PIEs: Pose Invariant Embeddings

Chih-Hui Ho Pedro Morgado Amir Persekian Nuno Vasconcelos
University of California, San Diego
{chh279,pmaravil,aperseki,nvasconcelos}@ucsd.edu

Abstract

The role of pose invariance in image recognition and retrieval is studied. A taxonomic classification of embeddings, according to their level of invariance, is introduced and used to clarify connections between existing embeddings, identify missing approaches, and propose invariant generalizations. This leads to a new family of pose invariant embeddings (PIEs), derived from existing approaches by a combination of two models, which follow from the interpretation of CNNs as estimators of class posterior probabilities: a view-to-object model and an object-to-class model. The new pose-invariant models are shown to have interesting properties, both theoretically and through experiments, where they outperform existing multiview approaches. Most notably, they achieve good performance for both 1) classification and retrieval, and 2) single and multiview inference. These are important properties for the design of real vision systems, where universal embeddings are preferable to task specific ones, and multiple images are usually not available at inference time. Finally, a new multiview dataset of real objects, imaged in the wild against complex backgrounds, is introduced. We believe that this is a much needed complement to the synthetic datasets in wide use and will contribute to the advancement of multiview recognition and retrieval.

1. Introduction

Convolutional neural networks (CNNs) are frequently used for classification and metric learning, among other tasks. Classification is the central problem of important computer vision applications, such as object and action recognition or detection. Metric learning plays a similar role for image retrieval, face recognition and identification, or zero shot learning. Despite the many different applications, the two tasks are closely related, since they both learn an embedding $g : \mathcal{X} \rightarrow \mathcal{G}$ that maps images $\mathbf{x} \in \mathcal{X}$ into features $g(\mathbf{x}) \in \mathcal{G}$ and are implemented with several CNN layers. Classification aims to produce a discriminant feature space $\mathcal{F}$, which separates the different classes, while metric learning aims to produce a feature space $\mathcal{F}$ with a certain metric structure, where similarity can be captured by some distance function, typically the Euclidean distance.

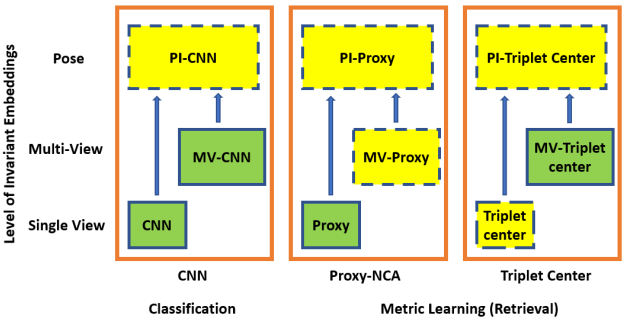


Figure 1. Taxonomy of embeddings learned by different methods according to different level of invariance. Green solid boxes represent methods in the literature and yellow dashed boxes represent methods proposed in this work. The proposed pose invariant embedding incorporates single view and multiview invariance and can be applied to different methods, including CNN, proxy-NCA and triplet center. While CNN is designed for classification, the other two aim for metric learning (retrieval task).

As shown in the bottom row of Figure 1, classification and metric learning have evolved in lockstep. While details of the architecture of $g(.)$ may favor one or the other, approaches to the two problems have differed mostly in the subsequent network layers and loss function. Classifiers complement the embedding g with a softmax layer trained with the logistic loss. Classic metric learning uses no additional layers and a different loss. While several variants have been proposed [5, 10, 21, 15], the most popular is the *triplet* loss of [27, 26, 1, 19]. In practice, however, the differences can be significant. Since triplets raise the dataset size to its cube, metric learning networks are more difficult to train than classifiers. To address this, much of the embedding literature has been devoted to triplet sampling strategies [26, 1, 19, 17, 21], aimed to increase training speed. Recently however, [15] has shown that much faster training is possible by using *proxy* embeddings, which make metric learning a lot more like classification. Inspired by a metric learning approach known as *neighborhood component analysis* (NCA) [9], it adds a layer that resembles a soft-

max classifier to the embedding and uses the logistic loss for training. A similar generalization of triplet embeddings has been proposed in [11], and denoted as *triplet center* embeddings.

Ideally, an embedding should map all the images of an object collected from multiple views, depths or under different illuminations, into a single point, known as the *object invariant* to these transformations. However, this is hard to achieve on datasets such as ImageNet, which tend to emphasize class diversity and maximize the number of objects imaged per class. They do not provide a dense covering of the transformations (imaged from different camera positions, variable lighting, etc) where an object may be subjected to. Recently, this problem has received significantly more attention, with the introduction of datasets such as ModelNet [29] or ShapeNet [4]. Being datasets of synthetic images rendered from 3D CAD models, these allow the generation of many views of each object labelled for view angle, also known as *object pose*.

The introduction of these multiview synthetic datasets motivated a new wave of algorithms for multiview [22, 12, 6] classification and retrieval, as shown in the middle row of Figure 1. These methods have been shown competitive, if not superior, to many methods based on 3D representations, such as voxels [29, 14, 3, 28] or point clouds [7, 31, 30]. This is important because view-based representations can be easily deployed in the real world, where 3D representations are much more expensive, if not completely infeasible. The most popular architecture for view-based classification is the multiview-CNN (MV-CNN) [22], which complements a standard CNN embedding with a view pooling mechanism that produces a shape descriptor. The shape descriptor is then fed to the softmax layer for classification. Similarly, [11] have introduced the triplet-center loss for multiview metric learning. This is a generalization of the triplet loss and center loss to a multiview level for NCA style metric learning.

While these approaches have been shown to be effective for multiview classification and retrieval, which can be performed easily in the CAD world (e.g ModelNet and ShapeNet), their usefulness for real vision systems is more questionable, for two reasons. First, it is not known how well they work on real images due to the absence of datasets of real images in the wild, with coverage of pose trajectories. While some dense pose datasets exist [2, 12, 8, 16], they are small and tend to depict objects on turntables, without complex backgrounds. Second, and more important, these approaches do not really learn *pose invariant* embeddings. While the shape descriptor is a summary of all the views of the object, the embedding of a single image is not constrained to be similar to this descriptor. In result, these methods tend not to perform well for *single view* recognition or retrieval, where they frequently have weaker performance than standard CNNs. This is important because the multiview setting is not realistic for most real world applications. While multiview training is of interest to enable learning algorithms to capture object variability under various transformations, applications frequently constrain inference to single views. To support the latter, multiview training must produce truly pose invariant embeddings.

In this work, we address these limitations through a combination of contributions. First we perform a review of various approaches in the literature, placing various methods on equal footing and enabling a better understanding of their relative strengths and weaknesses. This results in Figure 1, which groups embeddings by their level of invariance. Existing methods are identified by green boxes. It is clear that no truly pose invariant embeddings are available. While view-based embeddings have little invariance, multiview embeddings produce a shape descriptor that represents multiple views, but do not map individual views to this descriptor.

Second, we propose a number of new approaches, showed as yellow boxes in Figure 1. Some of these just fill holes in the layers populated by existing methods. For example, MV-Proxy is simple variants of [15] for multiview level and triplet center is variants of [11] for single view level. Other yellow boxes in pose invariant level (top row) are based on new loss functions that encourage embeddings that cluster individual images in the neighborhood of shape descriptors. This makes the shape descriptors truly invariant and enables better performance on single view retrieval and recognition tasks. Finally, we introduce a new multiview dataset for object recognition in the wild. This dataset is composed of objects belonging to ImageNet, and are in all aspects similar to ImageNet images. However, each object is imaged under a set of pre-defined poses, which are provided as additional labels. Similarly to ShapeNet and ModelNet, this enables the learning of pose invariant representations. However, because the images are real, the new dataset enables the testing of invariance in a more realistic setting. Experiments on both the proposed dataset and synthetic datasets show that the proposed pose invariant embedding is more robust to a variable number of views provided for inference.

2. Related work

Many works have addressed embeddings for classification and retrieval. We review the literature in this section, emphasizing the ideas that are directly relevant to this work.

Classification: Given observations and class labels drawn from random variables $X \in \mathbb{R}^m$ and $Y \in \{1, \dots, C\}$ the classifier of minimum probability of error is $y^* = \arg\max_y P_{Y|\mathbf{X}}(y|\mathbf{x})$. A CNN is a model for the posterior

probabilities

$$P_{Y|\mathbf{X}}(y|\mathbf{x}) = h_y(\mathbf{x}; \mathbf{W}, \mathbf{b}) = \frac{e^{\mathbf{w}_y^T g(\mathbf{x}) + b_y}}{\sum_{k=1}^{C} e^{\mathbf{w}_k^T g(\mathbf{x}) + b_k}} \quad (1)$$

composed of two stages. The first is an embedding $g(\mathbf{x}) \in \mathcal{F} \subset \mathbb{R}^d$, implemented by the layers of the network up to the last one, where g is a d dimension feature extractor. Usually, g consists of a combination of convolutions, pooling, and a ReLU non-linearity. The second is a softmax layer that resides at the top of the network and computes (1) using a layer of weights $\mathbf{W} \in \mathbb{R}^{d \times C}$ and biases $\mathbf{b} \in \mathbb{R}^C$. To minimize notational clutter, we will omit the bias vector in many of the expressions below. This follows the common practice of absorbing it in $\mathbf{W}$ and using homogeneous coordinates. CNNs are trained by cross-entropy minimization. Given a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ this consists of finding $\mathbf{W}$ and the parameters of g that minimize the risk

$$\mathcal{R}(\mathcal{D}) \quad = \quad \sum_i L(\mathbf{x}_i, y_i), \quad (2)$$

defined by the logistic loss $L(\mathbf{x}, y) = -\log h_y(\mathbf{x}; \mathbf{W})$.

Metric learning: Metric learning aims to endow the feature space $\mathcal{F}$ with a metric, usually the Euclidean distance

$$d(g(\mathbf{x}), g(\mathbf{y})) = ||g(\mathbf{x}) - g(\mathbf{y})||^2, \quad (3)$$

so as to allow the geometric implementation of operations like classification, e.g. using nearest neighbors. While many losses have been proposed [5, 10, 21], this is usually done with a loss function that operates on example triplets, pulling together (pushing apart) similar (dissimilar) examples [27, 26, 1, 19]. Given an anchor $\mathbf{x}$, a similar $\mathbf{x}^+$ and a dissimilar example $\mathbf{x}^-$, the triplet loss is defined as

$$L(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-) = \phi\left(d(g(\mathbf{x}), g(\mathbf{x}^-)) - d(g(\mathbf{x}), g(\mathbf{x}^+))\right), \quad (4)$$

where $\phi(.)$ is a margin loss, e.g. the hinge loss $\phi(v) = \max(0, m - v)$ or the logistic loss $\phi(v) = \log(1 + e^{-v})$. In general, similar and dissimilar examples are determined by the class labels of $\mathcal{D}$. We refer to these methods as *triplet* embeddings.

Modern CNNs are learned by stochastic gradient descent (SGD), processing the data in batches of relatively small size, e.g. $b = 32$. On a dataset of size n there are $O(n)$ examples and $O(n^3)$ triplets. Similarly, there are $O(b)$ examples and $O(b^3)$ triplets in a batch. Hence, while the number of batches needed to cover the dataset is $O(n/b)$ for examples, it becomes $O((n/b)^3)$ for triplets [15]. Since n/b is in the tens of thousands, triplet learning is cubically more complex than example-based learning. While many sampling strategies have been proposed to address this problem [19, 26, 17, 23], metric learning methods are substantially harder to use and slower to converge than classification methods.

Recently, [15] has shown that this problem can be overcame using a loss function inspired by neighborhood component analysis (NCA) [9]. This consists of defining a *proxy* $\mathbf{p}_y$ per class, adding a softmax-like layer

$$s_y(\mathbf{x}; \mathbf{P}) = \frac{e^{-d(g(\mathbf{x}), \mathbf{p}_y)}}{\sum_{k \neq y} e^{-d(g(\mathbf{x}), \mathbf{p}_k)}}, \quad (5)$$

where $\mathbf{P}$ is the matrix of proxies $\mathbf{p}_k$, and learning both $\mathbf{P}$ and $g(\mathbf{x})$ by minimizing the risk of (2) with the logistic loss $L(\mathbf{x}, y) = -\log s_y(\mathbf{x}; \mathbf{P})$. We refer to this method as *proxy* embedding.

Multiview classification: In multiview classification, each observation consists of a set of V views $\mathbf{X} = \{\mathbf{x}_k\}_{k=1}^V$ and parameters are learned from a multiview dataset $\mathcal{D}_m = \{(\mathbf{X}_i, y_i)\}_{i=1}^n = \{(\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iV}, y_i)\}_{i=1}^n$. The goal is to jointly classify all these views. A popular approach is the multiview-CNN (MV-CNN) [22], which implements two embeddings. Each individual image x_k, where x_k is imaged at k^{th} predefined viewpoint, is processed by a shared feature extractor g and all the resulting view descriptors $g(x_k)$ is then averaged to produce a *shape* descriptor

$$g_m(\mathbf{X}) = \frac{1}{V} \sum_{k=1}^{V} g(\mathbf{x}_k), \quad (6)$$

where subscript m denotes multiview. The embedding parameters are learned from a multiview dataset $\mathcal{D}_m$ by using g_m with softmax layer (1), the risk of (2), and the logistic loss. Several variants of this approach have been proposed, either making specific architectural enhancements to the embedding g [25, 18], or using weighted versions of (6) [6]. Similar enhancements are possible for all methods discussed in this work.

Multiview metric learning: Substantially less work has been devoted to multiview metric learning. [11] combined the MV-CNN embedding with the proxy-based idea of [15], but applied to the triplet loss. They denote proxies as *centers* and define the multiview triplet-center loss

$$L(\mathbf{X}, y, \mathbf{P}) = \phi\left(\min_{j \neq y} d(g_m(\mathbf{X}), \mathbf{p}_j) - d(g_m(\mathbf{X}), \mathbf{p}_y)\right) \quad (7)$$

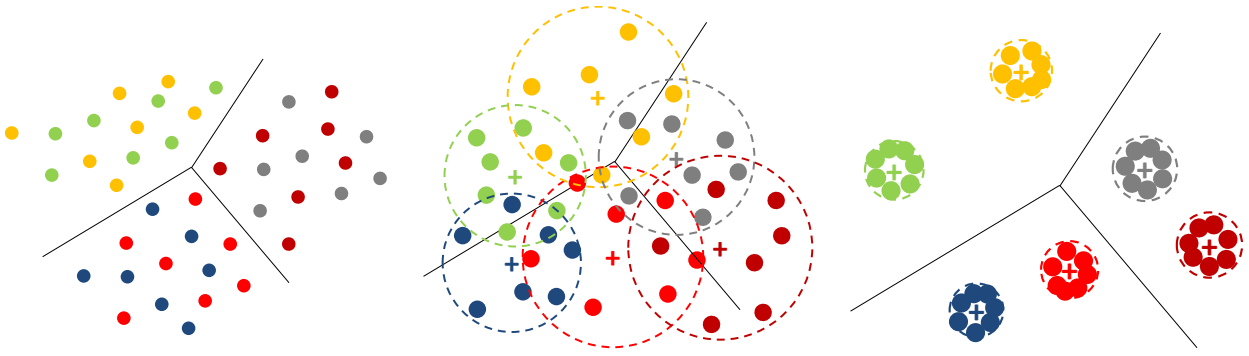
where $\mathbf{P}$ is the matrix of centers $\mathbf{p}_j$ and g_m is defined as in (6). We refer to this method as *multiview triplet center (MV-TC)* embedding.

3. Bringing object invariants to the real world

In this section, we discuss a number of contributions that follow from the above review.

3.1. New view-based and mutiview embeddings

Figure 1 provides a functional organization of embeddings for classification and metric learning. The bottom two



a) view-based b) multi-view c) invariant

Figure 2. Embeddings produced by methods at the three levels of invariance of Figure 1. In all plots, there are three classes, two objects per class, and each dot represents the embedding of an image. Dots of the same color correspond to different views of the same object. In b) and c), a $'+'$ is used to denote the shape descriptor and a dashed circle to denote the distribution of views of the associated object. Only the invariant embedding of c) guarantees a good clustering of both shape descriptors per class and individual views per object.

rows summarize the state of the literature, with green boxes identifying the approaches that have been proposed. They group these methods according to whether they embed single or multiple views. One immediate contribution is that there are a number of "missing" approaches (e.g multiview proxy and single view triplet center). We propose to fill the gaps, introducing several new embeddings, which are extensions of those available: the *triplet center* embedding is the view-based equivalent of the multiview triplet center embedding[11], replacing multiview triplet-center loss (7) with single view

$$L(\mathbf{x}, y, \mathbf{P}) = \phi\left(\min_{j \neq y} d(g(\mathbf{x}), \mathbf{p}_j) - d(g(\mathbf{x}), \mathbf{p}_y)\right), \quad (8)$$

and the *MV-proxy* generalizes the single view proxy embedding (5) to multiview

$$s_y^m(\mathbf{X}; \mathbf{P}) = \frac{e^{-d(g_m(\mathbf{x}), \mathbf{p}_y)}}{\sum_{k \neq y} e^{-d(g_m(\mathbf{x}), \mathbf{p}_k)}}, \quad (9)$$

where superscript m denotes multiview.

3.2. The need for invariant embeddings

A second, and practically more important, contribution of Figure 1 is to show that no attention has been given to the design of truly *invariant* embeddings. This is important for many real-world systems, where one would like to leverage multiview data for training but perform classification or retrieval on single views. In general, it is not realistic to expect that multiple views of an object will be available at classification or retrieval time. We refer to this problem as *pose invariant classification and retrieval*. Figure 2 illustrates the limitations of existing approaches to address this problem.

View-based embeddings do not leverage multiple object views during training, treating all views of all objects in the same class equally. In result, as illustrated in Figure 2 a), there is no guarantee that these embeddings will cluster views from same object. While clustering views into classes, they are free to intertwine the views of different objects in the same class. On the other hand, multiview embeddings (6) only constrain the shape descriptor, i.e. the *average* of single view embedding. As illustrated by Figure 2 b), where shape descriptors are denoted by a '+', this suffices to produce a good shape descriptor clustering. However, it does not guarantee a good clustering of all individual views from an object. Note that the shape descriptors are all correctly classified, but this is not the case for the individual views, which can spread across class boundaries. This is illustrated by the dashed circles, which identify the distribution of images of each object. Due to this problem, multiview approaches tend to underperform the single view embeddings of a) for single view classification and retrieval [12, 6].

In order to address these problem, a new form of embeddings is needed. Figure 2 c) shows the behavior desired for a truly invariant embedding, which should be both single view invariant and multiview invariant. We denote this new form of embedding as *pose invariant embedding (PIE)*. PIE guarantees two properties: that 1) single view embeddings (image descriptors) of an object are clustered around multiview embedding (shape descriptor) and 2) multiview embedding is clustered around the descriptor of its labeled class.

To guarantee the two properties, we return to the probabilistic formulation and introduce an intermediate object variable O, leading to

$$\begin{aligned} P_{Y|\mathbf{X}}(y|\mathbf{x}) &= \sum_n P_{Y|O,\mathbf{X}}(y|n, \mathbf{x}) P_{O|\mathbf{X}}(n|\mathbf{x}) \\ &= \sum_n P_{Y|O}(y|n) P_{O|\mathbf{X}}(n|\mathbf{x}) \end{aligned} \quad (10)$$

where we have used the fact that once the object is known the class is independent of the view. This provides a decom-

position of the posterior probabilities into an object-to-class $P_{Y|O}(y|n)$ and a view-to-object $P_{O|\mathbf{X}}(n|\mathbf{x})$ model. This decomposition can be exploited to enforce the two properties above. We next discuss how to do this for the various approaches of Figure 1.

3.3. Pose invariant proxy embedding

We start by extending the proxy embedding [15] of (5) with the conditional probabilities of (10). We then note that the multiview form of proxy embedding, given by (9), is an object-to-class model, if the shape descriptors is produced by averaging image descriptors associated with the same object (6). Hence, the object-to-class model can be identical to the multiview proxy embedding (9)

$$P_{Y|O}(y|n) = s_y^m(\mathbf{X}_n; \mathbf{P}). \tag{11}$$

The view-to-object model should be similar to single view proxy (5) but use a set of object proxies. To encourages the clustering of Figure 2 c), we propose adopting the shape descriptor produced by (6) as the proxy for the associated object. This leads to the model

$$P_{O|\mathbf{X}}(n|\mathbf{x}) = \frac{e^{-d(g(\mathbf{x}),g_m(\mathbf{X}_n))}}{\sum_{j\neq n} e^{-d(g(\mathbf{x}),g_m(\mathbf{X}_j))}}. \tag{12}$$

Pose invariant proxy (PI-Proxy) embedding can then be derived by combining the two models with conditional probability (10). The approximated probabilities in [15] is then used and we have

$$s_y^{inv}(\mathbf{x}, \mathbf{P}) = \frac{\sum_n e^{-d^{inv}(\mathbf{x},\mathbf{X}_n,\mathbf{p}_y)}}{\sum_{i\neq y,n} e^{-d^{inv}(\mathbf{x},\mathbf{X}_n,\mathbf{p}_i)}}, \tag{13}$$

where

$$d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y) = \alpha d(g(\mathbf{x}), g_m(\mathbf{X}_n)) + \beta d(g_m(\mathbf{X}_n), \mathbf{p}_y) \tag{14}$$

is denoted as the *pose invariant* distance. α, β are two hyperparmeters that enable control over the contribution of the two components of the distance. Note that the feature extractor g is exactly the same as in the MV-CNN, i.e. there is no additional parameters and no change in the network.

3.4. Properties of pose invariant distance

The pose invariant distance of (14) has several properties of interest. First, setting $\alpha = 0$ and $\beta = 1$ results in the distance of the MV-proxy embedding (9), which leads to Figure 2 b). Second, for $\alpha = \beta = 1$, it becomes Figure 2 c) and follows from the triangle inequality that

$$\begin{aligned} d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y) &= d(g(\mathbf{x}), g_m(\mathbf{X}_n)) + d(g_m(\mathbf{X}_n), \mathbf{p}_y) \\ &\geq d(g(\mathbf{x}), \mathbf{p}_y), \end{aligned} \tag{15}$$

i.e. the invariant distance is an upper bound on the distance of the single view proxy. While the α term encourages clustering of individual views around the object (shape descriptor), the β term encourages clustering of objects into object class. Hence, the PI-Proxy embedding offers a range of solutions between the behaviors of Figure 2 b) and c).

3.5. Generating pose invariant embeddings

The procedure above can be generalized to all approaches of Figure 1 that use proxies. This is also true for classifiers, where the weights $\mathbf{w}_y$ of (1) play the role of proxies . The procedure for producing a pose invariant model is as follows.

1. use the multiview model as object-to-class model $P_{Y|O}(y|n)$.
2. use the view-based model as view-to-object model $P_{O|\mathbf{X}}(n|\mathbf{x})$.
3. replace the proxies of $P_{O|\mathbf{X}}(n|\mathbf{x})$ by the shape descriptors of (6). Use the shape descriptor of object O as proxy for this object.
4. use the conditional probability (10) to combine the two models into a pose invariant model.

Applying this procedure to the CNN of (1) leads to the *pose invariant-CNN* (PI-CNN)

$$h_y^{inv}(\mathbf{x}, y; \mathbf{W}) = \frac{\sum_n e^{d^{inv}(\mathbf{x},\mathbf{X}_n,\mathbf{p}_y)}}{\sum_{n,j} e^{d^{inv}(\mathbf{x},\mathbf{X}_n,\mathbf{p}_j)}}, \tag{16}$$

where $d^{inv}(\mathbf{x}, \mathbf{X}_n, \mathbf{p}_y)$ is defined as in (14). This is identical to the MV-CNN when $(\alpha, \beta) = (0, 1)$. For larger α, the classifier also discriminates between objects in the same class, assigning each view to the corresponding object descriptor. only assigns views to objects.

Applying the procedure to the triplet center approach, leads to the *pose invariant triplet center* (PI-TC) embedding. This combines the multiview triplet center distance of (7) and the triplet center loss of (8), using shape descriptor as centers, leading to the loss function

$$\begin{aligned} L(\mathbf{x}, y, \mathbf{P}) = \\ = \quad & \phi(\alpha(\min_{k\neq n} d(\mathbf{x}, \mathbf{X}_k) - d(\mathbf{x}, \mathbf{X}_n)) \\ + \quad & \beta(\min_{i\neq y} d(\mathbf{X}_n, \mathbf{p}_i) - d(\mathbf{X}_n, \mathbf{p}_y)))) \end{aligned} \tag{17}$$

3.6. Learning and inference

The models of (13), (16), and (17) are all functions of the view and multiview embeddings, g and g_m. However, since view feature extractor g is shared by all views and g_m is the average over view features given by (6), the total number of parameters is equal to that of a single CNN. In this aspect, all invariant embeddings of Figure 1 have the same

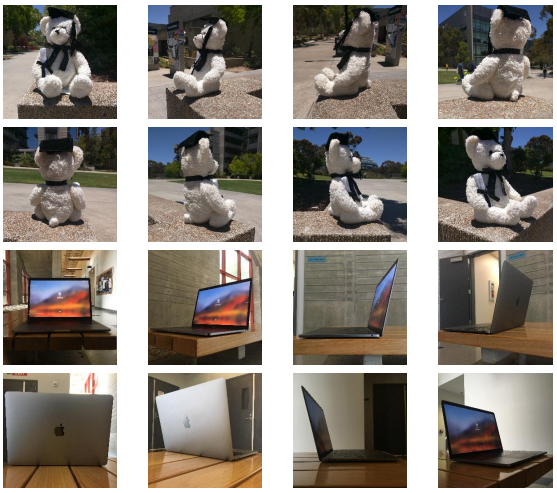


Figure 3. Examples of the 8 viewpoints of ObjectPI, for 2 objects.

complexity. Training boils down to learning the parameters of CNN, using (13), (16), and the logistic loss or (17) in risk $\mathcal{R}$ (2). This is a standard backpropagation learning problem. For inference, several modes are possible. In the multiview mode, only the model $P_{Y|O}(y|\mathbf{X})$ is used. This is equivalent to using the multivew methods in the second row of Figure 1, i.e. MV-CNN, MV-proxy (9), and MV-triplet center (7). However, these models can still benefit from invariant training. For pose invariant recognition and classification, the models are those of (13), (16), and (17). In the case where a single view $\mathbf{x}$ is available at inference time, i.e. $\mathbf{o}_n = g_m(\mathbf{X}_n)$ is not available, all expressions can be simplified. For example, the PI-CNN reduces to $h_y^{inv}(\mathbf{x}, y) = \frac{e^{-d(\mathbf{x},\mathbf{p}_y)}}{\sum_j e^{-d(\mathbf{x},\mathbf{p}_j)}}$. If partial views are available at inference time, the multiview mode is again used, but (6) is reformulated as $g_m(\mathbf{X}) = \frac{1}{V'}\sum_{k=1}^{V'} g(\mathbf{x}_k)$, where V' is the number of views available.

4. Pose invariance dataset

Existing multiview object datasets can be grouped in two classes. The first includes synthetic datasets such as ModelNet [29] or ShapeNet [4]. These are large and popular, but only depict computer graphics rendered objects. The second includes "turntable datasets", i.e. datasets imaged in the lab, by collecting images placed on a turntable, as it is rotated [2, 8, 16]. These are more realistic, but still lack natural backgrounds. In this work, we introduce a new dataset that addresses these limitations. It consists of images collected in the wild, by placing each object in a scene and taking pictures with a camera, which is moved around the object. An example of the views collected for an object is shown in Figure 3. The dataset contains 8 views per object, for 500 objects from 25 classes. These classes are chosen from ImageNet, to enable the use of CNNs pre-trained on the latter. The dataset is split into a training and test set, containing 16 and 4 objects respectively for each class. We refer to the dataset as the *object pose invariance* (ObjectPI)[1] dataset.

5. Experiments

In this section, we report on an experimental evaluation of the methods of Figure 1 on 5 different tasks, covering classification and retrieval at different levels of invariance.

5.1. Experimental setup

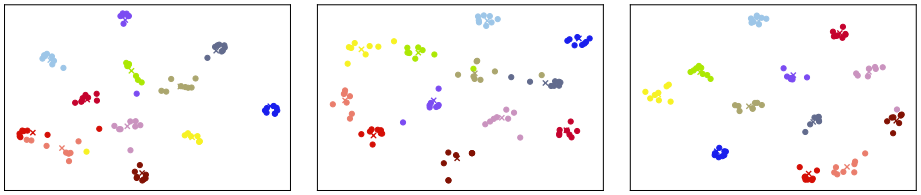
Dataset All experiments are based on three datasets.
ModelNet40 [29] is a 3D CAD dataset, of 40 object classes and 3183 objects . We use the training and testing splits of [22, 11], with 80 training and 20 test objects. For each object, 12 views are rendered uniformly (viewpoint interval 30 degree), identical to [22] and case (i) of [12]. Note that all reported results are for instance accuracy.
MIRO [12] is a dataset of real world objects. Each object is imaged from 10 elevations and 16 azimuths, to produce 160 images. We use the 16 images of 0^o elevation.
ObjectPI is described in Section 4.

Tasks All embeddings are tested on retrieval and classification and trained with all object views. Both single and multi-view inference are considered.
Classification: For CNN based methods, class is determined by the probabilities generated by the network, while for proxy and triplet center (TC) based methods, a nearest neighbor classifier is used. Classification accuracy is reported. **Single view classification** predicts the class of one image. **Multiview classification** predicts the class of a set of object views. For a CNN, this is done by averaging class probabilities over all views. For proxy and triplet center methods, a nearest neighbor classifier compares the shape descriptors extracted from the set of views to the class descriptors obtained from the training set.
Retrieval: Retrieval results are reported in terms of mean average precision (mAP). Three retrieval tasks are considered. **Single view retrieval** aims to retrieve images in the class of a query view. **Object retrieval** aims to retrieve other views of the object in the query view. These methods compare view descriptors. **Multiview retrieval** compares shape descriptors, aiming to retrieve the objects in the same class of the object used to generate a set of query views.
Implementation All experiments use a VGG16 [20] model implemented on Pytorch. For MV approaches, view pooling is performed before the softmax function. Learning rate is 1e-5 and Adam[13] optimizer is used in all experiments.

5.2. Joint classification and retrieval

The development of representations for joint classification and retrieval has shown to be difficult. Most methods

[1] All data collected in this work will be made available publicly.



Proxy MV-Proxy PI-Proxy ($\alpha = \beta = 1$)

Figure 4. TSNE visualization of proxy based embeddings on ObjectPI. Each dot is an object view, objects are identified by color, and their shape descriptors by 'x's.

Task		Proxy	MV-Proxy	PI-Proxy
Class.	Single	68.5	63.2	**68.7**
	Multi	78.8	78.3	**80.0**
(Acc.)	Avg	73.7	70.7	**74.4**
Retr.	Object	47.7	49.3	**49.4**
	Single	59.7	57.9	**62.6**
	Multi	76.8	74.7	**78.2**
(mAP)	Avg	61.4	60.6	**63.4**

Table 1. Proxy based methods on ObjectPI. $\alpha = \beta = 1$ for PI-Proxy.

specialize on one of the tasks, to the point that the papers do not even present results for the other. For example, [12] only addresses classification, while [11] is mainly designed for retrieval. The few works that report both classification and retrieval results use additional steps to prop at least one of the tasks. For example, [6, 22] train an additional low rank Mahalanobis metric to boost retrieval performance. In addition, only few methods report single image retrieval and classification result on classifier trained with multiview. It is simply accepted that view based embeddings have better performance for view classification and retrieval, while multiview embeddings are better for multiview classification and retrieval. It has so far not been shown that a single embedding can perform well on both tasks for both single and multiview.

Visualization: To study this issue in more detail, we consider the proxy based approaches of Figure 1, namely Proxy, MV-Proxy, and PI-Proxy. We start by visualizing, in Figure 4, the embeddings produced by the three approaches, using TSNE [24][2]. To simplify the plots, only 12 classes and 1 object per class are shown. Objects are identified by dots of the same color, which correspond to individual views. The shape descriptor of (6) is also shown as an 'x'. The classes and objects used in the visualization were chosen randomly. This plot confirms the predictions of Figure 2. While all methods succeed at separating the shape descriptors, the placement of individual views is very different. For Proxy and MV-Proxy, these may be embedded far away from the shape feature. MV-Proxy, which only optimizes the shape embedding (ignores the placement of views) produces the most scattered distribution. Proxy methods have more clustered embeddings, but the clustering is significantly inferior to that of PI-Proxy. In this case, most views cluster around the shape embedding produce object clusters of very small overlap. This is a direct consequence of the use of the pose invariant distance of (14).

Classification & Retrieval: Table 1 shows that PI-Proxy achieves the best performance of the three methods on all retrieval and classification tasks. While this is not surprising, given the clusterings of Figure 4, the differences can be quite significant, depending on the the task. Note that

[2]Similar TSNE visualizations of all approaches of Fig 1 can be found in the supplementary materials.

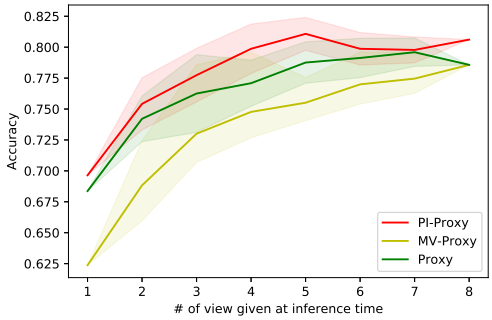


Figure 5. Classification accuracy of proxy based embedding on ObjectPI as a function of number of views at inference time.

MV-Proxy is particularly poor for single view classification. This is explained by the poor view clustering and is a well known limitation of multiview methods [6]. Proxy, is competitive with PI-Proxy on image classification, but inferior (2-3% points weaker) on the other tasks.

Robustness to number of views: Although multiview training improves classification accuracy [22], the latter often decreases dramatically for single view inference [6, 12]. In this setting multiview CNNs frequently underperform a standard single view classifier. This is unlike the proposed pose invariant embeddings, as shown in Figure 5.

The PI-Proxy embedding has performance comparable to that of MV-Proxy for multiple views, but much superior performance as the number of views decreases. This is again justified by the improved view clustering of Figure 4.

5.3. Comparison to the state of the art

We next performed a comparison of all embeddings of Figure 1 to other methods in the literature, on ModelNet, MIRO and ObjectPI datasets. Since most previous work has been done on ModelNet, we used the results on this dataset as guidance to select some state of the art models. It should be said that this is not easy, because the existing methods vary along many dimensions. This includes the use of different backbone network architectures (e.g. VGG-M instead of the more popular VGG16 that we adopt), architectural enhancements (e.g. view pooling layers that implement operations different from averaging view descriptor (6)) and complementary steps (e.g. optimizing the distance metric used for retrieval after the embedding is learned). All these variations are orthogonal to the invari-

Table 2. Comparison with state of the art methods on 3 different dataset for 5 different tasks on VGG16. The best result of each task is marked in bold and shadow denotes that the result of pose invariant based method is better or comparable than that of multiview based.

Method	ModelNet (12 views)							MIRO (16 views)						ObjectPI (8 views)						
	Classification (Accuracy %)			Retrieval (mAP %)				Classification (Accuracy %)			Retrieval (mAP %)			Classification (Accuracy %)			Retrieval (mAP %)			
	Single	Multi	Avg.	Object	Single	Multi	Avg.	Single	Multi	Avg.	Object	Single	Avg.	Single	Multi	Avg.	Object	Single	Multi	Avg.
RN[12]	80.2	89.0	84.6	22.6	20.2	63.9	35.6	93.2	**100**	96.6	33.0	33.0	33.0	37.5	63.2	50.3	40.1	25.2	41.9	35.7
MV-CNN[22]	71.0	87.9	79.4	29.6	41.7	71.5	47.6	**100**	**100**	**100**	92.0	92.0	92.0	62.1	74.1	68.1	42.6	53.8	72.3	56.2
PI-CNN	**85.4**	88.0	86.7	**50.8**	77.5	81.8	**70.0**	**100**	**100**	**100**	**100**	**100**	**100**	66.5	76.5	71.5	60.7	58.9	72.1	63.9
MV-TC[11]	77.3	88.9	83.1	36.6	63.5	84.0	61.4	**100**	**100**	**100**	99.8	99.8	99.8	65.7	79.2	72.4	51.8	59.5	77.3	62.9
PI-TC	81.2	88.9	85.1	41.4	71.5	84.2	65.7	**100**	**100**	**100**	**100**	**100**	**100**	**69.3**	77.5	73.2	**61.8**	**63.8**	76.7	**67.4**
MV-Proxy	79.7	**89.6**	84.7	35.0	66.1	**85.1**	62.1	**100**	**100**	**100**	99.8	99.8	99.8	63.2	78.3	70.7	49.3	57.9	74.7	60.6
PI-Proxy	85.1	88.7	**86.9**	40.6	**79.9**	**85.1**	68.6	**100**	**100**	**100**	**100**	**100**	**100**	68.7	**80.0**	**74.4**	49.4	62.6	**78.2**	63.4

ance issue studied in this work, and could be applied to any of the embeddings of Figure 1.

Furthermore, most existing methods only report results for few, sometimes even only one, of the 5 tasks that we consider. This allows for the detailed optimization of the embeddings for these tasks. Such optimization is not feasible under the experimental protocol now proposed, given the need to compare many embeddings on the 5 tasks and the goal of identifying embeddings that perform well across the 5 tasks. We believe that this is a set-up of greater practical significance, which future works in this area should adopt. Nevertheless, we used existing results to identify two state of the art models on ModelNet: the RotationNet (RN) [12] for classification and the triplet-center of [11] for retrieval. The later is what we denote by MV-triplet center (MV-TC) in Figure 1. For fair comparison, we re-trained these models under our set-up and tested them on the 5 tasks and 3 datasets that we now consider. For example, RN is re-trained with VGG16 instead of AlexNet[3]. We also present results for the other existing methods of Figure 1, namely the MV-CNN [22] and the proxy embedding of [15].

Table 2 summarizes the results of multiview and PIE based methods on the three datasets. Shadowed cells indicate that the PI-embedding outperforms the MV-embedding above it. Several conclusions can be drawn. First, pose invariant embeddings (PIEs) are clearly more robust than multiview embeddings (MVEs) on both classification and retrieval tasks. Among the 60 results listed in the table, PIEs outperformed MVEs on 46. In some cases, the difference was drastic. For example, for single view classification on ModelNet, the PI-CNN achieved 85.4% accuracy, outperforming the MVCNN by 14%. Second, one possibility to compare the performance of the different PIEs is to count the number of boldfaced entries. These indicate the number of "wins," i.e. how many times the method had equal or better performance than all others. Under this metric PI-proxy (12 wins) had slightly better performance, followed by PI-TC (10 wins), and PI-CNN (9 wins). However, the difference was not very significant. This shows that adding PIEs increases robustness regardless the approaches being used in the multiview level. Third, regarding classification vs. retrieval, the methods behave somewhat differently. While PI-Proxy achieved the best classification results on all datasets, PI-CNN had the best retrieval results in ModelNet and PI-TC on ObjectPI. However, the results of the three PIEs were close in most cases. Again, the most significant observation is how this differs from the behavior of the embeddings in the literature. For example, the RotationNet(RN) is competitive for classification but has very weak retrieval performance. Fourth, regarding datasets, best results were obtained on MIRO, then ModelNet, with ObjectPI posing the greatest challenge to most embeddings. This is not totally surprising, since MIRO and ModelNet have no backgrounds, MIRO is a relatively small dataset (120 objects), and ModelNet has no object textures. Nevertheless, these results confirm the need for a more realistic dataset, such as ObjectPI.

[3]Results of AlexNet model provided by [12] are reported in supplementary material.

6. Conclusion

This work makes several contributions to the study of pose invariance for image classification and retrieval tasks. We started by introducing a functional organization of embeddings to elaborate the relationships between existing methods. As the taxonomy is organized according to different level of invariance, some missing approaches are identified and existing approaches are further generalized. A new family of pose invariant embeddings (PIEs) is then derived from existing methods, by combining a view-to-object model and a object-to-class model. We show that the proposed PIEs have mathematically interesting properties and have good performance for both 1) classification and retrieval, and 2) single and multiview inference. The generalization of PIEs is important because such embeddings can be applied to different tasks and circumstances, which is a more realistic scenario for vision application. Finally, we introduced a multiview dataset, ObjectPI, with images of real objects captured with in the wild backgrounds. We believe that the proposed dataset will complement the synthetic datasets and contribute to the advancement of multiview study.

Acknowledgments This work was partially funded by NSF awards IIS-1546305 and IIS-1637941, a gift from Northrop Grumman, and NVIDIA GPU donations. We thank Brandon Leung, Erik Sandstroem, David Orozco and Yen Chang for the dataset collection.

References

[1] Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 34(4):98, 2015.

[2] A. Borji, S. Izadi, and L. Itti. ilab-20m: A large-scale controlled object dataset to investigate deep learning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2221–2230, June 2016.

[3] André Brock, Theodore Lim, James M. Ritchie, and Nick Weston. Generative and discriminative voxel modeling with convolutional neural networks. *CoRR*, abs/1608.04236, 2016.

[4] Angel X. Chang, Thomas A. Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qi-Xing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *CoRR*, abs/1512.03012, 2015.

[5] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 539–546. IEEE, 2005.

[6] Yifan Feng, Zizhao Zhang, Xibin Zhao, Rongrong Ji, and Yue Gao. Gvcnn: Group-view convolutional neural networks for 3d shape recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[7] A. Garcia-Garcia, F. Gomez-Donoso, J. Garcia-Rodriguez, S. Orts-Escolano, M. Cazorla, and J. Azorin-Lopez. Pointnet: A 3d convolutional neural network for real-time object class recognition. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 1578–1584, July 2016.

[8] Jan-Mark Geusebroek, Gertjan J Burghouts, and Arnold WM Smeulders. The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112, 2005.

[9] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. Neighbourhood components analysis. In *Advances in neural information processing systems*, pages 513–520, 2005.

[10] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognitionon*, pages 1735–1742. IEEE, 2006.

[11] Xinwei He, Yang Zhou, Zhichao Zhou, Song Bai, and Xiang Bai. Triplet-center loss for multi-view 3d object retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[12] Asako Kanezaki. Rotationnet: Learning object classification using unsupervised viewpoint estimation. *CoRR*, abs/1603.06208, 2016.

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[14] D. Maturana and S. Scherer. Voxnet: A 3d convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, Sept 2015.

[15] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[16] Sameer A Nene, Shree K Nayar, and Hiroshi Murase. Columbia object image library (coil-100).

[17] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4004–4012, 2016.

[18] Charles Ruizhongtai Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J. Guibas. Volumetric and multi-view cnns for object classification on 3d data. *CoRR*, abs/1604.03265, 2016.

[19] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[20] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[21] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in Neural Information Processing Systems*, pages 1857–1865, 2016.

[22] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015.

[23] Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *Advances in neural information processing systems*, pages 1988–1996, 2014.

[24] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[25] Chu Wang, Marcello Pelillo, and Kaleem Siddiqi. Dominant set clustering and pooling for multi-view 3d object recognition. In *BMVC*, 2017.

[26] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1386–1393, 2014.

[27] Kilian Q Weinberger, John Blitzer, and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2006.

[28] Jiajun Wu, Chengkai Zhang, Tianfan Xue, William T. Freeman, and Joshua B. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. *CoRR*, abs/1610.07584, 2016.

[29] Zhirong Wu, S. Song, A. Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and J. Xiao. 3d shapenets: A deep representation for volumetric shapes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, June 2015.

[30] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian. Foldingnet: Interpretable unsupervised learning on 3d point clouds. *CoRR*, abs/1712.07262, 2017.

[31] Haoxuan You, Yifan Feng, Rongrong Ji, and Yue Gao. Pvnet: A joint convolutional network of point cloud and multi-view for 3d shape recognition. *CoRR*, abs/1808.07659, 2018.