EXPLORING INVARIANCES IN DEEP CONVOLUTIONAL NEURAL NETWORKS USING SYNTHETIC IMAGES

Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko*
Department of Computer Science
University of Massachussetts Lowell
One University Avenue, Lowell, MA, USA
{xpeng,bsun,karim,saenko}@cs.uml.edu

ABSTRACT

Deep convolutional neural networks learn extremely powerful image representations, yet most of that power is hidden in the millions of deep-layer parameters. What exactly do these parameters represent? Recent work has started to analyse CNN representations, finding that, e.g., they are invariant to some 2D transformations, but are confused by particular types of image noise. In this paper, we delve deeper and ask: how invariant are CNNs to object-class variations caused by 3D shape, pose, and photorealism? These invariance properties are difficult to analyse using traditional data, so we propose an approach that renders synthetic data from freely available 3D CAD models. Using our approach we can easily generate an infinite amount of training images for almost any object. We explore the invariance of CNNs to various intra-class variations by simulating different rendering conditions, with surprising findings. Based on these results, we propose an optimal synthetic data generation strategy for training object detectors from CAD models. We show that our Virtual CNN approach significantly outperforms previous methods for learning object detectors from synthetic data on the benchmark PASCAL VOC2007 dataset.

1 INTRODUCTION

Deep convolutional neural networks have shown great potential for learning strong image representations. Large scale CNNs have achieved impressive gains on object classification (Krizhevsky et al., 2012), detection (Sermanet et al., 2013; Girshick et al., 2013; Simonyan & Zisserman, 2014) and as representations for many other tasks. These results indicate that CNN based features are significantly more powerful than their hand-designed counterparts like SIFT and HOG, and encode important invariances learned from data.

What is not as well known is the precise nature of these invariances. Besides tolerating object translation through explicit mechanisms of convolutions and pooling, the deep layers are likely learning to encode whichever invariances help with the task. Quantifying these invariances could help better understand the models and impove transfer to new domains, e.g., to non-photorealistic data. A small number of papers have started looking at this problem (Lenc & Vedaldi, 2014; Yosinski et al., 2014; Mahendran & Vedaldi, 2014), but many open questions remain, such as: are CNNs invariant to object color? texture? context? 3D pose? If so, which layers capture this information? Is it transferable to new tasks?

In this work, we propose the Virtual Convolutional Neural Network (VCNN), a method that learns from virtual 2D images generated using computer graphics (CG) techniques. Thanks to the flexibility of CG, we can probe the invariance of CNNs to factors that are difficult to isolate using 2D image data. For example, we can render images of objects without any texture, and then see if the network “hallucinates” the texture associated with that object shape.

Our goals in this work are two-fold. First, we design a series of experiments in order to “peer into the depths” of CNNs and analyse their invariance to several image formation factors. We make

*www.cs.uml.edu/~{xpeng,bsun,karim,saenko}
We explore the invariance of deep features learned on natural images to synthetic configurations of class shape, pose, texture and context. We then propose an improved method for learning from synthetic data generated from freely available 3D CAD models.

Our second goal is to leverage these observations to reduce training data annotation to a bare minimum, using synthetic images rendered from CAD models. Large-scale CNN frameworks are heavily dependent on large-scale training data. Unfortunately, challenge datasets provide only a limited number of annotated categories, e.g., 20 categories in PASCAL VOC (Everingham et al., 2010) and 200 in the more recent ImageNet (Deng et al., 2009). To train a robust detector for a novel category, an extensive training set must be compiled and annotated, taking care to cover as much variation as possible. We propose to bypass the expensive collection and annotation of real images and instead rely on freely available 3D CAD models to automatically generate virtual 2D training images. Sun & Saenko (2014) showed that DPMs (Felzenszwalb et al., 2010) learn equally well from such data, as they discard color and texture and model mostly the object outline/shape. As we discover, this is not always the case for CNNs.

Our experiments on the PASCAL VOC 2007 detection task show that when training data is limited or when there is no training data for a novel category, VCNN outperforms a similar model trained only on real data and the fast adaptation method of Sun & Saenko (2014). An advantage of our VCNN approach is that we can train a model for a novel object directly from its 3D shape. This could greatly expand available sources of visual knowledge and allow learning 2D detectors from the millions of CAD models available on the web. Surprisingly, we show that a VCNN trained with very limited real examples can achieve performance similar to that of DPM trained with extensive real data.

2 RELATED WORK

**Object Detection.** “Flat” hand-designed representations (HOG, SIFT, etc.) have dominated the object detection literature due to their considerable invariance to noise such as brightness, contrast and small translations. In combination with discriminative classifiers such as linear SVM, exemplar-based (Malisiewicz et al., 2011) or latent SVM (Felzenszwalb et al., 2010), they had proved powerful for learning to localize the global outline of an object. More recently, convolutional neural networks (LeCun et al., 1989) have overtaken flat features as clear front-runners in many image understanding tasks, including object detection. CNNs learn layered features starting with familiar pooled edges in the first layer, and progressing to more and more complex patterns with increasing spatial support. Extensions to detection have included sliding-window CNN (Sermanet et al., 2013) and Regions-CNN (RCNN) (Girshick et al., 2013).

**Understanding Deep CNNs.** There has been increasing interest in understanding the information encoded by the highly nonlinear deep layers. Zeiler & Fergus (2013) reversed the computation to find image patches that most highly activate an isolated neuron. A detailed study of what happens when one transfers network layers from one dataset to another was presented by Yosinski et al.
Mahendran & Vedaldi (2014) reconstruct an image from one layer’s activations, using image priors to recover the natural statistics removed by the network filters. Their visualizations confirm that a progressively more invariant and abstract representation of the image is formed by successive layers, but do not analyse the nature of the invariances. Invariance to simple 2D transformations (reflection, in-plane rotation) was explored by Lenc & Vedaldi (2014). In this paper, we study more complex invariances by “deconstructing” the image into 3D shape, texture, and other factors, and seeing which specific combinations result in representations discriminant of object categories.

Use of Synthetic Data. In the early days of computer vision, 3D models were used as the primary source of information to build object models (e.g., Nevatia & Binford (1977)). More recently, Stark et al. (2010); Liebelt & Schmid (2010); Sun et al. (2009) used 3D CAD models as their only source of labeled data, but limited their work to a few categories like cars and motorcycles. In this paper, we generate training data from crowdsourced 3D CAD models, which can be noisy and low-quality, but are free and available for many categories. We evaluate our approach on all 20 categories in the PASCAL VOC2007 dataset, which is much larger and more realistic than previous benchmarks. Previous works designed special features for matching synthetic 3D object models to real image data (Liebelt et al. (2008)), or used HOG features and linear SVMs (Sun & Saenko (2014)). We employ more powerful deep convolutional images features and demonstrate their advantage by directly comparing to Sun & Saenko (2014). Xiang et al. (2014) use CAD models and show results of both 2D detection and pose estimation, but train multi-view detectors not directly on 3D models but on real images labeled with pose. We avoid expensive manual bounding box annotation, and show results with minimum or no real image labels. Finally, several approaches had used synthetic training data for tasks other than object detection, for example, Jaderberg et al. (2014) recently proposed a synthetic text generation engine to perform text recognition in natural scenes.

3 EXPLORING THE INVARIANCES OF CNN FEATURES

In this section, we design an experimental paradigm to evaluate how variations in image formation factors affect the deep representation learned by the CNN. The main idea is to train classifiers on synthetic data using the network’s hidden layer activations as the feature representation, and test their performance on real image data.

Specifically, for each experiment, we follow these steps (see Figure 1): 1) select image formation parameters, 2) generate a batch of synthetic 2D images with those parameters, 3) sample positive and negative patches for each class, 4) extract hidden CNN layer activations from the patches as features, 5) train a classifier for each object category, 6) test the classifiers on real images.

We base our approach on the intuition that, if the network has learned to be invariant to a certain factor, then similar neurons will activate, whether or not that factor is present in the input image. For example, if the network is invariant to “cat” texture, then it will have similar activations on cats with and without texture, i.e. it will “hallucinate” the right texture when given a textureless cat shape. Then the classifier will learn equally well from both sets of training data. If, on the other hand, the network is not invariant to texture, then the feature distributions will differ. As a consequence, the classifier trained on textureless cat data will perform worse than the classifier trained on both shape and texture.

3.1 CNN MODEL AND TRAINING

We adopt the detection method of Girshick et al. (2013), which uses the eight-layer “AlexNet” architecture with over 60 million parameters Krizhevsky et al. (2012). This network had first achieved breakthrough results on the ILSVRC-2012 Berg et al. (2012) image classification, and remains the most studied and widely used visual convnet. The network is trained by fully supervised back-propagation (as in LeCun et al. (1989)) and takes raw RGB image pixels of a fixed size of 224 × 224 and outputs object category labels. The first five layers of the network have local spatial support and are convolutional, while the final three layers are fully-connected to each neuron from the previous layer, and thus include inputs from the entire image.

This network, originally designed for classification, was applied and fine-tuned to detection in Girshick et al. (2013) with impressive gains on the popular object detection benchmarks. To adapt AlexNet for detection, the RCNN applied the network to each image sub-region proposed by the
Selective Search method (van de Sande et al. (2013)), adding a background label, and applied non-maximal suppression to the outputs. Fine-tuning all hidden layers resulted in performance improvements. We refer the reader to Girshick et al. (2013) for more details.

Our hypothesis is that the network will learn different invariances, depending on how it is trained. Therefore, we evaluate two different variants of the network: one trained on the ImageNet ILSVRC 1000-way classification task, which we call IMGNET, and the same network also fine-tuned for the PASCAL detection task, which we call PASC-FT. For both networks, we extract the last hidden layer (fc7) as the feature representation. We choose to focus on the last hidden layer as it is the most high-level representation and has learned the most invariance.

3.2 Synthetic Data Generation

Object appearance is altered by many image formation factors, including object shape, pose, surface color, reflectance, location and spectral distributions of illumination sources, properties of the background scene, camera characteristics, etc. We choose a subset of factors that can easily be modeled using simple computer graphics techniques, namely, object texture and color, context/background appearance and color, 3D pose and 3D shape. We study the invariance of the CNN representation to these parameters using synthetic data. We also study the invariance to 3D rigid rotations using real data. The difference in treatment (using both synthetic and real data for pose) stems from the fact that while we can readily control for variations in 3D rigid rotations by making appropriate selections from existing datasets, the same cannot be said for object texture, color, shape and background appearance.

We downloaded CAD models from 3D Warehouse by searching for the name of 20 object categories represented in the PASCAL VOC Dataset. For each category, around 25 models were downloaded. We explore the effect of intra-class shape variation by restricting the number of models below. The original poses of the CAD models can be arbitrary (ex. upside-down chairs, or tilted cars). We therefore adjust the CAD models’s viewpoint manually to 3 or 4 “views”(as shown in Figure 1) that best represent intra-class pose variance for PASCAL objects. Next, for each manually specified model view, we generate several small perturbations by adding a random rotation. Finally, for each pose variation, we select the texture, color and background image and render a virtual image to include in our virtual training dataset. Next, we describe the detailed process for each factor, and present results.

3.3 Object Color, Texture and Context

We begin by investigating various combination of object colors and textures placed against a variety of background scene colors and textures. Previous work by Sun & Saenko (2014) has shown that when training a HOG model from virtual data, rendering natural backgrounds and texture is not helpful. The intuition is that a HOG-based classifier is focused on learning the “outlines” of the object shape. We hypothesise that the case is different for CNN representations, where certain neurons have been shown to respond to textures, colors and other mid-level patterns.

We examine the invariance of the CNN representation to two types of object textures, namely realistic color textures, and uniform grayscale textures (i.e., no texture at all). In the case of background scenes, we examine invariance to three types of scenes, namely real-image color scenes, real-image grayscale scenes, and a plain white background. Examples of our texture and background generation settings are shown in Table 1.

In order to simulate realistic object textures, we gathered 5 to 8 real images (per category) containing real objects and extracted the textures therein by annotating a bounding box. Likewise, in order to simulate realistic background scenes, we gathered about 40 (per category) real images of scenes where each category is likely to appear (e.g blue sky images for aeroplane, images of a lake or ocean for boat, etc.). When generating a virtual image, we first randomly select a background image from the available background pool and project it onto the image plane. Then, we select a random texture image from the texture pool and map it onto the CAD model before rendering the object. For this experiment, we used 1-2 pose perturbations per view and all views per category. The final number of images in the virtual training dataset is 2000, with 100 positive instances per each of the 20 categories.
Table 1: Detection results on the PASCAL VOC2007 test dataset. Each row is trained on different background and texture configuration of virtual data shown in the top table. In the middle table, the CNN is trained on ImageNet ILSVRC 1K classification data and finetuned on the PASCAL training data; in the bottom table, the network is not fine-tuned on PASCAL.

Figure 2: Top 10 regions with strongest activations for 2 pool5 units using the method of Girshick et al. (2013). Overlay of the unit’s receptive field is drawn in white and normalized activation value is shown in the upper-left corner. For each unit we show results on (top to bottom): real PASCAL images, RR-RR, W-RR, W-UG.
Table 2: Results of training on different synthetic views. The CNN used in the top table is trained on ImageNet-1K classification, the CNN in the bottom table is also finetuned on PASCAL 2007 detection.

To better explore the CNN’s invariance to color, texture and background, we visualize the patches which have the strongest activations for pool5 units, as shown in Figure 2. The value in the receptive field’s upper-left corner is normalized by dividing by max activation value over all units in a channel. The results are very interesting. The unit in the left subfigure fires on patches resembling tv-monitors in real images; when using our synthetic data, the unit still fires on tv-monitors even though the background and texture are removed. The unit on the right fires on white animals on green backgrounds in real and RR-RR images, and continues to fire on synthetic sheep with simulated texture, despite lack of green background. However, it fails on W-UG images, demonstrating its specificity to object color and texture.

3.4 Synthetic Pose

We also analyse the invariance of CNN features to 3D object pose. Through the successive operations of convolution and max-pooling, CNNs have a built-in invariance to translations and scale. Likewise, visualizations of learned filters at the early layers indicate a built-in invariance to local rotations. Thus while the CNN representation is invariant to slight translation, rotations and deformations, it remains unclear to what extent are CNN representation to large 3D rotations.

For this experiment, we fix the CAD models to three dominant poses: front-view, side-view and intra-view, as shown in Table 2. We change the number of views used in each experiment, but keep the total number of synthetic training images (RR-RR) exactly the same, by generating random small perturbations around the main view. Results indicate that for both networks adding side view to front view gives a boost, but improvement from adding the third view is marginal. We note that adding some views may even hurt performance (e.g., TV) as the PASCAL test set may not have objects in those views.

3.5 Real Image Pose

We also test view invariance on real images. We are interested here in objects whose frontal view presentation differs significantly (ex: the side-view of a horse vs a frontal view). To this end, we selected 12 categories from the PASCAL VOC training set which match this criteria. Held out categories included rotationally invariant objects such as bottles or tables. Next, we split the training data for these 12 categories to prominent side-view and front-view, as shown in Table 3.

We train classifiers exclusively by removing one view (say front-view) and test the resulting detector on the PASCAL VOC test set containing both side and front-views. We also compare with random view sampling. Results, shown in Table 3 point to important and surprising conclusions regarding the representational power of the CNN features. Note that mAP drops by less than 2% when detectors exclusively trained by removing either view are tested on the PASCAL VOC test set. Not only are those detectors never presented with the second view, but they are also trained with approximately half the data. While this invariance to large and complex pose changes may be explained by the fact the CNN model was itself trained with both views of the object present, and subsequently
Table 3: Results of training on different real image views. '-' represent removing a certain view.

<table>
<thead>
<tr>
<th>Net</th>
<th>Views</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASC-FT</td>
<td>all</td>
<td>64.2</td>
</tr>
<tr>
<td>PASC-FT</td>
<td>random</td>
<td>62.1</td>
</tr>
<tr>
<td>PASC-FT</td>
<td>-front</td>
<td>61.7</td>
</tr>
<tr>
<td>PASC-FT</td>
<td>-side</td>
<td>62.0</td>
</tr>
<tr>
<td>PASC-FT(-front)</td>
<td>-front</td>
<td>59.7</td>
</tr>
</tbody>
</table>

fine-tuned with both views again present, the level of invariance is nevertheless remarkable. In a last experiment, we reduce the fine-tuning training set by removing front-view objects, and note a larger mAP drop of 5 points (8%), but much less than one may expect. We conclude that, for both networks, the representation groups together multiple views of an object.

3.6 3D Shape

Finally, we experiment with reducing intra-class shape variation by using fewer CAD models per category. We otherwise use the same settings as in the RR-RR condition with PASC-FT. From our experiments, we find that the mAP decreases by about 5.5 points from 28.9% to 23.53% when using only a half of the 3D models. This shows a significant boost from adding more shape variation to the training data, indicating less invariance to this factor.

4 Virtual Convolutional Neural Network for Object Detection

To summarize the conclusions from the previous section, we found that CNNs learn a significant amount of invariance to texture, color and pose, if trained (or fine-tuned) on the same task. If not trained on the task, however, the degree of invariance is lower. Therefore, when learning a detection model for a new category with no or limited labeled real data available, it is preferable to simulate these factors in the synthetic data.

Based on these findings, we propose an approach for using CNNs with CAD-generated virtual images, which we call the Virtual CNN (VCNN). Our configuration uses all available 3D models and views, and selects a natural image background and real color image texture for each generated image (RR-RR in Table 1). We also propose to fine-tune the layers on the virtual data. Our VCNN model is similar to RCNN, with region proposals by selective search and supervised pre-training on ILSVRC. The difference is that, instead of depending on manually annotated real images, we use only the virtual images to fine-tune the pre-trained CNN, and use the features extracted by the fine-tuned VCNN to train the SVM detectors.

We note that there may be other advantages to using realistic backgrounds. Visualizations of the positive training data show that white background around the objects makes it harder to sample negative training data via selective search, as most of the interesting regions are on the object.

Learning novel categories. We present a set of experiments to simulate the situation where the number of labeled real images for a novel category is limited or zero. For every category, we randomly select 20 (10, 5) positive training images to form datasets \( R_{20} (R_{10}, R_{5}) \). The sizes of final datasets are 276 (120, 73); note that there are some images which contain two or more positive bounding boxes. The size of the virtual dataset (noted as \( V_{2k} \)) is always 2000. Our VCNN is pre-trained on Imagenet ILSVRC (IMAGENET network) and fine-tuned on \( V_{2k} \), then the SVMs are trained on both \( R_{x} + V_{2k} \). We compare the two best configurations from Table 1: RR-RR and RG-RR.

Baselines. We use datasets \( R_{x} (x = 20, 10, 5) \) to train the RCNN model, and \( R_{x} + V_{2k} \) to train the Fast Adaptation method described in Sun & Saenko (2014). The RCNN is pre-trained on Imagenet ILSVRC, however it is not fine-tuned on detection on \( R_{5} \) and \( R_{10} \) as data is very limited.
Results. The results in Figure 3 show that when the number of real training images is limited, VCNN will perform better than traditional RCNN. The VCNN also significantly outperforms the Fast-Adapt method, which is based on HOG features. We also confirm that our proposed RR-RR data synthesis methodology is better than not simulating background or texture. In particular, fine-tuning on virtual RR-RR data boosts mAP from 18.9% (Table 1) to 22% without using any real training examples, and to 28% with 5 real images per category, a 10% absolute improvement over RCNN. We also notice that the results for RG-RR are much lower than RR-RR, which is not very surprising. Unlike RR-RR, RG-RR with selective search generates many sub-regions without color, and using these regions to fine-tune likely decreases the CNN’s ability to recognize realistic color objects.

Note that the VCNN trained with 10 real images per category (200 total) is also using the approximately 900 real images of texture and background. However, this is still much fewer than the 15588 annotated bounding boxes in the PASCAL training set, and much easier to collect as only the texture images (about 130) need bounding box annotation. Yet the obtained 31% mAP is comparable to the 33% mAP achieved by the DPM (without context rescoring) trained on the full dataset. This speaks to the power of transferring deep representations and suggests that synthetic CAD data is a promising way to avoid tedious annotation for novel categories.

5 Conclusion

We investigated the sensitivity of convnets to various factors in the training data: 3D pose, foreground texture and color, background image and color. To simulate these factors we used synthetic data generated from 3D CAD models and a few real images.

Our results demonstrate that the popular deep convnet of Krizhevsky et al. (2012) fine-tuned for detection on real images for a set of categories is indeed invariant to these factors. Training on synthetic images with those variations leads to similar performance as training on synthetic images without those variations. However, if the network is not fine-tuned for the task on real images, its invariance is diminished. Thus for novel categories, adding synthetic variance along these dimensions and fine-tuning the layers proves useful. We stress that our experiments have only included categories in PASCAL which are a subset of those in the ILSVRC data used to learn the original network. Invariance properties of completely unseen categories are left for future work.

Finally, based on these findings, we proposed a new method for learning object detectors for new categories that avoids the need for costly large-scale image annotation. Our method generates syn-
thetic data from 3D CAD models and a few real images. This can be advantageous when one needs to learn a detector for a novel category or instance, beyond those available in labeled datasets.

REFERENCES


Sun, Baochen and Saenko, Kate. From virtual to reality: Fast adaptation of virtual object detectors to real domains. BMVC, 2014.


