Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 246 (2024) 216–225

Procedia Computer Science

www.elsevier.com/locate/procedia

28th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES 2024)

DIFAIR: Towards Learning Differentiated Image Representations

Quentin Christoffel[a], Aline Deruyver[a], Ali Ayadi[a], Anne Jeannin-Girardon[a]

[a]*Université de Strasbourg - ICube UMR7357, 1 rue Eugène Boeckel, 67000 Strasbourg, FRANCE*

Abstract

Neural network classifiers are generally trained to differentiate between the same classes during training and testing. In order to prevent incorrect predictions, when an input image contains a class that was not part of the training set, it should be detected. The process of detection of "unknown" classes is called *Open-Set Recognition* (OSR). Given that a neural network extracts a representation (a feature vector) describing an image, its capacity to detect the presence of a class in an image, through the recognition of specific features, should also imply the ability to detect the absence of a "known" class, through the absence of those features in the representation. In this article, we present DIFAIR (*DIFferentiAted Image Representations*), a novel approach aimed at learning a representation exhibiting: (i) class separability, through predefined class positions in the representation space; (ii) the extraction of distinct features, which remain inactive if not present in the image; and (iii) semantic meaning when comparing representations. We present a distance-based loss function to optimize a network, in a supervised way, to obtain the proposed representation. The evaluation of DIFAIR in OSR shows a performance close to a similar, distance-based, method, but below state-of-the-art methods. Finally, we visually inspect learned representations to identify the limits of our approach and present directions for future improvement.
Code and more figures are available at https://github.com/qchristoffel/DIFAIR.

Keywords: Open-Set Recognition; Neural Networks; Representation Learning; Unknown data

1. Introduction

Image classification is a crucial task in vision recognition, aimed at characterizing images and categorizing them into specific classes [30]. Recent advancements in this field have been propelled by the emergence of deep neural networks. Convolutional Neural Networks (CNNs), including VGG [27] and ResNets [11], have been used extensively and inspired many variants. Moreover, the adaptation of the transformer architecture [28] for image classification by [7] has achieved state-of-the-art performance. These models are trained using the cross-entropy loss function and

E-mail address: q.christoffel@unistra.fr

1877-0509
10.1016/j.procs.2024.09.268

employ the softmax function in the final layer to estimate class probabilities. These networks make predictions even when presented with images featuring classes absent from the training data, referred to as "unknown classes" in this paper. In this scenario, models are assumed to predict given the *same classes* during training and testing phases, a concept termed *closed-set classification* [26].

However, using a model designed for closed-set scenarios is inadequate for real-world applications, where the model operates in dynamic environments and encounters classes that might change. Furthermore, real-world applications often involve a significantly larger number of classes than those typically used for model training. It is unreasonable to expect a dataset that encompasses every conceivable object, both existing and future ones: hence the need for robust deep neural networks capable of handling unknown data. To address this issue, Scheirer et al. [26] introduced the notion of *Open-Set Recognition* (OSR), where unknown classes are presented to the model during the test phase with the dual objective of detecting them while still being able to classify known classes.

As discussed by [6], we hypothesize that if a model is capable of recognizing the presence of a class in an image, it should somehow be able to discern its absence. More precisely, if a model can identify features specific to a particular class, then if that class is not present in the image, the features learned by the model for this class should remain inactive, approaching values close to 0. Empirical experiments have supported this hypothesis, demonstrating that the norm of the feature vector when dealing with unknown classes is often lower than that for known classes [4, 29]. It is also possible that unknown classes share certain features with known classes. In such cases, the model may misclassify instances, leaving us without a clear interpretation of its decision. When examining the feature vectors learned by most neural networks, it becomes evident that we cannot readily discern whether an image belongs to known classes, primarily because we lack information about which features describe a particular class. Furthermore, the challenge is compounded by the fact that features can be shared by multiple classes due to the distributed representations inherent in neural networks [12], making the interpretation even more challenging.

Based on these observations, we propose three objectives for our approach: (i) ensuring that features absent from the image remain "disabled", by approaching values close to 0; (ii) enabling the model to extract distinct features that describe a determined class while (iii) retaining semantic meaning in the representation. Thus, the main goal of our work is to attain a representation where classes are semantically well-separated from each other in order to perform classification. This representation should contain class-associated features without distributed representations. We called our approach DIFAIR, standing for *DIFferentiAted Image Representations*.

To achieve these objectives, we aim to control the representation space by defining an association between specific dimensions and classes. Each dimension associated with a class should represent *different features* extracted for that class. This is particularly valuable for OSR, as we posit that an unknown image will activate features from multiple classes or none at all, facilitating its detection.

In this paper, our contributions regarding DIFAIR comprise three main aspects: (i) the introduction of a loss function to optimize the network's learned representation, aligning it with the constraints specified above; (ii) the utilization of OSR tasks for qualitative evaluation of the learned representation, since our loss function organizes the distribution of known instances in the latent space; and (iii) the visualization of the learned representations, showcasing how they provide valuable insights into the model's behavior compared to representations derived from models trained with cross-entropy.

2. Related work

Open-Set Recognition. Scheirer et al. [26] introduced the concept of OSR, a more realistic classification scenario than traditional closed-set recognition. OSR expects classifiers to not only classify instances belonging to known classes, but also detect unknown classes at test time. This stands in contrast to closed-set recognition where a model is evaluated only on classes that were seen during training. Bendale and Boult [1] pioneered OSR in deep learning with the OpenMax approach. This method measures the distance between a new instance's point and the mean activation points of known classes in the logit space. Utilizing Extreme Value Theory [14] on these distances, they calibrate network predictions to estimate the probability of belonging to an unknown class.

Various approaches, including the use of Generative Adversarial Networks (GANs) [8, 23, 4, 21], have been explored for OSR. These methods generate unknown data and train networks to make specific predictions when confronted with it. Real data examples from external datasets can also be used as unknown data [3]. OSR strategies may involve learning prototype points in the latent space to represent known or unknown classes [4, 19], with the distance

to these prototypes used for detection. Other methods optimize the representation space to have better inter-class separation and intra-class compactness [10, 20, 2]. Contrastive learning, combined with unknown examples generated using Mixup [32], has also been leveraged for pre-training a feature encoder space to separate known and unknown classes [31].

To our knowledge, current state-of-the-art OSR methods include ARPL+CS [4], OpenHybrid [33], DCHS [3], and ConOSR [31]. Vaze et al. [29] found a correlation between closed-set accuracy and open-set performance. Their study suggests that a baseline network trained with cross-entropy can match or exceed OSR state-of-the-art methods by using data augmentation, increased epochs, and specific learning rate scheduling. Unlike other OSR methods presented above, Vaze et al. [29]'s baseline is exclusively trained on known classes.

Class Anchor Clustering. We examine Class Anchor Clustering (CAC) by Miller et al. [20], an OSR method similar to our approach. CAC organizes the representation space using anchors, akin to our method. Those class anchors are in the logit space and are represented as scaled one-hot vectors. The network optimization aims to produce logits proximal to class anchors based on Euclidean distance. The proposed loss function includes two components: (i) the anchor loss, minimizing the distance between an instance's representation and its anchor, and (ii) a loss term maximizing the difference between the instance's distance to its true anchor and its distance to all other anchors.

During testing, distances $\boldsymbol{d}$ between predicted logits and each class anchor are measured to compute a rejection score $\gamma = \boldsymbol{d} \circ (1 - \text{softmin}(\boldsymbol{d}))$, indicating uncertainty about class membership. If $\min(\gamma)$ exceeds a predetermined threshold, the prediction is rejected as *unknown*; otherwise, it is accepted for the class with the lowest rejection score. According to the authors, anchored representations lack semantic meaning due to uniformly distributed anchors. They suggest that incorporating semantic features could enhance OSR performance. To address this, our approach, DIFAIR, aims to introduce additional semantic meaning into the learned representations.

3. Methods

3.1. OSR formalization and notations

In this work, we adopt the formalization of OSR proposed by Vaze et al. [29]. In *closed-set recognition*, a classifier is trained on a dataset $\mathbb{D}_{\text{train}} = \{(\boldsymbol{X}_i, y_i)\}_{i=1,\dots,n_{\text{train}}} \subset \mathbb{X} \times \mathbb{C}$ with $\mathbb{X}$ the input space (images in our case), and $\mathbb{C}$ is the set of "known" classes. We note $n_c = |\mathbb{C}|$ the number of known classes. The classifier is evaluated on a dataset $\mathbb{D}_{\text{test-closed}} = \{(\boldsymbol{X}_j, y_j)\}_{j=1,\dots,n_{\text{test}}} \subset \mathbb{X} \times \mathbb{C}$, where all test inputs belong to known classes. OSR considers a scenario where inputs from any classes could be given to the classifier. In addition to doing classification, OSR aims to detect inputs belonging to "unknown" classes. In this situation, the classifier is trained on $\mathbb{D}_{\text{train}}$ and evaluated on $\mathbb{D}_{\text{test-open}} = \{(\boldsymbol{X}_j, y_j)\}_{j=1,\dots,n'_{\text{test}}} \subset \mathbb{X} \times (\mathbb{C} \cup \mathbb{U})$, where $\mathbb{U}$ represents the infinite set of all unknown classes.

In the following, we refer to different elements of a neural network classifier formalized as $C(\boldsymbol{X}) = \Theta \circ \phi(\boldsymbol{X})$, where Θ is the classification head (generally a fully connected network), ϕ is the feature extractor (generally convolutional layers) and $\boldsymbol{X}$ is an input image. We note $\boldsymbol{z} = \phi(\boldsymbol{X})$ the *representation* learned by the feature extractor ϕ and $\hat{\boldsymbol{y}} = \Theta(\boldsymbol{z})$ the *logits* output of the classifier, i.e. no activation function is applied on $\hat{\boldsymbol{y}}$.

In the open-set setting, given an input $\boldsymbol{X}$, this classifier is expected to return a score for each known class *and* a score $S(y \in \mathbb{C}|\boldsymbol{X})$ used to detect whether image $\boldsymbol{X}$ contains a known class or not. An example of such a score is $S(y \in \mathbb{C}|\boldsymbol{X}) = \max \text{softmax}(\hat{\boldsymbol{y}})$, which is the highest probability output of a network trained with cross-entropy loss. Instances scoring above a determined threshold will be considered as *known* by the network and instances scoring below as *unknown*. Varying the threshold on the score allows us to compute AUROC for the task of unknown detection.

3.2. DIFAIR: DIFferentiAted Image Representations

DIFAIR aims to control the representation space $\boldsymbol{z}$ learned by a model. It ensures that instances of the same class activate *common predefined features* while avoiding the activation of features associated with other classes. This approach facilitates learning differentiated class representations that can provide insights into the model's behavior. These representations allow visual assessment of which class is most recognized and which feature of which class is activated. Moreover, the representations can be used to detect unknown classes, as the absence of known classes in the input data should be reflected in the representation.

3.2.1. Proposed representation

In the following, we employ the term *characteristics* to refer to inherent attributes of the image, while *features* will denote individual dimensions of the neural network representation. A feature may become *active* when the feature extractor ϕ detects characteristics in the image and expresses them on the corresponding feature dimension.

As in CAC [20], the DIFAIR approach initially defines class anchors for each class as *fixed points* in space around which the model will learn to represent class instances. Key distinctions that set DIFAIR apart are: (i) DIFAIR anchors in the *representation space* $\boldsymbol{z}$, unlike CAC, which anchors in the logit space $\hat{\boldsymbol{y}}$; (ii) DIFAIR *allocates and associates* multiple dimensions per class in the representation space. In contrast, CAC used only one dimension in the logit space to represent each class. Representing a class on more than one dimension is advantageous as each dimension can express the presence or absence of different characteristics from each class. We hypothesize that the association of each dimension of the representation with a specific class can yield more meaningful features that provide insights into predictions (as shown in Section 5). Instead of having a fully connected layer exploiting the representation to make a prediction, we aim to force the extracted features to be representative of a class so that the prediction could be deduced by examining the representation itself.

In the proposed approach, the anchors $\mathcal{A}^{(c)}$, with c the class index, are characterized by two hyperparameters: the number of dimensions, $\mathcal{N}$, allocated per class, and α the desired activation value on allocated dimensions. Consider a classification problem with three classes and $\mathcal{N} = 2$ dimensions per class. The three class anchors are set according to coordinates in Eq. 1. Here, we consider *non-zero dimensions to be associated with the respective class of the anchor*, because the objective will be to activate these specific dimensions when instances of the class are presented.

$$\mathcal{A}^{(1)} = (\alpha, \alpha, 0, 0, 0, 0) \quad \mathcal{A}^{(2)} = (0, 0, \alpha, \alpha, 0, 0) \quad \mathcal{A}^{(3)} = (0, 0, 0, 0, \alpha, \alpha) \tag{1}$$

For the purpose of this article, we consider the allocation of the same number of features per class and maintain a constant value α across all anchors. We leave as future work the exploration of utilizing anchors with varying parameters. With such representations, all anchors are equidistant and share the same norm, as in CAC. However, Miller et al. [20] noted that in this situation, extracted features lack semantic meaning. To allow our representations to bear semantic meaning, we allocate a hypersphere around each anchor. This hypersphere defines a space in which instances of a class can be represented. The radius of the hypersphere, denoted r, is an additional hyperparameter. More details on the semantics brought by the usage of a hypersphere are provided in Section 3.2.2.

To make a prediction using this representation, the Euclidean distances $\boldsymbol{d}$ are measured between the representation of a given instance and each anchor : $\boldsymbol{d} = \text{Eucl}(\boldsymbol{z}, \mathcal{A}) = (\|\boldsymbol{z}, \mathcal{A}^0\|_2, \ldots, \|\boldsymbol{z}, \mathcal{A}^{n_c}\|_2)$. The predicted class $\hat{y}$ is determined by $\hat{y} = \arg\min \boldsymbol{d}$. The measured distances are used to compute the OSR score $S(y \in \mathbb{C}|\boldsymbol{X}) = \min \boldsymbol{d}$, meaning that if an instance is not within a determined radius of an anchor, the prediction is rejected. Unlike CAC, anchors are not adjusted to the mean class representation at the end of learning, since shifting anchors would make it harder to interpret features to class association.

3.2.2. DIFAIR specificities

In this section, we highlight the key assumptions and hypotheses that underlie this work and explain the applicability of DIFAIR to OSR. Experiments and visualizations, presented in Sections 4 and 5, aim to test these hypotheses.

Semantically meaningful representations. We consider that semantic similarity in a representation space is equivalent to having a proximity between instances' representations of similar classes. In a setup like CAC, where class anchors are equidistant and instances are optimized to be clustered close to the anchors, there will be no such proximity between instances representations, thus no semantics in the representation. In contrast, DIFAIR's use of hyperspheres around anchors allows data points to be located anywhere within the hypersphere. This is promoting a more flexible representation that doesn't confine points strictly at the anchor. From there, it is possible (but not certain) for a point representing a cat instance to be located in the hypersphere on the dog's anchor side and there would therefore be a proximity with dog class for this instance. In order to express this proximity (and thus a semantic meaning), some features of the dog class should be activated in the cat instance representation. The activation of features from other classes is possible thanks to the usage of the threshold-based Euclidean distance. For those reasons, we conjecture that the DIFAIR-learned representations can retain a semantic meaning, enhancing the richness of the representations.

Open-Set Recognition. Our method aims at clustering data around well-separated anchors in space, and extracts different features on different dimensions, which are activated when characteristics are actually present in the images. From there, an image containing an unknown class might activate features of different known classes, in the situation where it shares some characteristics with known classes. No feature at all could be activated if the image lacks the learned features. However, it should not activate *all* features from one known class, otherwise the class would not be unknown. Based on these considerations, we hypothesized that the Euclidean distance is a suitable OSR score for our problem. Given a new instance, the more features from a single class are activated, the closer the instance will be to this class anchor. If features from different classes are activated, the instance representation might not be close to any anchor. In this situation, it will be detected as unknown thanks to a chosen threshold on the distance. For these reasons, we believe that our method can be used in OSR.

3.3. Implementation

3.3.1. Architecture modification

DIFAIR can fit any convolutional architecture, as long as the classification head Θ is removed and a convolutional layer which will output $\mathcal{N} \times n_c$ filters is added after the last convolution. It is assumed that each of these filters will detect class specific characteristics in the image [9]. A ReLU activation function [22] is applied to the output of the added convolutional layer to ensure that the features are non-negative. This has a beneficial impact on the distance to anchors, as negative values on other class dimensions would increase the distance to the anchor. After the added convolutional layer, global average pooling [18] is used to obtain the vector of features $\boldsymbol{z}$.

3.3.2. Distance loss

To optimize our model using DIFAIR, we use the Euclidean distance $d_c = \sqrt{\sum_i (z_i - \mathcal{A}_i^{(c)})^2}$ between the predicted representation $\boldsymbol{z}$ and the true class anchor $\mathcal{A}^{(c)}$. If the representation is outside the hypersphere of radius r, the loss is the distance to the hypersphere's border: $\mathcal{L}_{\text{out}} = \max(d_c - r, 0)$. Note that this loss is 0 if the representation is within the hypersphere. Preliminary results showed that lacking a penalization when the instance was represented within its hypersphere, made it challenging for the model to separate instances correctly, and leading to degraded OSR performances. This observation led to the introduction of the term $\mathcal{L}_{\text{in}} = 0.1 \times d_c$ used to apply a penalization when the instance representation is within its hypersphere. The final distance loss is then $\mathcal{L}_{\text{distance}} = \mathcal{L}_{\text{in}} + \mathcal{L}_{\text{out}}$.

The significant loss reduction when the distance is below r grants the network flexibility in representing the instance within the hypersphere. The proposed loss gives more importance to having the instance within the hypersphere than to having it close to the anchor.

Having the flexibility to represent data in hyperspheres avoids the optimization of an invariant representation for each class, whereas CAC needs logit outputs to be invariant to make a prediction. This enables the network to extract features that can be more or less activated, and embed semantic information in the representation (section 3.2.2).

3.3.3. Weights correlation loss

The first results obtained during our research showed that some filters of the added convolutional layer were converging to the same values, therefore *extracting the same information* from this layer's input. More precisely, let $\boldsymbol{W}^{(i)}, i \in \{1, \dots, n_c \times \mathcal{N}\}$ be the i-th filter of the added convolutional layer. For a given class of index $c \in \{1, \dots, n_c\}$, the set $\mathbb{W}_c = \{(c-1) \times \mathcal{N} + 1, \dots, (c-1) \times \mathcal{N} + \mathcal{N}\}$ contains the indices of filters associated with that class. Filters $\boldsymbol{W}^{(i)}, i \in \mathbb{W}_c$, i.e. linked to the same class, were converging towards the same values. While this approach satisfy the loss objective from section 3.3.2, it is not desirable as our objective is to extract independent features for each class.

To address this issue, we introduced a penalty on the linear correlation of the filters $\boldsymbol{W}^{(i)}$ of the last layer, which extracts features of the same class, to avoid the extraction of redundant information. In other words, a correlation between filters associated with different classes is still authorized, aligning with the idea that some features can be present in multiple classes. Thus, the penalization will not have a negative impact on the semantic meaning that we want the representation to have. The correlation loss is then defined as:

$$\mathcal{L}_{\text{correlation}} = \frac{1}{n_c} \sum_{c=1}^{n_c} \rho_c \quad \text{with} \quad \rho_c = \frac{2}{\mathcal{N}(\mathcal{N}-1)} \sum_{\substack{i,j \in \boldsymbol{W}_c \\ j>i}} \text{corr}(\boldsymbol{W}^{(i)}, \boldsymbol{W}^{(j)})$$

This loss forces the linear correlation of filters associated with a same class to be close to 0, preventing the filters from extracting the same information respectively to an affine transformation. In the end, the model is optimized using the loss $\mathcal{L}_{\text{DIFAIR}} = \mathcal{L}_{\text{distance}} + \lambda\mathcal{L}_{\text{correlation}}$, with λ a weight to balance the two losses.

4. Open-set results

With DIFAIR, we aim to learn well-separated representations in space as well as class-representative features on each dimension. To assess our approach in OSR, we follow Neal et al. [23] standard evaluation protocol, using their proposed network architecture. It is a compact VGG architecture [27], referred to as "VGG32" (implementation details are available on GitHub: https://github.com/qchristoffel/DIFAIR).

4.1. Datasets

Neal et al. [23] introduced a benchmark to evaluate OSR performances. This benchmark aims to use a standard closed-set dataset, where classes are split into 2 random groups, and uses one group as known classes for training, while the other group contains unknown classes to be detected when testing. Results are then averaged on 5 random splits. We use the same data splits as Vaze et al. [29] to obtain comparable results. MNIST [17] and SVHN [24] contain digits from two different domains: handwritten and from street view house numbers, respectively. CIFAR10 [15] contains 10 classes of animals and vehicles. Neal et al. [23] proposed a setting where the 4 vehicle classes from CIFAR10 are used as known classes and N random classes from CIFAR100 [15] are used as unknown classes. N can be either 10 or 50, so the experiments are referred to as CIFAR+N. It is important to note that classes from CIFAR10 and CIFAR100 are non-overlapping, ensuring that unknown classes are never seen during training. Lastly, TinyImageNet [16], a dataset of 200 classes subsampled from ImageNet [25] is used in a setting where 20 random classes are known, while the remaining 180 are considered unknown.

4.2. Methods

As a baseline, we use a neural network train with cross-entropy. From this baseline, we extract two OSR scores: the Maximum Softmax Probability (**MSP**) with $S(y|\boldsymbol{X}) = \max \text{softmax}(\hat{\boldsymbol{y}})$ and the Maximum Logit Score (**MLS**) proposed by Vaze et al. [29]: $S(y|\boldsymbol{X}) = \max \hat{\boldsymbol{y}}$.

We generated new results for Class Anchor Clustering (**CAC**) [20] to ensure a fair comparison with our proposed approach because the original results were reported on different random splits, which can influence performance. For CAC, we used the OSR score reported in their article (described in Section 2).

4.3. Experiments

For our experiments and CAC re-training, we follow the improved training protocol proposed by Vaze et al. [29]. This protocol involves training for a greater number of epochs (600), using RandAugment data augmentation [5], and using a learning rate scheduler. Experiment details are provided on our GitHub.

For DIFAIR experiments, the correlation loss is weighted using $\lambda = 1$, anchors are parameterized using $\mathcal{N} = 5$, $\alpha = 10$ and $r = 0.6 \times d_{aa}$ where $d_{aa} = \sqrt{2\mathcal{N}\alpha^2} \approx 31,6$, representing the Euclidean distance between two anchors. This configuration allocates 60% of the space between two anchors to each hypersphere, resulting in overlapping hyperspheres. While allowing hyperspheres to overlap may seem counterintuitive in our classification setting, it resulted in higher performance during hyperparameter search.

4.4. Results

Table 1 presents the closed-set accuracies obtained by the different methods and their OSR results on the benchmark established by [23]. We encountered difficulties in reproducing the same baseline results as those reported by Vaze et al. [29] with the Maximum Logit Score (MLS), thus we also reported their own results for comparison.

Both CAC and DIFAIR exhibit lower AUROC scores compared to the MLS baseline. Applying the Maximum Logit Score (MLS) proposed by Vaze et al. [29] to DIFAIR and CAC outputs did not improve results for both methods, excepted on TinyImageNet, where the model had difficulties to converge. For CAC, we noticed that modifying anchors at the end of training reduced AUROC on Tiny ImageNet, from 72.18 to 67.65. Overall, DIFAIR's performance closely approached that of CAC, with CAC yielding higher results in most cases, except on Tiny ImageNet where the AUROC is better using DIFAIR. DIFAIR's accuracy on the closed-set task is equivalent to the cross-entropy baseline, except for the TinyImageNet dataset where it is higher.

Table 1. **Results on Neal et al. [23] benchmark**. Closed-set accuracy(%) and AUROC scores for the task of detecting unknown classes, averaged over five known/unknown class splits. If results are not referenced, they originate from our work. The highest AUROC scores are in bold.

Method	MNIST	SVHN	CIFAR10	CIFAR+10	CIFAR+50	TinyImageNet
MSP	99.8/99.32	97.7/95.38	96.28/89.78	96.02/89.60	96.48/90.13	74.48/73.24
MLS	99.8/**99.55**	97.7/96.66	96.28/92.55	96.02/91.10	96.48/92.53	74.48/73.87
MLS [29]	- /99.3	- /**97.1**	- /**93.6**	- /**97.9**	- /**96.5**	- /**83.0**
CAC	99.81/99.11	98.11/96.61	96.42/86.43	96.20/86.03	96.43/87.28	77.68/67.65
DIFAIR	99.81/98.35	98.04/95.45	96.35/85.04	95.82/83.86	96.49/84.52	76.06/73.38

4.5. Discussion

With DIFAIR, features can be activated even when facing unknown data since there are no constraints on unknown data during training. The aim of our approach is that if a *known* characteristic is present in an image, then the corresponding feature should be activated in the representation, even for unknown images. That might be detrimental to OSR performance, as evidenced by the results of DIFAIR. However, we believe that being able to extract semantic information from unknown data is important as this data can then be reused in other tasks, based on the information extracted. It is interesting to note that state-of-the-art approaches [3, 4, 31] all incorporated data identified as unknown during training, which is not the case for DIFAIR and CAC.

The lower results obtained using MLS on both DIFAIR and CAC show that when models achieve convergence, the disparity of the maximum activation between known and unknown classes is less pronounced compared to the baseline. In the case of DIFAIR, it might be due to the fact that we allow features to be activated even for unknown classes, which often share features with known classes in these datasets. Furthermore, DIFAIR outputs multiple values per class, which are effectively utilized by the Euclidean distance measure, but not by the MLS. Future research could explore alternative methods for combining the extracted features into an OSR score.

Comparing DIFAIR to CAC, using the distance to anchors score, we note that adding dimensions and allocating hyperspheres around anchors in DIFAIR led to a degradation of the AUROC score. This may come from the fact that in CAC, in addition to using distance to anchor during learning, there is another loss term to ensure that an instance's representation will be represented far from class anchors other than its own. In contrast, DIFAIR does not use such a loss term because of our specific objective of maintaining semantic meanings in representations. DIFAIR allows for *semantic proximity* with other classes. This means that instance representations may exhibit the activation of certain features that are common to other classes, promoting a more semantically meaningful representation. Therefore, the trade-off between forced separation and semantic proximity highlights the different objectives of these two methods, which can influence their respective AUROC score in OSR tasks.

5. Representation analysis

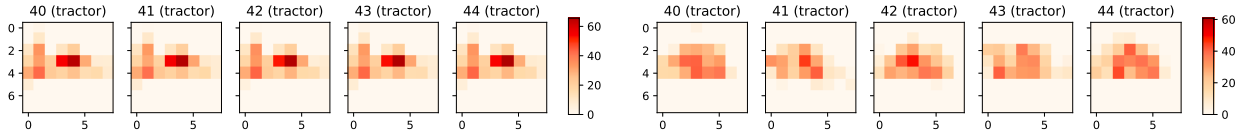
5.1. Visualization

To comprehend the OSR results obtained by DIFAIR, we leverage the association between the extracted features (as well as the activation maps preceding them) and pre-defined classes. In this section, our objective is to assess the hypotheses presented in Section 3.2.2. Namely, we want to determine whether a model trained with DIFAIR (i) extracts independent features across dimensions of the same class in the latent space; and (ii) if dimensions remain inactivated when features are absent. To achieve these goals, we generated diagrams from the representations in a manner similar to Hinton diagrams [13]. We inspected representations of a model trained on the first split of TinyImageNet. In this section, it is important to note that we illustrate DIFAIR results and show the approach limits on specific examples, which does not make it an exhaustive evaluation, but which are representative of the behavior we observed on multiple random examples, that can be visualized on our GitHub: `https://github.com/qchristoffel/DIFAIR/representations`.

Beforehand, note that the model did not achieve full convergence on TinyImageNet. For instance, the mean representations of known classes exhibited activation values averaging around 8 on their respective dimensions instead of $\alpha = 10$. Thus, instances are already distant from their anchor points on average. Moreover, classes "reel" and

"binocular" encountered even greater difficulties to converge, with average activations around 6, notably lower than the expected 10. Conversely, on the third split of CIFAR10, the model demonstrated a better convergence, with classes having average representations with values above 9.7 on dimensions. Despite the model's incomplete convergence on TinyImageNet, we maintain that the results remain relevant for analysis, and we show that the number of known classes, which is linked to the number of features to analyze, is not limiting the analysis.

The first objective of this analysis is to assess whether the learned representation extracts independent features across different dimensions of the latent space. To prevent feature duplication, we have implemented a loss term $\mathcal{L}_{\text{correlation}}$. Specifically, the loss term aimed at minimizing the correlation of filters associated with the same class, as presented in Section 3.3.3. This objective was easily optimized by the network, thus we decided not to explore its weighting λ in the total loss $\mathcal{L}_{\text{DIFAIR}}$ for the experiment reported here.



(a) Activation maps obtained without using correlation loss.

(b) Activation maps obtained using the correlation loss.

Fig. 1. **Comparison of activation maps for models trained with and without the correlation loss.** Given an image of a tractor, activation maps associated with the tractor class are shown. Above each feature map is its index and the class it is associated with. Axis graduations serve as coordinates for specific features.

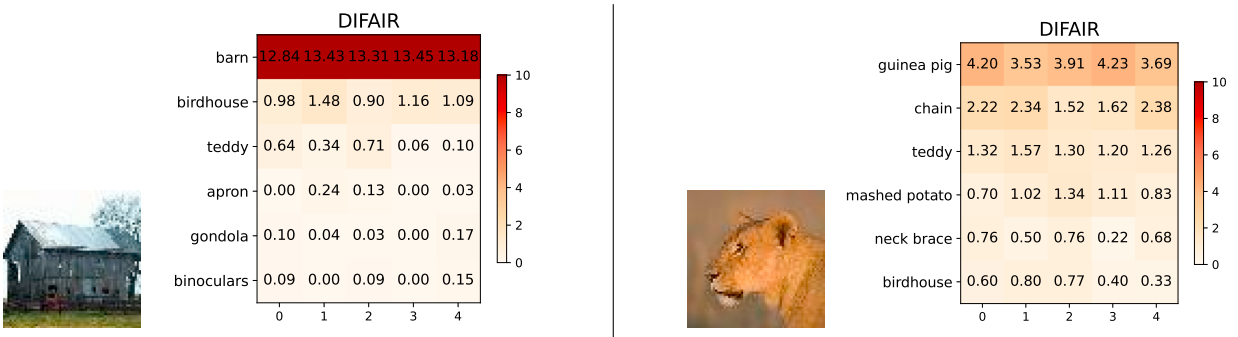
Rather than visualizing the effects of penalization on filters which are 3-dimensional, we visualize, in Figure 1, how it impacts different activation maps. The activation maps obtained without using the correlation loss (Figure 1a) are all similar, therefore the resulting features (obtained via global average pooling of activation maps) are all equal and not independent. In contrast, the activation maps obtained using the correlation loss (Figure 1b) exhibits activations patterns of different intensities, while remaining at the same localization, which is expected for a convolution. We can deduce from these graphs that the model effectively avoid the duplication of information on dimensions of a same class thanks to the correlation loss term, but we cannot state yet that the extracted features will be independent.

The second aspect we aim to examine is whether features are deactivated when the corresponding class is not present in the image, i.e. the characteristics of this class are not supposed to be in the image. For this qualitative analysis, we utilized example images of a "barn", a known class, and a "lion", an unknown class. The images and representations obtained using DIFAIR are shown in Figure 2.

A representation extracted from a cross-entropy trained model does not allow us to directly associate a feature with a class. Whereas in DIFAIR representation, the correspondence between features and classes is clearly defined by the approach. It is known which features should be activated for an instance to be classified into a particular class. A feature cannot be shared by multiple classes (i.e. a feature will contribute to the prediction of a single class), that is why in Figure 2b, "birdhouse" features are slightly activated in the "barn" representation, exhibiting some semantic similarity between the two classes in this example. This representation also reveals that when the model is confident in its prediction, it can activate features of the predicted class with high values, exceeding α (10 in our case). Apart from activations related to semantically similar classes (or errors, as it seems to be the case for "teddy" features activated in the "barn" representation), the model demonstrates the ability to deactivate features from most classes not present in the image, at least *when considering images of known classes*. It can also be noticed that *all* dimensions of the "barn" class are activated with close values.

When considering unknown classes, two activation scenarios were observed. The first possibility is to have multiple features activated with low values, as somewhat desired, but in this situation all the features from multiple classes are activated. The other situation consists in the activation of all features of a class with high values, as shown in Figure 2d. These values can be even higher in case the model misclassifies the instance. These activations are unexpected because unknown classes should not present *all characteristics* of multiple known classes.

We hypothesize that the model still learns some correlation between features of a same class, effectively activating all of them when the network detects that a certain class is more probable than the others, in order to be aligned with the optimization objective regarding the distance to the anchor. Features are not disentangled even though we forced filter weights to be decorrelated and that feature maps denotes different activations as seen in Figure 1b. Another



(a) Barn. (b) Representation of a DIFAIR-trained model given the image of Barn. (Known class)

(c) Lion. (d) Representation of a DIFAIR-trained model given the image of Lion. (Unknown class)

Fig. 2. **DIFAIR representations for known (left) and unknown (right) images.** The models were trained on the first split of TinyImageNet. DIFAIR representations show the *top 6 most activated* classes (instead of 20), determined by the sum of features associated with each class.

possible explanation for the activation of all features of a class is that the model learns to extract only features that are present in *all training images* of the class it has seen during training. This would explain why the model activates all features of a class when it detects the presence of a class and this would also partly explain why unknown classes also activate filters: because filters are too much "train-data" generalized and not enough "real-class" specific. These hypotheses are difficult to verify and we need to conduct further investigations to understand the behavior of the model in this situation.

5.2. Areas for improvement

Given observations made in Section 5.1, we can conclude that the penalization of the correlation of weights is not sufficient to attain a certain independence of extracted features. While the weights and features obtained indeed have different values, features of the same class are still activated with close values and all together. We expected that forcing the extraction of different features this way would lead to features that are more or less activated depending on the characteristics present in the input image, but whatever the image, all features of a class are activated with similar magnitudes. Therefore, we need to find ways to extract features that are activated more independently.

Considering the closed-set task, DIFAIR demonstrates performance similar to other evaluated methods. However, when confronted with the challenge of detecting unknown classes, which should not present all characteristics of a known class, class features still fire together. Extracting uncorrelated features, rather than the generalized features seemingly learned by the model, could result in better deactivation of features when the class is absent from the image.

6. Conclusion

In this paper, we presented our propositions for obtaining differentiated representations using neural network classifiers, where each extracted feature (dimension of the representation) is associated with a specific class, while maintaining a clear separation between different classes in the representation space. Features should be activated according to the presence or absence of specific characteristic in the input image. To achieve these objectives, we proposed DIFAIR, which relies on the definition of *class anchors* equally separated in the representation space. Around these anchors, respective class instances are represented within the boundaries of a hypersphere. We optimize neural networks in this setting using the Euclidean distance thresholded by the hypersphere radius, supplemented by an additional loss term to avoid the extraction of duplicated features across class dimensions.

We evaluate the quality of our learned representation through Open-Set Recognition tasks. While our method achieves results equivalent to Class Anchor Clustering [20], a similar approach, it remains below state-of-the-art results in the field [4, 29, 3, 31]. Nevertheless, utilizing distance and hyperspheres during training allows the activation of features from other classes, increasing the semantic meaningfulness of the representations at the expense of class separation in the representation space.

Visual analysis of the learned representations reveals some correlation across class dimensions of predicted class (i.e. they are all activated and with similar values), indicating that further refinement is needed to achieve the extraction of features that can be activated distinctly from others.

Despite challenges in OSR, our visual exploration of the DIFAIR representation uncovers ways of improvement towards obtaining differentiated representation. Future investigations will focus on exploring loss functions tolerant of absent features on true class dimensions, while working on the extraction of more distinct features for each class.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used ChatGPT in order to rewrite sentences and shorten some paragraphs. After using this tool, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

References

[1] Bendale, A., Boult, T.E., 2016. Towards open set deep networks, in: CVPR, pp. 1563–1572.
[2] Cevikalp, H., Uzun, B., Köpüklü, O., Ozturk, G., 2021. Deep compact polyhedral conic classifier for open and closed set recognition. Pattern Recognition 119, 108080.
[3] Cevikalp, H., Uzun, B., Salk, Y., Saribas, H., Köpüklü, O., 2023. From anomaly detection to open set recognition: Bridging the gap. Pattern Recognition 138, 109385.
[4] Chen, G., Peng, P., Wang, X., Tian, Y., 2021. Adversarial reciprocal points learning for open set recognition. TPAMI .
[5] Cubuk, E.D., Zoph, B., Shlens, J., Le, Q.V., 2020. Randaugment: Practical automated data augmentation with a reduced search space, in: CVPR Workshops, pp. 702–703.
[6] Dietterich, T.G., Guyer, A., 2022. The familiarity hypothesis: Explaining the behavior of deep open set methods. Pattern Recognition 132, 108931.
[7] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale, in: ICLR.
[8] Ge, Z., Demyanov, S., Garnavi, R., 2017. Generative openmax for multi-class open set classification, in: BMVC, pp. 42.1–42.12.
[9] Géron, A., 2019. Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow. volume 1. O'Reilly Media.
[10] Hassen, M., Chan, P.K., 2020. Learning a neural-network-based representation for open set recognition, in: SIAM International Conference on Data Mining, pp. 154–162.
[11] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: CVPR, pp. 770–778.
[12] Hinton, G.E., Rumelhart, D.E., McClelland, J.L., 1987. in: Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations. chapter 3. pp. 77–109. URL: https://www.cs.toronto.edu/~hinton/absps/pdp3.pdf.
[13] Hinton, G.E., Shallice, T., 1991. Lesioning an attractor network: investigations of acquired dyslexia. Psychological review 98, 74.
[14] Kotz, S., Nadarajah, S., 2000. Extreme value distributions: theory and applications. World Scientific.
[15] Krizhevsky, A., Hinton, G., et al., 2009. Learning multiple layers of features from tiny images. Technical Report, University of Toronto .
[16] Le, Y., Yang, X., 2015. Tiny imagenet visual recognition challenge. CS 231N 7, 3.
[17] LeCun, Y., Cortes, C., Burges, C., 2010. Mnist handwritten digit database. ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist .
[18] Lin, M., Chen, Q., Yan, S., 2014. Network in network, in: ICLR.
[19] Lu, J., Xu, Y., Li, H., Cheng, Z., Niu, Y., 2022. Pmal: Open set recognition via robust prototype mining, in: AAAI Conference on Artificial Intelligence, pp. 1872–1880.
[20] Miller, D., Sunderhauf, N., Milford, M., Dayoub, F., 2021. Class anchor clustering: A loss for distance-based open set recognition, in: WACV, pp. 3570–3578.
[21] Moon, W., Park, J., Seong, H.S., Cho, C.H., Heo, J.P., 2022. Difficulty-aware simulator for open set recognition, in: ECCV, pp. 365–381.
[22] Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines, in: ICML, pp. 807–814.
[23] Neal, L., Olson, M., Fern, X., Wong, Weng-Keenand Li, F., 2018. Open set learning with counterfactual images, in: ECCV, pp. 613–628.
[24] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., Ng, A.Y., 2011. Reading digits in natural images with unsupervised feature learning, in: NeurIPS. URL: http://ufldl.stanford.edu/housenumbers/nips2011_housenumbers.pdf.
[25] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al., 2015. Imagenet large scale visual recognition challenge. IJCV 115, 211–252.
[26] Scheirer, W.J., de Rezende Rocha, A., Sapkota, A., Boult, T.E., 2013. Toward open set recognition. TPAMI 35, 1757–1772.
[27] Simonyan, K., Zisserman, A., 2015. Very deep convolutional networks for large-scale image recognition, in: ICLR.
[28] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need, in: NeurIPS.
[29] Vaze, S., Han, K., Vedaldi, A., Zisserman, A., 2022. Open-set recognition: a good closed-set classifier is all you need?, in: ICLR.
[30] Wang, S., Su, Z., 2020. Metamorphic object insertion for testing object detection systems, in: ASE, pp. 1053–1065.
[31] Xu, B., Shen, F., Zhao, J., 2023. Contrastive open set recognition, in: AAAI Conference on Artificial Intelligence, pp. 10546–10556.
[32] Zhang, H., Cisse, M., Dauphin, Y.N., Lopez-Paz, D., 2018. mixup: Beyond empirical risk minimization, in: ICLR.
[33] Zhang, H., Li, A., Guo, J., Guo, Y., 2020. Hybrid models for open set recognition, in: ECCV, Springer. pp. 102–117.