DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition

Jeff Donahue*, **Yangqing Jia***, **Oriol Vinyals**, **Judy Hoffman**, **Ning Zhang**, **Eric Tzeng**, **Trevor Darrell**
{JDONAHUE,JIAYQ,VINYALS,JHOFFMAN,NZHANG,ETZENG,TREVOR}@EECS.BERKELEY.EDU
UC Berkeley & ICSI, Berkeley, CA, USA

Abstract

We evaluate whether features extracted from the activation of a deep convolutional network trained in a fully supervised fashion on a large, fixed set of object recognition tasks can be repurposed to novel generic tasks. Our generic tasks may differ significantly from the originally trained tasks and there may be insufficient labeled or unlabeled data to conventionally train or adapt a deep architecture to the new tasks. We investigate and visualize the semantic clustering of deep convolutional features with respect to a variety of such tasks, including scene recognition, domain adaptation, and fine-grained recognition challenges. We compare the efficacy of relying on various network levels to define a fixed feature, and report novel results that significantly outperform the state-of-the-art on several important vision challenges. We are releasing DeCAF, an open-source implementation of these deep convolutional activation features, along with all associated network parameters to enable vision researchers to be able to conduct experimentation with deep representations across a range of visual concept learning paradigms.

1. Introduction

Discovery of effective representations that capture salient semantics for a given task is a key goal of perceptual learning. Performance with conventional visual representations, based on flat feature representations involving quantized gradient filters, has been impressive but has likely plateaued in recent years.

It has long been argued that deep or layered compositional architectures should be able to capture salient aspects of a given domain through discovery of salient clusters, parts, mid-level features, and/or hidden units (Hinton & Salakhutdinov, 2006; Fidler & Leonardis, 2007; Zhu et al., 2007; Singh et al., 2012; Krizhevsky et al., 2012). Such models have been able to perform better than traditional hand-engineered representations in many domains, especially those where good features have not already been engineered (Le et al., 2011). Recent results have shown that moderately deep unsupervised models outperform the state-of-the art gradient histogram features in part-based detection models (Ren & Ramanan, 2013).

Deep models have recently been applied to large-scale visual recognition tasks, trained via back-propagation through layers of convolutional filters (LeCun et al., 1989). These models perform extremely well in domains with large amounts of training data, and had early success in digit classification tasks (LeCun et al., 1998). With the advent of large scale sources of category-level training data, e.g., (Deng et al., 2009), and efficient implementation with on-line approximate model averaging ("dropout") (Krizhevsky et al., 2012), they have recently outperformed all known methods on a large scale recognition challenge (Berg et al., 2012).

With limited training data, however, fully-supervised deep architectures with the representational capacity of (Krizhevsky et al., 2012) will generally dramatically overfit the training data. In fact, many conventional visual recognition challenges have tasks with few training examples; e.g., when a user is defining a category "on-the-fly" using specific examples, or for fine-grained recognition challenges (Welinder et al., 2010), attributes (Bourdev et al., 2011), and/or domain adaptation (Saenko et al., 2010).

In this paper we investigate semi-supervised multi-task learning of deep convolutional representations, where representations are learned on a set of related problems but applied to new tasks which have too few training examples to learn a full deep representation. Our model can either be considered as a deep architecture for transfer learning based on a supervised pre-training phase, or simply

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32. Copyright 2014 by the author(s). **Authors contributed equally.*

Caption Health Ex1053
Caption Health v. UBC
IPR2025-01422

as a new visual feature *DeCAF* defined by the convolutional network weights learned on a set of pre-defined object recognition tasks. Our work is also related to representation learning schemes in computer vision which form an intermediate representation based on learning classifiers on related tasks (Li et al., 2010; Torresani et al., 2010; Quattoni et al., 2008).

Our main result is the empirical validation that a generic visual feature based on a convolutional network weights trained on ImageNet outperforms a host of conventional visual representations on standard benchmark object recognition tasks, including Caltech-101 (Fei-Fei et al., 2004), the Office domain adaptation dataset (Saenko et al., 2010), the Caltech-UCSD Birds fine-grained recognition dataset (Welinder et al., 2010), and the SUN-397 scene recognition database (Xiao et al., 2010).

Further, we analyze the semantic salience of deep convolutional representations, comparing visual features defined from such networks to conventional representations. In Section 3, we visualize the semantic clustering properties of deep convolutional features compared to baseline representations, and find that convolutional features appear to cluster semantic topics more readily than conventional features. Finally, while conventional deep learning can be computationally expensive, we note that the run-time and resource consumption of deep-learned convolutional features are not exceptional, compared with features such as HOG (Dalal & Triggs, 2005) or KDES (Bo et al., 2010).

2. Related work

Deep convolutional networks have a long history in computer vision, with early examples showing successful results on using supervised back-propagation networks to perform digit recognition (LeCun et al., 1989). More recently, these networks, in particular the convolutional network proposed by Krizhevsky et al. (2012), have achieved competition-winning numbers on large benchmark datasets consisting of more than one million images, such as ImageNet (Berg et al., 2012).

Learning from related tasks also has a long history in machine learning beginning with Caruana (1997) and Thrun (1996). Later works such as Argyriou et al. (2006) developed efficient frameworks for optimizing representations from related tasks, and Ando & Zhang (2005) explored how to transfer parameter manifolds to new tasks. In computer vision, forming a representation based on sets of trained classifiers on related tasks has recently been shown to be effective in a variety of retrieval and classification settings, specifically using classifiers based on visual category detectors (Torresani et al., 2010; Li et al., 2010). A key question for such learning problems is to find a feature representation that captures the object category related information while discarding noise irrelevant to object category information such as illumination.

Transfer learning across tasks using deep representations has been extensively studied, especially in an unsupervised setting (Raina et al., 2007; Mesnil et al., 2012). However, reported successes with such models in convolutional networks have been limited to relatively small datasets such as CIFAR and MNIST, and efforts on larger datasets have had only modest success (Le et al., 2012). We investigate the "supervised pre-training" approach proven successful in computer vision and multimedia settings using a concept-bank paradigm (Kennedy & Hauptmann, 2006; Li et al., 2010; Torresani et al., 2010) by learning the features on large-scale data in a supervised setting, then transferring them to different tasks with different labels.

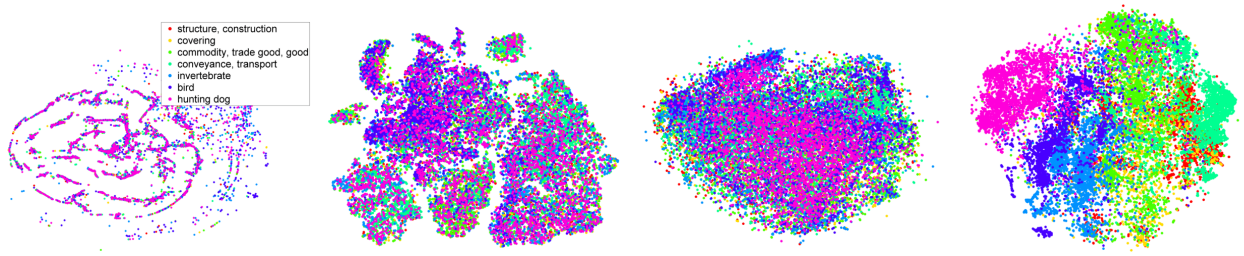
To evaluate the generality of a representation formed from a deep convolutional feature trained on generic recognition tasks, we consider training and testing on datasets known to have a degree of dataset bias with respect to ImageNet. We evaluate on the SUN-397 scene dataset, as well as datasets used to evaluate domain adaptation performance directly (Chopra et al., 2013; Kulis et al., 2011). This evaluates whether the learned features could undo the domain bias by capturing the real semantic information instead of overfitting to domain-specific appearances.

3. Deep Convolutional Activation Features

In our approach, a deep convolutional model is first trained in a fully supervised setting using a state-of-the-art method Krizhevsky et al. (2012). We then extract various features from this network, and evaluate the efficacy of these features on generic vision tasks. While the forward pass computed by the architecture discussed in this section does achieve state-of-the-art performance on ILSVRC-2012, at least two important questions remain: Do features extracted from the CNN generalize to other datasets? How does performance vary with network depth? We address these questions both qualitatively and quantitatively, via visualizations of semantic clusters below, and experimental comparision to current baselines in the following section.

3.1. An Open-source Convolutional Model

To facilitate the wide-spread analysis of deep convolutional features, we developed a Python framework that allows one to easily train networks consisting of various layer types and to execute pre-trained networks efficiently without being restricted to a GPU (which in many cases may hinder the deployment of trained models). Specifically, we adopted open-source Python packages such as `numpy/scipy` for efficient numerical computation, with



(a) LLC (b) GIST (c) DeCAF_1 (d) DeCAF_6

Figure 1. This figure shows several t-SNE feature visualizations on the ILSVRC-2012 validation set. (a) LLC , (b) GIST, and features derived from our CNN: (c) DeCAF_1, the first pooling layer, and (d) DeCAF_6, the second to last hidden layer (best viewed in color).

parts of the computation-heavy code implemented in C and linked to Python. Our implementation is able to process about 40 images per second with an 8-core commodity machine when the CNN model is executed in a minibatch mode.

Our implementation, `decaf`, is publicly available[1]. In addition, we have released the network parameters used in our experiments to allow for out-of-the-box feature extraction without the need to re-train the large network[2]. This also aligns with the philosophy of supervised transfer: one may view the trained model as an analog to the prior knowledge a human obtains from previous visual experiences, which helps in learning new tasks more efficiently.

As the underlying architecture for our feature we adopt the deep convolutional neural network architecture proposed by Krizhevsky et al. (2012), which won the ImageNet Large Scale Visual Recognition Challenge 2012 (Berg et al., 2012) with a top-1 validation error rate of 40.7%. [3] We chose this model due to its performance on a difficult 1000-way classification task, hypothesizing that the activations of the neurons in its late hidden layers might serve as very strong features for a variety of object recognition tasks. Its inputs are the mean-centered raw RGB pixel intensity values of a 224×224 image. These values are forward propagated through 5 convolutional layers (with pooling and ReLU non-linearities applied along the way) and 3 fully-connected layers to determine its final neuron activities: a distribution over the task's 1000 object categories. Our instance of the model attains an error rate of **42.9%** on the ILSVRC-2012 validation set – 2.2% shy of the 40.7% achieved by Krizhevsky et al. (2012).

We refer to Krizhevsky et al. (2012) for a detailed discussion of the architecture and training protocol, which we closely followed with the exception of two small differences in the input data. First, we ignore the image's original aspect ratio and warp it to 256×256, rather than resizing and cropping to preserve the proportions. Secondly, we did not perform the data augmentation trick of adding random multiples of the principle components of the RGB pixel values throughout the dataset, proposed as a way of capturing invariance to changes in illumination and color[4].

3.2. Feature Generalization and Visualization

We visualized the model features to gain insight into the semantic capacity of DeCAF and other features that have been typically employed in computer vision. In particular, we compare the features described in Section 3 with GIST features (Oliva & Torralba, 2001) and LLC features (Wang et al., 2010).

We visualize features in the following way: we run the t-SNE algorithm (van der Maaten & Hinton, 2008) to find a 2-dimensional embedding of the high-dimensional feature space, and plot them as points colored depending on their semantic category in a particular hierarchy. We did this on the validation set of ILSVRC-2012 to avoid overfitting effects (as the deep CNN used in this paper was trained only on the training set), and also use an independent dataset, SUN-397 (Xiao et al., 2010), to evaluate how dataset bias affects our results (see e.g. (Torralba & Efros, 2011) for a deeper discussion of this topic).

One would expect features closer to the output (softmax)

[1] `https://github.com/UCB-ICSI-Vision-Group/decaf-release`

[2] We note that although our CPU implementation allows one to also train networks, that training of large networks such as the ones for ImageNet may still be time-consuming on CPUs, and we rely on our own implementation of the network inside of the `cuda-convnet` GPU framework provided by Alex Krizhevsky to train such models.

[3] The model entered into the competition actually achieved a top-1 validation error rate of 36.7% by averaging the predictions of 7 structurally identical models that were initialized and trained independently. We trained only a single instance of the model; hence we refer to the single model error rate of 40.7%.

[4] According to the authors, this scheme reduced their models' test set error by over 1%, likely explaining much of our network's performance discrepancy.

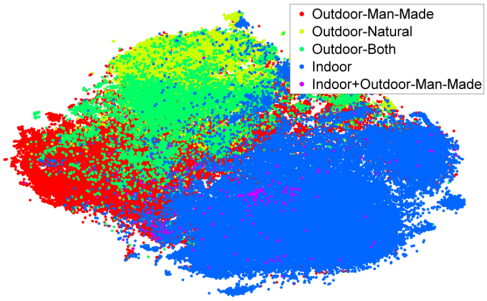


Figure 2. In this figure we show how our features trained on ILSVRC-2012 generalized to SUN-397 when considering semantic groupings of labels (best viewed in color).

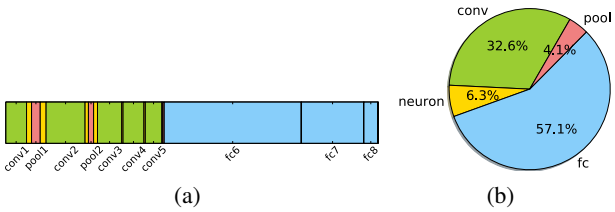


Figure 3. (a) The computation time on each layer when running classification on one single input image. The layers with the most time consumption are labeled. (b) The distribution of computation time over different layer types. In the piechart, fc = fully connected layers, conv = convolution layers, pool = pooling layers, and neuron = neuron layers such as ReLU, sigmoid, and dropout.

layer to be linearly separable, so it is not very interesting (and also visually quite hard) to represent the 1000 classes on the t-SNE derived embedding.

We first visualize the semantic segregation of the model by plotting the embedding of labels for higher levels of the WordNet hierarchy; for example, a strong feature for visual recognition should cluster indoor and outdoor instances separately, even though there is no explicit modeling through the supervised training of the CNN. Figure 1 shows the features extracted on the validation set using the first pooling layer, and the second to last fully connected layer, showing a clear semantic clustering in the latter but not in the former. This is compatible with common deep learning knowledge that the first layers learn "low-level" features, whereas the latter layers learn semantic or "high-level" features. Furthermore, other features such as GIST or LLC fail to capture the semantic difference in the image (although they show interesting clustering structure).[5]

More interestingly, in Figure 2 we can see the top performing features (DeCAF_6) on the SUN-397 dataset. Even there, the features show very good clustering of semantic classes (e.g., indoor vs. outdoor). This suggests DeCAF is a good feature for general object recognition tasks. Consider the case where the object class that we are trying to detect is not in the original object pool of ILSVRC-2012. The fact that these features cluster several intermediate nodes of WordNet implies that these features are an excellent starting point for generalizing to unseen classes.

3.3. Time Analysis

While it is generally believed that convolutional neural networks take a significant amount of time to execute, a detailed analysis of the computation time over the multiple layers involved is still missing in the literature. In this subsection we report a break-down of the computation time analyzed using the `decaf` framework.

In Figure 3(a) we lay out the computation time spent on individual layers with the most time-consuming layers labeled. We observe that the convolution and fully-connected layers take most of the time to run, which is understandable as they involve large matrix-matrix multiplications[6]. Also, the time distribution over different layer types (Figure 3(b)) reveals an interesting fact: in large networks such as the current ImageNet CNN model, the last few fully-connected layers require the most computation time as they involve large transform matrices. This is particularly important when one considers classification into a larger number of categories or with larger hidden-layer sizes, suggesting that certain sparse approaches such as Bayesian output coding (Hsu et al., 2009) may be necessary to carry out classification into even larger number of object categories.

4. Experiments

In this section, we present experimental results evaluating DeCAF on multiple standard computer vision benchmarks, comparing many possible featurization and classification approaches. In each of the experiments, we take the activations of the nth hidden layer of the deep convolutional neural network described in Section 3 as a feature DeCAF_n. DeCAF_7 denotes features taken from the final hidden layer – i.e., just before propagating through the final fully connected layer to produce the class predictions. DeCAF_6 is the activations of the layer before DeCAF_7, and DeCAF_5 the layer before DeCAF_6. DeCAF_5 is the first set of activations that has been fully propagated through the convolutional layers of the network. We chose not to evaluate features from any earlier in the network, as the

[5] Some of the features were very high dimensional (e.g. LLC had 16K dimension), in which case we preprocess them by randomly projecting them down to 512 dimensions – random projections are cheap to apply and tend to preserve distances well, which is all the t-SNE algorithm cares about.

[6] We implemented the convolutional layers as an `im2col` step followed by dense matrix multiplication, which empirically worked best with small kernel sizes and large number of kernels.

	DeCAF_5	DeCAF_6	DeCAF_7
LogReg	63.29 ± 6.6	84.30 ± 1.6	84.87 ± 0.6
LogReg with Dropout	-	86.08 ± 0.8	85.68 ± 0.6
SVM	77.12 ± 1.1	84.77 ± 1.2	83.24 ± 1.2
SVM with Dropout	-	$\mathbf{86.91 \pm 0.7}$	85.51 ± 0.9
Yang et al. (2009)		84.3	
Jarrett et al. (2009)		65.5	

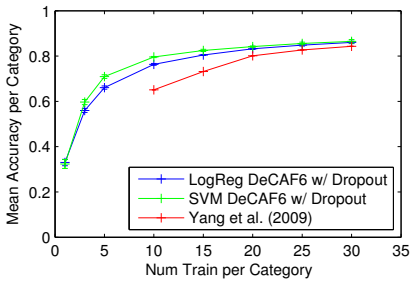


Figure 4. Left: average accuracy per class on Caltech-101 with 30 training samples per class across three hidden layers of the network and two classifiers. Our result from the training protocol/classifier combination with the best validation accuracy – SVM with Layer 6 (+ dropout) features – is shown in **bold**. Right: average accuracy per class on Caltech-101 at varying training set sizes.

earlier convolutional layers are unlikely to contain a richer semantic representation than the later features which form higher-level hypotheses from the low to mid-level local information in the activations of the convolutional layers. Because we are investigating the use of the network's hidden layer activations as features, all of its weights are frozen to those learned on the Berg et al. (2012) dataset.[7] All images are preprocessed using the procedure described for the ILSVRC images in Section 3, taking features on the center 224×224 crop of the 256×256 resized image.

We present results on multiple datasets to evaluate the strength of DeCAF for basic object recognition, domain adaptation, fine-grained recognition, and scene recognition. These tasks each differ somewhat from that for which the architecture was trained, together representing much of the contemporary visual recognition spectrum.

4.1. Object recognition

To analyze the ability of the deep features to transfer to basic-level object category recognition, we evaluate them on the Caltech-101 dataset (Fei-Fei et al., 2004). In addition to directly evaluating linear classifier performance on DeCAF_6 and DeCAF_7, we also report results using a regularization technique called "dropout" proposed by Hinton et al. (2012). At training time, this technique works by randomly setting half of the activations (here, our features) in a given layer to 0. At test time, all activations are multiplied by 0.5. Dropout was used successfully by Krizhevsky et al. (2012) in layers 6 and 7 of their network; hence we study the effect of the technique when applied to the features derived from these layers.

In each evaluation, the classifier, a logistic regression (LogReg) or support vector machine (SVM), is trained on a random set of 30 samples per class (including the background class), and tested on the rest of the data, with parameters cross-validated for each split on a 25 train/5 validation subsplit of the training data. The results in Figure 4, left, are reported in terms of mean accuracy per category averaged over five data splits.

Our top-performing method (based on validation accuracy) trains a linear SVM on DeCAF_6 with dropout, with test set accuracy of 86.9%. The DeCAF_5 features perform substantially worse than either the DeCAF_6 or DeCAF_7 features, and hence we do not evaluate them further in this paper. The DeCAF_7 features generally have accuracy about 1-2% lower than the DeCAF_6 features on this task. The dropout regularization technique uniformly improved results by 0-2% for each classifier/feature combination. When trained on DeCAF, the SVM and logistic regression classifiers perform roughly equally well on this task.

We compare our performance against the current state-of-the-art on this benchmark from Yang et al. (2009), a method employing a combination of 5 traditional hand-engineered image features followed by a multi-kernel based classifier. Our top-performing method outperforms this method by 2.6%. Our method also outperforms by over 20% the two-layer convolutional network of Jarrett et al. (2009), demonstrating the importance of the depth of the network used for our feature. Note that unlike our method, these approaches from the literature do not implicitly leverage an outside large-scale image database like ImageNet. The performance edge of our method over these approaches demonstrates the importance of multi-task learning when performing object recognition with sparse data like that available in the Caltech-101 benchmark.

We also show how performance of the two DeCAF_6 with dropout methods above vary with the number of training cases per category, plotted in Figure 4, right, trained with fixed parameters and evaluated under the same metric as before. Our one-shot learning results (e.g., 33.0% for SVM) suggest that with sufficiently strong representations like DeCAF, useful models of visual categories can often be learned from just a single positive example.

[7] We also experimented with the equivalent feature using randomized weights and found it to have performance comparable to traditional hand-designed features.

4.2. Domain adaptation

We next evaluate DeCAF for use on the task of domain adaptation. For our experiments we use the benchmark *Office* dataset (Saenko et al., 2010).The dataset contains three domains: `Amazon`, which consists of product images taken from `amazon.com`; and `Webcam` and `Dslr`, which consist of images taken in an office environment using a webcam or digital SLR camera, respectively.

In the domain adaptation setting, we are given a training (source) domain with labeled training data and a distinct test (target) domain with either a small amount of labeled data or no labeled data. We will experiment within the supervised domain adaptation setting, where there is a small amount of labeled data available from the target domain.

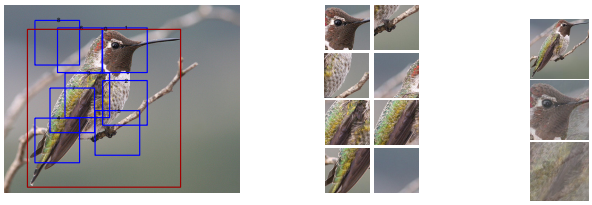
Most prior work for this dataset uses SURF (Bay et al., 2006) interest point features (available for download with the dataset). To illustrate the ability of DeCAF to be robust to resolution changes, we use the t-SNE (van der Maaten & Hinton, 2008) algorithm to project both SURF and DeCAF_6, computed for `Webcam` and `Dslr`, into a 2D visualizable space (See Figure 5). We visualize an image on the point in space corresponding to its low dimension projected feature vector. We find that DeCAF not only provides better within category clustering, but also clusters same category instances across domains. This indicates qualitatively that DeCAF removed some of the domain bias between the `Webcam` and `Dslr` domains.

We validate this conclusion with a quantitative experiment on the *Office* dataset. Table 1 presents multi-class accuracy averaged across 5 train/test splits for the domain shifts `Amazon`→`Webcam` and `Dslr` → `Webcam`. We use the standard experimental setup first presented in Saenko et al. (2010). To compare SURF with the DeCAF_6, and DeCAF_7 deep convolutional features, we report the multi-class accuracy for each, using an SVM and Logistic Regression both trained in 3 ways: with only source data (S), only target data (T), and source and target data (ST). We also report results for three adaptive methods run with each DeCAF we consider as input. Finally, for completeness we report a recent and competing deep domain adaptation result from Chopra et al. (2013). DeCAF dramatically outperforms the baseline SURF feature available with the *Office* dataset as well as the deep adaptive method of Chopra et al. (2013).

4.3. Subcategory recognition

We tested the performance of DeCAF on the task of subcategory recognition. To this end, we adopted one of its most popular tasks - the Caltech-UCSD birds dataset (Welinder et al., 2010), and compare the performance against several state-of-the-art baselines.

Following common practice in the literature, we adopted two approaches to perform classification. Our first approach adopts an ImageNet-like pipeline, in which we followed the existing protocol by cropping the images regions $1.5\times$ the size of the provided bounding boxes, resizing them 256×256 and then feeding them into the CNN pipeline to get the features for classification. We computed DeCAF_6 and trained a multi-class logistic regression on top of the features.

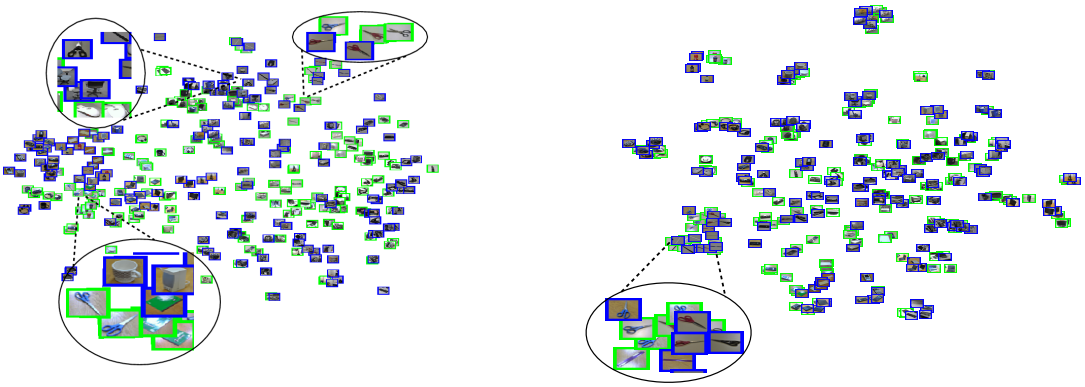


(a) DPM detections (b) Parts (c) DPD

Figure 6. Pipeline of deformable part descriptor (DPD) on a sample test images. It uses DPM for part localization and then use learned pooling weights for final pose-normalized representation.

Our second approach, we tested DeCAF in a pose-normalized setting using the deformable part descriptors (DPD) method (Zhang et al., 2013). Inspired by the deformable parts model (Felzenszwalb et al., 2010), DPD explicitly utilizes the part localization to do semantic pooling. Specifically, after training a weakly-supervised DPM on bird images, the pool weight for each part of each component is calculated by using the key-point annotations to get cross-component semantic part correspondence. The final pose-normalized representation is computed by pooling the image features of predicted part boxes using the pooling weights. Based on the DPD implementation provided by the authors, we applied DeCAF in the same pre-trained DPM model and part predictions and used the same pooling weights. Figure 6 shows the DPM detections and visualization of pooled DPD features on a sample test image. As our first approach, we resized each predicted part box to 256×256 and computed DeCAF_6 to replace the KDES image features (Bo et al., 2010) used in DPD paper.

Our performance as well as those from the literature are listed in Table 2. DeCAF together with a simple logistic regression already obtains a significant performance increase over existing approaches, indicating that such features, although not specifically designed to model subcategory-level differences, captures such information well. In addition, explicitly taking more structured information such as part locations still helps, and provides another significant performance increase, obtaining an accuracy of 64.96%, compared to the 50.98% accuracy reported in (Zhang et al., 2013). It also outperforms POOF (Berg & Belhumeur, 2013), to our knowledge the best accuracy reported in the literature prior to this work.



(a) SURF features (b) $DeCAF_6$

Figure 5. Visualization of the webcam (green) and dslr (blue) domains using the original released SURF features (a) and $DeCAF_6$ (b). The figure is best viewed by zooming in to see the images in local regions. All images from the scissor class are shown enlarged. They are well clustered and overlapping in both domains with our representation, while SURF only clusters a subset and places the others in disjoint parts of the space, closest to distinctly different categories such as chairs and mugs.

Method	Accuracy
$DeCAF_6$	58.75
DPD + $DeCAF_6$	**64.96**
DPD (Zhang et al., 2013)	50.98
POOF (Berg & Belhumeur, 2013)	56.78

Table 2. Accuracy on the Caltech-UCSD bird dataset.

We note again that in all the experiments above, no fine-tuning is carried out on the CNN layers since our main interest is to analyze how DeCAF generalizes to different tasks. To obtain the best possible result one may want to perform a full back-propagation. However, the fact that we see a significant performance increase without fine-tuning suggests that DeCAF may serve as a good off-the-shelf visual representation without heavy computation.

4.4. Scene recognition

Finally, we evaluate DeCAF on the SUN-397 large-scale scene recognition database (Xiao et al., 2010). Unlike object recognition, wherein the goal is to identify and classify an object which is usually the primary focus of the image, the goal of a scene recognition task is to classify the *scene* of the entire image. In the SUN-397 database, there are 397 semantic scene categories including *abbey*, *diner*, *mosque*, and *stadium*. Because DeCAF is learned on ILSVRC, an object recognition database, we are applying it to a task for which it was not designed. Hence we might expect this task to be very challenging for these features, unless they are highly generic representations of the visual world.

Based on the success of using dropout with $DeCAF_6$ and $DeCAF_7$ for the object recognition task detailed in Section 4.1, we train and evaluate linear classifiers on these dropped-out features on the SUN-397 database. Table 3 gives the classification accuracy results averaged across 5 splits of 50 training images and 50 test images. Parameters are fixed for all methods, but we select the top-performing method by cross-validation, training on 42 images and testing on the remaining 8 in each split.

Our top-performing method in terms of cross-validation accuracy was to use $DeCAF_7$ with the SVM classifier, resulting in 40.94% test performance. Comparing against the method of Xiao et al. (2010), the current state-of-the-art method, we see a performance improvement of 2.9% using only DeCAF. Note that, like the state-of-the-art method used as a baseline in Section 4.1, this method uses a large set of traditional vision features and combines them with a multi-kernel learning method. The fact that a simple linear classifier on top of our single image feature outperforms the multi-kernel learning baseline built on top of many traditional features demonstrates the ability of DeCAF to generalize to other tasks and its representational power as compared to traditional hand-engineered features.

5. Discussion

In this work, we analyze the use of deep features applied in a semi-supervised multi-task framework. In particular, we demonstrate that by leveraging an auxiliary large labeled object database to train a deep convolutional architecture, we can learn features that have sufficient representational power and generalization ability to perform semantic visual discrimination tasks using simple linear classifiers, reliably

	Amazon → Webcam			Dslr → Webcam		
	SURF	$DeCAF_6$	$DeCAF_7$	SURF	$DeCAF_6$	$DeCAF_7$
Logistic Reg. (S)	9.63 ± 1.4	48.58 ± 1.3	53.56 ± 1.5	24.22 ± 1.8	88.77 ± 1.2	87.38 ± 2.2
SVM (S)	11.05 ± 2.3	52.22 ± 1.7	53.90 ± 2.2	38.80 ± 0.7	91.48 ± 1.5	89.15 ± 1.7
Logistic Reg. (T)	24.33 ± 2.1	72.56 ± 2.1	74.19 ± 2.8	24.33 ± 2.1	72.56 ± 2.1	74.19 ± 2.8
SVM (T)	51.05 ± 2.0	78.26 ± 2.6	78.72 ± 2.3	51.05 ± 2.0	78.26 ± 2.6	78.72 ± 2.3
Logistic Reg. (ST)	19.89 ± 1.7	75.30 ± 2.0	76.32 ± 2.0	36.55 ± 2.2	92.88 ± 0.6	91.91 ± 2.0
SVM (ST)	23.19 ± 3.5	80.66 ± 2.3	79.12 ± 2.1	46.32 ± 1.1	$\mathbf{94.79 \pm 1.2}$	92.96 ± 2.0
Daume III (2007)	40.26 ± 1.1	$\mathbf{82.14 \pm 1.9}$	81.65 ± 2.4	55.07 ± 3.0	91.25 ± 1.1	89.52 ± 2.2
Hoffman et al. (2013)	37.66 ± 2.2	80.06 ± 2.7	80.37 ± 2.0	53.65 ± 3.3	93.25 ± 1.5	91.45 ± 1.5
Gong et al. (2012)	39.80 ± 2.3	75.21 ± 1.2	77.55 ± 1.9	39.12 ± 1.3	88.40 ± 1.0	88.66 ± 1.9
Chopra et al. (2013)		58.85			78.21	

Table 1. DeCAF dramatically outperforms the baseline SURF feature available with the *Office* dataset as well as the deep adaptive method of Chopra et al. (2013). We report average multi-class accuracy using both standard and adaptive classifiers, changing only the input feature from SURF to DeCAF. Surprisingly, in the case of Dslr→Webcam the domain shift is largely non-existent with DeCAF.

	$DeCAF_6$	$DeCAF_7$
LogReg	$\mathbf{40.94 \pm 0.3}$	40.84 ± 0.3
SVM	39.36 ± 0.3	40.66 ± 0.3
Xiao et al. (2010)	38.0	

Table 3. Average accuracy per class on SUN-397 with 50 training samples and 50 test samples per class, across two hidden layers of the network and two classifiers. Our result from the training protocol/classifier combination with the best validation accuracy – Logistic Regression with $DeCAF_7$ – is shown in **bold**.

outperforming current state-of-the-art approaches based on sophisticated multi-kernel learning techniques with traditional hand-engineered features. Our visual results demonstrate the generality and semantic knowledge implicit in these features, showing that the features tend to cluster images into interesting semantic categories on which the network was never explicitly trained. Our numerical results consistently and robustly demonstrate that our multi-task feature learning framework can substantially improve the performance of a wide variety of existing methods across a spectrum of visual recognition tasks, including domain adaptation, fine-grained part-based recognition, and large-scale scene recognition. The ability of a visual recognition system to achieve high classification accuracy on tasks with sparse labeled data has proven to be an elusive goal in computer vision research, but our multi-task deep learning framework and fast open-source implementation are significant steps in this direction. While our current experiments focus on contemporary recognition challenges, we expect our feature to be very useful in detection, retrieval, and category discovery settings as well.

Acknowledgements. We thank Alex Krizhevsky for his valuable help in reproducing the ILSVRC-2012 results, as well as for providing an open-source implementation of GPU-based CNN training. The Tesla K20 used in our experiments was donated by the NVIDIA Corporation. This research was supported by NSF, DoD, Toyota, and the Berkeley Vision and Learning Center sponsors. An earlier version of this work was made available as a technical report (Donahue et al., 2013).

References

Ando, R. and Zhang, T. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR*, 6, 2005.

Argyriou, Andreas, Evgeniou, Theodoros, and Pontil, Massimiliano. Multi-task feature learning. In *NIPS*, 2006.

Bay, H., Tuytelaars, T., and Gool, L. Van. SURF: Speeded up robust features. In *ECCV*, 2006.

Berg, A., Deng, J., and Fei-Fei, L. ImageNet large scale visual recognition challenge 2012. 2012. URL http://www.image-net.org/challenges/LSVRC/2012/.

Berg, T. and Belhumeur, P. POOF: Part-based one-vs-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*, 2013.

Bo, L., Ren, X., and Fox, D. Kernel descriptors for visual recognition. In *NIPS*, 2010.

Bourdev, Lubomir, Maji, Subhransu, and Malik, Jitendra. Describing people: A poselet-based approach to attribute classification. In *ICCV*, 2011.

Caruana, R. Multitask learning. *Machine Learning*, 28, 1997.

Chopra, S., Balakrishnan, S., and Gopalan, R. Dlid: Deep learning for domain adaptation by interpolating between domains. In *ICML Workshop on Challenges in Representation Learning*, 2013.

Dalal, N. and Triggs, B. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

Daume III, H. Frustratingly easy domain adaptation. In *ACL*, 2007.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., and Darrell, T. DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition. *arXiv e-prints*, 2013.

Fei-Fei, L., Fergus, R., and Perona, P. Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories. In *CVPR*, 2004.

Felzenszwalb, P., Girshick, R., McAllester, D., and Ramanan, D. Object detection with discriminatively trained part-based models. *PAMI*, 32, 2010.

Fidler, S. and Leonardis, A. Towards scalable representations of object categories: Learning a hierarchy of parts. In *CVPR*, 2007.

Gong, B., Shi, Y., Sha, F., and Grauman, K. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

Hinton, G. and Salakhutdinov, R. Reducing the dimensionality of data with neural networks. *Science*, 2006.

Hinton, G., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Hoffman, J., Rodner, E., Donahue, J., Saenko, K., and Darrell, T. Efficient learning of domain-invariant image representations. In *ICLR*, 2013.

Hsu, D., Kakade, S., Langford, J., and Zhang, T. Multi-label prediction via compressed sensing. *arXiv preprint arXiv:0902.1284*, 2009.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009.

Kennedy, L. and Hauptmann, A. LSCOM lexicon definitions and annotations (version 1.0). 2006.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

Kulis, B., Saenko, K., and Darrell, T. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011.

Le, Q., Zou, W., Yeung, S., and Ng, A. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR*, 2011.

Le, Q., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G., Dean, J., and Ng, A. Building high-level features using large scale unsupervised learning. In *ICML*, 2012.

LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., and Jackel, L. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1989.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. In *IEEE*, 1998.

Li, L., Su, H., Fei-Fei, L., and Xing, E. Object bank: A high-level image representation for scene classification & semantic feature sparsification. In *NIPS*, 2010.

Mesnil, G., Dauphin, Y., Glorot, X., Rifai, S., Bengio, Y., Goodfellow, I., Lavoie, E., Muller, X., Desjardins, G., Warde-Farley, D., Vincent, P., Courville, A., and Berkgstra, J. Unsupervised and transfer learning challenge: a deep learning approach. *JMLR*, 27, 2012.

Oliva, A. and Torralba, A. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 2001.

Quattoni, A., Collins, M., and Darrell, T. Transfer learning for image classication with sparse prototype representations. In *CVPR*, 2008.

Raina, R. at, Battle, A., Lee, H., Packer, B., and Ng, A. Self-taught learning: Transfer learning from unlabeled data. In *ICML*, 2007.

Ren, X. and Ramanan, D. Histograms of sparse codes for object detection. In *CVPR*, 2013.

Saenko, K., Kulis, B., Fritz, M., and Darrell, T. Adapting visual category models to new domains. In *ECCV*, 2010.

Singh, S., Gupta, A., and Efros, A. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, 2012.

Thrun, S. Is learning the n-th thing any easier than learning the first? In *NIPS*, 1996.

Torralba, A. and Efros, A. Unbiased look at dataset bias. In *CVPR*, 2011.

Torresani, L., Szummer, M., and Fitzgibbon, A. Efficient object category recognition using classemes. In *ECCV*. 2010.

van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *JMLR*, 9, 2008.

Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., and Perona, P. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Xiao, J., Hays, J., Ehinger, K., Oliva, A., and Torralba, A. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

Yang, J., L., Y., Tian, Y., Duan, L., and Gao, W. Group-sensitive multiple kernel learning for object categorization. In *ICCV*, 2009.

Zhang, N., Farrell, R., Iandola, F., and Darrell, T. Deformable part descriptors for fine-grained recognition and attribute prediction. In *ICCV*, 2013.

Zhu, L., Chen, Y., and Yuille, A. Unsupervised learning of a probabilistic grammar for object detection and parsing. In *NIPS*, 2007.