IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 18 February 2024, accepted 19 May 2024, date of publication 4 June 2024, date of current version 11 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3409587

RESEARCH ARTICLE

Foundation Models and Fine-Tuning: A Benchmark for Out of Distribution Detection

FRANCESCO CAPPIO BORLINO[1,2], LORENZO LU[1], AND TATIANA TOMMASI[1]
[1]Dipartimento di Automatica e Informatica (DAUIN), Politecnico di Torino, 10129 Turin, Italy
[2]Istituto Italiano di Tecnologia, 16163 Genoa, Italy

Corresponding author: Tatiana Tommasi (tatiana.tommasi@polito.it)

This work was supported by the Future Artificial Intelligence Research (FAIR) through the European Union Next-GenerationEU, PIANO NAZIONALE DI RIPRESA E RESILIENZA (PNRR)–MISSIONE 4 COMPONENTE 2, INVESTIMENTO 1.3–D.D. 1555 11/10/2022, under Grant PE00000013. The work of Tatiana Tommasi was supported by the European Project "European Lighthouse on Secure and Safe Artificial Intelligence" (ELSA).

ABSTRACT The rise of foundation models is pushing Computer Vision research towards a paradigm shift, in the wake of what already happened in the Natural Language Processing field. These models, trained at scale on huge data collections, provide high-quality representations that generalize well enough to be applied directly to downstream tasks, often outperforming task-specific models. The Out Of Distribution (OOD) detection problem, which involves the ability to recognize when test samples come from a previously unseen semantic category, represents one of the research fields in which this paradigm shift could have the greatest impact. However, existing testbeds are limited in scale and scope and get easily saturated when adopting foundation-based pretrainings. With this work, we introduce a new benchmark covering realistic yet harder OOD detection tasks to properly assess the performance of large pretrained models. We design an experimental framework to analyze specific choices in the model learning and use (which dataset, pretraining objective, OOD scoring function) and extensively evaluate the comparison to standard approaches that leverage a training phase on the available In Distribution (ID) data. The results highlight the actual performance benefits of leveraging foundation models in this context without any further learning effort, and identify situations where task-specific fine-tuning remains the best choice.

INDEX TERMS Fine-tuning-free, semantic novelty detection, open set recognition, out of distribution detection.

An autonomous agent navigating the word would inevitably encounter objects belonging to semantic classes it doesn't know about. Rather than assigning them to one out of a closed set of pre-defined labels, it should identify those samples as *unknowns* and potentially ask for external supervision. This behavior would avoid wrong decisions that may significantly impact safety, especially in autonomous driving or assistive robotics scenarios. The challenging task of managing unknown inputs is commonly named *Out Of Distribution* (OOD) *detection* and has largely attracted the attention of the computer vision community aiming for the development of dependable systems.

Most of the existing algorithmic solutions propose new approaches to learn from *In Distribution* (ID) support data and design training-time regularization or post-hoc scoring mechanisms to enhance OOD detection capabilities [1], [2], [3], [4]. Of course, growing the amount of ID data as well as model capacity would naturally mitigate the OOD detection problem, but covering everything a priori remains an unsustainable solution. Still, the rise of *foundation models* is starting to open new avenues for OOD research. They are large pretrained models whose internal embedding space captures extensive knowledge useful for a variety of downstream tasks [5], [6]. By exploiting their learned

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

representation, OOD detection becomes *training-free* and reduces to a simple similarity comparison between the test data and the support ID data. In this novel scenario, there are some major open research questions: how do the learning objective and dataset at the basis of the pretrained model affect the following OOD detection task? How should we compare two instances in the learned embedding space to get the best OOD detection performance? Is it useful to adapt the representation to the specific ID data via fine-tuning, and how to do that without losing relevant information?

With our work, we touch on all these questions to build a comprehensive picture of the state-of-the-art in training-free OOD detection. Our key contributions are:

- we design a novel testbed for OOD detection on high-resolution images, focusing on both intra-domain and cross-domain scenarios, to analyze models' performance in realistic deployment conditions. The intra-domain is composed of five settings with support and test samples drawn from the same database but covering partially overlapping label sets (semantic shift) for a wide variety of categorization tasks: from texture and aerial patterns to scenes, as well as coarse object classes and fine-grained car models. Instead, in the cross-domain scenario support and test data differ both in terms of label set (semantic shift) and visual style (covariate shift) (see Tab. 1);
- we present the first extensive analysis of training-free OOD detection approaches, considering both recently proposed algorithms and large-scale pretrained models. We analyze in depth the design choices that can significantly influence their effectiveness and show that foundation models enable a performance leap in OOD detection without any need for fine-tuning.
- we compare training-free approaches with traditional training-based ones, highlighting their advantages and disadvantages in relation to the application scenarios. We also discuss a possible combination to get the best out of both worlds.

We name our testbed OODDB[1] and publicly release it together with the whole code suite[2] used for the experiments to encourage further research.

TABLE 1. Setting definition with exemplar images for our OODDB dataset. In all the sub-cases, ID and OOD samples in the test belong to separate classes, thus we focus on semantic novelty detection. In the intra-domain track, support and test samples are drawn from the same dataset. In the cross-domain track, the support set belongs to a different visual domain from that of the test (covariate shift). This is obtained by always choosing the test as one of the DomainNet domains (single target ST), while the training support can either be one of the other domains (single source SS), or the combination of all the domains excluding that used for the test (multi-source MS). As a reference we also present the dataset benchmark from [7] where the OOD samples are drawn from a different dataset than the ID samples (semantic and covariate shift), thus defining a *far OOD* scenario.

Setting	Support (ID)	Test	
		ID	OOD
OODDB Intra-domain	Textures		
	PatternNet		
	SUN		
	Stanford Cars		
	Real (DomainNet)		
OODDB Cross-domain (DomainNet)	Real (SS)	Painting (ST)	
	No Painting (MS)		
Benchmark from [7]	ImageNet-1k		iNaturalist
			SUN
			Places
			Textures

I. RELATED WORKS

The independent and identically distributed (*iid*) assumption for training and testing data does not hold in many real-world applications, thus machine learning models require the capability of detecting and handling inputs from novel distributions. There are mainly two kinds of distribution shifts: in case of a *semantic* shift, in addition to seen classes, samples from unseen classes appear at test time and should be identified as unknown; in case of a *covariate* shift, the samples experienced at test time exhibit corruptions or style changes but semantically belong to one of the training categories and should be identified as such. The OOD detection literature mainly focuses on the former case, while the latter is a domain generalization problem [8], but the two shifts may occur together in realistic cross-domain OOD scenarios.

The first deep OOD detection baseline was proposed by Hendrycks et al. [1], who noticed that the prediction

[1]https://ooddb.github.io
[2]https://github.com/FrancescoCappio/OODDetectionBench

confidence of a model trained for classification on ID support data should be generally lower for OOD samples than for ID ones and could thus be used to separate the two groups. However, the overconfidence issue of deep classifiers [9], [10], has then pushed towards improving the known-unknown data separability, for example by temperature scaling [2], by activations rectification [11] and sparsification [4], or by introducing different scoring functions based on energy measures [3] and gradients [12]. All these are post-hoc OOD detection strategies that build on ID data classifiers and can also provide classification prediction for new ID samples, thus solving the so-called Open Set Recognition (OSR) task. Other OOD detection approaches are based on generative and reconstruction-capable models [13], [14]. In particular, the method proposed by Zhang et al. [15] combines a flow-based density estimation head with a classification one and is also suitable for OSR. Some works, rather than classification, exploit self-supervision to avoid the reliance on shortcuts often developed by supervised learning strategies. contrastive-based methods [16], [17] and masked image modeling are largely used for this scope. The latter can be naturally applied on tranformers-based architectures and provides a strong prior for OOD detection [18] by effectively applying reconstruction in a modern-fashion. Another family of OOD approaches learns distance metrics on ID support samples to then estimate the similarity of test samples to the training ID set [19], [20].

Most of the OOD detection methods have been evaluated on testbeds composed of a limited amount of low-resolution images (*e.g.* MNIST [21], CIFAR [22]) for which the cost of training on the ID support data is generally reduced. Only in the last years the OOD literature has started to consider large high-resolution computer vision databases, after Huang and Li [7] showed that the results on smaller datasets were not transferring to more realistic settings. In this context, the training effort can be partially alleviated by exploiting a pretrained model which is then fine-tuned on the ID support data. Still, as the pretrained model already contains a wealth of valuable information, it is reasonable to question the role of the fine-tuning process and wonder whether it may reduce the ability to recognize novelty rather than increasing it [23]. Recent research efforts have been directed towards designing OOD detection methods for high-resolution images that do not require fine-tuning on the ID support data. In particular, Cappio Borlino et al. [24] introduced a tailored pretraining strategy based on relational-reasoning representation learning that provides semantic similarity predictions on pairs of images. Lu et al. [25] later proposed an analysis of different relational-reasoning learning objectives designed to improve the model's transferability to the downstream OOD detection task. In the same work, it was also shown that some distance-based OOD detection methods originally proposed with a fine-tuning phase on ID support data, can be successfully applied without it [19] and [20]. This logic has been also extended to vision-language pretrained models defining the *zero-shot OOD detection* paradigm for which the access to ID samples is not even needed and the name of the ID classes is sufficient to identify the unknown data at deployment time [26], [27], [28].

The adoption of training-free OOD detection approaches can entail several advantages, which however are effectively unlocked only when using robust pretrained models able to generalize. In this respect some main ingredients deserve attention as the choice of the pretraining dataset, the network backbone, and the learning objective. The general trend of increasing the data cardinality (*e.g.* passing from ImageNet1k to ImageNet21k [29]) and the architecture capacity (*e.g.* passing from convolutional to transformer networks [30]) has been showing good performance [31]. At the same time, the improvements in terms of contrastive-learning and scalability of deep networks training, have enabled novel vision-language pretraining solutions as CLIP [6] and ALIGN [32], but also of purely vision-based self-supervised pretrainings as DINO [33] and DINOv2 [5]. In the following, we dive into these aspects to assess their effect on OOD detection.

II. DEFINITIONS AND PROBLEM STATEMENT

An OOD detection task is defined through two sets: the *support set* $\mathcal{S} = \{\boldsymbol{x}^s, y^s\}$ contains labeled samples, and the *test set* $\mathcal{T} = \{\boldsymbol{x}^t\}$ unlabeled ones. Support and test data are drawn from two different distributions with the main difference being a semantic shift: the set of support classes $\mathcal{Y}_\mathcal{S}$ does not match the set of classes to which test samples belong to, $\mathcal{Y}_\mathcal{T}$. In particular we are interested in the case $\mathcal{Y}_\mathcal{S} \subset \mathcal{Y}_\mathcal{T}$. We call ID the classes appearing in the support set and OOD the test classes not matching them: $\mathcal{Y}_{\mathcal{T}\setminus\mathcal{S}}$. The support set defines the *normality*, hence the purpose of an OOD detector is to detect test samples that deviate from this normality, by marking them as *unknown*. This means that the detector provides for each test sample a *normality score*, *i.e.* a scalar value that allows to discriminate known test samples from unknown ones by imposing a threshold on it.

We indicate as *training-based* OOD detection approaches all those strategies based on training or at least fine-tuning a deep network on the support data. This naturally provides a model aware of the reference data distribution (ID) which is then used to assess the ID likelihood of a test sample given the learned weights, gradients, or prediction output.

The *training-free* OOD detection methods bypass this learning step and offer some benefits. In particular, starting from a large and universally applicable pretrained model we expect it to generalize better than one fine-tuned on specific samples. Indeed, if the model is already proficient in differentiating and recognizing the ID classes, a further training stage may induce some bias that would hinder the semantic OOD detection process at deployment time rather than assisting it. Moreover, using a pretrained model would lead to a reduction in computational costs: it could be directly employed for any novel OOD detection task, provided that a method for comparing the features extracted from the

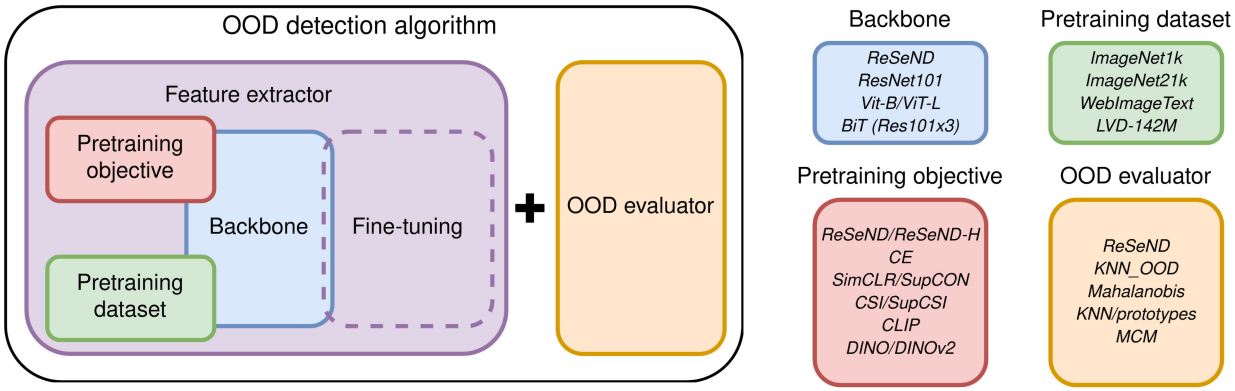

FIGURE 1. An OOD detection algorithm is composed of a feature extractor and an OOD evaluator. The former is obtained by choosing a backbone which is usually pretrained on a dataset with a specific objective and can be further fine-tuned on the ID support data. The latter defines how to compute a normality score using the learned representation. The right part of the figure illustrates the choices made for each component of the model in our experimental analysis.

reference ID support set data with those of the test data is established. This comparison technique is one of the essential components of a training-free OOD detection approach that we identify as OOD evaluator. In other words, its role is that of defining how a normality score should be computed for each test sample. The other component is the feature extractor which is simply tasked to process input data and provide representations. The feature extractor is usually a neural network backbone trained with a specific objective on a dataset. Fig. 1 shows a schematic representation of the OOD detection algorithms' components.

III. A COMPREHENSIVE OOD DETECTION BENCHMARK

The most straightforward method of selecting OOD samples to establish an evaluation testbed consists in choosing them out of a separate database from the one utilized for the ID set. However, this naïve approach disregards the semantic content of the images and may give rise to two issues. If the databases were gathered using comparable criteria in terms of acquisition devices, conditions, and covered classes, the distinction between ID and OOD may become too subtle or potentially non-existent, leading to an ill-posed problem [34]. On the other hand, if the databases are very different in terms of semantic content and visual domain, their distinction may be too easy (*far OOD*). On such an unrealistic benchmark (see the bottom part of Tab. 1) most of the methods provide exceptionally good results [26]. Only some minor changes have been introduced for the *full-spectrum* setting in [35] and [36]: the ID part of the test set is augmented with synthetically corrupted or style-changed images, while the OOD part remains drawn from a different dataset.

Given these considerations, we designed a novel large and comprehensive evaluation bed, for which we make sure to have ID and OOD samples in the test set drawn from the same dataset, but differing in semantic content (*near OOD*). The support ID data available for reference or for a potential training phase, may instead originate from the same or a different dataset than what was used for the test, thus defining an intra-domain and a cross-domain track.

A. INTRA-DOMAIN TRACK

In this track, the data of the support and test sets are drawn from the same visual distribution and the only shift considered is a semantic one. We selected five different datasets, chosen to represent a wide variety of categorization tasks (see the top part of Tab. 1). The available samples in every class of each dataset are divided into a train and a test split. Each OOD detection task is then built by randomly dividing the classes of the considered dataset into two groups: the train samples of ID classes compose the *support* set, while the test samples for both ID and OOD classes represent the *test* set. To improve the statistical significance of our analysis we repeat three times the random split of each datasets' classes into ID and OOD groups and report average results. We introduce the used datasets below.

Textures [37] is a dataset containing 5640 images of textural patterns belonging to 47 different classes. We randomly selected 23 of them as ID classes, and kept the remaining as OOD ones. We defined train and test splits by following the first fold provided by the original paper's authors for their cross-validation strategy, merging train and validation data. This dataset has already been used in the OOD detection literature, as part of the standard benchmark by Huang and Li [7] and to evaluate ReSeND in [24].

PatternNet [38] is a dataset of aerial high-resolution images which contains 38 classes with 800 images each. We selected 19 classes as ID, and kept the others as OOD. We used the train-test split provided by the original authors. To the best of our knowledge, this is the first time that this dataset has been used as part of an OOD detection benchmark. We chose this dataset as remote sensing categorization is an important task for many real-world applications and because its images differ significantly from the standard object-centric ones that are mostly used for research on visual categorization.

SUN [39] is a scene database containing 397 classes and 130k images in total. We selected 198 classes as ID, and kept the rest as OOD. We used the train-test split provided by the original authors. This dataset has already been used in the OOD detection literature [7], but this is the first time it is exploited with a class split in ID and OOD groups.

Stanford Cars [40] is a dataset designed for fine-grained cars classification. It includes more than 16k images divided into 196 classes. We adopted the train-test split provided by the original authors. We selected 98 classes as ID, and used the remaining 98 as OOD. This dataset has already been used as part of an OOD detection benchmark [26], but only as a whole to define ID classes.

DomainNet [41] is a large-scale dataset of common objects from 6 different visual domains. Our use of this database aligns with what was proposed in ReSeND [24], but we considered all 6 domains and 100 classes, selected through the Natural Language Toolkit [42] as the classes having the smallest semantic overlap with ImageNet1k's label set. We randomly chose 50 classes as ID, and the other 50 were kept as OOD. We used the same splits provided by the original authors. In the intra-domain track, both the support and test data come from the same visual domain. There are therefore 6 different intra-domain tasks for which we directly report the average results.

B. CROSS-DOMAIN TRACK

A covariate shift between training and test data may cause a drop in performance as the models tend to assimilate the training dataset's visual bias during the learning process. We deem robustness to this shift to be as important as the ability to detect a semantic novelty. For this reason, alongside the intra-domain track we consider a cross-domain one in which support and test sets come from different visual domains. It is composed of two settings both built on top of the DomainNet [41] dataset, for which we use the same ID-OOD and train-test splits of the Intra-domain case.

1) SINGLE-SOURCE → SINGLE-TARGET (SS → ST)

We adopted a single visual domain's ID train data as *support* set, while using another domain as *test* set. Overall DomainNet contains 6 domains, thus we create 30 different settings and report the average in our experimental analysis.

2) MULTI-SOURCE → SINGLE-TARGET (MS → ST)

We used 5 domains' ID train data together to build the *support* set, and left the remaining domain as *test*. We collected the results over all the 6 different settings that can be defined in this way and report the average

IV. EXPERIMENTAL SETTING

With our experimental analysis we want to investigate the effectiveness of OOD detection methods, mainly studying the role of each of the algorithms' components and evaluating if and when the fine-tuning process on the ID support data is needed. We provide here an overview of the choices operated on the different components.

A. OOD EVALUATOR

The OOD evaluator specifies how to compare the features extracted from the reference ID support data with those of the test data. A simple training-free way to do it is by exploiting sample distances and searching for the K nearest support neighbors (KNN) of each test sample for defining a normality score. This non-parametric approach has a variant proposed in [19], where the authors suggested to normalize the feature vectors before computing Euclidean distances (KNN_norm). Of course, these techniques require storing the feature representations of all support set samples, which can have a large memory and computational footprint in the case of very high support cardinality. A strategy that mitigates this issue and makes the predictions less sensitive to the presence of outliers, consists of calculating a prototype for each ID class by simple feature averaging. The test sample's normality is later estimated by computing their distance from the prototypes. Besides the Euclidean, the Mahalanobis distance [20] could be used after modeling the support data through class-conditional Gaussian distributions. It should be noticed that both the KNN_norm and the Mahalanobis strategies were originally proposed as fine-tuning-based approaches. Still, considering that they both exploit feature representations and do not need a classifier tailored for the downstream task, they can also be directly applied to representations extracted through a pretrained model.

Some OOD evaluators are also more tailored to the pretrained model learning process. In ReSeND [24] a dedicated relational module learns to compare pairs of samples and passes the information to a semantic similarity head. At deployment time the relational module is fed with the test sample and ID class prototypes to infer a normality score.

If the pretraining ran on data with combined vision and language cues, another training-free OOD detection strategy becomes possible via Maximum Concept Matching (MCM). Specifically, by using the CLIP's [6] text encoder the names of the ID classes identify the prototypes. In the inference phase, visual test samples are encoded through the CLIP's image encoder, and their distance from the known prototypes is estimated through cosine similarity, selecting the maximum value as zero-shot normality score.

Traditional training-based OOD methods perform a fine-tuning phase on support data. In that case, the most widely adopted evaluator is Maximum Softmax Prediction (MSP, [1]) which uses classification predictions to output normality scores. Alternative strategies such as ReACT [11] and ASH [4] exploit the energy score obtained by mapping the logit outputs to a scalar which is relatively lower for ID data.

B. FEATURE EXTRACTOR

For our analysis, we consider two macro-groups of OOD detection algorithms depending on the pretraining datasets:

one based on the standard ImageNet1k and the other exploiting modern representation learning paradigms from larger-scale data collections (ImageNet21k, WebImageText with 400M image-text pairs, and LVD-142M).

In terms of pretraining objective, adopting the supervised cross-entropy loss (CE) defines an essential reference baseline. We also investigate self-supervised contrastive learning with SimCLR [43], and SupCon [44] which merges the advantages of contrastive and supervised learning. Moreover, we include the contrasting shifted instances pretraining (CSI) which is a SimCLR variant with specific data augmentations tailored for novelty detection, as well as its supervised version (SupCSI) [16]. Among the most recent self-supervised approaches, we consider the knowledge self-distillation with no labels objective of DINO [33], and its variant DINOv2 [5] that exploits traditional contrastive learning jointly with masked image modeling. The latter provides a model with an intrinsic reconstruction ability which is well suited for OOD detection, as pointed out in [18].

Finally, in the case of large vision and language datasets, the CLIP contrastive objective consists of mapping the image embedding close to its corresponding text embedding. This happens by increasing the cosine similarity score of images and text that are associated, while minimizing the similarity between images and texts that do not occur together.

Regarding the backbones, we consider both convolutional and transformer architectures. Specifically, we use ResNet101 [45] (44M param.) and BiT [46] (380M param.) which is a Wide ResNet network with all the Batch Normalization (BN) layers replaced by a combination of Group Normalization (GN) [47] and Weight Standardization (WS) [48] ones, as the latter have been shown to provide better performance in a transfer scenario. From the Vision Transformer models [30], we adopt the base version ViT-B (86M param.) and the large one ViT-L (307M param.).

Overall, we examine various combinations of these components, ensuring coverage of both supervised and self-supervised objectives, and employing both convolutional and transformer-based architectures for small and large databases. To choose the combinations we take into account the most recent literature results and publicly available checkpoints of pretrained models: for instance, DINOv2 identifies not only a training objective but a whole model built with the ViT-L architecture on the LVD-142M dataset. Concerning the OOD evaluators, we adopt KNN and prototypes for all the considered algorithms. Indeed they allow us to assess the representation capabilities of various pretrainings as a foothold for OOD detection. When dealing with supervised CE-based approaches we also consider KNN_norm and test the Mahalanobis distance.

We remark that ReSeND stands apart as it was originally designed for training-free OOD detection with a tailored backbone, objective, and OOD evaluator. Its transformer-based architecture (40M param.) was trained on ImageNet1k to recognize if a pair of samples represents the same semantic class or not by estimating a binary score via the $L2$ loss. The variant ReSeND-H [25] exploits a hinge loss that allows for better controlling same-vs-different class separability.

We also treat CLIP as an exception with respect to all the other approaches as it is the only one trained both with vision and language modalities on the private WebImageText dataset and is evaluated in zero-shot fashion via MCM.

V. METRICS

We evaluate the OOD detection performance with two commonly used metrics based on the concepts of True Positive (TP, known samples detected as known), False Positive (FP, unknown samples detected as known), True Negative (TN, unknown samples detected as unknown) and False Negative (FN, known samples detected as unknown). The Area Under the Receiver Operating Characteristic Curve (AUROC, the higher the better, values in [0, 1]) is the area under the curve obtained by plotting the TP rate against the FP rate when varying the normality score threshold. It represents the probability that a known test sample receives a normality score higher than an unknown test sample. The FPR95 (the lower the better, values in [0,1]) is the FP rate when the TP rate is 95% (it is sometimes referred to as FPR@TPR95): it counts the ratio of unknown samples marked as known when the normality score threshold is set to mark 95% of known samples as known.

For what concerns the ability to correctly classify known samples, we simply use the classification accuracy (ACC) computed on all the known test samples.

VI. TRAINING-FREE OOD DETECTION RESULTS

Tab. 2 reports the results obtained by the described training-free methods on our benchmark.

A. INTRA-DOMAIN TRACK

The top left portion of the table presents the Imagenet1k-based pretraining. We can draw some general conclusions by looking at the AVG column. The ReSeND-H method (AUROC 0.660) outperforms the contrastive approaches (best AUROC SimCLR 0.533, SupCon 0.592, CSI 0.643, SupCSI 0.642) and it is competitive with CE (AUROC ResNet101 KNN 0.678, ViT-B KNN 0.632), except when CE exploits the KNN_norm evaluation (AUROC ResNet101 0.720, ViT-B 0.705). Indeed, normalizing the features to project all the data onto the unit hypersphere is particularly effective when the representation is obtained with a CE loss-based classifier. This loss is not zero in case of correct prediction and can be further minimized by increasing the norm of the feature vectors. In other words, the norm of the features extracted by models trained with this learning objective should be discarded as it brings misleading information. This explains the advantage of passing from KNN to KNN_norm. The described behavior does not occur in the case of contrastive self-supervised pretraining as these methods do not involve an unwanted maximization of the

TABLE 2. Training-free OOD detection results on the OODDB benchmark. We consider the experiments on ImageNet1k and those on larger datasets as separate settings, displayed in the two horizontal subparts of the table (top and bottom). For each of them, we use the bold font to highlight the best result per column. Moreover, the table presents in two vertical parts (left and right) the intra-domain and cross-domain track results.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
ImageNet1k	ReSeND	ReSeND	ReSeND	0.667	0.881	0.871	0.550	0.582	0.911	0.494	0.947	0.612	0.900	0.645	0.838	0.560	0.930	0.581	0.924
		ReSeND-H		0.679	0.867	0.909	0.463	0.586	0.910	0.496	0.951	0.628	0.894	0.660	0.817	0.553	0.933	0.575	0.927
	ResNet101	CE	KNN_norm	0.758	0.789	0.936	0.296	0.639	0.874	**0.571**	**0.929**	0.696	0.815	0.720	0.740	**0.576**	0.915	**0.617**	0.888
			Mahalanobis	0.641	0.885	0.850	0.550	0.574	0.908	0.544	0.936	0.609	0.893	0.643	0.835	0.540	0.934	0.555	0.929
			KNN	0.698	0.851	0.878	0.542	0.601	0.899	0.566	0.932	0.650	0.871	0.678	0.819	0.551	0.936	0.580	0.924
			prototypes	0.667	0.884	0.827	0.684	0.615	0.887	0.540	0.938	0.609	0.906	0.652	0.860	0.537	0.942	0.540	0.941
		SimCLR	KNN	0.504	0.941	0.663	0.877	0.501	0.953	0.500	0.945	0.498	0.950	0.533	0.933	0.488	0.954	0.487	0.955
			prototypes	0.496	0.948	0.564	0.956	0.498	0.951	0.498	0.941	0.499	0.947	0.511	0.949	0.500	0.950	0.506	0.951
		SupCon	KNN	0.563	0.931	0.838	0.653	0.510	0.952	0.513	0.945	0.536	0.939	0.592	0.884	0.503	0.947	0.509	0.946
			prototypes	0.528	0.948	0.647	0.872	0.492	0.953	0.503	0.947	0.515	0.947	0.537	0.933	0.502	0.949	0.506	0.950
		CSI	KNN	0.652	0.870	0.884	0.553	0.575	0.920	0.520	0.943	0.583	0.914	0.643	0.840	0.516	0.944	0.532	0.936
			prototypes	0.613	0.903	0.790	0.782	0.546	0.937	0.499	0.948	0.543	0.937	0.598	0.902	0.511	0.949	0.510	0.948
		SupCSI	KNN	0.618	0.896	0.877	0.528	0.578	0.917	0.533	0.939	0.604	0.911	0.642	0.838	0.525	0.942	0.543	0.934
			prototypes	0.590	0.912	0.783	0.708	0.561	0.928	0.511	0.944	0.565	0.930	0.602	0.885	0.518	0.944	0.517	0.945
	ViT-B	CE	KNN_norm	0.725	0.804	0.914	0.380	0.628	0.892	0.575	0.928	0.686	0.836	0.705	0.768	0.562	0.922	0.597	0.905
			Mahalanobis	0.610	0.915	0.795	0.742	0.590	0.931	0.542	0.940	0.602	0.905	0.628	0.887	0.530	0.941	0.542	0.934
			KNN	0.595	0.919	0.842	0.644	0.552	0.932	0.550	0.941	0.620	0.901	0.632	0.867	0.522	0.943	0.546	0.932
			prototypes	0.538	0.916	0.810	0.665	0.568	0.911	0.530	0.945	0.592	0.898	0.608	0.867	0.518	0.943	0.520	0.938
		DINO	KNN	**0.763**	**0.758**	**0.956**	**0.227**	**0.664**	**0.838**	0.566	0.929	**0.700**	**0.811**	**0.730**	**0.713**	0.575	**0.906**	0.616	**0.869**
			prototypes	0.764	0.775	0.932	0.325	0.683	0.801	0.530	0.938	0.664	0.843	0.714	0.736	0.558	0.918	0.573	0.907
ImageNet21k	ViT-L	CE	KNN_norm	0.726	0.810	0.812	0.674	0.726	0.828	0.571	0.933	0.699	0.793	0.707	0.808	0.600	0.902	0.622	0.881
			Mahalanobis	0.706	0.830	0.664	0.843	0.783	0.746	0.553	0.939	0.668	0.802	0.675	0.832	0.590	0.903	0.626	0.868
			KNN	0.710	0.850	0.809	0.675	0.711	0.859	0.565	0.934	0.689	0.832	0.697	0.830	0.586	0.922	0.611	0.903
			prototypes	0.675	0.865	0.758	0.758	0.782	0.791	0.553	0.930	0.676	0.855	0.689	0.840	0.563	0.931	0.588	0.915
	BiT (Res101x3)	CE	KNN_norm	**0.832**	**0.635**	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			Mahalanobis	0.709	0.871	0.756	0.723	0.653	0.891	0.524	0.942	0.614	0.892	0.651	0.864	0.545	0.935	0.576	0.917
			KNN	0.804	0.729	**0.950**	**0.254**	0.734	0.808	0.578	0.922	0.721	0.788	0.757	0.700	0.581	0.912	0.636	0.864
			prototypes	0.740	0.812	0.897	0.487	0.748	0.787	0.545	0.932	0.668	0.853	0.720	0.774	0.550	0.928	0.567	0.919
WebImageText	ViT-L	CLIP	MCM	0.701	0.861	0.776	0.851	0.766	0.742	0.517	0.944	**0.817**	0.698	0.716	0.819	**0.817**	**0.698**	**0.817**	**0.698**
LVD-142M		DINOv2	KNN	0.795	0.735	0.911	0.378	0.775	0.736	**0.720**	**0.848**	0.791	**0.684**	**0.798**	0.676	0.676	0.836	0.713	0.808
			prototypes	0.803	0.722	0.904	0.430	**0.834**	**0.593**	0.626	0.902	0.784	0.687	0.790	**0.667**	0.670	0.851	0.722	0.786

feature norms. This claim is confirmed by Fig. 2 which shows a comprehensive comparison of the KNN_norm results with the corresponding KNN ones (an extended version of this bar plot for all the experimental settings is provided in the appendix Sec. B-A).

Finally, the top results obtained with DINO (AUROC KNN 0.730) indicate that the most recent self-supervised approaches provide a significant leap over both supervised and standard self-supervised contrastive pretraining methods. It suggests that other potential improvements in the semantic novelty detection literature could be obtained by scaling up self-supervised strategies, even introducing new tailored objectives.

The comparison between KNN and prototypes deserves a last remark. In general, the first evaluation method provides better results than the second. However, the inference cost in the two cases is significantly different: with prototypes it is enough to calculate the distances of the test sample to each class centroid, while with KNN the whole support set is considered as reference. So for KNN it is necessary to evaluate the distance of the test sample to all the support data which may be problematic when their cardinality is so large to not fit into the system memory.

The bottom left part of the table contains the results produced with large-scale pretraining. The first rows of this group shows that applying traditional supervised CE-based learning on top of ImageNet21k does not provide an average performance improvement (AUROC ViT-L KNN_norm 0.707) over the best ImageNet1k case (AUROC ViT-B DINO KNN 0.730). A visible advantage can instead be obtained with a pretraining designed explicitly for transfer learning, which is the case of BiT (AUROC CE KNN_norm 0.767).

Interestingly, the vision and language CLIP pretraining with the MCM evaluation strategy, which exploits the class names as support prototypes, does not produce top results. Indeed the textual encoder works well for DomainNet and its object names, but struggles when dealing with Texture's or PatterNet's labels and fine-grained classes in Stanford Cars.

The best average performance is obtained with DINOv2, highlighting again the robustness of self-supervised representation learning approaches, especially when applied at scale. It is also remarkable that this pretraining obtains extremely similar results when tested with KNN or prototypes (AUROC 0.798 and 0.790), showing that it is possible to build compact known class clusters for which the centroids are not far away from the cluster boundaries. Thus, by exploiting the prototypes it is possible to significantly reduce the computational cost at deployment time without sacrificing performance. This also suggests that overclustering may provide further benefits. To verify this hypothesis we ran k-means on the support set features to select 10 centroids per class which are then leveraged instead of the whole feature set to represent the ID data. In this way, the average results of DINOv2 pass from AUROC 0.798 to 0.815 and from FPR95 0.667 to 0.626 (see Sec. B-B in the appendix for the complete set of results with k-means clustering centroids).

B. CROSS-DOMAIN TRACK

The right part of the table contains the cross-domain track results. At first glance, it is evident from the low AUROC values that the OOD detection task is now much more challenging than in the intra-domain case.

The top right table section contains the ImageNet1k-based results with the best performances obtained by

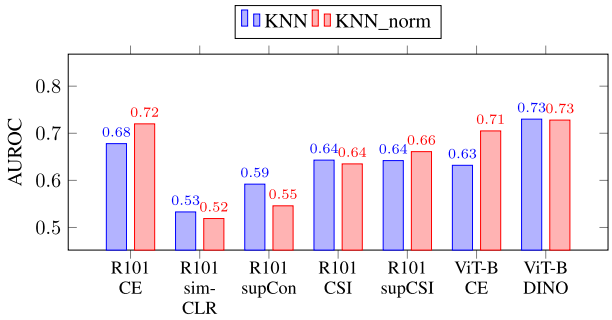

FIGURE 2. Training-free OOD detection results on the OODDB benchmark. KNN vs KNN_norm intra-domain performance for ImageNet1k-based pretrainings. Note that KNN_norm has a clear advantage over KNN when the pretraining objective is CE.

TABLE 3. Training-free OOD detection results on the OODDB benchmark. Analysis of the CLIP visual encoder when using KNN and prototypes evaluators vs the CLIP text encoder exploited by the MCM evaluator. We also report the same DINOv2 results presented in Tab. 2 as reference.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain AVG AUROC↑	Intra-domain AVG FPR95↓	Cross-domain SS → ST AUROC↑	Cross-domain SS → ST FPR95↓	Cross-domain MS → ST AUROC↑	Cross-domain MS → ST FPR95↓
WebImageText	ViT-L	CLIP	MCM	0.716	0.819	**0.817**	**0.698**	**0.817**	**0.698**
			KNN	**0.812**	**0.658**	0.648	0.915	0.695	0.894
			prototypes	0.798	0.699	0.636	0.928	0.677	0.921
LVD-142M		DINOv2	KNN	0.798	0.676	0.676	0.836	0.713	0.808
			prototypes	0.790	0.667	0.670	0.851	0.722	0.786

ResNet101 CE KNN_norm and ViT-B DINO KNN (*e.g.* in MS → ST respectively AUROC 0.617 and 0.616).

The bottom right section of the table presents the results with large-scale pretraining: we can observe how ViT-L with DINOv2 shows a gain over the small-scale pretraining (in MS→ ST prototypes DINOv2 has AUROC 0.722, higher than the top AUROC 0.617 at small-scale just mentioned above.)

Overall, the best results are those of CLIP with MCM (AUROC 0.817). Such an outcome could be expected in this particular setting as the zero-shot MCM evaluator completely avoids the use of the support data (confirmed by the fact that the SS→ST and MS→ST results are identical). Its normality score depends only on the distance of each test sample to the support class names, projected into the embedding space via the CLIP text encoder. Thus the language modality disregards the visual shift between the two domains.

Before concluding this section, it is worthwhile to delve deeper into the capabilities of the CLIP pretraining objective to disentangle the visual and textual representation when performing OOD detection. Specifically, we wonder what would be the results when using the support visual data and adopting KNN or prototypes as evaluation techniques. The results in Tab. 3 highlight that this strategy is much more effective than MCM in the intra-domain track, although it provides lower results (even below those of DINOv2) in the cross-domain track. Overall, this confirms the importance of exploiting language in case of visual domain shift but also indicates that blindly relying on it in every setting may be deleterious.

VII. COMPARISON WITH FINE-TUNING-BASED STATE-OF-THE-ART

As previously discussed, the OOD detection literature considers fine-tuning on the support set as a standard strategy. To assess the potential of the training-free methods we compare the top performing methods from Tab. 2 with representative fine-tuning-based state-of-the-art OOD detection methods. Specifically, we start from pretrained models on ImageNet1k, ImageNet21k, and LVD-142M. In terms of backbone architectures, we consider ViT-B, ViT-L, and BiT with the related CE and DINO/DINOv2 objectives. Once fine-tuned, we test the post-hoc methods MSP [1], ReAct [11], KNN_norm in its original fine-tuning-based version [19], ASH [4], and Flow which exploits for OOD detection the flow-based density estimator of the OSR method OpenHybrid [15].

In Tab. 4 we report the obtained results that overall show how fine-tuning provides a clear advantage in the intra-domain track (left part of the table). In the cross-domain track (right part of the table), the fine-tuning produces mixed results. For instance, the AUROC of the fine-tuned DINOv2 KNN_norm is 0.748 for MS→ST, with a clear improvement over the corresponding not-fine-tuned result of KNN 0.713. However, the trend inverts if we consider FPR95 (the lower the better), which is 0.832 for the fine-tuned KNN_norm and 0.808 for the not-fine-tuned KNN. A similar behavior holds for the SS→ST case. This inconsistency between AUROC and FPR95 is a direct consequence of having a biased feature extractor: OOD samples are represented with features that have been tailored for ID samples. In the fine-tuned embedding space the novel classes appear close to the known ones and in particular closer than they would have been when using the original not-fine-tuned representation. As a result, the ranges of normality scores provided for ID and OOD test samples become similar, as shown by Fig. 3 (see the black/blue colorful bars over the horizontal axis), this makes ID and OOD data less distinguishable despite the increased distance between the peaks and correspondingly increased AUROC. Indeed, a lower threshold has to be chosen to have a TPR of 95%, which leads to a worse FPR95 score.

Regardless of these details, the best performance remains that of the not-fine-tuned MCM that exploits language.

Overall, it is worth noticing that with fine-tuning the effect of the size of the pretraining datasets is less evident than in the training-free case. Specifically, the difference between the best AUROC when using ImageNet1k or ImageNet21k/LVD142M can get up to 0.1 in the training-free case and it is instead below 0.06 in the fine-tuned case. We also highlight that the very recent ASH approach is proficient only with convolutional architectures (BiT), while it performs poorly with transformer-based backbones. This holds for both the intra-domain and cross-domain tracks and confirms a behavior already noted in [35].

TABLE 4. Traning-free (top) vs Training-based (bottom) OOD detection results on the OODDB benchmark. For each setting we use the bold font to highlight the best result per column. Moreover, the table presents in two vertical parts (left and right) the intra-domain and cross-domain track results.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
training-free methods																**training-free methods**			
ImageNet1k	ViT-B	CE	KNN_norm	0.725	0.804	0.914	0.380	0.628	0.892	0.575	0.928	0.686	0.836	0.705	0.768	0.562	0.922	0.597	0.905
			Mahalanobis	0.610	0.915	0.795	0.742	0.590	0.931	0.542	0.940	0.602	0.905	0.628	0.887	0.530	0.941	0.542	0.934
			KNN	0.595	0.919	0.842	0.644	0.552	0.932	0.550	0.941	0.620	0.901	0.632	0.867	0.522	0.943	0.546	0.932
			prototypes	0.538	0.916	0.810	0.665	0.568	0.911	0.530	0.945	0.592	0.898	0.608	0.867	0.518	0.943	0.520	0.938
		DINO	KNN	0.763	0.758	**0.956**	**0.227**	0.664	0.838	0.566	0.929	0.700	0.811	0.730	0.713	0.575	0.906	0.616	0.869
			prototypes	0.764	0.775	0.932	0.325	0.683	0.801	0.530	0.938	0.664	0.843	0.714	0.736	0.558	0.918	0.573	0.907
ImageNet21k	BiT (Res101x3)	CE	KNN_norm	**0.832**	**0.635**	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			Mahalanobis	0.709	0.871	0.756	0.723	0.653	0.891	0.524	0.942	0.614	0.892	0.651	0.864	0.545	0.935	0.576	0.917
			KNN	0.804	0.729	0.950	0.254	0.734	0.808	0.578	0.922	0.721	0.788	0.757	0.700	0.581	0.912	0.636	0.864
			prototypes	0.740	0.812	0.897	0.487	0.748	0.787	0.545	0.932	0.668	0.853	0.720	0.774	0.550	0.928	0.567	0.919
WebImageText	ViT-L	CLIP	MCM	0.701	0.861	0.776	0.851	0.766	0.742	0.517	0.944	**0.817**	0.698	0.716	0.819	**0.817**	**0.698**	**0.817**	**0.698**
LVD-142M		DINOv2	KNN	0.795	0.735	0.911	0.378	0.775	0.736	**0.720**	**0.848**	0.791	**0.684**	**0.798**	0.676	0.676	0.836	0.713	0.808
			prototypes	0.803	0.722	0.904	0.430	**0.834**	**0.593**	0.626	0.902	0.784	0.687	0.790	**0.667**	0.670	0.851	0.722	0.786
training-based methods																**training-based methods**			
ImageNet1k	ViT-B	CE	MSP	0.793	0.748	0.985	0.063	0.771	0.789	0.867	0.604	0.816	0.732	0.847	0.587	0.620	0.910	0.708	0.861
			ReAct	0.814	0.691	0.986	0.061	0.801	0.734	0.863	0.644	0.837	0.662	0.860	0.558	0.632	0.898	0.730	0.840
			KNN_norm	0.808	0.692	**0.992**	**0.043**	0.785	0.768	0.861	0.621	0.842	0.661	0.858	0.557	0.644	0.899	0.735	0.840
			ASH	0.488	0.959	0.466	0.958	0.511	0.949	0.496	0.947	0.482	0.949	0.489	0.952	0.480	0.957	0.464	0.961
			Flow	0.785	0.753	0.992	0.048	0.776	0.764	0.829	0.736	0.833	0.679	0.843	0.596	0.636	0.900	0.730	0.847
		DINO	MSP	0.778	0.788	0.973	0.097	0.762	0.803	0.862	0.647	0.807	0.754	0.836	0.618	0.609	0.915	0.695	0.875
			ReAct	0.799	0.741	0.977	0.090	0.785	0.772	0.853	0.728	0.830	0.697	0.849	0.606	0.627	0.904	0.717	0.862
			KNN_norm	0.786	0.725	0.986	0.073	0.759	0.801	0.824	0.746	0.825	0.695	0.836	0.608	0.638	0.897	0.712	0.863
			ASH	0.519	0.940	0.493	0.939	0.497	0.951	0.500	0.944	0.508	0.948	0.503	0.944	0.503	0.950	0.509	0.948
			Flow	0.764	0.773	0.974	0.127	0.710	0.845	0.729	0.858	0.799	0.763	0.795	0.673	0.617	0.909	0.694	0.888
ImageNet21k	BiT (Res101x3)	CE	MSP	0.760	0.805	0.945	0.188	0.760	0.823	0.861	0.662	0.802	0.773	0.826	0.650	0.609	0.916	0.692	0.882
			ReAct	0.787	0.750	0.941	0.181	0.783	0.784	0.859	0.652	0.828	0.681	0.840	0.609	0.622	0.909	0.716	0.865
			KNN_norm	0.797	0.730	0.986	0.077	0.789	0.765	0.857	0.676	0.842	0.653	0.854	0.580	0.640	0.900	0.721	0.843
			ASH	0.780	0.751	0.869	0.615	0.778	0.768	0.852	0.652	0.813	0.695	0.818	0.696	0.612	0.911	0.711	0.860
			Flow	0.757	0.763	0.977	0.111	0.753	0.812	0.803	0.849	0.826	0.703	0.823	0.648	0.643	0.897	0.721	0.845
LVD-142M	ViT-L	DINOv2	MSP	0.820	0.732	0.969	0.092	0.797	0.767	0.910	0.408	0.844	0.686	0.868	0.537	0.665	0.878	0.723	0.851
			ReAct	**0.842**	0.627	0.973	0.093	**0.826**	**0.681**	**0.919**	**0.322**	**0.867**	0.582	0.885	**0.461**	0.686	0.855	0.747	**0.831**
			KNN_norm	0.840	**0.616**	0.986	0.060	0.822	0.703	0.913	0.350	**0.867**	**0.574**	**0.886**	**0.461**	0.696	**0.847**	**0.748**	0.832
			ASH	0.508	0.944	0.491	0.913	0.481	0.959	0.504	0.947	0.522	0.934	0.501	0.939	0.511	0.942	0.514	0.939
			Flow	0.813	0.683	0.978	0.096	0.808	0.726	0.884	0.534	0.862	0.600	0.869	0.528	**0.691**	**0.847**	0.735	0.858

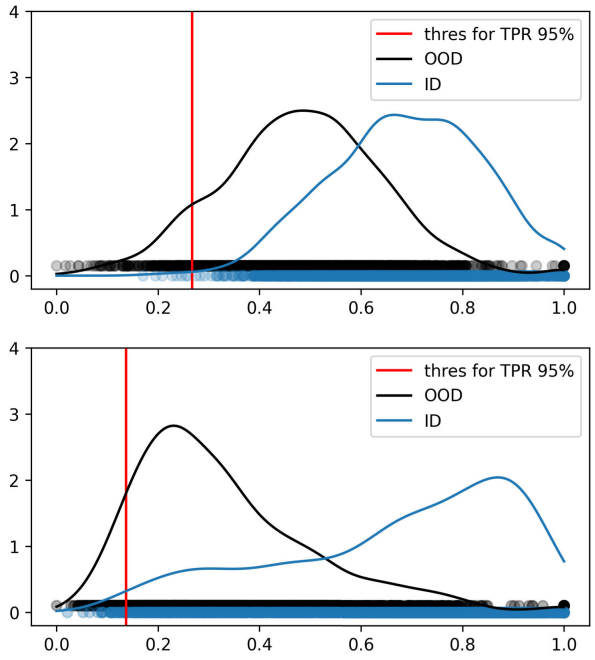

FIGURE 3. Distribution of normality scores for ID and OOD test samples on the All (MS) → Clipart (ST) set of the cross-domain track, using the pretrained DINOv2 model (top) and the corresponding fine-tuned one (bottom). Note how the range of score values, represented by the blue-ID and black-OOD horizontal bars over the x-axis, become similar in the bottom figure. Despite the peaks of the two distributions move apart, the separating threshold for TPR95 gets lower, causing a worse FPR95 score.

As mentioned in Sec III, a standard testbed for OOD detection is that presented in [7], covering only far OOD settings. We extended our analysis to it and collected the results in Tab. 5. Here the support set covers all the 1000 classes of ImageNet1k, but it is preferable to avoid fine-tuning, as already observed in [26].

If we compare what happens when the model pretraining is on ImageNet21k we can see that the training-based method Flow gets a higher AVG AUROC (0.886) than the not-fine-tuned BiT CE prototypes (0.873), while the corresponding FPR95 values show the opposite trend (BiT CE prototypes 0.484 < Flow 0.497). When the pretraining dataset is LVD-142M with objective DINOV2, the best fine-tuned results (Flow AUROC 0.904, ReAct FPR95 0.416) are worse than those of the training-free prototypes (AVG AUROC 0.925, and FPR95 0.316).

Interestingly, in this setting MCM produces top results when the OOD samples are from SUN and Places, but present much lower performance on iNaturalist and Texture. We claim that this behavior is connected to the use of language: it is relatively easy to position scene instances in a multimodal embedding created for object classes, while it is much more complicated to position real and synthetic patterns for which the language does not provide relevant hints. Indeed when we discard the language contribution by considering CLIP-KNN and CLIP-prototypes the results on SUN and Places decrease significantly, showing a larger drop than on iNaturalist and Texture for which the language was of minimal support.

When looking at these results it is also important to keep in mind that the computational cost of the training-based methods is significantly higher than that of training-free ones. Indeed, fine-tuning is a *full-network* training phase which

TABLE 5. Traning-free (top) vs Training-based (bottom) OOD detection results on the benchmark from [7]. ID dataset: ImageNet1k. The column titles indicate the OOD dataset. We use the bold font to highlight the best result per column and training setting.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	iNaturalist AUROC↑	iNaturalist FPR95↓	SUN AUROC↑	SUN FPR95↓	Places AUROC↑	Places FPR95↓	Texture AUROC↑	Texture FPR95↓	AVG AUROC↑	AVG FPR95↓
training-free methods													
ImageNet21k	BiT (Res101x3)	CE	KNN_norm	0.935	0.376	0.841	0.696	0.826	0.717	0.911	0.397	0.878	0.547
			Mahalanobis	0.961	0.220	0.822	0.690	0.758	0.810	**0.987**	**0.054**	0.882	0.443
			KNN	0.958	0.226	0.743	0.905	0.743	0.890	0.955	0.180	0.850	0.550
			prototypes	0.973	0.154	0.792	0.804	0.766	0.808	0.963	0.169	0.873	0.484
WebImageText	ViT-L	CLIP	MCM	0.950	0.283	**0.941**	**0.286**	**0.923**	**0.343**	0.831	0.625	0.911	0.384
			KNN	0.810	0.952	0.718	0.954	0.733	0.922	0.807	0.844	0.767	0.918
			prototypes	0.846	0.866	0.768	0.921	0.761	0.896	0.820	0.797	0.799	0.870
LVD-142M		DINOv2	KNN	0.922	0.333	0.808	0.742	0.828	0.696	0.859	0.561	0.854	0.583
			prototypes	**0.983**	**0.044**	0.916	0.372	0.894	0.449	0.906	0.400	**0.925**	**0.316**
training-based methods													
ImageNet21k	BiT (Res101x3)	CE	MSP	0.884	0.583	0.787	0.775	0.785	0.776	0.785	0.750	0.810	0.721
			ReAct	0.909	0.717	0.839	0.775	0.830	0.739	0.771	0.916	0.837	0.787
			KNN_norm	0.932	0.424	0.825	0.746	0.814	0.758	0.940	0.228	0.878	0.539
			ASH	0.936	0.453	0.852	0.673	0.834	0.677	0.843	0.748	0.866	0.637
			Flow	0.915	0.490	0.848	0.643	0.803	0.743	**0.977**	**0.114**	0.886	0.497
LVD-142M	ViT-L	DINOv2	MSP	0.931	0.408	0.831	0.687	0.825	0.694	0.833	0.657	0.855	0.611
			ReAct	**0.970**	**0.157**	0.882	0.505	0.867	0.538	0.885	0.464	0.901	**0.416**
			KNN_norm	0.950	0.310	0.864	0.623	0.864	0.601	0.839	0.633	0.879	0.542
			ASH	0.577	0.958	0.695	0.824	0.593	0.914	0.754	0.729	0.655	0.856
			Flow	0.960	0.240	**0.909**	**0.427**	**0.885**	**0.503**	0.864	0.602	**0.904**	0.443

may require a significant amount of time and has to be repeated for each OOD detection problem. This means that to obtain the numbers of a single row in Tab 4, fine-tuning-based methods perform 48 training sessions (considering the three random data orders for each experiment), while fine-tuning-free methods use the same fixed pretrained model for all the experiments.

VIII. A WISE WAY TO USE FINE-TUNING

As previously mentioned, the fine-tuning process may lead to *forgetting* some of the general knowledge acquired in the original model in favor of the new information collected from the fine-tuning data. This can be particularly problematic in case of domain shift between the distribution of the support data used for the fine-tuning process and the test one.

Recently, the authors of WiSE-FT [49] suggested to linearly combine the pretrained model with the fine-tuned one, showing that this greatly increases the robustness to distribution shifts of the latter while retaining its performance on the fine-tuning data distribution. Furthermore, the proposed approach eases the burden of the hyperparameters' choice (*e.g.* early stopping for fine-tuning), which is tricky and might heavily influence the final model's performance and robustness. Indeed, by simply tweaking the combination factor, it's possible to vary the contribution of the fine-tuning process to the final model without having to perform a new expensive training procedure. In light of these advantageous properties, we decided to test such a technique in our benchmark, both in the intra-domain and cross-domain tracks. While the original analysis of WiSE-FT specifically focused on CLIP-based models [6], and their zero-shot classification by leveraging the text encoder, we instead consider the vision-only based models that produced the best results in Tab. 4. Specifically, we choose the KNN_norm evaluator for our comparison. Regarding the models' combination, we adopt equal weights (0.5), following the authors' recommendation [49].

The obtained results are presented in Tab. 6. from which we can observe two different trends: on smaller pretraining datasets and models (*i.e.* ImagNet1k ViT-B CE and DINO) WiSE-FT obtains considerably worse performance compared to the sole fine-tuned networks, both in the intra-domain and in the cross-domain settings. Meanwhile, more complex models trained on larger datasets seem to benefit from the interpolation, with both ImageNet21k BiT-CE and LVD-142M ViT-L-DINOv2 showing a significant advantage in the cross-domain scenario (*e.g.* for MS→ST the WiSE-FT AUROC of LVD-142M ViT-L-DINOv2 is 0.786, while the fine-tuning result is 0.748), with also a slight performance improvement in the intra-domain one (WiSE-FT AUROC 0.891, fine-tuning AUROC 0.886). Thus, we can state that these models have enough capacity to benefit from such a simple ensemble, without canceling out either the general knowledge from the pretraining or the detailed one acquired during fine-tuning. At a higher level, a general conclusion is that vision-only based methods show unexpressed generalization potentialities that deserve dedicated investigations even without the integration with language.

IX. CLOSED SET RESULTS

Recently Vaze et al. [50] pointed out that the ability to detect OOD data is directly linked to a model's accuracy on ID samples. More precisely, for a model that can be used to perform both classification and detection of unknown classes, the performance on the two tasks will be roughly proportional. We are thus interested to understand whether this phenomenon also occurs with training-free methods. We assess the closed set accuracy (ACC) of the same methods included in Tab. 4 and report the average results in the

TABLE 6. Comparison between full fine-tuning, zero-shot models and WiSE-FT, using the KNN_OOD evaluator on the OODDB benchmark. The table presents in two vertical parts (left and right) the intra-domain and cross-domain track results. We use the bold font to highlight the best result per column.

Pretraining Dataset	Backbone	Pretraining Objective	Fine-tuning	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
ImageNet1k	ViT-B	CE	no	0.725	0.804	0.914	0.380	0.628	0.892	0.575	0.928	0.686	0.836	0.705	0.768	0.562	0.922	0.597	0.905
			yes	0.808	0.692	**0.992**	**0.043**	0.785	0.768	0.861	0.621	0.842	0.661	0.858	0.557	0.644	0.899	0.735	0.840
			WiSE-FT	0.783	0.742	0.982	0.091	0.711	0.821	0.644	0.897	0.801	0.713	0.784	0.653	0.625	0.901	0.711	0.839
		DINO	no	0.762	0.765	0.953	0.237	0.659	0.845	0.566	0.928	0.702	0.815	0.728	0.718	0.575	0.907	0.617	0.873
			yes	0.786	0.725	0.986	0.073	0.759	0.801	0.824	0.746	0.825	0.695	0.836	0.608	0.638	0.897	0.712	0.863
			WiSE-FT	0.780	0.728	0.983	0.091	0.719	0.811	0.709	0.898	0.797	0.723	0.797	0.650	0.624	0.895	0.703	0.842
ImageNet21k	BiT (Res101x3)	CE	no	0.832	0.635	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			yes	0.797	0.730	0.986	0.077	0.789	0.765	0.857	0.676	0.842	0.653	0.854	0.580	0.640	0.900	0.721	0.843
			WiSE-FT	0.830	0.622	0.989	0.051	0.813	0.699	0.803	0.769	0.853	0.601	0.858	0.548	0.671	0.870	0.746	0.815
LVD-142M	ViT-L	DINOv2	no	0.790	0.759	0.912	0.386	0.772	0.744	0.716	0.852	0.789	0.685	0.796	0.685	0.678	0.835	0.711	0.812
			yes	0.840	0.616	0.986	0.060	0.822	0.703	**0.913**	**0.350**	0.867	0.574	0.886	0.461	0.696	0.847	0.748	0.832
			WiSE-FT	**0.846**	**0.610**	0.986	0.069	**0.846**	**0.606**	0.905	0.384	**0.874**	**0.517**	**0.891**	**0.437**	**0.738**	**0.794**	**0.786**	**0.751**

TABLE 7. OOD detection (AUROC) and closed-set results (ACC) on our OODDB benchmark. The table is composed of three horizontal portions presenting the training-free, training-based, and WiSE-FT results. It is also divided vertically to show the results for the intra-domain (left) and cross-domain (right) tracks. We use the bold font to highlight the best result per column in each sub-part.

Pretraining Dataset	Backbone	Pretraining Objective	OOD Evaluator	Intra-domain		Cross-domain			
				AVG		SS → ST		MS → ST	
				AUROC↑	ACC↑	AUROC↑	ACC↑	AUROC↑	ACC↑
training-free methods						**training-free methods**			
ImageNet1k	ViT-B	CE	KNN_norm	0.706	0.704	0.562	0.251	0.597	0.389
			KNN	0.632	0.655	0.522	0.179	0.546	0.324
			prototypes	0.608	0.691	0.518	0.201	0.520	0.283
		DINO	KNN	0.730	0.710	0.575	0.254	0.616	0.404
			prototypes	0.715	0.689	0.558	0.206	0.573	0.310
ImageNet21k	BiT (Res101x3)	CE	KNN_norm	0.767	0.732	0.589	0.297	0.643	0.418
			KNN	0.757	0.725	0.581	0.282	0.636	0.414
			prototypes	0.720	0.730	0.550	0.244	0.567	0.361
WebImageText	ViT-L	CLIP	MCM	0.716	0.595	**0.817**	**0.719**	**0.817**	**0.719**
			KNN	**0.812**	**0.869**	0.648	0.548	0.695	0.642
			prototypes	0.798	0.862	0.636	0.490	0.677	0.608
LVD-142M		DINOv2	KNN	0.798	0.839	0.676	0.483	0.713	0.582
			prototypes	0.790	0.800	0.670	0.425	0.722	0.542
training-based methods						**training-based methods**			
ImageNet1k	ViT-B	CE	MSP	0.847	0.879	0.620	0.385	0.708	0.579
			ReAct	0.860	0.879	0.632	0.385	0.730	0.579
			KNN_norm	0.858	0.880	0.644	0.393	0.735	0.586
		DINO	MSP	0.836	0.862	0.609	0.347	0.695	0.546
			ReAct	0.849	0.862	0.627	0.347	0.717	0.546
			KNN_norm	0.836	0.863	0.638	0.360	0.712	0.549
ImageNet21k	BiT (Res101x3)	CE	MSP	0.826	0.871	0.609	0.363	0.692	0.566
			ReAct	0.840	0.871	0.622	0.363	0.716	0.566
			KNN_norm	0.854	0.873	0.640	0.374	0.721	0.563
LVD-142M	ViT-L	DINOv2	MSP	0.868	0.906	0.666	0.465	0.723	0.609
			ReAct	0.885	0.906	0.686	0.465	0.747	0.609
			KNN_norm	**0.886**	**0.909**	**0.696**	**0.482**	**0.748**	**0.614**
WiSE-FT						**WiSE-FT**			
ImageNet1k	ViT-B	CE	KNN_norm	0.784	0.830	0.625	0.363	0.711	0.559
		DINO		0.797	0.832	0.624	0.350	0.703	0.541
ImageNet21k	BiT	CE		0.858	0.886	0.671	0.432	0.746	0.601
LVD-142M	ViT-L	DINOv2		**0.891**	**0.924**	**0.738**	**0.566**	**0.786**	**0.669**

intra-domain and cross-domain settings, together with the corresponding AUROC, in the top part of Tab. 7. The results show that the proportionality between the ACC and AUROC is generally respected: the state-of-the-art OOD detection methods also show very good classification performances. Still, the cross-domain track appears particularly challenging also in terms of closed set. The table offers also a global overview of the results of the training-based methods (middle part of the table) as well as the WiSE-FT approach (bottom part of the table). In particular, the advantage of the latter over the former is evident even in closed set accuracy. Relying on the vision-language embedding of CLIP confirms itself as the best strategy across domains, but not in the intra-domain setting.

X. CONCLUSION

In this paper we analyzed the performance of training-free OOD detection approaches to draw a comprehensive picture of the state-of-the-art, in the wake of the appearance of foundation models in the computer vision field. We started by introducing OODDB, a novel large-scale benchmark designed to perform a fair and exhaustive evaluation of the OOD detection algorithms on a series of realistic deployment conditions. On this testbed, we analyzed training-free approaches considering both their application on traditional ImageNet-based pretrainings as well as the recently proposed foundation models CLIP [6] and DINOv2 [5]. We showed how the latter can be used to obtain state-of-the-art results on intra-domain scenarios, while the former's use of language information enables it to circumvent visual domain-shift problems and obtain excellent performance in the cross-domain setting. We then compared training-free methods with training-based ones, observing that fine-tuning enables the best intra-domain performance, even if training-free methods are rapidly catching up and already outperform training-based approaches in all other settings. Moreover, we discussed how to perform a simple combination of training-free and training-based methods with promising results. Overall, our analysis reveals the real performances unlocked by exploiting foundation models for OOD detection, and the newly proposed testbed puts under the spotlight some challenging settings that need more investigation, paving the way for future research.

APPENDIX A
IMPLEMENTATION DETAILS

We provide here some implementation details about the methods included in our analysis. Additional details can be found in our Github[3] page where we also release the code and the instructions to replicate all the results.

A. TRAINING-FREE METHODS

Where possible, we exploited the pretrained models provided by the original authors of each paper, or checkpoints provided by deep learning frameworks like PyTorch and HuggingFace. For CSI and supCSI [16] we performed the training of a ResNet101 on ImageNet1k using the code provided by the authors. For ReSeND [24] and Mahalanobis [20] we adapted

[3]https://github.com/FrancescoCappio/OODDetectionBench

the original inference code to integrate it into our benchmark. For KNN_norm [19] and MCM [26] we reimplemented the code from scratch, replicating the original results. The distance-based KNN and prototype methods were not presented as contributions of any specific publication, but are easy to implement: for the former, similarly to what was done for KNN_norm, we used faiss [51] for efficient similarity search. For the latter, we computed prototypes by per class *support* features averaging. This allows for reducing the number of comparisons significantly, which makes faiss unnecessary. For distance-based approaches, we generally rely on the Euclidean distance measure, the only exception being distances computed on the output of contrastive heads, for which the cosine distance is used for consistency with the contrastive learning objective.

We highlight that it is standard practice to discard the so-called *projector* before using self-supervised contrastive models for transfer learning [5], [33], [43]. The projector is also known as *contrastive head* and is usually an MLP appended at the end of the backbone, on top of which the learning objective is applied. Recently, Bordes et al. [52] pointed out that scraping the projector can be seen as a regularization technique, which they call Guillotine Regularization (GR), and is thus responsible for the improved generalization on downstream tasks. In our fine-tuning-free evaluation scenario, in which no training on downstream tasks is involved, the advantage of GR, over maintaining the contrastive projector head was not discussed in previous works so we ran a preliminary analysis that confirmed its advantage over keeping the projector. All the presented results were obtained with GR.

B. TRAINING-BASED METHODS

Testing fine-tuning-based methods require performing transfer learning, *i.e.* adopting pretrained models as a starting point for a whole network training on the *support* set of the task at hand. To perform fair comparisons, we adopted the vision-based pretrainings, which in the fine-tuning-free analysis obtained the top results, as a starting point for the fine-tuning: BiT [46] and DINOv2 [5]. We also included ViT-B-based CE and DINO [33] pretrainings, as we want to understand if the fine-tuning allows overcoming the performance gap among different pretrainings. As a general protocol, we performed 25 training epochs with a base learning rate of 10^{-4}, batch size 64, Adam optimizer with weight decay 10^{-5}. We adopted a learning rate scheduling protocol composed of 5 warmup epochs, and 20 cosine annealing ones. For large trainings (e.g. those on ImageNet1k) we reduced the base learning rate to 10^{-5} and we performed distributed training, linearly scaling the batch size and the learning rate.

For what concerns ReAct [11], which requires computing an activation threshold necessary to increase known-unknown separability, we adopt the original authors' proposed approach of keeping 90% of ID activations, but we apply this split on *training* activations, as most benchmarks do not include an ID validation set. For ASH [4] we apply the original author's code for activations shaping.

As for the flow-based approach, we used the same architecture proposed in [53], stacking 8 coupling blocks on top of the feature extractor of the base network. During the training procedure, we jointly updated the backbone and the classification head following a cross-entropy loss, while separately optimizing the flow module by maximizing the log-likelihood. When training on ImageNet1k, we reduced the learning rate for the flow module by a factor of 10 to avoid instabilities.

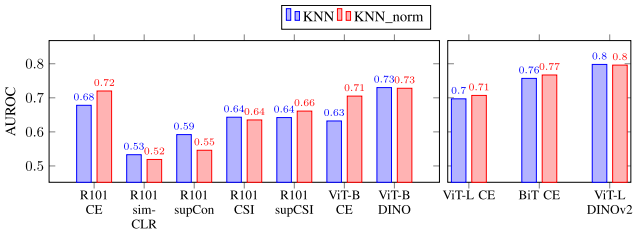

FIGURE 4. Training-free OOD detection results on the OODDB benchmark, intra-domain setting. KNN vs KNN_norm performance for ImageNet1k-based pretrainings (left) and larger ones (right).

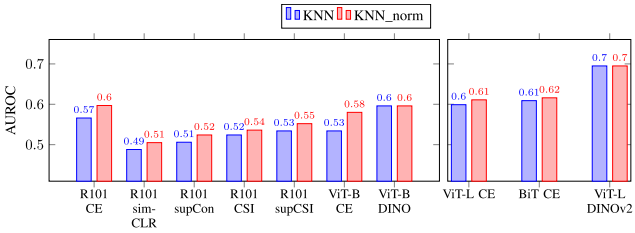

FIGURE 5. Training-free OOD detection results on the OODDB benchmark, cross-domain setting. KNN vs KNN_norm performance for ImageNet1k-based pretrainings (left) and larger ones (right).

APPENDIX B
ADDITIONAL ANALYSES

A. KNN AND KNN_NORM

As already discussed in the main paper, it might be beneficial to normalize the features provided by a model pretrained with the CE classification objective. We provide here empirical evidence for this claim over all the settings of our OODDB benchmark. Specifically, Fig. 4 shows results for the intra-domain track, and Fig. 5 for the cross-domain track. In the first case, KNN_norm has a visible advantage over KNN only for CE pretraining. In the second case, the normalization provides a slight general advantage but when using CE pretraining the gain is more evident.

We also highlight that for all our experiments we set $K = 1$. In a preliminary phase we evaluated higher values of K by computing the normality score based on the distance from the K-th nearest support neighbor. However, the results were always lower than those obtained with $K = 1$.

TABLE 8. Training-free (top) vs Training-based (bottom) OOD detection results on the OODDB benchmark when using k-means clustering to summarize the support set, followed by KNN_norm. We create $k = 10$ clusters per class and use only the respective centroids to represent the class. The left and right parts of the table present the intra-domain and cross-domain results. We do not use bold font here are the goal is not to highlight the top overall results but rather to compare the results when using k-means or not (-).

Pretraining Dataset	Backbone	Pretraining Objective	k-means	Intra-domain track												Cross-domain track			
				Textures		PatterNet		SUN		Stanford Cars		DomainNet Intra		AVG		SS → ST		MS → ST	
				AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓	AUROC↑	FPR95↓
training-free methods																**training-free methods**			
ImageNet1k	ViT-B	CE	-	0.725	0.804	0.914	0.380	0.628	0.892	0.575	0.928	0.686	0.836	0.705	0.768	0.562	0.922	0.597	0.905
			10	0.746	0.786	0.911	0.363	0.661	0.843	0.568	0.931	0.689	0.805	0.715	0.745	0.564	0.925	0.611	0.884
		DINO	-	0.762	0.765	0.953	0.237	0.659	0.845	0.566	0.928	0.702	0.815	0.728	0.718	0.575	0.907	0.617	0.873
			10	0.781	0.713	0.947	0.229	0.679	0.811	0.555	0.934	0.698	0.800	0.732	0.697	0.571	0.913	0.619	0.866
ImageNet21k	BiT (Res101x3)	CE	-	0.832	0.635	0.938	0.303	0.760	0.775	0.582	0.923	0.724	0.766	0.767	0.680	0.589	0.899	0.643	0.845
			10	0.841	0.610	0.927	0.348	0.786	0.713	0.574	0.922	0.719	0.755	0.769	0.670	0.577	0.920	0.640	0.870
LVD-142M	ViT-L	DINOv2	-	0.790	0.759	0.912	0.386	0.772	0.744	0.716	0.852	0.789	0.685	0.796	0.685	0.678	0.835	0.711	0.812
			10	0.823	0.681	0.920	0.338	0.813	0.624	0.709	0.854	0.807	0.632	0.815	0.626	0.685	0.848	0.750	0.759
training-based methods																**training-based methods**			
ImageNet1k	ViT-B	CE	-	0.808	0.692	0.992	0.043	0.785	0.768	0.861	0.621	0.842	0.661	0.858	0.557	0.644	0.899	0.735	0.840
			10	0.810	0.680	0.990	0.048	0.789	0.753	0.865	0.627	0.842	0.654	0.859	0.552	0.637	0.904	0.730	0.846
		DINO	-	0.786	0.725	0.986	0.073	0.759	0.801	0.824	0.746	0.825	0.695	0.836	0.608	0.638	0.897	0.712	0.863
			10	0.796	0.692	0.984	0.072	0.768	0.785	0.831	0.748	0.834	0.667	0.842	0.593	0.635	0.901	0.716	0.853
ImageNet21k	BiT (Res101x3)	CE	-	0.797	0.730	0.986	0.077	0.789	0.765	0.857	0.676	0.842	0.653	0.854	0.580	0.640	0.900	0.721	0.843
			10	0.799	0.694	0.983	0.088	0.788	0.759	0.858	0.680	0.839	0.660	0.854	0.576	0.633	0.906	0.711	0.863
LVD-142M	ViT-L	DINOv2	-	0.840	0.616	0.986	0.060	0.822	0.703	0.913	0.350	0.867	0.574	0.886	0.461	0.696	0.847	0.748	0.832
			10	0.844	0.587	0.984	0.067	0.826	0.684	0.916	0.343	0.872	0.549	0.889	0.446	0.691	0.858	0.748	0.825

B. K-MEANS

Despite the good results it can achieve, the greatest limitation of the KNN evaluator remains its computational cost, as it requires to perform for each test instance sample as many comparisons as the number of support samples $|\mathcal{S}|$ which is generally much higher than the number of support classes $\mathcal{C}$.

To overcome it, we tried running the k-means algorithm on the support set features for each known class, leveraging the obtained centroids rather than the whole feature set to represent the *normal* data. In this way, the complexity of the evaluation goes from $O(|\mathcal{S}|)$ to $O(\mathcal{C})$, in line with most of the other methods.

We used $k = 10$ and the results in Tab. 8 show how k-means doesn't affect the models' performance on average, with results being slightly better or worse depending on the considered scenario. Thus, we argue that multiple centroids can effectively summarize the support set information without introducing a significant performance drop, unlike the prototypes-based evaluation which leverages a single vector for each known category (see Tab. 2). The only notable exception is DINOv2, which experiences a small but consistent performance boost across all the test settings, again confirming that the compact class clusters obtained with this representation are highly beneficial for OOD detection.

REFERENCES

[1] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proc. ICLR*, 2017. [Online]. Available: https://openreview.net/forum?id=Hkg4TI9xl

[2] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proc. ICLR*, 2018. [Online]. Available: https://openreview.net/forum?id=H1VGkIxRZ

[3] W. Liu, X. Wang, J. Owens, and Y. Li, "Energy-based out-of-distribution detection," in *Proc. NeurIPS*, 2020, p. 21464.

[4] A. Djurisic, N. Bozanic, A. Ashok, and R. Liu, "Extremely simple activation shaping for out-of-distribution detection," in *Proc. ICLR*, 2023. [Online]. Available: https://openreview.net/forum?id=ndYXTEL6cZz

[5] M. Oquab et al., "DINOV2: Learning robust visual features without supervision," *Trans. Mach. Learn. Res.*, Jul. 2024.

[6] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proc. ICML*, 2021, pp. 8748–8763.

[7] R. Huang and Y. Li, "MOS: Towards scaling out-of-distribution detection for large semantic space," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8706–8715.

[8] G. Csurka, T. Hospedales, M. Salzmann, and T. Tommasi, *Visual Domain Adaptation in the Deep Learning Era*. San Rafael, CA, USA: Morgan & Claypool, 2022.

[9] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 427–436.

[10] H. Wei, R. Xie, H. Cheng, L. Feng, B. An, and Y. Li, "Mitigating neural network overconfidence with logit normalization," in *Proc. ICML*, 2022, pp. 23631–23644.

[11] Y. Sun, C. Guo, and Y. Li, "React: Out-of-distribution detection with rectified activations," in *Proc. NeurIPS*, 2021, pp. 144–157.

[12] R. Huang, A. Geng, and Y. Li, "On the importance of gradients for detecting distributional shifts in the wild," in *Proc. NeurIPS*, 2021, pp. 677–689.

[13] K. H. Kim, S. Shim, Y. Lim, J. Jeon, J. Choi, B. Kim, and A. S. Yoon, "RaPP: Novelty detection with reconstruction along projection pathway," in *Proc. ICLR*, 2020. [Online]. Available: https://openreview.net/forum?id=HkgeGeBYDB

[14] D. Abati, A. Porrello, S. Calderara, and R. Cucchiara, "Latent space autoregression for novelty detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 481–490.

[15] H. Zhang, A. Li, J. Guo, and Y. Guo, "Hybrid models for open set recognition," in *Proc. ECCV*, 2020, pp. 102–117.

[16] J. Tack, S. Mo, J. Jeong, and J. Shin, "Csi: Novelty detection via contrastive learning on distributionally shifted instances," in *Proc. NeurIPS*, 2020, pp. 11839–11852.

[17] V. Sehwag, M. Chiang, and P. Mittal, "Ssd: A unified framework for self-supervised outlier detection," in *Proc. ICLR*, 2021. [Online]. Available: https://openreview.net/forum?id=v5gjXpmR8J

[18] J. Li, P. Chen, Z. He, S. Yu, S. Liu, and J. Jia, "Rethinking out-of-distribution (OOD) detection: Masked image modeling is all you need," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 1–20.

[19] Z. Bukhsh and A. Saeed, "On out-of-distribution detection for audio with deep nearest neighbors," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–13.

[20] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proc. NeurIPS*, 2018, pp. 7167–7177.

[21] Y. LeCun, C. Cortes, and C. Burges. (2010). *Mnist Hand-Written Digit Database*. [Online]. Available: http://yann.lecun.com/exdb/mnist

[22] A. Krizhevsky. (2009). *Learning Multiple Layers of Features From Tiny Images*. [Online]. Available: https://www.cs.toronto.edu/

[23] A. Kumar, A. Raghunathan, R. M. Jones, T. Ma, and P. Liang, "Fine-tuning can distort pretrained features and underperform out-of-distribution," in *Proc. ICLR*, 2022, pp. 1–14.

[24] F. Cappio Borlino, S. Bucci, and T. Tommasi, "Semantic novelty detection via relational reasoning," in *Proc. ECCV*, 2022, pp. 1–30.

[25] L. L. Lu, G. D'Ascenzi, F. Cappio Borlino, and T. Tommasi, "Large class separation is not what you need for relational reasoning-based ood detection," in *Proc. ICIAP*, 2023, pp. 1–19.
[26] Y. Ming, Z. Cai, J. Gu, Y. Sun, W. Li, and Y. Li, "Delving into out-of-distribution detection with vision-language representations," in *Proc. NeurIPS*, 2022, pp. 35087–35102.
[27] H. Wang, Y. Li, H. Yao, and X. Li, "CLIPN for zero-shot OOD detection: Teaching CLIP to say no," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 20–29.
[28] S. Esmaeilpour, B. Liu, E. Robertson, and L. Shu, "Zero-shot out-of-distribution detection based on the pre-trained model clip," in *Proc. AAAI*, 2021, pp. 1–11.
[29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. ICLR*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy
[31] S. Fort, J. Ren, and B. Lakshminarayanan, "Exploring the limits of out-of-distribution detection," in *Proc. NeurIPS*, 2021, pp. 7068–7081.
[32] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. Le, Y.-H. Sung, Z. Li, and T. Duerig, "Scaling up visual and vision-language representation learning with noisy text supervision," in *Proc. ICML*, 2021, pp. 4904–4916.
[33] M. Caron, H. Touvron, I. Misra, H. Jegou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9630–9640.
[34] J. Yang, H. Wang, L. Feng, X. Yan, H. Zheng, W. Zhang, and Z. Liu, "Semantically coherent Out-of-Distribution detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 8281–8289.
[35] J. Zhang, J. Yang, P. Wang, H. Wang, Y. Lin, H. Zhang, Y. Sun, X. Du, K. Zhou, W. Zhang, Y. Li, Z. Liu, Y. Chen, and H. Li, "OpenOOD V1.5: Enhanced benchmark for out-of-distribution detection," in *Proc. NeurIPS Workshop Distrib. Shifts, New Frontiers Found. Models*, 2023. [Online]. Available: https://openreview.net/forum?id=vTapqwaTSi&referrer=%5Bthe%20profile%20of%20Yixuan%20Li%5D(%2Fprofile%3Fid%3D~Yixuan_Li1)
[36] J. Yang, K. Zhou, and Z. Liu, "Full-spectrum out-of-distribution detection," *Int. J. Comput. Vis.*, vol. 131, no. 10, pp. 2607–2622, Oct. 2023.
[37] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi, "Describing textures in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 3606–3613.
[38] W. Zhou, S. Newsam, C. Li, and Z. Shao, "PatternNet: A benchmark dataset for performance evaluation of remote sensing image retrieval," *ISPRS J. Photogramm. Remote Sens.*, vol. 145, pp. 197–209, Nov. 2018.
[39] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "SUN database: Large-scale scene recognition from abbey to zoo," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1–17.
[40] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3D object representations for fine-grained categorization," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2013, pp. 554–561.
[41] X. Peng, Q. Bai, X. Xia, Z. Huang, K. Saenko, and B. Wang, "Moment matching for multi-source domain adaptation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1406–1415.
[42] S. Bird, E. Klein, and E. Loper, *Natural Language Processing With Python: Analyzing Text With the Natural Language Toolkit*. Sebastopol, CA, USA: O'Reilly Media, Inc., 2009.
[43] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. ICML*, 2020, pp. 1597–1607.
[44] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," in *Proc. NeurIPS*, 2020, pp. 18661–18673.
[45] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
[46] A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby, "Big transfer (bit): General visual representation learning," in *Proc. ECCV*, 2020, pp. 491–507.
[47] Y. Wu and K. He, "Group normalization," *Int. J. Comput. Vis.*, vol. 128, pp. 742–755, 2020.
[48] S. Qiao, H. Wang, C. Liu, W. Shen, and A. Yuille, "Micro-batch training with batch-channel normalization and weight standardization," 2019, *arXiv:1903.10520*.
[49] M. Wortsman, G. Ilharco, J. W. Kim, M. Li, S. Kornblith, R. Roelofs, R. G. Lopes, H. Hajishirzi, A. Farhadi, H. Namkoong, and L. Schmidt, "Robust fine-tuning of zero-shot models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 7949–7961.
[50] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Open-set recognition: A good closed-set classifier is all you need," in *Proc. ICLR*, 2022. [Online]. Available: https://openreview.net/forum?id=5hLP5JY9S2d
[51] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with GPUs," *IEEE Trans. Big Data*, vol. 7, no. 3, pp. 535–547, Jul. 2021.
[52] F. Bordes, R. Balestriero, Q. Garrido, A. Bardes, and P. Vincent, "Guillotine regularization: Why removing layers is needed to improve generalization in self-supervised learning," *Trans. Mach. Learn. Res.*, 2023.
[53] M. Rudolph, B. Wandt, and B. Rosenhahn, "Same same but DifferNet: Semi-supervised defect detection with normalizing flows," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2021, pp. 1906–1915.

FRANCESCO CAPPIO BORLINO is currently pursuing the Ph.D. degree with Politecnico di Torino. He has published multiple papers at conferences and journals specialized in machine learning, including ECCV, IROS, WACV, CVIU, and NeurIPS. His research interests include deep learning techniques applied to computer vision, with a particular focus on out-of-distribution detection, domain adaptation, and domain generalization. He served as a reviewer at several other events, being named one of the Outstanding Reviewers at CVPR 2023.

LORENZO LU received the B.S. and M.S. degrees in computer engineering from Politecnico di Torino. He is currently a Research Assistant working on deep learning models applied to computer vision. His work mainly focuses on out-of-distribution detection and domain generalization.

TATIANA TOMMASI received the Ph.D. degree from EPFL, Lausanne, Switzerland, in 2013. She is currently an Associate Professor with the Department of Control and Computer Engineering, Politecnico di Torino, Italy, and the Director of the ELLIS Unit, Turin. She spent postdoctoral periods in Belgium and USA, before covering the role of an Assistant Professor with Sapienza University, Rome, Italy. She has published more than 50 papers at top conferences and journals in machine learning and computer vision. Her expertise lies in the development of theoretically grounded algorithms for automatic learning from images, particularly within the realms of robotics, medical applications, and human-machine interaction. She is a pioneer in the field of transfer learning in computer vision and possesses extensive experience in areas, such as domain adaptation, generalization, multimodal learning, and open-set learning. She is an Associate Editor of IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE and IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING.

Open Access funding provided by 'Politecnico di Torino' within the CRUI CARE Agreement