Dataset Curation through Renders and Ontology Matching

Yair Movshovitz-Attias

CMU-CS-15-119

September 2015

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Yaser Sheikh, Co-Chair
Takeo Kanade, Co-Chair
Abhinav Gupta, Carnegie Mellon
Leonid Sigal, Disney Research
Trevor Darrell, University of California Berkeley

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.*

This research was sponsored by the Honda R&D Company LTD, the Air Force Office of Scientific Research under grant number FA95500810356, the Office of Naval Research under grant number N000141010934 and N000141410595, the National Science Foundation under grant number EEC-0540865, Wicab Inc., the Defense Information Systems Agency under grant number F8721-05-C-003, and the Toyota Motor Corporation under grant number 1542940. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Computer Vision, Viewpoint Estimation, Fine Pose Estimation, Fine-Grained Classification, Deep Learning, Correlation Filters, Rendering, Synthetic Data

For Dana

Abstract

In this thesis we demonstrate the benefits of *automated labeled dataset creation* for fine-grained visual learning tasks. Specifically, we show that utilizing real-world, non-image information can significantly reduce the human effort needed for building large scale datasets.

Computer vision has seen great advances in recent years in a number of complex tasks, such as scene classification, object detection, and image segmentation. A key ingredient in such success stories is the use of large amounts of labeled data. In many cases, the limiting factor is the ability to create these training sets. Issues arise in three forms: (1) The act of labeling the data can be hard for human annotators, (2) in some cases it is hard to get a representative sample of the feature space, and (3) data for infrequent (yet potentially important) instances can be completely absent from the training set.

Business storefront classification is an example of (1). The number of possible labels is large, and assigning all relevant labels to an image is a time consuming task for annotators. Moreover, when the image contains a business from a country other than their own, annotators can get confused by the foreign language and produce erroneous labels. Annotators are also not consistent in their categorization of businesses into categories.

In vehicle viewpoint estimation, the images themselves are hard to come by. Getting sample images of all viewpoints is hard due to bias in the way people photograph cars. Current datasets for this task lack data for many viewpoints. In addition, the labeling task is hard for the annotators.

We address these issues by adding automation to the dataset creation process. Our approach is to utilize external information by matching the images to real world concepts. In the case of businesses, when images are mapped to an ontology of geographical entities, we are able to extract multiple relevant labels per image. For the viewpoint estimation problem, by using 3D CAD models we can render images in the desired viewpoint resolution, and assign precise labels to them. We provide a systematic examination of the rendering process, and conclude that render quality is key for training accurate models.

Acknowledgments

First, I would like to thank my advisors, Yaser Sheikh and Takeo Kanade. Yaser played a critical role in shaping my research style. He taught me how to step back, think about the big picture, and ask deep research questions. Takeo has vast knowledge and experience in computer vision, and at times it feels like he has seen it all done before (and probably did it as well). However, he never fails being supportive and optimistic about new directions. He has an uncanny ability to compress great life advice into catchy one liners. Two of my favorites are: “Think like an amateur, do like an expert”, and “Write a paper that people would want to read”.

I am thankful to my committee members Abhinav Gupta, Leonid Sigal, and Trevor Darrell. Their comments and suggestions have made my thesis stronger. I had the opportunity to work with some really great collaborators. My group-mates: Varun Ramakrishna, Natasha Kholgade Banerjee, Tomas Simon Kreuz, Hanbyul Joo, Minh Vo, Hyun Soo Park, and Eakta Jain. Vishnu Boddeti, that introduced me to correlation filters, Martial Hebert who is not only insightful but also really entertaining to work with. The MARVIN team: Iljoo Baek, Taylor Stine, Denver Dash, Fanyi Xiao, and Mei Chen. I’ve had two great internships at Google and I’m grateful for all I have learned from my mentors there. Sergey Ioffe and Alexander Toshev for teaching me about Fisher Vectors and other pooling based image representations and Liron Yatziv, Qian Yu, and Martin Stumpe for starting me off on Deep Learning, many great coffee breaks, and various life advice. I loved TAing for Ariel Procaccia, Emma Brunskill, and Kris Kitani.

Life at grad school can be hard at times, but having great friends around makes it all so much better. John Wright, Sarah Loos and Jeremy Karnowski, Gabe and Cat Weisz, Danai Koutra and Walter Lasecki, Aaditya Ramdas, Kate Taralova, Sam Gottlieb, João Martins, David Henriques, Akshay Krishnamurthy, David Naylor, Galit Regev and Tzur Frenkel, Or Sheffet, Yuzi Nakamura, Mary Wootters, Deby Katz, Nika Haghtalab. Y’all are awesome and it would not have been fun without you! The administrative staff at CMU are great, but two stand out above all: Deb Cavlovich and and Catherine Copetas who know all and can fix all.

I have a great family which I would like to thank. My parents Aviva and Israel Movshovitz have always been amazingly encouraging. My brother Naor Movshovitz has had a profound impact on my life - he introduced me to *The Selfish Gene* and *Demon Haunted World*, two books that got me into science and critical thinking. This started me on the slippery slope that ended up with a PhD. My grandparents Jacob and Ira Movshovitz who created a family environment in which education was highly valued.

Lastly, it is hard to put into words my gratitude, love, and appreciation for my wife Dana Movshovitz-Attias. You play so many parts in my life: wife, partner, best friend, office-mate, debugging buddy, collaborator and co-author, therapist, procrastination partner, and someone to pontificate with about anything and everything. Life with you is always so much fun, I can't wait to see what happens next!

Contents

Chapter 1

Introduction

Computer science is going through a *Big Data* revolution. The explosion of visual data being created opens up many exciting opportunities in computer vision as well. Data seems cheap to get, and in many ways it is, but the process of creating a high quality labeled dataset from a mass of data is time-consuming and expensive.

In this thesis we focus on semi-automating the dataset creation process. We apply our approach to two computer vision tasks – object viewpoint estimation, and fine-grained classification of storefronts. For predicting the viewpoint of objects, we explore the use of synthetic data for training. We use state-of-the-art rendering software to generate a large dataset of cars rendered densely in viewpoint space, with automatically assigned labels. We show that combining synthetic with a small amount of real data can improve estimation accuracy. In other work, we employ a detailed ontology of business categories. We combine geo-location and textual information from images to match them to the ontology and automatically assign them with multiple labels during train time.

Making Use of Non-Image Information

Computer vision is closely related to Signal Processing. Many methods that are used in the analysis of one-variable signals can easily be extended, and adapted to two dimensional image data, or other multidimensional data common in Computer Vision such as SIFT [Lowe, 2004], HOG [Dalal and Triggs, 2005], or Optical Flow [Lucas and Kanade, 1981, Horn and Schunck, 1981]. This perspective provides an effective toolbox that computer vision researchers can turn to when faced with a new task; the specific scene or object depicted in the image does not need to be taken into consideration, and the image is processed using a combination of tools from machine learning, time-frequency analysis, statistics, etc. Indeed many of the most successful methods in fundamental computer vision problems such as object detection, and image classification employ the standard machine learning methodology: (1) gather a labeled set of training examples, (2) extract features (either hand crafted or learned), and finally (3) train a model to predict the desired quantity. Building methods that naturally generalize to different tasks is appealing; however, by taking into account the specific *concept* that is to be recognized one can employ added information that is missing in the visual signal.

In the natural language processing community *grounding* has been defined as a mapping between words or sentences and the observed world, many times in the form of an ontology of concepts/entities [Siskind, 1996]. We define *visual-grounding* as a mapping between image parts and the physical world. The visual-grounding process provides the signal processing procedure with more information, and enables improved accuracy.

For example, cars come in many different shapes and sizes, but realizing that some car models are manufactured by the same company, and created by the same designer can add to our understanding of the differences between them. In our work we show the benefits of considering the specific object in an image (specifically a car) when estimating its pose [Movshovitz-Attias et al., 2014]. We employ high quality 3D model to generate training images with automatic viewpoint label annotations for an ensemble of exemplar classifiers. Our results are better or comparable to the state of the art on a number of publicly available datasets.

Of the many successful uses of big data in computer vision, few have used the unsupervised paradigm of Machine Learning. Again and again, the power of labeled data has been demonstrated [Russakovsky et al., 2014, Goodfellow et al., 2013a, Taigman et al., 2014]. However, acquiring labeled data is expensive. Here, we explore ways to automate the creation of large scale labeled datasets. A key insight is that the automation process is not only faster and therefore cheaper, but in some cases allows us to generate labels that human annotators would not be able to produce. We will show that when the concept to be classified/detected is not abstracted away to a “signal”, but is instead mapped to a real-world object (visually-grounded), significant gains in performance can be achieved. We focus on the use of 3D CAD models as a convenient way to generate large quantities of training data, and on methods to utilize structured data in the form of large ontologies to automatically generate labels. We evaluate our methods on fine grained classification and detection tasks.

The Limitations of Human-Provided Annotations

Traditionally image datasets were collected mostly using graduate-student-powered techniques. That is, small teams of graduate students would collect – either in person, or by crawling image search – a set of images, and label it themselves. Projects such as LabelMe tried to involve the broad community in a common effort to create a resource that would benefit researchers everywhere. They provided an online labeling tool, shared all the labeled data, and in return asked researchers to contribute to the effort by labeling a small number of images themselves. The project provided a great resource to the community, but it relied heavily on the heroic efforts of one person, Adela Barriuso, who has single handily annotated more than 250,000 objects [Barriuso and Torralba, 2012]! If large datasets were needed, the community had to find better ways to scale up annotation.

The introduction of the Amazon Mechanical Turk crowdsourcing Internet marketplace in 2005 was a real game changer. The service enables individuals to coordinate the use of human intelligence through large amounts of transient workers. Researchers were now able to "farm out" their annotation tasks to remote, unknown, workers for relatively lower costs. This type of Crowdsourcing has been shown to be an effective way to annotate large numbers of images for training computer vision systems [Russakovsky et al., 2014]. However, crowdsourcing systems suffer from a number of limitations: Crowds normally participate asynchronously in the task, as workers join in and drop out of the system when they desire. This means lack of a tight feedback loop between workers, their task, and the task manager; packaging a task for display to operators requires substantial programing effort for creating the labeling interface, and continuous infrastructure for serving the task to the workers, and managing their responses. This is an interesting open research problem in the HCI community [Lasecki et al., 2014, 2013].

Even when one has the needed infrastructure, and is able to shoulder the programming effort, there are tasks which require expert knowledge that is not common in the worker pool, e.g. labeling breeds of dogs [Liu et al., 2012] or models of cars [Krause et al., 2013]. Other tasks do not require specific knowledge but are difficult for workers to answer precisely, for example labeling the angle in which an object is viewed in an image [Movshovitz-Attias et al., 2014], or the age of a person in a video [Fu and Huang, 2008]. In all these cases, methods that automate as much of the labeling process as possible will open up applications for which labeled data collection is the bottleneck. In [Movshovitz-Attias et al., 2015] we noticed that workers were inconsistent in their annotations of images of businesses. We used extracted text from images and the geographical location of the images to match them to known establishments in an ontology of businesses. Using the ontology we were able to resolve many of the inconsistencies and achieve high accuracy predictions.

1.1 Core Contribution

This thesis addresses one critical bottleneck in fine-grained visual learning – building large scale datasets. We propose to employ external knowledge to add automation to the dataset creation process. Fine-grained learning requires detailed training data. As the number of classes grows, and the similarity between them increases (they become more fine-grained) the labeling process becomes harder, and the required labels are more sophisticated. It becomes difficult to create the big labeled datasets needed for the task. We advocate for the use of external, non-image knowledge.

External knowledge can be characterized as belonging to one of two classes. Unstructured knowledge, such as web text or a search engine's query stream is abundant but is considerably messy and noisy. Structured knowledge, such as entity ontologies and 3D CAD models, is carefully created and organized by people. While it is of smaller quantities, it tends to be very detailed and of high quality. We focus on structured external knowledge and propose methods that automate the acquisition of data, but more importantly the process of associating this data with *labels*.

We address the task of object viewpoint estimation and explore how rendered synthetic data, generated from 3D CAD models, can be used to build large scale datasets without paying the large cost of human annotation. We provide an examination of the parameter space of the rendering process and outline which parameters impact performance. We explore the classification as well as detection aspects of this task and show how to intelligently combine rendered and real data to improve accuracy.

Lastly we explore the use of structured data for fine-grained classification of business storefronts. When an ontology of the concepts of interest is present, we match training images to entities from the ontology, and propagate class information to provide multiple labels per training instance. This allows us to automatically generate a large scale training set which is used to create models that match human performance.

1.2 Synopsis

The work in this thesis can be broken down into 3 parts: (1) exploratory use of synthetic data for viewpoint detection and estimation, (2) investigation of the properties of synthetic rendered data for the same task, i.e. what is it about rendered data that makes it useful? And (3) Bootstrapping dataset creation using ontology matching.

In the following sections we provide a short synopsis of the thesis, separated according to this breakdown.

1.2.1 3D Pose-by-Detection of Vehicles via Discriminatively Reduced Ensembles of Correlation Filters

Estimating the precise pose of a 3D object in an image is challenging; explicitly identifying correspondences is difficult, particularly at smaller scales and in the presence of occlusion. Exemplar classifiers have demonstrated the potential of detection-based approaches to problems in which high resolution prediction is required. In particular, correlation filters explicitly suppress classifier response caused by slight shifts in the bounding box. This property makes them ideal exemplar classifiers for viewpoint discrimination, as small translational shifts can often be confounded with small rotational shifts. However, collecting labeled training data with accurate labels at fine grained angle resolution is a difficult task. Furthermore, exemplar based pose-by- detection is not scalable because, as the desired precision of viewpoint estimation increases, the number of exemplars needed increases as well.

We leverage the availability of high quality 3D CAD models by creating realistic renders of vehicles that cover the viewpoint space with high angle resolution and present a training framework to reduce an ensemble of exemplar correlation filters for viewpoint estimation by directly optimizing a discriminative objective. We show that the discriminatively reduced ensemble outperforms the state-of-the-art on two publicly available datasets, and is competitive on a third. We further introduce a new dataset for continuous car pose estimation in street scene images.

1.2.2 Rendered Data Generation for Viewpoint Estimation using Deep Convolutional Networks

Our experience with utilizing rendered images from CAD models for detection tasks [Movshovitz-Attias et al., 2014] convinced us that there is much to gain by expanding their use to more objects and more tasks; however, recent work has found conflicting evidence of the usefulness of rendered images. Sun and Saenko [Sun and Saenko, 2014] were successful in using rendered images as a basis for creating object detectors, while Lim et al. [Lim et al., 2013] showed that relying on rendered data ends up with sub par results, and instead they favor contours and edge maps. We believe that a factor that has been limiting the performance of methods using 3D models, is the actual models used – prior work mostly used simple models that were freely available from Trimble 3D warehouse. While free, these models are far from accurate in their shape and other photometric properties. The images rendered using these models do not look realistic and it is not surprising that classifiers trained on such images have a hard time generalizing to natural images. We attribute part of the success we had in [Movshovitz-Attias et al., 2014] to the detailed models we used, which were bought from `doschdesign.com`.

Another reason that can limit the performance of systems trained from renders is the number of different models used to generate the renders. While many renders can be created for each 3D model, the variability in shape will be limited if the number of models is small. We have been given access to the *TurboSquid* large database of high quality 3D CAD models. TurboSquid is an online marketplace where artists publish their created 3D models. TurboSquid models are used by computer game developers, news agencies, advertisers and many more. Figure 1.1 shows a number of photo-realistic renders obtained from objects from the Turbosquid database (top row). The models found in this database are much more accurate than those used by most prior work (bottom row). We extend our work on scalable generation of labeled training data by leveraging both the Dosch Design and TurboSquid databases, again focusing mainly on the car class. The TurboSquid database contains a diverse set of objects - cars, people, bookcases, etc. There are more than 300,000 models in the database, and for cars alone there are over 16,000 models.

We go one step further and also render realistic background scenes for our objects. By creating renders in which objects are placed in fully modeled environments, we can fully leverage the strength of our approach. Figure 1.2 shows one of our fully rendered scenes. It is extremely realistic. Light rays in the render are traced through interactions with multiple objects, shadows and reflections are fully modeled as well. Creating large numbers of rendered images such as the one shown in Figure 1.2 is computationally expensive and, without access to a large compute cluster, takes too long to be feasible. Instead, we show that a smaller number of these images can be used for validating trained systems, as a proxy for real images. As computing power continues to become cheaper, we foresee a future in which fully rendered scenes can replace natural images as training data.

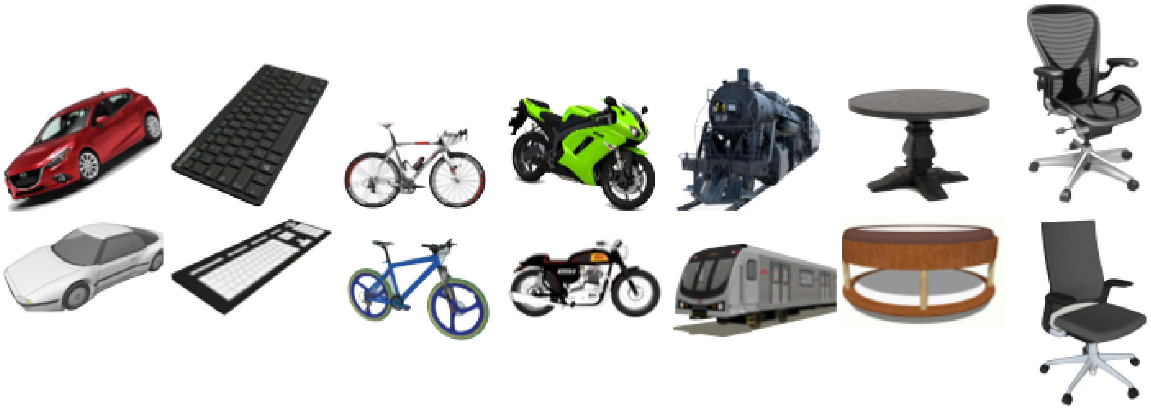


Figure 1.1: Top row: A number of example renders from the large database of 3D CAD models available on the online database turbosquid.com. Bottom Row: Renders from models of similar objects found in the free Trimble 3D Warehouse. The models found in turbosquid enable much higher quality rendering. By using a high end rendering software and accurate models one can create a large amount of photo-realistic renders that can be used to trained object detectors.

1.2.3 Ontological Supervision for Fine Grained Classification of Street View Storefronts

Modern search engines receive large numbers of business related, local aware queries, such as *"Mexican Restaurants in Pittsburgh"*, *"Laundromats around me open now"*, etc. These queries are best answered using accurate, up-to-date, business listings, that contain representations of business categories. Creating such listings is a challenging, and never ending, task as businesses often change hands or close down. For businesses with street side locations one can leverage the abundance of street level imagery, such as Google Street View, to automate the process. However, while data is abundant, labeled data is not; the limiting factor is creation of large scale labeled training data.

In this work, we utilize an ontology of geographical concepts to automatically propagate business category information and create a large, multi label, training data for fine grained storefront classification. We match street level imagery to known business information using both location and textual data extracted from images. We fuse information from the ontology to propagate category information such that each image is paired with labels with different levels of granularity. Our learner, which is based on the GoogLeNet/inception deep convolutional network architecture and classifies 208 categories, achieves human level accuracy.

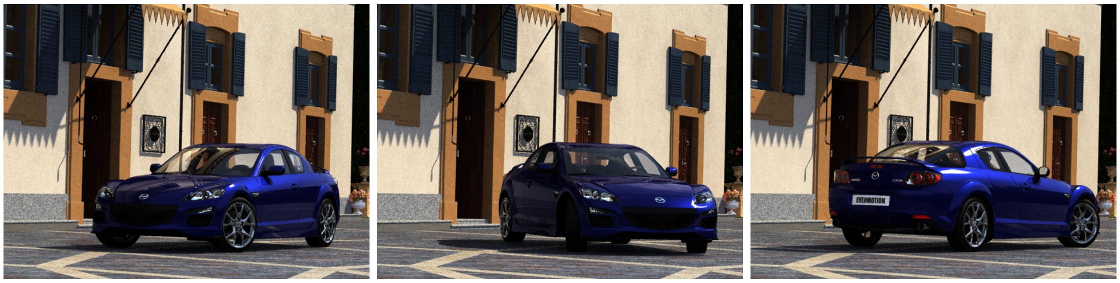


Figure 1.2: A fully rendered scene. To our knowledge, we are the first to utilize this level of realism in synthetic data generation.

1.3 Broad Impact

There are a number of ways in which this work will impact the broad computer vision community: (1) The output of our classification method can be used as the first step in a fully automated solution for object-model matching, which is a fundamental operation in many computer vision, graphics, and robotics applications. In graphics, it can be used for photo-manipulation – inserting new objects into images, and manipulating current ones [Kholgade et al., 2014]. In computer vision, it will aide 3D reasoning of scenes and augmented reality, while in robotics it can be used for robot manipulation. (2) By employing rendered images of 3D models, and automatically assigning labels, a limiting task in computer vision, that of collecting large labeled datasets, is effectively eliminated. The use of 3D CAD model databases for computer vision will enable tasks such as image classification, object detection, pedestrian tracking, and human pose estimation.

Moreover, we feel that insights gained from the work done in this thesis go beyond the use of 3D CAD models, and are applicable to the broad computer vision community. The main takeaway is that for constructing training sets, a computer vision researcher needs to be a true Renaissance Woman, employing skills not only of a machine learning hacker, but also those of natural language analysis, human computer interaction, library studies, behavioral economics, and even philosophy. There is a lot of high quality information that has already been curated, or created, by people using significant amounts of labor. By visually-grounding our tasks to the physical world we are able to leverage these knowledge sources.

1.4 Thesis Outline

The organization of this thesis is as follows.

Chapter 2: This chapter outlines previous work that focused on dataset building. It details some of the difficulties of using human annotators, and earlier approaches that utilized synthetic data.

Chapter 3: Here we describe our work on 3D object pose estimation [Movshovitz-Attias et al., 2014]. Our use of high quality 3D CAD models was a key to the success of the method, and motivated us to scale up our utilization of rendered images.

Chapter 4: We scale up our use of rendered data, and apply modern estimation methods, namely deep convolutional networks, to the problem of viewpoint estimation. Here we focus on generating large scale datasets and examine the effects that various aspects of the rendering have on classifier performance.

Chapter 5: This chapter focuses on our work in fine grained classification of business storefronts from Street View imagery [Movshovitz-Attias et al., 2015]. By automatically matching images to known businesses, and leveraging an ontology of business categories we created a set of over 1 million training images, with multiple labels per instance.

Chapter 6: Summarizes the key insights of this thesis and concludes with a clear take home message: *Focus on the labels*.

Chapter 7: Provides an outline of, what we believe, are important next steps in the space of semi-supervised label generation.

Chapter 2

Background

The price of computational power and storage has decreased dramatically over the last decade. This decrease ushered in a new era in computer vision, one that makes heavy use of highly distributed inference methods [Dean et al., 2012, Szegedy et al., 2014] and massive amounts of labeled data [Russakovsky et al., 2014, Goodfellow et al., 2013a]. This shift was accompanied by a need for efficient ways to quickly and accurately label these large scale datasets. Older and smaller datasets were normally collected and labeled by researchers themselves. This insured high quality labels but was not scalable to the large datasets needed for modern inference methods. As computer vision entered the age of "Big Data" researchers began searching for better ways to annotate large datasets.

2.1 Annotating Large Scale Data

Online labor markets such as Amazon's Mechanical Turk, and CrowdFlower have been used in the computer vision community to crowdsource simple tasks such as image level labeling, or bounding box annotation [Russakovsky et al., 2014, Law et al., 2011, Von Ahn and Dabbish, 2004, Von Ahn et al., 2006]. By utilizing game mechanics, previous work has used labor markets to glean some understanding to how people recognize concepts, and extract discriminative features from images [Deng et al., 2013]. However, labor markets often lack the expert knowledge, making some classes of tasks impossible to complete. Experts are rare in a population and if they are not properly identified, their answers will be ignored when not consistent with other workers – exactly on the instances where their expertise is most crucial. In [Heimerl et al., 2012] the authors addressed the problem of finding a community of experts by adding a physical presence to the task. They suggested the use of physical kiosks to elicit work from specific populations. They placed vending machines in University Computer Science building that provided snacks in return for grading Computer Science questions. In [Movshovitz-Attias et al., 2013] we have addressed the task of predicting which users will turn out to be experts in a social question-answering website for technical domains. We were able to accurately predicted experts within a month of use on the site, but that required access to years of interaction data – the task of finding experts remains difficult.

Many crowdsourced tasks are not only difficult to solve, but are also challenging to verify. One common approach is to rely on workers not only for solving tasks, but also for verifying solutions of other crowd members. This increases both the cost of obtaining the annotations and the time it takes to collect them.

The issues detailed above make for a compelling argument for automating the collection and labeling of datasets. The large increase in availability of 3D CAD models, and the drop in the costs of obtaining them present an appealing avenue of exploration: Rendered images can be tailored for many computer vision applications. Sun and Saenko [Sun and Saenko, 2014] used rendered images as a basis for creating object detectors, followed by an adaption approach based on decorrelated features. Stark et al. [Stark et al., 2010] used 3D models of cars as a source for labeled data. They learned spatial part layouts which were used for detection. However, their approach requires manual labeling of semantic part locations and it is not clear how easy this can scale up to the large number of objects now addresses in most systems. In our previous work we have also explored the car domain, and used sets of highly detailed renders for training ensembles of exemplar detectors [Movshovitz-Attias et al., 2014]. Our approach required no manual labeling but the joint discriminative optimization of the ensembles has a large computational footprint and will be hard to scale as well.

The resurgence of deep learning methods which are efficient in their use of computation, and can therefore process large sets of training data provided an exciting new avenue for utilizing large sets of rendered images. To facilitate work in this area, Xian et al. Everingham et al. [2011] created the PASCAL3D+ dataset by adding viewpoint annotations to images of the rigid classes of PASCAL [Everingham et al., 2011]. The annotation process is complex – first, annotators are asked to select the most similar 3D model to the object in the image out of a set of possible models. Then they mark the 2D location of a set of 12 marker points. If a marker is occluded, the annotators mark it as not visible. A video of the annotation process is available on their website [1]. This dataset provides an important service to the community, but the slow and difficult annotation process limits the number of images that can be labeled. The dataset contains about 36,000 images across 12 cate-

[1] http://cvgl.stanford.edu/projects/pascal3d.html

gories. For a challenging task such as viewpoint estimation this might not be enough data. Methods trained on it might over-fit.

In this thesis we show that detailed renders from a large set of high quality 3D CAD models can be a key part of scaling up collection of labeled data sets. This approach was unfeasible just 10 years ago as the computational costs were too high, but a graphics processing unit (GPU) in a high end desktop today has three orders of magnitude more computation power than the entire server farm used by Pixar for their 1995 movie Toy Story [Fatahalian, 2011, Henne et al., 1996]. The time is ripe for re-examining synthetic image generation as a tool for training computer vision models.

2.2 Viewpoint Estimation

The majority of this thesis will focus on utilizing rendered images for object fine pose estimation. In Chapter 3 we describe a system based on an ensemble of Correlation Filter detectors trained on rendered images. In Chapter 4 we scale up the use of rendered images an order of magnitude by using a large scale 3D model database from TurboSquid.

Contemporary approaches to pose estimation can be categorized into approaches that use local part correspondences, approaches that use a codebook of view specific detectors, and ones that are based on deep end-to-end learning. The correspondence based approaches use various forms of local correspondences from points [David et al., 2004], patches [Deng et al., 2005, Glasner et al., 2012], and parts [Campbell and Flynn, 2001, Schels et al., 2011, Liebelt and Schmid, 2010, Su et al., 2009, Savarese and Fei-Fei, 2007].

Recently, structure from motion was applied by Glasner et al. [2012] on a set of car images to build a 3D object representation, and a discriminative refinement process, comprised of eight viewpoint-aware Support Vector Machines (SVMs), was used to produce the final predictions. Stark et al. [2010] used 3D models to learn viewpoint specific car detectors with a parts based representation using rendered non-photo realistic 2D projections of the 3D car models. In a similar vein, Stark et al. [2012] trained a modified Deformable Parts Model [Felzenszwalb et al., 2010] detector using car images retrieved from Google Image Search, classifying cars into one of a discrete set of eight views. When noisy point correspondences are available, a perspective-N-point method such as SoftPosit [David et al., 2004] or EPnP [Lepetit et al., 2009] can be used to estimate 3D pose precisely from such local correspondences. While these methods are highly accurate, they are susceptible to failure when these correspondences are compromised, e.g., due to resolution or occlusion.

In contrast to the correspondence-based methods, detector-based approaches implicitly infer object view-point via view-specific detectors. Murase and Nayar [1995] pioneered the use of reduced representation of view-specific detectors based on object appearance. In [Ozuysal et al., 2009], SIFT histograms were used within a naive Bayes formulation, and Liebelt et al. [2008] used codebooks of SURF features for matching. In [Blanz et al., 1996], an image sequence of known view-point angles was used for training, and given a test image, distances from each training image were computed and an SVM classifier applied to decide the closest view point that the test image belongs to. In [Yörük and Vidal, 2013], a 3D model was constructed using 2D blueprints, and pose was recovered by optimizing a matching score of quantized-HOGs from 2D images and the 3D model. Zhang et al. [2013] noted that multi-view images of 3D objects lie on intrinsically low-dimensional manifolds. They used that intuition to decompose a view matrix $\mathbf{C}$ into $\mathbf{C} = \mathbf{USV}^T$ where $\mathbf{US}$ is a viewpoint basis, and $\mathbf{V}$ is an instance/category basis.

Several approaches to 3D pose estimation have extended the deformable parts model framework to 3D. These approaches [Hejrati and Ramanan, 2012, Pepik et al., 2012, Fidler et al., 2012, Zia et al., 2013] augment real 2D images either with synthetically generated renders of 2D projections of 3D object models or introduce additional 3D meta-data to the 2D images. However, the main focus of these methods is precise object localization in 3D, i.e., to predict 3D bounding boxes for objects and estimate object pose from these bounding boxes. Thus, these methods usually only estimate coarse object viewpoints, whereas predicting fine-grained viewpoint is the main focus of the work presented in this thesis.

Dimensionality reduction methods for learning a reduced set of a filter bank has also been extensively studied in the computer vision community. Most recently, Song et al. [2012] introduced a sparse coding based approach to directly learn a basis set of detectors that are shared across multiple object classes. However, in their approach the basis set was learned to best reconstruct the full filter set. In contrast our approach in Chapter 3 learns a filter set basis optimizing for discriminative ability. The method proposed by Song et al. [2013] is the closest to our work where they learn a discriminative set of parts filters for multi-class object detection.

In Chapter 3 we perform 3D pose-estimation via a set of exemplar classifiers, one for each pose, while addressing the computational and statistical efficiency in using these exemplar classifiers via dimensionality reduction. Unlike previous approaches, we directly learn a ensemble of detectors by optimizing for the *discriminability* of the reduced set, rather than its ability to reconstruct the complete detector ensemble. This encourages the distinctions that allow discrimination of precise pose differences to be preserved.

Following the success of deep learning in image classification [Krizhevsky et al., 2012], it was only natural for researchers to modify and utilize them for other tasks. Tulsiani and Malik [2015] use CNNs to first predict coarse viewpoint and then estimate the location of marker points for fine-grained pose estimation. They train their models on the PASCAL3D+ dataset, and show significant improvement over deformable parts based models. In the context of face verification, Zhu et al. [2014] proposed a deep network based MultiView Perceptron that decouples face identity and viewpoint. Their approach is inspired by research into the brains of macaque monkeys which shows that they have a face processing network, with neurons in some area being view specific while others target identity across views [Freiwald and Tsao, 2010]. The deep network the constructed separates viewpoint and identity features, and can generate images of input faces from novel orientations.

Following our success in utilizing rendered synthetic images as training data for estimating viewpoint we conduct further analysis of their use in Chapter 4. We examine the effect of various parameters of the rendering process on performance, and conclude that rendered images can be useful for training as well as validation data. While writing this thesis we came across an Arxiv submission by Su et al. [2015]. Their approach is most similar to ours. They employ a large set of 3D CAD models to create synthetic images which they use to train a deep network for detecting objects and estimating their pose. They present a loss function which encourages correlations between adjacent views, and is similar to the one we propose in Chapter 4. While we focus on realism of the rendering process and employ detailed models, they focus on variability in their model set, by performing she augmentation to models from the ShapeNet dataset of models [2].

Synthetic rendered images can be used not only for discriminative tasks such as viewpoint estimation, but also in a generative framework. Dosovitskiy et al. [2014] train a convolutional network to generate images of chairs from high level descriptions such as class, orientation, color, etc. Their system outputs RGB images, including a segmentation mask. A framework that combines a generative network with a discriminative one, pitting one against the other, can possibly train the generative network to create better renders, and the discriminative one to become better at predicting them. This approach as been experimented with on real images quite successful in [Goodfellow et al., 2014a].

2.3 Fine Grained Classification of Business Storefronts

Automation of labels is not limited to rendered images. In Chapter 5 we describe a system designed for classifying business store fronts in Street View imagery. Our key insight was

[2] http://shapenet.cs.stanford.edu/

to match, during training time, business images to a known ontology of businesses. We extracted text from a training image, and matched it to all the saved data about businesses in the area. If a match was found, we would use the ontology to provide multiple relevant labels for the image, thus automating the labeling process. The general literature on object classification is vast. Object category classification and detection [Felzenszwalb et al., 2010] has been driven by the Pascal VOC object detection benchmark [Everingham et al., 2011] and more recently the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Russakovsky et al., 2014]. Here, we focus on reviewing related work on analysis of street view data, fine-grained classification and the use of deep convolutional networks.

Since its launch in 2007, Google Street View [Vincent, 2007, Anguelov et al., 2010] has been used by the computer vision community as both a test bed for algorithms [Lee et al., 2013, Xiao and Quan, 2009] and a source from which data is extracted and analyzed [Goodfellow et al., 2013a, Zamir and Shah, 2010, Micusik and Kosecka, 2009, Flores and Belongie, 2010, Cornelis et al., 2008]. Early work on leveraging street level imagery focused on 3D reconstruction and city modeling. Cornelis et al. [Cornelis et al., 2008] focused on supplying textured 3D city models at ground level for car navigation system visualizations. Micusik et al. [Micusik and Kosecka, 2009] used image segmentation cues and piecewise planar structures to build a robust 3D modeling system.

Later work has focused on extracting knowledge from Street View and leveraging it for particular tasks. In [Zamir and Shah, 2010] the authors presented a system in which SIFT descriptors from $100,000$ Street View images were used as reference data to be queried upon for image localization. Xiao et al. [Xiao and Quan, 2009] proposed a multi view semantic segmentation algorithm that classified image pixels into high level categories such as ground, building, person, etc. Lee et al. [Lee et al., 2013] described a weakly supervised approach that mined midlevel visual elements, and their connections in geographic data sets. Their approach discovered elements that vary smoothly over location, and evaluated their method using 4,455 Street View images along the eastern coast of the United States. Their classifiers were able to predict geographical location with a resolution of about 70 miles.

Most similar to our work, is that of Goodfellow et al. [Goodfellow et al., 2013a]. Both utilize Street View as a map making source, and data mine information about real world objects. They focused on understanding street numbers, while we are concerned with local businesses. They described a method for street number transcription in Street View data. Their approach unified the localization, segmentation, and recognition steps by using a Deep Convolutional Network that operated directly on image pixels. The key idea behind their approach was to train a probabilistic model $P(S|X)$, where S is a digit sequence, and X an image patch, by maximizing $\log P(S|X)$ on a large training set. Their method, which was evaluated on tens of millions of annotated street number images from Street View, achieved above 90% accuracy and was comparable to human operators.

Recently there has been renewed interest in Fine Grained classification [Yao et al., 2012, 2011, Hillel and Weinshall, 2007]. Yao et al. [Yao et al., 2011] modeled images by densely sampling rectangular image patches, and the interactions between pairs of patches, such as the intersection of the feature vectors of two image patches. In [Yao et al., 2012] the authors proposed a codebook-free representation which samples a large number of random patches from training images. They described an image by its response maps to matching the template patches. Branson et al. [Branson et al., 2010] and Wah et al. [Wah et al., 2011] proposed hybrid human-computer systems, which they described as a visual version of the *20-question game*. At each stage of the game, the algorithm chooses a question based on the content of the image, and previous user responses.

Convolutional Networks [Fukushima, 1980, LeCun et al., 1998] are neural networks that contain sets of nodes with tied parameters. Increases in size of available training data and availability of computational power, combined with algorithmic advances such as piecewise linear units [Jarrett et al., 2009, Goodfellow et al., 2013b] and dropout training [Hinton et al., 2012] have resulted in major improvements in many computer vision tasks. Krizhevsky et al. [Krizhevsky et al., 2012] showed a large improvement over state of the art in object recognition. This was later improved upon by Zeiler and Fergus [Zeiler and Fergus, 2013], and Szegedy et al. [Szegedy et al., 2014].

On immense datasets, such as those available today for many tasks, overfitting is not a concern; increasing the size of the network provides gains in testing accuracy. Optimal use of computing resources becomes a limiting factor. To this end Dean et al. developed DistBelif [Dean et al., 2012], a distributed, scalable implementation of Deep Neural Networks. We base our system on this infrastructure.

Chapter 3

Discriminatively Reduced Ensembles of Correlation Filters for Viewpoint Estimation

3.1 Introduction

Accurate estimation of the pose of a 3D model in an image is a fundamental operation in many computer vision and graphics applications, such as 3D scene understanding [Satkin et al., 2012], inserting new objects into images [Karsch et al., 2012], and manipulating current ones [Chen et al., 2013]. One class of approaches to pose estimation is correspondence- based [Stark et al., 2010, 2012, Li et al., 2010]: individual parts of the object are detected, and a pose estimation algorithm (e.g., perspective-N-point) can be used to find the pose of the 3D object in the image. When the parts are visible, these methods produce accurate continuous estimates of pose. However, if the size of the object in the image is small or if the individual parts are not detectable (e.g., due to occlusion, specularities, or other imaging artifacts), the performance of such methods degrades precipitously. In contrast to correspondence-based approaches, pose-by- detection methods use a set of view-specific detectors to classify the correct pose; these methods have appeared in various forms such as filter banks, visual sub-categories, and exemplar classifier ensembles [Ozuysal et al., 2009, Liebelt et al., 2008, Yörük and Vidal, 2013]. While such approaches have been shown to be robust to many of the short-comings of correspondence-based methods, their primary limitation is that they provide discrete estimates of pose and as finer estimates of pose are required, larger and larger sets of detectors are needed.

To maintain scalability, dimensionality reduction has been explored in prior work [Murase and Nayar, 1995, Elgammal and Lee, 2013, Zhang et al., 2013]. Reduced representations are attractive because of their statistical and computational efficiency. Most approaches reduce the set of classifiers via the classic notion of minimizing the reconstruction error of the original filter set. Such a reduction does not directly guarantee optimal preservation of *detection* performance. This is particularly problematic in the case of viewpoint discrimination, as filters of proximal pose angles are similar. Reduction designed to minimize reconstruction error often results in a loss of view-point precision as the distinctive differences in proximal detectors are averaged out by the reduction.

Correlation filters [Kumar et al., 2005] are designed to explicitly suppress side lobes (false classifier response caused by small translational shifts in the bounding box). As small translational shifts confound small rotational shifts, this property makes correlation filters ideally suited for viewpoint discrimination. In this chapter, we present a pose-by-detection approach that uses an ensemble of correlation filters for precise viewpoint discrimination, by using a 3D CAD model of the vehicle to generate renders from viewpoints at the desired precision. A key contribution of this project is a training framework that generates a discriminatively reduced ensemble of exemplar correlation filters [Boddeti et al., 2013] by explicitly optimizing the detection objective. As the ensemble is estimated jointly, this approach intrinsically calibrates the ensemble of exemplar classifiers during construction, precluding the need for an after-the-fact calibration of the ensemble. The result is a scalable approach for pose-by-detection at the desired level of pose precision.

While our method can be applied to any object, we focus on 3D pose estimation of vehicles since cheap, high quality, 3D CAD models are readily available. We demonstrate results that outperform the state-of-the-art on the Weizmann Car View Point (WCVP) dataset [Glasner et al., 2012], the EPFL car multi-view car dataset [Ozuysal et al., 2009], and the VOC2007 car viewpoint dataset [Arie-Nachimson and Basri, 2009]. We also report results on a new data-set based on the CMUcar dataset [Boddeti et al., 2013]) for precise viewpoint estimation and detection of cars. These results demonstrate that pose-by- detection based on ensemble of exemplar correlation filters can achieve and exceed the level of precision of correspondence based methods in real datasets; and that discriminative reduction of an ensemble of exemplar classifiers allows scalable performance at higher precision levels.

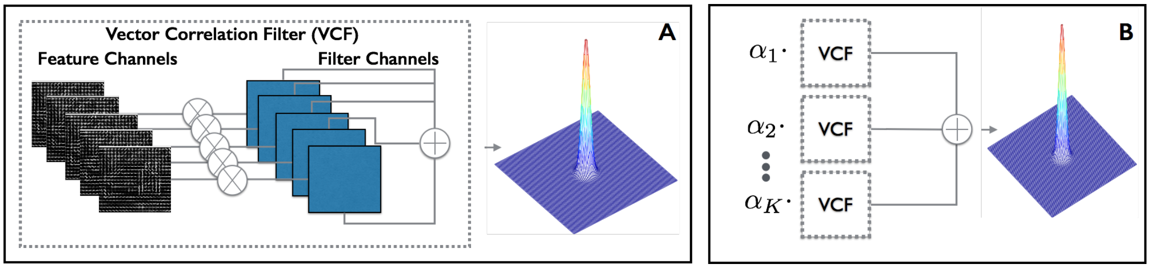


Figure 3.1: Overview of learning the Exemplar Correlation Filter Basis (EECF). (a) The Vector Correlation Filter (VCF) design aggregates the responses of all feature channels to produce a correlation output which is constrained to have a sharp peak only at the target location. We use $\otimes$ for convolution and $\oplus$ for element-wise sum. (b) Our method (EECF) jointly learns a set of Vector Correlation Filters such that their linear combination produces the sharp peak.

3.2 Method

Our approach learns a discriminatively reduced ensemble of exemplar classifiers that spans the vehicle's appearance as it changes with respect to viewpoint. Given a 3D model and a desired precision of d°, we densely sample $V = \lceil 360/d \rceil$ viewpoints of the object (along one axis of rotation) and create renders using an empty background in a graphics rendering package (Maya). Exemplar classifiers [Malisiewicz et al., 2011] are trained using a single positive instance and a large set of negative instances. This procedure creates a classifier that is tuned to the characteristics of the single positive instance. We use the vector correlation filter formulation[1] introduced in [Boddeti et al., 2013] with Histogram of Oriented Gradients (HOG) features [Dalal and Triggs, 2005].

3.2.1 Ensemble of Exemplar Classifiers for Pose-by-Detection

Exemplar classifiers are suited to the task of pose-by-detection. For each one of the V viewpoint renders we train an Exemplar Correlation Filter (ECF) using the rendered image as the single positive, and $N-1$ image patches selected randomly from a background set of images that do not contain the object instance. Each ECF is trained to detect the object from a specific viewpoint.

Let $\{\mathbf{x}_i\}_{i=1}^{N}$ be a set of Histogram of Oriented Gradients (HOG) representations of the training examples, consisting of one positive exemplar rendering of the v-th view and $N-1$ negative bounding boxes. Also, define $\left\{\mathbf{g}_v^1, \cdots, \mathbf{g}_v^C\right\}$ as the ECF for a viewpoint v, where C is the number of channels of the HOG feature representation (commonly 32). The response of an image $\mathbf{x}_i$ to the filter is defined as

$$\sum_{c=1}^{C} \mathbf{x}_i^c \otimes \mathbf{g}_v^c = \text{Correlation Output}, \tag{3.1}$$

where $\otimes$ denotes the 2D convolution operator. The ECF design is posed as:

$$\min_{\mathbf{g}_v^1, \cdots, \mathbf{g}_v^C} \sum_{i=1}^{N} \left\| \sum_{c=1}^{C} \mathbf{x}_i^c \otimes \mathbf{g}_v^c - \mathbf{r}_i \right\|_2^2 + \lambda \sum_{c=1}^{C} \left\| \mathbf{g}_v^c \right\|_2^2, \tag{3.2}$$

where $\mathbf{r}_i$ is the matrix holding the desired correlation output of the i-th training image, and λ moderates the degree of regularization. The desired correlation output $\mathbf{r}_i$ is set to a positively scaled Gaussian for the positive exemplar and to a negatively scaled Gaussian for the negative patches. This choice of the desired output correlation shape also implicitly calibrates the different exemplar classifiers. The minimization problem can be equivalently posed in the frequency domain to derive a closed form expression, which in turn lends itself to an efficient solution [Boddeti et al., 2013]. It should be noted that, as a complete set, each view $v \in V$ is trained independently, and that increase in the desired precision d increases the size of the ensemble (linearly for one axis of rotation, quadratically for two, and cubically for all three). Figure 3.1 (a) shows the training configuration for one exemplar correlation filter. For visualization clarity we do not show negative images.

[1] Correlation Filters [Kumar et al., 2005] are a type of classifier that explicitly controls the shape of the entire cross-correlation output between the image and the filter. They are designed to give a sharp peak at the location of the object in the image and no such peak elsewhere. In contrast to SVMs, which treat the HOG feature channels as independent of each other, the vector CF design jointly optimizes all the feature channels to produce the desired output via interactions between multiple channels. The Correlation Filter optimization has an analytical solution, which can be solved efficiently, significantly faster than traditional classifiers (such as SVMs).

3.2.2 Discriminative Reduction of Ensembles of Correlation Filters

The procedure described in Section 3.2.1 produces a large set of exemplar classifiers, one per view that needs to be resolved. Let $\mathbf{G} \in \mathbb{R}^{D \times V}$ be the matrix of all V filters arranged as column vectors, where D is the dimensionality of the feature. This set is an exhaustive representation of the object's appearance from many views, but applying all the filters during test time is computationally expensive. It is also highly redundant as many views of the object are similar in appearance. Our reduced Ensemble of Exemplar Correlation Filter (EECF) approach is designed to jointly learn a set of K exemplar correlation filters $\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_K]$ (each with C channels) and a set of V sparse coefficient vectors $\mathbf{A} = [\alpha_1, \ldots, \alpha_V]$ such that a detector $\mathbf{g}_v$ for any viewpoint v of the object is defined by

$$\mathbf{g}_v = \mathbf{F}\alpha_v. \tag{3.3}$$

As before, there are V positive training images, one corresponding to each view that is expected to be resolved. Define B to be a set of randomly selected negative background patches. To learn a reduced EECF, we define the following discriminative objective:

$$\begin{aligned} \arg\min_{\mathbf{F},\mathbf{A}} & \underbrace{\sum_{i:\mathbf{x}_i \in V} \left\| \sum_{k=1}^{K} \alpha_k^i \Big(\sum_{c=1}^{C} \mathbf{f}_k^c \otimes \mathbf{x}_i^c \Big) - \mathbf{r}^{\text{pos}} \right\|_2^2}_{\text{Controls EECF behavior for positive images}} \\ & + \underbrace{\sum_{j:\mathbf{x}_j \in B} \sum_{i:\mathbf{x}_i \in V} \left\| \sum_{k=1}^{K} \alpha_k^i \Big(\sum_{c=1}^{C} \mathbf{f}_k^c \otimes \mathbf{x}_j^c \Big) - \mathbf{r}^{\text{neg}} \right\|_2^2}_{\text{Controls EECF behavior for negative images}} \\ & + \underbrace{\lambda_1 \|\mathbf{F}\|_2^2 + \lambda_2 \|\mathbf{A}\|_1}_{\text{Regularization and sparsity}}, \end{aligned} \tag{3.4}$$

where $\mathbf{x}_i$ and $\mathbf{r}_i$ are as defined for Eq. (3.2) and $\mathbf{f}_k^c$ is the c-th channel of the k-th reduced filter. α^i is the sparse mixing coefficient for the i-th training image, and λ_1, λ_2 control regularization and enforce sparseness. The need for sparsity will be explained presently.

The first part of the equation guides the optimization to find a reduced set of correlation filters F and a matrix A of coefficients such that Eq. (3.3) holds. That is, that a detector for any viewpoint can be estimated by a linear combination of the columns of F, weighted by α_i. The second part of the equation controls the discriminability of the ensemble. The key idea is that, as there is no value of α that can be defined for a negative instance, we enforce a negative response $\mathbf{r}_j$ for each negative instance, with any of the learned α. This optimization can be solved efficiently by posing the problem in the Fourier-domain.

The mental picture one should have in mind when learning the $\mathbf{F}$ matrix, is that shown in Figure 3.1 (b). The full basis of K filters is convolved with the image and the convolution with $\mathbf{f}_k$ are weighted by α_k.

3.3 Optimization

3.3.1 Learning Vector Correlation Filters

We first provide a short derivation of VCF here as it will be helpful for understanding our joint approach in 3.2.2. For an in depth tutorial, we direct the reader to [Boddeti, 2012].

We can restate Eq. (3.2) in the frequency domain using Parseval's theorem [Oppenheim et al., 1983]

$$\arg\min_{\hat{\mathbf{f}}^1,\ldots\hat{\mathbf{f}}^c} \sum_{i=1}^{N} \left\| \sum_{c=1}^{C} \hat{\mathbf{X}}_i^c \hat{\mathbf{f}}^c - \hat{\mathbf{r}}_i \right\|_2^2 + \lambda \sum_{c=1}^{C} \hat{\mathbf{f}}^{c\dagger} \hat{\mathbf{f}}^c. \tag{3.5}$$

Where we use the $\hat{}$ notation for the Fourier Transform of a vector, for example $\hat{\mathbf{r}}_i$ is the Fourier Transform of $\mathbf{r}_i$. $\hat{\mathbf{X}}_i^c$ is a diagonal matrix with the values of $\hat{\mathbf{x}}_i^c$ on its main diagonal. $\hat{\mathbf{f}}^{c\dagger}$ is the conjugate transpose of $\hat{\mathbf{f}}^c$. Let $\hat{\mathbf{f}}$ and $\hat{\mathbf{y}}$ be

$$\hat{\mathbf{f}} = \begin{bmatrix} \hat{\mathbf{f}}^1 \\ \vdots \\ \hat{\mathbf{f}}^C \end{bmatrix} \qquad \hat{\mathbf{y}} = \sum_{i=1}^{N} \hat{\mathbf{y}}_i = \sum_{i=1}^{N} \begin{bmatrix} \hat{\mathbf{X}}_i^1 \hat{\mathbf{r}}_i \\ \vdots \\ \hat{\mathbf{X}}_i^C \hat{\mathbf{r}}_i \end{bmatrix}, \tag{3.6}$$

Then Eq. (3.5) can be written in matrix form as

$$\arg\min_{\hat{\mathbf{f}}} \hat{\mathbf{f}}^\dagger \hat{\mathbf{S}} \hat{\mathbf{f}} - 2\hat{\mathbf{f}}^\dagger \hat{\mathbf{y}}, \tag{3.7}$$

where $\hat{\mathbf{S}} = \hat{\mathbf{D}} + \lambda \mathbf{I}$ and

$$\hat{\mathbf{D}} = \sum_{i=1}^{N} \hat{\mathbf{D}}_i = \sum_{i=1}^{N} \begin{bmatrix} \hat{\mathbf{X}}_i^{1\dagger} \hat{\mathbf{X}}_i^1 & \ldots & \hat{\mathbf{X}}_i^{1\dagger} \hat{\mathbf{X}}_i^C \\ \vdots & \ddots & \vdots \\ \hat{\mathbf{X}}_i^{C\dagger} \hat{\mathbf{X}}_i^1 & \ldots & \hat{\mathbf{X}}_i^{C\dagger} \hat{\mathbf{X}}_i^C \end{bmatrix}. \tag{3.8}$$

The matrix $\hat{\mathbf{S}}$ is block diagonal. $\hat{\mathbf{D}}$ is the interaction energy between different channels of the feature. For a HOG template of size m each block is of size $m \times m$, and the matrix is of size $mC \times mC$.

The desired filter is

$$\hat{\mathbf{f}} = \hat{\mathbf{S}}^{-1}\hat{\mathbf{y}}. \tag{3.9}$$

3.3.2 Learning the Basis Filters

The filter basis can be solved for analytically in the frequency domain by weighted averaging of the inputs. Define

$$\hat{\mathbf{f}} = \left[\hat{\mathbf{f}}_1^1 \dots \hat{\mathbf{f}}_1^C \dots \hat{\mathbf{f}}_K^1 \dots \hat{\mathbf{f}}_K^C\right]^T. \tag{3.10}$$

The solution that minimizes Eq. (3.4) is

$$\hat{\mathbf{f}} = (\hat{\mathbf{D}} + \lambda_1 \mathbf{I})^{-1}\hat{\mathbf{y}}. \tag{3.11}$$

where

$$\hat{\mathbf{D}} = \sum_{i=1}^{N} \begin{bmatrix} \alpha_1^i \alpha_1^i \hat{\mathbf{D}}_i, \dots \alpha_1^i \alpha_K^i \hat{\mathbf{D}}_i \\ \vdots \quad \ddots \quad \vdots \\ \alpha_K^i \alpha_1^i \hat{\mathbf{D}}_i, \dots \alpha_K^i \alpha_K^i \hat{\mathbf{D}}_i \end{bmatrix}, \hat{\mathbf{y}} = \sum_{i=1}^{N} \begin{bmatrix} \alpha_1^i \hat{\mathbf{y}}_i \\ \vdots \\ \alpha_K^i \hat{\mathbf{y}}_i \end{bmatrix} \tag{3.12}$$

and $\hat{\mathbf{D}}_i$ and $\hat{\mathbf{y}}_i$ are as defined in Eq. (3.8) and Eq. (3.6)

The matrix $\hat{\mathbf{S}}$ is now of size $mKC \times mKC$ which can become large — typical values are $m = 450, K = 20, C = 32$. However this matrix is block diagonal and sparse. Each $\hat{\mathbf{X}}_i$ block is of size $m \times m$ but only has m non zero elements. The number of non zero elements in $\hat{\mathbf{S}}$ is $m(CK)^2$. The block structure of the matrix allows for efficient inversion using Schur complement matrix inversion.

We initialize the matrix of coefficients A using sparse coding on the HOG features of the training images. This provides a reasonable starting point for the optimization.

Figure 3.2 shows a number of filters from learned Ensemble of Exemplar Correlation Filter (EECF). Note that the ensemble members capture the shape of the car from different viewpoints. This is interesting as the optimization does not restrict the learned filters to be specific viewpoints.

3.4 Predicting the Viewpoint

The reduced EECF is used to reconstruct the filter responses of the complete ensemble set $\mathbf{G}$. As the learned coefficient matrix $\mathbf{A}$ is sparse this procedure will be much faster

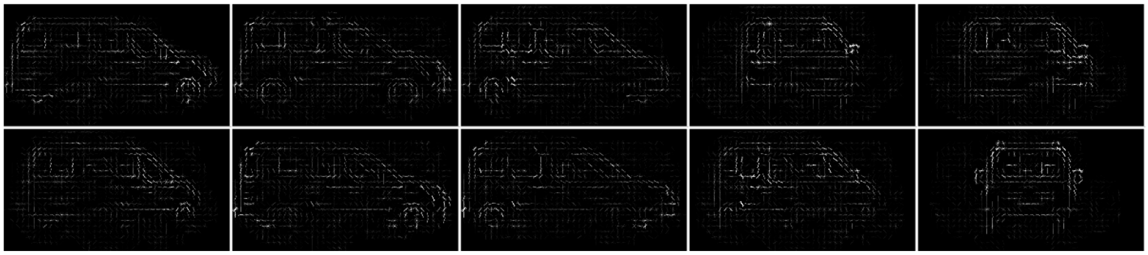


Figure 3.2: Some filters from a learned Ensemble of Exemplar Correlation Filter (EECF) of size 40. The filters are learned in HOG space. We use the standard HOG visualization in which light pixels show orientations with a large positive weight.

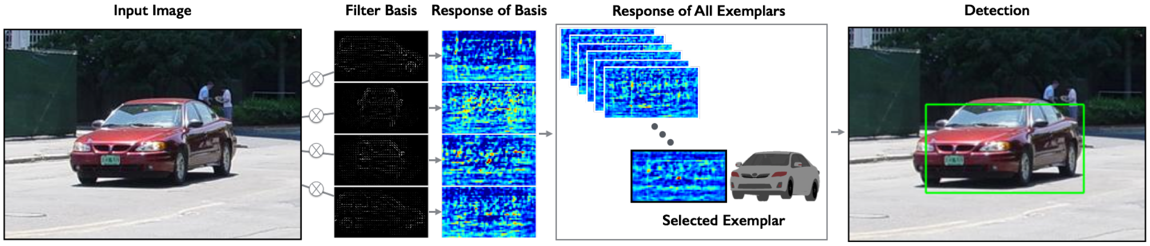


Figure 3.3: Predicting the Viewpoint: The learned Exemplar Correlation Filter Basis is applied to the input image. The response of any viewpoint $v \in V$ is estimated by a linear combination of the responses of the basis filters. Locations with a high response value are predicted as car detections.

than applying all ECFs on the image. The reduced EECF is applied on an input image at varying scales and locations. The response of all views is estimated using the sparse coefficients. Peaks in these responses are used to predict the location and viewpoint of the car in the image.

Let $\mathbf{r}_v^g \in \mathbb{R}^M$ be the response of evaluating ECF $\mathbf{g}_v$ on a test image I of M HOG cells; $\mathbf{r}_v^g$ can be expressed as $\mathbf{r}_v^g = \mathbf{g}_v \otimes I$ where $\mathbf{g}_v$ is the v-th column of $\mathbf{G}$. With Eq. (3.3),

$$\tilde{\mathbf{r}}_v^g = \left(\sum_{k=1}^{K} \alpha_k^v \mathbf{f}_k\right) \otimes I = \sum_{k=1}^{K} \alpha_k^v (\mathbf{f}_k \otimes I) = \sum_{k=1}^{K} \alpha_k^v \mathbf{r}_k^f, \tag{3.13}$$

where $\tilde{\mathbf{r}}_v^g$ is an estimator of $\mathbf{r}_v^g$. That is, the response of the ECF corresponding to the v-th view, can be estimated as a weighted sum of the responses of the K ensemble elements. We can reshape $\mathbf{r}_v^g$ and $\mathbf{r}_k^f$ as vectors, and arrange them as the columns of the matrices $\mathbf{R}^g$

and $\mathbf{R}^f$ respectively. An estimator $\tilde{\mathbf{R}}^g$ for the response of all the exemplar filters on the image is

$$\tilde{\mathbf{R}}^g = \mathbf{R}^f \mathbf{A}. \tag{3.14}$$

Note that as $\mathbf{A}$ is sparse this multiplication is efficient even though $\mathbf{R}^f$ is large.

3.4.1 Context Rescoring

To reduce the effect of false positive detections we pool information from nearby boxes. Each box is associated with a specific angle and we can use that information to make a better decision. For a box b with a detection score $\hat{s}(b)$, we assign the following score

$$s(b) = \frac{\sum_{b_n \in \mathbb{B}} \mathrm{OS}(b, b_n) \cdot \mathcal{K}(b, b_n) \cdot \hat{s}(b_n)}{\sum_{b_n \in \mathbb{B}} \mathrm{OS}(b, b_n) \cdot \mathcal{K}(b, b_n)}, \tag{3.15}$$

where $\mathbb{B}$ is the set of all boxes in the image, $\mathrm{OS}(b, b_n)$ is a function that measures the overlap of two boxes, and $\mathcal{K}(b, b_n)$ is a Gaussian Kernel in angle space (circular normal distribution), centered at the angle of box b. The rescoring function reduces the score of false positives as they are unlikely to overlap with other boxes that predict the same angle.

3.5 Evaluation

We focus our evaluation on viewpoint precision for two main use-case scenarios on four datasets: WCVP, CMU-Car, VOC2007 car viewpoint, and EPFL multi-view cars. The first case (**KNOWN**) is where the image contains a car for which we have a 3D CAD model corresponding to the particular make and model. In the second case (**UNKNOWN**), the image contains a car for which we do not have the exact 3D CAD model. When this is the case, we need to fall back to a generic strategy. We create views for four representative car models: a sedan, a van, a compact car, and a hatchback.

For each car, we create $V = 360$ renders by rotating a synthetic camera around the 3D model in increments of 2 degrees in azimuth and increments of 10 degrees for elevation values (for elevation of 0 and 10 degrees). We train 360 exemplar correlation filters for each one of the views and learn a reduced ensemble with $K = 20$, and $K = 40$ ensemble elements. For a given test image, we apply all the ensemble elements at varying scales and locations and use their output to estimate the cross correlation response of all 360 ECFs. Finally, we apply non-maxima suppression on the predicted bounding boxes. Each bounding box prediction is associated with a specific exemplar and we use that exemplar's

(a) Azimuth Estimation: WCVP

Method	Median Angular Error *KNOWN*	 *UNKNOWN*
EECF, $K = 20$	10.2°	9.4°
EECF, $K = 40$	7.6°	8.4°
ECF full (360)	6.9°	7.5°
Glasner et al. [Glasner et al., 2012]	-	12.25°

(b) Pose Estmation: CMU Car

Method	Median Angular Error Azimuth	 Elevation
EECF, $K = 20$	26.0°	3.8°
EECF, $K = 40$	11.48°	3.6°
ECF Full (360)	3.2°	3.0°
Glasner et al. [Glasner et al., 2012]	-	-

Table 3.1: (a) Median angular error in azimuth estimation for WCVP dataset. When the car model in the image is known, using 40 filters, our ensemble exemplar correlation filter method achieves a median error of 7.6°. When the car model is unknown a basis of 40 filters has a median error of 8.4°. Previous results on this dataset have a median error of 12.25°. (b) Median angular error in azimuth and elevation estimation for CMU Car dataset using unknown models.

angle as our prediction. For the *UNKNOWN* case we follow the protocol described above, but use filters learned from the four representative 3D models.

WCVP. The Weizmann Car Viewpoint (WCVP) dataset [Glasner et al., 2012] contains 1530 images of cars, split into 22 sets. Each set has approximately 70 images and shows a different car model. On this dataset we evaluate both the *KNOWN* and the *UNKNOWN* scenarios. For the *KNOWN* case, we obtained 10 3D CAD models of cars from this dataset from the on-line repositories Dosch Design and Trimble 3D Warehouse. There are 683 images in the data set for these 10 models and we evaluate on those. For the *UNKNOWN* case we evaluate on all the images in WCVP. Table 3.1(a) shows the median angular error for azimuth prediction over the 10 car models tested for the *KNOWN* use case, and the full dataset for the *UNKNOWN* use case. Using a known 3D CAD model and a full set of exemplar correlation filters as described in Section 3.2.1 produces an angular error of

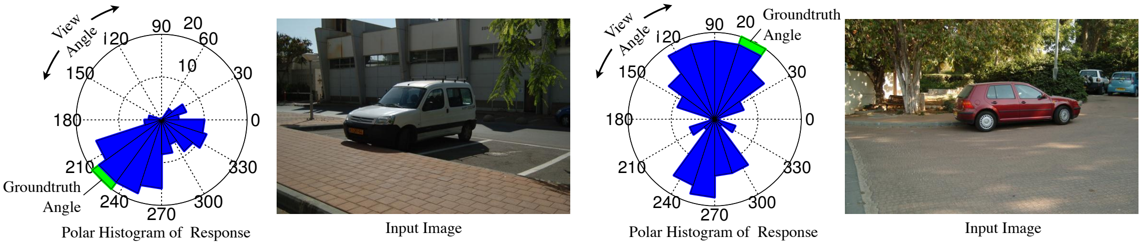


Figure 3.4: Polar histogram of scores. The left example shows a van at an oblique angle, with little ambiguity in the distribution of responses. The right example shows a side view with the distinctive symmetric ambiguity.

$6.9°$, which is a reduction of 40% in error rate from the $12.25°$ reported by Glasner et al. [Glasner et al., 2012]. Using a reduced set of 40 filters the error increases by less than $1°$ to 7.6°. When the model is unknown, a 40 ensemble filter produces an error of $8.4°$. This quantifies the benefits of using a known 3D CAD model, compared to the harder problem of using a model trained on a holdout set as in [Glasner et al., 2012] or in the *UNKNOWN* use case. Figure 3.4 shows a polar histograms of the predicted angles for two examples. The figure shows a distinctive ambiguity in prediction cause by the car's symmetric structure.

CMU-Car. The MIT street scene data set [Bileschi, 2006] was augmented by Boddeti et al. [Boddeti et al., 2013] with landmark annotations for $3,433$ cars. To allow for evaluation of precise viewpoint estimation we further augment this data set by providing camera matrices for $3,240$ cars. To get the camera viewpoint matrices, we manually annotated a 3D CAD car model with the same landmark locations as the images and used the POSIT algorithm to align the model to the images. To ensure a clean ground truth set, we then back projected the 3D points to the 2D plane and only used cars for which the sum of reprojection error over all landmark points was smaller than 8 pixels. The CAD model used was different from those used later in testing.

We use CMU-Car to evaluate the *UNKNOWN* scenario. For each car, we use the ground truth bounding box to crop a large image section around it (the area of the cropped image is 3 times the area of the bounding box and may contain other cars). Figure 3.6 shows examples of detections and viewpoint prediction on this dataset. Table 3.1(b) shows pose estimation results on this dataset. Most images in this dataset were taken from standing height which explains the low elevation error. The median error in azimuth is $11°$ when using an EECF of 40 filters. Figure 3.5 (Right) shows the distribution of angular errors for an ensemble of size 40. The errors made by the algorithm can be split in two; small estimation errors, and 180°ambiguity errors. There are few errors in the $[30°, 165°]$ range.

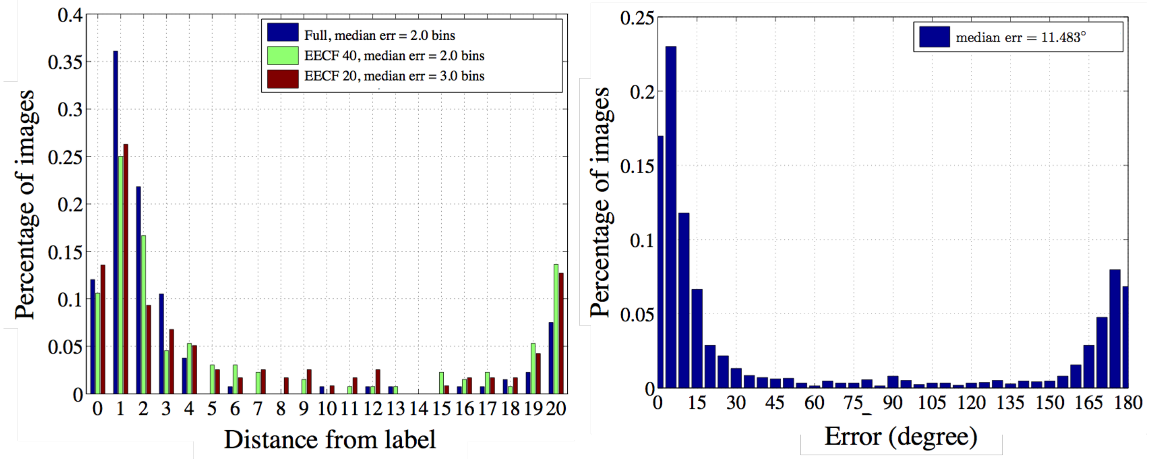
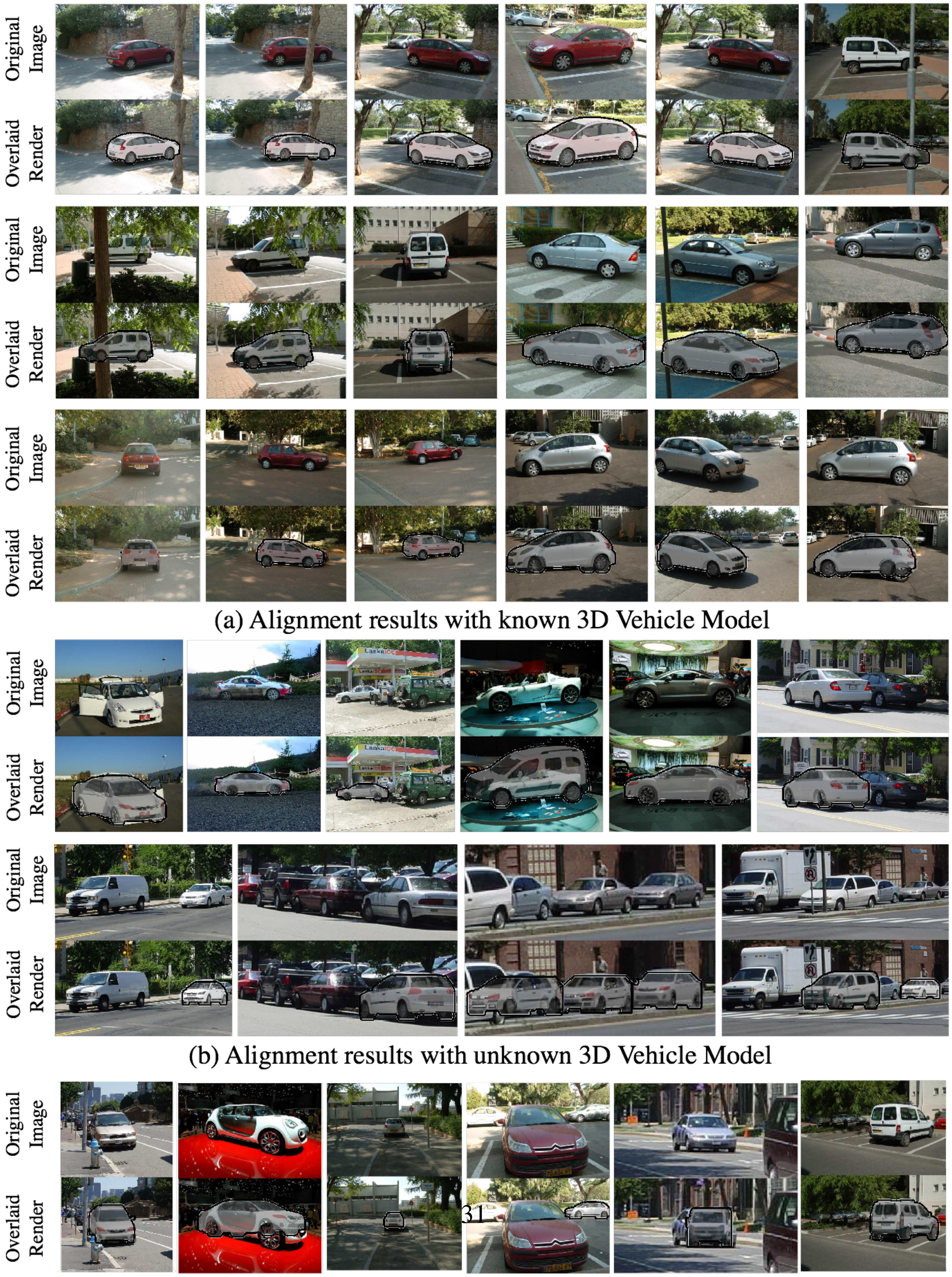


Figure 3.5: (Left) A histogram of view errors for the VOC2007 car viewpoint dataset. Most images were matched near their most similar pose and so there is a peak around a 1 bin error. (Right) Histogram of angular error on the CMU-Car dataset. The Median error is 11.48°. In both cases, the smaller peak is due to the 180° symmetricity.

VOC2007 car viewpoint. In [Arie-Nachimson and Basri, 2009], Arie-Nachimson and Basri provided viewpoint annotations for 200 cars from the PASCAL VOC 2007 test set car category. Each car is labeled with one of 40 viewpoint labels that correspond to reference images. Figure 3.5 (Left) shows a histogram of prediction distance from true labels. The majority of the predictions are within a distance of 2 to the ground truth label. This is an improvement over the results of [Arie-Nachimson and Basri, 2009] which had a median distance of 3 bins.

EPFL Multi-View Cars [Ozuysal et al., 2009]. This dataset contains 2299 images of 20 different car models. Each car is imaged at intervals of 3° for a full turn. We apply the *UNKNOWN* use case on this dataset as outlined above. Using 40 ensemble filters we achieve a median error of 19° compared with 24.83° reported in [Glasner et al., 2012]. we refer the reader to [Movshovitz-Attias et al., 2014] for more detail about all our work.



(a) Alignment results with known 3D Vehicle Model

(b) Alignment results with unknown 3D Vehicle Model

(c) Failure cases

Figure 3.6: Example results from all of the datasets used. Each row shows input images (top) and overlaid pose estimation results (bottom). (a) Results using a *Known* 3D model, (b) results using an *Unknown* 3D model, and (c) failure cases.

3.6 Computational Efficiency

Exemplar-based approaches have been gaining popularity as they can provide state-of-the-art detection for precise detection problems. However, these methods scale poorly; as the desired precision of viewpoint estimation increases, the number of exemplars needed increases as well. We present a pose-by-detection framework that considers both computational and statistical efficiency. Our approach *directly* optimizes discriminative power to efficiently detect the viewpoint of an object. The need for reducing the number of applied filters is especially prominent at the two extremes of scale: on a mobile platform where the available computation power is limited, and on the data-center scale where the number of images to be evaluated is vast. In lieu of reported computation times that depends on hardware specification and implementation, we analyze the computational complexity of our approach. Let $M \times C$ be the size of the HOG representation of an image, and $m \times C$ the size of the learned filters. Furthermore let V be the number of exemplars, and K the size of the learned ensemble of filters. When all exemplars are convolved, the complexity is

$$\mathcal{O}(CVM \log_2 m)$$

With the reduced set the complexity is

$$\mathcal{O}(CKM \log_2 m + \gamma MKV)$$

where γ is the fraction of non-zero elements in $\mathbf{A}$. Thus, the computational savings are $\mathcal{O}\left(K/V + \gamma K/C \log_2 m\right)$.

3.7 Discussion

Our method produced state-of-the-art results on the WCVP dataset and the EPFL dataset, significantly reducing the error from previous results. Additionally, we have introduced a new dataset, CMU-Car, for viewpoint estimation that contains more than 3000 images. On this dataset, we achieve $11.5°$ azimuth error, and $3.6°$ elevation error by using a 40 element EECF basis. A fundamental limitation of pose-by-detection approaches, including the method presented in this paper, is that as the precision is increased beyond a point, it becomes increasingly harder to discriminate between nearby viewpoints, because the underlying features are designed to provide invariance to small spatial variations. This suggests an important direction of future work, tying feature selection into the detector optimization. Our results clearly indicate that rendered data can be effectively used to train computer vision systems. In Chapter 4 we scale up our use of rendered data for viewpoint estimation, and indeed couple feature selection to the optimization criteria using a deep learner with a specialized loss layer.

Chapter 4

Rendered Data Generation for Viewpoint Estimation

4.1 Introduction

The computer vision community has been building datasets for decades. And as long as we have been building them, we have also been fighting their dreaded biases. From the early days of COIL-100 [Nene et al., 1996] and the Corel Stock Photos and 15 Scenes datasets [Oliva and Torralba, 2001] up to and including newer datasets such as PASCAL VOC [Everingham et al., 2011] and Imagenet [Deng et al., 2009] researchers have been experiencing a universal truth: every sample of the world is biased in some way. Our task as computer vision researchers has been to build algorithms that achieve the best performance on these datasets. In effect, we have been "hacking" each new dataset that is proposed - exploring it, identifying its weaknesses, and in some cases flawlessly fitting to it.

In a now famous act of data exploration, Antonio Torralba took the training images of the Caltech101 dataset and averaged them per class to create average representations. Figure 4.1 shows these average images for a number of the classes. Note how objects are centered in all images, in many classes the object is clearly visible in the image due to lack of pose variability in the data. For an in depth analysis of the evolution of datasets (and an enjoyable read) we refer the interested reader to Torralba and Efros [2011]. In short, there are two main ways in which our community has addressed bias – making new datasets, and building bigger datasets. By making new datasets we continuously get new samples of the visual world, and make sure our techniques handle more of its variability. By making

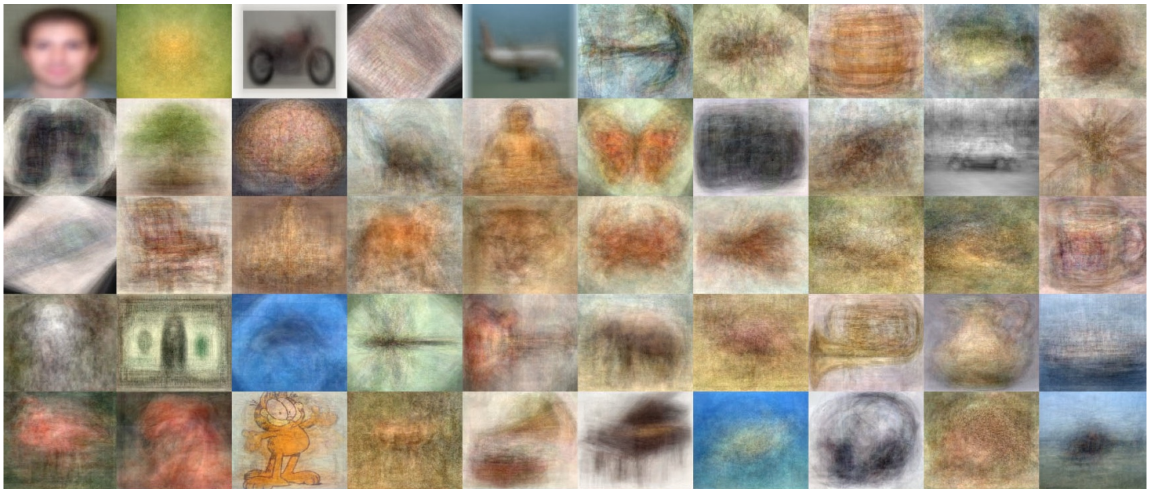
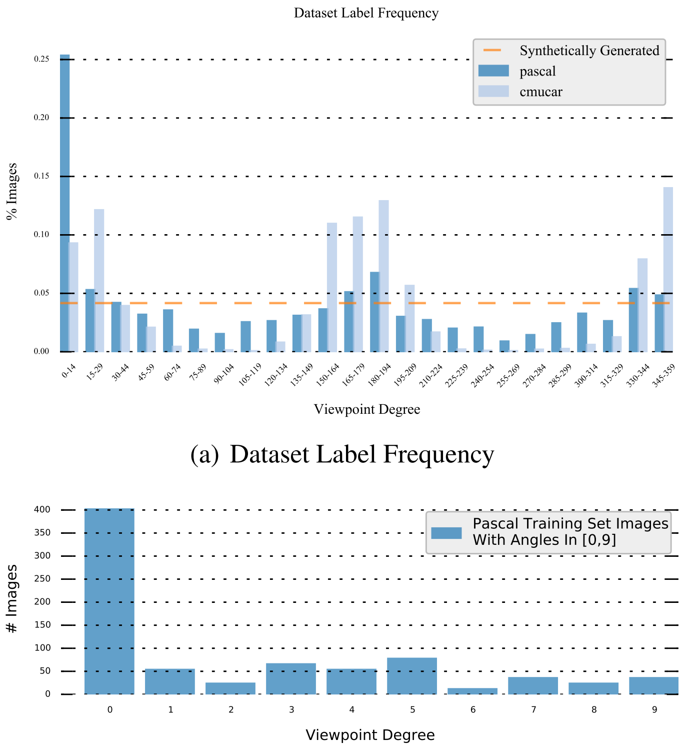


Figure 4.1: All training images for a number of classes from Caltech-101. Images are averaged per class. The bias in the dataset is clearly visible.

our datasets larger we make it harder for machine learning algorithms to over-fit to the dataset's individual idiosyncrasies.

This approach has been remarkably successful. It requires, however, a great amount of effort to generate new datasets and label them with ground truth annotations. Even when great care has been taken to minimize the sampling bias it has a way of creeping in. As an example, for our task of object viewpoint estimation, we observed clear bias in the distribution viewpoint angles when we explored real image datasets. Figure 4.2(a) shows the distribution of azimuth angles for the training sets of the car class of PASCAL VOC, and the CMUCar dataset. There is clear oversampling of some angles, mainly around $0°$ and $180°$ – Figure 4.2(b). However, with rendered data, we can control for this, and create a completely uniform distribution. We can also adequately sample lighting conditions and occlusions by other objects. Figure 4.3 shows various samples created for a single object, an IKEA bed. Note how we can sample the different angles, lighting conditions, and occlusions. We will, of course, have other types of bias, and this suggests an interesting approach – augment real image datasets with rendered data to reduce observable bias. We explore this idea below.

In this chapter, we scale up our use of rendered data as training samples for a 3D viewpoint estimation task with the aim to understand the relationship of various parameters of renders for visual learning. 3D viewpoint estimation is an ideal task for the use of renders, as it requires a high degree of accuracy in labeling. We assert that a factor limiting the success of many computer vision algorithms is the scarcity of labeled data and the preci-



(a) Dataset Label Frequency

(b) Zoomed View: Angles 0-9

Figure 4.2: Objects have *canonical* viewpoints that photographs tend to capture them in. When real images are used as a training dataset these viewpoints are oversampled. (a) Viewpoint distribution of two real image datasets (the car class from PASCAL VOC, and the CMUcar dataset). Note the oversampling of certain views, particularly those close to $0°$ and $180°$. In comparison, an advantage of a dataset created from synthetic images is that it is created to specification. The most natural distribution of training instances might be uniform, as is shown by the dashed line. (b) Extreme Oversampling in PASCAL training set for the 0th angle.

sion of the provided labels. For viewpoint estimation in particular the space of possible angles is immense, and collecting enough samples of every angle is hard. Furthermore, accurately labeling each image with the ground truth angle proves difficult for human annotators. As a result, most current 3D viewpoint datasets resort to one of two methods for labeling data: (1) Provide coarse viewpoint information in the form of viewpoint classes (usually 8 to 16). (2) Use a two step process of labeling. First, ask annotators to locate about a dozen keypoints on the object (e.g., front-left most point on a car's bumper), then

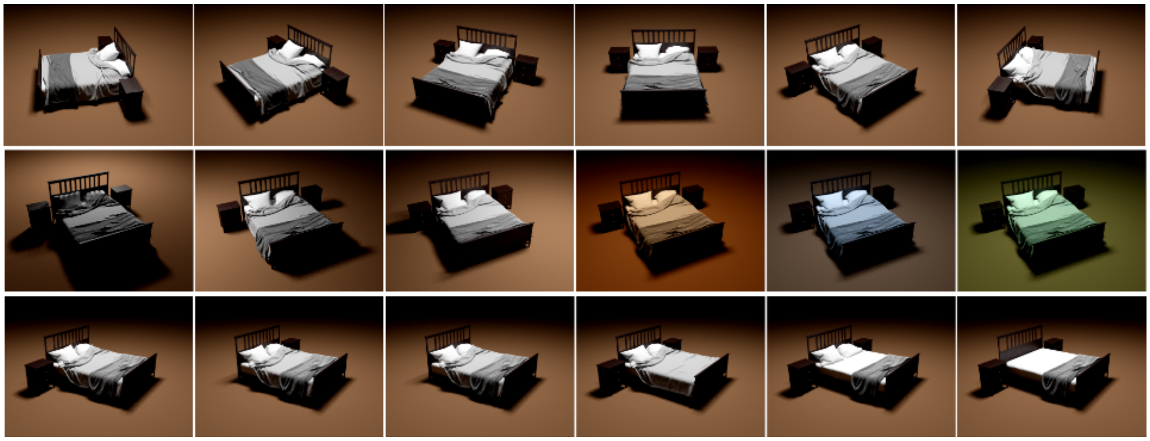


Figure 4.3: Rendered images of an IKEA bed. Generating synthetic data allows us to sample image properties in a controlled way. The top row shows sampling of viewpoint angle. In the middle row we sample lighting conditions, and in the bottom row occlusions by controlling the placement of night stands, and linens.

manually locate those same points in 3D space on a preselected model. Finally, perform PnP optimization [Lepetit et al., 2009] to learn a projection matrix from 3D points to 2D image coordinates, from which angle labels are calculated. Both methods are unsatisfying. For many downstream applications a coarse pose classification is not enough, and the complex point correspondence based process is time consuming and expensive to crowdsource. By generating synthetic, rendered, images one can create extremely large datasets, with any level of granularity for the labels.

In this chapter we explore the benefits of synthetically generated data for viewpoint estimation. We utilize a large database of accurate, highly detailed, 3D models to create a large number of synthetic images. To diversify the generated data we vary many of the rendering parameters. We use the generated dataset to train a deep convolutional network using a loss function that is optimized for viewpoint estimation. Our experiments show that models trained on rendered data are as accurate as those trained on real images. We further show that synthetic data can be also be used successfully during validation, opening up opportunities for large scale evaluation of trained models.

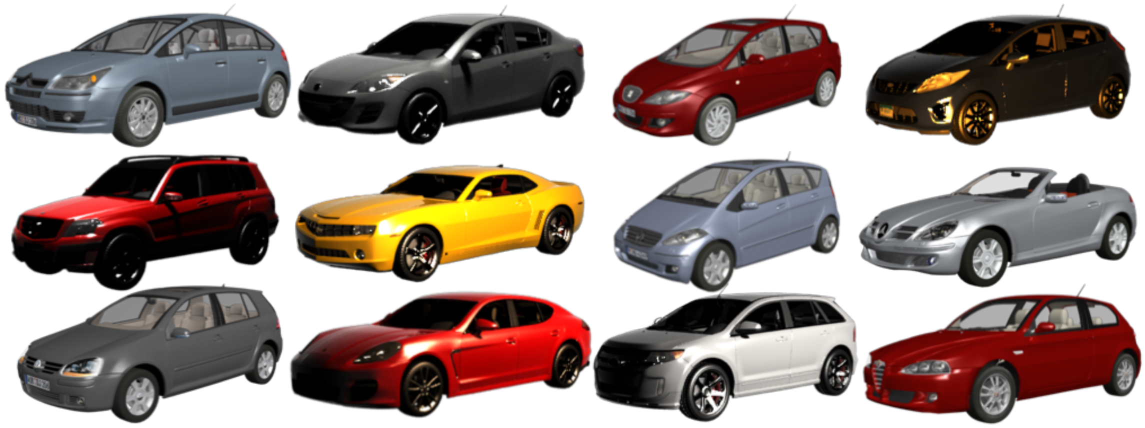


Figure 4.4: A number of models from our database of highly detailed 3D CAD models of cars. It contains a large variety of car types. Notice some of the variablity we introduce in the rendering process. For example, consider the car in the top-right corner. It is lighted with a low intensity, yellowish light. In comparison, the cars below it are lighted with a particularly strong, white, light.

4.2 Data Generation Process

To highlight the benefits of synthetic data we opt to focus on the car object class. We use a database of 91 highly detailed 3D CAD models obtained from Dosch Design[1] and Turbo Squid[2]. Figure 4.4 shows a few of the models used. For each model we perform the following procedure. We define a sphere of radius R centered at the model. We create virtual cameras evenly spaced on the sphere in one degree increments over rings at 5 elevations: $-5°, 0°, 10°, 20°, 30°$. Each camera is used to create a render of one viewpoint of the object. For each rendered image we modify the following rendering parameters:

Lighting position We uniformly sample the location of a directed light on the sphere, in elevations between $10°$ to $80°$.

Light Intensity We uniformly sample the Luminous power (total emitted visible light power measured in lumens) betwen 1,400 and 10,000. A typical 100W incandescent light bulb emits about 1500 lumens of light, and normal day light is between 5,000 and 10,000 lumens. The amount of power needed by the light source also depends on the size of the object, and the distance of the light source from it. As models

[1] www.doschdesign.com

[2] www.turbosquid.com

Light Type	Candle	40W Tungsten	100W Tungsten	Halogen	Carbon Arc	High Noon Sun	Direct Sunlight	Overcast Sky	Clear Blue Sky
Temperature	1900	2600	2850	3200	5200	5400	6000	7000	20000
Color	🟧	🟧	🟧	⬜	⬜	⬜	⬜	🟦	🟦

Figure 4.5: Temperature of light sources used in rendering process. We randomly choose one of these temperatures when for each render created by our system.

were built at varying scales, the sampling of this parameter might require some adjustments between models.

Light Temperature We randomly pick one of $K = 9$ different light temperature profiles. Each profile is designed to mimic a real world light scenario, such as midday sun, tungsten light bulb, overcast sky, halogen light, etc.

Camera F-stop We sample the camera aperture F-stop uniformly between 2.7 and 8.3. This parameter controls both the amount of light entering the camera, and the depth of field in which the camera retains focus.

Camera Shutter Speed We uniformly sample shutter speeds between $^1/_{25}$ and $^1/_{200}$ of a second. This parameters controls the amount of light entering the camera.

Lens Vignetting This parameter simulates the optical vignetting effect of real world cameras. Vignetting is a reduction of image brightness and saturation at the periphery compared to the image center[3]. For 25% of the images we add vignetting with a constant radius.

Background Each rendered image is layered with a natural image background patch. The patches are randomly selected from PASCAL training that are not from the "Car" class.

For rendering the images we use 3DS MAX, with the production quality VRAY rendering plug-in [vra]. There has been considerable evidence that data augmentation methods can contribute to the accuracy of trained deep networks [Wu et al., 2015]. Following this, we also augment our rendered images by creating new images in the following way

Compression Effects Most of the images that the trained classifier will get to observe during test time are JPEG compressed images. Our renders are much cleaner, and

[3] https://en.wikipedia.org/wiki/Vignetting

are saved in lossless PNG format. In most cases JPEG compression is not visually distinctive enough to be noticed by human observers, but it was shown that these compression artifacts can influence classifier performance. We therefore JPEG compress all our renders.

Color Cast For each channel of the image, with probability 50% we add a constant value sampled uniformly in $(-20, 20)$.

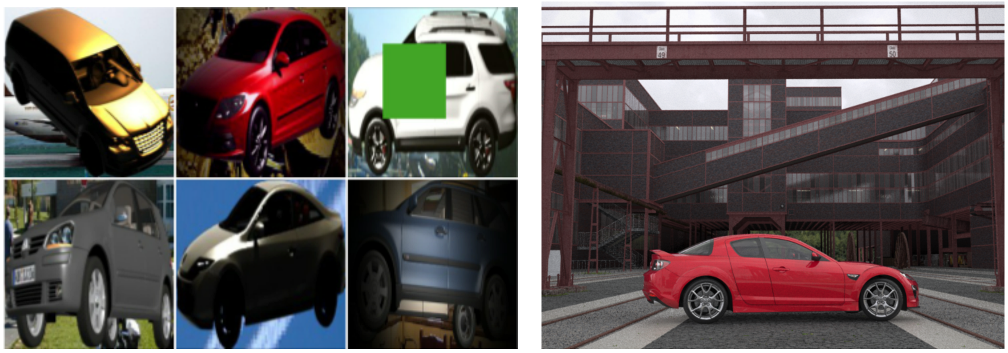
Channel Swap With probability 50% we randomly swap the color channels.

Image degradation Some ground truth bounding boxes in the PASCAL dataset are very small. The resulting object resolution is much different than our higher resolution rendered images. In order for the deep model to learn to classify correctly these lower resolution images, we first estimate the bounding box area using the PASCAL training set, and then down-sample 25% of our images to a size that falls in the lower 30% of the distribution.

Occlusions We want our model to be robust to occlusions. We therefore randomly place rectangular patches either from the PASCAL training set, or of uniform color, on the images. The width and height of the rectangle are uniformly sampled to be between 20% and 60% of the image size.

Finally, we perform a train/test split of the data such that images from 90 models are used for training, and one model is held out for testing. The resulting datasets have 819,000 and 1,800 images respectively. Figure 4.6(a) shows a number of rendered car images after the application of the data augmentation methods listed above. We name this dataset RenderCar.

With the steady increase in computational power, the lowered cost of rendering software, and the availability of 3D CAD models, for the first time it is now becoming possible to fully model not just the object of interest, but also its surrounding environment. Figure 4.6(b) shows a fully rendered images from one such scene. Rendering such a fully realistic image takes considerably more time and computational resources than just the model. Note the interaction between the model and the scene – shadows, reflections, etc. While we can not, at the moment, produce a large enough dataset of such renders to be used for training purposes, we can utilize a smaller set of rendered scene images for *validation*. In Section 4.4 we show that such a set is useful for evaluating models trained on real images.



(a) Sample Images From RenderCar

(b) Fully Rendered Scene

Figure 4.6: (a) A number of training images after application of all data augmentation processing methods. The natural backgrounds are randomly selected from PASCAL training images from all but the *car* class. (b) To Evaluate the usefulness of rendered images as validation data we construct a scene in which the object is placed in a fully modeled environment. We then render fully realistic images of the object from 360 views. We term this dataset RenderScene, and report performance on it of models trained on real and synthetic images. In Table 4.1, we show that the RenderScene dataset is challenging for models trained on real images.

4.3 Deep Network Architecture and Loss Function

Following the success of deep learning based approaches in object detection and classification, We base our network architecture on the widely used AlexNet model [Krizhevsky et al., 2012] with a number of modifications. Figure 4.7 shows our modified AlexNet architecture. The most important of those changes is our introduced loss function. Most previous work on viewpoint estimation task have simply reduced it to a regular classification problem. The continuous viewpoint space is discretized into a set of class labels, and a classification loss, most commonly SoftMax loss, is used to train the network. This approach has a number of appealing properties: (1) SoftMax is a well understood loss and one that has been used successfully in many computer vision tasks. (2) The predictions of a SoftMax are associated with probability values that indicate the model's confidence in the prediction. (3) The classification task is easier than a full regression to a real value angle which can reduce over-fitting.

There is one glaring problem with this reduction - it does not take into account the circular nature of angle values. The discretized angles are just treated as class labels, and any mistake the model makes is penalized in the same way. There is much information

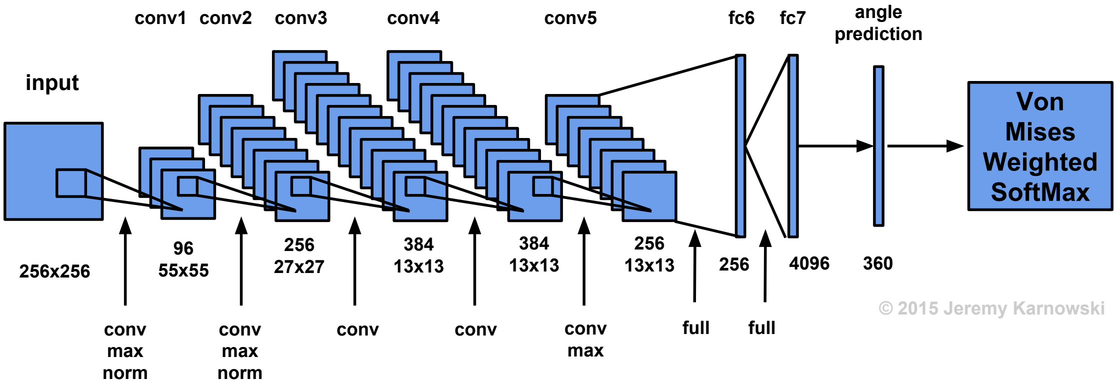


Figure 4.7: Our model is based on the popular AlexNet architecture, with a number of modifications. The most important is a new loss layer based on a weighted SoftMax, with weights established according to a Von Mises distance Kernel. The loss layer encourages similarities in estimation between nearby views. Note also that we reduce the fc6 fully connected layer to 256 neurons. This dimensionality reduction reduces over- fitting. Visualization modified with permission from [Karnowski, 2015].

that is lost if the distance between the predicted angle and the ground truth is not taken into account when computing the error gradients. We therefore propose to use a generalization of the SoftMax loss function that allows us to take into account distance-on-a-ring between the angle class labels:

$$E = -\frac{1}{N}\sum_{n=1}^{N}\sum_{k=1}^{K} w_{l_n,k}\log(p_{n,k}), \tag{4.1}$$

where N is the number of images, K the number of classes, l_n the ground truth label of instance n, and p_n the probability of the class k in example n. $w_{l_n,k}$ is a Von Mises kernel centered at l_n

$$w_{l_n,k} = \exp(-\frac{\min(|l_n - k|\,, K - |l_n - k|)}{\sigma^2}), \tag{4.2}$$

where σ controls the width of the kernel.

The Von Mises kernel implements a circular normal distribution centered at $0°$. This loss formulation penalizes predictions that are far away, in angle space, more than smaller mistakes. By acknowledging the different types of mistakes, there is more information that flows back through the gradients. An intuitive way to understand this loss, is as a matrix of class-to-class weights as is shown in Figure 4.8. The y-axis shows true class, and the

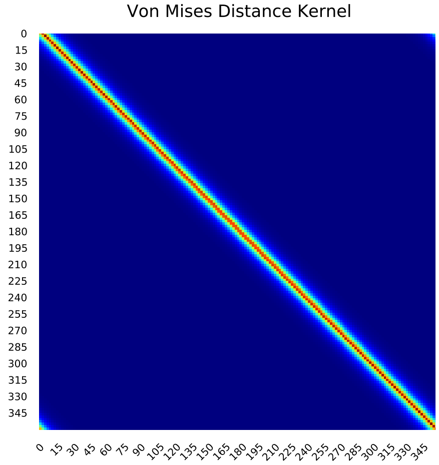


Figure 4.8: Visualization of the Von Mises kernel used in our proposed loss function (4.2). The y-axis shows true class, and the x-axis shows the predicted class. The values indicate the weights w. Weight is given to nearby classes to penalize smaller mistakes less severely. This guides the optimization procedure smoothly towards the correct angle.

x-axis shows the predicted class. The values indicate the weights w. A standard Softmax loss would have a weight of 1 on the diagonal of the matrix, and zero anywhere else. In the angle-aware version, some weight is given to nearby classes, and there is a wrap around, such that there is also weight that is given when the distance between the predicted and true class crosses the $0°$ boundary (Note the values at the top-right and bottom-left corners of the matrix). See Section 4.4 for a comparison between this loss function and standard Softmax.

4.4 Evaluation

Our main objective is to evaluate the usefulness of synthetic, rendered, data as a training tool. We start by comparing the result of the deep network architecture described in Section 4.3 on two fine-grained viewpoint estimation datasets. The datasets used are:

CMU-Car. The MIT street scene data set [Bileschi, 2006] was augmented by Boddeti et al. [Boddeti et al., 2013] with landmark annotations for $3,433$ cars. To allow for evaluation of precise viewpoint estimation we further augment this data set by providing camera matrices for $3,240$ cars. To get the camera viewpoint matrices, we manually annotated a

3D CAD car model with the same landmark locations as the images and used the POSIT algorithm to align the model to the images. To ensure a clean ground truth set, we then back projected the 3D points to the 2D plane and only used cars for which the sum of reprojection error over all landmark points was smaller than 8 pixels. The CAD model used was different from those used to generate renders.

PASCAL3D+. The dataset built by [Xiang et al., 2014] augments 12 rigid categories of the PASCAL VOC 2012 with 3D annotations. Similar to above, the annotations were collected by manually matching correspondence points from CAD models to images. On top of those, more images are added for each category from the ImageNet dataset. PASCAL3D+ images exhibit much more variability compared to the existing 3D datasets, and on average there are more than 3,000 object instances per category. Most prior work however do not use the added ImageNet data and restrict themselves to only the images from PASCAL VOC 2012.

We also define two synthetic datasets, which consist the the two types of rendered images described in Section 4.2 – RenderCar and RenderScene. The RenderCar dataset includes images from the entire set of 3D CAD models of vehicles. We use the various augmentation methods described above. For RenderScene we use a single car model placed in a fully modeled environment which depicts a fully realistic industrial scene as shown in figure 4.6(b), rendered over all 1800 angles as described in Section 4.2.

We focus our evaluation on the process of viewpoint prediction, and work directly on ground truth bounding boxes. Most recent work on detecting objects and estimating their viewpoint, first employ an RCNN style bounding box selection process.

We split each dataset into train/validation sets and train a separate deep model on each one of the training sets. We then apply every model to the validation sets and report the results. We perform viewpoint estimation on ground truth bounding boxes for images of the car class. For PASCAL3D+ and CMUCar these bounding boxes were obtained using annotators, and for the rendered images these were automatically created as the tightest rectangle that contains all pixels that belong to the rendered car.

Figure 4.9 shows the median angular azimuth error of 4 models when evaluated on the PASCAL validation set. While the model trained on PASCAL performs better than the one trained on rendered data it has an unfair advantage - the rendered model needs to overcome domain adaptation as well as the challenging task of viewpoint estimation. Note that the model trained on rendered data performs much better than the one trained on CMUCar. To us this indicates that some of the past concerns about generality of models trained from synthetic data may be unfounded. It seems that the adaptation task from synthetic data is not harder than from one real image set to another. Lastly note that best performance is

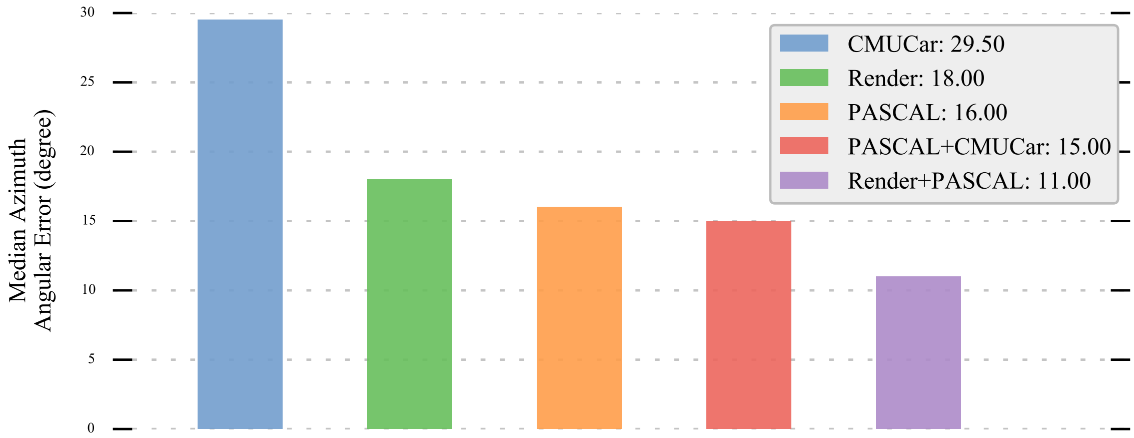


Figure 4.9: Median angular error on the PASCAL test set for models trained on PASCAL, CMUCar, and RenderCar. The model trained on PASCAL is the best model trained on a single dataset, but it has the advantage of having access to PASCAL images during training. Note that the model trained on synthetic data performs better than the one trained on CMUCar. When combining both rendered and real images, performance increases. This is not only due to an increase in the number of images. A model trained on a combination of PASCAl and CMUCar images does not perform as well.

achieved when combining real data with synthetic data. This model achieves better performance than when combining PASCAL data with images from CMUCar. As CMUCar images are all street scene images taken by a standing person they have a strong bias, and do not add much to a model's ability to generalize.

Figure 4.10 shows a number of example results on the ground truth bounding boxes from the PASCAL3d+ test set. Successful predictions are shown in the top two rows, and failure cases in the bottom row. The test set is characterized by many cropped and occluded images, some of very poor image quality. Mostly the model is robust to these, but when it fails it is usually due to these issues, or the $180°$ ambiguity.

Table 4.1 shows model error for azimuth estimation for all train/validation combinations. First, it is easy to spot dataset bias - the best performing model on each dataset is the one that was trained on a training set from that dataset. This is consistent with the findings of [Torralba and Efros, 2011] which show that there are strong dataset bias effects. It is also interesting to see that some datasets are better for generalizing than others. The two right most columns average prediction error across multiple datasets. The *Avg* column averages across all datasets, and *Avg On Natural* averages the results on the 3 columns

	Validation						
Training	PASCAL	CMUCar	RenderCar	Render Full Scene	PASCAL + CMUcar	Avg	Avg On Natural
PASCAL	16°	6°	18°	14°	8°	12.4°	10°
CMUCar	29.5°	**2**°	27°	13°	5°	15.3°	12.17°
RenderCar	18°	6°	2°	8°	8°	8.4°	10.67°
PASCAL + CMUCar	15°	3°	13°	9°	5°	9°	7.67°
PASCAL + RenderCar	**11**°	6°	4°	6°	6°	6.6°	7.67°
CMUCar + RenderCar	15°	**2**°	2°	**5**°	4°	5.6°	7°
PASCAL + CMUCar + RenderCar	12°	**2**°	**1**°	8°	**3**°	**5.2**°	**5.67**°

Table 4.1: Median Angular Error of car viewpoint estimation. The estimation is done on ground truth bounding boxes. Note the distinct effect of dataset bias - the best model on each dataset is the one that was trained on a training set from the same dataset. Also note that, on average, the model trained on rendered images performs similarly to that trained on PASCAL, and better than one that is trained on the CMUCar dataset. Combining rendered data with natural images produces lower error than when combining two natural-image datasets. Combining all three datasets provides lowest error.

that use natural images for testing. Notice that the model trained on PASCAL data performs better overall than the one trained on CMUcar. Also notice that the model trained on RenderCar performs almost as well as the one trained on PASCAL. When combining data from multiple datasets the overall error is reduced. Unsurprisingly, combining all datasets produces the best results. It is interesting to note, however, that adding rendered data to one of the natural image datasets produces better results than when combining the two real-image ones. We conclude that this is because the rendered data adds variation in two ways: (1) It provides access to regions of the label space that were not present in the small natural- image datasets. (2) The image statistics of rendered images are different than those of natural images and the learner is forced to generalize better in order to classify both types of images. We further examine the effect of combining real and synthetic data in Sections 4.4.2 and 4.4.1.

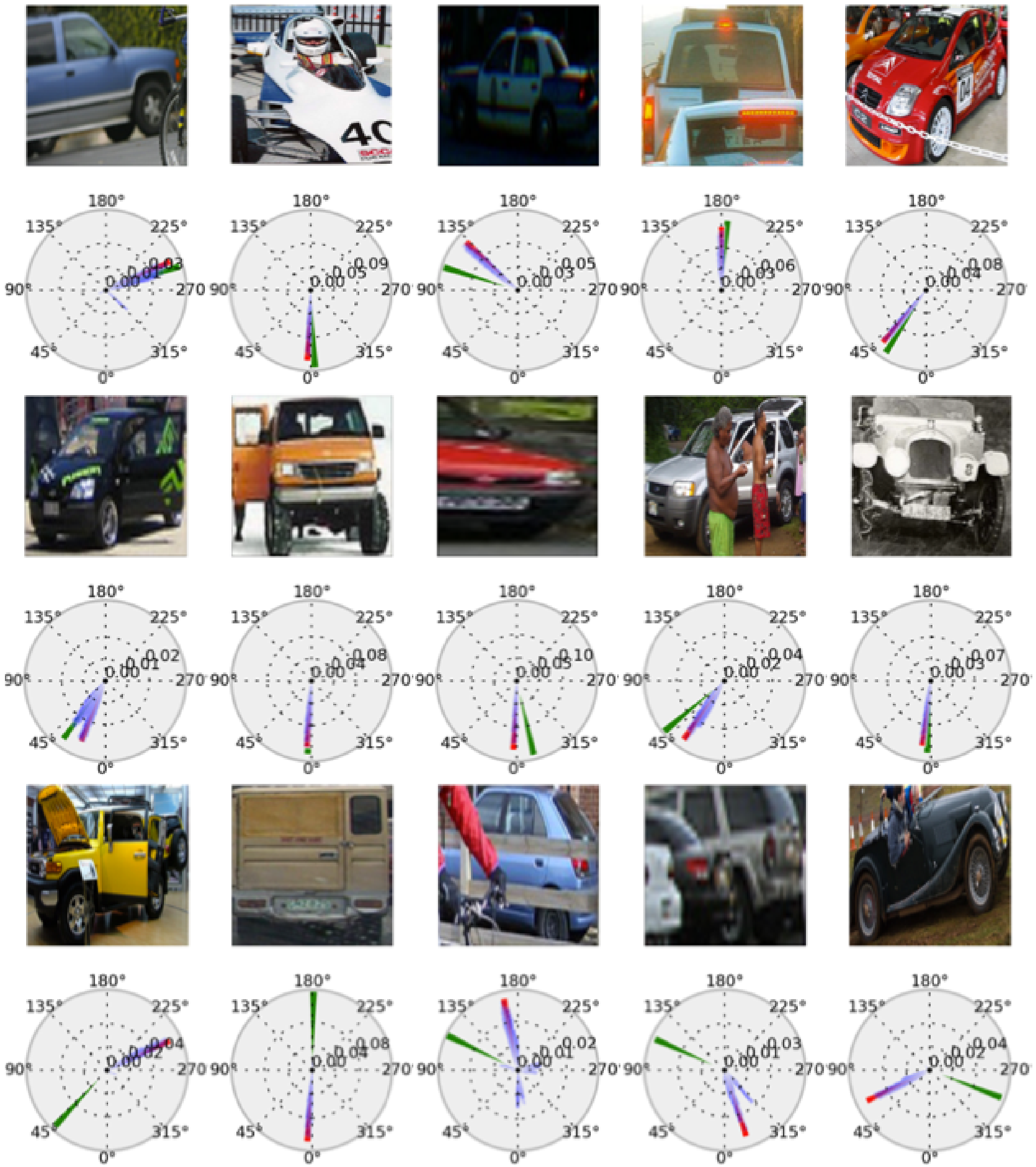


Figure 4.10: Example results of our proposed method on a sample of images from the test set of PASCAL3D+. Below each image we show the viewpoint probabilities assigned by the model (blue), predictions (red), and ground truth (green). Last row shows failure cases. Heavy occlusion or cropping and 180° ambiguity are the most common failures.

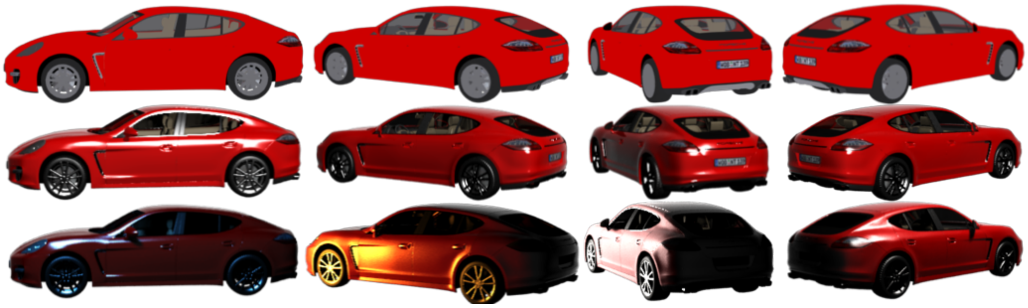


Figure 4.11: We evaluate the effect of render quality by creating 3 different conditions. The basic case uses simple object material, and ambient lighting (top row). Our intermediate setup has complex material and ambient lighting (middle row), and the sophisticated case has complex material, and directional lighting whose location, color, and strength are randomly selected.

4.4.1 Render Quality

Renders can be created in varying degrees of quality and realism. Simple, almost cartoon-like renders are fast to generate, while ones that realistically model the interplay of lighting and material can be quite computationally expensive and time consuming. Are these higher quality renders worth the added cost?

Figure 4.11 shows 3 render conditions we use to evaluate the effect of render quality on system performance. The top row shows the basic condition, renders created using simplified model materials, and uniform ambient lighting. In the middle row are images created using a more complex rendering procedure – the materials used are complex. They more closely resemble the metallic surface of real vehicles. We still use ambient lighting when creating them. The bottom row shows images that were generated using complex materials and directional lighting. The location, color, and strength of the light source are randomly selected.

Figure 4.12 shows median angular error as a function of dataset size for the 3 render quality conditions. We see that when the rendering process becomes more sophisticated the error decreases. Interestingly, when using low quality renders the error increases once there are too many renders and they dominate the training set. We do not see this phenomena with higher quality renders. We conclude that using complex materials and lighting is an important aspect of synthetic datasets.

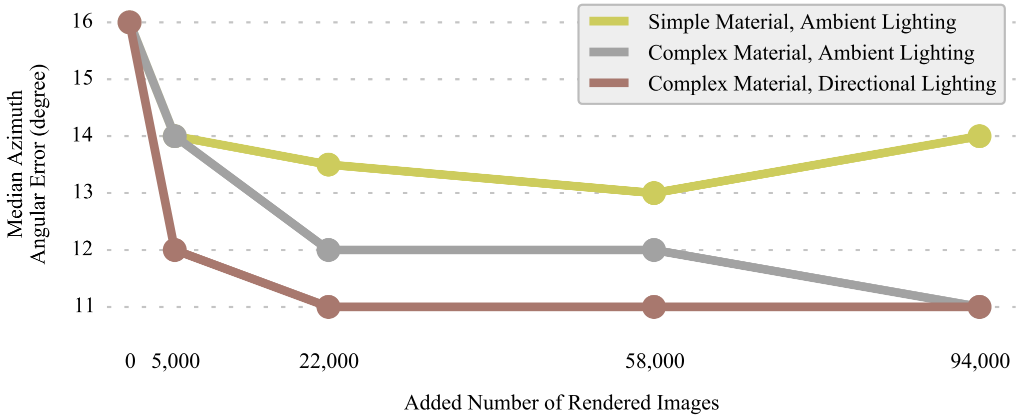


Figure 4.12: Error as a function of dataset size for the 3 render quality conditions described in Figure 4.11. All methods add rendered images to the PASCAL training set. There is a decrease in error when complex model materials are used, and a further decrease when we add random lighting conditions.

	Training Set			
	PASCAL	Render	Adaptive Balancing	Random Balancing
Median Angular Error	26.5°	26.0°	16.0°	17.5°

Table 4.2: Median angular error on a sample of the PASCAL test set which is uniformly distributed in angle labels. When all angles are equally represented the performance of rendered based training set is superior.

4.4.2 Balancing a Training Set Using Renders

So far we have seen that combining real images with synthetically generated ones improves performance. Intuitively this makes sense, the model is given more data, and thus learns a better representation for the task. But is there something more than that going on? Consider a trained model. What are the properties we would want it to have? We would naturally want it to be as accurate as possible. We would also want it to be unbiased – to be similarly accurate in all locations in feature/label space. But biases in the training set makes it hard to achieve this. This is one way in which rendered data can be useful.

In Figure 4.13 we quantify this requirement. It shows the accuracy of 3 different models as a function of the angle range in which they are applied. For example, a model trained on 5,000 PASCAL images (top row), has a little bit over 80% accuracy when it is

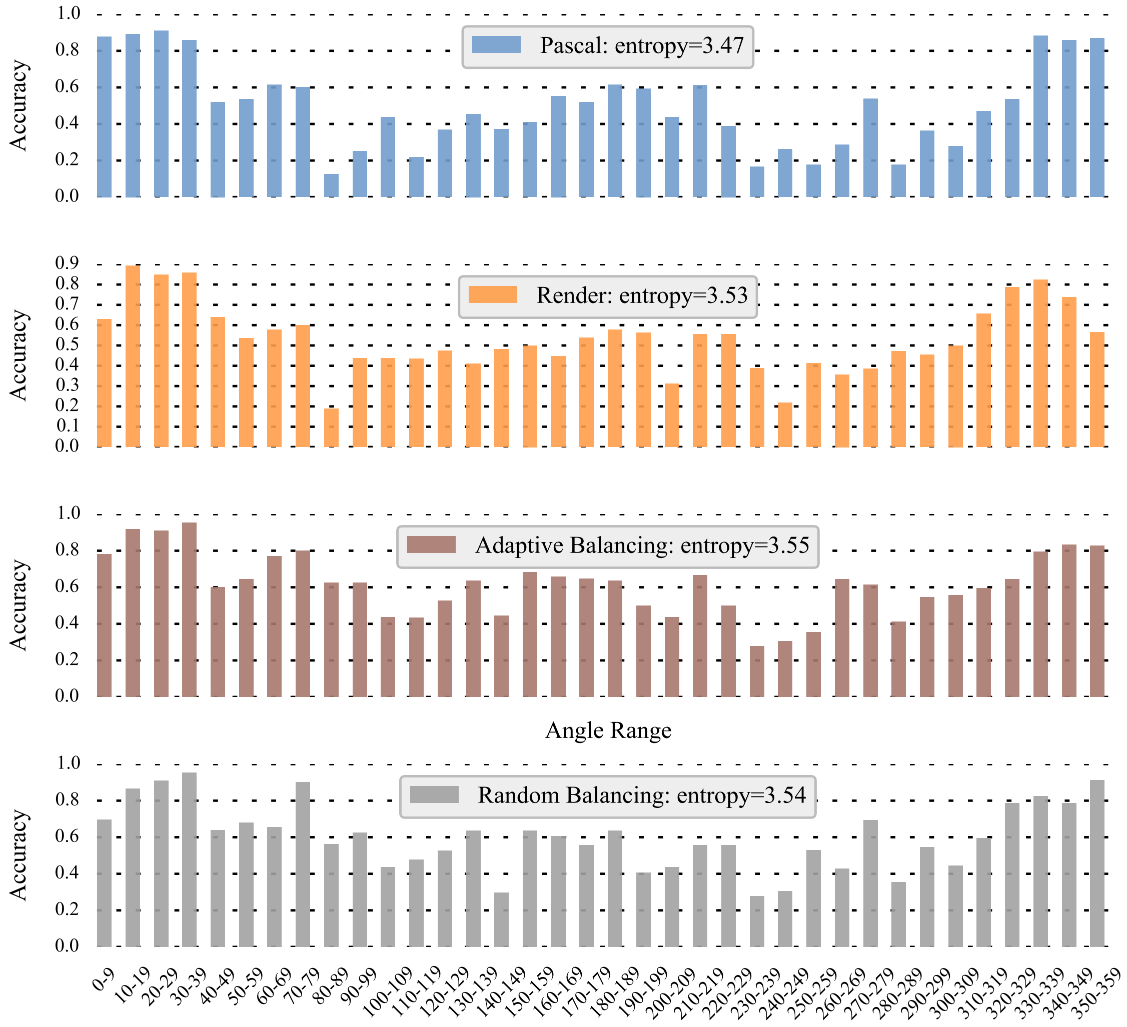


Figure 4.13: Evaluating model accuracy as a function of test image angle. The model evaluated in the top row was trained on PASCAL images. Ideally a model should perform uniformly well across all angles. However, we see that the accuracy mirrors the train set distribution (see Figure 4.2). We calculate the entropy of each distribution to show deviation from uniform. The model shown in the middle row was trained on rendered images. Its entropy is higher, but surprisingly it is not uniform as well. This is due to biases in the test set. The bottom row shows the accuracy distribution of a model for which rendered images were used to balance the training set. We see that its entropy is highest. All models are tested on PASCAL test images.

applied at test time on images with a ground truth label in $[0, 9]$.

We want the shape of the accuracy distribution to be as close to uniform as possible. Instead, we see that the model performs much better on some angles, such as $[0, 30]$, than others, $[80, 100]$. If the shape of the distribution seems familiar it is because it closely mirrors the training set distribution shown in Figure 4.2. We calculate the entropy of each model's accuracy distribution as a tool for comparison. Higher entropy indicates a more uniform distribution. Total uniformity would have a score of 5.88. When using a completely balanced training set of rendered images (2nd row), the test entropy increases. This is a positive sign, as the model contains less angle bias.

Rendered images can be used as a way to balance the training set to reduce bias while still getting the benefits of natural images. We experiment with two methods for balancing which we call adaptive balancing (3rd row), and random balancing (bottom row). In adaptive balancing we sample each angle reversely proportional to its frequency in the training set. This method transforms the training set to uniform using the least number of images. In random balancing the added synthetic images are sampled uniformly. As the ratio of rendered images in the training set increases, the set becomes more uniform. For this experiment both methods added 2,000 images to the PASCAL training set. Interestingly, the two balancing methods have a similar prediction entropy. However, no method's distribution is completely uniform. From that we can conclude that there are biases in the test data other than angle distribution, or that some angles are just naturally harder to predict than others.

These results raise an interesting question: how much of the performance gap we have seen between models trained on real PASCAL images, and those trained on rendered data stems from the angle bias of the test set? Table 4.2 shows the median errors of these models on a sample of the PASCAL test set in which all angles are equally represented. Once the test set is uniform we see that models based on a balanced training set perform best, and the model trained solely on PASCAL images has the worst accuracy.

Lastly, we hold constant the number of images used for training and vary the proportion of rendered images. Table 4.3 shows the error of models trained with varying proportions of real-to-rendered data. Models are trained using 5,000 images – the size of the PASCAL training set. Notice that there is an improvement when replacing up to 50% of the images with rendered data. This is likely due to the balancing effect of the renders on the angle distribution which reduces bias. When most of the data is rendered, performance drops. This is likely because the image variability in renders is smaller than real images. More synthetic data is needed for models based purely on it to achieve the lowest error – 18°.

	Rendered Image Proportion in Training Set				
	0%	25%	50%	75%	100%
Median Angular Error	16°	14°	14°	15°	22°

Table 4.3: Median angular error on a the PASCAL test set as a function of the proportion of rendered images used in training. We keep the size of the training set fixed at 5,000 images (the full size of the PASCAL training set) and modify the proportion of rendered images used. Notice that there is an improvement by replacing up to 50% of the images with renders. The renders help balance the angle distribution and reduce bias.

	Training Set Size							
	10k		50k		330k		820k	
	SM	wSM	SM	wSM	SM	wSM	SM	wSM
Median Angular Error	31°	27.5°	54.5°	21°	26°	21.5°	23°	18°

Table 4.4: Effect of training set size and loss layer. Up to and including 330,000 images, all training sets were comprised of 50% cropped images. For the biggest training set, all images after the first 330,000 were random crops of former images. We generally see a trend of better performance with increased size of training set, but the effect quickly diminishes. Clearly, more than just the raw size of the rendered training set influences performance. The Von Mises weighted SoftMax (wSM) performs better for each training set size than regular SoftMax (SM).

4.4.3 Size of Training Set

We examine the role of the raw number of generated images in performance. We keep the number of models constant at 90, and uniformly sample rendered images from our large pool of rendered images. Many of the images in PASCAL are only partially visible, and we want the models to learn this, so in each set half the rendered images show full cars, and half contain random 60% crops Table 4.4 summarizes the results of this experiment. Better performance is achieved when increasing the number of renders, but there are diminishing returns. There is a limit to the variability a model can learn from a fixed set of CAD models, and one would need to increase the size of the model set to overcome this. Obtaining a large amount of 3D models can be expensive and so this motivates creation of large open source model datasets.

σ	2	3	4	10	15
Effective Width	6°	8°	12°	30°	50°
Median Angular Error	13	14	14	13	15

Table 4.5: Effect of kernel width. σ values for the Von Mises kernel used as a loss function. The loss function is not sensitive to the selection of kernel width.

4.4.4 Loss Layer

An interesting property of the Von Mises kernel-based weighted SoftMax loss function is that it is impossible to obtain zero loss. In a regular SoftMax, if the model assigns 100% probability to the correct class it can achieve a loss of zero, but in the weighted case, when there is some weight assigned to more than one class, a perfect prediction will actually result in infinite loss – if the correct class has 100% probability, then all other classes have probability value of zero, and so the log-loss assigned to it will be infinite. Minimum loss will be reached when prediction probabilities are spread out across nearby classes. This is both a downside of this loss function, but also its strength. It does not allow the model to make wild guesses at the correct class. Nearby views are required to have similar probabilities. In a way we are trading some angle resolution with added stability of prediction.

Table 4.4 compares the results of SoftMax loss based models and models trained using our weighted SoftMax layer over a number of rendered dataset sizes. The weighted loss layer performs better for all dataset sizes. This confirms our hypothesis that viewpoint estimation should not be treated as a standard classification task. Most experiments in this section were performed using $\sigma = 2$ as the standard deviation of the Von Mises kernel in Equation (4.2). This amounts to an effective width of 6°, meaning that predictions that are farther away from the ground truth prediction will not be assigned any weight. Table 4.5 shows the effect of varying this value. Interestingly we see that the method is fairly robust to this parameter, and performs well for a wide range of values. It appears that even a relatively weak correlation between angles provides benefits.

4.4.5 Occlusion

The PASCAL test data contains many instances in which the car is partially occluded. When augmenting the synthetic data we add randomly sized occlusions to a subset of the rendered images. Table 4.6 shows the effect on performance of the added occlusions. It is clear that there is some benefit from generating occlusions, but when too many of the

	Proportion of training images with occlusion					
	0.0	0.1	0.35	0.4	0.5	1.0
Median Angular Error	25	25	21	25	28	26

Table 4.6: Effect of added occlusions to the training set. All models evaluated used datasets of 50,000 rendered images.

training images are occluded it becomes harder for the model to learn. What is the best way to generate occlusions? In our work we have experimented with simple, single color, rectangular occlusions, as well as occlusions based on random patches from PASCAL. We saw no difference in model performance. It would be interesting to examine the optimal spatial occlusion relationship. This is likely to be object class dependent.

4.5 Discussion

In this chapter we proposed the use of synthetically generated rendered images as a way to automatically build datasets for viewpoint estimation. We have shown that the accuracy of models trained on rendered images is close to the accuracy when training on real images, and is even at times higher. We have noticed that the gap in performance can be explained by domain adaptation. Moreover, models trained on a combination of synthetic data and natural images perform better than ones trained only on real images.

The need for large scale datasets is increasing due to the growing size of the models researchers build. Based on the results detailed here we believe that synthetic data should be an important part of any strategy for creating datasets. Most realistically we feel that a combination a small set of real images, carefully annotated by human operators, combined with a larger number of synthetic renders, with automatically assigned labels, will provide the best cost-to-benefit ratio.

This strategy is not limited to viewpoint estimation, and can be employed for a range of computer vision tasks. Specifically we feel that future research should focus on human pose estimation, depth prediction, wide-baseline correspondence learning, and structure from motion.

Acknowledgments

All graphs in this chapter were made using Matplotlib [Hunter, 2007]. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research.

Chapter 5

Ontological Supervision for Fine Grained Classification of Street View Storefronts

5.1 Introduction

Following the popularity of smart mobile devices, search engine users today perform a variety of locality-aware queries, such as *Japanese restaurant near me*, *Food nearby open now*, or *Asian stores in San Diego*. With the help of local business listings, these queries can be answered in a way that is tailored to the user's location.

Creating accurate listings of local businesses is time consuming and expensive. To be useful for the search engine, the listing needs to be accurate, extensive, and importantly, contain a rich representation of the business category. Recognizing that a JAPANESE RESTAURANT is a type of ASIAN STORE that sells FOOD, is essential in accurately answering a large variety of queries. Listing maintenance is a never ending task as businesses often move or close down. In fact it is estimated that 10% of establishments go out of business every year, and in some segments of the market, such as the restaurant industry, the rate is as high as 30% [Parsa et al., 2005].

The turnover rate makes a compelling case for automating the creation of business listings. For businesses with a physical presence, such as restaurants and gas stations, it is a natural choice to use data from a collection of street level imagery. Probably the most recognizable such collection is Google Street View which contains hundreds of millions of 360° panoramic images, with geo-location information.

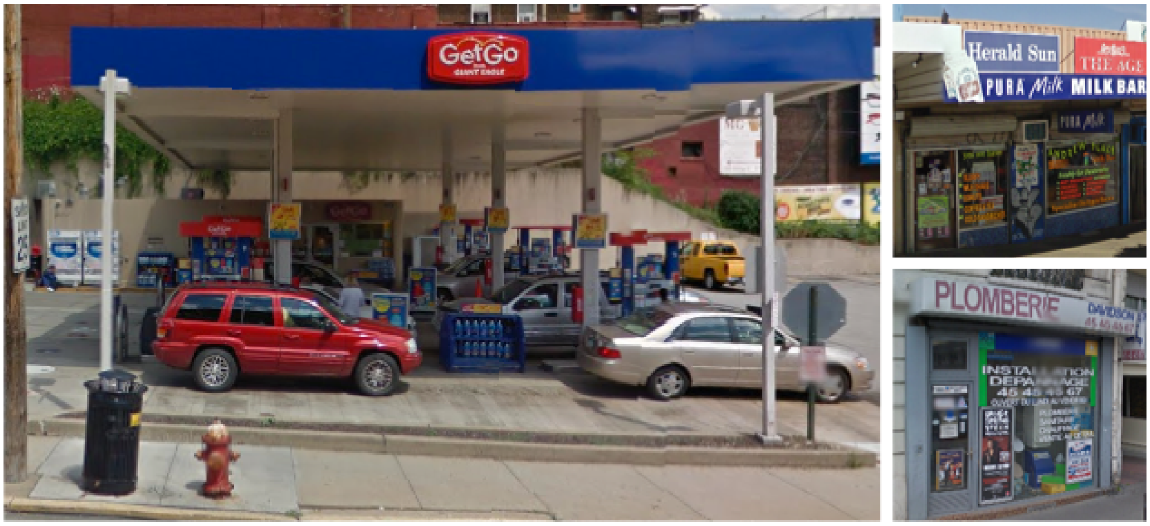


Figure 5.1: The multi label nature of business classification is clear in the image on the left; the main function of this establishment is to sell fuel, but it also serves as a convenience store. The remaining images show the fine grained differences one expects to find in businesses. The shop in the middle image is a grocery store, the one on the right sells plumbing supplies; visually they are similar.

In this work we focus on business storefront classification from street level imagery. We view this task as a form of multi-label fine grained classification. Given an image of a storefront, extracted from a Street View panorama, our system is tasked with providing the most relevant labels for that business from a large set of labels. To understand the importance of associating a business with multiple labels, consider the gas station shown in Figure 5.1 (left). While its main purpose is fueling vehicles, it also serves as a convenience or grocery store. Any listing that does not capture this subtlety will be of limited value to its users. Similarly, stores like Target or Walmart sell a wide variety of products from fruit to home furniture, all of which should be reflected in their listings. The problem is fine grained as business of different types can differ only slightly in their visual appearance. An example of such a subtle difference is shown in Figure 5.1 (right). The top image shows the front of a grocery store, while the image on the bottom is of a plumbing supply store. Visually they are similar. The discriminative information can be very subtle, and appear in varying locations and scales in the image; this, combined with the large number of categories needed to cover the space of businesses, require large amounts of training data.

The contribution of this work is two fold. First, we provide an analysis of challenges

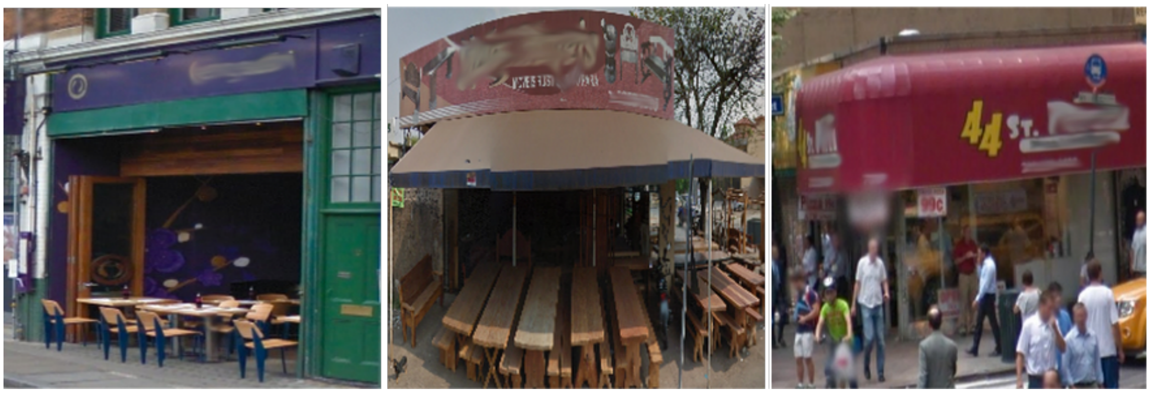


Figure 5.2: Examples of 3 businesses with their names blurred. Can you predict what they sell? Starting from left they are: Sushi Restaurant, Bench store, Pizza place. The intra-class variation can be bigger than the differences between classes. This example shows that the textual information in images can be important for classifying the business category. However, relying on OCR has many problems as discussed in Section 5.2.

of a storefront classification system. We show that the intra-class variations can be larger than differences between classes (see Figure 5.2). Textual information in the image can assist the classification task, however, there are various drawbacks to text based models: Determining which text in the image belongs to the business is a hard task; Text can be in a language for which there is no trained model, or the language used can be different than what is expected based on the image location (see Figure 5.3). We discuss these challenges in detail in Section 5.2.

Finally, we propose a method for creating large scale labeled training data for fine grained storefront classification. We match street level imagery to known business information using both location and textual data extracted from images. We fuse information from an ontology of entities with geographical attributes to propagate category information such that each image is paired with multiple labels with different levels of granularity. We then train a Deep Convolutional Network that achieves human level accuracy.

5.2 Challenges in Storefront Classification

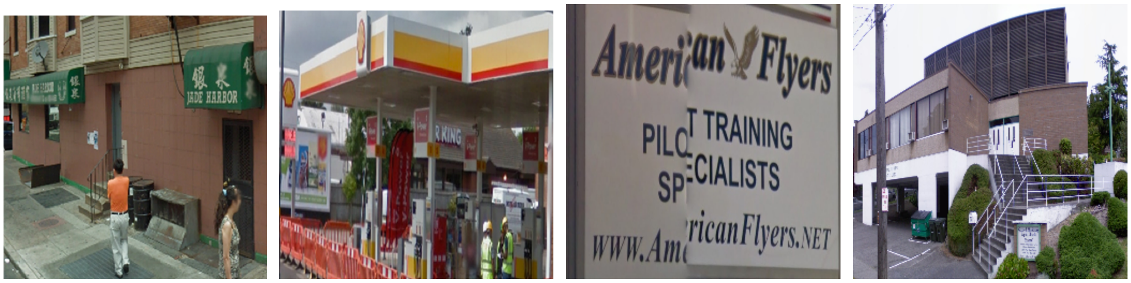
Large Within-Class Variance. Predicting the function of businesses is a hard task. The number of possible categories is large, and the similarity between different classes can be smaller than within class variability. Figure 5.2 shows three business storefronts. Their names have been blurred. Can you tell the type of the business without reading its name? Two of them are restaurants of some type, the third sells furniture, in particular store benches (middle image). It is clear that the text in the image can be extremely useful for the classification task in these cases.

Extracted Text is Often Misleading. The accuracy of text detection and transcription in real world images has increased significantly over the last few years [Wang and Belongie, 2010, Nguyen et al., 2014b], but relying on the ability to transcribe text has drawbacks. We would like a method that can scale up to be used on images captured across many countries and languages. When using extracted text, we need to train a dedicated model per language, this requires a lot of effort in curating training data. Operators need to mark the location, language and transcription of text in images. When using the system it would fail if a business had a different language than what we expect for its location or if we are missing a model for that language (Figure 5.3a). Text can be absent from the image, and if present can be irrelevant to the type of the business. Relying on text can be misleading even when the language model is perfect; the text can come from a neighboring business, a billboard, or a passing bus (Figure 5.3b). Lastly, panorama stitching errors may distort the text in the image and confuse the transcription process (Figure 5.3c).

However, it is clear that the textual parts of the image do contain information that can be helpful. Ideally we would want a system that has all the advantages of using text information, without the drawbacks mentioned. In Section 5.5.3 we show that our system implicitly learns to use textual cues, but is more robust to these errors.

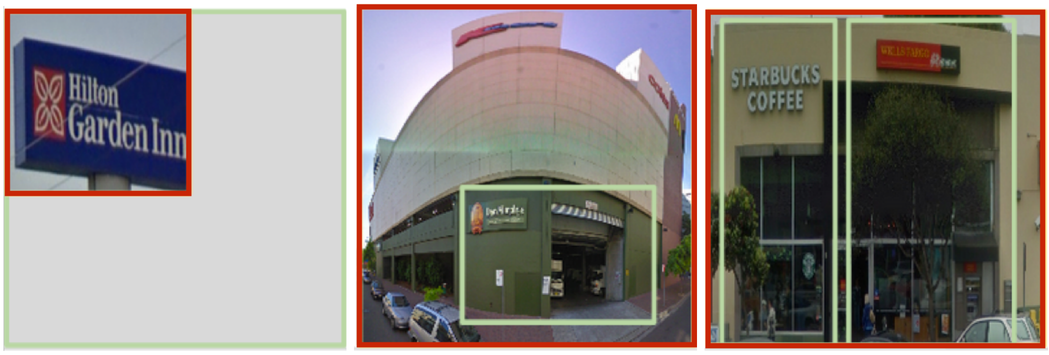
Business Category Distribution. The natural distribution of businesses in the world exhibits a clear "long tail". Some businesses (such as McDonalds) are very frequent, but most of the mass of the distribution is in the large number of businesses that only have one location. This same phenomena is also true of categories. Some labels can have an order of magnitude more samples than others. As an example, for the FOOD AND DRINK category which contains restaurants, bars, cafes, etc, we have roughly 300,000 images, while for the LAUNDRY SERVICE label our data contains only 13,000 images. We note that a large part of the distribution's mass is in these smaller categories.

Labeled Data acquisition. Acquiring a large set of high quality labeled data for training is a hard task in and of itself. We provide operators with Street View panoramas captured at urban areas in many cities across Europe, Australia, and the Americas. The operators are asked to mark image areas that contain business related information. We call these areas *biz-patches*. This process is not without errors. Figure 5.4 shows a number of common mistakes made by operators. The operators might mark only the business signage (5.4a), an area that is too large and contains unneeded regions (5.4b), multiple businesses in the same biz- patch (5.4c).



(a) Unexpected Language (b) Misleading Text (c) Stitching Errors (d) Absent Text

Figure 5.3: Text in the image can be informative but has a number of characteristic points of failure. (a) Explicitly transcribing the text requires separate models for different languages. This requires maintaining models for each desired language/region. If text in one language is encountered in a an area where that language was not expected, the transcription would fail. (b) The text can be misleading. In this image the available text is part of the Burger King restaurant that is behind the gas station. (c) Panorama stitching errors can corrupt text and confuse the transcription process. (d) Textual information can be absent altogether from the image.



(a) Area Too Small (b) Area Too Large (c) Multiple Businesses

Figure 5.4: Common mistakes made by operators: a red box shows the area marked by an operator, a green box marks the area that should have been selected. (A) Only the signage is selected. (B) An area much larger than the business is selected. (C) Multiple businesses are selected as one business.

5.3 Ontology Based Generation of Training Data

Learning algorithms require training data. Deep Learning methods in particular are known for their need of large quantities of training instances, without which they overfit. In this section we describe a process for collecting a large scale training set, coupled with ontology-based labels.

To build a training set we need to create a matching between an extracted biz-patch p and a set of relevant category labels. As a first step, we match a biz-patch with a particular business instance from a database of previously known businesses $\mathcal{B}$ that was manually verified by operators. We use the textual information and geographical location of the image to match it to a business. We detect text areas in the image, and transcribe them using an OCR software. This process suffers from the drawbacks of extracting text, but is useful for creating a set of candidate matches. This provides us with a set $\mathcal{S}$ of text strings. The biz-patch is geolocated and we combine the location information with the textual data. For each known business $b \in \mathcal{B}$, we create the same description, by combining its location and the set $\mathcal{T}$ of all the textual information that is available for it; name, telephone number, operating hours, etc. We decide that p is a biz-patch of the business b if the physical distance between them is less than approximately one city block, and enough extracted text from $\mathcal{S}$ matches $\mathcal{T}$.

Using this technique we create a set of 3 million pairs (p, b). However, due to the factors that motivated our work, the quality and completeness of the information varies greatly between businesses. For many businesses we do not have category information. Moreover, the operators who created the database were inconsistent in the way they selected categories. For example, a McDonalds can be labeled as a HAMBURGER RESTAURANT, a FAST FOOD RESTAURANT, a TAKE AWAY RESTAURANT, etc. It is also plausible to label it simply as RESTAURANT. Labeling similar businesses with varying labels will confuse the learner.

We address this in two ways. First, by defining our task as a multi label problem we teach the classifier that many categories are plausible for a business. This, however, does not fully resolve the issue – When a label is missing from an example, the image is effectively used as a *negative* training instance for that label. It is important that training data uses a consistent set of labels for similar businesses. Here we use a key insight: the different labels used to describe a business represent different levels of specificity. For example, a hamburger restaurant *is a* restaurant. There is a containment relationship between these categories. Ontologies are a commonly used resource, holding hierarchical representations of such containment relations [Bhogal et al., 2007, Noy, 2004]. We use an ontology that describes containment relationships between entities with a geographical

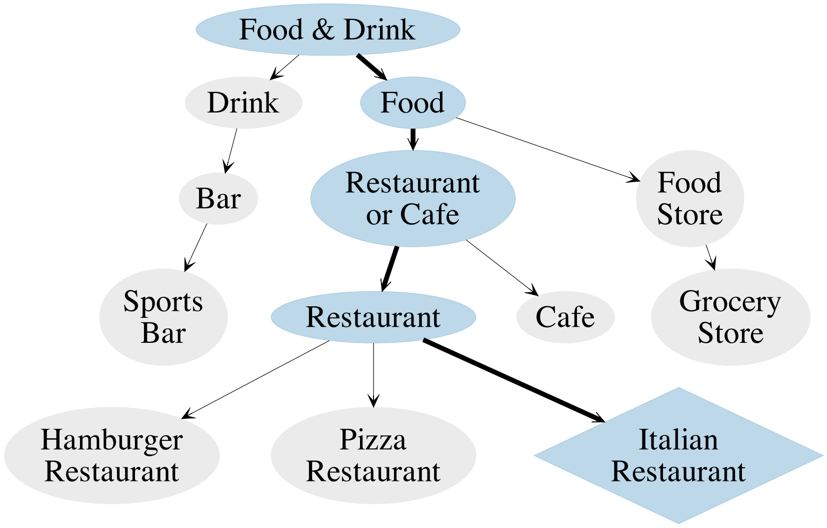


Figure 5.5: Using an ontology that describes relationships between geographical entities we assign labels at multiple granularities. Shown here is a snippet of the ontology. Starting from the ITALIAN RESTAURANT concept (diamond), we assign all the predecessors' categories as labels as well (shown in blue).

presence, such as RESTAURANT, PHARMACY, and GAS STATION. Our ontology, which is based on Google Map Maker's ontology, contains over 2,000 categories. For a pair (p, b) for which we know the category label c, we locate c in the ontology. We follow the containment relations described by the ontology, and add higher-level categories to the label set of p. The most general categories we consider are: ENTERTAINMENT & RECREATION, HEALTH & BEAUTY, LODGING, NIGHTLIFE, PROFESSIONAL SERVICES, FOOD & DRINK, SHOPPING. Figure 5.5 shows an illustration of this process on a snippet from the ontology. Starting from an ITALIAN RESTAURANT, we follow containment relations up predecessors in the ontology, until FOOD & DRINK is reached.

This creates a large set of pairs (p, s) where p is a biz-patch image and s is a matching set of labels with varying levels of granularity. To ensure there is enough training data per label we omit labels whose frequency in the dataset is very low and are left with 1.3 million biz-patches and 208 unique labels.

5.4 Model Architecture and Training

We base our model architecture on the winning submission for the ILSVRC 2014 classification and detection challenges by Szegedy et al. named GoogLeNet [Szegedy et al., 2014]. The model expands on the Network-in-Network idea of Lin et al. [Lin et al., 2013] while incorporating ideas from the theoretical work of Arora et al. [Arora et al., 2013]. Szegedy et al. forgo the use of fully connected layers at the top of the network and, by forcing the network to go through dimensionality reduction in middle layers, they are able to design a model that is much deeper than previous methods, while dramatically reducing the number of learned parameters. We employ the DistBelief [Dean et al., 2012] implementation of deep neural networks to train the model in a distributed fashion.

We create a train/test split for our data such that 1.2 million images are used for training the network and the remaining 100,000 images are used for testing. As a business can be imaged multiple times from different angles, the splitting is location aware. We utilize the fact that Street View panoramas are geotagged. We cover the globe with two types of tiles. Big tiles with an area of 18 kilometers, and smaller ones with area of 2 kilometers. The tiling alternates between the two types of tiles, with a boundary area of 100 meters between adjacent tiles. Panoramas that fall inside a big tile are assigned to the training set, and those that are located in the smaller tiles are assigned to the test set. This ensures that businesses in the test set were never observed in the training set while making sure that training and test sets were sampled from the same regions. This splitting procedure is fast and stable over time. When new data is available and a new split is made, train/test contamination is not an issue as the geographical locations are fixed. This allows for incremental improvements of the system over time.

We first pretrain the network using images and ground truth labels from the ImageNet large scale visual recognition challenge with a Soft Max top layer, and once the model has converged we replace the top layer, and continue the training process with our business image data. This pretraining procedure has been shown to be a powerful initialization for image classification tasks [Razavian et al., 2014, Chatfield et al., 2014]. Each image is resized to 256×256 pixels. During training random crops of size 220×220 are given to the model as training images. We normalize the intensity of the images, add random photometric changes and create mirrored versions of the images to increase the amount of training data and guide the model to generalize. During testing a central box of size 220×220 pixels is used as input to the model. We set the network to have a dropout rate of 70% (each neuron has a 70% chance of not being used) during training, and use a Logistic Regression top layer. Each image is associated with all the labels found by the method described in Section 5.3. This setup is designed to push the network to share features between classes that are on the same path up the ontology.

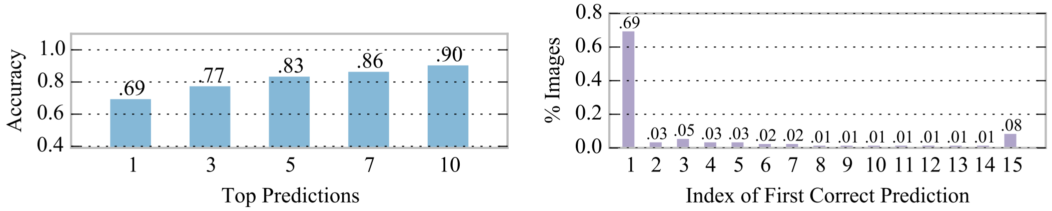
5.5 Evaluation

In this section we describe our experimental results. We begin by providing a quantitative analysis of the system's performance, then describe two large scale human performance studies that show our system is competitive with the accuracy of human operators and conclude with quantitative results that provide understanding as to what features the system managed to learn.

When building a business listing it is important to have very high accuracy. If a listing contains wrong information it will frustrate its users. The requirements on coverage however can be less strict. If the category for some business images can not be identified, the decision can be postponed to a later date; each street address may have been imaged many times, and it is possible that the category could be determined from a different image of the business. Similarly to the work of Goodfellow et al. [Goodfellow et al., 2013a] on street number transcription, we propose to evaluate this task based on recall at certain levels of accuracy rather than evaluating the accuracy over all predictions. For automatically building listings we are mainly concerned with recall at 90% precision or higher. This allows us to build the listing incrementally, as more data becomes available, while keeping the overall accuracy of the listing high.

5.5.1 Fine Grained Classification

As described in section 5.3 each image is associated with one or more labels. We first evaluate the classifier's ability to retrieve at least one of those labels. For an image i, we define the ground truth label set g_i. The predictions p_i are sorted by the classifier's confidence, and we define the top-k prediction set p_i^k as the first k elements in the sorted prediction list. A prediction for image i is considered correct if $g_i \cap p_i^k \neq \emptyset$. Figure 5.6a shows the prediction accuracy as a function of labels predicted. The accuracy at top-k is shown for $k \in \{1, 3, 5, 7, 10\}$. Top-1 performance is comparable to human annotators (see Section 5.5.2), and when the top 5 labels are used the accuracy increases to 83%. Figure 5.6b shows the distribution of first-correct-prediction, i.e. how far down the sorted list of predictions does one need to search before finding the first label that appears in g_i. We see that the first predicted label is by far the most likely and that the probability of having a predicted set p_i^k that does not contain any of the true labels decreases with k. In order to save space we sum up all the probabilities for $k \in [15, 208]$ in one bin.



(a) Accuracy at K

(b) First Correct Prediction

Figure 5.6: (a) Accuracy of classification for top K predictions. Using the top-1 prediction our system is comparable to human operators (see Table 5.1). When using the top 5 predictions the accuracy increases to 83%. (b) Percentage of images for which the first correct prediction was at rank K. To save space the values for $K \geq 15$ are summed and displayed at the 15th bin.

When building a business listing it is important to have very high accuracy. If a listing contains wrong information it will frustrate its users. The requirements on coverage however can be less strict. If the category for some business images can not be identified, the decision can be postponed to a later date; each street address may have been imaged many times, and it is possible that the category could be determined from a different image of the business. Similarly to the work of Goodfellow et al. [Goodfellow et al., 2013a] on street number transcription, we propose to evaluate this task based on recall at certain levels of accuracy rather than evaluating the accuracy over all predictions. For automatically building listings we are mainly concerned with recall at 90% precision or higher. This allows us to build the listing incrementally, as more data becomes available, while keeping the overall accuracy of the listing high.

Figure 5.7 shows precision recall curves for some of the top performing categories and summary curve of the full system (dashed). The precision and recall of the full system is calculated by using the top-1 prediction. For many categories we are able to recover the majority of businesses while precision is held at 90%. For a classification system to be useful in a practical setting the classifier's returned confidence must be well correlated with the quality of its prediction. Figure 5.8 shows a histogram of the number of correctly predicted labels in the top 5 predictions on a set of images whose labels were manually verified. The mean prediction confidence is indicated by color intensity (darker means higher confidence). Note the strong correlation between confidence and accuracy; for confidence above 80% normally at least 4 of the top labels are correct.

Figure 5.10 shows 30 sample images from the test set with their top 5 predictions. The model is able to classify these images with their high level categories (e.g. shopping) and

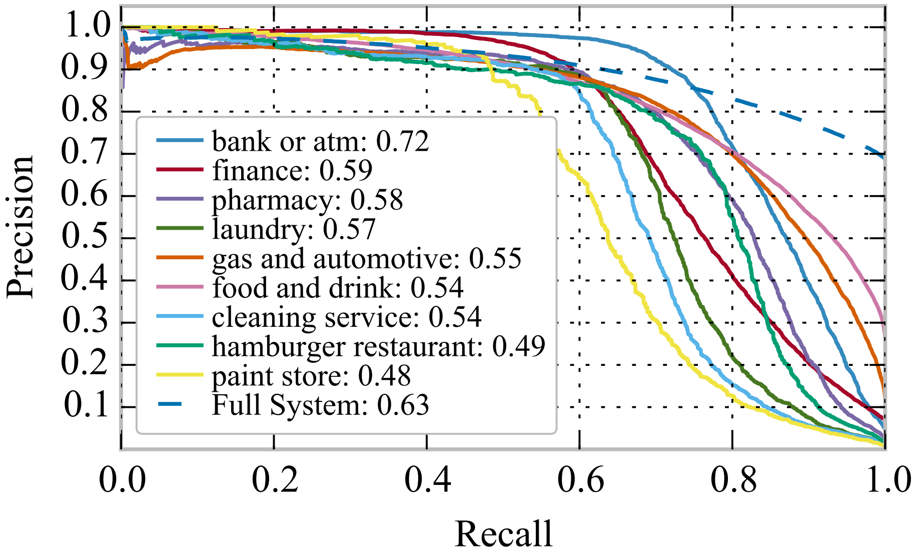
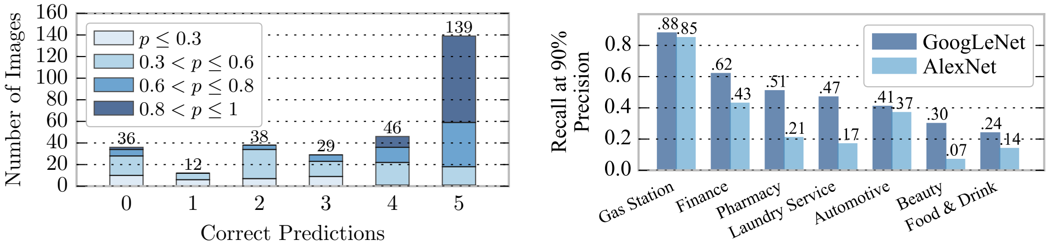


Figure 5.7: Precision recall curves for some of the top performing categories. The precision curve of the full system is shown as a dashed line. Recall at 90% precision is shown in the legend.



(a) Confidence-Accuracy correlation

(b) AlexNet/GoogLeNet Comparison

Figure 5.8: (a) Histogram of correct labels in the top 5 predictions for a set of 300 manually verified images. Color indicates mean prediction confidence. Note that the confidence in prediction is strongly correlated with the accuracy. (b) Recall at 90% precision for the network architecture used by our method and the AlexNet architecture [Krizhevsky et al., 2012] on a 13 category, classification task. We show categories for which at least one of the methods has recall ≥ 0.1. The GoogLeNet [Szegedy et al., 2014] network performs better on all categories. On some, such as PHARMACY it more than doubles the recall.

with their fine grained, specialized classes. For example, the top-right image was classified by the model as: BEAUTY, BEAUTY SALON, COSMETICS, HEALTH SALON, and NAIL

SALON.

5.5.2 Human Performance Studies

Our system needs to perform at human level accuracy. Otherwise it will require a human verification post process. To estimate human accuracy on this task we have conducted two large scale human performance studies that check the agreement of operator-provided labels for the same business. In our experiments, subjects were shown images of businesses and selected one of 13 options (12 categories, and an OTHER category that stands for “none of the above”.) Note that the studies used full resolution images as opposed to the 256×256 images given to the algorithm. The first study had $73,272$ images, each was shown to two operators. The operators agreed on 69% of image labels. In our second study we had $20,000$ images, but each image was shown to *three* to *four* subjects. We found that the average agreement of the human operators was 78%. Table 5.1 shows a detailed summary of the human study results.

5.5.3 Analysis: Learning to Read With Weak Labels

For some of the images in Figure 5.10, such as the image of the dental center (top row, second image from right) it is surprising that the model was capable of classifying it correctly. It is hard to think of a “canonical” dental center look, but even if we could, it doesn’t seem likely that this image would be it. In fact, without reading the text, it seems impossible to correctly classify it. This suggests that the system has learned to use text when needed. Figure 5.9 shows images from Figure 5.10 for which we have manually blurred the discriminative text. Note especially the image of the dental center, and of the auto dealer. After blurring the test “Dental” the system is confused about the dental center; it believes it is a beauty salon of some sorts. However, for the auto dealer, it is still confident that this is a place that sells things, and is related to transportation and automotive. To take this experiment to its logical extreme, we also show a synthetic image, which contains only the word *Pharmacy*. The classifier predicts the relevant labels for it.

To us this is a compelling demonstration of the power of Deep Convolutional Networks to learn the correct representation for a task. Similar to a human in a country in which she does not know the language, it has done the best it can – learn that some words are correlated with specific types of businesses. Note that it was never provided with annotated text or language models. It was only provided with what we would consider as *weak textual labels*, images that contain text and labeled with category labels. Furthermore,

Operator Agreement	Number of images *Study 1*	 *Study 2*
100%	50,425	9,938
75%	-	9
66%	-	8,535
50%	-	133
0%	22,847	1,300
Average Agreement	**69%**	**78%**

Table 5.1: Human Performance studies. In two large scale human studies we have found that manual labelers agree on a label for 69% and 78% of the images. Our system's accuracy is comparable with these results (see Figure 5.6).

when text is not available the system is able to make an accurate prediction if there is distinctive visual information.

5.6 Discussion

Business category classification, is an important part of location aware search. In this chapter we have proposed a method for fine grained, multi label, classification of business storefronts from street level imagery. We show that our system learned to extract and associate text patterns in multiple languages to specific business categories without access to explicit text transcriptions. Moreover, our system is robust to the absence of text, and when distinctive visual information is available, it is able to make correct predictions. We show our system achieves human level accuracy.

Using an ontology of entities with geographical attributes, we propagate label information during training, and produce a large set of 1.3 million images for a fine grained, multi label task. The use of non visual information, such as an ontology, to "ground" image data to real world entities is an exciting research direction, and there is much that can be done. For example, propagating information using the ontology at test time can increase both accuracy and recall. Node similarity in the ontology can be used to guide feature sharing between classes, and improve performance for seldom viewed classes.

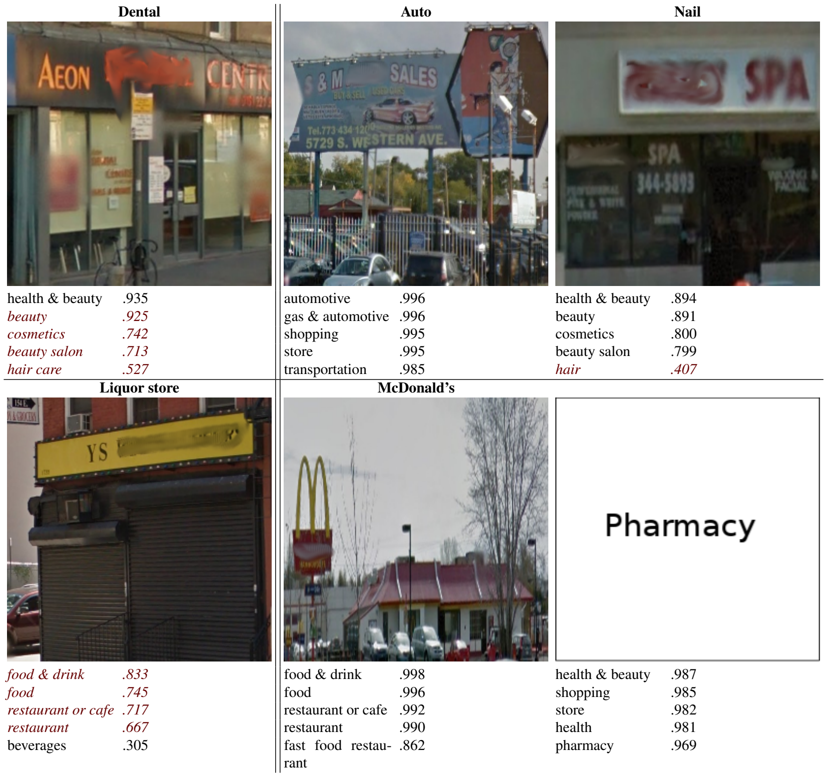
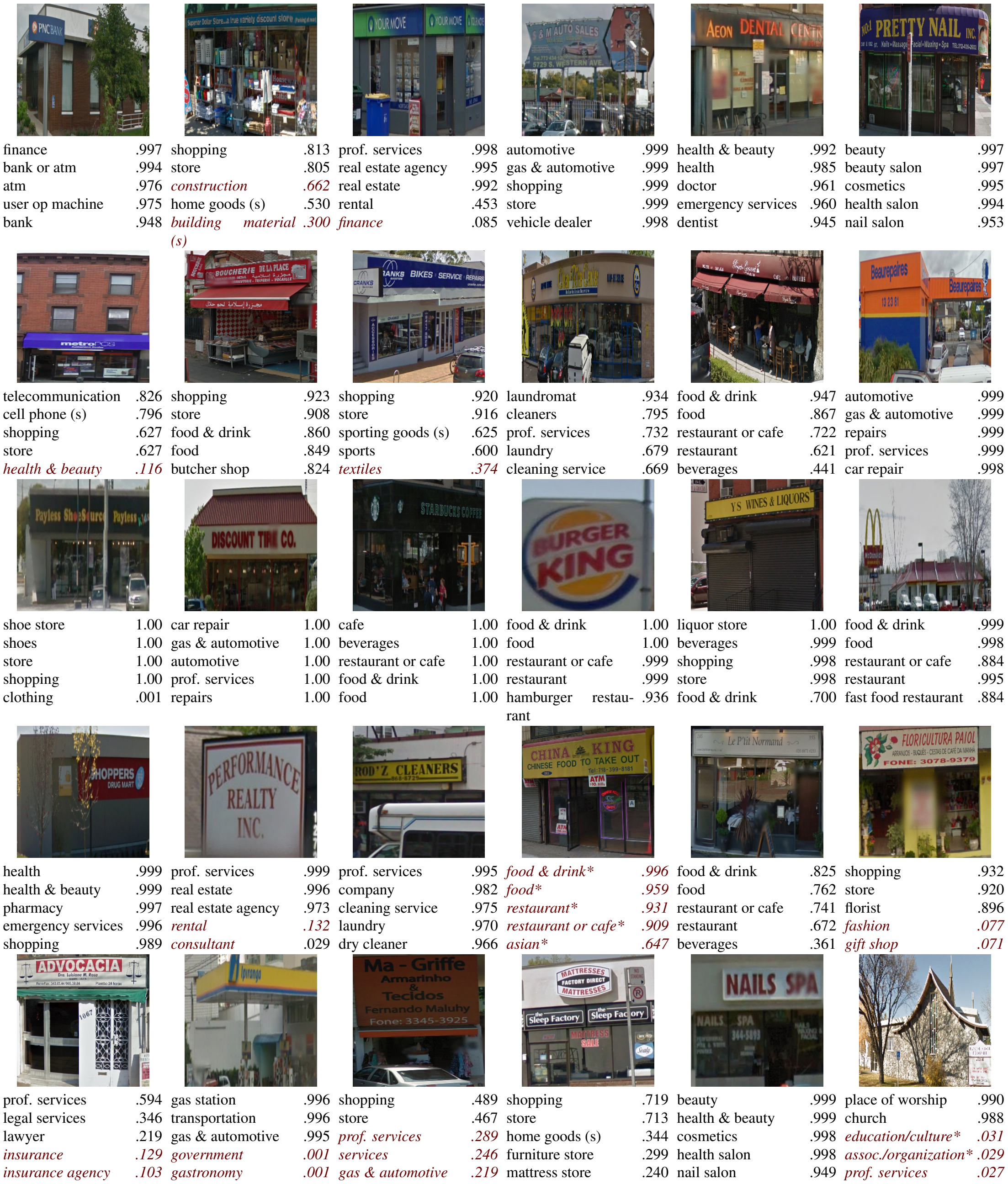


Figure 5.9: A number of images from Figure 5.10 with the discriminative text in the image blurred (noted above the image). For some images, without seeing the discriminative word the algorithm is confused (left column). For example, for the dental center, without the word *dental* it believes this is a beauty salon. For other images, there is enough non textual information for the algorithm to still be confident of the business category even when the text is blurred, for example the car dealership. Note the image of the nail spa: when the word *nail* is blurred the classifier falls back to more generic classes that fit the visual information - beauty salon, cosmetics etc. As a final indicator to the ability of the network to learn textual cues we show an image where the only visual information is the word *pharmacy*, which we created synthetically. The network predicts relevant labels.



finance	.997	shopping	.813	prof. services	.998	automotive	.999	health & beauty	.992	beauty	.997
bank or atm	.994	store	.805	real estate agency	.995	gas & automotive	.999	health	.985	beauty salon	.997
atm	.976	_construction_	_.662_	real estate	.992	shopping	.999	doctor	.961	cosmetics	.995
user op machine	.975	home goods (s)	.530	rental	.453	store	.999	emergency services	.960	health salon	.994
bank	.948	_building material (s)_	_.300_	_finance_	_.085_	vehicle dealer	.998	dentist	.945	nail salon	.953
telecommunication	.826	shopping	.923	shopping	.920	laundromat	.934	food & drink	.947	automotive	.999
cell phone (s)	.796	store	.908	store	.916	cleaners	.795	food	.867	gas & automotive	.999
shopping	.627	food & drink	.860	sporting goods (s)	.625	prof. services	.732	restaurant or cafe	.722	repairs	.999
store	.627	food	.849	sports	.600	laundry	.679	restaurant	.621	prof. services	.999
health & beauty	_.116_	butcher shop	.824	_textiles_	_.374_	cleaning service	.669	beverages	.441	car repair	.998
shoe store	1.00	car repair	1.00	cafe	1.00	food & drink	1.00	liquor store	1.00	food & drink	.999
shoes	1.00	gas & automotive	1.00	beverages	1.00	food	1.00	beverages	.999	food	.998
store	1.00	automotive	1.00	restaurant or cafe	1.00	restaurant or cafe	.999	shopping	.998	restaurant or cafe	.884
shopping	1.00	prof. services	1.00	food & drink	1.00	restaurant	.999	store	.998	restaurant	.995
clothing	.001	repairs	1.00	food	1.00	hamburger restaurant	.936	food & drink	.700	fast food restaurant	.884
health	.999	prof. services	.999	prof. services	.995	_food & drink*_	_.996_	food & drink	.825	shopping	.932
health & beauty	.999	real estate	.996	company	.982	_food*_	_.959_	food	.762	store	.920
pharmacy	.997	real estate agency	.973	cleaning service	.975	_restaurant*_	_.931_	restaurant or cafe	.741	florist	.896
emergency services	.996	_rental_	_.132_	laundry	.970	_restaurant or cafe*_	_.909_	restaurant	.672	_fashion_	_.077_
shopping	.989	_consultant_	_.029_	dry cleaner	.966	_asian*_	_.647_	beverages	.361	_gift shop_	_.071_
prof. services	.594	gas station	.996	shopping	.489	shopping	.719	beauty	.999	place of worship	.990
legal services	.346	transportation	.996	store	.467	store	.713	health & beauty	.999	church	.988
lawyer	.219	gas & automotive	.995	_prof. services_	_.289_	home goods (s)	.344	cosmetics	.998	_education/culture*_	_.031_
insurance	_.129_	_government_	_.001_	_services_	_.246_	furniture store	.299	health salon	.998	_assoc./organization*_	_.029_
insurance agency	_.103_	_gastronomy_	_.001_	_gas & automotive_	_.219_	mattress store	.240	nail salon	.949	_prof. services_	_.027_

Figure 5.10: Top predictions for 30 sample images. Predictions marked in _red_ disagree with ground truth labels; in some cases the classifier is correct and the ground truth label is wrong (marked with *). For example, see asian restaurant (fourth image from left, row before last), for which the top 5 predictions are all correct, but do not agree with the ground truth. The mark (s) is an abbreviation for store.

Chapter 6

Conclusions

This thesis make the following observation: High performing computer vision systems depend on high quality, large scale, *labeled* datasets. In most cases, obtaining data is relatively easy, but assigning accurate labels to it is both hard and expensive. In some cases, acquiring the data can be hard on its own.

Based on this we advocate for methods that insert a degree of automation into the labeling process. We believe without automation it will become prohibitively expensive to build larger datasets. We explore this concept through two computer vision tasks: viewpoint estimation, and fine-grained classification.

In our viewpoint estimation work, we focus on the task of estimating the pose of vehicles in consumer images. We utilize the abundance of high quality 3D CAD models that are publicly available to render synthetic images of cars from any desired viewpoint, and train models that accurately predict object pose. "Can models trained on synthetic data generalize to real images?" This question has been debated for a long time in our community. Based on our work we firmly believe that the answer to that questions is yes! We have reached a time in which rendering engines are powerful enough, and the amount of 3D models is large enough, to generate detailed images. In our work (Chapter 4) we show that the drop in performance between models trained on a real image dataset, and on a synthetic one is similar to the domain adaptation drop in performance that is expected when switching between two natural image datasets. This insight is not limited to viewpoint estimation, many other tasks in computer vision can benefit from rendered synthetic data: depth estimation, stereo matching, human body pose estimation, face key-point tracking, and more.

In Chapter 5 we outline a method to automatically assigning training images of business storefronts to multiple fine-grained labels, through the use of text extracted from the image, and its geo-tagged location. We match images to previously built business listing, and propagate labels using an ontology that defines business categories. This allows us to bootstrap from a known database of businesses, and create new business listings for many countries and languages. In our experiments we show that a method based purely on human annotators does not scale and the resulting annotations are of lower quality than our automatic method. When the amount of labeled data available to our deep learner reached a million images, we saw a significant increase in accuracy. Manually labeling such a large number of images is beyond the budget of most projects, both in industry and in academia. Furthermore, to manually annotate the complete dataset is equivalent to one worker year.

Following our experience we conclude that further research in automating the labeling of datasets can result in significant gains for the community, and has financial benefits to industry. In Chapter 7 we discuss promising next steps.

6.1 The Power of Labels

The prominent theme in this thesis is *Big Label* – without large amounts of labeled data it is hard to build accurate computer vision systems. Throughout our work we have experienced not only the importance of the quantity of labels, but that their quality is a critical property as well. Learning algorithms, deep ones as well as shallow, find it difficult to overcome the noise injected by erroneous labels. We experienced this most dramatically in the work described in chapter 5. Our first approach was based on human annotators to provide labels for building a large dataset. It failed, even when the set of labels was confined to just a handful. Operators, whether they were uncommitted to the task or just found it difficult, produced a lot of mistaken annotations and the deep learner had difficulties converging to an accurate state when trained with these labels. However, when the automatic labeling scheme was introduced, and label quality improved, we were able to train the system successfully. This observation has theoretical backing as well – Arora et al. [1993] show that learning half spaces in the presence of label noise is NP-Hard.

6.2 Linking Rendering and Optimization

Many machine learning algorithms only change their representation when they make a mistake. For example, consider that gradient information is only available to a deep learner if the loss is non zero. Finding hard examples to provide a model is therefore important in order to train highly accurate models that generalize well. Two well known examples of techniques for finding difficult examples are hard negative mining for SVM training [Malisiewicz et al., 2011], and “Fooling Images” for deep networks [Nguyen et al., 2014a, Goodfellow et al., 2014b]. In hard negative mining a large set of examples is searched for instances that are close to the margin. This focuses the efforts of the learner on the hardest areas to classify. Fooling images are created by changing the gradient of a deep learner such that the confidence in a wrong class increases. The result are images that are indistinguishable from the originals to the human eye, but are misclassified by the model. Adding such images to back into the training set, provides information to the learner about which modes of image variation are important for the classification.

Consider a deep learner trained only on rendered images. The bottleneck in training a model is usually access to enough data, but when rendered data is used the learner can potentially be exposed to an infinite amount of images. The challenge becomes how to generate the images that best help the model train. Deep models are usually trained in batches of examples. Can we generate on-line the next batch of training images such that we maximize the information the learner can extract from them? This will require a fast rendering technique.

This concept is related to the idea of the differentiable renderer introduced by Loper and Black [2014]. Renderers are designed for the forward process of image synthetics known as Graphics. *Inverse graphics* is the problem of inferring 3D shape, illumination, material, etc from input images. The differentiable renderer explicitly models the relationship between rendering parameters and changes in the output image. Using gradient descent one can optimize over them to predict the underlying properties of a test image by finding the rendering parameters that minimize the difference between the generated render and the test image.

Chapter 7

Future Directions

7.1 Fully Realistic Rendered Scenes

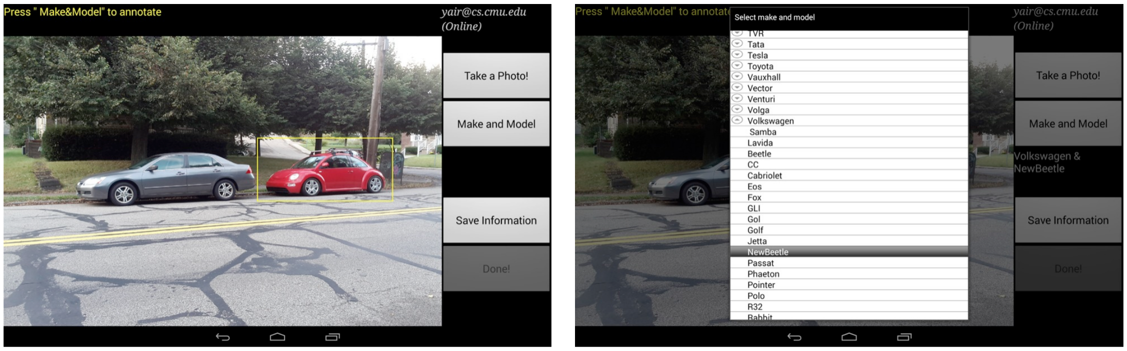
We have shown that rendered data can be successfully used for training a number of learning models. While the rendered objects are detailed and realistic, some questions about the image backdrop remain open. How much can we gain from rendering fully realistic rendered scenes, in which the object and the background are both rendered, as well as the interaction between them (reflections, shadows, occlusion etc)? Can a model be trained and validated solely on this type of synthetic data, and later perform well at deployment?

Do we need to fully and independently render each new frame? This is computationally expensive. One possible solution might be to fully render only a fraction of the full dataset, and use the simpler rendering process for the rest of the images. Then, match and transfer color and light information from fully rendered scenes to the simpler ones.

7.2 Moving Annotations Close To Capture Time

In the usual dataset building workflow, a set of images is collected in some way and in a later time the images are shown to annotators. The annotators' task is to analyze the image in some way, and provide some sort of tagging/labeling to it. The process of image capture and image annotation is therefore disjoint. In some cases combining these into a single process can improve the quality of the provided labels. We first observed this when we were looking to create a dataset of fine-grained car labels.

Labeling cars with their *Make*, *Model*, and *Year* is a challenging task for most people.



(a) (b)

Figure 7.1: Screen shots from the CarSpotting annotation android application. Users can photograph cars in their surroundings, and annotate them with their make and model. Using a small team of volunteers we have collected a detailed dataset of about 3,000 images.

While car experts or enthusiasts are good at this task, most of us can only recognize a small number of models. The standard approach for annotating data in the computer vision community - collecting a large set of images, showing the images to human annotators and asking them to provide the labels - just doesn't work in this case. However, the person who captured the image is in a unique position to provide the label - they can either choose to snap a picture of a car they know, or can examine the physical car after taking the picture to verify its make and model.

In order to improve the process of fine-grained label generation for car model classification we have built a mobile application named CarSpotting (Figure 7.1) that runs on Android-powered devices. The app helps users capture images of cars in their surroundings and annotate them with fine grained labels. Using this application, and a small team of dedicated volunteers, we have collected a high quality labeled set of roughly 3,000 images. To scale up this collection process and to create a large scale dataset paid annotators can be used.

This approach can be used in more settings in which the annotation process requires expertise or knowledge that might be found closer to the image capturing event. One such case is early childhood autism evaluation [Rehg, 2011] in which an video of a child's social interaction with an adult is used to detect autism risk factors, and suggest early intervention methods. Training data is collected by capturing play sessions of toddlers with professional evaluators. The evaluators are more qualified to provide an analysis of the child's social skills than an annotator which only gets the footage after the fact.

7.3 Crowd Sourced Annotations

In this thesis we have focused on adding automation to the process of building large datasets. We have tried to minimize the reliance on human annotators. Human computer interaction research has produced a number of insights that can greatly speed up crowd based work. An important concept is that of micro-specialization. Instead of providing each worker with a long, often complicated, task one can break it into a set of smaller micro-tasks that in aggregate form the desired work to be done. Worker are assigned to one of the smaller micro-tasks and can quickly become skilled in performing them.

Micro-specialization has been shown to greatly increase crowd workers' speed in many tasks. However, the lack of the full context of the task can confuse workers and can cause them to make mistakes. In the Macro-context paradigm, workers are provided a broader understanding of the full task but are only in charge of performing their micro-task. Using this combination of specialization and broad context workers produce faster and more accurate results.

Humans are good at finding outliers with a quick glance. Machines are good at processing large amounts of data. Once a small number of images are annotated a simple learner can be trained, and applied over an entire dataset. Crowd workers can then be asked to quickly identify erroneous examples in sets of images. The correct images can then be used to train a more complex model. Such feedback iteration can be used to quickly annotate a large set of images.

In an ongoing collaboration with HCI researchers, we have been exploring these concepts in order to build a tool that will allow us to create human annotated datasets for viewpoint estimation and body pose prediction, that are an order of magnitude larger than what is currently available to the research community.

Bibliography

URL http://www.chaosgroup.com. 4.2

Dragomir Anguelov, Carole Dulong, Daniel Filip, Christian Frueh, Stphane Lafon, Richard Lyon, Abhijit Ogale, Luc Vincent, and Josh Weaver. Google street view: Capturing the world at street level. *Computer*, 2010. 2.3

Mica Arie-Nachimson and Ronen Basri. Constructing implicit 3d shape models for pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1341–1348. IEEE, 2009. 3.1, 3.5

Sanjeev Arora, Lszl Babai, Jacques Stern, and Z. Sweedy. The hardness of approximate optima in lattices, codes, and systems of linear equations. In *Foundations of Computer Science, 1993. Proceedings., 34th Annual Symposium on*. IEEE, 1993. 6.1

Sanjeev Arora, Aditya Bhaskara, Rong Ge, and Tengyu Ma. Provable bounds for learning some deep representations. *arXiv:1310.6343 [cs, stat]*, oct 2013. 5.4

Adela Barriuso and Antonio Torralba. Notes on image annotation. *arXiv:1210.3448 [cs]*, 2012. 1

Jagdev Bhogal, Andy Macfarlane, and Peter Smith. A review of ontology based query expansion. *Information processing & management*, 43(4):866–886, 2007. 5.3

Stanley Michael Bileschi. *StreetScenes: Towards scene understanding in still images*. PhD thesis, Massachusetts Institute of Technology, 2006. 3.5, 4.4

Volker Blanz, Bernhard Schölkopf, HCBVV Bülthoff, Chris Burges, Vladimir Vapnik, and Thomas Vetter. Comparison of view-based object recognition algorithms using realistic 3d models. In *Artificial Neural Networks-ICANN*. Springer, 1996. 2.2

Vishnu Naresh Boddeti. *Advances in correlation filters: vector features, structured prediction and shape alignment*. PhD thesis, Carnegie Mellon University, 2012. 3.3.1

Vishnu Naresh Boddeti, Takeo Kanade, and BVK Kumar. Correlation filters for object alignment. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2013. 3.1, 3.2, 3.2.1, 3.5, 4.4

Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. Visual recognition with humans in the loop. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 2010. 2.3

Richard J Campbell and Patrick J Flynn. A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding*, 2001. 2.2

Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *Proceedings of the British Machine Vision Conference (BMVC)*, 2014. 5.4

Tao Chen, Zhe Zhu, Ariel Shamir, Shi-Min Hu, and Daniel Cohen-Or. 3-sweep: Extracting editable objects from a single photo. *ACM Transactions on Graphics*, 2013. 3.1

Nico Cornelis, Bastian Leibe, Kurt Cornelis, and Luc Van Gool. 3d urban scene modeling integrating recognition and reconstruction. *International Journal of Computer Vision*, 2008. 2.3

Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2005. 1, 3.2

Philip David, Daniel Dementhon, Ramani Duraiswami, and Hanan Samet. Softposit: Simultaneous pose and correspondence determination. *International Journal of Computer Vision*, 2004. 2.2

Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marcaurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Quoc V. Le, and Andrew Y. Ng. Large scale distributed deep networks. In *Advances in neural information processing systems (NIPS)*, pages 1223–1231. 2012. 2, 2.3, 5.4

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 4.1

Jia Deng, Jonathan Krause, and Li Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2013. 2.1

Yi Deng, Qiong Yang, Xueyin Lin, and Xiaoou Tang. A symmetric patch-based correspondence model for occlusion handling. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2005. 2.2

Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. *arXiv preprint arXiv:1411.5928*, 2014. 2.2

Ahmed Elgammal and Chan-Su Lee. Homeomorphic manifold analysis (hma): Generalized separation of style and content on manifolds. *Image and Vision Computing*, 2013. 3.1

Mark Everingham, Luc Van Gool, Chris Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge 2011 (voc2011) results, 2011. 2.1, 2.3, 4.1

Kayvon Fatahalian. *Enolving the Real-Time Graphics Pipeline for Micropolygon Rendering*. PhD thesis, Stanford University, 2011. 2.1

Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2010. 2.2, 2.3

Sanja Fidler, Sven Dickinson, and Raquel Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *Advances in neural information processing systems (NIPS)*, 2012. 2.2

Arturo Flores and Serge Belongie. Removing pedestrians from google street view images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010. 2.3

Winrich A. Freiwald and Doris Y. Tsao. Functional compartmentalization and viewpoint generalization within the macaque face-processing system. *Science*, 2010. 2.2

Yun Fu and Thomas S. Huang. Human age estimation with regression on discriminative aging manifold. *Multimedia, IEEE Transactions on*, 2008. 1

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological cybernetics*, 36 (4), 1980. 2.3

Daniel Glasner, Meirav Galun, Sharon Alpert, Ronen Basri, and Gregory Shakhnarovich. Viewpoint-aware object detection and continuous pose estimation. *Image and Vision Computing*, 2012. 2.2, 3.1, 3.0(a), 3.0(b), 3.5, 3.5

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2014a. 2.2

Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv preprint arXiv:1312.6082*, 2013a. 1, 2, 2.3, 5.5, 5.5.1

Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. *arXiv preprint arXiv:1302.4389*, 2013b. 2.3

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. *arXiv:1412.6572 [cs, stat]*, 2014b. 6.2

Kurtis Heimerl, Brian Gawalt, Kuang Chen, Tapan Parikh, and Bjrn Hartmann. CommunitySourcing: Engaging local crowds to perform expert work via physical kiosks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 2012. 2.1

Mohsen Hejrati and Deva Ramanan. Analyzing 3d objects in cluttered images. In *Advances in neural information processing systems (NIPS)*, 2012. 2.2

Mark Henne, Hal Hickel, Ewan Johnson, and Sonoko Konishi. The making of toy story [computer animation]. In *Compcon'96.'Technologies for the Information Superhighway'Digest of Papers*. IEEE, 1996. 2.1

Aharon Bar Hillel and Daphna Weinshall. Subordinate class recognition using relational object models. *Advances in neural information processing systems (NIPS)*, 2007. 2.3

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 2.3

Berthold K. Horn and Brian G. Schunck. Determining optical flow. In *1981 Technical Symposium East*. International Society for Optics and Photonics, 1981. 1

John. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3), 2007. 4.5

Kevin Jarrett, Koray Kavukcuoglu, M. Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009. 2.3

Jeremy Karnowski. Alexnet + svm, 2015. URL https://jblkacademic.files.wordpress.com/2015/07/alexnet2.png. 4.7, 7.3

Kevin Karsch, Ce Liu, and Sing Bing Kang. Depth extraction from video using non-parametric sampling. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012. 3.1

Natasha Kholgade, Tomas Simon, Alexei Efros, and Yaser Sheikh. 3d object manipulation in a single photograph using stock 3d models. *ACM Transactions on Graphics (TOG)*, 2014. 1.3

Jonathan Krause, Jia Deng, Michael Stark, and Li Fei-Fei. Collecting a large-scale dataset of fine-grained cars. *Second Workshop on Fine-Grained Visual Categorization*, 2013. 1

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *nips*, 2012. 2.2, 2.3, 4.3, 5.8, 7.3

BVK Vijaya Kumar, Abhijit Mahalanobis, and Richard D Juday. *Correlation pattern recognition*. Cambridge Univ. Press, 2005. 3.1, 1

Walter S. Lasecki, Young Chol Song, Henry Kautz, and Jeffrey P. Bigham. Real-time crowd labeling for deployable activity recognition. In *Proceedings of the 2013 conference on Computer supported cooperative work*. ACM, 2013. 1

Walter S. Lasecki, Leon Weingard, George Ferguson, and Jeffrey P. Bigham. Finding dependencies between actions using the crowd. In *Proceedings of the 32nd annual ACM conference on Human factors in computing systems*. ACM, 2014. 1

Edith Law, Burr Settles, Aaron Snook, Harshit Surana, Luis Von Ahn, and Tom Mitchell. Human computation for attribute and attribute value acquisition. In *Proceedings of the First Workshop on Fine-Grained Visual Categorization (FGVC)*, 2011. 2.1

Yann LeCun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11), 1998. 2.3

Yong Jae Lee, Alexei A. Efros, and Martial Hebert. Style-aware mid-level representation for discovering visual connections in space and time. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013. 2.3

Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o(n) solution to the pnp problem. *International Journal Computer Vision*, 2009. 2.2, 4.1

Li-Jia Li, Hao Su, Li Fei-Fei, and Eric P Xing. Object bank: A high-level image representation for scene classification and semantic feature sparsification. In *Advances in neural information processing systems (NIPS)*, 2010. 3.1

Joerg Liebelt and Cordelia Schmid. Multi-view object class detection with a 3d geometric model. In *Computer Vision and Pattern Recognition*. IEEE, 2010. 2.2

Joerg Liebelt, Cordelia Schmid, and Klaus Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2008. 2.2, 3.1

Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2992–2999. IEEE, 2013. 1.2.2

Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2013. 5.4

Jiongxin Liu, Angjoo Kanazawa, David Jacobs, and Peter Belhumeur. Dog breed classification using part localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 2012. 1

Matthew M. Loper and Michael J. Black. Opendr: An approximate differentiable renderer. In *Proceedings of the European Conference on Computer Vision (ECCV)*. 2014. 6.2

David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJVC)*, 2004. 1

Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981. 1

Tomasz Malisiewicz, Abhinav Gupta, Alexei Efros, et al. Ensemble of exemplar-svms for object detection and beyond. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2011. 3.2, 6.2

Branislav Micusik and Jana Kosecka. Piecewise planar city 3d modeling from street view panoramic sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009. 2.3

Dana Movshovitz-Attias, Yair Movshovitz-Attias, Peter Steenkiste, and Christos Faloutsos. Analysis of the reputation system and user contributions on a question answering website: StackOverflow. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2013. 2.1

Yair Movshovitz-Attias, Vishnu Naresh Boddeti, Zijun Wei, and Yaser Sheikh. 3d pose-by-detection of vehicles via discriminatively reduced ensembles of correlation filters. In *Proceedings of the British Machine Vision Conference (BMVC)*, Nottingham, UK, September 2014. 1, 1, 1.2.2, 1.4, 2.1, 3.5

Yair Movshovitz-Attias, Qian Yu, Martin Stumpe, Vinay Shet, Sacha Arnoud, and Liron Yatziv. Ontological supervision for fine grained classification of street view storefronts. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 1.4

Hiroshi Murase and Shree K Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision (IJCV)*, 1995. 2.2, 3.1

Sameer A Nene, Shree K Nayar, Hiroshi Murase, et al. Columbia object image library (coil-20). Technical report, Technical Report CUCS-005-96, 1996. 4.1

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *arXiv preprint arXiv:1412.1897*, 2014a. 6.2

Phuc Xuan Nguyen, Kai Wang, and Serge Belongie. Video text detection and recognition: Dataset and benchmark. In *Winter Conference on Applications of Computer Vision (WACV)*, Steamboat Springs, CO, March 2014b. 5.2

Natalya F Noy. Semantic integration: a survey of ontology-based approaches. *ACM Sigmod Record*, 33(4):65–70, 2004. 5.3

Aude Oliva and Antonio Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International journal of computer vision*, 42(3):145–175, 2001. 4.1

A.V. Oppenheim, A.S. Willsky, and S.H. Nawab. *Signals and systems*. Prentice-Hall Englewood Cliffs, NJ, 1983. 3.3.1

Mustafa Ozuysal, Vincent Lepetit, and Pascal Fua. Pose estimation for category specific multiview object localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2009. 2.2, 3.1, 3.5

HG Parsa, John T Self, David Njite, and Tiffany King. Why restaurants fail. *Cornell Hotel and Restaurant Administration Quarterly*, 46(3):304–322, 2005. 5.1

Bojan Pepik, Peter Gehler, Michael Stark, and Bernt Schiele. $3d^2pm - 3d$ deformable part models. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2012. 2.2

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, 2014. 5.4

James M. Rehg. Behavior Imaging: Using Computer Vision to Study Autism. In *MVA*, 2011. 7.2

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge, 2014. 1, 1, 2, 2.1, 2.3

Scott Satkin, Jason Lin, and Martial Hebert. Data-driven scene understanding from 3d models. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2012. 3.1

Silvio Savarese and Li Fei-Fei. 3d generic object categorization, localization and pose estimation. In *International Conference on Computer Vision*. IEEE, 2007. 2.2

Johannes Schels, Jörg Liebelt, Klaus Schertler, and Rainer Lienhart. Synthetically trained multi-view object class and viewpoint detection for advanced image retrieval. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*. ACM, 2011. 2.2

Jeffrey Mark Siskind. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition*, (1), 1996. 1

Hyun O Song, Trevor Darrell, and Ross B Girshick. Discriminatively activated sparselets. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013. 2.2

Hyun Oh Song, Stefan Zickler, Tim Althoff, Ross Girshick, Mario Fritz, Christopher Geyer, Pedro Felzenszwalb, and Trevor Darrell. Sparselet models for efficient multiclass object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2012. 2.2

Michael Stark, Michael Goesele, and Bernt Schiele. Back to the future: Learning shape models from 3d cad data. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2010. 2.1, 2.2, 3.1

Michael Stark, Jonathan Krause, Bojan Pepik, David Meger, James J Little, Bernt Schiele, and Daphne Koller. Fine-grained categorization for 3d scene understanding. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2012. 2.2, 3.1

Hao Su, Min Sun, Li Fei-Fei, and Silvio Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2009. 2.2

Hao Su, Charles R. Qi, Yangyan Li, and Leonidas Guibas. Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3d Model Views. *arXiv:1505.05641 [cs]*, 2015. 2.2

Baochen Sun and Kate Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2014. 1.2.2, 2.1

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv:1409.4842 [cs]*, sep 2014. 2, 2.3, 5.4, 5.8, 7.3

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 1

Antonio Torralba and Alexei Efros. Unbiased look at dataset bias. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2011. 4.1, 4.4

Shubham Tulsiani and Jitendra Malik. Viewpoints and Keypoints. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 2.2

Luc Vincent. Taking online maps down to street level. *Computer*, 2007. 2.3

Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 2004. 2.1

Luis Von Ahn, Ruoran Liu, and Manuel Blum. Peekaboom: a game for locating objects in images. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM, 2006. 2.1

Catherine Wah, Steve Branson, Pietro Perona, and Serge Belongie. Multiclass recognition and part localization with humans in the loop. In *IEEE International Conference on Computer Vision (ICCV)*, 2011. 2.3

Kai Wang and Serge Belongie. Word spotting in the wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Heraklion, Crete, Sept. 2010. URL http://vision.ucsd.edu/~kai/grocr/. 5.2

Ren Wu, Shengen Yan, Yi Shan, Qingqing Dang, and Gang Sun. Deep image: Scaling up image recognition. *arXiv preprint arXiv:1501.02876*, 2015. URL http://arxiv.org/abs/1501.02876. 4.2

Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *Winter Conference on Applications of Computer Vision (WACV)*, 2014. 4.4

Jianxiong Xiao and Long Quan. Multiple view semantic segmentation for street view images. In *Computer Vision, 2009 IEEE 12th International Conference on*, 2009. 2.3

Bangpeng Yao, Aditya Khosla, and Li Fei-Fei. Combining randomization and discrimination for fine-grained image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 2.3

Bangpeng Yao, Gray Bradski, and Li Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 2.3

Erdem Yörük and Rene Vidal. Efficient object localization and pose estimation with 3d wireframe models. In *4th IEEE Workshop on 3D Representation and Recognition (ICCV)*. IEEE, 2013. 2.2, 3.1

Amir Roshan Zamir and Mubarak Shah. Accurate image localization based on google maps street view. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2010. 2.3

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional neural networks. *arXiv preprint arXiv:1311.2901*, 2013. 2.3

Haopeng Zhang, Tarek El-Gaaly, Ahmed Elgammal, and Zhiguo Jiang. Joint object and pose recognition using homeomorphic manifold analysis. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013. 2.2, 3.1

Zhenyao Zhu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Multi-view perceptron: a deep model for learning face identity and view representations. In *Advances in Neural Information Processing Systems*, 2014. 2.2

M Zeeshan Zia, Michael Stark, Bernt Schiele, and Kaspar Schindler. Detailed 3d representations for object recognition and modeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 2013. 2.2

List of Figures

List of Tables