Improving Generalization via Scalable Neighborhood Component Analysis

Zhirong Wu[1,2], Alexei A. Efros[1], and Stella X. Yu[1]

[1] UC Berkeley / ICSI
[2] Microsoft Research Asia

Abstract. Current major approaches to visual recognition follow an end-to-end formulation that classifies an input image into one of the predetermined set of semantic categories. Parametric softmax classifiers are a common choice for such a closed world with fixed categories, especially when big labeled data is available during training. However, this becomes problematic for open-set scenarios where new categories are encountered with very few examples for learning a generalizable parametric classifier. We adopt a non-parametric approach for visual recognition by optimizing feature embeddings instead of parametric classifiers. We use a deep neural network to learn the visual feature that preserves the neighborhood structure in the semantic space, based on the Neighborhood Component Analysis (NCA) criterion. Limited by its computational bottlenecks, we devise a mechanism to use augmented memory to scale NCA for large datasets and very deep networks. Our experiments deliver not only remarkable performance on ImageNet classification for such a simple non-parametric method, but most importantly a more generalizable feature representation for sub-category discovery and few-shot recognition.

Keywords: k-nearest neighbors · large-scale object recognition · neighborhood component analysis · transfer learning · few-shot learning

1 Introduction

Deep learning with end-to-end problem formulations has reshaped visual recognition methods over the past few years. The core problems of high-level vision, e.g. recognition, detection and segmentation, are commonly formulated as classification tasks. Classifiers are applied image-wise for recognition [20], region-wise for detection [30], and pixel-wise for segmentation [23]. Classification in deep neural network is usually implemented as multi-way parametric softmax and assumes that the categories are fixed between learning and evaluation.

However, such a "closed-world" assumption does not hold for the open world, where new categories could appear, often with very few training examples. For example, for face recognition [43, 42], new identities should be recognized after just one-time occurrence. Due to the open-set nature, one may want to generalize the feature embedding instead of learning another parametric classifier. A

code & models available: https://github.com/zhirongw/snca.pytorch

common practice for embedding is to simply chop off the softmax classification layer from a pretrained network and take the last layer features. However, such a transfer learning scheme is not optimal because these features only make sense for a linear classification boundary in the training space, most likely not for the new testing space. Instead of learning parametric classifiers, we can learn an embedding to directly optimize a feature representation which preserves distance metrics in a non-parametric fashion. Numerous works have investigated various loss functions (e.g. contrastive loss [11], triplet loss [15, 26]) and data sampling strategies [50] for improving the embedding performance.

Non-parametric embedding approaches have also been applied to computer vision tasks other than face recognition. Exemplar-based models have shown to be effective for learning object classes [2] and object detection [25]. These non-parametric approaches build associations between data instances [24], and turn out to be useful for meta-knowledge transfer [25] which would not be readily possible for parametric models. So far, none of these non-parametric methods have become competitive in the state-of-the-art image recognition benchmarks such as ImageNet classification [31] and MSCOCO object detection [22]. However, we argue that time might be right to revisit non-parametric methods to see if they could provide the generalization capabilities lacking in current approaches.

We investigate a neighborhood approach for image classification by learning a feature embedding through deep neural networks. The core of our approach is a metric learning model based on Neighborhood Component Analysis (NCA) [9]. For each training image, NCA computes its distance to all the other images in the embedding space. The distances can then be used to define a classification distribution according to the class labels. Batch training with all the images is computationally expensive, thereby making the original NCA algorithm difficult to scale to large datasets. Inspired by prior works [51, 52], we propose to store the embedding of images in the entire dataset in an augmented non-parametric memory. The non-parametric memory is not learned by stochastic gradient descent, but simply updated after each training image is visited. During testing, we build a k-nearest-neighbor (kNN) classifier based on the learned metrics.

Our work makes three main contributions. 1) We scale up NCA to handle large-scale datasets and deep neural networks by using an augmented memory to store non-parametric embeddings. 2) We demonstrate that a nearest neighbor classifier can achieve remarkable performance on the challenging ImageNet classification benchmark, nearly on par with parametric methods. 3) Our learned feature, trained with the same embedding method, delivers improved generalization ability for new categories, which is desirable for sub-category discovery and few-shot recognition.

2 Related Works

Object Recognition. Object recognition is one of the holy grail problems in computer vision. Most prior works cast recognition either as a category naming problem [3, 4] or as a data association problem [24]. Category naming assumes

that all instances belonging to the same category are similar and that category membership is binary (either all-in, or all-out). Most of the research in this area is focused on designing better invariant category representations (e.g. bag-of-words [47], pictorial models [5]). On the other hand, data association approaches [2, 53, 24, **?**] regard categories as data-driven entities emergent from connections between individual instances. Such non-parametric paradigms are informative and powerful for transferring knowledge which may not be explicitly present in the labels. In the era of deep learning, however, the performance of exemplar-based approaches hardly reaches the state-of-the-art for standard benchmarks on classification. Our work revisits the direction of data association models, learning an embedding representation that is tailored for nearest neighbor classifiers.

Learning with Augmented Memory. Since the formulation of LSTM [14], the idea of using memory for neural networks has been widely adopted for various tasks [13]. Recent approaches on augmented memory fall into two camps. One camp incorporates memory into neural networks as an end-to-end differentiable module [10, 49], with automatic attention mechanism [33, 45] for reading and writing. These models are usually applied in knowledge-based reasoning [10, 45] and sequential prediction tasks [40]. The other camp treats memory as a non-parametric representation [44, 51, 52], where the memory size grows with the data set size. Matching networks [44] explore few-shot recognition using augmented memory, but their memory only holds the representations in current mini-batches of $5 - 25$ images. Our memory is also non-parametric, in a similar manner as storing instances for unsupervised learning [51]. The key distinction is that our approach learns the memory representation with millions of entries for supervised large-scale recognition.

Metric Learning. There are many metric learning approaches [18, 9], some achieving the state-of-the-art performance in image retrieval [50], face recognition [36, 42, 46], and person re-identification [52]. In such problems, since the classes during testing are disjoint from those encountered during training, one can only make inference based on its feature representation, not on the subsequent linear classifier. Metric learning learning encourages the minimization of intra-class variations and the maximization inter-class variations, such as contrastive loss [1, 39], triplet loss [15]. Recent works on few-shot learning [44, 38] also show the utility of metric learning, since it is difficult to optimize a parametric classifier with very few examples.

NCA. Our work is built upon the original proposal of Neighborhood Component Analysis (NCA) [9] and its non-linear extension [32]. In the original version [32], the features for the entire dataset needs to be computed at every step of the optimization, making it computationally expensive and not scalable for large datasets. Consequently, it has been mainly applied to small datasets such as MNIST or for dimensionality reduction [32]. Our work is the first to demonstrate that NCA can be applied successfully to large-scale datasets.

3 Approach

We adopt a feature embedding framework for image recognition. Given a query image x, we embed it into the feature space by $v = f_\theta(x)$. The function $f_\theta(\cdot)$ here is formulated as a deep neural network parameterized by parameter θ learned from data D. The embedding v is then queried against a set of images in the search database D', according to a similarity metric. Images with the highest similarity scores are retrieved and information from these retrieved images can be transferred to the image x.

Since the classification process does not rely on extra model parameters, the non-parametric framework can naturally extend to images in novel categories without any model fine-tuning. Consider three settings of D'.

1. When $D' = D$, i.e., the search database is the same as the training set, we have closed-set recognition such as the ImageNet challenge.
2. When D' is annotated with labels different from D, we have open-set recognition such as sub-category discovery and few-shot recognition.
3. Even when D' is completely unannotated, the metric can be useful for general content-based image retrieval.

The key is how to learn such an embedding function $f_\theta(\cdot)$. Our approach builds upon NCA [9] with some of our modifications.

3.1 Neighborhood Component Analysis

Non-parametric formulation of classification. Suppose we are given a labeled dataset of n examples $x_1, x_2, ..., x_n$ with corresponding labels $y_1, y_2, ..., y_n$. Each example x_i is embedded into a feature vector $v_i = f_\theta(x_i)$. We first define similarity s_{ij} between instances i and j in the embedded space as cosine similarity. We further assume that the feature v_i is ℓ_2 normalized. Then,

$$s_{ij} = \cos(\phi) = \frac{v_i^T}{\|v_i\| \, \|v_j\|} = v_i^T v_j, \tag{1}$$

where ϕ is the angle between vector v_i, v_j. Each example x_i selects example x_j as its neighbor with probability p_{ij} defined as,

$$p_{ij} = \frac{\exp(s_{ij}/\sigma)}{\sum_{k \neq i} \exp(s_{ik}/\sigma)}, \quad p_{ii} = 0. \tag{2}$$

Note that each example cannot select itself as neighbors, i.e. $p_{ii} = 0$. The probability thus is called *leave-one-out* distribution on the training set. Since the range of the cosine similarity is in $[-1, 1]$, we add an extra parameter σ to control the scale of the neighborhood.

Let $\Omega_i = \{j | y_j = y_i\}$ denote the indices of training images which share the same label with example x_i. Then the probability of example x_i being correctly classified is,

$$p_i = \sum_{j \in \Omega_i} p_{ij}. \tag{3}$$

The overall objective is to minimize the expected negative log likelihood over the dataset,

$$J = \frac{1}{n}\sum_i J_i = -\frac{1}{n}\sum_i \log(p_i). \tag{4}$$

Learning proceeds by directly optimizing the embedding without introducing additional model parameters. It turns out that each training example depends on all the other exemplars in the dataset. The gradients of the objective J_i with respect to v_i is,

$$\frac{\partial J_i}{\partial v_i} = \frac{1}{\sigma}\sum_k p_{ik} v_k - \frac{1}{\sigma}\sum_{k\in\Omega_i} \tilde{p}_{ik} v_k, \tag{5}$$

and v_j where $j \neq i$ is,

$$\frac{\partial J_i}{\partial v_j} = \begin{cases} \frac{1}{\sigma}(p_{ij} - \tilde{p}_{ij})v_i, & j \in \Omega_i \\ \frac{1}{\sigma} p_{ij} v_i, & j \notin \Omega_i \end{cases} \tag{6}$$

where $\tilde{p}_{ik} = p_{ik} / \sum_{j\in\Omega_i} p_{ij}$ is the normalized distribution within the groundtruth category.

Differences from parametric softmax. The traditional parametric softmax distribution is formulated as

$$p_c = \frac{\exp(w_c^T v_i)}{\sum_j \exp(w_j^T v_i)}, \tag{7}$$

where each category $c \in \{1, 2, ..., C\}$ has a parametrized prototype w_c to represent itself. The maximum likelihood learning is to align all examples in the same category with the category prototype. However, in the above NCA formulation, the optimal solution is reached when the probability p_{ik} of negative examples ($k \notin \Omega_i$) vanishes. The learning signal does not enforce all the examples in the same category to align with the current training example. The probability of some positive examples ($k \in \Omega_i$) can also vanish so long as some other positives align well enough to *i-th* example. In other words, the non-parametric formulation does not assume a single prototype for each category, and such a flexibility allows learning to discover inherent structures when there are significant intra-class variations in the data. Eqn 5 explains how each example contributes to the learning gradients.

Computational challenges for learning. Learning NCA even for a single objective term J_i would require obtaining the embedding as well as gradients (Eqn 5 and Eqn 6) in the entire dataset. This computational demand quickly becomes impossible to meet for large-scale dataset, with a deep neural network learned via stochastic gradient descent. Sampling-based methods such as triplet loss [42] can drastically reduce the computation by selecting a few neighbors.

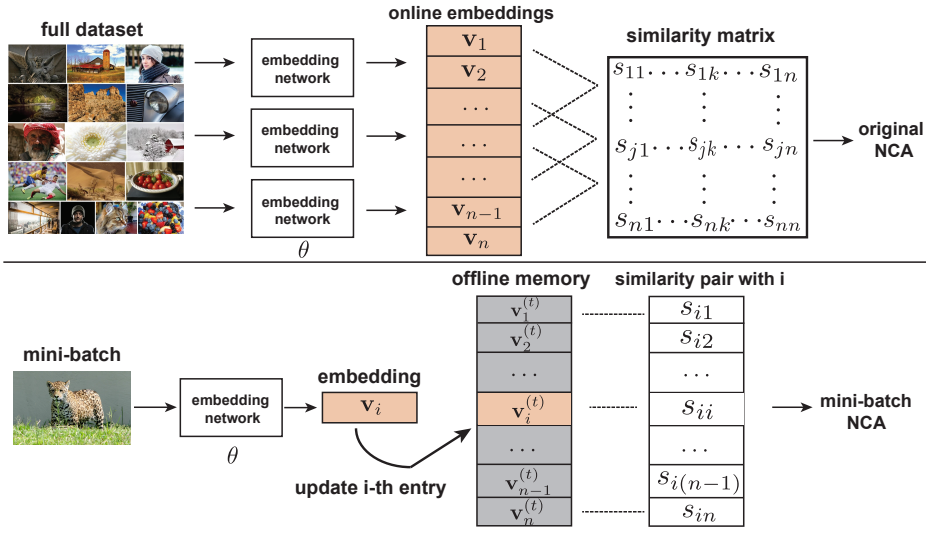

Fig. 1: The original NCA needs to compute the feature embeddings for the entire dataset for each optimization step. This is not scalable for large datasets and deep neural networks optimized with stochastic gradient descent. We overcome this issue by using an augmented memory to store offline embeddings forwarded from previous optimization steps. The online embedding is learned by back-propagation, while the offline memory is not.

However, hard-negative mining turns out to be crucial and typical batch size with 1800 examples [42] could still be impractical.

We take an alternative approach to reduce the amount of computation. We introduce two crude approximations.

1. We only perform gradient descent on $\partial J_i/\partial v_i$ as in Eqn 5, but not on $\partial J_i/\partial v_j$, $j \neq i$ as in Eqn 6. This simplification disentangles learning a single instance from learning among all the training instances, making mini-batch stochastic gradient descent possible.
2. Computing the gradient for $\partial J_i/\partial v_i$ still requires the embedding of the entire dataset, which would be prohibitively expensive for each mini-batch update. We introduce augmented memory to store the embeddings for approximation. More details follow.

3.2 Learning with Augmented Memory

We store the feature representation of the entire dataset as augmented non-parametric memory. We learn our feature embedding network through stochastic gradient descent. At the beginning of the t+1-*th* iteration, suppose the network parameter has the state $\theta^{(t)}$, and the non-parametric memory is in the form of

$M^{(t)} = \{v_1^{(t)}, v_2^{(t)}, ..., v_n^{(t)}\}$. Suppose that the memory is roughly up-to-date with the parameter $\theta^{(t)}$ at iteration t. This means the non-parametric memory is close to the features extracted from the data using parameter $\theta^{(t)}$,

$$v_i^{(t)} \approx f_{\theta^{(t)}}(x_i), \quad i = 1, 2, ..., n. \tag{8}$$

During the t+1-*th* optimization, for training instance x_i, we forward it through the embedding network $v_i = f_{\theta^{(t)}}(x_i)$, and calculate its gradient as in Eqn 5 but using the approximated embedding in the memory as,

$$\frac{\partial J_i}{\partial v_i} = \frac{1}{\sigma} \sum_k p_{ik} v_k^{(t)} - \frac{1}{\sigma} \sum_{k \in \Omega_i} \tilde{p}_{ik} v_k^{(t)}. \tag{9}$$

Then the gradients of the parameter can be back-propagated,

$$\frac{\partial J_i}{\partial \theta} = \frac{\partial J_i}{\partial v_i} \cdot \frac{\partial v_i}{\partial \theta}. \tag{10}$$

Since we have forwarded the x_i to get the feature v_i, we update the memory for the training instance x_i by the empirical weighted average [52],

$$v_i^{(t+1)} \leftarrow m \cdot v_i^{(t)} + (1 - m) \cdot v_i. \tag{11}$$

Finally, network parameter θ is updated and learned through stochastic gradient descent. If the learning rate is small enough, the memory can always be up-to-date with the change of parameters. The non-parametric memory slot for each training image is only updated once per learning epoch. Though the embedding is approximately estimated, we have found it to work well in practice.

3.3 Discussion on Complexity

In our model, the non-parametric memory $M^{(t)}$, similarity metric s_{ij}, and probability density p_{ij} may potentially require a large storage and pose computation bottlenecks. We give an analysis of model complexity below.

Suppose our final embedding is of size $d = 128$, and we train our model on a typical large-scale dataset using $n = 10^6$ images with a batch size of $b = 256$. Non-parametric memory M requires 0.5 GB ($O(dn)$) of memory. Similarity metric and probability density each requires 2 GB ($O(bn)$) of memory for storing the value and the gradient. In our current implementation, other intermediate variables used for computing the intra-class distribution require another 2 GB ($O(bn)$). In total, we would need 6.5 GB for the NCA module.

In terms of time complexity, the summation in Eqn 2 and Eqn 3 across the whole dataset becomes the bottleneck in NCA. However, in practice with a GPU implementation, the NCA module takes a reasonable 30% amount of extra time with respect to the backbone network. During testing, exhaustive nearest neighbor search with one million entries is also reasonably fast. The time it takes is negligible with respect to the forward passing through the backbone network.

The complexity of our model scales linearly with the training size set. Our current implementation can deal with datasets at the ImageNet scale, but cannot scale up to 10 times more data based on the above calculations. A possible strategy to handle bigger data is to subsample a few neighbors instead of the entire training set. Sampling would help reduce the linear time complexity to a constant. For nearest neighbor search at the run time, computation complexity can be mitigated with proper data structures such as ball-trees [8] and quantization methods [17].

4 Experiments

We conduct experiments to investigate whether our non-parametric feature embedding can perform well in the closed-world setting, and more importantly whether it can improve generalization in the open-world setting.

First, we evaluate the learned metric on the large-scale ImageNet ILSVRC challenge [31]. Our embedding achieves competitive recognition accuracy with k-nearest neighbor classifiers using the same ResNet architecture. Secondly, we study an important property of our representation for sub-category discovery, when the model trained with only coarse annotations is transferred for fine-grained label prediction. Lastly, we study how our learned metric can be transferred and applied to unseen object categories for few-shot recognition.

4.1 Image Classification

We study the effectiveness of our non-parametric representation for visual recognition on ImageNet ILSVRC dataset. We use the parametric softmax classification networks as our baselines.

Network Configuration. We use the ConvNet architecture ResNet[12] as the backbone for the feature embedding network. We remove the last linear classification layer of the original ResNet and append another linear layer which projects the feature to a low dimensional 128 space. The 128 feature vector is then ℓ_2 normalized and fed to NCA learning. Our approach does not induce extra parameters for the embedding network.

Learning Details. During training, we use an initial learning rate of 0.1 and drops 10 times smaller every 40 epochs for a total of 130 epochs. Our network converges a bit slower than the baseline network, in part due to the approximated updates for the non-parametric memory. We set the momentum for updating the memory with $m = 0.5$ at the start of learning, and gradually increase to $m = 0.9$ at the end of learning. We use a temperature parameter $\sigma = 0.05$ in the main results. All the other optimization details and hyper-parameters remain the same with the baseline approach. We refer the reader to the PyTorch implementation [28] of ResNet for details. During testing, we use a weighted k nearest neighbor classifier for classification. Our results are insensitive to parameter k; generally any k in the range of $5 - 50$ gives very similar results. We report the accuracy with $k = 1$ and $k = 30$ using single center crops.

Table 1: Top-1 classification rate on ImageNet validation set using k-nearest neighbor classifiers.

ResNet18				ResNet34				ResNet50			
Feature	d	k=1	k=30	Feature	d	k=1	k=30	Feature	d	k=1	k=30
Baseline	512	62.91	68.41	Baseline	512	67.73	72.32	Baseline	2048	71.35	75.09
+PCA	128	60.43	66.26	+PCA	128	65.58	70.67	+PCA	128	69.72	73.69
Ours	128	67.39	**70.58**	Ours	128	71.81	**74.43**	Ours	128	74.34	**76.67**

Table 2: Performance comparison of our method with parametric softmax.

Feature	baseline		ours	
	top-1	top-5	top-1	top-5
ResNet18	69.64	88.98	**70.58**	**89.38**
ResNet34	73.27	**91.43**	**74.43**	91.35
ResNet50	76.01	**92.93**	**76.67**	92.84

Table 3: Ablation study on the feature size and the temperature parameter.

d	k=1	k=30
256	67.54	70.71
128	67.39	70.59
64	65.32	69.54
32	64.83	68.01

σ	k=1	k=30
0.1	63.87	67.93
0.05	67.39	70.59
0.03	66.98	70.33
0.02	N/A	N/A

Main Results. Table 1 and Table 2 summarize our results in comparison with the features learned by parametric softmax. For baseline networks, we extract the last layer feature and evaluate it with the same k nearest neighbor classifiers. The similarity between features is measured by cosine similarity. Classification evaluated with nearest neighbors leads to a decrease of $6\% - 7\%$ accuracy with $k = 1$, and $1\% - 2\%$ accuracy with $k = 30$. We also project the baseline feature to 128 dimension with PCA for evaluation. This reduction leads to a further 2% decrease in performance, suggesting that the features learned by parametric classifiers do not work equally well with nearest neighbor classifiers. With our model, we achieve a 3% improvement over the baseline using $k = 1$. At $k = 30$, we have even slightly better results than the parametric classifier: Ours are 1.1% higher on ResNet34, and 0.7% higher on ResNet50. We also find that predictions from our model disagree with the baseline on 15% of the validation set, indicating a significantly different representation has been learned.

Figure 2 shows nearest neighbor retrieval comparisons. The upper four examples are our successful retrievals and the lower four are failure retrievals. For the failure cases, our model has trouble either when there are multiple objects in the same scene, or when the task becomes too difficult with fine-grained categorization. For the four failure cases, our model predictions are "paddle boat", "tennis ball", "angora rabbit", "appenzeller" respectively.

Ablation study on model parameters. We investigate the effect of the feature size and the temperature parameter in Table 3. For the feature size, 128 features and 256 features produce very similar results. We start to see performance degradation as the size is dropped lower than 64. For the temperature parameter, a lower temperature which induces smaller neighborhoods generally produces better results. However, the network does not converge if the temperature is too low, e.g., $\sigma = 0.02$.

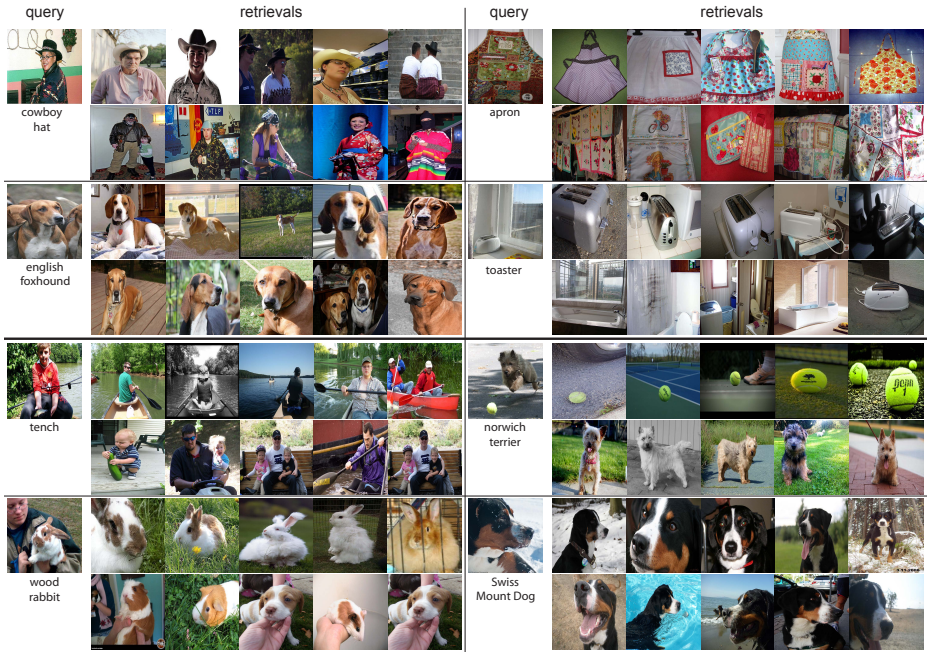

Fig. 2: Given a query, the figure shows 5 nearest neighbors from our model (1st row) and from the baseline model (2nd row). Top four examples show the successful cases and bottom four show the failure cases.

4.2 Discovering Sub-Categories

Our non-parametric formulation of classification does not assume a single prototype for each category. Each training image i only has to look for a few supporting neighbors [34] to embed the features. We refer nearest neighbors whose probability density $\sum_j p_{ij}$ sum over a given threshold as a support set for i. In Figure 3, we plot the histograms over the size of the support set for support density thresholds 0.5, 0.7 and 0.9. We can see most of the images only depend on around $100 - 500$ neighbors, which are a lot less than 1,000 images per category in ImageNet. These statistics suggest that our learned representation allows sub-categories to develop automatically.

The ability to discover sub-categories is of great importance for feature learning, as there are always intra-class variations no matter how we define categories. For example, even for the finest level of object species, we can further define object pose as sub-categories.

To quantitatively measure the performance of sub-category discovery, we consider the experiment of learning the feature embedding using coarse-grained object labels, and evaluating the embedding using fine-grained object labels. We can then measure how well feature learning discovers variations within categories. We refer this classification performance as induction accuracy as in [16]. We train

Table 4: Top-1 induction accuracy on CIFAR100 and ImageNet1000 using model pretrained on CIFAR20 and ImageNet127. Numbers are reported with k nearest neighbor classifiers.

CIFAR		
Task	20 classes	100 classes
Baseline	81.53	54.17
Ours	81.42	**62.32**

ImageNet		
Task	127 classes	1000 classes
Baseline	81.48	48.07
Ours	81.62	**52.75**

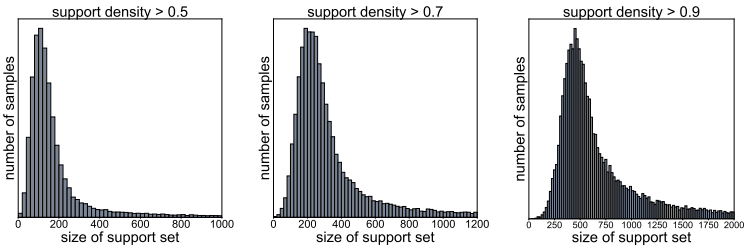

Fig. 3: Histogram of the size of support set in the ImageNet validation set given various support density thresholds.

the network with the baseline parametric softmax and with our non-parametric NCA using the same network architecture. To be fair with the baseline, we evaluate the feature from the penultimate layer from both networks. We conduct the experiments on CIFAR and ImageNet, and their results are summarized in Table 4.

CIFAR Results. CIFAR100 [19] images have both fine-grained annotations in 100 categories and coarse-grained annotations in 20 categories. It is a proper testing scenario for evaluating sub-category discovery. We study sub-category discovery by transferring representations learned from 20 categories to 100 categories. The two approaches exhibit similar classification performances on the 20 category setting. However, when transferred to CIFAR100 using k nearest neighbors, baseline features suffer a big loss, with 54.17% top-1 accuracy on 100 classes. Fitting a linear classifier for the baseline features gives an improved 58.66% top-1 accuracy. Using k nearest neighbor classifiers, our features are 8% better than the baselines, achieving a 62.32% recognition accuracy.

ImageNet Results. As in [16], we use 127 coarse categories by clustering the 1000 categories in a top-down fashion by fixing the distances of the nodes from the root node in the WordNet tree. There are 65 of the 127 classes present in the original 1000 classes. The other 62 classes are parental nodes in the ImageNet hierarchical word tree. The two models achieve similar classification performance ($81\% - 82\%$) on the original 127 categories. When evaluated with 1000 class annotations, our representation is about 5% better than the baseline features. The baseline performance can be improved to 52.0% by fitting another linear classifier on the 1000 classes.

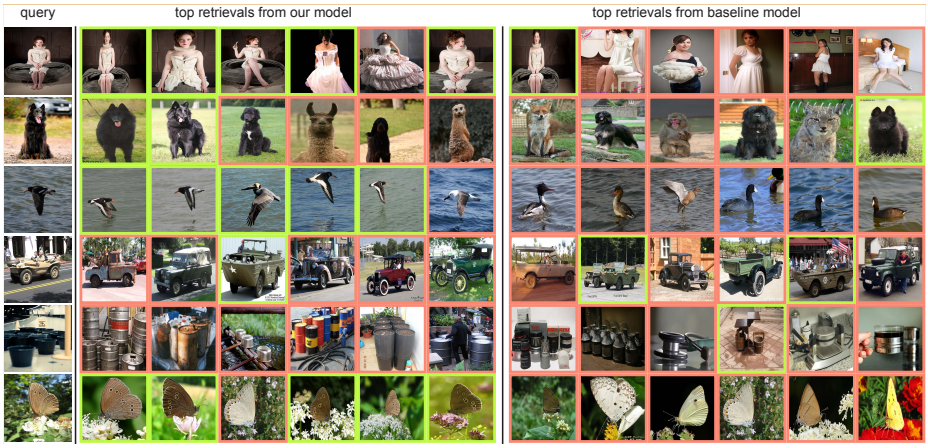

Fig. 4: Nearest neighbors from the models trained with ImageNet 127 classes and evaluated on the fine-grained 1000 classes. Correct retrievals are boxed with green outlines and wrong retrievals are with orange.

Discussions. Our approach is able to preserve visual structures which are not explicitly presented in the supervisory signal. In Figure 4, we show nearest neighbor examples compared with the baseline features. For all the examples shown here, the ground-truth fine-grained category does not exist in the training categories. Thus the model has to discover sub-categories in order to recognize the objects. We can see our representation preserves apparent visual similarity (such as color and pose information) better, and is able to associate the query with correct exemplars for accurate recognition. For example, our model finds similar birds hovering above water in the third row, and finds butterflies of the same color in the last row. In Figure 5 we further show the prediction gains for each class. Our model is particularly stronger for main sub-categories with rich intra-class variations.

4.3 Few-shot Recognition

Our feature embedding method learns a meaningful metric among images. Such a metric can be directly applied to new image categories which have not been seen during training. We study the generalization ability of our method for few-shot object recognition.

Evaluation Protocol. We use the mini-Imagenet dataset [44], which consists of 60,000 colour images and 100 classes (600 examples per class). We follow the split introduced previously [29], with 64, 16, and 20 classes for training, validation and testing. We only use the validation set for tuning model parameters. During testing, we create the testing episodes by randomly sampling a set of observation and query pairs. The observation consists of c classes (c-way) and s images (s-shot) per class. The query is an image from one of the$c$ classes. Each testing

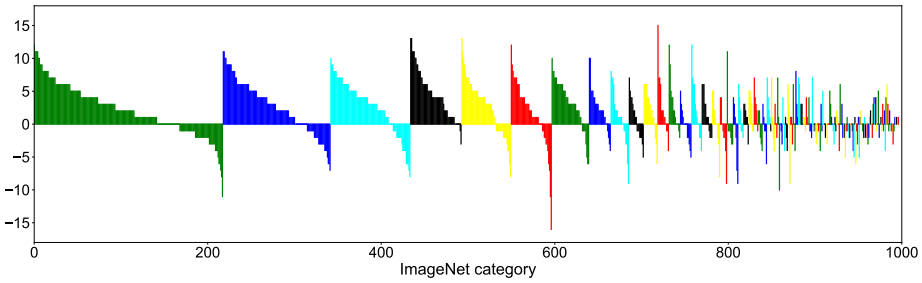

Fig. 5: Results for sub-category discovery on ImageNet. x axis scans through the fine-grained 1000 ImageNet categories. Each recycled color represents a coarse category. All coarse categories are sorted with decreasing order in terms of the number of sub-categories. y axis indicates the prediction gains of our model against the baseline model. Within each coarse category, the prediction gains for sub-categories are also sorted in a decreasing order.

Table 5: Few-shot recognition on Mini-ImageNet dataset.

Method	Network	FineTune	5-way Setting		20-way Setting	
			1-shot	5-shot	1-shot	5-shot
NN Baseline [44]	Small	No	41.1±0.7	51.0±0.7	-	-
Meta-LSTM [29]	Small	No	43.4±0.8	60.1±0.7	16.7±0.2	26.1±0.3
MAML [6]	Small	Yes	48.7±0.7	63.2±0.9	16.5±0.6	19.3±0.3
Meta-SGD [21]	Small	No	50.5±1.9	64.0±0.9	17.6±0.6	28.9±0.4
Matching Net [44]	Small	Yes	46.6±0.8	60.0±0.7	-	-
Prototypical [38]	Small	No	49.4±0.8	**68.2**±0.7	-	-
RelationNet [41]	Small	No	**51.4**±0.8	61.1±0.7	-	-
Ours	Small	No	50.3±0.7	64.1±0.8	**23.7**±0.4	**36.0**±0.5
SNAIL [27]	Large	No	55.7±1.0	68.9±0.9	-	-
RelationNet [41]	Large	No	57.0±0.9	71.1±0.7	-	-
Ours	Large	No	**57.8**±0.8	**72.8**±0.7	**30.5**±0.5	**44.8**±0.5

episode provides the task to predict the class of query image given $c \times s$ few shot observations. We create $3,000$ episodes for testing and report the average results.

Network Architecture. We conduct experiments on two network architectures. One is a shallow network which receives small 84×84 input images. It has 4 convolutional blocks, each with a $3 \times 3 \times 64$ convolutional layer, a batch normalization layer, a ReLU layer, and a max pooling layer. A final fully connected layer maps the feature for classification. This architecture is widely used in previous works [6, 44] for evaluating few-shot recognition. The other is a deeper version with ResNet18 and larger 224×224 image inputs. Two previous works [27, 41] have reported their performance with similar ResNet18 architectures.

Results. We summarize our results in Table 5. We train our embedding on the training set, and apply the representation from the penultimate layer for

Fig. 6: Few shot learning examples in mini-Imagenet test set. Given one shot for each five categories, the model predicts the category for the new query image. Our prediction is boxed with green and the baseline prediction is with orange.

evaluation. Our current experiment does not fine-tune a local metric per episode, though such adaptation would potentially bring additional improvement. As with the previous experiments, we use k nearest neighbors for classification. We use $k = 1$ neighbor for the 1-shot scenario, and $k = 5$ for the 5-shot scenario.

For the shallow network setting, while our model is on par with the prototypical network [38], and RelationNet [41], our method is far more generic.

For the deeper network setting, we achieve the state-of-the-art results for this task. MAML [6] suggests going deeper does not necessarily bring better results for meta learning. Our approach provides a counter-example: Deeper network architectures can in fact bring significant gains with proper metric learning.

Figure 6 shows visual examples of our predictions compared with the baseline trained with softmax classifiers.

5 Summary

We present a non-parametric neighborhood approach for visual recognition. We learn a CNN to embed images into a low-dimensional feature space, where the distance metric between images preserves the semantic structure of categorical labels according to the NCA criterion. We address NCA's computation demand by learning with an external augmented memory, thereby making NCA scalable for large datasets and deep neural networks. Our experiments deliver not only remarkable performance on ImageNet classification for such a simple non-parametric method, but most importantly a more generalizable feature representation for sub-category discovery and few-shot recognition. In the future, it's worthwhile to re-investigate non-parametric methods for other visual recognition problems such as detection and segmentation.

Acknowledgements

This work was supported in part by Berkeley DeepDrive. ZW would like to thank Yuanjun Xiong for helpful discussions.

References

1. Bromley, J., Guyon, I., LeCun, Y., Säckinger, E., Shah, R.: Signature verification using a" siamese" time delay neural network. In: NIPS (1994)
2. Chum, O., Zisserman, A.: An exemplar model for learning object classes. In: CVPR. IEEE (2007)
3. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR. IEEE (2009)
4. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. IJCV (2010)
5. Felzenszwalb, P.F., Huttenlocher, D.P.: Pictorial structures for object recognition. IJCV (2005)
6. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. arXiv preprint arXiv:1703.03400 (2017)
7. Freund, Y., Schapire, R.E.: A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences
8. Friedman, J.H., Bentley, J.L., Finkel, R.A.: An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software (TOMS) (1977)
9. Goldberger, J., Hinton, G.E., Roweis, S.T., Salakhutdinov, R.R.: Neighbourhood components analysis. In: NIPS (2005)
10. Graves, A., Wayne, G., Danihelka, I.: Neural turing machines. arXiv preprint arXiv:1410.5401 (2014)
11. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR. IEEE (2006)
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
13. Hinton, G., Deng, L., Yu, D., Dahl, G.E., Mohamed, A.r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., et al.: Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine (2012)
14. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation (1997)
15. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: International Workshop on Similarity-Based Pattern Recognition. Springer (2015)
16. Huh, M., Agrawal, P., Efros, A.A.: What makes imagenet good for transfer learning? arXiv preprint arXiv:1608.08614 (2016)
17. Jegou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. PAMI (2011)
18. Koestinger, M., Hirzer, M., Wohlhart, P., Roth, P.M., Bischof, H.: Large scale metric learning from equivalence constraints. In: CVPR. IEEE (2012)
19. Krizhevsky, A., Hinton, G.: Learning multiple layers of features from tiny images (2009)
20. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS (2012)
21. Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: Learning to learn quickly for few shot learning. arXiv preprint arXiv:1707.09835 (2017)
22. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. Springer (2014)

23. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR (2015)
24. Malisiewicz, T., Efros, A.A.: Recognition by association via learning per-exemplar distances. In: CVPR. IEEE (2008)
25. Malisiewicz, T., Gupta, A., Efros, A.A.: Ensemble of exemplar-svms for object detection and beyond. In: ICCV. IEEE (2011)
26. Mensink, T., Verbeek, J., Perronnin, F., Csurka, G.: Distance-based image classification: Generalizing to new classes at near-zero cost. PAMI (2013)
27. Mishra, N., Rohaninejad, M., Chen, X., Abbeel, P.: Meta-learning with temporal convolutions. arXiv preprint arXiv:1707.03141 (2017)
28. Paszke, A., Chintala, S., Collobert, R., Kavukcuoglu, K., Farabet, C., Bengio, S., Melvin, I., Weston, J., Mariethoz, J.: Pytorch: Tensors and dynamic neural networks in python with strong gpu acceleration, may 2017
29. Ravi, S., Larochelle, H.: Optimization as a model for few-shot learning (2016)
30. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NIPS (2015)
31. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015)
32. Salakhutdinov, R., Hinton, G.: Learning a nonlinear embedding by preserving class neighbourhood structure. In: Artificial Intelligence and Statistics (2007)
33. Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D., Lillicrap, T.: Meta-learning with memory-augmented neural networks. In: International conference on machine learning (2016)
34. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural computation (2001)
35. Scholkopf, B., Smola, A.J.: Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press (2001)
36. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: CVPR (2015)
37. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
38. Snell, J., Swersky, K., Zemel, R.: Prototypical networks for few-shot learning. In: NIPS (2017)
39. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: NIPS (2016)
40. Sukhbaatar, S., Weston, J., Fergus, R., et al.: End-to-end memory networks. In: NIPS (2015)
41. Sung, F., Yang, Y., Zhang, L., Xiang, T., Torr, P.H., Hospedales, T.M.: Learning to compare: Relation network for few-shot learning. arXiv preprint arXiv:1711.06025 (2017)
42. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: Deepface: Closing the gap to human-level performance in face verification. In: CVPR (2014)
43. Turk, M.A., Pentland, A.P.: Face recognition using eigenfaces. In: CVPR. IEEE (1991)
44. Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al.: Matching networks for one shot learning. In: NIPS (2016)
45. Vinyals, O., Fortunato, M., Jaitly, N.: Pointer networks. In: NIPS (2015)
46. Wang, H., Wang, Y., Zhou, Z., Ji, X., Li, Z., Gong, D., Zhou, J., Liu, W.: Cosface: Large margin cosine loss for deep face recognition. arXiv preprint arXiv:1801.09414 (2018)

47. Weber, M., Welling, M., Perona, P.: Unsupervised learning of models for recognition. In: ECCV. Springer (2000)
48. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. JMLR (2009)
49. Weston, J., Chopra, S., Bordes, A.: Memory networks. arXiv preprint arXiv:1410.3916 (2014)
50. Wu, C.Y., Manmatha, R., Smola, A.J., Krähenbühl, P.: Sampling matters in deep embedding learning. arXiv preprint arXiv:1706.07567 (2017)
51. Wu, Z., Xiong, Y., Stella, X.Y., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018)
52. Xiao, T., Li, S., Wang, B., Lin, L., Wang, X.: Joint detection and identification feature learning for person search. In: CVPR (2017)
53. Zhang, H., Berg, A.C., Maire, M., Malik, J.: Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In: CVPR. IEEE (2006)