IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 21 March 2024, accepted 17 April 2024, date of publication 24 April 2024, date of current version 2 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3393296

RESEARCH ARTICLE

A Robust Framework for Distributional Shift Detection Under Sample-Bias

BIRK TORPMANN-HAGEN[1], MICHAEL A. RIEGLER[2], PÅL HALVORSEN[2], AND DAG JOHANSEN[1]
[1]Institute for Computer Science, UiT: The Arctic University of Tromsø, 9019 Tromsø, Norway
[2]Department of Holistic Systems, SimulaMet, 0167 Oslo, Norway

Corresponding author: Birk Torpmann-Hagen (birk.s.torpmann-hagen@uit.no)

ABSTRACT Deep Neural Networks have been shown to perform poorly or even fail altogether when deployed in real-world settings, despite exhibiting excellent performance on initial benchmarks. This typically occurs due to relative changes in the nature of the production data, often referred to as distributional shifts. In an attempt to increase the transparency, trustworthiness, and overall utility of deep learning systems, a growing body of work has been dedicated to developing distributional shift detectors. As part of our work, we investigate distributional shift detectors that utilize statistical tests of neural network-based representations of data. We show that these methods are prone to fail under sample-bias, which we argue is unavoidable in most practical machine learning systems. To mitigate this, we implement a novel distributional shift detection framework which explicitly accounts for sample-bias via a simple sample-selection procedure. In particular, we show that the effect of sample-bias can be significantly reduced by performing statistical tests against the most similar data in the training set, as opposed to the training set as a whole. We find that this improves the stability and accuracy of a variety of distributional shift detection methods on both covariate- and semantic-shifts, with improvements to balanced accuracy typically ranging between 0.1 and 0.2, and false-positive-rates often being eliminated altogether under bias.

INDEX TERMS Deep learning, system support, neural networks, distributional shift, OOD detection, sample bias, statistical tests.

I. INTRODUCTION

Although Deep Neural Networks (DNNs) have been shown to exhibit excellent performance in a wide variety of tasks and in a multitude of domains, recent research has shown that this performance decreases rapidly if the model encounters data that is not identically distributed to the training data [1], [2], [3], [4], [5]. For example, deep image segmentation models trained to segment polyps in the gastrointestinal tract have been shown not to generalize to other hospitals or lighting conditions than those observed in training data [6]. Similar results have also been shown in skin-lesion classification [1] and image-captioning models [3]. Even models trained on large, general-purpose datasets such as ImageNet have been shown to fail to generalize to ImageNet-like datasets [7]. This is referred to as *generalization failure*, and has been shown to be ubiquitous in predictive deep learning [1], [8]. Data augmentation [9], multi-domain training [4], [10], ensemble-models [11], [12] and consistency-based methods [13], [14], [15] appear to show some promise toward increasing generalizability, however, no existing methods endow the models with sufficient generalizability for reliable performance in practical conditions. There has also been considerable research to address these shortcomings through alternative training paradigms [4], but meta-analyses have shown that these methods do not outperform conventional training by a significant margin [16].

The apparent inability of DNNs to generalize severely diminishes their potential utility, as distributional shifts cannot realistically be prevented in most practical settings. This is further complicated by the opaque nature of the predictive process of DNNs. It is difficult to ascertain whether the model is failing to generalize, and though implementing explanations can assist somewhat in this regard [17], many

The associate editor coordinating the review of this manuscript and approving it for publication was Diego Bellan.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

domains require a higher degree of transparency in the model's performance than these methods can provide [18]. This has led to research towards Distributional Shift Detectors (DSDs), which detects when the incoming data differs from the data with which the predictive model has been developed.

This paper considers a common approach to distributional shift detection based on performing statistical tests on batches of neural-network-based representations of the data. We show empirically that these methods readily yield false positives under sample-bias. Although sample-bias can be considered a form of distributional shift, we argue that these shifts are not important with respect to the fundamental goal of DSDs; namely, determining whether the model can perform well on the incoming data or if the data should be deferred for further processing. For example, it is clear that a model trained to detect objects of interest in an autonomous vehicle should not be adversely affected by a decreased rate of object occurrence when compared to a random sampling of the training set. Though this can be considered a distributional-shift – in particular, *label shift* – the model has likely been trained on data that does not contain cars, pedestrians, or other objects, and thus can be expected to perform well so long as a more malignant shift has not occurred simultaneously. Similarly, the aforementioned model is not likely to be affected by sample-bias in time, subsequent frames from on-board cameras. Indeed, in many deep-learning systems this is considered regular operation. As our work will show, such biases would nevertheless cause distributional shift detectors to predict positively and thus raise alerts on data on which the system would otherwise perform well.

We address this by augmenting the distributional shift detection framework with a sample selection component, such that the statistical tests are performed with respect to the samples' nearest neighbours in the training population with respect to their latent representations. We compare this framework, k-Nearest Neighbour Distributional Shift Detection (kNNDSD), to our baselines and show that our framework yields superior performance. kNNDSD exhibits reduced false-positive rates, improved accuracy, and improved separability of Out of Distribution (OOD) and In-Distribution (InD) samples under sample-bias, while maintaining equivalent performance without sample-bias.

Overall, we summarize our contributions as follows:

- We demonstrate experimentally that sample-based DSDs fail under sampling- and label-bias given this perspective.
- We propose the addition of a sample-selection procedure to sample-based DSDs, which involves performing statistical tests with respect to the samples' nearest neighbours in the embedding space.
- We show that this sample-selection procedure reduces the effect of sampling-bias and label-bias.
- We perform a study of the impact of our method on real-world datasets with organic distributional shifts and sampling-biases.

All code and raw data used in this paper is available via https://github.com/BirkTorpmannHagen/kNNDSD.

II. RELATED WORK

In this section, we motivate our work through contextualizing the need for DSDs as a means of addressing generalization failure. We then discuss related work on DSDs and uncertainty estimation.

A. GENERALIZATION FAILURE

The performance of Deep Learning (DL) systems can only be guaranteed under the independently and identically distributed (IID) assumption [5]. In practical scenarios, however, such assumptions rarely hold. Reference [1] explore the effect this has on various DL systems across multiple domains, and argue that overparametrization with respect to the data is a primary contributing factor in generalization-failure. They show that identically trained models can exhibit very different levels of generalization on novel data, dependent only on their random initialization. Reference [3] consider generalization failure from a perspective of the types of features learned, and show through multiple case-studies that DNNs readily learn dataset-specific shortcuts rather than generalizable features.

Reference [6] review the results of the EndoCV2021 competition [19], which is centered around generalizability in the polyp-segmentation domain. The best-performing approaches utilized ensemble-models [11], [12], but still exhibited considerable performance drops when deployed on data collected from different hospitals or with different lighting conditions.

Reference [4] perform a survey on methods for generalizable learning, summarizing the key challenges and approaches in the field. Reference [16] show that few of these methods outperform conventional approaches outside of the datasets upon which they were originally implemented, demonstrating the difficulty involved in improving the generalizability of DL models.

Reference [20] finds that CNNs are sensitive to noise and common perturbations. Another study showed that Google's vision API is highly sensitive to noise [21]. Reference [22] show that models trained on ImageNet are biased towards textural features, and that imposing shape-biases improves generalizability. Reference [7] investigate the generalizability of DNNs trained on ImageNet, and observe that these models do not generalize well to other ImageNet-like datasets. Similar behaviour has also been found in models trained on the CIFAR10 dataset [23].

Evidently, attaining sufficient levels of generalization is a highly challanging and open problem in the field. Thus, practical deployment of DL systems requires either that variables that can influence the nature of the data are carefully controlled, or system support which can raise alerts when incoming data is likely to induce generalization failure. While the former may be attainable in certain domains, the latter is a more general solution, and one of the motivators for recent work on distributional shift detection.

B. DISTRIBUTIONAL SHIFT DETECTION

Fundamentally, the detection of distributional-shift involves determining whether or not some data is identically distributed to a known dataset. Reference [24] provide a unifying perspective on distributional shift, and argue that it can generally be categorized according to the affected variables. This includes:

- *covariate shift*, which occurs when there is a change in the distribution of any of the input variables, i.e., $P(x)$,
- *prior probability shift*, which occurs when there is a change in the class-probabilities, i.e., $P(y)$,
- and *Semantic Shift* which occurs when the underlying relationship of the neural network is intended to represent changes, i.e., a change in $P(y|x)$

Many DSDs do not explicitly differentiate between these shifts. Pragmatically, however, different shifts affect the network in different ways, which is worth noting in a practical implementation of a DSD.

Reference [25] present a survey of methods of distributional shift detection, organizing methods according to a taxonomy including anomaly detection, open-set recognition, outlier detection, novelty detection, and OOD detection. They highlight that a majority of distributional shift detection methods revolve around detecting concept shift, that the evaluation of DSDs is often lacking, and that existing methods often neglect the more practical dimensions of the problem.

Reference [26] perform an empirical study of sample-based methods of detecting various distributional shifts. They test various dimensionality-reducing techniques and statistical tests, observing that label-classifiers with univariate testing exhibit the best performance. Reference [27] employ a similar method, but instead implement a measure of the typicality of a given population of samples, noting that the majority of samples in high-dimensional space is distributed around a region referred to as the typical set. Whereas the methods studied in [26] perform tests on the features themselves, the test utilized in [27] utilize likelihood estimates as produced by generative models. The DSDs outlined in [26] and [27] are characterized by the utilization of multiple samples, thus comparing the target distribution with the source distribution somewhat directly.

Other works often consider approaches that do not require multiple samples, and instead only rely on various properties that can be observed in the inference of OOD samples. Reference [28] implement a DSD based on the observation that InD samples tend to exhibit higher probability scores than OOD samples. Reference [29] investigate the use of gradients towards distributional shift detection for classification, and find that the gradient norm can be used to detect OOD samples. Reference [30] builds upon the general idea of [28] and implements a shift-detector that exploits the discrepancy in softmax scores between InD and OOD data after gradient-based perturbation [30], which they note delineates InD and OOD data more efficiently than just considering the softmax scores. These methods can, however, also be implemented in a multiple-sample configuration. This typically yields improved results, as the separability of OOD and InD samples will increase as a result of the sampling process and regression towards the mean.

Few of these works effectively distinguish between shifts that affect the performance of model – therein sample bias – and those that do not, however. Though sample-bias indeed does constitute a distributional shift, and thus is often correctly detected as such by conventional DSDs, we adopt a more pragmatic view. The fundamental purpose of DSDs are to serve as monitors in DL systems, and should thus only yield alerts when the shift is liable to decrease the overall system performance. This means that shifts that do not affect performance generally should not be detected. There are, however, some caveats to this. In systems that utilize active learning, for instance, sample bias may indeed adversely affect system performance in much the same way as neglecting to shuffle a dataset prior to gradient descent [31]. Systems with time-dependent networks such as Recursive Neural Networks or Transformers may also be affected by sample bias, in particular if the bias is dependent on time. Suppose for instance if such a system receives a sequence of identical data due to a sensor malfunction. This is likely to induce some form of performance degradation, if not in terms of correctness with respect to the data but in terms of overall system performance, and should thus be detected. Therefore, the practical implementation of DSDs must be done with respect to a clearly defined set of system requirements, including a delineation of what shifts should and should not be detected. For the vision-systems we study in our work, we identify sample-bias, label-bias, and temporal-bias as benign shifts.

III. K-NEAREST NEIGHBOUR DISTRIBUTIONAL SHIFT DETECTION

Pragmatically, the role of a DSD is to alert the system when the Machine Learning (ML) system encounters data that it cannot accurately predict. As the performance of a DNNs can only be guaranteed under the assumption that the incoming data is identically distributed to the training data, a simple abstraction of this task is to detect when the incoming data is *not* identically distributed to the training data, hence the need for DSDs.

Formally, this can be expressed as determining the equivalence of the training- and test-distributions: $P_{train}(x, y) = P_{test}(x, y)$. As these distributions cannot realistically be expressed in closed form, they must be approximated. DSDs can thus be broadly decomposed into a component that approximates the distributions, and a component which compares their similarity. In Reference [26], for example, this involved the extraction of features from input data as an approximation method and various statistical tests as similarity measures. In Reference [27], the distributions were approximated using generative models, and a statistical test based on the notion of typical sets was used as the similarity measure. Single-sample methods, such as ODIN [30], GradNorm [29] and KL-divergence [28], do not explicitly approximate the test distribution, and instead simply rely on the expressiveness of their respective similarity-measures. This can, however, be interpreted as an $N = 1$ sampling of $P_{test}(x, y)$.

Though there is a large body of work focusing on developing both novel similarity measures and approximation methods [25], few methods account for the fact that a rigorous estimation of either distribution requires the sampling of data, nor do they consider the nature of the shift and its effect on the neural network. For example, it is not given that there is sufficient data to adequately sample $P_{test}(x, y)$, or the shift may not affect the performance of the neural network at all, and as a result not warrant deference.

To illustrate, assume that $P_{train}(x, y) = P_{test}(x, y)$. Suppose that we introduce a binary variable s that influences whether or not a datum is included in the sampling ($s = 1$) or not included ($s = 0$) – i.e., a *sample bias*. The test distribution is then given by $P_{test}(x, y|s)$. If s is independent of x, this simply corresponds to randomly sampling the distribution. Ifs is dependent on x, however, a covariate shift will occur [24]. However, recall that since the sample distribution is still drawn from $P_{test}(x, y)$, the samples are by definition InD to ($P_{train}(x, y)$. The changes in covariates that the shift induced by the biased sampling are still wholly represented in the training set. As a result, the performance of the predictive model will be unaffected, and thus an ideal DSDs should not consider this type of shift from a purely pragmatic perspective. However, most of the existing implementations of DSDs predict positively these benign changes. Since sample-bias is practically unavoidable in deployment scenarios, this severely reduces the potential utility of DSDs.

Many systems, for instance autonomous vehicles, medical imaging systems, and surveillance systems, operate on streams of sensor data. For a system which implements a sample-based DSD, all incoming data from these streams would be considered as having undergone a distributional-shift, as the data will not be sampled uniformly from the overall distribution. Instead, it is sampled sequentially in time and will thus exhibit a bias for covariates that depend on time. In the case of an autonomous vehicle, for instance, test batches would necessarily contain biases for the weather conditions, the time of day, the type of traffic, the type of road, traffic-obstacles, etc. For medical applications, there would be a bias for patient demographics, imaging equipment, or preparation routines. In all these cases, the model will generally perform well as long as the data is included in the distribution of the training data. Fig. 1 shows a PCA projection of the effect of a few of the types of sample-bias studied in our work on the latent representation of the data for three datasets. In these plots, all of the test data is InD and is clearly drawn from a region that is represented in the dataset, but as the test data is not sampled uniformly from the *whole* dataset, statistical tests yield low p-values. Sample-based DSDs would as a result classify this data as OOD.

To address these shortcomings, we formulate an extension to the framework described in [26]. In addition to extracting OOD-ness measures from a neural network and performing statistical-testing, we also apply a sampling procedure such that the statistical tests are computed only with respect to the most similar samples in the training set, rather than the training set as a whole. In particular, for each test-datum, we select the k closest training samples that are the closest in terms of l_2 distance in the latent-space of a suitably low-dimension layer in the representation model, e.g. the penultimate layer of classifiers or the middle layer of encoder-decoder architectures. Statistical tests are then computed between the OOD-ness measures of these k-nearest training samples and the test-samples, yielding de-biased p-values. We refer to this framework as k-Nearest Neighbour Distributional Shift Detection (kNNDSD). We summarize this process in Algorithm 1.

Algorithm 1 The kNNDSD Algorithm. $f(\cdot)$ Denotes the Representation Model Used to Compute Features and Encodings, for Instance, a Classifier or a Generative Model. X_{train} Represents the Training Data, and X_{test} Represents a Batch of Test Samples

```
Require: $f(\cdot)$, $X_{train}$, $X_{test}$, k
Ensure: $p\_value$
  $\mathbf{z}_{train} \leftarrow$ compute_encodings($f(\cdot), X_{train}$)
  $\mathbf{z}_{test} \leftarrow$ compute_encodings($f(\cdot), X_{test}$)
  $\mathbf{y}_{train} \leftarrow$ compute_features($f(\cdot), X_{train}$)
  $\mathbf{y}_{test} \leftarrow$ compute_features($f(\cdot), X_{test}$)
  debiased_idxs $\leftarrow$ []
  for $i$ = 0 to $len(\mathbf{z}_{test}) - 1$ do
    dists $\leftarrow$ [$(\mathbf{z}_{test}[i] - \mathbf{z}_{train})^2$ for $i$ in range($len(\mathbf{z}_{test})$)]
    kn_idxs$\leftarrow$ argpartition($dists, k$)
    debiased_idxs = concat(debiased_idxs, kn_idxs)
  end for
  p_value $\leftarrow$ ks_test($\mathbf{y}_{train}$[kn_indices], $\mathbf{y}_{test}$)
  return p_value
```

Our method can be understood in a more general sense through the lens of manifold learning [32], [33], [34]. The union of manifolds hypothesis describes how natural data generally lie along a low-dimensional manifold within the high-dimensional data. One can smoothly interpolate between data within this manifold, as each dimension in the space corresponds to some intrinsic property to the data. The latent-space we use to extract neighbours can be interpreted as a representation of this manifold, with semantically similar datum lying in contiguous regions to one another. The nearest neighbours of a datum thus represent the most semantically similar data in the training distribution in terms of the learnt features, that is, the features with respect to which the biased sampling was performed. Selecting nearest neighbours

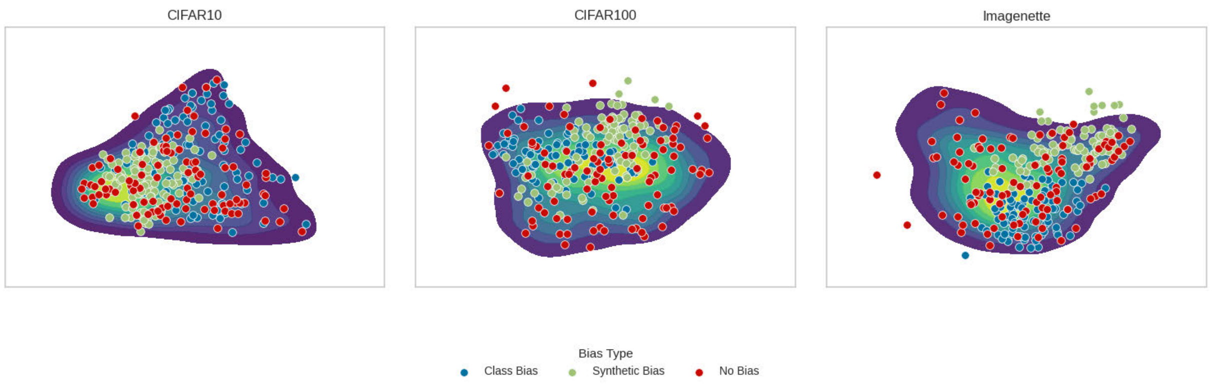


FIGURE 1. PCA visualisation of a kernel-density-estimation of the validation datasets for three of the datasets tested in our work alongside batches of size N=100 sampled with differing bias types. Observe how biased data, though drawn from the same distribution, appear to be distributed differently.

therefore explicitly de-biases the data with respect to the manifold such that any remaining difference between the samples can be attributed to the data being off-manifold and thus being OOD. However, this assumes that the manifold is sufficiently well represented by the training data, which, as will be discussed in Section V, may not always hold. Moreover, this method is also contingent on the assumption that a given neural network has learnt a semantically coherent manifold, which may not hold for all types of architecture or training methods.

We further extend this framework so that more general DSD-oriented distance metrics can be utilized, such as Grad-Norm [29], ODIN [30], or KL-Divergence [28]. Although these works evaluate their methods on a samples size of $N = 1$, they can just as easily be incorporated into the sample-based frameworks by treating the distance metrics as features for the statistical tests. That is, whereas [26] and [27] both perform statistical tests on the encodings themselves, in our framework, we also implement statistical tests on gradient-based or otherwise derived OOD-ness measures. Finally, while the framework described in [26] selects thresholds according to a set confidence interval (with Bonferroni correction), we opt instead to compute thresholds according to the minimum value found in the validation dataset. The overall framework is summarized in Fig. 2.

IV. EXPERIMENTS

In our experiments, we evaluated the ability of several popular DSDs to classify distributional shifts under different forms of sample-bias. We evaluated these methods under both covariate shifts and semantic shifts. We observed that all distributional shift detectors exhibited significantly reduced performance under sample-bias. Implementing our proposed sample-selection step prior to tests was shown to yield superior performance under sample-bias and equivalent performance without sample-bias.

A. METHODOLOGY

We divide our experiments into two sections: In the first part, we analyze the effect of our methods on the detection of covariate shifts under sample-bias. We utilize both common benchmarking datasets and datasets containing real-world instances of distributional shift. To this end, we implement sample-biases for labels, data order, and a synthetic sample-bias. In the second section we analyze the effect of our methods on the detection of semantic shifts under bias. We evaluate several popular implementations of DSDs.

1) DATASETS

We evaluated DSDs on both real-world and benchmarking datasets. In particular, we used the following datasets for our analysis of covariate shifts:

- Benchmarking Datasets with additive noise
  - **CIFAR-10** and **CIFAR100** [35], two benchmarking datasets.
  - **Imagenette** [36] - a small version of ImageNet.
- Datasets with organic distributional shifts
  - **NjordVid** [37]: a fishing trawler video analytics dataset, containing videos with various lighting- and weather-conditions. The dataset contains both temporal annotations and bounding boxes for four classes, but we only consider it in the context of object-detection.
  - **NICO++** [38], a domain-adaptation classification benchmark organized by contexts and classes. The dataset contains 60 classes, depicted in ten common contexts. We treat each context as a distributional shift.
  - **Polyp-Datasets**: **CVC-ClinicDB** [39], **Kvasir-SEG** [40] and **Etis-LaribDB** [41]: three polyp segmentation datasets, collected from different hospitals, with the first two being used as InD datasets and Etis-LaribDB as the OOD dataset.

We constructed three folds for each dataset: a training fold, which was used as a training set for the predictive models and served as the reference population for the DSDs; a validation fold, which was used as validation data for the predictive models and InD data for the DSDs; and finally, an OOD fold, which was curated such that the models we trained

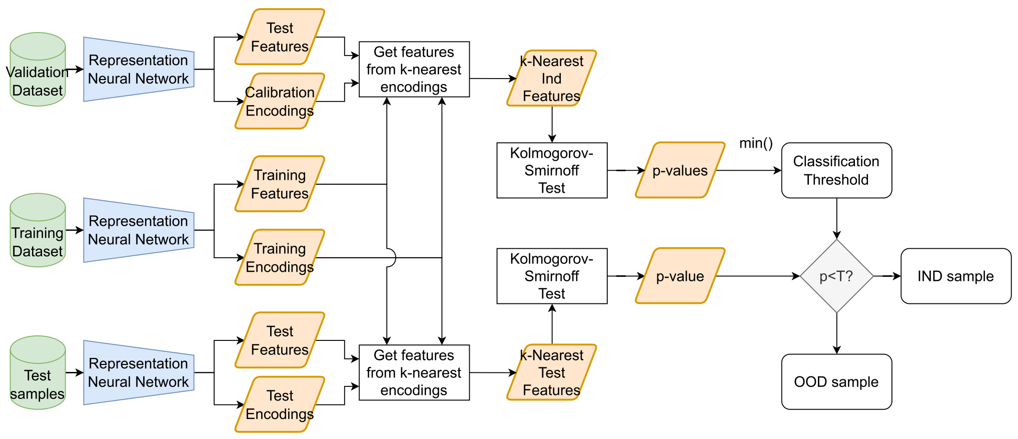


FIGURE 2. **kNNDSD: A neural network first computes representation vectors and OOD-ness features for the training data. This process is then repeated at runtime for incoming batches of data. For each datum in the batch, the features corresponding to the k-nearest training-samples in the encoding space are extracted.**
A Kolmogorov-Smirnoff test is then performed between these extracted features and the batch features, yielding a p-value. This is then compared to a p-value threshold as bootstrapped from a known InD dataset, such as the validation dataset. If the batch p-value is lower than the minimum p-value computed in the validation dataset, it is considered OOD, and InD otherwise.

for each dataset exhibited higher loss-values for all samples than in the InD folds. For NICO, NjordVid, and the Polyp datasets, this involved partitioning the dataset according to contexts, weather conditions, and hospital, respectively. For the remaining datasets, we used additive uniform noise sampled from $[0, 0.2]$.

For our analyses under semantic shift, we utilize four conventional benchmarking sets. Semantic shift detection is typically considered a more difficult task than covariate shift detection due to the tendency of neural networks to yield high-confidence predictions on data with different semantics [42]. This applies particularly to the so-called *near-OOD* datasets, where the data appears similar, but are semantically different:

- **CIFAR10**, with **MNIST** and **EMNIST** as OOD sets, and **CIFAR100** as a near-ood set.
- **CIFAR100**, with **MNIST** and **EMNIST** as OOD sets, and **CIFAR10** as a near-ood set.
- **MNIST**, with **CIFAR100** and **CIFAR10** as OOD sets, and **EMNIST** as a near-ood set.
- **EMNIST**, with **CIFAR100** and **CIFAR10** as OOD sets, and **MNIST** as a near-ood set.

2) SAMPLING-BIASES

We implement three forms of sampling-bias namely:

- **Temporal Bias**: test-samples are selected from a limited timeframe in the test-dataset, i.e., subsequent frames in video-datasets.
- **Label Bias**: test-samples are extracted according to class.
- **Synthetic sampling-bias**: test-samples are extracted specifically from a certain region of the distribution as determined by a K-Means clustering of the representation-model encodings. The value of K is selected according to the sample-size being used in the experiments, so that every sample being tested is drawn from the same cluster.

TABLE 1. **Overview of the datasets considered in our experiments and the applicable types of sampling-bias.**

Dataset	Temporal	Label	Synthetic
CIFAR10		✓	✓
CIFAR100		✓	✓
Imagenette		✓	✓
NjordVid	✓		✓
NICO++		✓	✓
Polyps	✓		✓
MNIST		✓	✓
EMNIST		✓	✓

We also include results without sampling-bias. Table 1 shows an overview over the datasets used, the shift types, and the applicable sampling-biases. All bias types were implemented as sampler objects in PyTorch.

3) METRICS

We follow convention and evaluate DSDs in terms of classification metrics:

- *Area Under Precision-Recall Curve (AUPR)*, which measures the trade-off between precision and recall.
- *Area Under Receiver-Operator Curve (AUROC)*, which measures the trade-off between sensitivty and specificity.
- *False Positive Rate (FPR)*, which estimates the likelihood of an InD sample being classified as OOD.
- *False Negative Rate (FNR)*, which estimates the likelihood of an OOD sample being classified as InD.
- *Balanced Accuracy (BA)*, which estimates the likelihood of correct detection balanced across both classes, i.e. $\frac{1-fnr+1-fpr}{2}$.

We note, however, that threshold-free metrics – i.e., AUPR and AUROC – do not account for the non-trivial problem of selecting an optimal threshold. The domain of p-values for the OOD samples is not known, except for that they should have a lower p-value than any of the p-values generated for the InD set. As mentioned in Section III, we selected a p-value threshold according to the lowest p-value in the validation set. However, we provide these metrics primarily to give an idea of the general seperability of the OOD and InD samples.

4) BASELINES

To adequately test our methods, we implement several baselines:

- **KS**: the best-performing configuration studied in [26], based on performing Kolmogorov-Smirnoff (KS)-tests on a classifier/VAE's latent-encodings.
- **Typicality** [27], a likelihood-based DSD based on the notion of typical sets. In practice, this involves computing log-likelihood estimates of generative models.
- **KL-divergence** [28], a common baseline for DSDs based on computing the cross-entropy between the output logits and a uniform distribution.
- **ODIN** [30], which in addition to computing KL-divergence also adds an adversarial perturbation to the inputs.
- **GradNorm** [29], which computes the norm of the gradients with respect to the KL-divergence term.

As these methods are primarily intended for classification tasks, some extensions were necessary for evaluation on the Njord and Polyp datasets, which, respectively, contain object-detection annotations and segmentation masks. For the GradNorm, ODIN, and KL-divergence DSDs we replaced the KL-divergence terms with loss-terms appropriate to the respective tasks. We used $k = 5$ for all experiments selecting neighbours in kNNDSD. This was selected on the basis of results from initial testing, though we note that differing values may yield differing results. In particular, higher values of k mean that the data are compared to a larger number of samples, which increases the effect of bias. Conversely, lower values of k mean that the data are compared to a lower number of samples, which often yields a less clear delineation simply due to the low sample sizes. In practical contexts, k should be adjusted in accordance with the size and variability of the data set, although this is a step we neglected due to the computational resources required.

The remaining baselines were implemented as described in their respective works, with the only difference being the use of statistical testing of multiple-samples as described in Section III, as opposed to using the OOD-ness measures to set thresholds directly or the bootstrapping procedure described for Typicality testing. In particular, we treated the outputs of all the DSDs as features for KS tests.

5) IMPLEMENTATION

We trained predictive models for all of our datasets to serve as representation models for the DSD methods and in order to verify the effect of the distributional shifts on performance. We used ResNets [43] for all classification datasets [44], DeepLabV3+ [45] for the polyp-datasets as implemented in segmentation-models-pytorch [46], and Ultralytics YoloV5 [47] for the Njord dataset. These models were selected for their relative simplicity in terms of implementation, popularity across domains, and decent performance. The models were trained using a Nvidia RTX3070 using PyTorch Lightning [48]. In the case that these models did not contain intermediate layers with a suitable dimensionality for use as a representation space, we instead used a Variational Autoencoder (VAE). For the Typicality tests, Glow was used as the representation model. This required resizing data to 32×32 in order for the model and its representations to fit on the available memory on our hardware. We observed that this yielded poor results on the Njord and Polyp datasets on both baselines and with kNNDSD during initial testing, and as a result we used a VAE for these datasets, computing likelihoods using importance sampling.

We implemented the sample-bias types as samplers, which were used in the dataloaders during our experiments. For the label-sampler, the data was sorted according to class-index, such that each batch consisted of samples of identical class. For the temporal bias, we sorted the samples according to the frame number in the video, which was the default order due to the naming conventions. For the synthetic bias, we sorted the data in accordance with the cluster-index as returned from the K-Means clustering.

As the DSDs studied in this work are sample-based, their performance varies considerably depending on sample-size. We evaluate the DSDs on N=30, 50, 100, 200, and 500. Where not specified, our metrics were averaged across this range.

B. COVARIATE SHIFTS UNDER BIAS

In this section, we present the results of our experiments on how DSDs behave for covariate shift prediction under sample-bias.

Table 2 summarizes our results, showing metrics aggregated over the tested sample-sizes and all three different sample-bias configurations for each dataset. In general, kNNDSD outperforms the baseline by a large margin in most of the tested configurations. kNNDSD is less likely to classify instances of sample-bias as distributional shift, and hence reduces the FNR by a significant margin of every dataset except NjordVid. The AUROC and AUPR scores also show that kNNDSD often improves the separability of InD and OOD samples. As discussed in Section IV-A, it should be noted, however, that AUROC and AUPR are not necessarily indicative of real-world performance due to the complexity involved in finding an optimal threshold.

For the Njord dataset, all the DSDs we tested generally performed poorly. kNNDSD outperforms the baselines for the majority of the DSDs, but worse with ODIN and equivalently poorly with typicality tests. We attribute this to the challenging nature of the dataset. As the data is collected from surveillance cameras, the sampled frames

TABLE 2. Aggregated results across sample-sizes and bias-types. The arrows in the header denote whether lower is better (↓) or higher is better (↑).

OOD Detector	FPR↓	FNR↓	BA↑	AUROC↑	AUPR↑
CIFAR100					
CrossEntropy	0.25/**0.12**	**0.07**/0.10	0.84/**0.89**	0.97/**0.99**	0.98/**0.99**
GradNorm	0.30/**0.12**	0.00/0.00	0.85/**0.94**	1.00/1.00	1.00/1.00
KS	0.67/**0.38**	0.00/0.00	0.67/**0.81**	1.00/1.00	1.00/1.00
ODIN	0.11/**0.08**	0.32/**0.05**	0.79/**0.94**	0.94/0.99	0.94/**0.99**
Typicality	0.31/**0.14**	0.00/0.00	0.84/**0.93**	1.00/1.00	1.00/1.00
CIFAR10					
CrossEntropy	0.31/**0.07**	0.00/0.00	0.85/**0.96**	0.98/**1.00**	0.99/**1.00**
GradNorm	0.24/**0.07**	0.00/0.00	0.88/**0.97**	1.00/1.00	1.00/1.00
KS	0.67/**0.30**	0.00/0.00	0.67/**0.85**	1.00/1.00	1.00/1.00
ODIN	0.21/**0.03**	0.00/0.00	0.89/**0.99**	1.00/1.00	1.00/1.00
Typicality	0.37/**0.11**	0.00/0.00	0.81/**0.94**	1.00/1.00	1.00/1.00
NICO					
CrossEntropy	0.25/**0.09**	0.08/**0.00**	0.83/**0.95**	0.94/**1.00**	0.64/**1.00**
GradNorm	0.29/**0.10**	0.08/**0.00**	0.82/**0.95**	0.93/**1.00**	0.63/**0.99**
KS	0.67/**0.41**	0.00/**0.01**	0.67/**0.79**	0.97/**0.99**	0.90/**0.98**
ODIN	0.22/**0.11**	0.32/**0.26**	0.73/**0.81**	0.81/**0.94**	0.44/**0.77**
Typicality_glow	0.24/**0.16**	**0.09**/0.11	0.84/**0.87**	0.95/**0.97**	0.77/**0.84**
Njord					
CrossEntropy	**0.49**/0.60	0.16/**0.02**	0.68/**0.69**	0.84/**0.94**	0.73/**0.87**
GradNorm	**0.53**/0.56	0.23/**0.01**	0.62/**0.71**	0.80/**0.94**	0.67/**0.86**
KS	**0.56**/0.65	0.41/**0.00**	0.51/**0.67**	0.55/**0.91**	0.48/**0.85**
ODIN	0.01/0.01	0.54/0.92	0.73/0.53	0.90/0.76	0.90/0.77
Typicality	0.01/**0.00**	1.00/**0.31**	0.49/**0.84**	0.00/**0.97**	0.31/**0.97**
Polyp					
CrossEntropy	0.52/**0.30**	**0.00**/0.07	0.74/**0.81**	0.96/**0.97**	0.97/**0.98**
GradNorm	**0.14**/0.23	**0.94**/1.00	0.46/0.38	**0.10**/0.08	**0.49**/0.48
KS	0.62/**0.35**	0.00/0.00	0.69/**0.83**	**1.00**/0.99	1.00/1.00
ODIN	0.62/**0.20**	**0.04**/0.15	0.67/**0.82**	0.86/**0.91**	0.90/**0.94**
Typicality	**0.00**/0.24	0.00/0.00	**1.0**/0.88	1.00/1.00	1.00/1.00
Imagenette					
CrossEntropy	0.43/**0.10**	0.08/**0.07**	0.74/**0.92**	0.95/**0.99**	0.95/**0.99**
GradNorm	0.45/**0.03**	0.00/0.00	0.77/**0.98**	0.99/**1.00**	0.99/**1.00**
KS	0.67/**0.49**	0.00/0.00	0.67/**0.76**	1.00/1.00	1.00/1.00
ODIN	0.22/**0.10**	**0.06**/0.13	0.86/**0.89**	0.97/**0.98**	0.97/0.97
Typicality	**0.19**/0.22	0.72/**0.00**	0.54/**0.89**	0.51/**1.00**	0.42/**1.00**

often consist of multiple nearly identical frames, which can skew the resulting p-values. The exception appeared to be ODIN, which performed well for large sample sizes, as shown in Fig. 5. Interestingly, kNNDSD increased the FPR and reduced the FNR for this dataset. One possible reason for this is that the VAE that was used as the representation model may not have learned a latent-space mapping which sufficiently discriminated between the pertinent covariates, and thus that the nearest-neighbour sampling proved less effective for reducing false positives. The AUROC and AUPR scores were generally increased with kNNDSD, indicating improved seperability between the InD and OOD classes.

Table 3 shows a breakdown of our results across sampling-biases. Without sampling-bias - i.e. sampling randomly from the entire dataset - all DSDs exhibit excellent performance, often with perfect accuracy. With sampling-bias, however, kNNDSD outperforms the baselines across all forms of bias.

Whereas data samples that on aggregate exhibit comparable loss values can generate a wide range of p-values with the baseline framework, kNNDSD yields comparable p-values for data exhibiting similar losses. This is illustrated for the KS-tests on the CIFAR10 dataset in Fig. 3. Sample-biases induce a shift in the p-values, which cause InD data to be classified as OOD, as they are under the p-value threshold computed from the validation set. Although it can be argued that it may be possible to circumvent this via a more robust threshold selection procedure, for instance finding the minimum p-value after a bootstrapping procedure under several forms of bias - kNNDSD provides a simpler and more robust solution. In addition to stabilizing the performance under naive thresholding, kNNDSD also improves the stability of the generated p-values. This is shown in Fig. 4. Note that in addition to the three distributions being more similar in the case of kNNDSD, the bias distributions also exhibit lower variability.

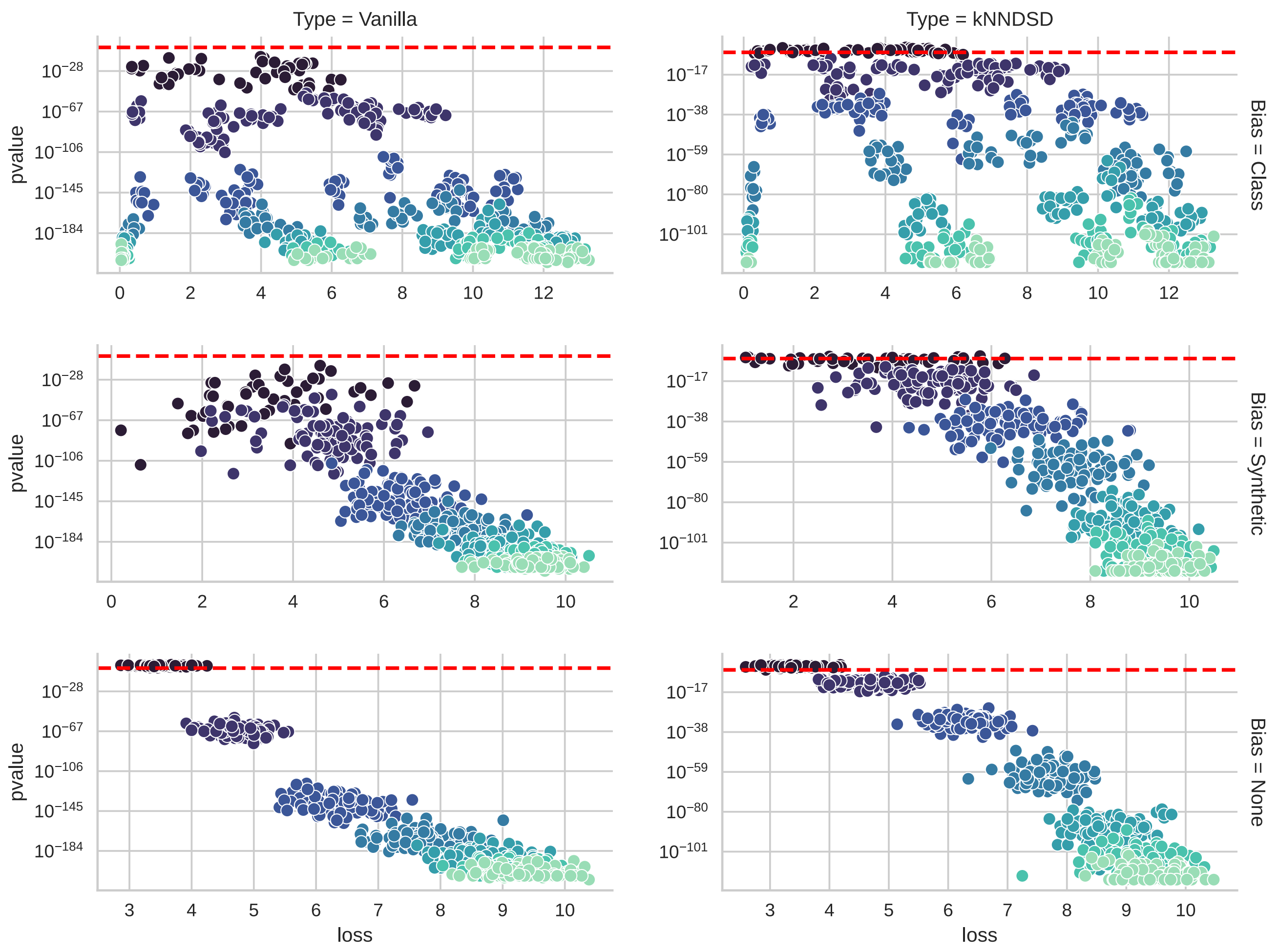


FIGURE 3. P-value vs. Loss plots for CIFAR10 across the sample-sizes. The p-values as generated by the baselines are greatly affected by sample-bias, whereas kNNDSD is more robust to sample-bias. The red-line represents the threshold extracted from the InD validation set without sample-bias.

The DSDs tested in this paper are all sample-based. The choice of sample-size affects both the computing overhead and the performance of the DSDs, and is thus an important consideration in practical scenarios. In some deployment scenarios, large sample sizes may be impractical, whereas in others, the added accuracy afforded by high sample sizes may be more beneficial than rapid execution. To investigate how kNNDSD behaved in this regard, we evaluated the sensitivity of the performance of our methods to the sample-size. We observed that the relationship between sample size and performance are more or less comparable across all tested methods, with kNNDSD typically outperforming the vanilla methods. For small sample-sizes, sample-bias is less of a factor, hence why many of the DSDs perform well even at low sample sizes. This is shown in Fig. 5. There does not appear to be a significant difference between kNNDSD and the baselines in this regard.

C. SEMANTIC SHIFTS UNDER BIAS

In this section, we present the results of our methods on semantic shifts. Overall, kNNDSD still generally outperforms the baselines. Table 4 summarizes the results. Aggregated across all data sets, kNNDSD exhibits consistently lower FPRs and higher BAs, with improved separability.

Table 5 shows metrics for kNNDSD and baselines for each pair of datasets, aggregated over the DSD methods and sample sizes. For the near-OOD CIFAR datasets, kNNDSD often increased the FNR and performed somewhat worse than the baselines. This is due to near-OOD samples often being close to borderline InD-samples in both latent- and feature-space. Thus, kNNDSD perceives these samples as similarly distributed, while the baselines are more likely to classify them correctly as they will be compared with the entire distribution. However, all DSDs performed poorly in this configuration, with the improved FPR for kNNDSD being offset by the FNR and vice versa.

V. DISCUSSION

In this section, we discuss our results, present the limitations of our work, and outline potential avenues of further work.

As we have shown, DSDs that utilize multiple samples are liable to fail under sampling bias, which is arguably an

TABLE 3. Breakdown of BAs by bias type, aggregated over sample-sizes.

OOD Detector	None	Label	Synthetic
CIFAR100			
CrossEntropy	**0.97**/0.94	0.84/**0.89**	0.71/**0.84**
GradNorm	1.00/1.00	0.80/**0.93**	0.75/**0.89**
KS	1.00/**1.00**	0.50/**0.77**	0.50/**0.66**
ODIN	0.84/**0.98**	0.76/**0.92**	0.76/**0.91**
Typicality	1.00/1.00	0.78/**0.90**	0.75/**0.89**
CIFAR10			
CrossEntropy	1.00/1.00	0.87/**0.98**	0.67/**0.91**
GradNorm	1.00/1.00	0.89/**0.97**	0.74/**0.92**
KS	1.00/1.00	0.50/**0.83**	0.50/**0.71**
ODIN	1.00/1.00	0.82/**0.99**	0.85/**0.97**
Typicality	1.00/1.00	0.72/**0.92**	0.72/**0.91**
NICO			
CrossEntropy	0.94/**1.00**	0.84/**0.93**	0.71/**0.93**
GradNorm	0.94/**1.00**	0.81/**0.92**	0.69/**0.92**
KS	**1.00**/0.99	0.50/**0.73**	0.50/**0.67**
ODIN	0.83/**0.86**	0.78/**0.81**	0.58/**0.77**
Typicality	**0.96**/0.95	0.80/**0.82**	0.75/**0.83**

OOD Detector	None	Synthetic	Temporal
Njord			
CrossEntropy	0.84/**0.98**	**0.60**/0.55	**0.60**/0.53
GradNorm	0.80/**0.99**	0.48/**0.61**	**0.60**/0.54
KS	0.54/**1.00**	0.48/**0.50**	**0.53**/0.52
Typicality	0.50/0.50	0.50/0.50	**0.51**/0.50
Polyp			
CrossEntropy	1.00/1.00	**0.69**/0.64	0.52/**0.80**
GradNorm	0.50/0.50	**0.52**/0.50	0.50/0.50
KS	1.00/1.00	0.54/**0.74**	0.52/**0.74**
ODIN	**0.97**/0.89	0.50/**0.84**	0.55/**0.74**
Typicality	1.00/1.00	**1.00**/0.65	**1.00**/0.99

OOD Detector	None	Label	Synthetic
Imagenette			
CrossEntropy	0.95/0.95	0.66/**0.91**	0.62/**0.90**
GradNorm	1.00/1.00	0.68/**0.77**	0.64/**0.74**
KS	1.00/1.00	0.50/**0.70**	0.50/**0.57**
ODIN	**0.98**/0.93	**0.87**/0.86	0.73/**0.88**
Typicality	1.00/1.00	0.57/**0.83**	0.51/**0.83**

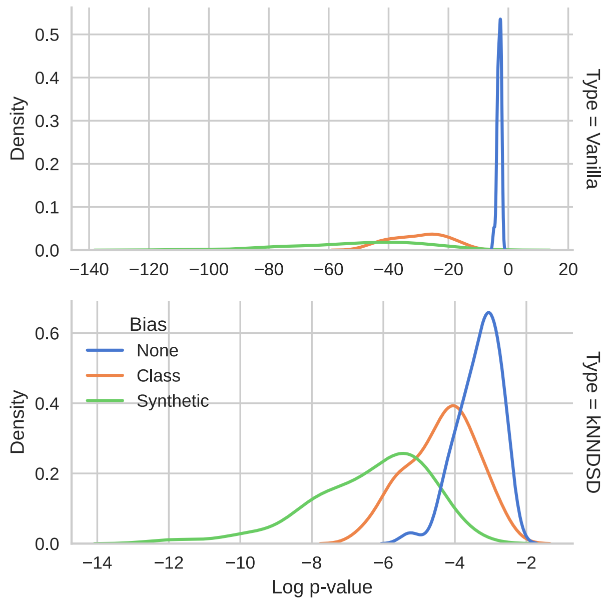


FIGURE 4. Distributions of p-values for kNNDSD and vanilla KN-tests under sample-bias for CIFAR10.

TABLE 4. Results for semantic shift detection, aggregated over OOD-sets.

OOD Detector	FPR	FNR	DR	AUROC	AUPR
CIFAR10					
CrossEntropy	0.31/**0.07**	**0.44**/0.79	0.63/**0.57**	0.75/**0.53**	0.61/**0.41**
GradNorm	0.24/**0.07**	**0.17**/0.25	0.79/**0.84**	**0.91**/0.89	**0.85**/0.84
KS	0.67/**0.27**	**0.10**/0.34	0.61/**0.70**	**0.78**/0.73	**0.75**/0.73
ODIN	0.21/**0.03**	0.60/**0.50**	0.59/**0.73**	0.66/**0.81**	0.49/**0.71**
Typicality	0.37/**0.11**	**0.18**/0.31	0.72/**0.79**	**0.85**/0.78	**0.82**/0.76
CIFAR100					
CrossEntropy	0.25/**0.12**	**0.31**/0.35	0.72/**0.76**	0.81/**0.85**	0.67/**0.76**
GradNorm	0.30/**0.12**	**0.20**/0.25	0.75/**0.81**	0.86/**0.87**	0.80/**0.81**
KS	0.61/**0.33**	**0.07**/0.24	0.66/**0.71**	**0.90**/0.78	**0.87**/0.78
ODIN	0.11/**0.08**	0.67/**0.55**	0.61/**0.69**	0.70/**0.78**	0.50/**0.63**
Typicality	0.31/**0.09**	**0.19**/0.29	0.75/**0.81**	**0.88**/0.87	**0.83**/0.82
EMNIST					
CrossEntropy	0.40/**0.14**	**0.02**/0.19	0.79/**0.83**	0.91/**0.96**	0.90/**0.96**
GradNorm	0.38/0.11	0.14/0.12	0.74/0.89	0.82/**0.96**	0.86/**0.96**
KS	0.49/**0.38**	**0.02**/0.00	0.74/**0.81**	0.74/**1.00**	0.80/**1.00**
ODIN	0.27/**0.08**	**0.11**/0.13	0.81/**0.90**	0.91/**0.98**	0.91/**0.97**
Typicality	0.28/**0.09**	0.06/**0.02**	0.83/**0.94**	0.96/**1.00**	0.96/**0.99**
MNIST					
CrossEntropy	0.63/**0.30**	0.00/0.00	0.69/**0.85**	0.98/**1.00**	0.93/**0.99**
GradNorm	0.64/**0.23**	0.11/**0.01**	0.63/**0.88**	0.54/**0.99**	0.49/**0.97**
KS	0.67/**0.64**	0.00/0.00	0.66/**0.68**	0.52/**1.00**	0.54/**1.00**
ODIN	0.43/**0.16**	**0.06**/0.22	0.75/**0.81**	0.90/**0.95**	0.78/**0.87**
Typicality	0.44/**0.18**	0.00/0.00	0.78/**0.91**	1.00/1.00	1.00/1.00

unavoidable property of real-world systems, in particular on streams of data. KNNDSD mitigates the effects of sample bias to a significant extent and produces superior performance in biased data while maintaining equivalent performance on unbiased data. Our method is relatively simple to implement, and is flexible enough to be utilized nearly with any sample-based DSD method. In general, we conjecture kNNDSD serves as a first step towards practically viable sample-based distributional shift detectors.

However, there are some limitations to our work. For example, we neglected to evaluate the various DSDs with a wider diversity of representation models. We suspect that the performance of our method - and more generally any representation-based DSDs - could be further improved by harnessing more sophisticated representation-learning frameworks, for instance contrastive-representation learning [49]. Fundamentally, all the methods we have tested in this work are based on the representative ability of the neural network, whether in terms of its gradients, its features, or its generative capacity. Implementing neural networks such that special care is taken towards generating salient representations

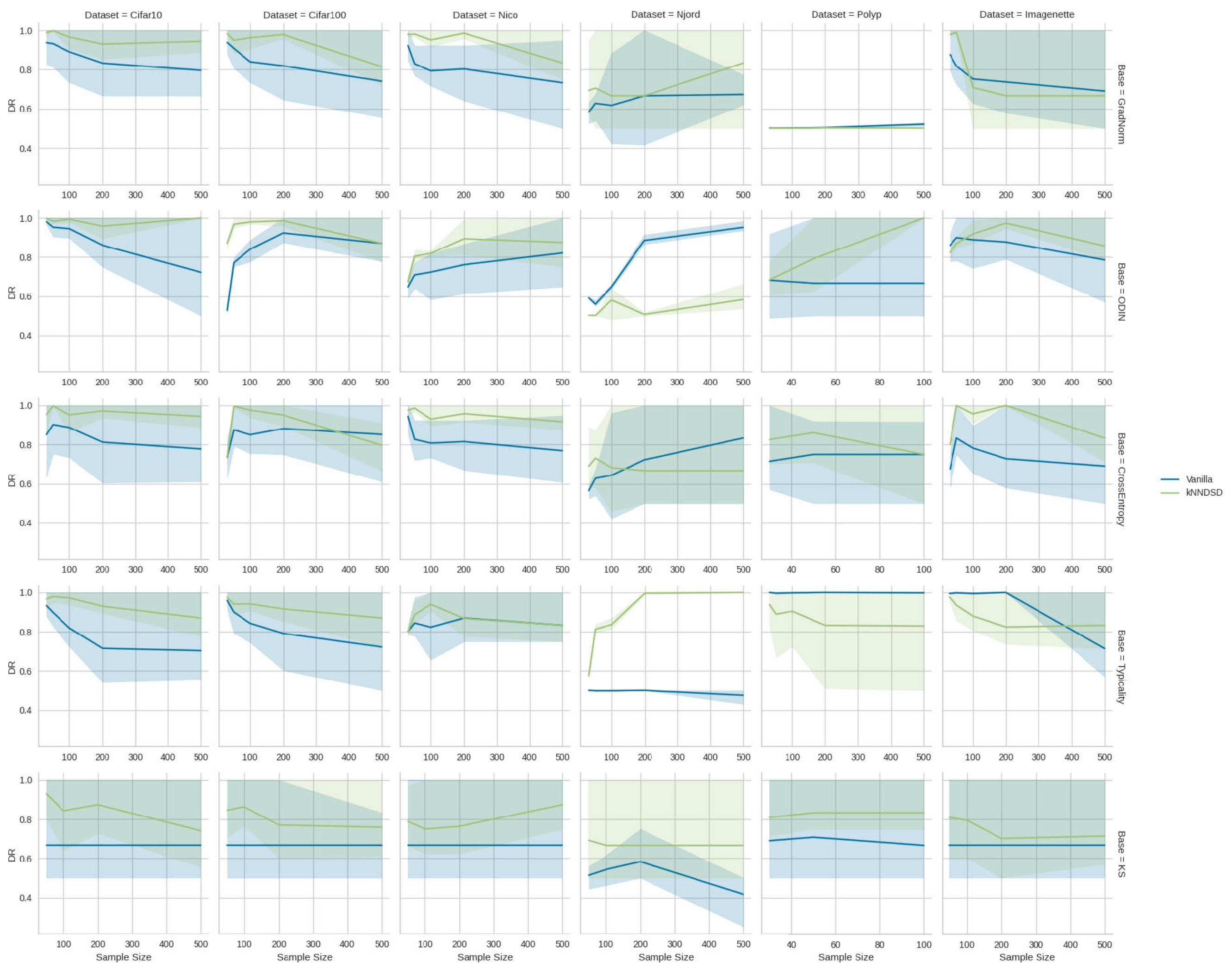


FIGURE 5. BA across sample-sizes and datasets. The green lines represent kNNDSD and the blue lines represent vanilla methods. The bands represent the span of p-values subject to the different tested sample-biases.

may improve the separability of InD and OOD samples, and is, as a consequence, likely to yield improved results. This may, in turn, facilitate a more robust selection of neighbors, further improving the performance of kNNDSD. Though investigating the extent thereof was beyond the scope of this paper, harnessing modern representation-learning frameworks and generative models is a promising direction for further research.

Moreover, we trained only one model instance for each DSD backbone and the corresponding dataset. Equivalently trained models can encode widely different characteristics [1], and these characteristics can exhibit different levels of utility to distinguish between samples InD and OOD. Our methodology could be improved by training multiple representation model instances. Additionally, similar to how ensembles improve the performance of classifiers, ensemble-based DSDs may yield improved performance, in particular in systems where ensembling is already utilized.

The potential utility of kNNDSD and sample-based DSDs in general is limited by the high computational costs involved. Although encodings of the predictive model can be used, a surrogate model is sometimes necessary, such as a VAE or other generative models. This effectively doubles the inference requirements for the system as a whole, on top of the computing time required for statistical testing and the time required to compute OOD-ness measures. This is further complicated by the need for computing the distances between each sample and the training population as well as the partial sorting of the resulting distances to find the nearest neighbours. For a sample size of $n = 100$ and encoding size $m = 512$, our method would on average take approximately 17 seconds to compute each p-value on our hardware. The capacity for real-time execution is as a result somewhat limited for sample-based DSDs, and particularly so for kNNDSD. Sample-based frameworks are also limited by their memory usage. Performing statistical tests requires

TABLE 5. Semantic Shift results split by OOD datasets aggregated over sample sizes and DSDs methods.

OOD	Baselines/KNNDSD				
OOD Detector	**FPR↓**	**FNR↓**	**DR↑**	**AUROC↑**	**AUPR↑**
		CIFAR10			
CIFAR100	0.36/**0.06**	**0.53**/0.87	**0.56**/0.53	**0.59**/0.50	**0.50**/0.38
EMNIST	0.36/**0.06**	**0.14**/0.24	0.75/**0.85**	**0.92**/0.89	**0.82**/0.81
MNIST	0.36/**0.06**	**0.23**/0.30	0.71/**0.82**	0.87/**0.88**	0.80/**0.83**
		CIFAR100			
CIFAR10	0.24/**0.11**	**0.57**/0.76	**0.59**/0.56	**0.64**/0.60	**0.51**/0.44
EMNIST	0.24/**0.11**	0.29/**0.24**	0.73/**0.83**	0.86/**0.93**	0.71/**0.83**
MNIST	0.24/**0.11**	0.16/**0.15**	0.80/**0.87**	0.93/**0.97**	0.89/**0.94**
		EMNIST			
CIFAR10	0.34/**0.11**	**0.03**/0.04	0.82/**0.92**	0.92/**0.99**	0.93/**0.99**
CIFAR100	0.34/**0.11**	0.04/0.04	0.81/**0.92**	0.91/**0.99**	0.92/**0.99**
MNIST	0.34/**0.11**	**0.14**/0.20	0.75/**0.84**	0.83/**0.95**	0.85/**0.94**
		MNIST			
CIFAR10	0.56/**0.29**	**0.02**/0.03	0.71/**0.84**	0.82/**1.00**	0.82/**0.99**
CIFAR100	0.56/**0.29**	0.03/**0.02**	0.71/**0.84**	0.81/**1.00**	0.82/**1.00**
EMNIST	0.56/**0.29**	**0.04**/0.10	0.70/**0.80**	0.79/**0.97**	0.67/**0.91**

storing features (and/or encodings) for all samples in the training set. Storing the training-encodings for the Imagenet dataset would, for instance, require approximately 53GB of memory, should all encodings be stored.

Our methodology could also be further refined by testing on a larger number of sample-bias and DSDs methods. One can argue that our use of synthetic k-Means clustering biasing is improper methodology, as kNNDSD explicitly selects sample according to the nearest neighbours found in this clustering space. We included an additional source of bias for this reason on all datasets, however we concede that additional biases would significantly increase the generality of our results. Furthermore, although we consider our baseline choices to cover a wide range of different approaches, we have not included any density-based DSDs or reconstruction-based DSDs.

As most of the work on DSDs has been done in the context of images, it may also be interesting to consider the viability of these methods in non-image domains, for instance as a means of detecting sensor-drift in ECG-data or other sensor-based datasets. This may also be a useful case study with regard to the relationship between anomaly-detection and distributional-shift detection, as there is a significant body of work focusing on anomaly-detection in time-series data.

There are also several means by which kNNDSD could be further improved and refined beyond utilizing more sophisticated representation models. The k-nearest-neighbour sample selection we implemented is relatively simple, for instance, and can probably be improved. One could, for instance, constrain the selection of multiple samples to a certain distance, further reducing the effect of sample-bias. A more sophisticated alternative is to perform kernel-density-estimation in the region around a given sample and select samples from this neighbourhood in accordance with the density. However, this would require a larger population of InD-samples, though we hypothesize that this can be achieved through data augmentation.

As discussed, the computational costs involved with our framework are likely a limiting factor for practical purposes. Optimizing kNNDSD such that real-time execution is feasible is as a result a necessary prerequisite for practical deployment. To this end, we plan to investigate more efficient methods of sample-selection, for instance through explicit modelling of the training manifold, as well as investigating whether or not it is possible to replace the statistical tests with a simpler measure, for instance cosine-similarity or some other distance-metric, while maintaining or improving performance. Finally, it may be possible to compute sparse representations of the training distribution, as opposed to storing the encodings themselves, reducing the overall memory footprint of our methods.

We also plan to investigate extensions of our framework in order to further improve the practical utility, for instance, by generating explanations of the DSDs outputs. In this way, users will not only be informed that the model is failing to generalize to the new data, but also *why* the model is failing. This may be achieved through, for instance, visualizing the features in the encodings that correspond to the distributional shift or by generating salience maps with respect to the most anomalous encodings in the latent space.

We also observed that the relationship between the p-values generated by our framework and the sample-losses appeared relatively well-behaved. This suggests that it may be possible to leverage our framework towards estimating the loss of arbitrary samples, significantly improving both the interpretability of DSDs and the transparency of the ML system in question. We plan on investigating this in further detail.

VI. CONCLUSION

In this paper, we have investigated the effect of sampling-bias on DSDs. We showed that existing methods fail under sampling-bias, and we proposed a simple extension that dampens these effects by limiting the source populations used in statistical tests to the nearest neighbours of the test population. We applied these methods on several datasets with various forms of distributional-shift and sampling-bias, and observed that our methods outperform the baseline by a significant margin when the dataset is subject to sampling-bias under both covariate and semantic shifts, with improvements in balanced accuracy typically ranging between 0.1 and 0.2. FPRs are halved when aggregated across all bias types, and often eliminated altogether on biased data.

REFERENCES

[1] A. D'Amour et al., "Underspecification presents challenges for credibility in modern machine learning," 2020, *arXiv:2011.03395*.

[2] J. Kauffmann, L. Ruff, G. Montavon, and K.-R. Müller, "The Clever Hans effect in anomaly detection," 2020, *arXiv:2006.10609*.

[3] R. Geirhos, J.-H. Jacobsen, C. Michaelis, R. Zemel, W. Brendel, M. Bethge, and F. A. Wichmann, "Shortcut learning in deep neural networks," *Nature Mach. Intell.*, vol. 2, no. 11, pp. 665–673, Nov. 2020, doi: 10.1038/s42256-020-00257-z.

[4] J. Liu, Z. Shen, Y. He, X. Zhang, R. Xu, H. Yu, and P. Cui, "Towards out-of-distribution generalization: A survey," 2021, *arXiv:2108.13624*.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[6] S. Ali et al., "Assessing generalisability of deep learning-based polyp detection and segmentation methods through a computer vision challenge," 2022, *arXiv:2202.12031*.

[7] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do ImageNet classifiers generalize to ImageNet?" 2019, *arXiv:1902.10811*.

[8] A. G. Wilson and P. Izmailov, "Bayesian deep learning and a probabilistic perspective of generalization," 2020, *arXiv:2002.08791*.

[9] T. Gokhale, S. Mishra, M. Luo, B. S. Sachdeva, and C. Baral, "Generalized but not robust? Comparing the effects of data modification methods on out-of-domain generalization and adversarial robustness," 2022, *arXiv:2203.07653*.

[10] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, Oct. 2018, doi: 10.1016/j.neucom.2018.05.083.

[11] A. Honga, G. Leeb, H. Leec, J. Seod, and D. Yeoe, "Deep learning model generalization with ensemble in endoscopic images," in *Proc. 3rd Int. Workshop Challenge Comput. Vis. Endoscopy (EndoCV), 17th IEEE Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2021, pp. 80–89. [Online]. Available: http://ceur-ws.org/Vol-2886/paper8.pdf

[12] V. Thambawita, S. A. Hicks, P. Halvorsen, and M. A. Riegler, "DivergentNets: Medical image segmentation by network ensemble," 2021, *arXiv:2107.00283*.

[13] B. Torpmann-Hagen, V. Thambawita, K. Glette, P. Halvorsen, and M. A. Riegler, "Segmentation consistency training: Out-of-Distribution generalization for medical image segmentation," 2022, *arXiv:2205.15428*.

[14] X. Li, L. Yu, H. Chen, C.-W. Fu, L. Xing, and P.-A. Heng, "Transformation consistent self-ensembling model for semi-supervised medical image segmentation," 2019, *arXiv:1903.00348*.

[15] X. Luo, J. Chen, T. Song, Y. Chen, G. Wang, and S. Zhang, "Semi-supervised medical image segmentation through dual-task consistency," 2020, *arXiv:2009.04448*.

[16] N. Ye, K. Li, H. Bai, R. Yu, L. Hong, F. Zhou, Z. Li, and J. Zhu, "OoD-bench: Quantifying and understanding two dimensions of out-of-distribution generalization," 2021, *arXiv:2106.03721*.

[17] A. Adadi and M. Berrada, "Peeking inside the black-box: A survey on explainable artificial intelligence (XAI)," *IEEE Access*, vol. 6, pp. 52138–52160, 2018.

[18] A. Jacovi, A. Marasović, T. Miller, and Y. Goldberg, "Formalizing trust in artificial intelligence: Prerequisites, causes and goals of human trust in AI," 2020, *arXiv:2010.07487*.

[19] S. Ali, N. Ghatwary, D. Jha, and P. Halvorsen, Eds., "Proceedings of the 3rd international workshop and challenge on computer vision in endoscopy (EndoCV 2021) co-located with with the 17th IEEE international symposium on biomedical imaging (ISBI 2021)," Apr. 2021, pp. 1–8.

[20] D. Hendrycks and T. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," 2019, *arXiv:1903.12261*.

[21] H. Hosseini, B. Xiao, and R. Poovendran, "Google's cloud vision API is not robust to noise," 2017, *arXiv:1704.05051*.

[22] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness," 2018, *arXiv:1811.12231*.

[23] B. Recht, R. Roelofs, L. Schmidt, and V. Shankar, "Do CIFAR-10 classifiers generalize to CIFAR-10?" 2018, *arXiv:1806.00451*.

[24] J. G. Moreno-Torres, T. Raeder, R. Alaiz-Rodríguez, N. V. Chawla, and F. Herrera, "A unifying view on dataset shift in classification," *Pattern Recognit.*, vol. 45, no. 1, pp. 521–530, Jan. 2012. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320311002901

[25] J. Yang, K. Zhou, Y. Li, and Z. Liu, "Generalized out-of-distribution detection: A survey," 2021, *arXiv:2110.11334*.

[26] S. Rabanser, S. Günnemann, and Z. C. Lipton, "Failing loudly: An empirical study of methods for detecting dataset shift," 2018, *arXiv:1810.11953*.

[27] E. Nalisnick, A. Matsukawa, Y. W. Teh, and B. Lakshminarayanan, "Detecting out-of-distribution inputs to deep generative models using typicality," 2019, *arXiv:1906.02994*.

[28] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," 2016, *arXiv:1610.02136*.

[29] R. Huang, A. Geng, and Y. Li, "On the importance of gradients for detecting distributional shifts in the wild," 2021, *arXiv:2110.00218*.

[30] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," 2017, *arXiv:1706.02690*.

[31] S. Farquhar, Y. Gal, and T. Rainforth, "On statistical bias in active learning: How and when to fix it," 2021, *arXiv:2101.11665*.

[32] P. P. Brahma, D. Wu, and Y. She, "Why deep learning works: A manifold disentanglement perspective," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 27, no. 10, pp. 1997–2008, Oct. 2016.

[33] M. Meilă and H. Zhang, "Manifold learning: What, how, and why," 2311, *arXiv:2311.037570*.

[34] S. Shalev-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.

[35] A. Krizhevsky and G. Hinton. (2009). *Learning Multiple Layers of Features From Tiny Images*. [Online]. Available: https://www.cs.utoronto.ca/~kriz/learning-features-2009-TR.pdf

[36] J. Howard. (Nov. 2020). *Imagenette*. [Online]. Available: https://github.com/fastai/imagenette

[37] T.-A.-S. Nordmo, A. B. Ovesen, B. A. Juliussen, S. A. Hicks, V. Thambawita, H. D. Johansen, P. Halvorsen, M. A. Riegler, and D. Johansen, "Njord: A fishing trawler dataset," in *Proc. 13th ACM Multimedia Syst. Conf.* New York, NY, USA: Association for Computing Machinery, Jun. 2022, p. 197, doi: 10.1145/3524273.3532886.

[38] X. Zhang, Y. He, R. Xu, H. Yu, Z. Shen, and P. Cui, "NICO++: Towards better benchmarking for domain generalization," 2022, *arXiv:2204.08040*.

[39] J. Bernal, F. J. Sánchez, G. Fernández-Esparrach, D. Gil, C. Rodríguez, and F. Vilariño, "WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians," *Computerized Med. Imag. Graph.*, vol. 43, pp. 99–111, Jul. 2015.

[40] D. Jha, P. H. Smedsrud, M. A. Riegler, P. Halvorsen, T. de Lange, D. Johansen, and H. D. Johansen, "Kvasir-SEG: A segmented polyp dataset," 2019, *arXiv:1911.07069*.

[41] J. Silva, A. Histace, O. Romain, X. Dray, and B. Granado, "Toward embedded detection of polyps in WCE images for early diagnosis of colorectal cancer," *Int. J. Comput. Assist. Radiol. Surg.*, vol. 9, no. 2, pp. 283–293, Mar. 2014.

[42] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult, "Toward open set recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 7, pp. 1757–1772, Jul. 2013.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015, *arXiv:1512.03385*.

[44] S. Marcel and Y. Rodriguez, "Torchvision the machine-vision package of torch," in *Proc. 18th ACM Int. Conf. Multimedia*. New York, NY, USA: Association for Computing Machinery, Oct. 2010, p. 1485, doi: 10.1145/1873951.1874254.

[45] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017, *arXiv:1706.05587*.

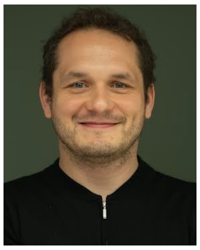
[46] P. Yakubovskiy. (2020). *Segmentation Models PyTorch*. [Online]. Available: https://github.com/qubvel/segmentation_models.pytorch

[47] G. Jocher et al., Oct. 29, 2020, "ultralytics/yolov5: v3.1—Bug fixes and performance improvements," *Zenodo*, doi: 10.5281/zenodo.4154370.

[48] W. Falcon. (Mar. 2019). *PyTorch Lightning*. PyTorch Lightning Team. [Online]. Available: https://github.com/Lightning-AI/lightning

[49] P. H. Le-Khac, G. Healy, and A. F. Smeaton, "Contrastive representation learning: A framework and review," *IEEE Access*, vol. 8, pp. 193907–193934, 2020, doi: 10.1109/ACCESS.2020.3031549.

BIRK TORPMANN-HAGEN received the bachelor's and master's degrees in robotics and intelligent systems from the University of Oslo, in 2020 and 2022, respectively. He is currently pursuing the joint Ph.D. degree with UiT: The Arctic University of Norway, in collaboration with SimulaMet. He has spent five semesters as a Teaching Assistant in courses covering AI, image analysis, and mechatronics; and has interned with Norwegian Defence Research Institute (FFI) and Domos, a tech start-up based in Oslo. His primary research interests include system support for deep learning, malware in neural networks, and distributional shift detection.



MICHAEL A. RIEGLER received the Ph.D. degree from the Department of Informatics, University of Oslo, Oslo, Norway, in 2017. In 2024, he holds the position of a Chief Research Scientist with SimulaMet, Oslo, and a Professor with OsloMet University, Oslo. His research interests include machine learning, video and image analysis and understanding, image processing, image retrieval, crowdsourcing, social computing, and biological and medical applications of machine learning.

PÅL HALVORSEN received the Ph.D. degree from the University of Oslo, Norway, in 2001. He is currently a Chief Research Scientist with SimulaMet, a Professor with OsloMet University, and a Professor II with the University of Oslo, Norway. At SimulaMet, he is also leading the Holistic Systems Research Department, which investigates challenges of complete end-to-end pipelines, with a particular focus on sport and medical applications. In this respect, his current research interests include distributed (multimedia) systems and content analysis from a performance and efficiency point of view. For more information visit the link (http://home.simula.no/ paalh).

DAG JOHANSEN is currently a Professor in computer science with UiT The Arctic University of Norway. His research interest includes distributed computer systems, with a focus on systems support for machine learning. He is exploring interdisciplinary research problems at the intersection of sport science, medicine, and law. A use-case receiving special attention is elite soccer performance development and quantification technologies as basis for evidence-based decisions. Focus is on intervention technologies where compliance and security are first-order concerns and design principles. He has co-founded a series of startups. He is an elected member of Norwegian Academy of Technological Sciences.