

Heatmaps.



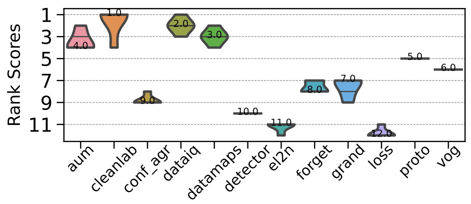
(a) Cover



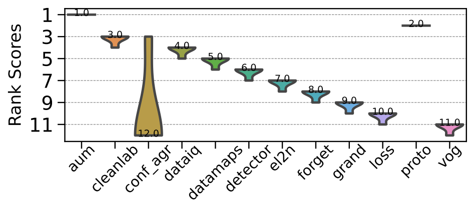
(b) Diabetes

Figure 78: TABULAR: Uniform mislabeling - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.



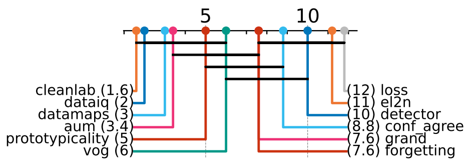
(a) Cover



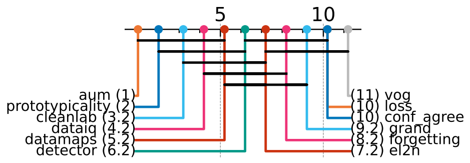
(b) Diabetes

Figure 79: TABULAR: Uniform mislabeling - AUPRC. Ranking of HCMs.

CD diagrams.



(a) Cover

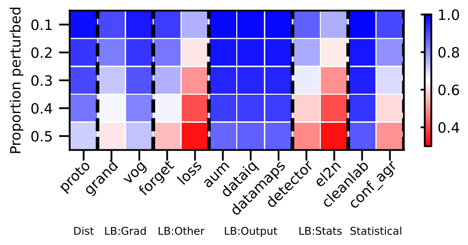


(b) Diabetes

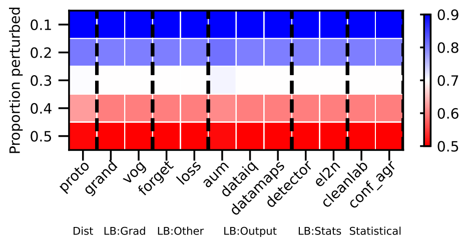
Figure 80: TABULAR: Uniform mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

D.6.2 Mislabeling: Asymmetric

Heatmaps.



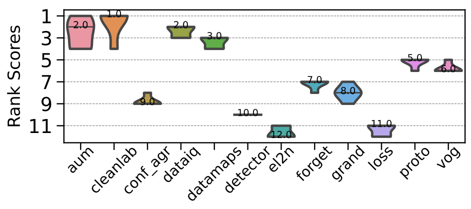
(a) Cover



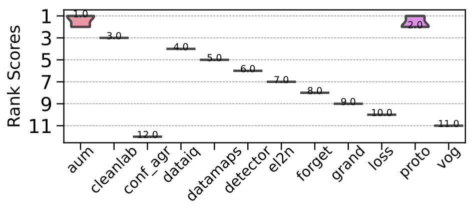
(b) Diabetes

Figure 81: TABULAR: asymmetric mislabeling - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.



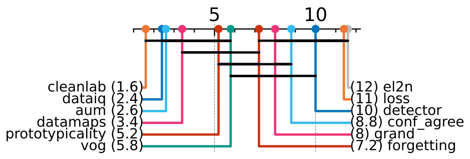
(a) Cover



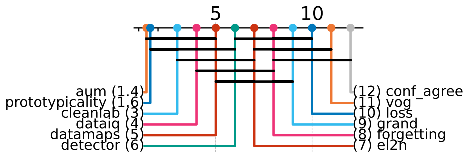
(b) Diabetes

Figure 82: TABULAR: asymmetric mislabeling - AUPRC. Ranking of HCMs.

CD diagrams.



(a) Cover

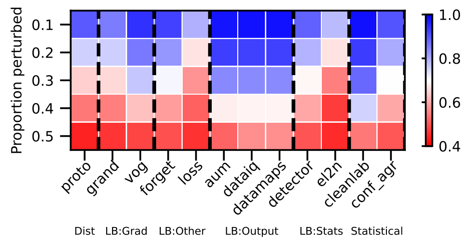


(b) Diabetes

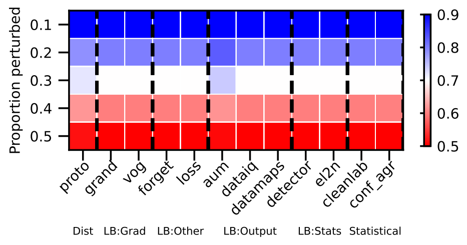
Figure 83: TABULAR: asymmetric mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

D.6.3 Mislabeling: Instance

Heatmaps.



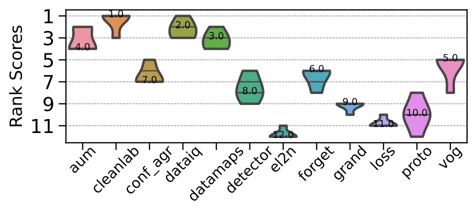
(a) Cover



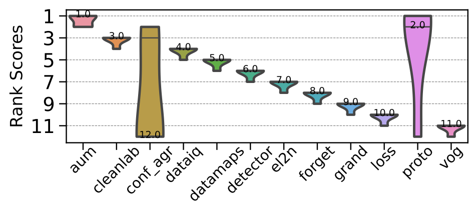
(b) Diabetes

Figure 84: TABULAR: instance mislabeling - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.



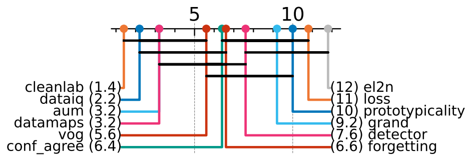
(a) Cover



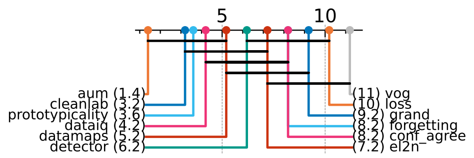
(b) Diabetes

Figure 85: TABULAR: instance mislabeling - AUPRC. Ranking of HCMs.

CD diagrams.



(a) Cover

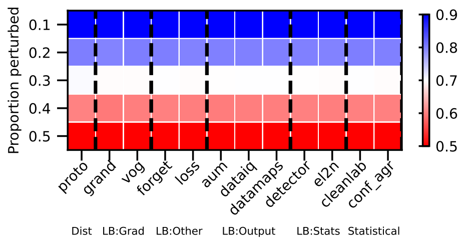


(b) Diabetes

Figure 86: TABULAR: instance mislabeling - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

D.6.4 NEAR OOD: COVARIATE

Heatmaps.


(a) Cover

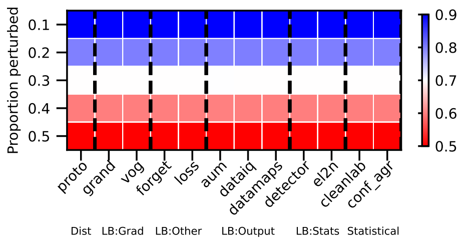
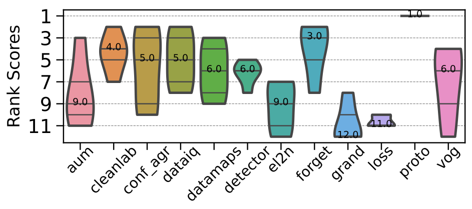

(b) Diabetes

Figure 87: TABULAR: Near OOD - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.


(a) Cover

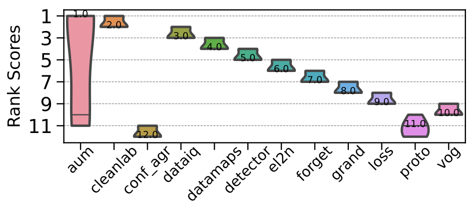
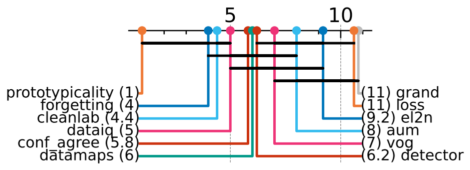

(b) Diabetes

Figure 88: TABULAR: Near OOD- AUPRC. Ranking of HCMs.

CD diagrams.


(a) Cover

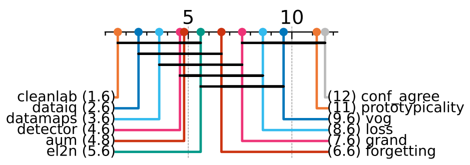
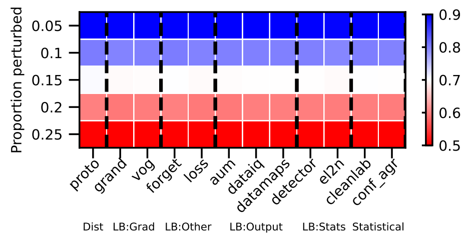

(b) Diabetes

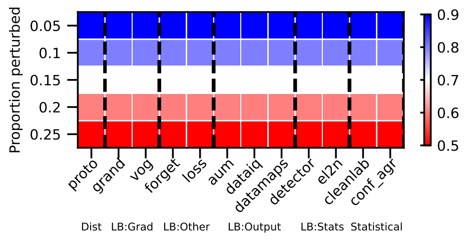
Figure 89: TABULAR: Near OOD - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

D.6.5 Far OOD

Heatmaps.



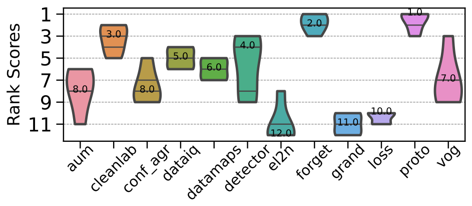
(a) Cover



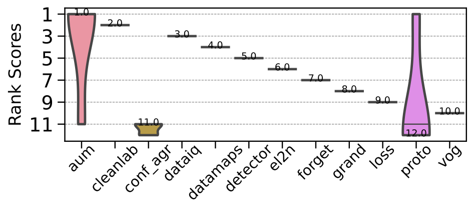
(b) Diabetes

Figure 90: TABULAR: far ood - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.



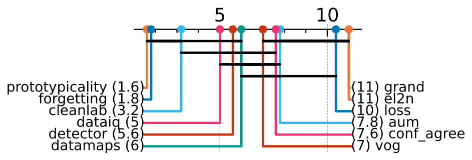
(a) Cover



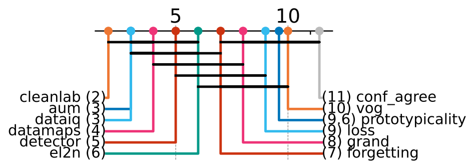
(b) Diabetes

Figure 91: TABULAR: far ood - AUPRC. Ranking of HCMs.

CD diagrams.



(a) Cover

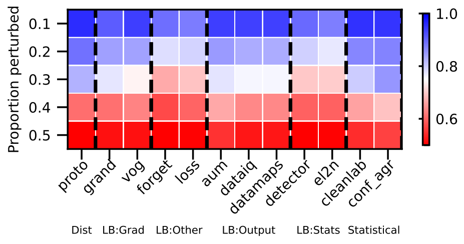


(b) Diabetes

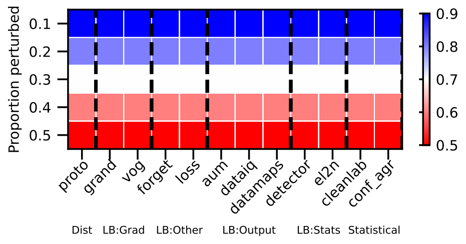
Figure 92: TABULAR: far ood - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

D.6.6 ATYPICAL

Heatmaps.



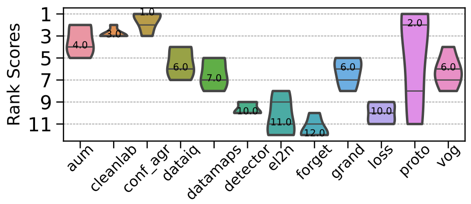
(a) Cover



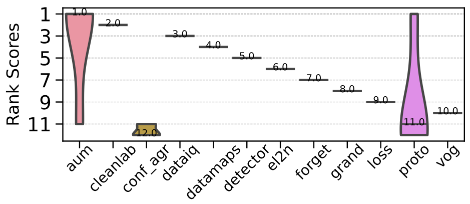
(b) Diabetes

Figure 93: TABULAR: atypical - AUPRC. We vary the proportion perturbed. i.e. proportion of hard examples. Blue is better, red worse.

Rankings.



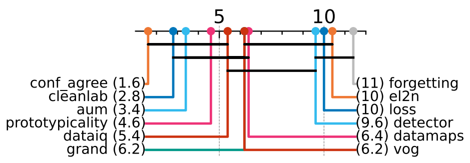
(a) Cover



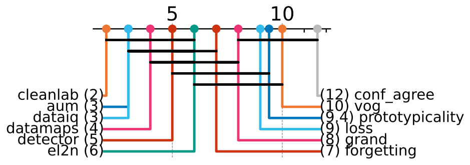
(b) Diabetes

Figure 94: TABULAR: atypical - AUPRC. Ranking of HCMs.

CD diagrams.



(a) Cover



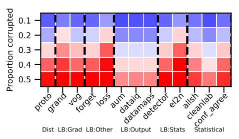
(b) Diabetes

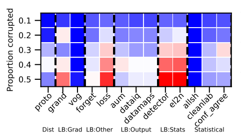
Figure 95: TABULAR: atypical - AUPRC. Critical difference diagrams. Black lines connect methods that are not statistically significantly different.

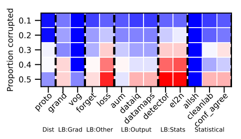
D.7 Additional results - Medical X-Ray data

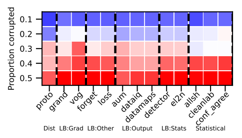
Overview. Our main experiments are conducted on the MNIST and CIFAR-10 datasets. The reason is that it is important to have "clean" datasets for the purposes of ensuring a good simulation environment. e.g. we do not want the original data to already have mislabeling. We now extend our analysis of HCMs to an additional medical imaging dataset from the NIH (Majkowska et al., 2020). These contain chest x-rays which have been audited by multiple expert radiologists to ensure they are "clean". We repeat our experiments from the main paper and show the results below.

We now analyze the results and show the findings about HCMs are retained for the X-Ray dataset.


(a) Uniform


(b) Near-OoD (Covariate)


(c) Far-OoD


(d) Atypical (Zoom)

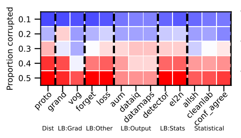
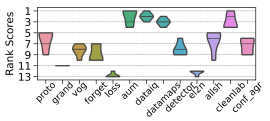
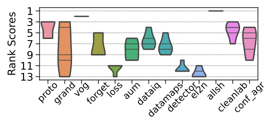
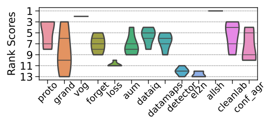
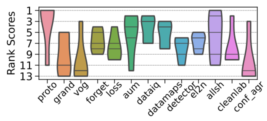

(e) Atypical (Crop)

Figure 96: AUPRC on medical X-Ray for different HCMs for different hardness types aggregated across setups. We vary the proportion perturbed, i.e. the proportion of hard examples. Blue is better, red worse. We see variability of HCM capabilities across hardness types and proportions.


(a) Uniform


(b) Near-OoD (Covariate)


(c) Far-OoD


(d) Atypical (Zoom)

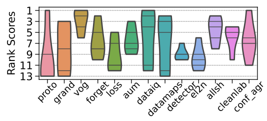
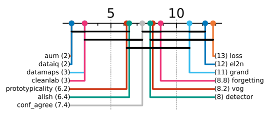
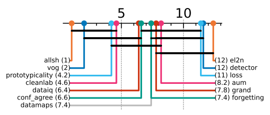

(e) Atypical (Crop)

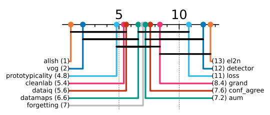
Figure 97: Performance rankings of HCMs vary depending on the hardness type.



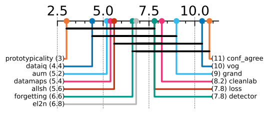
(a) Uniform



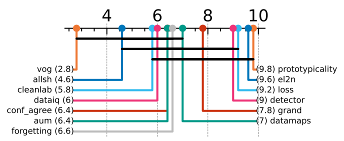
(b) Near-OoD (Covariate)



(c) Far-OoD



(d) Atypical (Zoom)



(e) Atypical (Crop)

Figure 98: Critical difference diagrams highlight that similar categories/classes of HCMs *do not* have a statistically significant difference in their performance, indicated by the horizontal black lines linking HCMs which are not statistically different. The numbers in brackets denote mean rank.

We highlight the following main takeaways which are maintained from the main paper experiments, showing the findings indeed are similar across datasets:

- **Hardness types vary in difficulty.** Similarly to the main text, we find that different types of hardness are easier or harder to characterize. For instance, uniform mislabeling or Far-OoD are much easier than data-specific hardness like Atypical. Hence, we still see the performance differences.
- **Learning dynamics-based methods with respect to output confidence are effective general-purpose HCMs.** In selecting a general-purpose HCM, we find similar to the main text that HCMs that characterize samples using learning dynamics on the confidence — uncertainty-based methods, which use probabilities (DataMaps, Data-IQ) or logits (AUM), are the best performing in terms of AUPRC across the board.
- **HCMs typically used for computational efficiency are surprisingly uncompetitive.** Similarly to the main text, we find that HCMs that leverage gradient changes (e.g. GraNd), typically used for selection to improve computational efficiency, fare well at low p. However, at higher p, they become notably less competitive compared to simpler and computationally cheaper methods.
- **HCMs should only be used when hardness proportions are low.** In general, different HCMs have significantly reduced performance at higher proportions of hardness. This is expected as we get closer to 0.5 since it's harder to identify a clear difference between samples.
- **Individual HCMs within a broad "class" of methods are NOT statistically different.** We find from the critical difference diagrams that methods falling into the same class of characterization are not statistically significant from one another (based on the black connected lines), despite the minor performance differences between them. Hence, practitioners should select an HCM within the broad HCM class most suitable for the application.
- **Selecting an HCM based on the hardness is useful.** We find that confidence is a good general-purpose tool if one doesn't know the hardness. However, if one knows the hardness, one can better select the HCM.

Of course, there are subtle differences given the difference in datasets. As an example, the performance rankings exhibit higher variability. That said, this is a function of running fewer seeds compared to the main paper. The most important compared to raw ranking is the CD Diagram which provides an analysis of the statistical significance of the performance differences which the raw scores don't provide. In that case, despite the rankings being different, we see very many of the same HCMs from the main paper are similarly connected by black lines, indicating the performance differences are not statistically different.

D.8 INSIGHTS: EFFECT OF SEVERITY OF PERTURBATION ON HARDNESS SCORES

Overview. We ask what is the effect of the severity of the perturbation on the hardness score. Specifically, are samples with greater perturbations considered "harder" than others by HCMs based on their scores.

We conducted an experiment for zoom shift, crop shift, and Near-OOD (covariate), where we have two severities of perturbation namely small and large. Specifically, for zoom (2x vs. 10x), crop shift (5 pixels vs. 20 pixels), and Near-OOD (std deviation of 0.5 vs. 2).

We report the results in Fig 99. The results show the percentage changes in hardness scores (in the direction of hardness, e.g. increase or decrease) for different HCMs for the different hardness types considered. In all cases, we see an increase in hardness score for increased severity, with some HCM scores showcasing greater sensitivity and changes than others. In particular, we see the gradient and loss-based approaches have the greatest changes as compared to the methods computing scores based on probabilities.

These results shed interesting insights into the sensitivities of different HCMS.

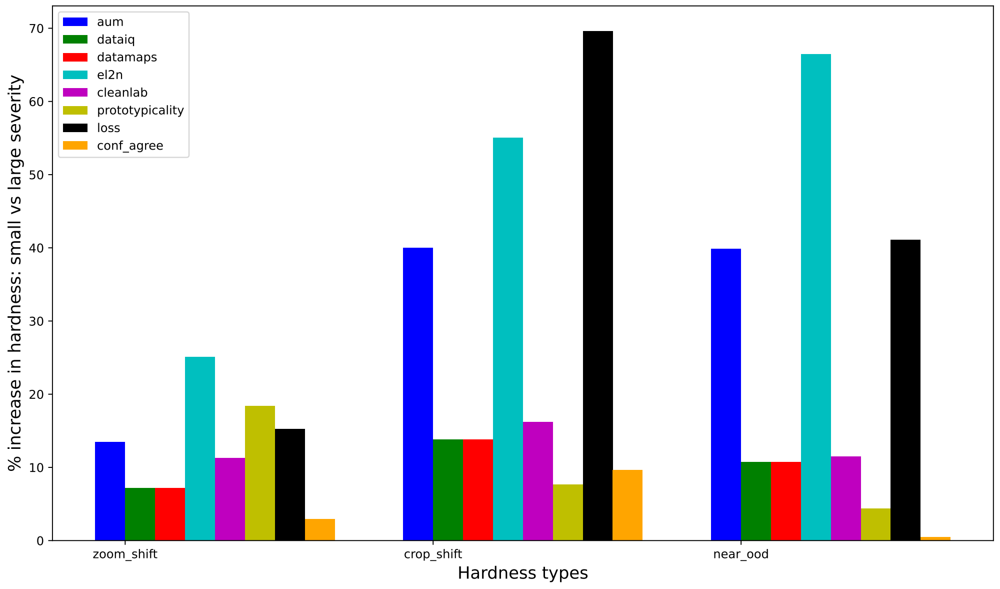


Figure 99: Effect on hardness scores with respect to perturbation severity from small to large.