TOWARDS ROBUST VISUAL PERCEPTION SYSTEMS IN REAL-WORLD ENVIRONMENTS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Hubert Lin

May 2022

TOWARDS ROBUST VISUAL PERCEPTION SYSTEMS IN REAL-WORLD ENVIRONMENTS

Hubert Lin, Ph.D.

Cornell University 2022

Computer vision models are conventionally trained on, and benchmarked against, curated datasets. While very high performance can be achieved in these settings, it is more challenging to deploy these models in the real world. When computer vision models are used in new environments, they must overcome distribution shift between their training data and their test environment. We consider computer vision models to be *robust* when they can perform well on a variety of images captured from different environments. This dissertation explores several research problems towards building perception systems that are robust in the real world.

Building and labeling datasets is a key step for training a strong computer vision system. Capturing and representing a large diverse set of images allows our models to have experience with images from a variety of environments. However, data annotation is difficult, so efficiently labeling data is an important challenge to consider. It is also useful to consider the human visual system as a gold-standard point of reference for a robust visual system, as humans can readily understand new images. Paintings are an interesting type of image which are created by humans for human consumption – a key property of artwork lies in its ability to convey perceptual realism without necessarily being physically realistic. We would like our computer vision models to be able to understand paintings as well, and learning from paintings may allow computer vision models to be

more robust. Lastly, even our best efforts to build robust models will not lead to a perfect model, and it is important to reason about failures and uncertainties when these models are used. We touch upon each of these challenges in the dissertation.

First, a direct way to overcome distribution shift is to as closely mimic the real world distribution of `(image, label)` pairs as possible in the datasets we use. However, labeling data is very expensive and time-consuming. In the first work, we propose a new efficient annotation method for semantic segmentation. Our pipeline divides the traditional per-pixel annotation task for an entire image into per-pixel annotation for image subregions. This task is more palatable for annotators and leads to an increase in label quality for a lower cost. Furthermore, we find that only annotating a small number of image subregions is sufficiently informative for models to outperform conventional annotation and inexpensive weakly-supervised annotation methods.

Next, we explore paintings as a medium for studying perception systems, and their utility as an alternative source of training data from natural images. The image distribution of paintings are different from the distribution of natural photographs, but paintings often depict meaningful objects and scenery that parallel those found in the real world. Common computer vision models are developed for natural photographs, and most existing labeled datasets focus on natural images. We explore several use cases of such models applied to paintings instead. Furthermore, paintings may emphasize features or invariances that humans utilize for robust perception, as they can be perceptually realistic without being physically realistic. Indeed, for finegrained fabric classification, we find evidence that models trained on paintings focus on cues that are both more interpretable and generalizable.

The next work extends our previous findings by systematically exploring the invariances encoded in paintings for natural image recognition. We study how models behave when trained with real paintings and style transfer. Style transfer is a type of data augmentation that promises to create painting-like images from natural photographs. We train models on natural photographs, paintings, and/or stylized images, and evaluate them on test data representing real-world distribution shifts. Perception models must overcome such shifts to successfully understand different environments – for example, the test photographs may contain noise, or be drawn from another dataset which was sampled from different viewpoints than the training images. Our results show that learning from a combination of natural photographs and paintings leads to models that are far more robust than learning from natural photographs alone. Interestingly, we also find that style transfer does not capture the same invariances as paintings, and that paintings are unique among various artforms in enabling recognition models to learn useful invariances for natural image recognition.

Finally, we conclude with a real world case study in using visual perception for autonomous navigation. Even the most accurate perception model will not be perfect, and it is important to account for failures when using these models as a building block in an autonomous sytem. We propose a planning pipeline that reasons about uncertainty in semantic segmentations to find a safe path in unknown environments. Given predictions of terrain and obstacles from a view of a scene, the autonomous agent determines which subsequent views of the environment to capture. By taking multiple views of the environment, the agent increases its confidence in successfully selecting a viable path across safe terrain to a goal location. Our results show that this pipeline allows safe paths to be planned when deployed in real world environments with noisy and inaccurate

model predictions.

These works represent meaningful steps towards tackling some key challenges in building robust perception systems: acquiring high-quality and diverse training data, learning robust features for recognition, and reasoning about perception uncertainties within broader autonomous systems. However, the problem of building a robust perception system is multifaceted in its complexities and challenges, and many exciting research problems remain to be explored in future work.

BIOGRAPHICAL SKETCH

Hubert Lin was born in Illinois, USA in 1994, and grew up in Michigan, USA and Ontario, Canada. After completing an Honours Bachelor of Science in Physics and Computer Science at the University of Toronto in 2016, he has pursued a PhD in Computer Science at Cornell University.

This page intentionally left blank.

ACKNOWLEDGEMENTS

My research was supported in part by NSERC (PGS-D 516803 2018), NSF (CHS-1617861, CHS-1513967, CHS-1900783, CHS-1930755), and PERISCOPE MURI (N00014-17-1-2699).

I would like to thank my advisor Kavita Bala (KB) for her kind yet firm mentorship. The last several years have been an important period of growth for me, both as a researcher and as an individual. KB's guidance has been invaluable to me in both of these aspects. I would also like to thank my committee members Bharath Hariharan and David Bindel for their gentle feedback as I approached important milestones during the PhD. All of my research have been the result of successful collaborations with peers and colleagues, and I thank them for their collaborative spirit, insightful discussions, and many hours spent hammering away at problems. I would like to thank Mitchell van Zuijlen, Maarten Wijntjes, and Sylvia Pont for their friendly presence and expertise over the years, all while wrestling with the challenges of meeting across timezones and interdisciplinary research. I spent many nights working with Yutao Han, Jacopo Banfi, and Hadi AlZayer on a variety of interesting robotics problems, and it was always just a little bit harder than we expected to get the robots to finally run. Balazs Kovacs and Paul Upchurch played pivotal roles in mentoring me throughout my early projects, and I thank them for their sharing experience as senior students when I joined the group and throughout my time here. I would like to thank the members of the group for their friendly attitudes and diverse research interests which made for interesting group meeting discussions: Scott Wehrwein, Paul Upchurch, Balazs Kovacs, Fujun Luan, Utkarsh Mall, Hadi AlZayer, Aaron Gokaslan, and many others who were here for a shorter while. I also enjoyed the quiet respectful work environment built by the students in the broader

graphics/vision lab.

My friends and family have offered unconditional support over the years. They remind me to look at life beyond work, while also encouraging me to push towards my goals in research. I'm grateful for the friends that I have met here – thank you for all the shared meals, rock climbing sessions, board game nights, laughter, and moments of commiseration. I look forward to many more in the years to come. To my friends from home who have kept in close touch over many many years: you know who you are, and how important you are to me. Thank you for all the video calls, silly group messages, serious discussions, nights of gaming, and for reminding me of home even as we venture out on our own paths in different parts of the world. I would like to thank my partner, Nicole Wang, for her loving support, caring heart, and fun sense of humor. I am inspired by her resilience as a scientist and I cannot wait to see where she chooses to go as she approaches the completion of her PhD. I'd like to conclude by mentioning three important dogs in my life. First, my family's two miniature schnauzers, Noble and Jake, were with me through many milestones in my life and saw me off to Ithaca for my first day at Cornell. I wish they both could have been here to celebrate this step with me. Finally, Nicole's westie, Momo – a bundle of youthful energy who always gets into just enough trouble to keep us on our toes, and reminds me to enjoy the little things.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Humans perceive the world around them through many senses: touch, taste, sound, smell, and sight. Each of these senses serve to shape our understanding of the world, and for many of us, visual perception plays a uniquely important role in our lives. From a young age, we learn about the world from observing how people interact, how objects move, and how light bounces off different surfaces in a room. With experience, we learn to scan our environment for important visual cues that tell us where objects might move, which surfaces are safe to walk on, and what obstacles we need to be aware of. Through visual media, whether in the form of photographs, videos, artwork, or written text, we are able to communicate new ideas, find entertainment, and understand different cultures.

Can a machine meaningfully perceive the visual world? The field of computer vision aims to understand how we can build computer models that can understand visual information. A decade ago, deep learning unveiled its potential in computer vision – by "learning" with a series of interconnected artificial neurons, the seminal AlexNet model boasted a 37% reduction in error rate relative to the best performing method at the time (Table 2 of [78]) on a challenging image recognition task. At the time of writing this dissertation, the original AlexNet paper [78] has accumulated over 98000 citations (To the reader: how many citations does AlexNet have now?). Deep learning quickly took over as *the* paradigm for building high-performing computer vision systems. Today, these systems are widely used, playing significant roles in common consumer products such as cell-phones and social media, and in cutting edge developments in self-driving,

medical diagnostics, and drug discovery.

Training a deep neural network amounts to optimizing the hundreds of thousands to billions parameters that define its artificial neurons; this optimization requires a large amount of data to meaningfully fit the model. Historically, deep learning models have been developed on – and benchmarked against – curated datasets, allowing for controlled scientific comparisons between advancements in new deep learning techniques. On the now-famous ImageNet-1K dataset, computer vision models have long surpassed human performance [59] – a superhuman feat (literally!).

Of course, the story is not that simple. Early work revealed a weakness of deep models to seemingly innocuous patterns of imperceptible noise – noise so weak in magnitude that human observers could not identify the existence of these patterns in manipulated images. Yet, these patterns would cause deep networks to confidently misidentify the content of manipulated images [156], such as classifying an image of a "bus" as an "ostrich". This brittleness of deep learning to such adversarial examples gives evidence that the superhuman nature of deep networks hinges on carefully controlled laboratory conditions provided by benchmark datasets. Even without the presence adversarial manipulations designed to break deep learning models, computer vision systems can still struggle to perform well across data drawn different sources. Scientists have studied the ability of models to generalize to different visual domains [168, 193] – can a model trained on photographs learn to recognize objects in paintings?; or, can a model that has only seen images of one city recognize attributes of scenes in another city? In many cases, deep computer vision models are unable to transfer its performance between these different settings.

These failures arise from *distribution shift* between the training data and test environment [136, 182]. Many characterizations of distribution shift have been studied, with a popular assumption being *covariate shift* in which the distribution of image features are different between training and testing. Suppose a classifier is trained to distinguish cats from dogs. However, imagine all of the training images of cats are of orange cats and all of the training images of dogs are of black dogs. The classifier may place a very high weight on the color of the animal to minimize its loss during training, but this will cause it to perform quite poorly in reality where it may be exposed to cats or dogs of different colors.

As deep learning becomes ever more integrated in our lives, and as it continues to play a more significant role in multidisciplinary scientific research, it is important to consider the behavior of these models in different real-world environments. Failures of these models can lead to unintended consequences for human safety or societal fairness. Therefore, there is a clear need for *robust* perception systems – systems that perform well in different real-world environments, and hold their own against the naturally-ocurring distribution shifts between the data they are trained on and the environments they are applied in. This dissertation aims to address several facets of this problem.

One way to mitigate distribution shift in the real world is to simply learn from more data. Unfortunately, the world is a big place; and the amount of images produced today is never-ending. Proprietary industry datasets, like JFT300M [63, 154] and the Instagram hashtag dataset [105], contain labels for millions to billions of images, but these only represent a fraction of images in the world. What is the best way to take advantage of this data? Although methods for learning directly from unlabeled data are promising, the best performing

models are those trained with some amount of annotations [37]. How can we efficiently annotate data, which data samples should be annotated, and what kind of annotations should we strive for?

Instead of directly modeling the image distribution by labeling a representive dataset, we can also focus on learning the most useful features for recognition. How do we encourage model to focus on useful features in images? A simple but very effective solution is enforcing robust model invariances via *data augmentation*. Data augmentations are transformations applied to images that preserve its semantic content [148]. For example, humans are able to recognize most objects in a mirror. Applying a left-right reflection to an image is a common form of data augmentation that corresponds to this. By learning from images and their flipped versions, models are encouraged to be invariant to such mirroring transformations when recognizing objects. Which invariances should be learned by our models? As mentioned in the opening, humans are able to understand the visual world quite well, and it could be beneficial for our computer vision systems to learn similar invariances to the human perception system. Note that data augmentation can also be viewed as expanding or smoothing the empirical distribution of the training dataset through semantically-invariant transformations, and so it is closely related to the data annotation approach discussed previously.

Finally, even the best efforts in mitigating distribution shift will never be perfect, and models will inevitably make incorrect predictions. When visual perception models are used in a larger system, how can we account for model failures? This is especially critical for systems where safety is of concern, such as in autonomous navigation.

This dissertation explores some of the above questions in the following chapters.

First, Chapter 2 describes a new data annotation method to efficiently supervise perception models [96]. A direct way to overcome distribution shift is to as closely mimic the real world distribution of `(image, label)` pairs as possible in the datasets we use. However, labeling data is very expensive and time-consuming. For tasks like semantic segmentation, which aims to assign labels to every pixel, annotators must spend minutes to hours to label just a single image [28, 192]. Efficient data labeling can be achieved by simplifying the annotation task through task size reduction; focusing on providing partially-complete, yet informative, labels; or better (semi-automated) labeling tools. We propose to divide the traditional per-pixel annotation task for an entire image into per-pixel annotation for image subregions. This task is more palatable for annotators and leads to an increase in label quality for a lower cost. Furthermore, we find that only annotating a small number of image subregions is sufficiently informative for models to outperform conventional and recent weak annotation methods with any fixed budget. This work was published in ICCV 2019 [96].

Next, Chapter 3 explores paintings as a medium for studying perception systems, and their utility as an alternative source of data from natural images for training models [98]. The image distribution of paintings are different from the distribution of natural photographs, but paintings often depict meaningful objects and scenery that parallel those found in the real world. The human visual system is robust and can readily understand the depictions found in paintings. In recent work, we created a new large-scale dataset of depictions of materials in paintings spanning more than 500 years [162]. Here, we present some studies of

how common computer vision models perform when applied to the recognition of such depicted materials, like wood or fabric. Although common computer vision models are developed for natural photographs, our results show that they may be used out of the box or quickly finetuned with labeled data for material understanding in paintings. However, adapting models without labels is difficult, and we found that standard domain adaptation algorithms may *reduce* performance when applied to this data. Furthermore, an interesting attribute of paintings is that they encode quirks of human visual system through their ability to convey realism without necessarily conforming to physical reality [19, 107]. This suggsts that paintings may emphasize features or invariances that are useful for robust human-like perception. For finegrained fabric classification, we find evidence that models trained on paintings focused on cues that were both more interpretable and generalizable. This work was published in the International Workshop on Fine Art Pattern Extraction and Recognition, ICPR 2020 [98], and builds upon work that was eventually published in PLOS One 2021 [162].

Chapter 4 extends our previous findings by systematically exploring the invariances encoded in paintings for natural image recognition [97]. We study how models behave when trained with real paintings and style transfer. Style transfer is a type of data augmentation that promises to create painting-like images from natural photographs. We train models on natural photographs, paintings, and/or stylized images, and evaluate them on test data representing real-world distribution shifts. Perception models must overcome such shifts to successfully understand different environments – for example, the test photographs may contain noise, or be drawn from another dataset which was sampled from different viewpoints than the training images. Our results show that learning from a combination of natural photographs and paintings leads to models that are

far more robust than learning from natural photographs alone. Interestingly, we also find that style transfer does not capture the same invariances as paintings, and that paintings are unique among various artforms in enabling recognition models to learn useful invariances for natural image recognition. This work was published in CVPR 2021 [97].

Finally, Chapter 5 presents a real world case study in using visual perception for autonomous navigation [55]. Even the most accurate perception model will not be perfect, and it is important to account for failures when using these models as a building block in an autonomous sytem. We propose a planning pipeline that reasons about uncertainty in semantic segmentations to find a safe path in unknown environments. Given predictions of terrain and obstacles from a view of a scene, the autonomous agent determines which subsequent views of the environment to capture. By taking multiple views of the environment, the agent increases its confidence in successfully selecting a viable path across safe terrain to a goal location. Our results show that this pipeline allows safe paths to be planned when deployed in real world environments with noisy and inaccurate model predictions. This work was published in ICRA 2020 [55].

This dissertation touches upon several challenges of building robust perception systems: better data annotation pipelines for acquiring high-quality and diverse training data, learning robust features for recognition, and reasoning about perception uncertainties when using these visual recognition models are used in autonomous systems. The directions we explore in this work represent only a small fraction of the number of exciting research directions towards solving each of these challenges. In Chapter 6, we conclude with a brief discussion of fruitful directions for future work and relevant lines of concurrent research.

These include data labeling and sample selection, learning from partially-labeled or unlabeled data, building test benchmarks that represent diverse image distributions, learning robust features with data augmentation or different model architectures, and adapting models to failures on the fly.

CHAPTER 2

EFFICIENT IMAGE ANNOTATION FOR SEMANTIC SEGMENTATION

2.1 Overview

Semantic segmentation models are tasked with predicting labels for every pixel in an image. Image datasets with high-quality pixel-level annotations are valuable for learning this task, and datasets with fully-annotated images in which every pixel in an image are labeled ensure that rare classes and small objects are modeled during training. However, full-image annotations are expensive, with experts spending minutes to hours per image. As such, better annotation pipelines are needed. Cost-efficient annotation should satisfy one or more of the following properties: (a) it produces labels that contain highly informative signals for model learning (so that fewer annotations are required to train a strong perception model), (b) the task should be easy or fun to perform (so that the annotation task can be outsourced to crowdworkers without demanding high skill prequisites), and/or (c) the task should be fast to perform (so that more labels can be acquired in a shorter time period). We propose a novel annotation method which produces informative, high quality labels while also being easy to perform.

This chapter presents a novel block sub-image annotation as a replacement for conventional full-image annotation. In other words, an image is sub-divided into smaller regions, and a worker is tasked with annotating such regions from various images rather than entire images at once. Despite the attention cost of frequent task switching, we find that block annotations can be crowdsourced at higher quality compared to full-image annotation with equal monetary cost

using existing annotation tools developed for full-image annotation. Surprisingly, we find that 50% pixels annotated with blocks allows semantic segmentation to achieve equivalent performance to 100% pixels annotated. This is because sub-image blocks are still densely annotated with valuable semantic boundary information, which is crucial for learning high performing semantic segmentation models. In fact, as little as 12% of pixels annotated allows performance as high as 98% of the performance with dense annotation. In weakly-supervised settings, block annotation outperforms existing methods by 3-4% (absolute) given equivalent annotation time. To recover the necessary global structure for applications such as characterizing spatial context and affordance relationships, we propose an effective method to inpaint block-annotated images with high-quality labels without additional human effort. As such, fewer annotations can also be used for these applications compared to full-image annotation.

The work in this chapter was published in ICCV 2019 as "Block Annotation: Better Image Annotation with Sub-Image Decomposition" [96] with Paul Upchurch and Kavita Bala.

2.2 Introduction

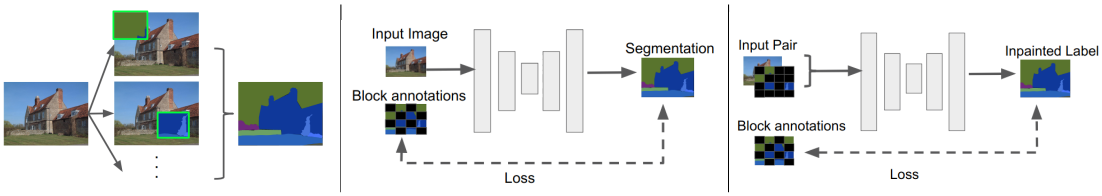


Figure 2.1: **Block Annotation Overview.** (a) Sub-image block annotations are more effective to gather than full-image annotations (b) Training on sparse block annotations enables semantic segmentation performance equivalent to full-image annotations (c) Block labels can be inpainted with high-quality labels.

Recent large-scale computer vision datasets place a heavy emphasis on high-quality fully dense annotations (in which over 90% of the pixels are labelled) for hundreds of thousands of images. Dense annotations are valuable for both semantic segmentation and applications beyond segmentation such as characterizing spatial context and affordance relationships [17, 58]. The long-tail distribution of classes means it is difficult to gather annotations for rare classes, especially if these classes are difficult to segment. Annotating every pixel in an image ensures that pixels corresponding to rare classes or small objects are labelled. Dense annotations also capture pixels that form the boundary between classes. For applications such as understanding spatial context between classes or affordance relationships, dense annotations are required for principled conclusions to be drawn. In the past, polygon annotation tools have enabled partially dense annotations (in which small semantic regions are densely annotated) to be crowdsourced at scale with public crowd workers. These tools paved the way for the cost-effective creation of large-scale partially dense datasets such as [10, 99]. Despite the success of these annotation tools, fully dense datasets have relied extensively on expensive expert annotators [186, 28, 116, 192, 119] and private crowdworkers [17].

We propose annotation of small blocks of pixels as a stand-in replacement for full-image annotation (figure 2.1). We find that these annotations can be effectively gathered by crowdworkers, and that annotation of a sparse number of blocks per image can train a high performance segmentation network. We further show these sparsely annotated images can be extended automatically to full-image annotations.

We show block annotation has:

- **Wide applicability.** (Section 2.4) Block annotations can be effectively crowdsourced at higher quality compared to full annotation. It is easy to implement and works with existing advances in image annotation.

- **Cost-efficient Design.** (Section 2.4) Block annotation reflects a cost-efficient design paradigm (while current research focuses on reducing annotation time). This is reminiscent of gamification and citizen science where enjoyable tasks lead to low-cost high-engagement work.

- **Complex Region Annotation.** (Section 2.4) Block annotation shifts focus from labeling regions of specific semantic categories (e.g., "label all dogs in this image") to spatial regions (e.g., "label all regions in this part of the image"). When annotating categorical regions, workers segment simple objects before complex objects. With spatial regions, informative complex regions are forced to be annotated.

- **Weakly-Supervised Performance.** (Section 2.5) Block annotation is competitive in weakly-supervised settings, outperforming existing methods by 3-4% (absolute) given equivalent annotation time.

- **Scalable Performance.** (Section 2.5) Full-supervision performance is achieved by annotating 50% of blocks per image. Thus, blocks can be annotated until desired performance is achieved, in contrast to methods such as scribbles.

- **Scalable Structure.** (Section 2.6) Block-annotated images can be effectively inpainted with high quality labels without additional human effort.

2.3 Related Work and Background

In this section we review recent works on pixel-level annotation in three areas: human annotation, and human-machine annotation, and dense segmentation with weak supervision.

Human Annotation. Manual labeling of every pixel is impractical for large-scale datasets. A successful method is to have crowdsource workers segment polygonal regions to click on boundaries. Employing crowdsource workers offers its own set of challenges with quality control and task design [173, 10, 160]. Although large-scale public crowdsourcing can be successful [99] recent benchmark datasets have resorted to in-house expert annotators [28, 118]. Annotation time can be reduced through improvements such as autopan, zoom [10] and shared polygon boundaries [186]. Polygon segmentation can be augmented by painted labels on superpixel groups [17] and Bezier curves [186]. Pixel-level labels for images can also be obtained by (1) constructing a 3D scene from an image collection, (2) grouping and labeling 3D shapes and (3) propagating shape labels to image pixels [111]. In our work, we investigate sub-image polygon annotation, which can be further combined with other methods (sec. 2.4.)

Human-Machine Annotation. Complex boundaries are time-consuming to trace manually. In these cases the cost of pixel-level annotation can be reduced by automating a portion of the task. Matting and object selection [139, 88, 89, 8, 180, 179, 16, 84, 181] generate tight boundaries from loosely annotated boundaries or few inside/outside clicks and scribbles. [124, 110] introduced a predictive method which automatically infers a foreground mask from 4 boundary clicks, and was extended to full-image segmentation in [3]. The number of boundary

clicks was further reduced to as few as one by [1]. Predictive methods require an additional human verification step since the machine mutates the original human annotation. The additional step can be avoided with an online method. However, online methods (e.g., [1, 84, 3]) have higher requirements since the algorithm must be translated into the web browser setting and the worker's machine must be powerful enough to run the algorithm[1]. Alternatively, automatic proposals can be generated for humans to manipulate: [6] generates segments, [5] generates a set of matting layers, [188] generates superpixel labels, and [134] generates boundary fragments. In our work, we show that human-annotated blocks can be extended automatically into dense annotations (sec. 2.6), and we discuss how other human-machine methods can be used with blocks (sec. 2.4.6).

Weakly-Supervised Dense Segmentation. There are alternatives to training with high-quality densely annotated images which substitute quantity for label quality and/or richness. Previous works have used low-quality pixel-level annotations [195], bounding boxes [125, 75, 137], point-clicks [9], scribbles [9, 95], image-level class labels [125, 147, 4], image-level text descriptions [64] and unlabeled related web videos [64] to train semantic segmentation networks. Combining weak annotations with small amounts of high-quality dense annotation is another strategy for reducing cost [11, 66]. [145] proposes a two-stage approach where image-level class labels are automatically converted into pixel-level masks which are used to train a semantic segmentation network. We find a small number of sub-image block annotations is a competitive form of weak supervision (sec. 2.5.3).

[1]Offloading online methods onto a cloud service offers a different landscape of higher costs (upfront development and ongoing operation costs).

2.4 Block Annotation

Sub-image block annotation is composed of three stages: (1) Given an image I, select a small spatial region I'; (2) Annotate I' with pixel-level labels; (3) Repeat (with different I') until I is sufficiently annotated. In this chapter, we explore the case where I' is rectangular, and focus on the use of existing pixel-level annotation tools.

Can block annotations be gathered as effectively as full-image annotations with existing tools? In section 2.4.1, we show our annotation interface. In section 2.4.2, we explore the quality of block annotation versus full-image annotation. In section 2.4.3, we examine block annotation for a real-world dataset. In section 2.4.4, we discuss the cost of block annotation and show worker feedback. In section 2.4.5, we discuss how blocks for annotation can be selected in practice. Finally, in section 2.4.6 we discuss the compatibility of block annotation with existing annotation methods.

2.4.1 Annotation Interface

Our block annotation interface is given in figure 2.2 and implemented with existing tools [10]. For full image annotation, the highlighted block covers the entire image. Studies are deployed on Amazon Mechanical Turk.

(a) Highlighted block. (b) Finished block annotation.

Figure 2.2: **Block Annotation UI**. Annotators are given one highlighted block to annotate with the remainder of the image as context.

2.4.2 Quality of Block Annotation

We explore the quality of block annotations compared to full-image annotations on a synthetic dataset. How does the quality and cost compare between block and full annotations? We find that *the average quality for block-annotated images is higher while the total monetary cost is about the same*.

The average quality of block annotations is consistently higher, including in small regions (figure 2.3 shows an example). The overall block annotation error is 12% lower than the error from full annotation; for regions smaller than 0.5% of the image, the block annotation error is 6% lower. Figure 2.4 presents the annotation error for regions where the ground truth segments are within some range of sizes proportional to the full image size (480×640 px). Block annotations consistently have a lower error rate in these ranges, indicating that block annotation is advantageous regardless of the scale of segments. For completeness, the rightmost bars show the error rate for segments of all sizes.

In figure 2.5, the cost and quality of block versus full image annotation is shown. Remarkably, we find that *workers are willing to work on block annotation tasks for a significantly lower hourly wage. This indicates that block annotation is*

(a) (b) (c)

Figure 2.3: **SUNCG/CGIntrinsics annotation**. (a) Ground truth. (b) Block annotation (zoomed-in) (c) Full annotation (zoomed-in). White dotted box highlights an example where block annotation qualitatively outperforms full annotation. More in appendix.

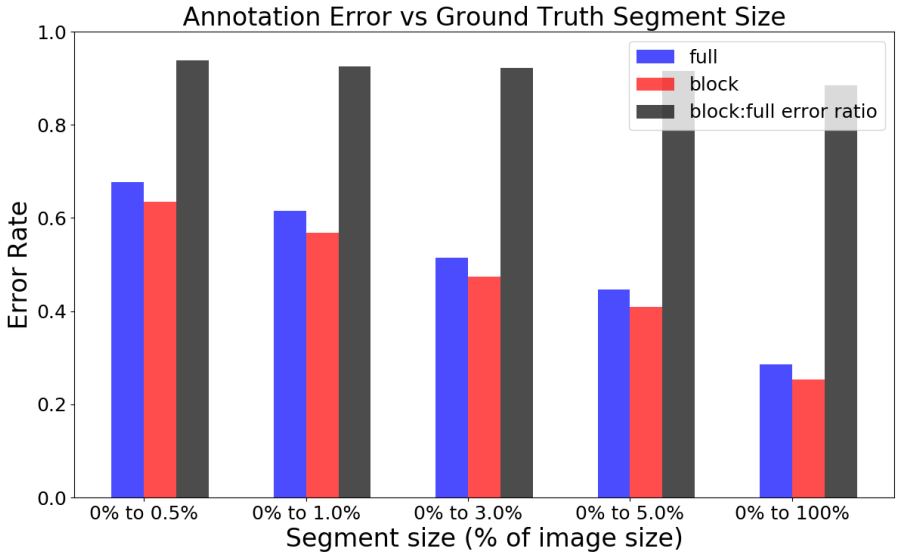
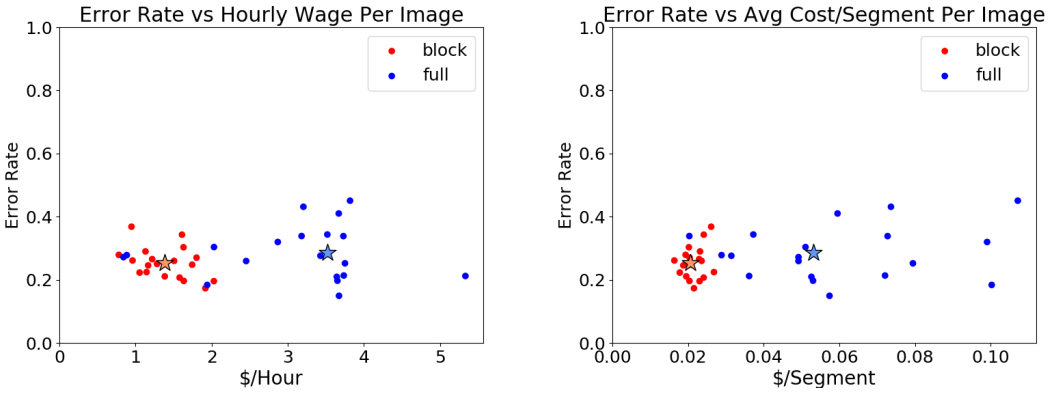


Figure 2.4: **Annotation error rate** for block and full annotation for differently sized ground truth segments. **Lower is better**.

more intrinsically palatable for crowdworkers, in line with [67] which shows task design can influence quality of work. Moreover, workers are more likely to over-segment objects with respect to ground truth (e.g. individual cushions on a couch, handles on cabinets) with block annotation tasks. Note that block boundaries may also divide semantic regions. Table 2.1 contains additional statistics. Despite similar costs to annotate an image in blocks or in full, we show

in section 2.5 that competitive performance is achieved with less than half of the blocks annotated per image.



(a) (b)

Figure 2.5: **Annotation error rate** for block and full annotation. Each point represents one image. The same set of images are both block annotated and full-image annotated. The stars represent the centroid (median). Cost/time include estimated cost/time to assign labels for each segment [10]. **Lower-left is better**. With block annotation, workers (a) choose to work for lower wages and (b) segment more regions for less pay per region. The overall quality is higher for block annotation.

.

	Block	Full
Error Error (small regions)	0.253 0.636	0.286 0.677
$ / hr	$1.40 / hr	$3.12 / hr
Total cost Total cost (median)	$2.00 $1.99	$2.05 $2.23
# segments $ / segment	95.68 $0.0215	38.95 $0.0595

Table 2.1: **Block vs Full Annotation**. Average cost and error statistics per image. Error is measured via IoU against ground truth segments in the synthetic SUNCG/CGIntrinsics dataset.

Study Details. For these experiments, we chose to use a synthetic dataset. While human annotations may contain mistakes, synthetic datasets are generated with known ground truth labels with which annotation error can be

computed. The CGIntrinsics dataset [94] contains physically-based renderings of indoor scenes from the SUNCG dataset [151, 190]. We use the more realistic CGIntrinsics renderings and the known semantic labels from SUNCG. The labels are categorized according to the NYU40[52] semantic categories. Due to the nature of indoor scenes, the depth and field of view of each image is smaller than outdoor datasets. The reduced complexity means that crowdworkers are able to produce good full-image annotations for this dataset.

We select MTurk workers who are skilled at both full-image annotation and block annotation in a pilot study (a standard quality control practice [10]). The final pool consists of 10 workers. Image difficulty is estimated by counting the maximum number of ground truth segments in a fixed-size sliding window. Windows, mirrors, and void regions are masked out in the images so that workers do not expend effort on visible content for which ground truth labels do not exist (such as objects seen through a window or mirror). We manually cull images that include transparent glass tables which are not visible in the renderings, or doorways through which visible content can be seen but no ground truth labels exist. After filtering, twenty of the one hundred most difficult images are selected. We choose a block size so that an average of 3.5 segments are in each block. This results in 16 blocks per image. For each task, a highlighted rectangle outlines the block to be annotated. We find that workers will annotate up to the inner edge of the highlighted boundary. Therefore, we ensure the edges of the rectangle do not overlap with the region to be annotated.

Workers are paid \$0.06[2] per block annotation task and \$0.96 per full-image task. Bonuses up to 1.5 times the base pay are awarded to attempt to raise the effective hourly wage for difficult tasks to \$4 / hr. Our results show that

[2]\$ refers to USD in throughout this chapter.

workers are willing to work on block annotation tasks beyond the time threshold for bonuses, effectively producing work for an hourly wage significantly lower than the intended $4 / hr. On the other hand, workers do not often exhibit this behavior with full annotation tasks. Different workers may work on different blocks belonging to the same image. We use two forms of quality control: (1) annotations must contain a number of segments greater than 25% of the known number of ground-truth segments for that task and (2) annotations cannot be submitted until at least 10 seconds / 3 minutes (block / full) have passed. All submissions satsifying these conditions are accepted during the user study. For an overview of QA methods, please refer to [135]. Labels are assigned by majority ground-truth voting, with cost estimated from [10].

To evaluate the quality of annotations in an image with K classes, we measure the class-balanced error rate (class-balanced Jaccard distance):

$$\begin{aligned} error\,rate &= \frac{1}{K}\sum_{c=1}^{K}\frac{(FP_c + FN_c)}{(TP_c + FP_c + FN_c)} \\ &= 1 - mIOU \end{aligned} \tag{2.1}$$

2.4.3 Viability of Real-World Block Annotation

How does block annotation fare with a real-world non-synthetic dataset? To study the viability of block annotating real-world datasets with scalable crowdsourcing, we ask crowdworkers to annotate blocks from images in Cityscapes [28]. We choose Cityscapes for the annotation complexity of its scenes – 1.5 hours of expert annotation effort is required per image. In contrast, other datasets such as [116, 17] require less than 20 minutes of annotation effort per image.

We expect crowd work to be worse than expert work, so it is a surprisingly positive result that the quality of the crowdsourced segments are visually comparable to the expert Cityscapes segments (figure 2.6). Some crowdsourced segments are very high quality. segments (20% have fewer segments, and 33% have the same # of segments). A summary of the cost is given in table 2.2 which compares public crowdworkers to trained experts. It is feasible block annotation time will decrease with expert training. Given 100 uniformly sized blocks per image, we ask an expert to create equal-quality block and full annotations; we find one block is 1.56% of the effort of a full image.

Figure 2.6: **Crowdsourced vs expert segments**. Crowdsourced block-annotated segments are compared to expert Cityscapes segments. Crowdsourced segments are colored for easier comparison. Top-left is a high-quality example. See appendix for more.

Study Details. We searched for workers who produce high-quality work in a pilot study and found a set of 7 workers. These workers were found within a hundred pilot HITs (for a total cost of $4). We approved all of their submissions during the user study. We do not restrict workers from annotating outside of the

	Block (Crowd)	Full (Expert [28, 119])
$ / Task	$0.13	-
Time / Task	2 min	1.5 hr

Table 2.2: **Real-world cost of annotation.** Cost evaluated on Cityscapes. Each block is annotated by MTurk workers. Full-image is annotated by experts in [28]. Note: [28] annotates instance segments. See table 2.1 for crowd-to-crowd comparison.

block, and we do not force workers to densely annotate the block. We do not include the use of sentinels or tutorials as in [10].

Thirteen randomly selected validation images from Cityscapes are annotated by crowdworkers. Each image is divided into 100 uniformly shaped blocks. A total of 650 (50 per image) are annotated in random order. Workers are paid $0.06 per task. Workers are automatically awarded bonuses so that the effective hourly wage at least $5 for each block, with bonuses capped at $0.24 to prevent abuse. For one block, the total base payout is $0.06 with an average of $0.0636 in bonuses over 93 seconds of active work. On average, each annotated block contains 3.5 segments. Assigning class labels will cost an additional $0.01 and 26 seconds [10]. To be consistent with Cityscapes, we instruct workers to not segment windows, powerlines, or small regions of sky between leaves. However, workers will occasionally choose to do so and submit higher quality segments than required.

2.4.4 Annotation Cost and Worker Feedback

Our costs (tables 2.1, 2.2) are aligned with existing large-scale studies. Large-scale datasets [10, 99] show that cultivating good workers produces high quality data at low cost. Table 2 of [56] reports a median wage of $1.77/hr to $2.11/hr; the

median MTurk wage in India is $1.43/hr [57]. For "image transcription", the median wage is $1.13/hr over 150K tasks.

Workers gave overwhelmingly positive feedback for block annotation (table 2.3), and we found that some workers would reserve hundreds of block annotation tasks at once. Only 3 out of the 57 workers who successfully completed at least one pilot or user study task requested higher pay; all other given feedback was positive or neutral. In contrast, our pilot studies showed that *workers are unwilling to accept full-image annotation tasks if the payment is reduced to match the wage of block annotation.* We conjecture that task enjoyment leads to long term high-quality output (c.f. [67]).

	"Nice" "Good" "Great"	"Fun" "Happy"	"Easy"	"Okay"	Release More HITs	Increase Pay
#	8	5	4	2	2	3

Table 2.3: **Block annotation worker feedback**. Free-form responses are aggregated over SUNCG and Cityscapes experiments, and collected at most once per worker. All 24 sentiments across all 19 worker responses are summarized.

2.4.5 Block Selection

Our experiments show that workers are comfortable annotating between 3 to 6 segments per block. Therefore, block size can be selected by picking a size such that the average number of segments per block falls in this range. For a novel dataset, this can be done fully labelling several samples and producing an estimate from the fully labelled samples. Without priors on spatial distribution of rare classes or difficult samples within an image, a checkerboard or pseudo-checkerboard pattern of blocks focuses attention (across different tasks)

uniformly across the image. Far apart pixels within an image are less correlated than neighboring pixels. Therefore, it is good to sample blocks that are spread out to encourage pixel diversity within images.

2.4.6 Compatibility with Existing Annotation Methods

Block annotation is compatible with many annotation tools and innovations besides polygon boundary annotation.

Point-clicks and Scribbles. Annotations such as point clicks or scribbles are faster to acquire than polygons, which leads to a larger and more varied dataset at the same cost. Combining this with blocks will further increase annotation variety due to the diversity that come from annotating a few blocks in many images over annotating fewer number of images fully. Additionally, [9, 11] show that the most cost-effective method for semantic segmentation is a combination of densely annotated images and a large number of point clicks. The densely annotated images can be replaced by polygon block annotations since they also contain class boundary supervision for the segmentation network.

Superpixels. Superpixel annotations enable workers to mark a group of visually-related pixels at once [17]. This can reduce the annotation time for background regions and objects with complex boundaries. Superpixel annotation can be easily deployed to our block annotation setting.

Polygon Boundary Sharing. Boundary sharing reuses existing boundaries so that workers do not need to trace each boundary twice [186]. This approach can be easily deployed in our block annotation setting.

Curves. Bezier tools allow workers to quickly annotate curves [186]. It can be easily deployed in our block annotation setting but it may be less effective on long curves since each part of the curve must be fit separately.

Interactive Segmentation. Recent advances in interactive segmentation (e.g., [1, 110, 3]) utilize neural networks to convert sparse human inputs into high quality segments. For novel domains without large-scale training data, block-annotated images can act as cost-efficient seed data to train these models. Once trained, these methods can be applied directly to each block, although further analysis should be conducted to explore the efficiency of such an approach due to block boundaries splitting semantic regions.

2.5 Segmentation Performance

How well do block annotations serve as training data for semantic segmentation? In section 2.5.1, the experimental setup is summarized. In section 2.5.2, we evaluate the effectiveness of block annotations for semantic segmentation. In section 2.5.3, we compare block annotation with existing weakly supervised segmentation methods.

2.5.1 Experimental Setup

Pixel Budget. We vary the "pixel budget" in our experiments to explore segmentation performance across a range available annotated pixels. "Pixel budget" refers to the % of pixels annotated across the training dataset, which can be

controlled by varying the number of annotated images, the number annotated blocks per image, and the size of blocks per image. Our block sizes are fixed in our experiments.

Block Size. We divide images into a 10-by-10 grid for our experiments.

Block Selection. We experiment with two block selection strategies: (a) Checkerboard annotation and (b) Pseudo-checkerboard annotation. Checkerboard annotation means that every other block in a variable number of images are annotated. Pseudo-checkerboard annotation means that every N blocks are annotated in every image, where N is $\frac{\texttt{\# pixels in dataset}}{\texttt{pixel budget}}$. For example, with a pixel budget equivalent to 25% of the dataset, every fourth block is annotated for the entire dataset. At pixel budget 50%, checkerboard and pseudo-checkerboard are identical.

For the remainder of the chapter, "Block-X%" refers to pseudo-checkerboard annotation in which X% of the blocks per image are annotated.

Sementation Model. We use DeepLabv3+[24] initialized with the official pretrained checkpoint (pretrained on ImageNet [33] + MSCOCO [99] + Pascal VOC [39]). The network is trained for a fixed number of epochs. See appendix for additional details.

Datasets. Cityscapes is a dataset with ground truth annotations for 19 classes with 2975 training images and 500 validation images. ADE20K contains ground truth annotations for 150 classes with 20210 training images and 2000 validation images. These datasets are chosen for their high quality dense ground truth annotations and for their differences in number of images / classes and types of

scenes represented. The block annotations are synthetically generated from the existing annotations.

2.5.2 Evaluation

Blocks vs Full Image. How does block annotation compare to full-image annotation for semantic segmentation? We plot the mIOU achieved when trained on a set of annotations against pixel budget in figure 2.7.

For both Cityscapes and ADE20K, *block annotation significantly outperforms full-image annotation*. The performance gap widens as the pixel budget is decreased – at pixel budget 12%, the reduction in error from full annotation to block annotation is 13% on Cityscapes (10% for ADE20K). For any pixel budget, pseudo-checkerboard block annotation annotates fewer pixels per image which means more images are annotated. Therefore, our results indicate that the quantity of annotated images is more valuable than the quantity of annotations per image. The pseudo-checkerboard block selection pattern consistently outperforms the checkerboard block selection pattern and full annotation.

Number of Blocks vs Optimal Performance. How many blocks need to be annotated for segmentation performance to approach the performance achieved by training on full-image annotations for the entire dataset? In table 2.4, we show results when the network is trained on the full dataset compared to pseudo-checkerboard blocks.

Remarkably, we find that *checkerboard blocks with 50% pixel budget allow the network to achieve similar performance to the full dataset with 100% pixel budget*,

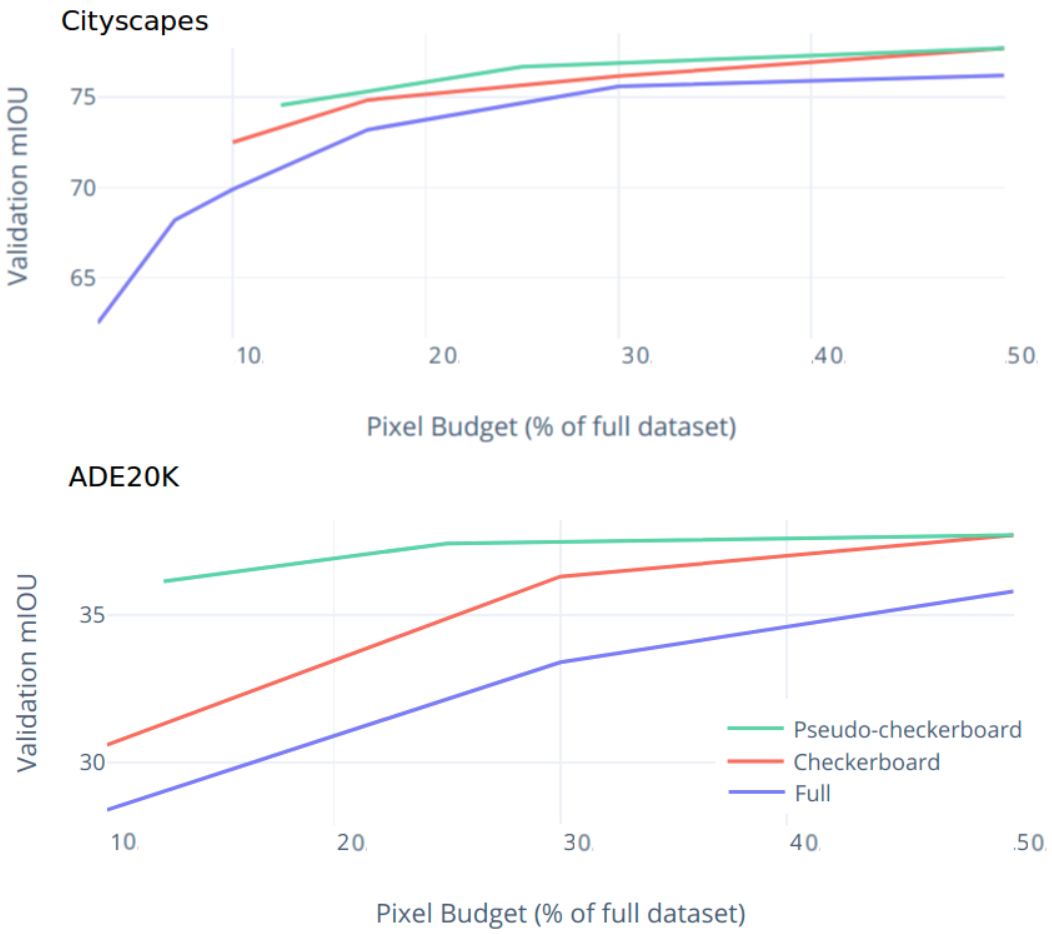


Figure 2.7: **Semantic segmentation performance**. Training images are annotated with different pixel budgets. Pseudo-checkerboard block annotation outperforms checkerboard and full annotation.

	Optimal (Full)	Block-50%	Block-12%
Cityscapes	77.7%	77.7%	74.6%
ADE20K	37.4%	37.2%	36.1%

Table 2.4: **Semantic segmentation performance** when trained on all images. Training with block annotations uses fewer annotated pixels than full annotation but achieves equivalent performance.

indicating that at least 50% of the pixels in Cityscapes and ADE20K are redundant for learning semantic segmentation. Furthermore, with only 12% of the pixels in the dataset annotated, relative error in segmentation performance is within 12%/2% of the optimal for Cityscapes/ADE20K. These results suggest that fewer

than 50% of the blocks in an image need to be annotated for training semantic segmentation, reducing the cost of annotation reported in section 2.4.

Block Locations vs Performance. How does the location of blocks sampled within an image affect semantic segmentation performance? In this experiment, we train a model with either (a) blocks sampled in a checkerboard pattern or (b) uniformly randomly sampled from the same grid. In both cases, 50% of the blocks are labelled in each image.

The network trained on randomly sampled blocks achieves 77.1% mIOU while the network trained on checkerboard blocks achieves 77.7% mIOU. The increase in performance in checkerboard annotations over randomly sampled blocks is due to pixel diversity (pixels far apart in an image are less correlated than neighboring pixels). This is similar to the effect observed in the first experiment which shows that pixel diversity due to image diversity increases performance. These observations are aligned with our expectations in section 2.4.5. In this experiment, we used all images from Cityscapes and created: (a) random block annotations (sample 50% of blocks for each image) and (b) checkerboard block annotations. To ensure that results are not due to sampling bias, we create split (a) three times (i.e. sample 50% of blocks per image three times) and average the results over the three splits.

2.5.3 Weakly Supervised Segmentation Comparison

Block annotation can be considered a form of weakly supervised annotation where a small number of pixels in an image are labelled. Representative works in this area include [95, 9, 126, 125, 30]. Table 3 of [95] is replicated here (table 2.5) for

reference, and extended with our results. All existing results show performance with a VGG-16 based model. We train a MobileNet based model which has been shown to achieve similar performance to VGG-16 (71.8% vs 71.5% Top-1 accuracy on ImageNet) while requiring fewer computational resources [65, 141]. Our fully-supervised implementation pretrained on ImageNet achieves 69.6% mIOU on Pascal VOC 2012 [39]; in comparison, the reference DeepLab-VGG16 model achieves 68.7% mIOU [23] and the re-implementation in [95] achieves 68.5% mIOU.

Method	Annotations	mIOU (%)
MIL-FCN [126]	Image-level	25.1
WSSL [125]	Image-level	38.2
point sup. [9]	Point	46.1
ScribbleSup [95]	Point	51.6
WSSL [125]	Box	60.6
BoxSup [30]	Box	62.0
ScribbleSup [95]	Scribble	63.1
Ours: Block-1%	Pixel-level Block	61.2
Ours: Block-5%	Pixel-level Block	67.6
Ours: Block-12%	Pixel-level Block	68.4
Full Supervision	Pixel-level Image	69.6

Table 2.5: **Weakly-supervised segmentation performance**. Evaluated on Pascal VOC 2012 validation set. Original table from [95]. Blocks (N%) indicates N% of image pixels (N pseudo-checkerboard blocks) are labelled.

Cityscapes	**Ours:** Block (7 min)	Coarse (7 min [28])	Full Supervision (90 min [28])
mIOU (%)	**72.1**	68.8	77.7
Pascal	**Ours:** Block (25 sec)	Scribbles (25 sec [95])	Full Supervision (4 min [116])
mIOU (%)	**67.2**	63.1 [95]	69.6

Table 2.6: **Weakly-supervised segmentation performance given equal annotation time**. For time comparison of scribbles against other methods, please refer to [95].

Performance Comparison. With only 1% of the pixels annotated, block annotation achieves comparable performance to existing weak supervision methods. Based on our results in section 2.4.2, the cost of annotation for 1% of pixels with blocks will be 100× less than the cost of full-image annotation. Increasing the budget to 5%-12% significantly increases performance. With 12% of pixels annotated with blocks, the segmentation performance (error) is within 98% (4%) of segmentation performance (error) with 100% of pixels annotated.

Note that block annotations can be directly transformed into gold-standard fully dense annotations by simply gathering more block annotations within an image. This is not feasible with other annotations such as point clicks, scribbles, and bounding boxes. Furthermore, in section 2.6, we demonstrate a method to transform block annotations into dense annotations without any additional human effort.

Equal Annotation Time Comparison. Given equal annotation time, block annotation significantly outperforms coarse and scribble annotations by ∼3-4% mIOU (table 2.6). On Pascal, 97% of full-supervision mIOU is achieved with 1/10 annotation time. We convert annotation time to number of annotated blocks as follows. Block annotation may use up to 2.2× the time of full-image annotation. Given an image divided into 100 blocks, an annotation time of T leads to $\frac{T}{0.022F}$ blocks annotated, where F is the full-image annotation time.

2.6 Block-Inpainting Annotations

Although block annotations are useful for learning semantic segmentation, the full structure of images is required for many applications. Understanding the spatial context or affordance relationships [17, 58] between classes relies on understanding the role of each pixel in an image. Shape-based retrieval, object counting [87], or co-occurrence relationships [115] also depend on a global understanding of the image. The naive approach to recover pixel-level labels is to use automatic segmentation to predict labels. However, this does not leverage existing annotations to improve the quality of predicted labels. In section 2.6.1, we propose a method to inpaint block-annotated images by using annotated blocks as context. In section 2.6.2, we examine the quality of these inpainted annotations.

2.6.1 Block-Inpainting Model

The goal of the block-inpainting model is to inpaint labels for unannotated blocks given the labels for annotated blocks in an image. For full implementation details, please refer to the appendix.

Architecture. The block-inpainting model is based on DeepLabv3+. The input layer is modified so that the RGB image, $I \in \mathbb{R}^{h \times w \times 3}$, is concatenated with multichannel "hint" (ala [189]) of 1-0 class labels $W \in \mathbb{R}^{h \times w \times K}$ where K is the number of classes. At inference time, the hint contains known labels for the annotated blocks of an image which serve as context for the inpainting task. Hidden layers are augmented with dropout which will be used to control quality

by estimating epistemic uncertainty [44, 45].

Estimating Uncertainty. Inpainting fills all missing regions without considering the trade off between quantity and quality. Existing datasets have high-quality annotations for 92-94% of pixels [192, 17]. Therefore, we modify our network to produce uncertainty estimates which allow us to explicitly control this trade off. The uncertainty of predictions is correlated with incorrect predictions [86, 81]. Uncertainty is computed by activating dropout at inference time. The predictions are averaged over the g trials giving us $U \in \mathbb{R}^{h \times w}$, a matrix of uncertainty estimates per image. We take the sample standard deviation corresponding to the predicted class for each pixel to be the uncertainty. For each pixel (i, j), the mean softmax vector over g trials is:

$$\boldsymbol{\mu}^{(i,j)} = \frac{\sum_{t=1}^{g} \mathbf{p}^{(i,j)}(y|I, W)}{g} \tag{2.2}$$

where $\mathbf{p}(y|I, W) \in \mathbb{R}^K$ is the softmax output of the network. The corresponding uncertainty vector is:

$$\mathbf{U}'^{(i,j)} = \sqrt{\frac{\sum_{t=1}^{g} (\mathbf{p}^{(i,j)}(y|I, W) - \boldsymbol{\mu}^{(i,j)})^2}{g - 1}} \tag{2.3}$$

Thus, the uncertainty for each pixel (i, j) is:

$$U^{(i,j)} = U'^{(i,j)}_m, \text{ where } m = \arg\max_k \mu_k^{(i,j)} \tag{2.4}$$

Training. Block annotations serve both as hints and targets. This means that no additional data (or human annotation effort) is required to train the block-inpainting model. For our experiments, we use (synthetically generated) Block-50% annotations. For each image, half of annotated blocks are randomly selected

online at training time to be hints. All of the annotated blocks are used as targets. This encourages the network to "copy-paste" hints in the final output while leveraging the hints as context to inpaint labels for regions where hints are not provided.

2.6.2 Evaluation

Quality of Inpainted Labels. How good are inpainted labels? We compare labels produced by the block-inpainting network with low $U^{(i,j)}$ against the known human labels in Cityscapes and ADE20K. *The block-inpainting model produces labels whose human-agreement is competitive with that achieved by human annotators.* We inpaint Block-50% annotations in this experiment. At a relative uncertainty threshold of 0.2 on Cityscapes (0.4 on ADE20K), over 94% of the pixels are labelled – recall that existing datasets have over 92-94% [192, 17] pixels annotated. For Cityscapes, the mean pixel agreement is 99.8% and the class-balanced error rate is 3.1%, while for ADE20K, the mean pixel agreement is 98.7% and the class-balanced error rate is 28%. Recall that ADE20K features significantly more semantic class categories than Cityscapes. These machine-human label agreements are extremely good. Previous work show that human-human label agreement across annotators is 66.8% to 73.6% while annotator self-agreement is 82.4% to 97.0% [192, 17]. Human annotators fail to agree in non-trivial fashion – [192] shows that annotator self-agreement fails in three ways: variations in complex boundaries (32%), incorrect naming of ambiguous classes (34%), and failure to segment small objects (34%). In figure 2.8, a visualization of labels generated by the block-inpainting model is shown. The number of pixel disagreements decreases with a higher uncertainty threshold.

(a) Full human labels

(b) Original image

(c) Inpainted labels (all)

(d) Label agreement (white)

(e) Inpainted labels (<20% relative uncertainty)

(f) Label agreement (white)

Figure 2.8: **Block-inpainted labels.** Example of human labels vs human Block-50% + inpainted labels. Void labels are masked out.

Block Inpainting vs Automatic Segmentation. Consider a scenario in which a small number of pixels in a dataset are annotated, and the remainder are automatically labelled to produce dense annotations. Why should block inpainting be used instead of automatic segmentation? *Full pixel-level labels produced by block inpainting are superior to automatic segmentation.* On Cityscapes, automatic segmentation achieves 78% validation mIOU while block inpainting Block-50% annotations achieves 92% validation mIOU. With Block-12% annotations, automatic segmentation achieves 75% validation mIOU while block inpainting achieves 82% validation mIOU.

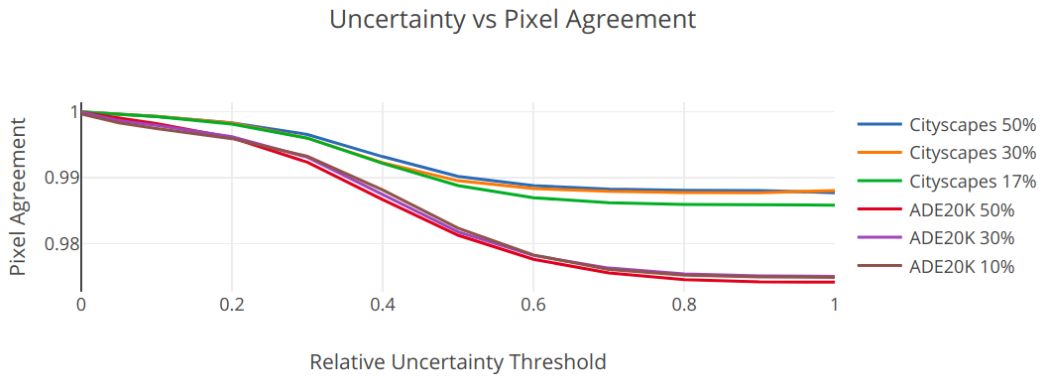


Figure 2.9: **Block-Inpainting Model** uncertainty versus human pixel-wise agreement for inpainted labels. Curves for different pixel budgets shown for comparison.

2.6.3 Ablations

Effect of Uncertainty Threshold vs Pixel Agreement. How does the uncertainty threshold affect pixel agreement for block-inpainting? In figure 2.9, we show the mean pixel agreement with human labels for varying thresholds. Lower uncertainty threshold for rejection results in higher pixel agreement. The pixel agreement with lower pixel budgets are shown for comparison. The pixel budget is the number of block-annotated pixels in the dataset with which the block-inpainting model is trained. All experiments use checkerboard block hints.

Effect of Uncertainty Threshold vs Pixel Coverage. How does the uncertainty threshold affect pixel coverage for block-inpainting? In figure 2.10, we show the mean pixel coverage for varying uncertainty thresholds as a fraction of maximum uncertainty. Lower uncertainty threshold for rejection results in lower pixel coverage. The pixel coverage with lower pixel budgets are shown for comparison. The pixel budget is the number of human block-annotated pixels in the dataset with which the block-inpainting model is trained. All experiments use checkerboard block hints.

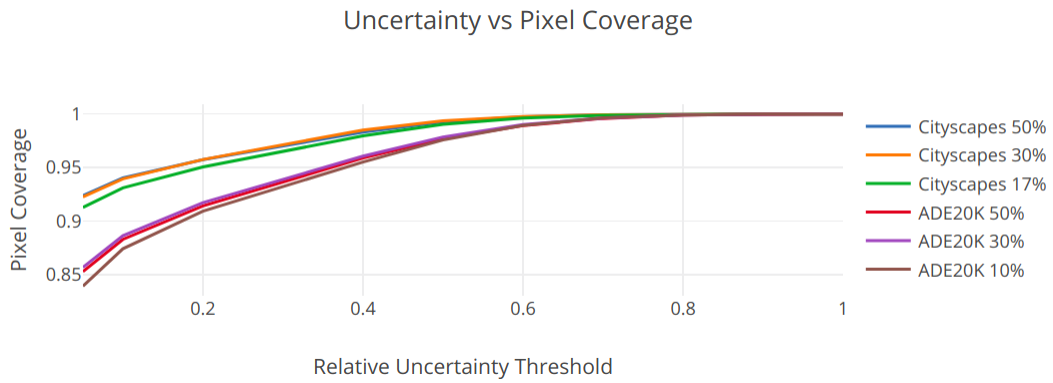


Figure 2.10: **Block-Inpainting Model** uncertainty versus pixel coverage for human checkerboard + automatic labels. x-axis truncated at 0.05 on left. Curves for different pixel budgets shown for comparison.

Block Selection vs Block-Inpainting Quality. How does the checkerboard pattern compare to other block selection strategies as hints to the block-inpainting model? Intuitively, it is easier to infer labels for pixels that are close to pixels with known labels than for pixels that are further away. Consider a scenario in which every other pixel in an image is annotated. Reasonably good labels for the unannotated pixels can be inferred with a simple nearest-neighbors algorithm. In practice, it is impossible to precisely annotate single pixels in an image. However, we can approximate the same properties of labelling every other pixel by labelling every other block instead (i.e., a checkerboard pattern).

In table 2.7, we show the block-inpainting model mIOU when different types of hints are given. The rightmost column ("every other pixel") is not feasible to collect in practice. Checkerboard annotations outperform random block annotations even though the network is trained to expect random block hints. Providing only boundary annotations within each block (i.e. annotating pixels within 10 pixels of each boundary in each block) allows the network to achieve nearly the same performance as full block hints. This suggests that the most informative pixels for the block-inpainting model are those near a

	None	Random (Semantic Boundaries)	Random (Full)	Checker (Full)	Every Other Pixel
Rel. mIOU	0.77	0.90	0.92	0.95	1.0

Table 2.7: **Block-inpainting with different types of hints.** "Every other pixel" annotations represents an ideal sampling of pixels to annotate as hints – this sampling strategy is infeasible in practice. Relative performance of inpainting by utilizing hints with respect to "every other pixel"-hints is shown. Checkerboard sampling with fully annotated blocks outperform no hints, random blocks (with only semantic boundaries annotated within blocks), and random blocks (with fully annotated blocks).

boundary.

2.7 Discussion and Future Work

An incredibly large number of images exist in the world – from social media to car dashcam reels to security camera footage to other publicy shared photography collections. It is important to efficiently annotate such unlabeled images with useful labels to train a strong computer vision model. In this chapter, we have introduced a new annotation method for efficiently assigning semantic segmentation labels to the vast number of unlabeled images at our disposal. We proposed block annotation as a crowdworker-friendly replacement for traditional full-image annotation. For training semantic segmentation models, Block-12% offers strong performance at 1/8th of the monetary cost. Block-5% offers competitive weakly-supervised performance at equal annotation time to existing methods. For optimal semantic segmentation performance, or to recover global structure with inpainting, Block-50% can be utilized; as we found in our experiments, not every pixel in each image needs to be annotated for maximal performance to be achieved.

There are many directions for future work. Our crowdworker subimage annotation tasks are similar to full-image annotation tasks, so it may be possible to improve the gains with more exploration and development of boundary marking algorithms similar to those designed for traditional full-image annotation. We have explored some block patterns and further exploration may reveal even better trade-offs between annotation quality, cost and image variety. Beyond block subimages, it can useful to focus on alternative shapes based on image content; however, non-rectangular annotation areas may pose additional challenges for workers who are familiar with conventional image annotation tasks. Another interesting direction is acquiring instance-level annotations by merging segments across block boundaries. Finally, active learning can be used to select blocks of rare classes, and workers can be assigned blocks so that annotation difficulty matches worker skill.

Notes and Acknowledgements. I appreciate the efforts of the anonymous MTurk workers who participated in our user studies.

CHAPTER 3

ANALYZING AND LEARNING FROM DEPICTIONS OF MATERIALS IN PAINTINGS

3.1 Overview

Computer vision models are often both trained on and applied to natural images. In this chapter, we examine the relationship between such visual recognition systems and the rich information available in paintings. Paintings are an interesting form of visual information – although they may depict objects and other meaningful content, paintings are not necessarily photorealistic and thus represent a different image distribution than the distribution of natural photographs. Despite this, humans can still recognize the objects and materials found in paintings. Furthermore, paintings can encode invariances of the human visual system due to their ability to convey realism without necessarily conforming to physical reality [19, 107], and it may be beneficial to train our models on paintings to learn similar human-like invariances.

Existing work has focused primarily on understanding objects in paintings; here, we explore depictions of materials found in paintings. Material perception is interesting as it relies on a complex combination of different cues: shape, glossiness, texture, color, and more [43]. Our experiments are based on a large-scale annotated database of material depictions in paintings (Materials in Paintings [162]; https://materialsinpaintings.tudelft.nl), which was the result of a long-term collaboration with colleagues at TU Delft. First, we find that visual recognition systems designed for natural images can work surprisingly well on paintings. In particular, we find that interactive segmentation tools can be

used to cleanly annotate polygonal segments within paintings, a task which is time consuming to undertake by hand. We also find that FasterRCNN, a model which has been designed for object recognition in natural scenes, can be quickly repurposed for detection of materials in paintings. Second, we show that learning from paintings can be beneficial for neural networks that are intended to be used on natural images. We find that training on paintings instead of natural images can improve the quality of learned features and we further find that a large number of paintings can be a valuable source of test data for evaluating domain adaptation algorithms.

The work in this chapter was published in the International Workshop on Fine Art Pattern Extraction and Recognition, ICPR 2020 as "Insights From A Large-Scale Database of Material Depictions In Paintings" [98], and the Materials In Paintings dataset was published in PLOS One 2021 as "Materials In Paintings (MIP): An interdisciplinary dataset for perception, art history, and computer vision" [162], with Mitchell van Zuijlen, Maarten Wijntjes, Sylvia Pont, and Kavita Bala.

3.2 Introduction

Deep learning has enabled the development of high performing recognition systems across a variety of image-based tasks [47, 54, 183]. These systems are often trained on natural photographs with applications in real world recognition like self-driving. Furthermore, applying recognition systems to large collections of images can also reveal cultural trends or give us insight into the visual patterns in the world (e.g. [113, 106, 100]). Human-created images, such as paintings, are

particularly interesting to analyze from this perspective. Artistic depictions can reveal insights into culturally relevant ideas throughout time, as well as insights into human visual perception through the realism depicted by skilled artists.

Whereas most computer vision systems focusing on digital art history are concerned with *object* recognition (e.g., [29]), it is the depiction of *space* and *materials* that visually characterized the course of art history. The depiction of space has had considerable attention in scientific literature [122, 174, 73, 129] while recently the depiction of materials has gained scientific interest [13, 175, 130, 176]. Therefore, it is interesting to investigate the interplay between deep learning systems designed for natural image analysis and the rich visual information found in paintings, especially with respect to artistic depictions of materials.

The remainder of this chapter is organized into three parts. In Section 3.3, we briefly describe the dataset that subsequent experiments are based on. In Section 3.4, we explore how deep learning systems that have primarily been developed for use on natural photographs can be used to analyze paintings. Specifically, we explore (a) segmentation and (b) detection of materials in paintings. Recognition of materials in paintings can be useful for digital art history as well as general public interest. In Section 3.5, we explore how paintings can be a useful source of data from which better recognition systems can be built. Specifically, we investigate (c) the generalizability and interpretability of classifiers trained on paintings, and we investigate (d) the role that a large-scale painting dataset can play in evaluating visual recognition models.

3.3 The Materials in Paintings (MIP) Dataset

All experiments in this chapter utilize data from the Materials in Paintings dataset (https://materialsinpaintings.tudelft.nl), a large-scale annotated dataset of physical material depictions in paintings. This dataset was the result of a longterm collaboration with our collaborators at TU Delft: Mitchell van Zuijlen, Sylvia Pont, and Maarten Wijntjes. Extensive details and analysis of this database are available in a separate manuscript [162]. For context and completeness, we summarize a few relevant details here. The dataset consists of 19K high resolution paintings downloaded from the online collections of international art galleries, which span over 500 years of art history. The galleries with corresponding number of paintings are: The Rijksmuseum (4,672), The Metropolitan Museum of Art (3,222), Nationalmuseum (3,077), Cleveland Museum of Art (2,217), National Gallery of Art (2,132), Museo Nacional del Prado (2,032), The Art Institute of Chicago (936), Mauritshuis (638), and J. Paul Getty Museum (399). The distribution of paintings by year is shown in Fig. 3.1. The dataset includes crowdsourced extreme click [123] bounding box annotations over 15 material categories, which are further delineated into 50 finegrained categories. Fig. 3.2 shows a few examples of the annotated bounding boxes available in the dataset.

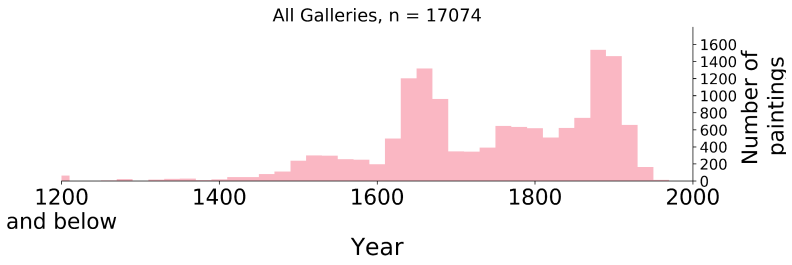


Figure 3.1: **Year Distribution of Paintings in Dataset.** Each bin equals 20 years. There are peaks in the paintings in the 1700s and 1900s. The former corresponds to the European golden ages; it is less clear what explains the latter peak.

Figure 3.2: **Examples of Annotated Bounding Boxes.** Left to Right: Liquid, Fabric, Ceramic, Metal, and Food.

3.4 Using Computer Vision to Analyze Paintings

Research in computer recognition systems have focused primarily on natural images. For example, semantic segmentation benchmarks (of objects, 'stuff', or materials) [80, 99, 40, 17, 10, 11] emphasize parsing in-the-wild photos, with applications in robotics, self-driving, and so forth. However, the analyses of paintings can also benefit from the use of visual recognition systems. Paintings can encode both cultural and perceptual biases, and being able to analyze paintings at scale can be useful for a variety of scientific disciplines including digital art history and human visual perception.

In this section, we explore the effectiveness of interactive segmentation methods (which can be used to select regions of interest in photographs for the purpose of image editing or data annotation) when applied to paintings. We also explore how well an *object* bounding box detector can be finetuned to detect *materials* depicted in unlabeled paintings, which could be used for content-based retrieval.

3.4.1 Extracting Polygon Segments with Interactive Segmentation

Polygon segmentation masks are useful for reasoning about boundary relationships between different semantic regions of an image, as well as the shape of the regions themselves. However, annotating segmentations is expensive and many modern datasets rely on expensive manual annotation methods [10, 99, 192, 17, 28]. Recent work has focused on more cost effective annotation methods (e.g. [96, 110, 12, 101]). The use of interactive segmentation methods that transform sparse user inputs into polygon masks can ease annotation difficulty. For paintings, it is unclear whether these methods (especially deep learning methods trained on natural images) would perform well. Semantic boundaries in paintings likely have a different, and more varying structure than in photos. Paintings can have ambiguous or fuzzy boundaries between objects or materials [114] which can potentially be problematic for color-based methods. This can be due to variations in artistic style which can emphasize different aspects of depictions – for example, Van Gogh uses lines and edges to create texture, but such edges could potentially appear as boundaries to a segmentation model. In this experiment, we study the difficulty of segmenting paintings and whether innovations are necessary for existing methods to perform well.

Experimental Setup

We experiment with GrabCut [139] (an image-based approach) and DEXTR [110] (a modern deep learning approach). We evaluated the performance of these methods against 4.5k high-quality human annotated segmentations from [163].

The inputs to these methods are generated from the extreme points of the regions we are interested in. We use a variant of the GrabCut initialization proposed in [123], as well as a rectangular initialization for reference. For DEXTR, we consider models pretrained on popular object datasets [99, 40] as a starting point.

Results

We found that both GrabCut and DEXTR perform quite well on paintings. Surprisingly, DEXTR transfers quite well to materials in paintings despite being trained only with natural photographs of objects. The performance of DEXTR can be further improved by finetuning on COCO with a smaller learning rate (10% of original learning rate for 1 epoch). Finetuning DEXTR on Grabcut segments or iteratively finetuning with output of DEXTR does not seem to yield further improvements. The performance is summarized in Table 3.1, and samples are visualized in Fig. 3.3.

3.4.2 Detecting Materials in Unlabeled Paintings

To allow the public to view and interact with art collections, museums and galleries provide extensive online functionality to search and navigate through the collections. Currently, to our knowledge, no online collections allows online visitors to query the collection for depicted *materials* within painting, which can be of interest to the public. Furthermore, depiction of materials plays a crucial role in characterizing art history. Detecting materials within novel paintings will be particularly beneficial to digital art historians who study materials such as stone [7, 36] or skin [14, 85]. Having access to specific materials can also digital

Segmentation mean IOU (%)

Grabcut Rectangle	Grabcut Extr	DEXTR Pascal-SBD	DEXTR COCO	DEXTR Finetuned
44.1	72.4	74.3	76.4	78.4

DEXTR Finetuned IOU By Class (%)

Animal	Ceramic	Fabric	Flora	Food
76.9	86.8	79.1	77.0	87.5
Gem	Glass	Ground	Liquid	Metal
74.4	83.2	69.6	73.0	75.5
Paper	Skin	Sky	Stone	Wood
86.1	78.9	78.5	81.7	67.4

Table 3.1: **Segmentation Performance**. Grabcut Extr is based on [123] with small modifications: (a) minimum cost boundary is computed with the negative log probability of a pixel belonging to an edge; (b) in addition to clamping the morphological skeleton, the extreme points centroid and extreme points are clamped; (c) GC is computed directly on the RGB image. DEXTR [110] is pretrained on Pascal-SBD and COCO. Note that Pascal-SBD and COCO are natural image datasets of objects, but DEXTR transfers surprisingly well across both visual domain (paintings vs. photos) and annotation categories (materials vs. objects).

art historians to compare these depictions directly with respect to painting style or technique. We experiment with automatic bounding box detection to ease access to material depictions in unlabeled collections.

Experimental Setup

We train a FasterRCNN [138] bounding box detector to localize and label material boxes with on 90% of annotated paintings in the dataset, and evaluate on the remaining 10%. Default COCO hyperparameters from [178] are used. Given the non-spatially-exhaustive nature of the annotations, many detected bounding boxes will not be matched against labeled ground truth boxes. However, the dataset is exhaustively annotated at an image level, and therefore, we report image-level accuracies. This can be interpreted as the accuracy of the model in

Figure 3.3: **Extreme Click Segmentations.** Left to right: Original Image, Ground Truth Segment, Grabcut Extr Segment, DEXTR COCO Segment. Both Grabcut and DEXTR use extreme points as input. For evaluation, the extreme points are generated synthetically from the ground truth segments. In practice, extreme clicks can be crowdsourced. Bottom-right corner shows the IOU for each segmentation.

tagging each image with the types of materials present. The validity of each localized box can be further quantified through a user study, but we did not perform this study at this time.

Results

Table 3.2 shows the performance. We found that the FasterRCNN model is able to accurately detect materials in paintings by finetuning on the annotated bounding boxes directly without any changes to the network architecture or

training hyperparameters. It is certainly promising to see that an algorithm designed for object localization in natural images can be readily applied to material localization in paintings. A qualitative sample of detected bounding boxes is given in Fig. 3.4. To improve the spatial-specificity of the detected materials, it can be interesting to train an instance detector like MaskRCNN on segments extracted using methods discussed in the previous section. It would also be useful to combine material recognition with conventional object-based detection to extract complementary forms of information that improve the ability for users to filter data by their specific needs.

Class Accuracy (%) (Mean = 83.3%)

Animal	Ceramic	Fabric	Flora	Food
85.6	92.7	66.0	85.0	94.9
Gem	Glass	Ground	Liquid	Metal
88.4	91.3	86.5	86.4	70.7
Paper	Skin	Sky	Stone	Wood
92.4	70.2	89.4	74.8	74.9

Table 3.2: **Image-level Detection Accuracy.** Bounding boxes are detected with FasterRCNN trained on paintings. Because the dataset is not exhaustively annotated spatially, image-level accuracy is reported instead of box precision and recall. Overall, images are tagged with the correct materials with high accuracy.

Figure 3.4: **Detected materials in Unlabeled Paintings.** Automatically detecting materials can be useful for content retrieval and for filtering online galleries by viewer interests.

3.5 Using Paintings to Build Better Recognition Systems

In recent work for machine perception systems, art has been used in various ways. Models that learn to convert photographs into painting-like or sketch-like images have been studied extensively for their application as a tool for digital artists [71]. Recent work has shown that such neural style transfer algorithms can also produce images that are useful for training robust neural networks [49]. Artworks have also been used directly to evaluate the robustness of neural networks under "domain shifts" in which a model trained to recognize objects from photographs are shown artistic depictions of such objects instead [91, 127].

We use the MIP dataset of material depictions in paintings to explore two directions. First, we hypothesize that the perceptually focused depictions of artists can allow neural networks to learn better cues for classification. We find that learning from paintings can improve the interpretability of the cues it uses for its predictions. In a second experiment, we investigate the utility of the MIP dataset as a benchmark for computer vision models under domain shifts for material classification; this dataset also features more samples than is available than is typical in existing domain adaptation benchmark datasets. We find that existing domain adaptation algorithms can fail to behave as expected in this setting.

3.5.1 Learning Robust Cues for Finegrained Fabric Classification

The task of distinguishing between images of different semantic content is a standard recognition task for computer vision systems. Increasing attention is being given to "fine-grained" classification where a model is tasked with distinguishing images of the same broad category (e.g. distinguishing different species of birds or different types of flora [172, 164, 161]). Fine-grained classification is particularly challenging for deep learning systems. Such a task depends on recognizing specific attributes for each finegrained class; in comparison, classifiers can perform well on coarse-grained classification by relying on context alone. We hypothesize that the painted depictions of materials can be beneficial for this task. Since some artistic depictions focus on salient cues for perception through perceptual shortcuts [108, 19, 35], it is possible that a network trained on such artwork is able to learn a more robust feature representation by focusing on these cues.

Experimental Setup

We experiment with the task classifying cotton/wool versus silk/satin. The latter can be recognized through local cues such as highlights on the cloth; such cues are carefully placed by artists in paintings. To understand whether artistic depictions of fabric allow a neural network to learn better features for classification, we train a model with either photographs or paintings. High resolution photographs of cotton/wool and silk/satin fabric and clothing (dresses, shirts) are downloaded and manually filtered from publicly available photos licensed under the Creative

Commons from Flickr. In total, we downloaded roughly 1K photos. We sample cotton/wool and silk/satin samples from our dataset to form a corresponding dataset of 1K paintings.

Generalizability of Classifiers. Does training with paintings improve the generalizability of classifiers? To test cross-domain generalization, we test the classifier on types of images that it has not seen before. A classifier that has learned more robust features will perform better on this task than one that has learned to classify images based on more spurious correlations. We test the trained classifiers on both photographs and paintings.

Interpretability of Classifier Cues. Are the cues used by each classifier interpretable to humans? We produce evidence heatmaps with GradCAM [142] from the feature maps in the network before the fully connected classification layer. We extract high resolution feature maps from images of size 1024×1024 (for a feature map of size 32×32). The heatmaps produced by GradCAM show which regions of an image the classifier uses as evidence for a specific class. If a classifier has learned a good representation, the evidence that it uses should be more interpretable for humans. For both models, we compute heatmaps for test images corresponding to their ground truth label. We conduct a user study on Amazon Mechanical Turk to find which heatmaps are preferred by humans. Users are shown images with regions corresponding to heatmap values that are above 1.5 standard deviations above the mean. Fig 3.5 illustrates an example. Our user study resulted in responses from 85 participants, 57 of which were analyzed after quality control. For quality control, we only kept results from participants who spent over 1 second on average per task item.

Figure 3.5: **Classifier Cues.** Left to Right: Original Image, Masked Image (Painting Classifier), and Masked Image (Photo Classifier). The unmasked regions represent evidence used by the classifiers for predicting "silk/satin" in this particular image.

Results

We find that the classifier trained with paintings exhibits better cross-domain generalization, and uses cues that humans prefer over the photo classifier. This suggests that paintings can improve the robustness of classifiers for this task of fabric classification.

Generalizability of Classifiers. In Table 3.3, the performance of the two classifiers are summarized. We find that both classifiers perform similarly well on the domain they are trained on. However, when the classifiers are tested on cross-domain data, we find that the painting-trained classifier performs better than the photo-trained classifier. This suggests that the classifier trained on paintings has learned a more generalizable feature representation for this task.

Interpretability of Classifier Cues. Overall, we find that the classifier trained on paintings uses evidence that is better aligned with evidence preferred by humans (Table 3.4 and Fig. 3.6). Due to domain shifts when applying classifiers to out-of-domain images, we would expect the cues selected by the painting classifier to be preferable on paintings, and the cues selected by the photo classifier to be

preferable on photos. Interestingly, this does not hold for photos of satin/silk (column 2 of Table 3.4) – we find that users equally prefer the evidence selected by the painting classifier to the evidence selected by the photo classifier. This suggests that either (a) the painting classifier has learned the "correct" human-interpretable cues for recognizing satin/silk, or (b) that the photo classifier has learned to classify satin/silk based on some spurious contextual signals. We asked users to elucidate their reasoning when choosing which set of cues they preferred. In general, users noted that they preferred the network which picks out regions containing the target class. Therefore, it seems that the network trained on paintings has learned better to distinguish fabric through the actual presence of such fabrics in the image over other contextual signals.

Taken together, our results provide evidence that a classifier trained on paintings can be more robust than a classifier trained on photographs. It would be interesting to explore this further. A limitation of this study is the relatively small number of data samples, and very limited number of material types (two: cotton/wool and silk/satin) that we explored. Are there other materials or objects which deep neural networks can learn to recognize better from paintings than photographs?

	Photo → Photo	Painting→ Painting
MEAN F1 Score	79.6%	80.5%
	Photo → Painting	Painting→ Photo
MEAN F1 Score	49.5%	57.8%

Table 3.3: **Classifier Generalization**. Classifiers are trained to distinguish cotton/wool from silk/satin. One classifier is trained on photographs and another classifier is trained on paintings. Both classifiers perform similarly well on images of the same type they were trained on, but the classifier trained on paintings performs better on photographs than vice versa. This suggests that the features learned from paintings are more generalizable for this task on this set of data.

	Cotton/Wool Photos	Silk/Satin Photos	Cotton/Wool Paintings	Silk/Satin Paintings	MEAN
Photo Classif. Preferred	64.7 ± 3.5%	48.9 ± 3.1%	26.8 ± 2.5%	39.1 ± 2.1%	44.9 ± 1.9%
Painting Classif. Preferred	35.3 ± 3.5%	51.1 ± 3.1%	73.2 ± 2.5%	60.9 ± 2.1%	55.1 ± 1.9%

Table 3.4: **Human Agreement with Classifier Cues.** On average, humans prefer the cues used by the painting-trained classifier to make its predictions over the cues used by the photo-trained classifier. Interestingly, the human judgements also indicate that the painting-trained classifier uses cues that are just as good to the cues used by the photo-trained classifier for silk/satin photos despite never seeing a silk/satin photo during training (column 2). A pictorial representation of the results is given in Fig. 3.6.

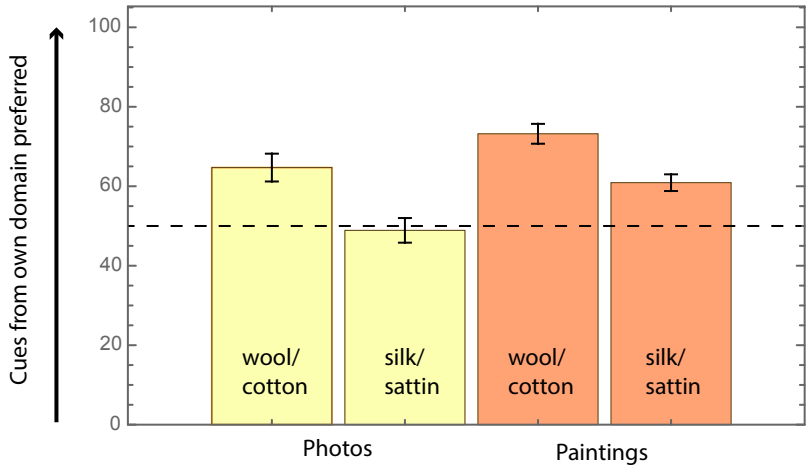


Figure 3.6: **Human Agreement with Classifier Cues.** Pictorial representation of user study results from Table 3.4. The y-axis represents how often humans prefer the cues from a classifier trained on the same domain as the test images. It is clear that humans prefer the painting classifier for paintings more than they prefer the photo classifier for photos. Interestingly, the painting and photo classifiers are equally preferred for silk/satin photos despite the painting classifier never seeing a photo during training (bar 2).

3.5.2 Benchmarking Unsupervised Domain Adaptation

In unsupervised domain adaptation (UDA), models are trained on a 'source' dataset with annotated labels as well as an unlabeled 'target' dataset. The goal is to train a model which performs well on unseen target dataset samples. Existing

domain adaptation benchmark datasets for classification focus primarily on object recognition and tend to be limited in number of data samples, with most class categories containing on the order of 1000 samples or fewer (for example, refer to Table 1 of [127]). In contrast, the dataset we use here has the unique properties of (a) focusing primarily on material classification and (b) containing on the order of 10-30K for 9 of the 15 annotated classes (e.g. fabric, wood), with the remainder in the range of 2K-5K (e.g. ground). This positions this data as a valuable addition for benchmarking for UDA algorithms.

Experimental Setup

For this study, we focus on a family of domain adaptation algorithms which aim to explicitly minimize feature discrepancy across the source and target domains. Existing work has shown that class-conditional UDA in which labels are estimated for target domain samples during training can be better than class-agnostic UDA where adaptation is performed without using any estimated label information at all. We choose CDD [72] and MMD [103, 159] as representative methods for class-conditional and class-agnostic discrepancy minimization. CDD estimates class labels for the target domain via clustering; for details, refer to the original paper. All methods are trained with default settings from publicly available source code for CDD, which includes the use of domain batch normalization [21]. We selected 10 material categories: ceramic, fabric, foliage, glass, liquid, metal, paper, skin, stone, and wood. For our painting dataset, we sampled as-class-balanced-as-possible from these classes to form a dataset with 10K samples and a dataset with 60K samples. A corresponding photograph dataset is constructed from Opensurfaces/MINC/COCO with 10K and 60K samples as

well.

Results

We find that the studied domain adaptation algorithms can indeed behave differently than would be expected from results on existing benchmark datasets. This is could due to more data being available or a more difficult domain shift than conventional adaptation benchmark datasets.

Effect of Dataset Size. Results are summarized in Table 3.5. With the conventional 1K samples per class, we confirm domain adaptation yields gains over source-only as found on existing benchmark datasets. In contrast to results on existing benchmarks however, we find that class-conditional adaptation does not necessarily outperform class-agnostic adaptation. We hypothesize this occurs due to failures in target label estimation for the class-conditional case – we discuss this further below. Next, with 6K samples per class (which is $6\times$ more data samples per class than conventional UDA benchmarks), we find that source-only (i.e., no adaptation) performs very competitively. In fact, source-only strictly outperforms adaptation for Painting to Photo transfer in this data regime! This result suggests that domain adaptation is useful in lower data regimes, but source-only is a competitive alternative when more data is available. Whether this is due to the classification task itself, quirks specific to this dataset, or something else is an interesting direction to explore. We leave a deeper exploration of the source of negative transfer when MMD or CDD are applied to this dataset as future work.

Effect of Class Label Estimation. As found above, class-conditional adaptation can underperform class-agnostic adaptation despite utilizing more information.

As class-conditional adaptation depends on estimated target labels, large domain shifts that hamper label estimation can harm adaptation. To confirm this, we consider two experiments: CDD with intraclass discrepancy minimization only (instead of both intraclass minimization and interclass maximization), and CDD with ground truth labels (i.e., perfect label estimation). Results are in Table 3.6. In both cases, we see performance improves. In the case where perfect label estimation is assumed, then CDD does outperform intraCDD and MMD as found on existing datasets. Therefore, estimating class labels for domain adaptation is useful in practice, but only if the labels are estimated sufficiently well.

ResNet18 *1K imgs/class per domain*	Photo → Painting	Painting → Photo
Source-Only	35.2% (—)	46.9% (—)
MMD	**46.1%** (+10.9%)	56.4% (+9.5%)
CDD	40.5% (+5.3%)	**57.4%** (+10.5%)
ResNet18 *6K imgs/class per domain*	Photo → Painting	Painting → Photo
Source-Only	38.7% (—)	**53.6%** (—)
MMD	**43.5%** (+4.8%)	51.5% (–2.1%)
CDD	35.6% (–3.1%)	49.4% (–4.2%)

Table 3.5: **Effect of Dataset Size.** UDA from photo (source) to painting (target) and painting (source) to photo (target). Source-only refers to a reference baseline where no adaptation is used. The gap between source-only and UDA decreases as data samples increases from 1K images per class to 6K images per class. Furthermore, in contrast to behavior found on existing benchmark datasets, the class-conditional method of CDD does not necessarily outperform the class-agnostic counterpart MMD.

3.6 Discussion and Future Work

In this chapter, we explored how modern deep learning tools developed for natural images can be used to analyze paintings, and in turn, how paintings can

ResNet18 *1K imgs/class per domain*	Photo → Painting	Painting → Photo
Source-Only	35.2% (—)	46.9% (—)
MMD	**46.1%** (+10.9%)	56.4% (+9.5%)
IntraCDD	44.4% (+9.2%)	**58.5%** (+11.6%)
CDD	40.5% (+5.3%)	57.4% (+10.5%)
IntraCDD w/ GT labels	57.6% (+22.4%)	64.4% (+17.5%)
CDD w/ GT labels	**61.6%** (+26.4%)	**72.5%** (+25.6%)

Table 3.6: **Effect of Class Label Estimation.** Reducing the reliance class label estimation improves class-conditional UDA when label estimation for target data is poor. MMD does not require class label estimation, and so its performance is relatively good here. Due to poor label estimation, we find that IntraCDD (which considers only intraclass discrepancy) outperforms CDD (which considers both intraclass and interclass discrepancy) as IntraCDD relies less on accurately estimated class labels. (green) Assuming perfect class label estimation using ground truth (GT) labels, CDD recovers performance gains over intraCDD and MMD.

be used to improve deep learning systems in a series of experiments. Paintings represent an interesting distribution of images. Despite not necessarily being photorealistic, paintings still convey meaningful imagery that evoke similar feelings as natural photos. Paintings are crafted by humans for humans, and understanding paintings can deepen our understanding of cultures and human perception.

As most computer vision systems are built for natural images, labeled datasets for paintings do not exist at the same scale as those for photos. Our results in this chapter demonstrate that interactive annotation tools can work surprisingly well on paintings, suggesting that these tools can be used for easing the construction of labeled painting datasets. We also showed that labeled paintings can be used to finetune object detectors into material detectors. Automated methods for retrieving depictions of specific materials can be very valuable for digital art historians, who have studied specific styles or techniques with respect to depicted

content in paintings. These experiments represent a step towards understanding how to create better tools for managing painting data. More annotations and automated content analysis of paintings can prove to be valuable for many disciplines, including those beyond computer vision and deep learning.

On the other hand, deep learning systems may benefit from learning from paintings as well. As paintings emphasize useful perceptual cues so that humans can understand the depictions in paintings, training a computer vision model on paintings may to allow models to learn such cues for recognition. Our experiment with glossy versus rough fabric classification showed that models trained on paintings were indeed able to focus on more interpretable cues. While this study was limited to a small specific classification task (wool/cotton versus silk/satin), it would be interesting to explore further whether models trained on paintings can learn robust representations for more general settings. We explore this direction further in the next chapter. Finally, extensive existing efforts have been made in creating algorithms for adapting models to new image distributions. We found that two existing unsupervised domain adaptation algorithms can struggle to perform well when used to adapt a model trained for material recognition from photos to paintings or vice versa. By evaluating models and algorithms in their ability to enable generalization to new domains during testing, deep learning systems and training algorithms can be implicitly guided towards more robust design paradigms.

Notes and Acknowledgements. I thank Mitchell for his efforts in leading the collection of the Materials in Paintings dataset, upon which this work was performed.

CHAPTER 4

LEARNING ROBUST NATURAL IMAGE RECOGNITION FROM PAINTINGS

4.1 Overview

A common method to improve a model's robustness to distribution shifts is through data augmentation. These are image transformations that are applied to existing training images that preserve the semantic content of each image. Data augmentations encourage models to learn desired invariances, such as invariance to horizontal flipping or small changes in color. Beyond simple geometric or photometric transformations, recent work has shown that style transfer can be used as a form of data augmentation to encourage invariance to textures [49]. Style transfer is a technique which aims to create painting-like images from photographs [71]; often, these methods rely on operations that manifest as changes in texture throughout the original photograph. However, a stylized photograph is not quite the same as an artist-created painting. Artists depict perceptually meaningful cues in paintings so that humans can recognize salient components in scenes, an emphasis which is not enforced in style transfer. As found in the previous chapter, learning from paintings may allow models to better grasp important cues for generalizable recognition.

Therefore, in this chapter, we study how style transfer and paintings differ in their impact on model robustness. First, we investigate the role of paintings as style images for stylization-based data augmentation. Arbitrary style transfer algorithms transform a photo by transferring the style from some source style image; conventionally, this style image is selected from a set of paintings. However,

we find that style transfer functions well even without paintings as style images, suggesting that the power of style transfer as data augmentation is orthogonal to the style found in actual paintings. Second, we show that learning from real paintings as a form of perceptually-grounded implicit data augmentation can improve model robustness. That is, each painting can be considered an augmented version of some photograph of a scene, even though the photograph and/or scene may not necessarily exist. Finally, we investigate the invariances learned from stylization and from paintings, and show that models learn different invariances from these differing forms of data. Our results provide insights into how stylization improves model robustness, and provide evidence that artist-created paintings can be a valuable source of data for improving model robustness to naturally-occuring distribution shifts.

The work in this chapter was published in CVPR 2021 as "What Can Style Transfer and Paintings Do For Model Robustess?" [97] with Mitchell van Zuijlen, Sylvia Pont, Maarten Wijntjes, and Kavita Bala.

4.2 Introduction

Model robustness can be defined as the capability of a model to generalize to unseen image distributions. These can be the result of real-world effects, like weather and camera noise [60], adversarial noise [102], or distribution shifts due to differences in environments in which the images are captured. The performance of standard recognition models can degrade drastically in these settings, but robust models are critical for applications such as self-driving or medical diagnostics.

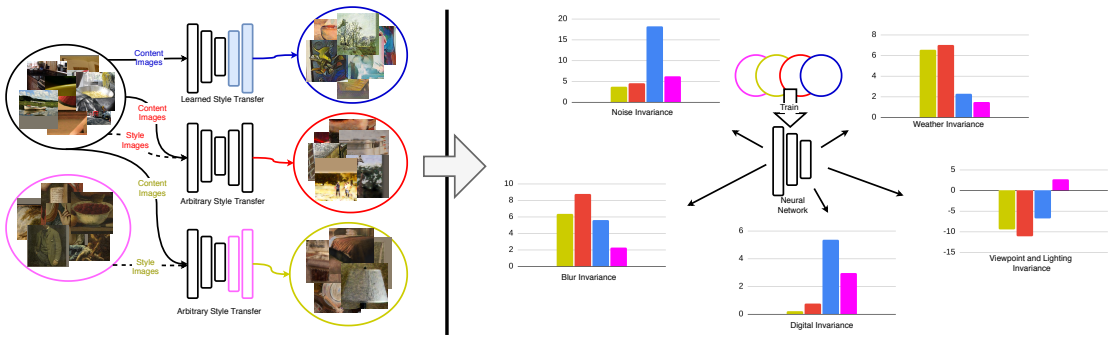


Figure 4.1: **What invariances are learned from real and fake paintings?** Left: Natural photographs (black), paintings (magenta), and stylized photographs (olive/red/blue) from the Materials dataset (Section 4.4.2), Right: Relative robustness to various types of transformations for models trained with different sets of images with respect to a model trained on only natural photos. Stylization algorithms can transform photographs into painting-like images, but it is not clear that models will learn the same invariances from these images. This chapter explores a series of hypotheses to understand the different ways in which style transfer and paintings improve model robustness.

A common strategy is to improve generalization through data augmentation [187, 34, 61, 102]. Conventional data augmentation applies transformations to encourage invariance to heuristic rules (e.g., flipping for invariance to image mirroring). Recent work has found that image stylization can encourage models to learn invariance to texture [49]. While style transfer has focused on visual fidelity [71], we argue that current style transfer models do not yet fully capture the essence of artistic paintings. For example, a family of style transfer algorithms act by manipulating feature distributions to create a stylized photo which holistically mimics a painting [93] – in effect, mid-level textures are manipulated in the stylized photo. However, paintings are more than a style filter applied to a photo. An artist can choose lighting, contours, and scene context to convey realism in important scene regions while foregoing perceptual details less important areas. This artistic manipulation can affect our perceptual understanding of the scene.

In this chapter, we explore a series of hypotheses to understand how style

transfer and paintings impact model robustness. Fig. 4.1 illustrates that various types of images can differently affect model robustness. First, we examine how style images play a role in stylization-based data augmentation in Section 4.5. Second, we investigate the role of paintings as a form of training data, and contrast it to other artforms such as sketches in Section 4.6. Finally, we probe models to empirically understand their learned invariances, and discuss how style transfer and artistic paintings can contribute to robust natural image recognition models in Section 4.7. Our contributions are:

- We demonstrate that arbitrary style transfer can be used as effective data augmentation even without painting style images. We attribute their effectiveness to the diversity of style images rather than artistic style.
- We argue that paintings can be considered a form of perceptual data augmentation, and demonstrate that it can improve model robustness. We contrast paintings with other forms of art such as sketches.
- We explore the invariances learned from arbitrary style transfer, learned artistic style transfer, and paintings. We find that models do not learn the same invariances from stylized photos and paintings, and show that the learned invariances are complementary.

4.3 Related Work and Background

Model Robustness. Recent work in robustness for CNNs has focused on both adversarial robustness [18] as well as robustness to real-world transformations [60, 50]. This view of model robustness is human-centric, where the settings considered are those where the human visual system has been shown to be

robust (e.g., [144, 50, 49]), rather than enforcing model robustness under arbitrary settings. A related line of work is in domain generalization, where the task is to generalize to unseen domains, (e.g., [51, 112, 92, 91]), by learning a shared representation on a set of seen domains. While a common justification for domain generalization is model robustness, domain generalization is subtly different. Domain generalization algorithms assume the target domain is unspecified, and do not rely on domain-specific signals at inference time. However, robust natural image recognition can benefit from learning from natural images directly.

Data Augmentation. Data augmentations are transformations applied to images to enforce useful model invariances. Beyond basic transformations like flipping, recent work in data augmentation has focused on more complex augmentations such as image occlusion [34], class-mixing [187], and compositions of transformations [61]. Data-driven augmentations such as adversarial or stylization transformations [170, 102, 68] can also be used to model nuanced invariances.

Style Transfer. Style transfer aims to transform photos into painting-like images by transferring artistic styles. While increasing attention has been given to arbitrary style transfer (e.g., [68, 150, 146, 155, 167]) which aims to efficiently transfer unseen styles, artist-specific style transfer models (e.g., [140, 77]) are typically able to better capture nuances from a collections of images. Beyond its role as a tool for artistic creation, stylization has also been used as a form of data augmentation to enforce invariances to textures [49], as well as regularization for tasks such as human re-identification [70].

4.4 Preliminaries

4.4.1 Evaluating Robustness

We evaluate robustness to common image corruptions and distribution shifts from the training distribution. These settings serve as a proxy for real-world robustness. Furthermore, the behavior of models on these scenarios gives us insight into the invariances learned – for example, a model which is robust to noise has likely learned to be more invariant to (i.e., to rely little on) high-frequency signals in an image. All experiments use an ImageNet-pretrained ResNet18 architecture, and results are averaged over three independent runs. For complete training details and experiments with alternative architectures, please refer to the appendix.

Common Image Corruptions. Common image corruptions are inspired by transformations that can be encountered in real-world settings [60]. There are 15 corruptions which span 4 broad categories (noise, blur, weather, and digital) with 5 severity levels per corruption. We use the released code to corrupt our test images. Figure 4.2 illustrates these corruptions. For each corruption, we compute the mean accuracy over each severity, and then compute the mean over each set of broad corruption categories C. Given a model Θ, the mean corruption accuracy is:

$$\text{Acc}_{\text{Mean}}(\Theta) = \frac{1}{4}\sum_{C} \text{Acc}_{C}(\Theta) \tag{4.1}$$

$$\text{where } \text{Acc}_{C}(\Theta) = \frac{1}{5n_C}\sum_{corr \in C}\sum_{s=1}^{5} \text{Acc}(\Theta, \mathcal{D}_{corr,s})$$

$\mathcal{D}_{corr,s}$ denotes the test dataset of images transformed by corruption $corr$ with severity s.

Small Distribution Shifts. Out-of-distribution photographs will be used to evaluate robustness to small domain shifts not unlike the domain shifts that models must overcome when they are used in different real world environments. For the PACS dataset, we use a subset of the YFCC100M dataset [158] as the out-of-distribution test set. This subset is curated by downloading 100 images per class and then manually filtering to remove irrelevant retrievals down to 50 images per class. This test set is released for reproducibility. For the Materials dataset, we use the Flickr Material Database (FMD) [144] as the out-of-distribution test set.

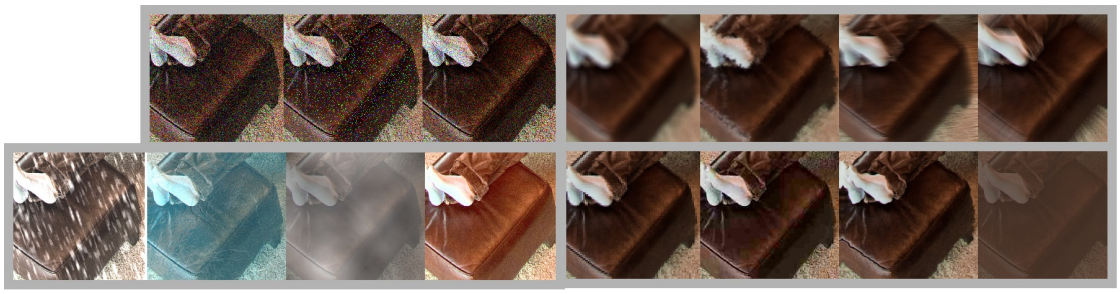


Figure 4.2: **Image Corruptions.** Top-Left to Bottom-Right: Noise(×3), Blur(×4), Weather(×4), Digital(×4).

4.4.2 Datasets

We select datasets which contain both photographs and paintings, and conduct experiments across two recognition tasks (object classification and material classification).

Object Classification. We use the PACS dataset [91] which consists of 10K images across 7 categories and 4 domains (photographs, paintings, cartoons, and sketches).

Material Classification. We construct a dataset from existing large-scale photograph datasets [10, 11, 17], and a large-scale painting dataset with material annotations [162]. We will refer to this dataset as 'Materials'. This dataset consists of 120K images across 10 categories and 2 domains (photographs and paintings). See appendix for details. [49] found that stylization-based augmentation can reduce bias towards textures, but material recognition relies on texture understanding [2]. As such, it is interesting to explore whether stylization can improve robustness for this task.

4.4.3 Notation

Some common notation used throughout is given here. Let $\mathcal{D}_n$ be a set of natural photographs and $\mathcal{D}_p$ be a set of paintings. For each image x, its class label is denoted by y_x. Finally, let $l(\hat{y}, y)$ denote the cross entropy loss.

4.5 Style Transfer as Data Augmentation

Style transfer aims to transform the style of an image into the style of another set of images [71]. There is evidence [49] that training on stylized images [68] can improve object recognition on ImageNet by encouraging networks to focus more on shape than texture. In this view, we can consider style transfer as a form of data augmentation. Style transfer is often applied with painting style images from datasets such as Wikiart [157, 177]. In its role as a tool to mimic artistic creation, this is certainly appropriate. However, in its role as a form of data augmentation, it is not strictly necessary for the style images to be paintings. Indeed, arbitrary stylization methods can be applied to any pair of content and style images (hence 'arbitrary'). Although work such as [49] utilize style transfer in the conventional manner with painting styles, it's important to ask whether models can learn robust invariances from *photo* style images alone.

To answer this question in a general way, we experiment with three representative deep-learning based arbitrary style transfer methods. Each of these methods act in deep feature space, but follow a different paradigm: AdaIN [68] transfers style by matching the mean and standard deviation of features, ET-Net [150] iteratively refines a stylized image by computing residual error maps, and TPFR [155] transfers style by recombining features in the content image to match those of the style image. We explore the following:

- **Hypothesis H1.** Painting styles are necessary for stylization-based augmentation to improve robustness.
- **Hypothesis H2.** Style image diversity is important.

4.5.1 Are Painting Style Images Necessary?

We experiment with: (a) a network trained with photos plus photos stylized by paintings and (b) a network trained with photos plus photos stylized by other photos. We will refer to (b) as "intradomain stylization" as photos are being stylized by other photos from within the same domain. For reference, we also consider (c) a network trained with photos alone (no stylization). Specifically, let $\phi(x, x_s)$ be an arbitrary stylization algorithm which stylizes content image x with style image x_s. For a network Θ, the objectives are given by:

$$\begin{aligned}
&\text{(a)} \min_{\Theta} \mathbb{E}_{x,x_s \sim \mathcal{D}_n, \mathcal{D}_p} \big[\frac{1}{2}\big(l(\Theta(x), y_x) + l(\Theta(\phi(x, x_s)), y_x)\big)\big] \\
&\text{(b)} \min_{\Theta} \mathbb{E}_{x,x_s \sim \mathcal{D}_n, \mathcal{D}_n} \big[\frac{1}{2}\big(l(\Theta(x), y_x) + l(\Theta(\phi(x, x_s)), y_x)\big)\big] \\
&\text{(c)} \min_{\Theta} \mathbb{E}_{x \sim \mathcal{D}_n} \big[l(\Theta(x), y_x)\big]
\end{aligned} \tag{4.2}$$

In practice, we approximate the objectives by sampling x_s once for each x instead of minimizing over all independent combinations of x and x_s.

The results are shown in Fig. 4.3. Across both PACS and Materials, we find that intradomain stylization significantly improves robustness over the photo-only baseline. With a large dataset (Materials), we find that intradomain stylization can meet or even exceed the performance of conventional painting-based stylization. Thus, in contrast to common practice, stylization-based data augmentation does not need painting style images. This finding is also supported by recent work which shows that online feature moment matching across different training images is an effective form of data augmentation [90] (which we can frame as roughly equivalent to intradomain stylization with AdaIN),

and work which shows stylization with images from non-painting domains (including intradomain stylization) can be useful for domain generalization [149]. We have shown explicitly here that intradomain stylization can replace painting stylization for robust natural image recognition when enough data is available.

Answer to H1: *Intradomain stylization can improve network robustness to an extent that is comparable to painting stylization when there is sufficient data – that is, paintings do not play a unique role when arbitrary style transfer is used as data augmentation.*

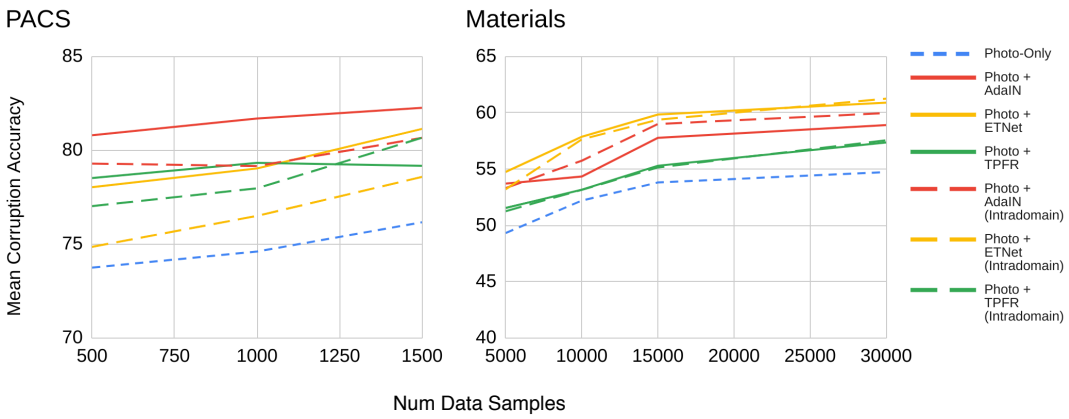


Figure 4.3: **Stylization: Painting vs Photo Styles.** Left: PACS, Right: Materials. In general, intradomain stylization (red/green/yellow) improves robustness over no stylization (blue). Further, when sufficient data is available (Materials), intradomain stylization (dashed lines) results in similar robustness gains to conventional painting stylization (solid lines). This means that paintings are not uniquely responsible for robustness gains from stylization.

4.5.2 The Role of Style Diversity

The finding that intradomain stylization can be comparable to painting stylization leads to the hypothesis that it is the diversity in image statistics between style and content images that plays a key role. For example, consider AdaIN – the extent to which images are transformed by stylization depends on the magnitude

of the difference in feature distribution moments between the content image and the style image. This is why intradomain stylization is comparable to painting stylization on a large dataset like Materials.

We test this hypothesis by restricting the style photo for intradomain stylization to be drawn from images that share the same class label as the content image. With this restriction, the style images are likely to be more similar to the content image given that they share similar semantic content. Let $\mathcal{D}_n^y$ be the subset of natural photographs with class label y. Then, the objective is given by:

$$\min_{\Theta} \mathbb{E}_{x\sim\mathcal{D}_n}\left[\mathbb{E}_{x_s\sim\mathcal{D}_n^{y_x}}\left[\frac{1}{2}\big(l(\Theta(x),y_x)+l(\Theta(\phi(x,x_s)),y_x)\big)\right]\right] \tag{4.3}$$

In general, we find that this restriction does indeed reduce the effectiveness of intradomain stylization (Fig. 4.4). As an exception, TPFR does not appear to rely heavily on the choice of style images. This can be explained by the adversarial loss used in TPFR – the decoder is trained explicitly to fool a style discriminator that discriminates between stylized images and real paintings during training. Therefore, it is possible that the decoder is encoding painting-like style signals regardless of the style image used. This also suggests that a style transfer algorithm which explicitly transfers painting styles can be useful instead of relying on a diverse style dataset during training (we explore this in Section 4.7). In general, biases in stylization models can contribute to improved robustness independently of style images.

Answer to H2: *Access to style images which are diverse with respect to content images is key for stylization-based augmentation.* Against conventional wisdom, style images need not contain statistics that manifest as visible textures or artistic style per se.

As long as each style image is sufficiently different from its corresponding content image, it will suffice. "Sufficiently different" means "depicting different semantic content" in our analysis here. Interestingly, we found that style differences measured by the Gram matrix distance between a stylized image and its original counterpart do not correlate with robustness (see appendix) – further analysis is left for future work.

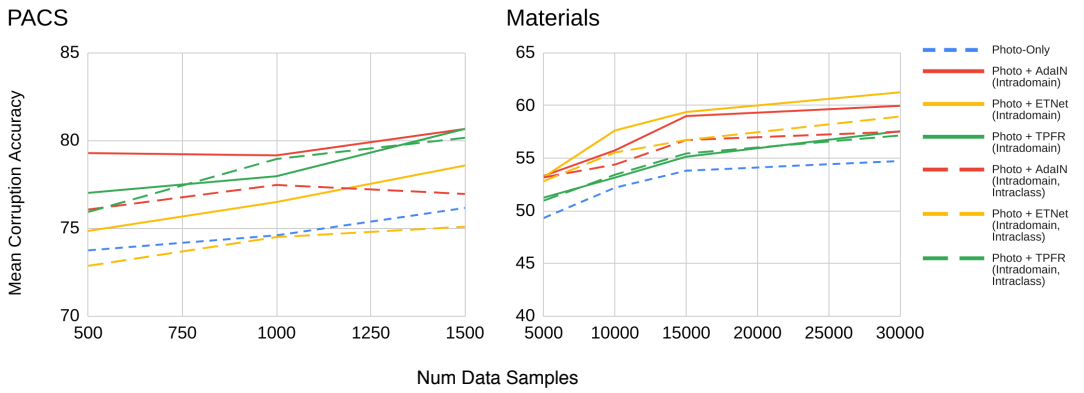


Figure 4.4: **Stylization: Unrestricted vs Intraclass Styles.** Left: PACS, Right: Materials. Across both datasets, restricting style images to the class as content images (dashed lines) results in smaller robustness gains compared to unrestricted stylization (solid lines). This reduction in robustness is explained by the reduction in diversity between content images and style images.

4.6 Paintings as Perceptual Data Augmentation

In Section 4.5, we found that stylization as data augmentation works well as long as the set of style images are diverse. This diversity does not necessarily depend on the image statistics found specifically in paintings. If sufficiently diverse mid-level statistics is found by stylization with photos, then perhaps photos can fulfill the role of paintings entirely.

Instead, we argue that paintings are more than just a set of mid-level style

features overlaid on top of a photograph. Our key insight is that perceptually realistic paintings can be considered a form of 'perceptual' data augmentation. Unconstrained by physical reality, artists are free to depict varying level of perceptual realism [19]. Paintings are *perceptually* realistic in regions where the artist has deemed viewer attention should be focused. For example, a painting of a giraffe might include perceptually relevant details on the giraffe itself while the background is depicted in an less realistc and more abstract manner. In a collection of paintings, important cues for objects or materials of interest are depicted frequently in a perceptually sound manner while unimportant details are abstracted away.

Even so, the domain shift between paintings and photos can be problematic, and it is likely that models trained on paintings will fail to perform well on photos if domain shift is not accounted for. Furthermore, many of the arguments made for paintings above can also apply to other artforms, and it is interesting to consider alternatives. We explore the following:

- **Hypothesis H3.** (a) Learning from paintings improves natural image robustness after accounting for domain shift, and (b) this improvement is greater than that found from photos alone.
- **Hypothesis H4.** Other artforms can encode similar invariances to paintings.

4.6.1 Learning Robust Natural Image Recognition From Paintings

A classifier trained directly on both photos and paintings is required to learn boundaries that satisfy both of these domains. Consequently, the accuracy on photographs can suffer. Since our goal is to train a robust model for natural image classification, we alleviate this by considering two alternatives: (a) a shared feature extractor with multiple domain-specific classifiers (multitask learning) or (b) a photo-only classifier that is finetuned after shared feature learning. For reference, we also consider the default option of training (c) a joint classifier on both photos and paintings. Specifically, let Θ_f be a feature extractor (i.e., ResNet18 without the final fully connected layer). Let η be a linear classifier (i.e., a fully connected layer). Then the objective for (a) is given by:

$$\min_{\Theta_f,\eta_n,\eta_p} \mathbb{E}_{x_n,x_p\sim\mathcal{D}_n,\mathcal{D}_p}\big[\frac{1}{2}\big(l((\eta_n \circ \Theta_f)(x_n), y_{x_n}) + l((\eta_p \circ \Theta_f)(x_p), y_{x_p})\big)\big] \tag{4.4}$$

For (b), two objectives are optimized sequentially:

$$\begin{aligned} &\text{(i)} \ \min_{\Theta_f,\eta_n} \mathbb{E}_{x_n,x_p\sim\mathcal{D}_n,\mathcal{D}_p}\big[\frac{1}{2}\big(l((\eta_n \circ \Theta_f)(x_n), y_{x_n}) + l((\eta_n \circ \Theta_f)(x_p), y_{x_p})\big)\big] \\ &\text{(ii)} \ \min_{\eta_n} \mathbb{E}_{x_n\sim\mathcal{D}_n}\big[l((\eta_n \circ \Theta_f)(x_n), y_{x_n})\big] \end{aligned} \tag{4.5}$$

For (c), the objective is simply Eq. 4.5(i). In all cases, the model defined by

$(\eta_n \circ \Theta_f)$ is used at inference time. Both options (a) and (b) allow paintings to be used for feature learning while keeping the inference classifier specific to photos.

Results are summarized in Fig. 4.5. Despite domain differences between photos and paintings, the default classifier (c) has improved robustness over a classifier that is trained on photos alone. A finetuned classifier (b) does not yield much improvement over the default option (c), while domain-specific classifiers (a) do yield significant improvement. This suggests that paintings are useful for feature learning since they can guide the feature extractor towards perceptually relevant features, but constraining the feature space to jointly separate photos and paintings across different classes can restrict the breadth of learned features. The clean accuracy of a joint classifier (finetuned or not) suffers since it can no longer rely on some photo-specific features for classification. We will use domain-specific classifiers in remaining experiments unless otherwise specified.[1]

Answer to H3a: *Surprisingly, we find that paintings can improve model robustness out-of-the-box without accounting for domain shift. However, accounting for domain shift with domain-specific classifiers increases both clean accuracy and robustness significantly.*

To control for robustness gains from photos, we assume a 1:1 cost for photos:paintings with a fixed annotation budget. Fig. 4.6 shows that it is beneficial to allocate up to 50% of any annotation budget for paintings with respect to model robustness.

Answer to H3b: *Using paintings is cost-effective – annotating a combination of photos and paintings results in higher robustness over photos alone for any fixed budget.*

[1]We experimented with domain-specific classifiers in the context of stylization, but found they did not improve robustness over a joint classifier.

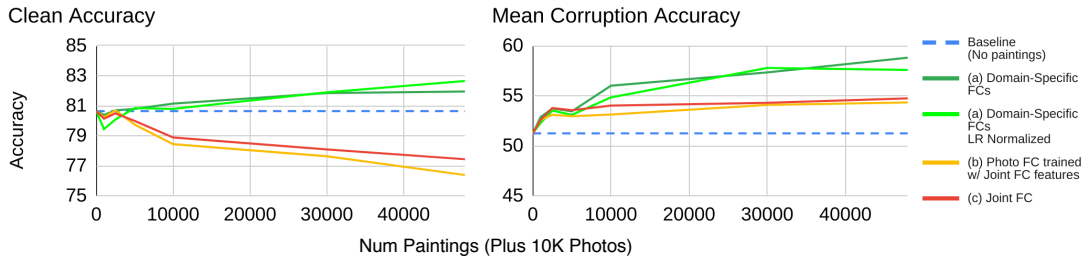


Figure 4.5: **Learning from Paintings.** Left: Clean Accuracy, Right: Corruption Accuracy. Domain-specific classifiers (green) result in the highest robustness while also improving clean accuracy. "LR normalized" refers to fixed effective learning rates to account for additional gradients from the extra classifier head. Even without accounting for domain shifts, training with paintings improves robustness (red/yellow). Results are on Materials.

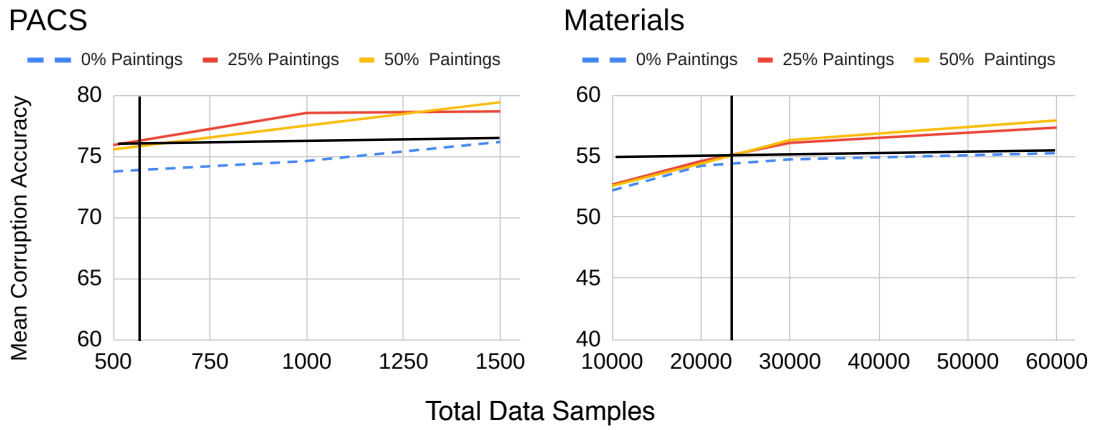


Figure 4.6: **Trade-off Between Photos and Paintings.** Left: PACS, Right: Materials. For a fixed annotation budget, learning from both photos and paintings (25%/50% paintings) results in higher robustness than photos alone (0% paintings), with a bit over $\frac{1}{3}$ of the total number of data samples required to be annotated to match the maximal robustness achieved by only photos.

4.6.2 Paintings vs. Other Visual Artforms

Many artforms are created with an artistic emphasis on perceptually important cues. For example, a line sketch is an abstraction which focuses on salient contours to depict recognizable objects. While sketches are quite good at abstracting away unimportant signals, they also abstracts away many realistic cues in favor

of a sparse line-based representation. In the following experiment, we consider models trained on photographs with different visual artforms.

Table 4.1 summarizes results across four datasets. We find that robustness can be harmed by sparse visual representations like PACS line sketches or DomainNet quickdraw. However, DomainNet sketches, which include more realistic shading and detail, do improve robustness. This is aligned with our expectation that the inclusion of perceptually relevant cues is important for feature learning. VisDA renderings are untextured and shaded with a single directional light source and ambient lighting. Similar to line sketches, we find that these minimal renderings reduce model robustness.

Answer to H4: *Our results position paintings as a unique artform for improving model robustness due to their fine balance between perceptual realism and abstraction.*

4.7 Do Stylized Images and Paintings Induce Similar Invariances?

As shown in Sections 4.5 and 4.6, both stylized images and paintings can improve model robustness. We argued that paintings are a form of perceptual data augmentation in which artists manipulate perceptual cues to emphasize salient regions of scenes. However, it remains unclear whether models are indeed learning perceptual invariances from paintings – it is possible that the robustness gains from paintings arise purely through their mid-level image statistics and textures instead. If paintings are improving robustness through different mechanisms than stylized photos, we can expect different behavior from models

Training Data (# Samples)	Mean Corruption Acc (%)
Materials	
Photo (30K)	54.73±0.25
Photo + **Painting** (15K + 15K)	**56.31**±0.27 (+)
PACS	
Photo (1500)	76.16±0.34
Photo + **Painting** (750 + 750)	**79.41**±0.55 (+)
Photo + Cartoon (750 + 750)	75.38±0.36 (−)
Photo + Sketch (750 + 750)	73.85±0.39 (−)
DomainNet [127]	
Photo (120K)	36.59±0.12
Photo + **Painting** (90K + 30K)	**39.00**±0.14 (+)
Photo + Sketch (90K + 30K)	37.57±0.22 (+)
Photo + Clipart (90K + 30K)	37.00±0.07 (+)
Photo + Quickdraw (90K + 30K)	35.87±0.20 (−)
Photo + Infograph (90K + 30K)	34.60±0.18 (−)
VisDA [128]	
Photo (30K)	**65.97**±0.33
Photo + Rendering (15K + 15K)	63.90±0.21 (−)

Table 4.1: **Robustness from Different Artforms.** Paintings improve model robustness while more abstract artforms can reduce robustness. (+)/(−) indicate whether an artform improves/reduces model robustness. ± indicates standard deviation over 3 runs.

trained on stylized photos and paintings. To investigate how stylized photos and paintings act on model robustness, we empirically probe models to understand their learned invariances. We explore the following:

- **Hypothesis H5.** Models trained on stylized photos and paintings learn different invariances to (a) common image corruptions and (b) viewpoint and lighting shifts, and so (c) models can learn complementary invariances by training on both paintings and stylized photos.
- **Hypothesis H6.** Stylization injects high-frequency signals that improve model robustness.

4.7.1 Probing Learned Invariances

To explore the relative invariances learned by different models, we consider the behavior of models on various types of common image corruptions. We also consider behavior on out of distribution images – in general, these images have a different distribution of viewing angles, viewing scales, and lighting than the original training photos. We experiment with models trained on paintings and AdaIN-stylized photos. In addition to arbitrary style transfer, it is natural to consider learned artistic style transfer. We experiment with SACL [140], which transfers the style of various artists independently with separately trained models. We stylize each photo with a random artist to parallel the real painting datasets which include multiple artists and styles.

Behavior with respect to common corruptions is summarized in Table 4.2. Stylization and paintings both consistently improve robustness to each form of common corruption. On average, SACL outperforms both AdaIN and paintings, giving credence to an argument that stylization methods with strong biases (i.e., learned styles) may be more practical than real paintings or arbitrary stylization methods that depend on a diverse style set (c.f. Section 4.5.2). Observe that the relative performance of paintings fluctuates between datasets – paintings outperform AdaIN on noise and digital on Materials but underperform AdaIN on PACS. As discussed earlier, a collection of paintings encodes perceptual invariances. Since these invariances are not agreed upon a priori for every painting, it follows that *a large set of paintings is required to adequately capture implicitly encoded perceptual invariances*. Finally, all methods are similarly invariant to weather and digital transformations. This can be explained by their mid-level statistics. Weather transformations such as snow, fog, and frost are effectively

overlaid textures on an image while digital transformations such as pixelate and elastic transform resemble the fuzzy boundaries found in both types of images.

Answer to H5a: *Both stylization and paintings improve robustness to various image corruptions. However, learned stylization strictly outperforms paintings, suggesting that invariances from learned style transfer supersedes those from paintings with respect to common corruptions.*

Method	Noise	Blur	Weather	Digital
Materials (30K Samples/Domain)				
Photo-Only	43.70±0.65	58.76±0.14	55.25±0.33	61.20±0.69
Photo + AdaIN	47.33±0.22	**65.09**±0.21	**61.78**±0.18	61.41±0.16
Photo + SACL	**61.87**±0.16	64.36±0.20	57.49±0.24	**66.55**±0.17
Photo + Painting	49.82±0.56	61.03±0.13	56.69±0.10	64.15±0.14
PACS (1.5K Samples/Domain)				
Photo-Only	62.64±1.48	72.75±0.04	83.24±0.22	86.33±0.14
Photo + AdaIN	70.17±1.70	81.18±0.20	88.37±0.23	**89.32**±0.19
Photo + SACL	**85.98**±0.56	**84.61**±0.15	**89.73**±0.33	88.74±0.48
Photo + Painting	68.83±0.83	75.80±0.95	86.88±0.66	87.07±0.14

Table 4.2: **Per-Corruption Accuracy.** (blue) SACL generally outperforms both AdaIN and paintings, particularly on noise. (red) Paintings can outperform AdaIN on some corruptions with a large dataset (Materials), but underperform when fewer images are available (PACS). See main text for discussion. ± indicates standard deviation over 3 runs.

Performance with respect to out-of-distribution images is summarized in Fig. 4.7. In striking contrast to the robustness against image corruption results above, stylization consistently *harms* robustness. The reduced performance of stylization can be explained by model overfitting to view- or lighting-specific signals in the original photo dataset, as the signals in common between a clean photo and its stylized counterpart are seen twice as often by the network during training. On the other hand, paintings are not simply a transformed photograph, and thus do not suffer from this problem. A straightforward explanation of the robustness found through paintings is in the differences in viewpoints and lighting depicted

compared to photos due to circumstance (that is, the paintings simply depict more diverse scenes than the photos). However, paintings are constrained by cultural norms and artistic conventions [153, 107], so it is unlikely that artistic paintings contain a more diverse set of viewpoints than in-the-wild photos. Instead, *we argue it is the emphasis on depicting regions of interest with recognizable characteristics while de-emphasizing details in the background that is helping networks to learn better viewpoint invariance from paintings.* The model is better able to learn to focus on the objects or materials themselves over background context.

Answer to H5b: *For viewpoint and lighting transformations found in out-of-distribution images, using stylization consistently hurts performance while using paintings consistently improves performance.*

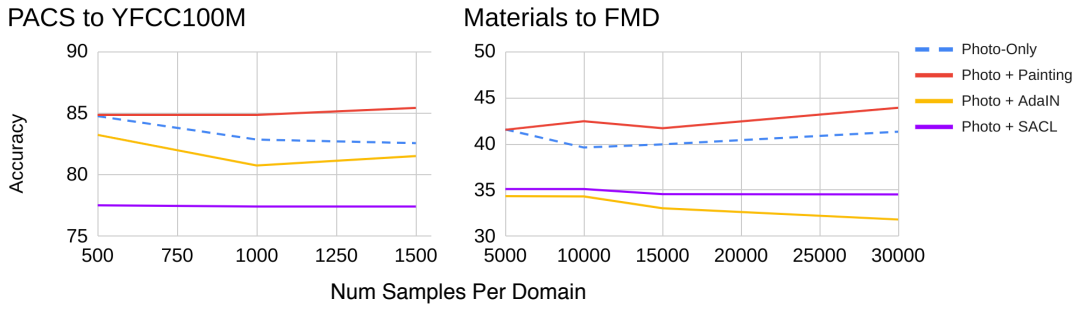


Figure 4.7: **Out-of-Distribution Accuracy.** Left: PACS, Right: Materials. Training with paintings (red) improves robustness to out-of-distribution photos while training with stylized photos (purple/yellow) hurts robustness. Paintings can improve invariance to viewpoints and lighting by encouraging models to focus on objects / materials of interest over background context. Stylization encourages overfitting, an effect which can be exacerbated with more training samples.

Since the behavior of models trained on stylized photos and paintings are indeed different, we explore whether models trained on both sources of data learn complementary invariances, or if the differences result in conflicting behavior. Our results in Table 4.3 suggests the former.

Answer to H5c: *Training with both paintings and stylized photos improves robustness in a complementary manner.*

Method	MEAN	Corr.	OOD
Materials (30K Samples/Domain)			
Photo-Only	48.03±0.21	54.73±0.25	41.33±0.62
Photo + SACL	48.56±0.45	**62.67**±0.03	34.54±0.91
Photo + Painting	50.92±0.22	57.92±0.09	**43.92**±0.47
Photo + SACL + Painting	**51.49**±0.69	61.47±0.50	41.50±1.38
PACS (1.5K Samples/Domain)			
Photo-Only	79.37±0.17	76.16±0.34	82.57±0.00
Photo + SACL	82.35±0.37	87.27±0.10	77.43±0.84
Photo + Painting	82.54±0.59	79.65±0.49	**85.43**±0.70
Photo + SACL + Painting	**85.42**±0.18	**87.31**±0.30	83.52±0.27

Table 4.3: **Learning from Stylization and Paintings.** Training with both stylized images and paintings improves average robustness to image corruptions and out-of-distribution photos, indicating that the invariances learned from these images are complementary. ± indicates standard deviation over 3 runs.

4.7.2 The Role of High Frequency Signals

We have focused our intuitions about the source of invariances learned from stylization and paintings through the visible structure of these images. Existing work has shown that CNNs can learn to extract features from high frequency signals in images [166, 102]. It is also well-known that deconvolutional decoders, such as those used in stylization models, can introduce artifacts in images [120]. It is difficult to form intuitions about these signals, but we can measure whether they play a significant role in improving model robustness.

We apply an ideal circular low-pass filter to zero out high-frequency components. Given an image I, the filtered frequency components of the image are:

$$X_{\text{filtered}} = \mathcal{F}(I) \odot C \tag{4.6}$$
$$\text{where } C_{ij} = \mathbf{1}_{r<\tau}(r(i,j))$$

$\mathcal{F}$ denotes the discrete 2D Fourier transform, $\mathbf{1}$ denotes the indicator function, and τ is the radius of the low-pass filter. We set $\tau = 60$ in our experiments. Fig. 4.8 illustrates images before and after filtering at image resolution 224×224. Note that the filtered images are perceptually identical to the original images at a glance. Therefore, we can train models on the filtered images to measure the impact of the visually negligible high frequency signals which were filtered out.

Table 4.4 summarizes the results. With filtered images, robustness against noise drops significantly for models trained on photos stylized with SACL. This means visible high frequency textures (such as the brush strokes in a Monet stylized photo) are not enough to explain robustness against noise. This effect of invisible high-frequency signals on noise is similar to evidence that learning from adversarial perturbations improves robustness to high frequency corruptions [185]. On the other hand, the effect of high frequency signals on the noise robustness of paintings is much smaller.

Answer to H6: *For learned style transfer, it is the presence of invisible high frequency signals that are doing the heavy lifting against noise. In contrast, paintings are primarily improving invariance towards noise through visible human-perceivable signals.*

Figure 4.8: **Reducing High-Frequency Signals.** Top: Original Image, Bottom: Low Frequency Image. Columns 1 and 3 are stylized photos; columns 2 and 4 are artist-created paintings. Reducing the magnitude of sufficiently high frequency components from images does not alter perceptual quality of images. At a glance, the top and bottom images are perceived to be identical.

Method	Noise	Blur	Weather	Digital	OOD
Materials (30K Samples/Domain)					
Photo-Only	43.70±0.65	58.76±0.14	55.25±0.33	61.20±0.69	41.33±0.62
Photo + SACL	61.87±0.16	64.36±0.20	57.49±0.24	66.55±0.17	34.54±0.91
Photo + Painting	49.82±0.56	61.03±0.13	56.69±0.10	64.15±0.14	43.92±0.47
Photo+SACL (LF)	45.82±1.36	64.24±0.39	57.06±0.13	66.37±0.29	36.92±1.15
Photo+Painting (LF)	44.95±0.66	60.87±0.29	56.82±0.23	63.69±0.46	41.21±0.56
PACS (1.5K Samples/Domain)					
Photo-Only	62.64±1.48	72.75±0.04	83.24±0.22	86.33±0.14	82.57±0.00
Photo + SACL	85.98±0.56	84.61±0.15	89.73±0.33	88.74±0.48	77.43±0.84
Photo + Painting	68.83±0.83	75.80±0.95	86.88±0.66	87.07±0.14	85.43±0.70
Photo+SACL (LF)	77.55±2.60	85.4±0.11	88.93±0.22	88.53±0.15	77.43±0.47
Photo+Painting (LF)	71.16±1.31	75.97±0.71	86.82±0.37	87.35±0.36	83.71±0.40

Table 4.4: **Robustness without High Frequency Signals.** "LF" denotes filtered low frequency images. Photos are always unfiltered. Filtering invisible high frequency components mainly impacts noise robustness. (blue) Filtering stylized photos significantly reduces noise robustness while (red) filtering paintings has a relatively smaller effect. ± indicates standard deviations over 3 runs.

4.8 Discussion and Future Work

In this chapter, we performed an extensive exploration of style transfer and artistic paintings for model robustness. We found that style transfer is able to

improve model robustness *without* painting style images at all (**H1**). Instead, stylization relies on a combination of diversity between style-content image pairs and learned biases to improve model robustness (**H2**). This suggests that although style transfer can improve model robustness, it does so independently from paintings (or the styles found within paintings). This is contrary to conventional wisdom that asserts style transfer improves model robustness through its ability to mimic the textures and styles found in real paintings. Next, we proposed the direct use of paintings as a form of implicit perceptual data augmentation. We confirmed our hypothesis that learning from real paintings can improve robustness, and saw greater gains by accounting for the domain shift between paintings and photos (**H3**). When considered against other artforms such as sketches or cartoons, we found that the fine balance of abstraction and realism in paintings allowed it to uniquely enhance model robustness while other artforms do not directly lead to improved robustness (**H4**). Finally, we found that models learn different invariances from paintings and stylized photos, and that robustness can be improved by training on both forms of data (**H5**,**H6**). Both forms of images allowed models to improve robustness to common corruptions in images. However, models trained on stylized photos found *reduced* robustness to viewpoint, illumination, or other context distribution shifts while paintings improved robustness to such shifts.

From a practical standpoint, our results suggest that learned stylization methods should be considered over arbitrary style transfer methods in data augmentation pipelines. Our results also suggest that training with paintings is a straightforward way to improve model robustness, and should be used if they are available.

There are interesting research directions for future exploration. Work has been done to improve the controls available in style transfer or image editing models [27, 169, 48]. It would be interesting to apply these controls in a perceptually-grounded manner when style transfer is applied to mimic the artistic process. In this chapter, we have found that artforms like sketches are unable to improve model robustness. It would be interesting to explore how coarser abstractions found in art can be leveraged for model robustness, perhaps by encouraging models to learn a hierarchy of invariances.

CHAPTER 5

UNCERTAINTY-AWARE PLANNING WITH SEMANTIC SCENE UNDERSTANDING

5.1 Overview

Computer vision models are often used within larger systems, where their perception of the world feeds into other components. For autonomous navigation, visual reasoning is useful for understanding safe terrain and obstacles in the environment. This understanding can allow the autonomous agent to plan safe paths and perform their tasks. However, different environments will inevitably introduce distribution shift, and even the best computer vision models will make incorrect predictions at times. The previous chapters have explored the question of how to reduce the impact of distribution shift from training to testing. Now, we will explore the important question of how to account for model failures when they arise during deployment, and how a system can still leverage uncertain model predictions.

In this chapter, we propose a pipeline for planning in unstructured environments. This is a challenging task that relies on perception, scene reconstruction, and reasoning about various uncertainties to be successful. To perceive the environment, we use a semantic segmentation model trained to segment terrain and obstacles in scenes. Given a view of the environment, the model allows the agent to reason about potentially safe or unsafe paths through predicted terrain. The model is augmented with dropout to compute uncertainties so that the agent can account for potentially incorrect model predictions. With sparse next-best-view measurements, the agent is able to find new measurements that decreases the

uncertainty in the predicted semantics of the environment. Overall, our algorithmic pipeline consists of: a deep Bayesian neural network which segments surfaces with uncertainty estimates; a flexible point cloud scene representation; a next-best-view planner which minimizes the uncertainty of scene semantics using *sparse* visual measurements; and a hypothesis-based path planner that proposes multiple kinematically feasible paths with evolving safety confidences given next-best-view measurements. Our pipeline iteratively decreases semantic uncertainty along planned paths, filtering out unsafe paths with high confidence. We show that our framework plans safe paths in real-world environments where existing path planners typically fail.

The work in this chapter was published in ICRA 2020 as "DeepSemanticHPPC: Hypothesis-based Planning over Uncertain Semantic Point Clouds" [55] with Yutao Han, Jacopo Banfi, Kavita Bala, and Mark Campbell. Yutao, Jacopo, and I contributed equally to this work.

5.2 Introduction

Path planning for complex outdoor environments is challenging due to the unstructured nature of environments that do not fall neatly into discretized space. Moreover, different terrain surface types can be difficult to detect with traditional sensing modalities. In indoor environments, a grid space representation with lidar sensors is sufficient [152, 69, 83]. Outdoor environments exhibit complex geometries and surface types, which are difficult–if not impossible–to differentiate using just lidar data. Therefore, a more flexible scene representation, surface classification using computer vision techniques, and reasoning about

scene uncertainties are necessary.

Previous work for outdoor planning has focused on classifying terrain and surface roughness using SVM classifiers [165, 104], neural networks [22, 32], and various other computer vision techniques [109, 42]. While these techniques can differentiate between simple terrain types, they do not model the inherent uncertainties and ambiguities in complex scenes which makes it difficult to differentiate between terrain types (e.g., an offroad robot driving through a patch of grass with small rocks). Many current outdoor planning approaches still rely on grid maps which do not model the complex geometry of an outdoor scene (e.g., a field with irregular bumps and rocks) [121, 41]. Recent work models outdoor maps for planning with a point cloud [79], which is more flexible and suitable for unstructured scenes; however, [79] uses traditional lidar sensing which cannot differentiate between different surface types as broadly as camera-based computer vision.

In this chapter, we present DeepSemanticHPPC (Deep Semantic Hypothesis-based Planner over Point Clouds), a novel algorithmic pipeline for planning over uncertain semantic point clouds, which leverages a Bayesian neural network (BNN) [45, 74] to extract principled estimates of segmentation uncertainty. This allows our framework to reason about ambiguous terrain as well as robustly handle false positive detections by taking additional measurements to reduce semantic uncertainty in the scene. However, each measurement is costly due to the computationally expensive nature of Bayesian neural networks operating on a robotic platform with limited computing power. Our planner hence employs next-best-view (NBV) techniques [15, 62, 31] to optimize for new measurements. DeepSemanticHPPC includes:

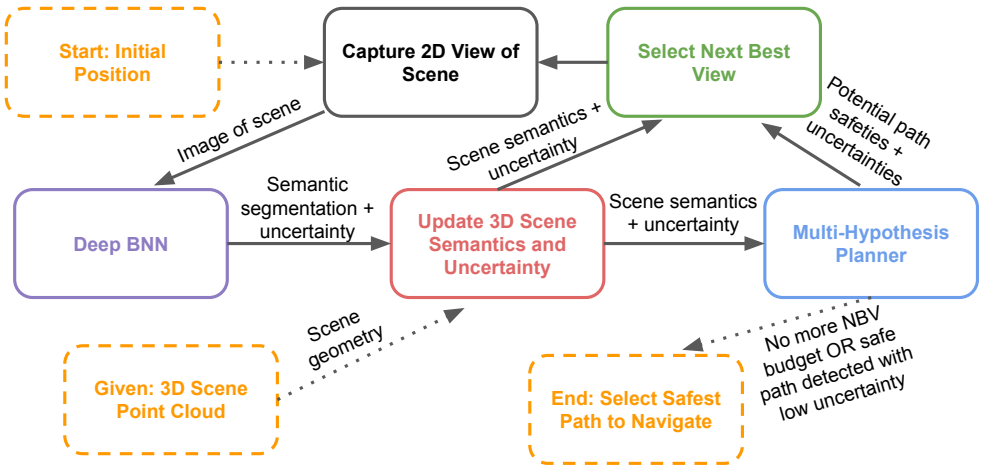


Figure 5.1: The DeepSemanticHPPC pipeline. (1) Given an initial view and scene geometry, a multi-hypothesis graph of possible paths is generated. (2) The uncertainty in the scene is iteratively reduced by selecting next-best-views and path costs are updated. (3) This iterative uncertainty-reduction stage is terminated early if a safe path is confirmed or all considered paths are confirmed as unsafe. (4) Finally, a path is selected.

- the employment of a deep Bayesian neural network [45, 74] to obtain surface and obstacle semantics with uncertainty estimates for unstructured outdoor environments;
- a flexible point cloud scene representation;
- a next-best-view planner which minimizes the uncertainty of terrain semantics using *sparse* visual measurements;
- a hypothesis-based path planner (extending [79]) that proposes multiple kinematically feasible paths with evolving safety confidences given the NBV measurements.

Experimental results with real environments show that our pipeline plans safe paths in real-world environments where existing path planners typically fail. Fig. 5.1 illustrates DeepSemanticHPPC. In the first stage, a multi-hypothesis

planner generates multiple hypotheses of possible safe paths given a scene belief. In the second stage, a NBV function calculates NBV poses and associated rewards. These poses and rewards are input to a NBV selection block which selects the best *feasible* NBV. A BNN extracts semantic segmentations and associated uncertainties from the NBV measurement, which are used to generate a new scene belief. The new scene belief reduces hypothesis uncertainty, and the second stage is repeated for a set number of iterations. Finally, a safe path (hypothesis) with high confidence is selected; if all paths are classified as unsafe, then no path is selected. The algorithm terminates once a path is confirmed safe or all paths are confirmed unsafe.

5.3 Related Work and Background

RRT-Based Non-Holonomic Planning over a Point Cloud We build upon an existing rapidly-exploring random tree (RRT) [82] based planner for finding kinematically feasible trajectories over non-planar point cloud environments [79]. 6D robot poses are expressed by transformation matrices belonging to the Special Euclidean Group SE(3). A matrix $\mathbf{T}_{\text{MR}}$ specifies the position and orientation of a robot-fixed coordinate frame R expressed in a given reference map frame M. [79] considers the following planning problem: given start and goal poses $\mathbf{T}_{\text{MS}}$, $\mathbf{T}_{\text{MG}}$, and a point cloud $\mathcal{M} = \{\mathbf{m}^i\}$ with $\mathbf{m}^i \in \mathbb{R}^3$, compute a connecting trajectory $\pi : \mathbb{R}_{>0} \to \text{SE(3)}$. The trajectory has to satisfy a number of constraints up to a given degree of approximation: contact with the terrain surface, static traversability (e.g. bounded roll and pitch angles), and kinematic constraints – including bounded continuous curvature. Trajectories are represented as piecewise continuous functions in the 6D space of robot poses, and are specified by a sequence of nodes

$\widehat{\pi} = [\mathcal{N}^k]$, where each $\mathcal{N}^k$ is a tuple $(\mathbf{T}_{\text{MR}^k}, \tau^k, \mathbf{w}^k, \kappa^k)$. Here, $\mathbf{T}_{\text{MR}^k}$ is a 6D pose attached to the terrain surface, $\tau^k \in [0, 1]$ is the associated static traversability value, $\mathbf{w}^k$ is a parameter vector specifying a short *planar* trajectory segment connecting $\mathbf{T}_{\text{MR}^k}$ to the next pose in the sequence, and κ^k is the curvature at the beginning of the trajectory segment. $\mathbf{w}^k$ specifies a trajectory segment as a cubic curvature polynomial [117] evolving along the planar patch defined by the xy plane of the coordinate frame R attached to $\mathbf{T}_{\text{MR}^k}$. The end point of such a trajectory segment gives the subsequent pose $\mathbf{T}_{\text{MR}^{k+1}}$ through a projection on the terrain surface via $f : (\mathcal{M}, \mathbf{T}_{\text{MR}}) \mapsto \mathbf{T}_{\text{MR}}$; f queries $\mathcal{M}$ for the K nearest-neighbors of the end point, which can be thought of as the points the robot will lie on at $\mathbf{T}_{\text{MR}^{k+1}}$ (K depends on the size of the robot and point cloud density). We use $\phi(\mathcal{N}^{k+1})$ to denote such points.

Leveraging the above trajectory representation, [79] proposes to define a small set of *motion primitives* (short trajectory segments) and use them to grow two RRTs, one from the start pose and one from the goal pose, and iteratively try to connect them. Each new pose is associated with a node $\mathcal{N}^k$, which is accepted in the tree only if $\tau^k > 0$. [79] also proposes a technique to derive a better trajectory (in terms of smoothness and distance) starting from an initial one. This second optimization stage is not explicitly considered in this work, because it is easily generalized and applied to all "safe" trajectories according to our method.

Segmentation with Bayesian Neural Networks Although segmentation networks for 3D point clouds exist [132, 133, 194, 131, 171], 3D data repositories are focused on object recognition and part segmentation (e.g. [20]) or only contain a small number of scenes [53]. In contrast, existing large-scale image segmentation datasets [76, 17, 192, 11] contain varied surfaces and obstacles in diverse

real-world outdoor scenes. Therefore, we leverage a state-of-the-art image segmentation network [24] (Section 5.5.1), and update the point cloud environment from per-pixel image segmentations (Section 5.5.2). Furthermore, we augment the network to estimate output uncertainty, allowing uncertainty in surface predictions to be embedded in the point cloud. Uncertainty in surface type is used to guide path-safety evaluation.

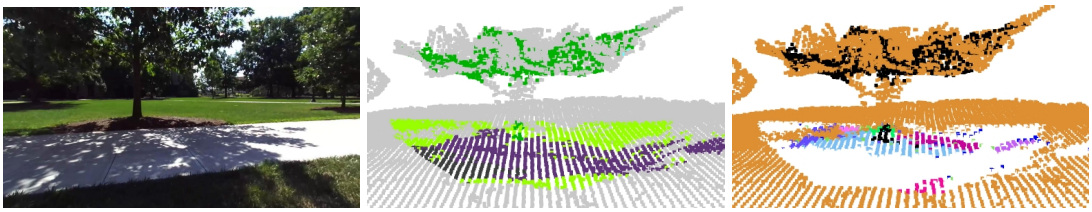
At inference time, forward passes with active dropout layers can be interpreted as an approximation of the posterior distribution of model weights of a neural network [45, 74]. The uncertainty of predictions are computed by taking the sample standard deviation across multiple forward passes. For each pixel $(i,j)_X$ of image X, the mean softmax vector over T forward passes is:

$$\mathbf{p}^{(i,j)_X} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{s}_t^{(i,j)_X}(y|X) \tag{5.1}$$

where $\mathbf{s}^{(i,j)_X}(y|X) \in \mathbb{R}^C$ is the softmax output of the network. The corresponding uncertainty vector on $\mathbf{p}^{(i,j)_X}$ is:

$$\boldsymbol{\sigma}^{(i,j)_X} = \sqrt{\frac{\sum_{t=1}^{T}\left(\mathbf{s}_t^{(i,j)_X}(y|X) - \mathbf{p}^{(i,j)_X}\right)^2}{T-1}} \tag{5.2}$$

In our framework, image X corresponds to a view with known camera parameters. $\mathbf{p}^{(i,j)_X}$ and $\boldsymbol{\sigma}^{(i,j)_X}$ are combined with existing measurements for each point $\mathbf{m} \in \mathcal{M}$ that maps to pixel $(i,j)_X$ (section 5.5.2).



(a) (b) (c)

Figure 5.2: An example point cloud. **(a)** Image view of a portion of the environment. **(b)** Point cloud colored with the most likely class predicted from image (a) (bright green: "grass"; dark green: "tree"; purple: "sidewalk"; dark grey: "road"; light grey: no information available). All the classes except "tree" belong to the set S. The region around the tree is actually mulch/woodchips, which should be classified as "dirt" (belonging to U). **(c)** Point cloud colored to show safe (white), unsafe (black), and unclear regions $\mathcal{R}$ (random colors).

5.4 Approach Overview

The planner presented in Section 5.3 does not leverage critical semantic information about terrain types. In this work, we assume an initial point cloud is given in the form $\mathcal{M} = \{\mathbf{e}^i\}$, where each element $\mathbf{e}_i$ is a tuple $(\mathbf{m}^i, \mathbf{p}^i, \boldsymbol{\sigma}^i)$. Here, $\mathbf{m}^i \in \mathbb{R}^3$ as before; $\mathbf{p}^i = (p_1^i, p_2^i, \ldots, p_C^i)$ is a vector specifying the probabilities that the point belongs to each one of the C possible semantic classes (gravel, water, etc.); $\boldsymbol{\sigma}^i = (\sigma_1^i, \sigma_2^i, \ldots, \sigma_C^i)$ is a vector specifiying the uncertainties of $\mathbf{p}^i$, as discussed in Section 5.3. Points are initialized with uniform semantic probabilities and maximum uncertainties. Updating the semantic point cloud is discussed in Section 5.5.2. Fig. 5.2(b) shows a pointcloud obtained in a real (ambiguous) environment, where each point $\mathbf{m}^i$ is associated with the color corresponding to the class label j whose p_j^i is maximum.

We assume the semantic classes have been partitioned into two sets: the safe set S (e.g. gravel, grass) and the unsafe set U (e.g. water, snow). For each point $\mathbf{m}^i$, the points are defined as $p_S^i = \sum_{j \in S} p_j^i$, $p_U^i = 1 - p_S^i = \sum_{j \in U} p_j^i$, and

$\sigma^i = \min(\sqrt{\sum_{j \in S} \sigma_j^{i2}}, \sqrt{\sum_{j \in U} \sigma_j^{i2}})$. Each point is then classified as:

- **safe** if $p_S^i - w_\sigma \sigma^i \geq \theta_s$;
- **unsafe** if $p_U^i - w_\sigma \sigma^i \geq \theta_u$;
- **unclear** otherwise.

Intuitively, this implies that points are safe/unsafe given high probability (p_S^i, p_U^i) and low uncertainty (σ^i), and unclear otherwise. w_σ, θ_s, θ_u are defined by the mission planner (with $1 - \theta_s < \theta_u$). We use $\mathcal{M}_{\text{safe}}$, $\mathcal{M}_{\text{unsafe}}$, $\mathcal{M}_{\text{unclear}}$ to denote the partition of $\mathcal{M}$ obtained from the above classification. Note that a point is labeled safe (unsafe) even with large uncertainty on p_S^i (p_U^i), provided there is small uncertainty on p_U^i (p_S^i). This is captured by the $\min$ in the definition of σ^i. For example, the network may be uncertain between gravel and grass (both safe), but it is sure that the point is neither water nor snow (both unsafe).

Consider now a trajectory $\widehat{\pi} = [\mathcal{N}^k]$, and recall that $\phi(\mathcal{N}^k)$ denotes the set of points on which the robot lies when at pose $\mathbf{T}_{\text{MR}^k}$. Depending on the semantic information initially available, it might be very difficult –if not impossible– to immediately find a trajectory whose node points $\phi(\mathcal{N}^k)$ all belong to $\mathcal{M}_{\text{safe}}$. DeepSemanticHPPC works in two stages:

1) **Compute a set of candidate paths** traversing different *unclear regions*, and
2) **Reduce the uncertainty of such paths** by taking *new views* in the proximity of the robot's starting position of the *most promising path*.

We relax the path planning problem in a natural way – instead of reaching a specific goal pose, we require the robot to reach a goal pose region G defined

around $\mathbf{T}_{\text{MG}}$. Then, the points in $\mathcal{M}_{\text{unclear}}$ are organized into a set $\mathcal{R}$ of *unclear regions*. To build the set $\mathcal{R}$, we use the following two-stage clustering process: first, DBSCAN [38] performs a large-scale clustering of the points in $\mathcal{M}_{\text{unclear}}$, obtaining a set of large unclear regions $\widehat{\mathcal{R}}$. Then, the points of each $\hat{r} \in \widehat{\mathcal{R}}$ are further partitioned according to their most likely class (treating the points not associated with any prediction as belonging to a special class); DBSCAN is called again on each partition. Fig. 5.2(c) shows the result of this process on our example with $\theta_s = 0.9, \theta_u = 0.3, w_\sigma = 3$.

The remaining task is to compute a set of candidate paths from $\mathbf{T}_{\text{MS}}$ to G. Section 5.5.3 presents a variant of a standard RRT algorithm which constructs multiple hypothesis for the safest path, traversing different unclear regions. These paths are stored in the directed graph $G = (V, A)$, where each $v \in V$ is associated with a potential trajectory node $\mathcal{N}^v$ and each $a \in A$ represents the existence of a short trajectory segment connecting two poses. The cost for each node is based on how far it is from satisfying our safety constraint: $\bar{p}^v = \frac{1}{|\phi(\mathcal{N}^v)|} \sum_{i \in \phi(\mathcal{N}^v)} \min(1, \max(0, \theta_s - p_S^i + w_\sigma \sigma^i))$ for for each $v \in V$. However, if the node region sufficiently intersects with an unsafe region, the cost is infinite; and if the node lies entirely in a safe region, the cost is zero.

Summarizing, $c(v)$ is defined as:

$$c(v) = \begin{cases} 0 & \text{if } |\mathcal{M}_{\text{safe}} \cap \phi(\mathcal{N}^v)| = |\phi(\mathcal{N}^v)| \\ \infty & \text{if } |\mathcal{M}_{\text{unsafe}} \cap \phi(\mathcal{N}^v)| \geq \phi_v \\ \bar{p}^v & \text{otherwise,} \end{cases} \tag{5.3}$$

where ϕ_v is a user-defined threshold. The first condition can be relaxed for the

nodes in close proximity of the starting pose. Although a vertex with infinite cost is never obtained when the candidate paths are initially computed, its cost might tend to infinity when additional views are taken during the NBV stage (Section 5.5.4). A predefined number of NBV iterations are run. At each iteration, the m most promising (shortest) paths according to the above cost function are computed by a k-shortest paths algorithm (we use Yen's [184]). The associated vertices v such that $0 < c(v) < \infty$ are then considered in the function that computes the best additional view. The value of m is also decided by the mission planner: $m = 1$ corresponds to an aggressive setting, while $m > 1$ is preferred given a large temporal budget for taking additional views.

5.5 Technical Details

5.5.1 Predicting Semantic Labels

We curate a segmentation dataset for outdoor navigation in unstructured environments from the existing large-scale COCO panoptic dataset [76]. First, images of unstructured outdoor scenes are selected using a Places365 [191] classifier. An image is kept if (a) the classifier's top one (highest) prediction is an unstructured outdoor category with $> 50\%$ probability, or (b) two or more of the top five predictions are unstructured outdoor categories. Next, the 133 categories in COCO panoptic are merged: (a) all outdoor terrains (e.g. grass, dirt, snow, pavement) are retained; (b) obstacles are merged into four categories: fixed obstacles (e.g. buildings), moving human-made obstacles (e.g. vehicles), humans, and animals; (c) all indoor categories are removed. Our final dataset consists

of 34K training images with 22 categories. Our navigation segmentation categories (from COCO panoptic), and filtered list of COCO panoptic images are at: https://deepsemantichppc.github.io

For our network architecture, we use DeepLabv3+[24] with Xception65 [25] backbone augmented with dropout in the middle and exit flow blocks for semantic segmentations. At inference time, 50 forward passes are used to predict semantics and uncertainties (Eqs. (5.1)-(5.2)).

5.5.2 Associating Semantic Labels to a Point Cloud

Given a viewpoint with known pose and camera intrinsics, image segmentation probabilities and uncertainties are mapped to the point cloud. To map pixel $(i, j)_X$:

1. Estimate depth map $\mathbf{D}_X$ for view X.
2. Each pixel is backprojected to a single point corresponding to the center of the projected pixel. This approximation does not hold for pixels with very large depths, but typically produces good results. Backprojected pixel point $\tilde{\mathbf{m}}^{(i,j)_X}$ is computed as:

$$\tilde{\mathbf{m}}^{(i,j)_X} = \mathbf{P}_X[\mathbf{D}_X^{(i,j)}\mathbb{I}_{3\times 3}|[0\ 0\ 1]^T]^T\mathbf{K}_X^{-1}[i\ j\ 1]^T \tag{5.4}$$

   where $\mathbf{K}_X$, $\mathbf{P}_X$ are intrinsic and pose matrices.
3. Each backprojected point is merged with its nearest neighbor $\mathbf{m}^{\text{nn}}$ in the point cloud $\mathcal{M}$ within threshold distance R. If a backprojected point does not have a neighbor within the threshold, the point is discarded.

4. $\mathbf{p}^{(i,j)_X}$ and $\boldsymbol{\sigma}^{(i,j)_X}$ are merged with existing measurements for point $\mathbf{m}^{\text{nn}}$. The combined measurement is the best linear unbiased estimator under the simplifying assumption that the per-class predictions are independent. Given a set of K measurements $\{(\mathbf{p}^{(k)}, \boldsymbol{\sigma}^{(k)})\}$, the combined measurement $(\tilde{\mathbf{p}}, \tilde{\boldsymbol{\sigma}})$ for class c is:

$$\left(\tilde{p}_c = \frac{1}{Z} \sum_{k=1}^{K} w^{(k)} p_c^{(k)} \quad , \quad \tilde{\sigma}_c = \sqrt{\sum_{k=1}^{K} (w^{(k)})^2 (\sigma^{(k)})^2} \right)$$
$$\text{where } w_c^{(k)} = \frac{(\sigma_c^{(k)})^{-2}}{\sum_{c' \in C} (\sigma_{c'}^{(k)})^{-2}} \quad , \quad Z \text{ s.t. } \sum_{c' \in C} \tilde{p}_{c'} = 1 \tag{5.5}$$

5.5.3 RRT-Based Multi-hypothesis Planner

The algorithm to compute $G = (V, A)$ starts by building an initial RRT from a root vertex v_s associated with the projected start pose $\mathbf{T}_{\text{MS}}$ via a predefined set of motion primitives. Instead of building two RRTs as is done in [79], we build a single RRT from the start pose and bias the sampling to the point that is closest to the projected ideal goal pose. Sampling is performed on the points in $\mathcal{M}$ not lying in the *forbidden points set* $\mathcal{F}$, which is initialized as $\mathcal{F} \leftarrow \mathcal{M}_{\text{unsafe}}$. Once the first path $\widehat{\pi} = [\mathcal{N}^v]$ is found, the algorithm examines which regions in $\mathcal{R}$ are traversed by the vertices of $\widehat{\pi}$ by checking their intersections with each set of points $\phi(\mathcal{N}^v)$. Except for regions containing points belonging to the K-nearest neighbors of $\mathbf{T}_{\text{MS}}$ and $\mathbf{T}_{\text{MG}}$, all regions traversed by $\widehat{\pi}$ are placed into the *removal candidates set* $\mathcal{C}$.

A heuristic is then used to decide which region(s) should be removed from subsequent planning stages. In this work, we simply remove the largest region $\hat{c}$,

and $\mathcal{F}$ is updated as $\mathcal{F} \leftarrow \mathcal{F} \cup \hat{c}$. When $\hat{c}$ is removed, the RRT vertices lying on it, including those of $\hat{\pi}$, are removed from the RRT. The algorithm then proceeds to optimize and find a new path to the goal, by continuously expanding the RRT component containing v_s. This time, however, the algorithm also tries to connect (by computing *ad hoc* trajectory segments) new vertices to those that were disconnected from the RRT due to the removal of $\hat{c}$.

When a new path is found, the process repeats. If at any iteration, the algorithm finds a path only contained in the regions of $\mathbf{T}_{\text{MS}}$ and $\mathbf{T}_{\text{MG}}$, it can be restarted with a different random seed. Graphs produced by different random seeds are merged at the end. Fig. 5.3 shows an example multi-hypothesis graph computed on the example of Section 5.4 (Fig. 5.2).

If any path computed on the multi-hypothesis graph $G = (V, A)$ has zero cost, the robot starts to follow that path since all the underlying points lie in $\mathcal{M}_{\text{safe}}$. Otherwise, the robot enters the NBV stage described below.

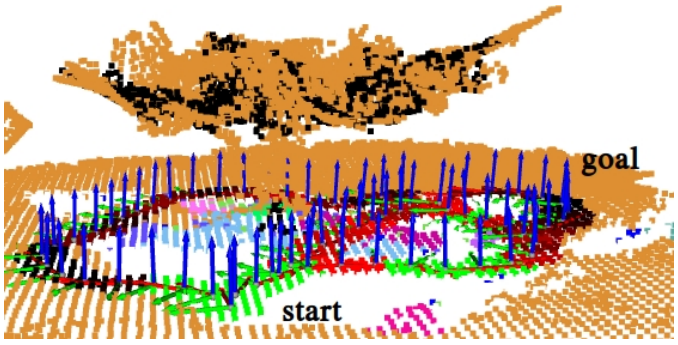


Figure 5.3: An example graph $G = (V, A)$, with poses. Vertices with $c(v) = 0$ are in green, while vertices with $0 < c(v) \leq 1$ are in red (the darker, the closer to 1). The blue, green, and red axes indicate the pose of the robot.

5.5.4 Next-Best-View (NBV) Planning

A set of n viable NBV poses $\mathbf{T}_{\text{viable}} = \{\mathbf{T}_{\text{MV}_1}, \ldots, \mathbf{T}_{\text{MV}_n}\}$ is computed by growing a RRT starting from $\mathbf{T}_{\text{MS}}$. The candidate poses $\mathbf{T}_{\text{viable}}$ are a subsample of the RRT vertices. All points lying within $\mathcal{M}_{\text{unclear}}$ are treated as *unsafe*, and sampling is performed on the safe points lying within a given radius r from $\mathbf{T}_{\text{MS}}$. The robot should not travel too far to take a new view; otherwise it is more appropriate to follow the most promising path of $G = (V, A)$.

A reward $J(\mathbf{T}_{\text{MV}_j})$ is calculated for each candidate pose $\mathbf{T}_{\text{MV}_j}$. The NBV pose $\mathbf{T}_{\text{NBV}}$ is selected by picking the pose with the highest value of J that also allows a safe path back to the current pose from which all the paths can be followed.

$$J(\mathbf{T}_{\text{MV}_j}) = \beta_d D + \beta_\gamma \gamma + \beta_{\text{vis}} N_{\text{vis}} + \beta_Q \overline{Q}, \tag{5.6}$$

where D is a distance metric, γ is a viewing angle metric, N_{vis} is the number of visible vertices from $\mathbf{T}_{\text{MV}_j}$, and $\overline{Q}$ is the average information gain over visible vertices from $\mathbf{T}_{\text{MV}_j}$. The weight $\beta_d, \beta_\gamma, \beta_{\text{vis}}, \beta_Q$ sum to 1 and D, γ, N_{vis}, and $\overline{Q}$ are normalized. Distance and change in viewing angle are used based on the assumption that closeness and view diversity will reduce vertex uncertainty. N_{vis} puts more weight on candidate poses which have higher chances of reducing the uncertainty of multiple segments of the multipath graph $G = (V, A)$. $\overline{Q}$ represents the expected reduction in uncertainty in the graph vertices given $\mathbf{T}_{\text{MV}_j}$. Each of these components are defined as follows.

Begin by defining the set of vertices $v \in V_{\text{NBV}}$, where V_{NBV} is the set of unclear vertices belonging to the m most promising paths (see the end of Section 5.4). For each candidate pose $\mathbf{T}_{\text{MV}_j} \in \mathbf{T}_{\text{viable}}$, only vertices visible from $\mathbf{T}_{\text{MV}_j}$ are

considered in calculating the reward. Visible vertices are defined as vertices which occupy greater than a predefined number of pixels in the image plane rendering of the point cloud from $\mathbf{T}_{\text{MV}_j}$. Visible vertices are added to the set $v \in V_{\text{vis},j}$, where $V_{\text{vis},j} \in V_{\text{NBV}}$.

To calculate D, the distances from $\mathbf{T}_{\text{MV}_j}$ to $v \in V_{\text{vis},j}$ are normalized. Since a lower distance should correspond to a higher reward, we subtract the normalized distances from 1. To calculate γ: the set of negative cosine distances of the angle between $\mathbf{T}_{\text{MS}}$ and $\mathbf{T}_{\text{MV}_j}$ to $v \in V_{\text{vis},j}$ are used. N_{vis} is the size of $V_{\text{vis},j}$.

The information gain metric $Q(v)$ is calculated for each $v \in V_{\text{NBV}}$. $Q(v)$ represents the expected reduction in uncertainty for each $v \in V_{\text{NBV}}$, and is a function of the visibility and uncertainty of the points lying in $\phi(\mathcal{N}^v)$.

The visibility $I(v, \mathbf{T}_{\text{MV}_j})$ of a vertex $v \in V_{\text{NBV}}$, given a candidate pose $\mathbf{T}_{\text{MV}_j}$, is the pixel coverage of $\phi(\mathcal{N}^v)$ in the rendered image plane of $\mathbf{T}_{\text{MV}_j}$. Per-point bounding squares of size equal to half the point cloud resolution are used to compute occlusions and pixel coverage for surface points. The number of pixels that $\phi(\mathcal{N}^v)$ occupies in the rendering is the predicted visibility $I(v, \mathbf{T}_{\text{MV}_j})$ of v at $\mathbf{T}_{\text{MV}_j}$.

The uncertainty $\sigma(v)$ of a vertex $v \in V_{\text{NBV}}$ is the average sum of the uncertainties of the points in $\phi(\mathcal{N}^v)$:

$$\sigma(v) = \frac{1}{|\phi(\mathcal{N}^v)|} \sum_{i \in \phi(\mathcal{N}^v)} \sum_{j=1}^{C} \sigma_i^j \tag{5.7}$$

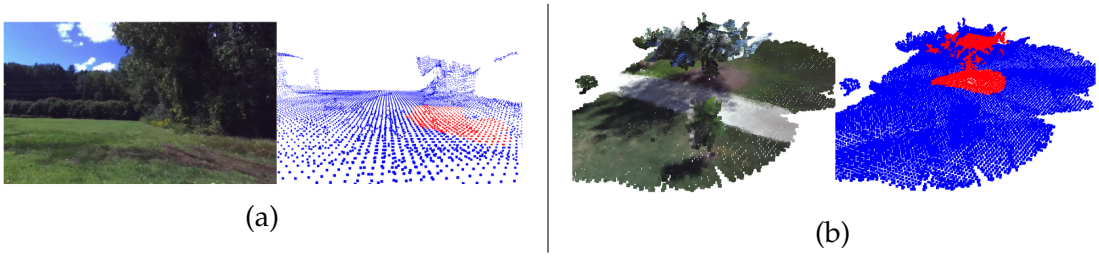
The information gain metric $Q(v)$ of vertex $v \in V_{\text{NBV}}$ can be formally written as

$$Q(v) = \alpha_i I(v, T_{MV_j}) + \alpha_\sigma \sigma(v), \tag{5.8}$$

where α_I, α_σ are weights that sum to 1 and $I(v, \mathbf{T}_{\mathrm{MV}_j})$ and $\sigma(v)$ are normalized.

5.6 Validation

5.6.1 Validation Scenes and Overview



(a) (b)

Figure 5.4: **(a)** Image from Cass Park in Ithaca. There are multiple different terrains in the scene including grass, mud, and water. The point cloud is annotated with safe (blue) and unsafe (red) regions. **(b)** Image from Mann Library in Cornell with similarly annotated safe/unsafe regions.

For real world validation, we collect data of two different scenes using the ZED stereo camera from Stereolabs. One scene is at Cass Park in Ithaca (Fig. 5.4(a)) and the other scene is next to the Mann Library in Cornell University (Fig. 5.4(b)) and These scenes are selected due to their varying (but common) terrain types. The scenes are representative of common unstructured outdoor environments. The ZED camera API is used to extract depth maps and generate point cloud reconstructions of the scenes.

For these scenes, we heuristically select a set of candidate NBV poses instead of growing a RRT from $\mathbf{T}_{\mathrm{MS}}$, and assume a path between these poses and the start

pose exists. This is because we did not implement the algorithm onboard a robot for real-time NBV selection, and we could not feasibly sample all possible NBVs from the test scenes prior to our experiments. Candidate NBV poses are selected to be (a) near the start pose and (b) oriented in the general direction of the goal while covering a wide range of the scene. Our method (and baseline methods) choose NBVs from the set of candidate NBV poses. We did also implement our full pipeline in the AirSim simulator [143] and confirmed that the RRT-based NBV selection module works as intended. Nevertheless, our path uncertainty and safety evaluations will be focused on the real world scenes as the simulated scenes are far more simplistic than the real world.

5.6.2 Multipath Planner Evaluation

The multipath planner is evaluated on the simulated Africa dataset from Airsim. To evaluate the ability of the multipath planner to plan a diverse set of paths, we compare the number of paths planned to the number of iterations the RRT in the multipath planner runs for. The number of iterations is 5000 per trial. 300 trials were run with uniformly sampled starting and goal poses within a ten meter radius of a predefined start and goal anchor locations. Due to the exclusion of the forbidden regions $\mathcal{F}$ after every new path is found, Table 5.1 demonstrates the ability of the multipath planner to plan a *spatially diverse* set of paths. A boxplot illustrating the median, quartiles, and outliers is shown in 5.5.

RRT Iterations	1000	2000	3000	4000	5000
# Paths (Mean)	0.65	1.42	2.39	3.82	5.14
# Paths (Std)	0.77	1.49	2.02	2.46	2.97

Table 5.1: Mean and standard deviation of the number of paths found over 300 trials.

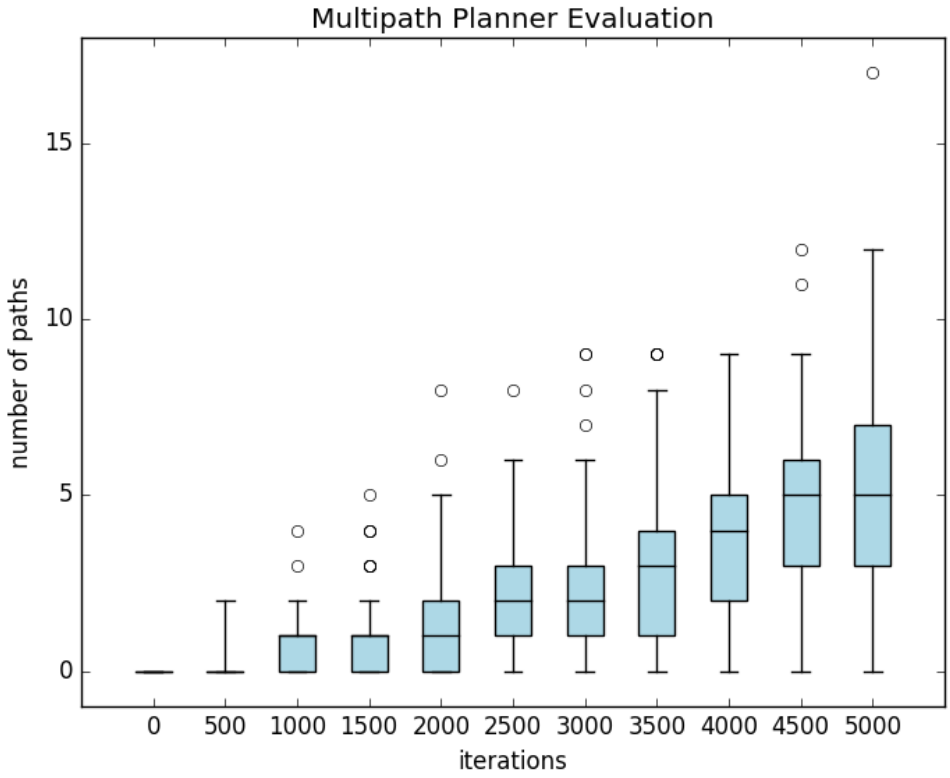


Figure 5.5: Boxplots of the number of paths found at different number of iterations of the RRT over 300 trials are shown. The outliers are shown as black circles.

5.6.3 NBV evaluation

To evaluate the performance of the NBV function, we examine the change in uncertainty of the path vertices as the number of NBVs increases. The complete NBV reward function (Eq. 5.6) is compared against: (a) random selection, (b) geometry-only reward, and (c) uncertainty-only reward. For the geometry-only NBV reward, we set $\{\beta_Q\}$ to zero, and for the uncertainty-only NBV reward, we set $\{\beta_d, \beta_\gamma, \beta_{\text{vis}}\}$ to zero. The change in uncertainties summed across all the classes and points for each path vertex averaged over 500 trials is shown in Fig. 5.6. In our experiments, the full NBV reward weights are set as follows:

$\{\beta_d = 0.4, \beta_\gamma = 0.05, \beta_{\text{vis}} = 0.25, \beta_Q = 0.3, \alpha_I = 0.5, \alpha_\sigma = 0.5\}$ for the Mann Library scene, and $\{\beta_d = 0.15, \beta_\gamma = 0.05, \beta_{\text{vis}} = 0.2, \beta_Q = 0.6, \alpha_I = 0.3, \alpha_\sigma = 0.7\}$ for the Cass Park scene. A higher weight is assigned to uncertainty terms for the Cass scene because the boundaries between surface types (e.g. water, mud, grass) are more ambiguous than in the Mann scene.

For both scenes, the full NBV reward function consistently achieves the lowest uncertainty with 2 or more NBVs (Fig. 5.6). This illustrates the importance of both geometry and uncertainty terms in the reward function. In the Mann scene, the baseline reward functions converge to a higher uncertainty than the full reward function. Due to the small size of the scene and the nature of all candidate NBVs being oriented towards the goal pose, random selection performs quite well. It initially outperforms uncertainty-only which does not take into account point visibility or viewpoint diversity, while random sampling implicitly selects diverse views. In the Cass scene, the baseline reward functions converge to a higher uncertainty than the full reward function, except for uncertainty-only which converges to the same point as the full reward function. The overall uncertainty in the BNN predictions are much higher at Cass park, so heavily weighting the uncertainty in the reward function performs well. In the Mann scene, geometry-only outperforms uncertainty-only, whereas the opposite result holds for the Cass scene. Mann has less inherent ambiguity so geometry terms are more important, while Cass has more ambiguous regions so uncertainty terms are more important.

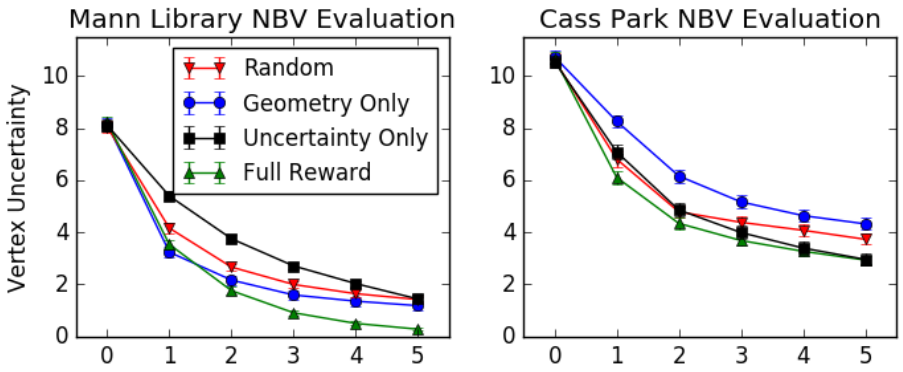


Figure 5.6: Change in uncertainty of path vertices (y-axis) as the number of NBV measurements increase (x-axis).

5.6.4 Path Safety Evaluation

To evaluate the real world application of DeepSemanticHPPC, we study the safety of selected paths. We annotate a point cloud (Fig. 5.4 (a)/(b)**right**) with MeshLab [26] into ground truth safe and unsafe regions. For Mann library mulch (dirt) is labeled as unsafe, and for Cass Park mud (dirt) and water are labeled as unsafe. Any path which contains a vertex that overlaps with the unsafe region with over N_{unsafe} points is unsafe. We set $N_{\text{unsafe}} = 4$. Two baselines are considered: (a) B1: planning without semantic information (based on [79]) and (b) B2: planning with semantic information from a single initial view without taking any NBV measurements to reduce path uncertainty. We also study the performance of DeepSemanticHPPC as the number of NBVs increase.

Table 5.2 and Figure 5.7 shows the path safety results for the Mann Library and Cass Park scenes over 500 trials. Since both safe and unsafe terrain surfaces can be geometrically similar, baseline B1 cannot reliably avoid unsafe semantic regions. Baseline B2 performs significantly better than B1 because of the inclusion of semantic surface types in the planner. However, because the semantic segmentations can be incorrect, especially in regions with high uncertainty, this

Mann	**B1**	**B2**	**1N**	**2N**	**3N**	**4N**	**5N**
Safe %	0	23.4	81.6	79.8	81.4	83.6	**86.0**
Unsafe %	100	76.6	18.4	15.4	11.2	6.6	**3.8**
CS %	N/A	N/A	0	0	4.0	18.0	28.0
CN %	N/A	N/A	0	4.8	7.4	9.8	10.2
Cass	**B1**	**B2**	**1N**	**2N**	**3N**	**4N**	**5N**
Safe %	13.2	33.4	54.0	57.4	57.4	59.0	**59.2**
Unsafe %	86.8	66.6	46.0	41.6	39.8	37.4	**36.6**
CS %	N/A	N/A	0	0	0	0	0
CN %	N/A	N/A	0	1.0	2.8	3.6	4.2

Table 5.2: 500 trials of path safety evaluation. The columns are the path planning methods used: **B1** is the planner based on [79], **B2** includes semantic reasoning without any next-best-views (NBVs), and X**N** is DeepSemanticHPPC (ours) with X NBVs. The rows are the metrics: **Safe** is the number of trials where the final selected path is safe, **Unsafe** is the number of trials where the final selected path is unsafe (lower is better), **CS** is the number of trials where a safe path is confirmed with sufficiently high confidence prior to selection, **CN** is the number of trials where all multipaths are confirmed as unsafe (so no paths are selected).

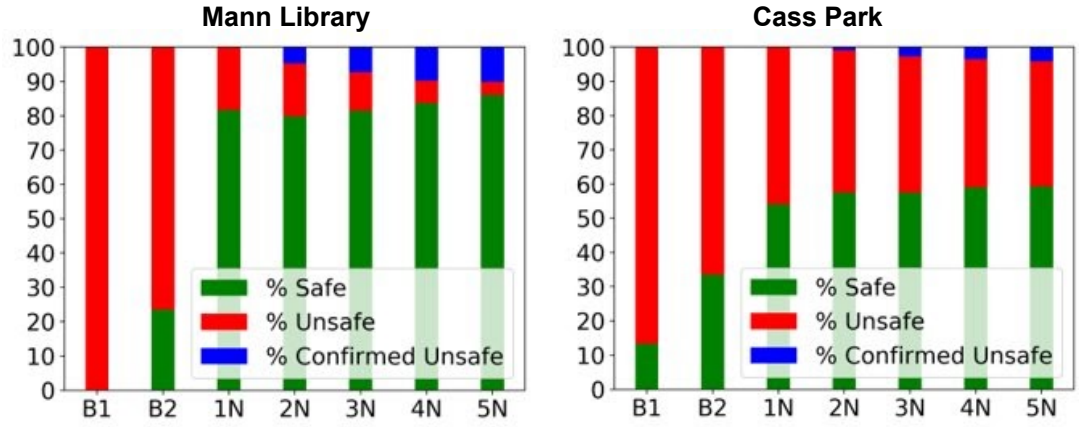


Figure 5.7: 500 trials of path safety evaluation. **B1** is the planner based on [79], **B2** includes semantic reasoning without any next-best-views (NBVs), and X**N** is DeepSemanticHPPC (ours) with X NBVs. Green represents trials where a safe path is selected; Red represents trials where an unsafe path is selected; and Blue represents trials where all paths are determined to be unsafe (so no paths are selected). More green and blue is better.

planner still plans over unsafe terrain.

Our DeepSemanticHPPC framework is significantly better than the two base-

lines. NBVs allow the planner to discard unsafe paths as semantic uncertainties decrease. With just one NBV, the percentage of safe paths taken increases drastically from 23.4% to 81.6% (Mann) and increases from 33.4% to 54.0% (Cass). As NBVs increase, the percentage of safe paths selected generally increases while the percentage of unsafe paths selected decreases. With 5 NBVs, 86% (59.2%) of paths selected for Mann (Cass) are safe, and only 3.8% (36.6%) of paths selected for Mann (Cass) are unsafe. With 5 NBVs, uncertainty is sufficiently reduced so that in 10.2% (4.2%) of trials, all multipaths are confirmed to be unsafe for Mann (Cass) and no path is selected. The complexity of the Cass scene is reflected in these results.

5.7 Discussion and Future Work

In this chapter, we presented DeepSemanticHPPC, a novel framework for planning in unstructured outdoor environments while accounting for uncertain terrain types. Our framework plans multiple feasible paths to a goal location and attempts to select a safely navigable path. For unstructured outdoor environments, existing planners have focused on assessing navigability via geometric constraints such as terrain steepness. Yet, some terrain types, such as mud or water, may be unsafe for traversal for robots and cannot be easily detected through environment geometry alone. In this work, we utilized a semantic segmentation model to perceive terrain and obstacles in the environment. However, the perception model may fail to accurately parse regions of the scene due to occlusions, unsuitable lighting, or unfamiliar features different from what was encountered during training. Therefore, we proposed to account for model uncertainties during planning, as highly uncertain regions may be correlated with incorrect

semantic predictions. To refine ambiguities, the framework focuses on selecting new views of the world to measure that reduce uncertainties along potential paths to the goal. Our real world experimental results showed that DeepSemanticHPPC reduces semantic uncertainty in planned paths and increases the safety of paths planned in environments with unsafe terrains. In particular, this framework greatly outperforms planners that focus only on terrain geometry, or planners that do not attempt to take intelligent measurements to reduce perception uncertainty along planned paths.

In the future, it would be interesting to implement DeepSemanticHPPC on a robot for real-time navigation experiments. In this work, we have assumed a point cloud map of the environment already exists to separate reasoning about semantics from noisy geometry. However, real-time exploration of an unknown environment requires online mapping and reconstruction as well. It would be interesting to explore the ability to build the point cloud online and incorporate scene geometry uncertainties into this pipeline.

Notes and Acknowledgements. We thank Eric Wu and Hadi AlZayer for insightful discussions and assistance with preliminary software implementation.

CHAPTER 6

CONCLUSION

For both scientists working at the cutting edge of research and consumers enjoying everyday technology, computer vision is playing an increasingly larger role in modern society. Given the diverse set of environments in which systems that rely on visual perception are deployed, it is pertinent to design systems that are able to perform well in different settings. In this dissertation, we explored several problems related to building such robust visual perception systems for real-world deployment.

In the first part of the dissertation, we explored efficient data annotation as a means of creating large-scale datasets that better represent the image distributions found in the real world. A salient challenge for robust deep learning in the real-world is in overcoming distribution shifts between the training data and the deployment environment. Large-scale data annotation is time-consuming and expensive, with high quality labels typically produced by skilled expert workers. We designed a method that is friendly to lower-skilled crowdworkers, while also producing annotations that can effectively supervise computer vision models.

In the next two parts of the dissertation, we explored paintings as an alternative image distribution from natural photographs for deep learning systems to be applied to and trained on. Paintings are created by humans for humans – as such, they represent an interesting collection of images to be analyzed, containing insights into both human culture and visual perception. We further explored the role of paintings as a source of data for learning better models intended for deployment in real world settings. As paintings are semantically meaningful without necessarily being photorealistic, they implicitly represent

semantics-preserving image transformations that are applied to some set of natural photographs. Learning from images altered by these transformations can encourage computer vision models to be invariant to such transformations. We studied the robustness of models trained on paintings and images transformed by style transfer (which are handcrafted or learned photo-to-painting transformations); we showed that real paintings induce different invariances in visual perception models than style transfer.

In the last part of the dissertation, we explored how noisy visual perception can be used as a building block in a larger system to guide an autonomous agent for path planning and safe navigation. Reasoning about model uncertainty and capturing new measurements of the environment to reduce uncertainty allows our proposed framework to plan safe paths in environments with initially unknown terrain types, despite potentially incorrect model predictions.

The problem of building robust perception systems for real-world environments is extremely challenging, and there are many fruitful directions to explore. As the amount of visual data in the world grows everyday, it is important to continue exploring methods to better take advantage of these images and videos. Which data samples can bring our training data distribution closer to the test distribution? Active learning and rare example mining can be used to find the most meaningful data samples to label. Better semi-automated annotation tools can ease the workload of workers who are tasked with high quality data annotation. Learning directly from unlabeled data without any human annotations through self-supervisory signals found in images or videos is a promising area of research; combining insights from unsupervised learning with supervised learning can lead to better semi-supervised frameworks. It is also useful to consider a variety

of image distributions during training, and to consider benchmarking models against a wider set of test distributions when designing new computer vision models. Domain generalization research has focused on both building algorithms that can learn from multiple distributions, as well as constructing meaningful benchmarks that better capture the wide variety of real world environments that perception models can be applied in. Robust features can be learned through various data augmentations or new model architectures with different inductive biases. Finally, I believe that continual learning, online characterizations of model uncertainties, and anomaly detection are important research directions to follow – models will inevitably face new, uncertain situations, and it is valuable for them to have the ability to adapt to new information and to provide accurate feedback about their confidence in new situations.

APPENDIX A

APPENDIX: EFFICIENT IMAGE ANNOTATION FOR SEMANTIC SEGMENTATION

This is an appendix for Chapter 2. In Sections A.1 and A.2, we detail the parameters used in our experiments to reproduce our results. In Section A.3, we include additional visualizations of crowdsourced annotations and inpainted labels.

A.1 Deeplabv3+ and Mobilenetv2

In our experiments, we use the open-source implementation of Deeplabv3+ found at https://github.com/tensorflow/models/tree/master/research/deeplab. For our comparison against weakly supervised methods, we use the open-source implementation of Mobilenetv2 found at https://github.com/tensorflow/models/tree/master/research/slim/nets/mobilenet.

A.1.1 Architecture

For Deeplabv3+, we use Xception65 [25] as the backbone. ASPP atrous rates are set to 6,12,18 for output stride 16 and 12,24,36 for output stride 8 as in [24]. Training uses output stride 16 while evaluation uses output stride 8. Mobilenetv2 does not use ASPP or decoder.

A.1.2 Training Procedure

Training hyperparameters are based on [24]. All hyperparameters use settings in [24] unless otherwise noted here. Batch normalization parameters are not finetuned due to low batch size (GPU memory constraints). Batch normalization parameters are initialized and frozen with the pretrained checkpoint. All Deeplabv3+ experiments initialize the network with the official pretrained model on ImageNet + MSCOCO + Pascal VOC and are trained for 100K iterations. Experiments with Mobilenetv2 initialize the network with the official Mobilenetv2 (width 1.0, input resolution 224) ImageNet model. We use polynomial learning rate decay as in [24]. During training time, we ignore the loss for pixels whose predicted class has a softmax probability greater than 0.95. This can be considered a form of hard negative mining where samples (pixels) with high confidence are ignored. Our preliminary experiments suggest that this appears to improve rate of convergence.

In any experiments that use fewer than 100% of the images in the training dataset, the images are shuffled and the first N images are selected. The shuffling order is determined once and fixed across experiments.

Cityscapes. Deeplabv3+. Learning rate 0.0005. Batch size 2 with input crop size 769×769.

ADE20K. Deeplabv3+. Learning rate 0.001. Batch size 4 with input crop size 513×513.

Pascal VOC. Mobilenetv2. Learning rate 0.001. Batch size 16 with input crop size 513×513. Batch norm parameters are finetuned.

A.1.3 Evaluation Procedure

After training for a fixed 100K iterations (no early stopping), models are tested on the validation set. 100K iterations is chosen based on settings in [24] which uses 90K iterations for Cityscapes. For consistency, 100K iterations is also used for ADE20K and Pascal VOC. Evaluation is performed at single scale without flipping on full resolution images. For completeness, we report here the validation mIOU when the network is trained out-of-the-box with the full training data using the outlined procedure.

Cityscapes. Validation images are padded to 1025×2049. Validation mIOU is 77.7%.

ADE20K. Validation images are padded to 513×513. Validation mIOU is 37.39%.

Pascal VOC. Validation images are padded to 513×513. Validation mIOU is 69.6%.

A.2 Block-Inpainting Model

The block-inpainting model is based on Deeplabv3+ with some architectural and training modifications.

A.2.1 Architectural Modifications

The input to the block-inpainting model is a tensor of shape $h \times w \times (3 + K)$ where K is the number of classes in the dataset (see main text). The weights of the first layer must be expanded to accommodate the K additional channels. These weights are initialized from a unit normal distribution. For each dataset, the weights are initialized once and then fixed for every experiment.

To compute uncertainty, dropout is added to the middle and exit flow blocks of the Xception65 backbone. The dropout keep probability is 0.8 as in [86]. No dropout is added to the decoder as preliminary experiments suggest that this will degrade performance.

A.2.2 Training Details

During training time, the block-inpainting model is trained with randomly drawn block hints from the set of block-annotated images (see main text). The block-inpainting model is trained for 100K iterations following section A.1. The entire set of block-annotated images are used as training targets.

A.2.3 Inference Details

At inference time, the block-inpainting model keeps dropout activated to estimate uncertainty [45]. The block-inpainting model outputs are averaged over 100 forward passes (100 is found to be sufficient in [44]) to form the final prediction. The uncertainty is computed by taking the sample variance of the softmax

probabilities for the predicted class (see main text).

A.3 Additional Visualizations

We show samples of crowdsourced annotations and block-inpainted labels.

A.3.1 Crowdsourced Annotations (SUNCG/CGIntrinsics)

Figure A.1 shows a sample of five annotated images. This figure is best viewed in color on screen with high zoom. See main text for annotation details.

For each image, segments from block annotation and segments from full annotation are shown. Synthetic labels are assigned using majority ground truth voting. Note that assigning synthetic labels in this way will cause detailed crowdsourced segmentations to be lost. See main text for estimate of cost to assign labels. For regions without segments, "void" label is assigned (color is black). For comparison, the dataset ground truth is shown in the final row. With block annotation, notice that workers segment small regions (e.g. the stool in image 1, row 2; the chair back in image 2, row 2; and the faucets in image 4, row 2) and oversegment regions (e.g. the cushions on the couch in image 3, row 1). With full annotation, notice that workers miss large regions, perhaps due to fatigue (e.g. the right window in image 3, row 4).

A.3.2 Crowdsourced Annotations (Cityscapes)

Figure A.2 shows a sample of three block-annotated images. This figure is best viewed in color on screen with high zoom. Regions that are not block-annotated are masked out in this visualization. At annotation time, the worker sees the entire image for context (see main text). See main text for annotation details.

For easier comparison against Cityscapes, crowdsourced segments are colored. Colors are random because class labels have not been assigned to crowdsourced segments (see main text for estimate of cost to assign labels). Crowdsourced segments with synthetic class labels are also included. Synthetic labels are assigned by taking the majority class label for the expert-labelled pixels in the crowdsourced segment. Note that assigning synthetic labels in this way will cause detailed crowdsourced segmentations to be lost. These visualizations are included to provide a better sense of the crowdsourced segments across the entire image rather than within individual blocks. Cityscapes segments are colored by class with "void" labels masked out.

In the main text, we compare the number of segments to expert full-image annotation. To compare the number of segments, we compute the number of label connected-components in expert annotations. Blocks with more than 50% void expert labels are ignored for a fair comparison.

A.3.3 Block-Inpainted Labels

We show set of samples of automatically assigned labels by the block-inpainting model (trained and tested with Block-50%) in figures A.3, A.4. For compari-

Figure A.1: SUNCG/CGIntrinsics Annotation Samples. Top to bottom: (Row 1) Crowdsourced blocks (boundaries). (Row 2) Crowdsourced blocks (synthetic labels). (Row 3) Crowdsourced full (boundaries). (Row 4) Crowdsourced full (synthetic labels). (Row 5) Ground truth. NOTE: Synthetic labels are the majority ground truth label for pixels in each segment. This means finely segmented crowdsourced segments (such as cushions on couches) will be lost in visualization. White dotted boxes highlight examples where block annotation qualitatively outperforms full annotation.

son, we show the human expert labels and the agreement between the block-inpainting model labels and the human labels (agreement is in white). With uncertainty threshold 0.2 on Cityscapes and 0.4 on ADE20K, over 94% of the pixels in the images are labelled by the block-inpainting model.

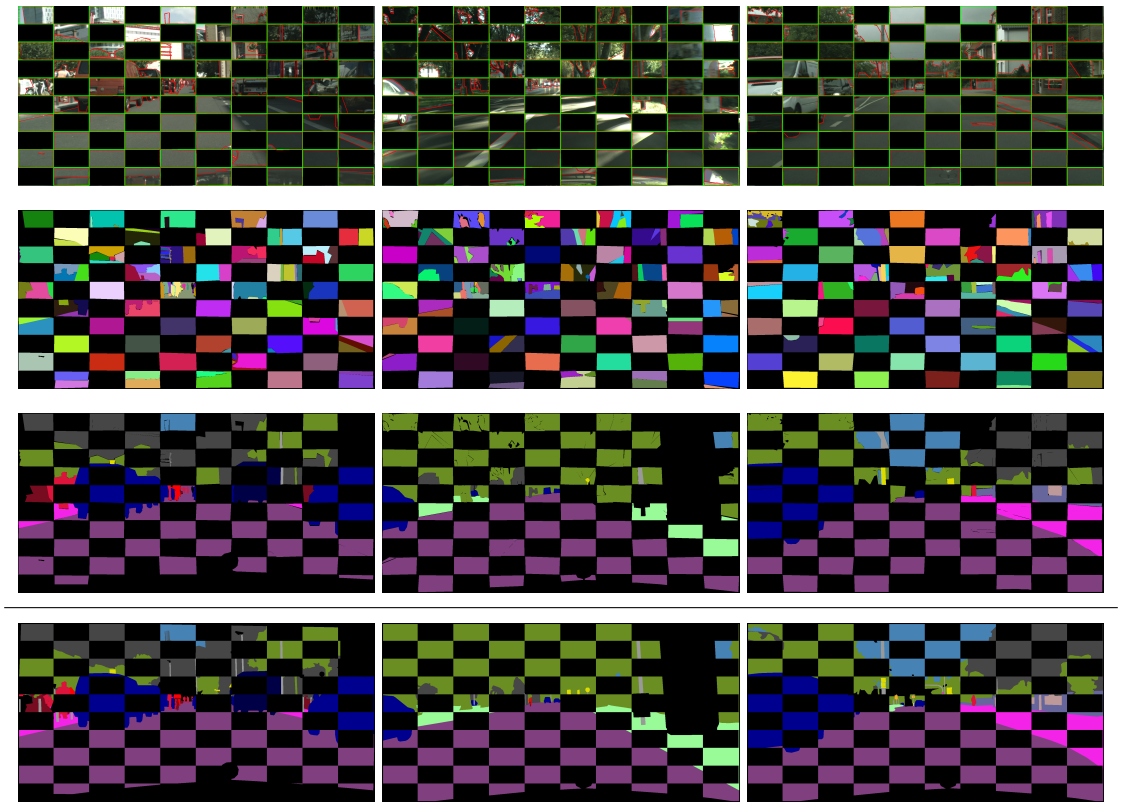


Figure A.2: Cityscapes Annotation Samples. Top to bottom: (Row 1) Crowdsourced (boundaries). (Row 2) Crowdsourced (randomly colored). (Row 3) Crowdsourced (synthetic labels). (Row 4) Expert Cityscapes. NOTE: Synthetic labels are the majority expert label for pixels in each segment. This means finely segmented crowdsourced segments (such as sky between leaves) will be lost in visualization.

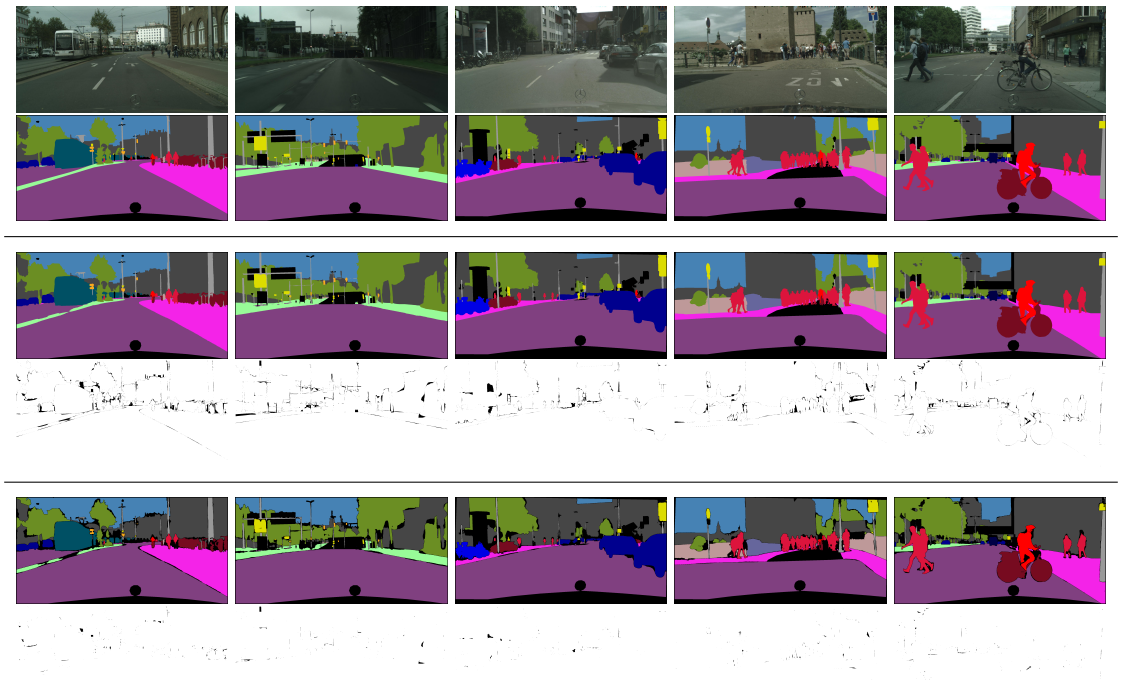


Figure A.3: Block-Inpainting Cityscapes Samples. Top to bottom: (Row 1) Original image. (Row 2) Human labels. (Row 3) Inpainted labels (all). (Row 4) Agreement (row 3 vs row 2). (Row 5) Inpainted labels (<20% relative uncertainty). (Row 6) Agreement (row 5 vs row 2). Void labels and rejected inpainted labels are masked out.

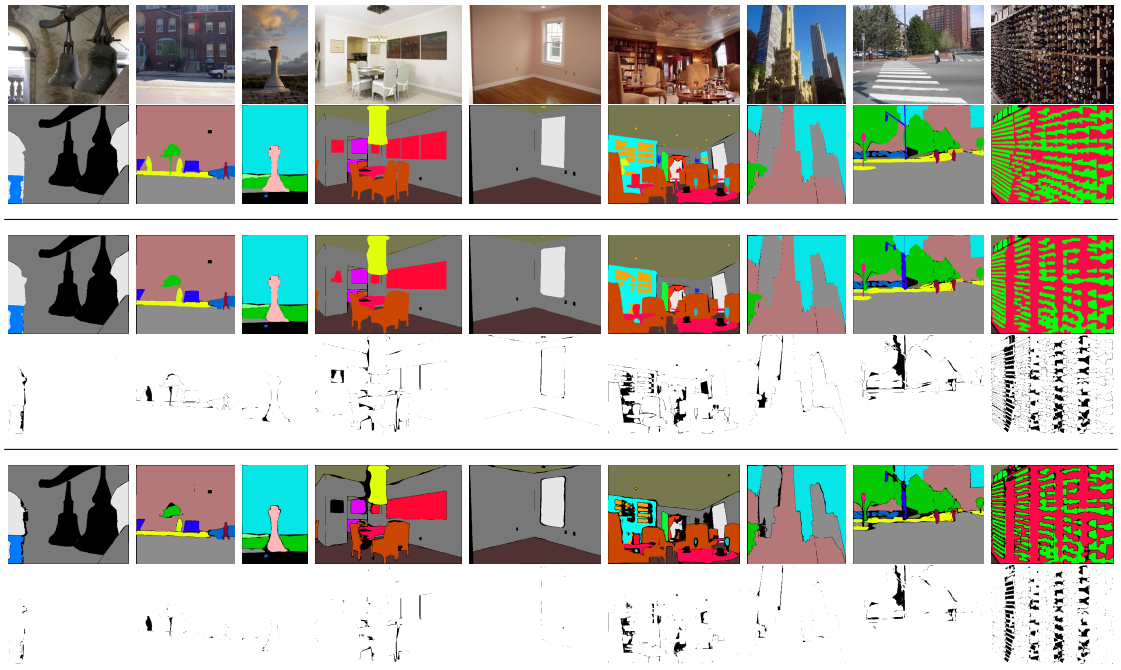


Figure A.4: Block-Inpainting ADE20K Samples. Top to bottom: (Row 1) Original image. (Row 2) Human labels. (Row 3) Inpainted labels (all). (Row 4) Agreement (row 3 vs row 2). (Row 5) Inpainted labels (<40% relative uncertainty). (Row 6) Agreement (row 5 vs row 2). Void labels and rejected inpainted labels are masked out.

APPENDIX B

APPENDIX: LEARNING ROBUST NATURAL IMAGE RECOGNITION FROM PAINTINGS

This is an appendix for Chapter 4. This appendix provides additional details to enable reproducibility, and additional visualizations and results to complement the main findings of the chapter. In Section B.1, we specify the creation of the Materials dataset. In Section B.2, we detail the experimental setup for the classification robustness experiments. In Section B.3, we describe the implementation and parameters used for style transfer, and we show visualizations of these methods in Section B.4. In Section B.5, we extend the discussion in Section 4.5 of the main text by analyzing the effect of stylization strength versus robustness. In Section B.7, we visualize the power spectra of stylized images and compare them to natural images. Finally, in Section B.8, we frame model robustness as domain generalization and discuss how domain-invariance can affect model robustness.

B.1 Materials Dataset Details

In the main text, we have briefly described the two primary datasets on which we focused our experiments. PACS [91] is a standard benchmark dataset while Materials is a novel dataset of photographs and paintings that was created by sampling image patches from existing datasets with material annotations. In this section, we give additional information on the creation of Materials. This dataset is released for reproducibility at https://github.com/hubertsgithub/style_painting_robustness.

Natural photographs. We acquired image patches from OpenSurfaces[10], COCO stuff [17], and MINC-2500 [11]. To create image patches for image classification from segmentation annotations, we constructed bounding boxes around segments, and cropped out these bounding boxes to form image patches. We constructed square bounding boxes with side length equal to 150% of the minimum side length of tight bounding box around the segment. Non-tight bounding boxes are used since it is important to include some context for the patch. We also sampled from MINC-2500 which already contains annotated image patches that do not require additional processing. Image crops that extend beyond the boundary of the full image are padded to square with ImageNet mean padding, and all final images patches are resized to 224×224. We sampled from OpenSurfaces and MINC first, before sampling from COCO if necessary. We created subsets of data of up to 60K photos, and each subset was created to be as-class-balanced-as-possible. For illustration, we provide per-class counts for two such subsets of data in Table B.1.

Natural-10K	Count	Natural-60K	Count
Ceramic	1000	Ceramic	3132**
Fabric	1000	Fabric	8006
Foliage	1000	Foliage	8006
Glass	1000	Glass	7216**
Liquid	1000	Liquid	7174**
Metal	1000	Metal	7204**
Paper	1000	Paper	3258**
Skin	1000	Skin	2276**
Stone	1000	Stone	5716**
Wood	1000	Wood	8006

Table B.1: Training datasets are sampled to be as class-balanced as possible. ** indicates that all training samples of that category are included in the training set, and no further samples exist. Natural-10K is a subset of Natural-60K. The test set contains 200 samples of each category.

Paintings. We sample paintings across the same material categories as above from [162]. We only sample patches that are at least 128×128 pixels in area to avoid very low-resolution annotations. The image patches are padded and resized in the same manner as above, and data is also sampled to be as-class-balanced-as-possible.

B.2 Classification Parameters

For all classification experiments, we use the following setup.

- Network architecture: ResNet18, ImageNet pretrained.
- Training hyperparameters: 30 epochs with initial learning rate (LR) 1e-3, LR reduced to 1e-4 at epoch 24. The LR of the classification layer is increased by 10×.
- Optimizer: SGD with 0.9 momentum.
- Training data augmentation: horizontal flipping, random scaling, color jitter, and ImageNet normalization.
- For experiments that train a model on both photos and stylized photos, all photos are stylized exactly once offline and included in the training set as an independent image from the original photo.
- Evaluation accuracies are averaged over 3 independent runs for each experiment.

Our results do not appear sensitive to the choice of training hyperparameters. Therefore, we train all networks with this configuration and evaluate the final model after training. Our experiments suggest that this training schedule

is sufficient for convergence without overfitting across all datasets we experimented with. Increasing training epochs to 100 or more does not improve results. Increasing number of training epochs is required if starting from random initialization, but ImageNet pretraining is standard practice so we do not extensively experiment with random initialization.

B.3 Style Transfer Parameters

For all applications of style transfer used in this work, we use pretrained models from publicly available implementations. The sources are provided here:

- AdaIN [68]: https://github.com/bethgelab/stylize-datasets
- ETNet [150]: https://github.com/zhijieW94/ETNet
- TPFR [155]: https://github.com/nnaisense/conditional-style-transfer
- SACL [140]: https://github.com/CompVis/adaptive-style-transfer

Our initial experiments showed that applying style transfer at 224×224 resolution yielded visually poor results (except for AdaIN). Therefore, we apply style transfer at a higher resolution and downsample the final result to 224×224. For AdaIN, ETNet, and SACL, we apply style transfer at 768×768 resolution. For TPFR, we apply style transfer at 512×512 instead of 768×768 due to GPU memory constraints. All other hyperparameters are set to the default settings found in the implementations for each respective method.

B.4 Visualizations of Stylized Photos

We show examples of images stylized by various style transfer methods on PACS (Fig. B.3) and Materials (Fig. B.4). The visualizations also include examples of intradomain stylization in which images are stylized by photos instead of by paintings. Notice that intradomain stylization yields stylizations that are, in general, visually similar to stylizations with painting style images. Overall, stylizations across all methods are holistically similar to natural paintings.

B.5 Style Distance vs Robustness

In Section 4.5, we found that arbitrary stylization with style images that share the same semantic content as the content image ("intraclass stylization") results in lower gains in robustness. Since images with similar semantic content may be more visually similar, this suggests that intraclass stylization will lead to less stylized images, i.e. weaker augmentation. To verify this, we measured style differences via the Gram matrix distance between stylized images and their original counterparts. Table B.2 summarizes differences on PACS. While intraclass stylization does result in smaller differences in style for each method, the Gram matrix distance across methods is not necessarily correlated with gains in robustness. For example, ETNet produces the largest style differences overall, but previous results in Fig. 4.3 show that AdaIN improves robustness more than ETNet on PACS. As such, the strength of stylization alone is not indicative of the downstream robustness learned by models trained on these images.

Method	Painting	Intradomain	Intradomain (Intraclass)
AdaIN	1.58±0.93	1.28±0.79	1.16±0.85
ETNet	2.33±1.09	2.13±1.04	1.81±1.03
TPFR	1.52±0.90	1.38±0.87	1.27±0.91

Table B.2: **Style (Gram Matrix) Distance.** Gram matrices computed from ImageNet pretrained ResNet18 features on PACS. Mean distance between `(image, stylized image)` pairs is reported. ↑ distance implies ↑ style difference. ± denotes standard deviation across 1.5K pairs.

B.6 Biases of Stylization Algorithms

As found in Section 4.5 of the main text, the choice of style images can affect the robustness gains from various arbitrary stylization methods. Beyond the choice of style images, the biases of style transfer models can have an impact on the final stylizations, and thus the robustness of models trained on these images. Therefore, we explore:

- **Hypothesis H1A:** Stylization biases from different methods affect model robustness differently.

To visualize the artifacts produced by style transfer, we stylize an image with itself as a style image in Fig. B.1. With a perfect decoder, this transformation should be the identity transformation. However, imperfections during encoding or decoding produce visible biases in the output image. For example, all of the models induce blurring, ETNet has prominent checkerboard artifacts, and TPFR induces a clear shift in color distribution. The result is that content images will be inevitably "stylized" by at least some fixed extent regardless of the style image with different biases from different stylization models. Table B.3 shows that training models on self-stylized images changes its robustness from training on

photos alone. Corruptions like blur and digital generally benefit from stylization biases while noise appears dependent on both the classification task (objects vs materials) and style transfer method.

Answer to H1A: *The biases encoded in different style transfer algorithms contributes to changes in model robustness, with different effects dependent on both style transfer algorithm and downstream classification task.*

Method	Noise	Blur	Weather	Digital
Materials (30K Samples/Domain)				
Photo-Only	43.71	58.76	55.25	61.20
Photo + AdaIN	39.98	60.75	56.72	61.38
Photo + ETNet	45.61	59.92	56.44	66.96
Photo + TPFR	47.53	64.00	55.07	69.02
PACS (1.5K Samples/Domain)				
Photo-Only	62.64	72.75	83.24	86.33
Photo + AdaIN	58.69	74.69	85.59	87.92
Photo + ETNet	55.80	71.16	82.41	87.42
Photo + TPFR	60.85	76.56	85.10	87.58

Table B.3: **Effect of Stylization Biases.** Per-corruption accuracy for models trained on photos plus photos stylized by themselves. Self-stylization reveals stylization biases in arbitrary style transfer models. Notice that the robustness of models differs between different style transfer methods when self-stylization is applied.

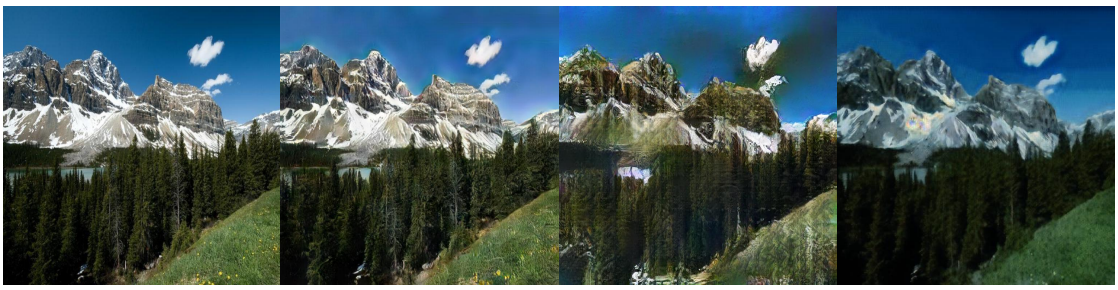


Figure B.1: **Arbitrary Stylization Biases**. Left to Right: Original image, Image stylized by itself using AdaIN, ETNet, and TPFR. Ideally, an image stylized by itself should not change. Notice that style transfer introduces artifacts, shifts in color, and other biases.

B.7 Power Spectra of Different Image Types

In Section 4.7 of the main text, we found that SACL improves robustness against noise with imperceptible high frequency signals in the stylized images. Here we show the power spectra of stylized images and compare them to the spectra for natural photos and natural paintings. The radial power spectrum for an image is computed as:

$$\text{power}(r) = ||X_r||^2$$
$$\text{where } X_r = \underset{\sqrt{i^2+j^2} \in \mathcal{R}(r)}{\text{mean}} ||X_{ij}||$$

X_{ij} are the frequency components given by the 2D discrete Fourier transform. Since (i, j) are discrete, the radial frequency component X_r is computed as an average over $||X_{ij}||$ for (i, j) that fall in a bin $\mathcal{R}(r)$. In Fig. B.2, we visualize the mean radial power spectra for natural photos, natural paintings, and SACL-stylized photos. We observe that stylized photos contain higher magnitude high-frequency components relative to natural photos and natural paintings. As noted in Section 4.7 of the main text, reducing the magnitude of sufficiently high-frequency components does not affect the perceptual quality of images.

B.8 Domain-Invariant Feature Learning

Our results from Section 4.6 of the main text provide evidence that models can learn more robust feature representations from the addition of paintings to a dataset of photographs. We can take this further by explicitly enforcing similar (or

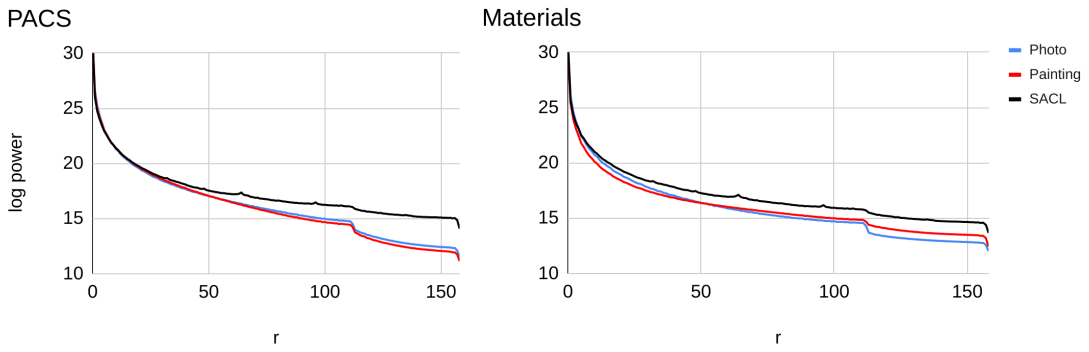


Figure B.2: **Power Spectrum of Images**. Left: PACS, Right: Materials. The plots depict the mean power spectrum for different sets of images. Photos stylized by SACL have larger magnitude high frequency components than natural photos or natural paintings.

domain-invariant) feature representations across photos and paintings. Domain-invariance is a common approach to the problem of domain generalization, where models are trained on multiple domains with the goal of generalizing to unseen domains, e.g. [51, 112]. In our setting, we can consider images with common corruptions to be the set of unseen domains. Perfect domain-invariant feature extraction can be harmful if it prevents useful features in photos from being extracted due to an underrepresentation of such features in paintings. Since the target task is recognition of photos, losing robust photo-specific signals can be detrimental. Therefore, we explore the following:

- **Hypothesis H2A:** Explicitly learning domain-invariance from paintings and photos may negatively impact model robustness.

We use an adversarial domain discriminator to learn domain invariant features [46, 112]. In Table B.4, we find that explicitly learning domain invariant features from paintings results in lower robustness than unrestricted feature learning with paintings. However, learning domain-invariant features does still

improve robustness over the photo-only baseline. Existing work in domain generalization has shown that domain-invariance is an effective method for learning to recognize images from unseen domains, e.g. [112]. Our finding here suggests that in the special case of domain generalization to corrupted versions of natural photographs, it is advantageous to retain photo-specific features for recognition. This is consistent with our hypothesis and discussion above – an underrepresentation of any particular photo-specific features in paintings can result in such features being ignored entirely when domain-invariance is enforced, even if such features are useful for robust recognition.

Answer to H2A: *Explicitly learning domain-invariant features from paintings negatively impacts model robustness with respect to unrestricted feature learning with paintings. However, domain-invariant features do still improve robustness relative to photos only.*

Method	MEAN	Noise	Blur	Weather	Digital
Materials (30K Samples/Domain)					
Photo-Only	54.73	43.71	58.76	55.25	61.20
Photo + Painting	**57.92**	**49.82**	**61.03**	**56.68**	**64.15**
Photo + Painting (DA)	55.99	46.97	59.60	54.51	62.90
PACS (1.5K Samples/Domain)					
Photo-Only	76.16	62.64	72.75	83.24	86.33
Photo + Painting	**78.99**	68.04	**74.72**	**86.26**	**86.92**
Photo + Painting (DA)	77.44	**68.86**	72.59	84.09	84.23

Table B.4: **Effect of Domain-Invariant Features.** "DA" refers to feature learning with an adversarial domain discriminator loss [46]. Learning domain-invariant features (red) reduces robustness relative to unrestricted feature learning from paintings (blue), but still improves robustness over photo-only.

B.9 Additional Architectures

We expect our findings to hold across architectures and datasets. As a sanity check, we have extended Table 4.2 with two additional architectures. The results (Table B.5) follow similar trends to those found in Table 4.2. For example, SACL outperforms both AdaIN and Paintings on Noise.

Resnet-18

Method	Noise	Blur	Weather	Digital
Materials (30K Samples/Domain)				
Photo-Only	43.70±0.65	58.76±0.14	55.25±0.33	61.20±0.69
Photo + AdaIN	47.33±0.22	65.09±0.21	61.78±0.18	61.41±0.16
Photo + SACL	61.87±0.16	64.36±0.20	57.49±0.24	66.55±0.17
Photo + Painting	49.82±0.56	61.03±0.13	56.69±0.10	64.15±0.14
PACS (1.5K Samples/Domain)				
Photo-Only	62.64±1.48	72.75±0.04	83.24±0.22	86.33±0.14
Photo + AdaIN	70.17±1.70	81.18±0.20	88.37±0.23	89.32±0.19
Photo + SACL	85.98±0.56	84.61±0.15	89.73±0.33	88.74±0.48
Photo + Painting	68.83±0.83	75.80±0.95	86.88±0.66	87.07±0.14

WideResnet-50-2

Method	Noise	Blur	Weather	Digital
Materials (30K Samples/Domain)				
Photo + AdaIN	57.80±1.79	73.77±0.11	67.75±0.51	66.96±0.06
Photo + SACL	69.39±0.72	70.00±0.34	64.00±0.54	73.05±0.30
Photo + Painting	60.72±0.83	68.09±0.49	61.15±0.23	70.98±0.24
PACS (1.5K Samples/Domain)				
Photo + AdaIN	82.05±1.33	86.89±0.64	93.98±0.15	94.39±0.30
Photo + SACL	93.79±1.35	89.64±0.36	95.19±0.17	93.63±0.11
Photo + Painting	83.92±1.81	85.38±0.27	94.19±0.08	92.63±0.24

Densenet-121

Method	Noise	Blur	Weather	Digital
Materials (30K Samples/Domain)				
Photo + AdaIN	54.32±0.23	71.08±0.24	67.31±0.37	66.47±0.13
Photo + SACL	67.22±0.16	68.89±0.16	63.08±0.33	71.87±0.62
Photo + Painting	54.83±1.20	68.21±0.38	61.29±0.39	70.66±0.13
PACS (1.5K Samples/Domain)				
Photo + AdaIN	76.96±4.12	85.79±0.50	94.96±0.13	92.34±0.19
Photo + SACL	91.33±0.28	88.92±0.37	94.18±0.49	94.12±0.54
Photo + Painting	76.65±2.22	83.22±0.19	94.00±0.62	91.72±0.14

Table B.5: **Per-Corruption Accuracy (Additional Architectures).** Trends across different architectures are generally consistent. For example, SACL (blue) greatly outperforms AdaIN and paintings (red) for noise robustness. ± indicates standard deviation over 3 runs.

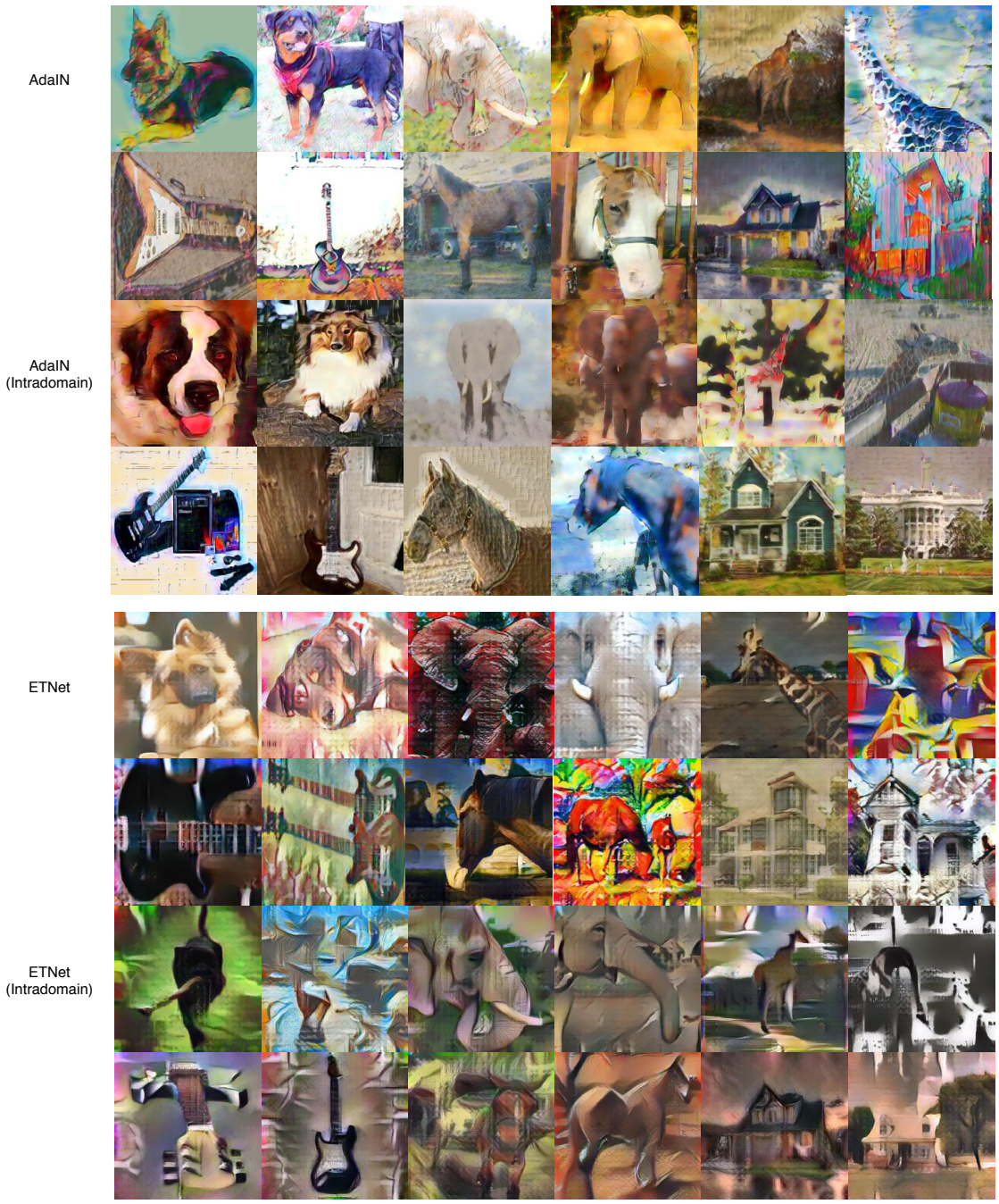


Figure B.3: **Stylized Photos (PACS) (1/2)**. Intradomain refers to stylization with photos as style images instead of paintings as style images. SACL is a learned style transfer method that is applied with different models pretrained to transfer the style of different artists. (*Continued on next page*)

Figure B.3: **Stylized Photos (PACS) (2/2)**. Intradomain refers to stylization with photos as style images instead of paintings as style images. SACL is a learned style transfer method that is applied with different models pretrained to transfer the style of different artists.

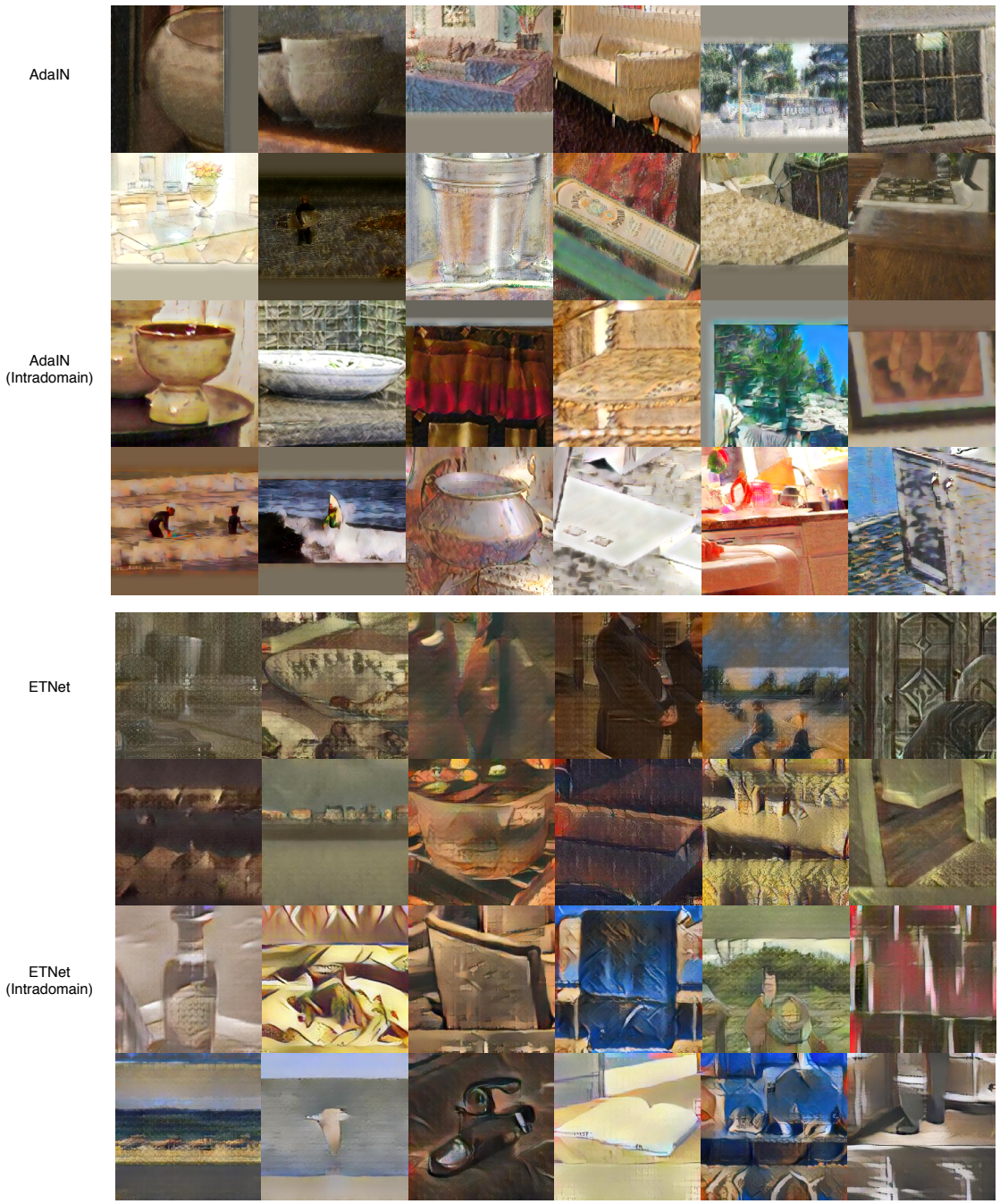


Figure B.4: **Stylized Photos (Materials) (1/2)**. Intradomain refers to stylization with photos as style images instead of paintings as style images. SACL is a learned style transfer method that is applied with different models pretrained to transfer the style of different artists. (*Continued on next page*)

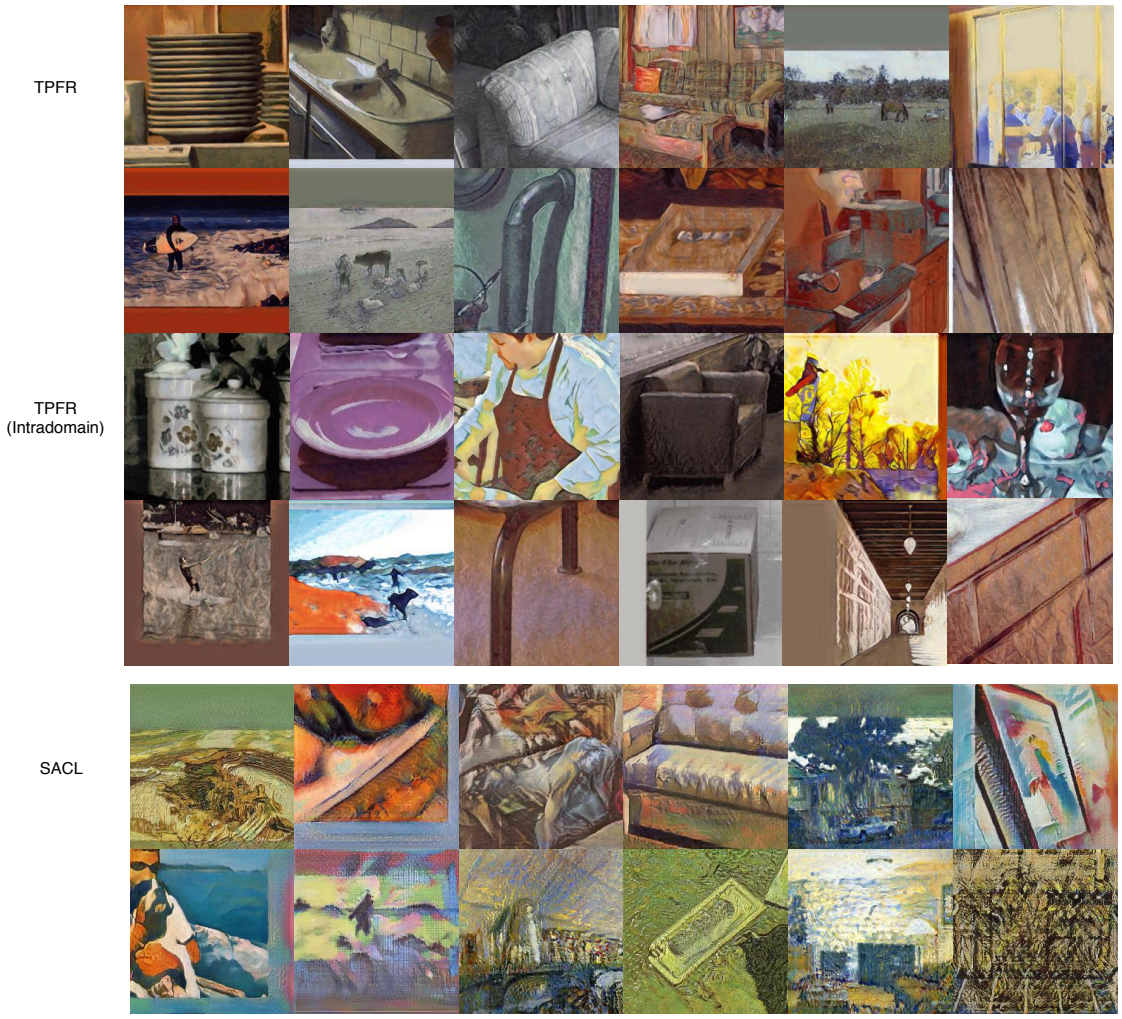


Figure B.4: **Stylized Photos (Materials) (2/2)**. Intradomain refers to stylization with photos as style images instead of paintings as style images. SACL is a learned style transfer method that is applied with different models pretrained to transfer the style of different artists.

APPENDIX C

APPENDIX: UNCERTAINTY-AWARE PLANNING WITH SEMANTIC SCENE UNDERSTANDING

This is an appendix for Chapter 5 which provides additional details for the semantic segmentation model used in the work. In Section C.1, we specify the construction of the outdoor semantic segmentation dataset. In Section C.2, we provide architecture, training, and inference details.

C.1 Outdoor Navigation Segmentation Dataset

As mentioned in the main text, we curate a segmentation dataset for outdoor navigation in unstructured environments from the existing large-scale COCO panoptic dataset [76]. Images of unstructured outdoor scenes are selected using a Places365 [191] classifier. An image is kept if (a) the classifier's top one (highest) prediction is an unstructured outdoor category with $> 50\%$ probability, or (b) two or more of the top five predictions are unstructured outdoor categories. Second, the 133 categories in COCO panoptic are merged: (a) all outdoor terrains (e.g. grass, dirt, snow, pavement) are retained; (b) obstacles are merged into four categories: fixed obstacles (e.g. buildings), moving human-made obstacles (e.g. vehicles), humans, and animals; (c) all indoor categories are removed. Our final dataset consists of 34K training images with 22 categories. Our navigation segmentation categories (from COCO panoptic), and filtered list of COCO panoptic images are at: `https://deepsemantichppc.github.io/segmentation_dataset/README.html`. The 22 semantic categories (plus one `VOID/IGNORED` class) are as follows:

rock-merged, gravel, platform, railroad, river, road, sand, sea, snow, pavement-merged, mountain-merged, grass-merged, dirt-merged, water-other, flower, tree-merged, playingfield, person, obstacles-fixed, obstacles-dynamic-manmade, obstacles-dynamic-animal, background, VOID

The obstacles-*, background, and VOID classes are formed by merging several categories in COCO panoptic:

```
OBSTACLES_FIXED (MERGED):
  stairs
  wall-brick
  wall-stone
  wall-tile
  wall-wood
  fire hydrant
  stop sign
  parking meter
  bench
  chair
  couch
  potted plant
  bed
  dining table
  toilet
  cardboard
  counter
```

```
  curtain
  door-stuff
  house
  shelf
  fence-merged
  table-merged
  building-other-merged
  wall-other-merged
  net
  cabinet-merged
OBSTACLES_DYNAMIC_MANMADE (MERGED):
  skateboard
  surfboard
  bicycle
  car
  airplane
  bus
  train
  truck
  boat
  skis
  snowboard
  motorcycle
OBSTACLES_DYNAMIC_ANIMAL (MERGED):
  bird
  cat
```

```
  dog
  horse
  sheep
  cow
  elephant
  bear
  zebra
  giraffe
BACKGROUND (MERGED):
  sky-other-merged
VOID (MERGED):
  VOID
  teddy bear
  kite
  oven
  keyboard
  floor-other-merged
  cup
  tv
  backpack
  orange
  laptop
  blanket
  vase
  spoon
  banner
```

towel

baseball glove

frisbee

paper-merged

window-blind

clock

microwave

ceiling-merged

knife

tent

umbrella

toothbrush

sink

floor-wood

baseball bat

sports ball

roof

pillow

apple

bridge

tennis racket

scissors

traffic light

banana

fork

donut

suitcase

cake

wine glass

carrot

mouse

hair drier

food-other-merged

rug-merged

toaster

bowl

book

tie

pizza

sandwich

fruit

handbag

cell phone

broccoli

refrigerator

mirror-stuff

window-other

remote

hot dog

light

bottle

C.2 Network Architecture, Training, and Inference

For our network architecture, we use DeepLabv3+[24] with Xception65 [25] backbone augmented with dropout in the middle and exit flow blocks for semantic segmentations. For details, refer to [24, 25]. ASPP atrous rates are set to 6,12,18 for output stride 16 and 12,24,36 for output stride 8 as in [24]. Training uses output stride 16 while inference uses output stride 8.

Training Details. We initialize from a network pretrained on ImageNet [33] + MSCOCO [99] + Pascal VOC [39]. This pretrained model is available at `https://github.com/tensorflow/models/blob/master/research/deeplab/g3doc/model_zoo.md`. The model is trained for 160000 iterations using SGD with momentum with batchsize 4 on our dataset. Initial learning rate 0.01, momentum 0.9, and polynomial learning rate decay with power 0.9 are used. Images are resized to 513x513.

Inference. At inference time, 50 forward passes are used to predict semantics and uncertainties (Eqs. (5.1)-(5.2)).

BIBLIOGRAPHY

[1] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 859–868, 2018. 14, 25

[2] Edward H Adelson. On seeing stuff: the perception of materials by humans and machines. In *Human vision and electronic imaging VI*, volume 4299, pages 1–12. International Society for Optics and Photonics, 2001. 68

[3] Eirikur Agustsson, Jasper RR Uijlings, and Vittorio Ferrari. Interactive full image segmentation by considering all regions jointly. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11622–11631, 2019. 13, 14, 25

[4] Jiwoon Ahn and Suha Kwak. Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation. *arXiv preprint arXiv:1803.10464*, 2018. 14

[5] Yağız Aksoy, Tae-Hyun Oh, Sylvain Paris, Marc Pollefeys, and Wojciech Matusik. Semantic soft segmentation. *ACM Trans. Graph. (Proc. SIGGRAPH)*, 37(4):72:1–72:13, 2018. 14

[6] Mykhaylo Andriluka, Jasper RR Uijlings, and Vittorio Ferrari. Fluid annotation: a human-machine collaboration interface for full image annotation. *arXiv preprint arXiv:1806.07527*, 2018. 14

[7] Isabella Augart, Maurice Sa, and Iris Wenderholm. *Steinformen*. De Gruyter, Berlin, Boston, 2018. 46

[8] Xue Bai and Guillermo Sapiro. Geodesic matting: A framework for fast interactive image and video segmentation and matting. *International journal of computer vision*, 82(2):113–132, 2009. 13

[9] Amy Bearman, Olga Russakovsky, Vittorio Ferrari, and Li Fei-Fei. What's the point: Semantic segmentation with point supervision. In *European Conference on Computer Vision (ECCV)*, pages 549–565. Springer, 2016. 14, 24, 29, 30

[10] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. OpenSurfaces:A

Richly Annotated Catalog of Surface Appearance. *ACM Transactions on Graphics*, 32(4):1, 2013. xiv, 11, 13, 15, 18, 19, 20, 22, 44, 45, 68, 126

[11] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Material recognition in the wild with the Materials in Context database. *Computer Vision and Pattern Recognition (CVPR)*, 2015. 14, 24, 44, 68, 93, 126

[12] Rodrigo Benenson, Stefan Popov, and Vittorio Ferrari. Large-scale interactive object segmentation with human annotators. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11700–11709, 2019. 45

[13] Willem Beurs. *"De groote waereld in't kleen geschildert, of Schilderagtig tafereel van's weerelds schilderyen. Kortelijk vervat in ses boeken: Verklarende de hooftverwen, haare verscheide mengelinge in oly, en der zelver gebruik..."*. By Johannes en Gillis Janssonius van Waesberge, 1692. 42

[14] Marjolijn Bol and Ann-Sophie Lehmann. Painting skin and water: towards a material iconography of translucent motifs in early netherlandish painting. In *Rogier van der Weyden in context: papers presented at the Seventeenth Symposium for the Study of Underdrawing and Technology in Painting held in Leuven, 22-24 October*, pages 215–228. Peeters, 2012. 46

[15] R. Border, J. D. Gammell, and P. Newman. Surface edge explorer (SEE): Planning next best views directly from 3d observations. In *Proc. ICRA*, pages 6116–6123, 2018. 90

[16] Ali Sharifi Boroujerdi, Maryam Khanian, and Michael Breuß. Deep interactive region segmentation and captioning. In *Signal-Image Technology & Internet-Based Systems (SITIS), 2017 13th International Conference on*, pages 103–110. IEEE, 2017. 13

[17] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. COCO-Stuff: Thing and stuff classes in context. In *Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018. 11, 13, 20, 24, 32, 33, 34, 44, 45, 68, 93, 126

[18] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On evaluating adversarial robustness. *arXiv preprint arXiv:1902.06705*, 2019. 64

[19] Patrick Cavanagh. The artist as neuroscientist. *Nature*, 434(7031):301–307, 2005. 6, 40, 51, 74

[20] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 93

[21] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7354–7362, 2019. 56

[22] R. Omar Chavez-Garcia, Jérôme Guzzi, Luca Maria Gambardella, and Alessandro Giusti. Learning ground traversability from simulations. *IEEE RAL*, 3:1695–1702, 2017. 90

[23] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 30

[24] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 26, 93, 99, 115, 116, 117, 147

[25] François Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, pages 1610–02357, 2017. 99, 115, 147

[26] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. Meshlab: an open-source mesh processing tool. In *Eurographics Italian chapter conference*, volume 1, pages 129–136, 2008. 108

[27] Edo Collins, Raja Bala, Bob Price, and Sabine Susstrunk. Editing in style: Uncovering the local semantics of gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5771–5780, 2020. 87

[28] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes dataset for semantic urban scene understanding. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016. x, 5, 11, 13, 20, 22, 30, 45

[29] Elliot J. Crowley and Andrew Zisserman. The state of the art: Object retrieval in paintings using discriminative regions. In *British Machine Vision Conference*, 2014. 42

[30] Jifeng Dai, Kaiming He, and Jian Sun. Boxsup: Exploiting bounding boxes to supervise convolutional networks for semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1635–1643, 2015. 29, 30

[31] Jonathan Daudelin and Mark Campbell. An adaptable, probabilistic, next best view algorithm for reconstruction of unknown 3d objects. *IEEE RAL*, 2(3):1540–1547, 2017. 90

[32] Jeffrey A. Delmerico, Alessandro Giusti, Elias Mueggler, Luca Maria Gambardella, and Davide Scaramuzza. "on-the-spot training" for terrain classification in autonomous air-ground collaborative teams. In *Proc. ISER*, 2016. 90

[33] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition (CVPR)*, pages 248–255. IEEE, 2009. 26, 147

[34] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 63, 65

[35] Francesca Di Cicco, Maarten W.A. Wijntjes, and Sylvia C. Pont. Understanding gloss perception through the lens of art: Combining perception, image analysis, and painting recipes of 17th century painted grapes. *Journal of Vision*, 19(3):1–15, 2019. 51

[36] RV Dietrich. Rocks depicted in painting & sculpture. *Rocks & Minerals*, 65(3):224–236, 1990. 46

[37] Zeyad Emam, Andrew Kondrich, Sasha Harrison, Felix Lau, Yushi Wang, Aerin Kim, and Elliot Branson. On the state of data in computer vision: Human annotations remain indispensable for developing deep learning models. *arXiv preprint arXiv:2108.00114*, 2021. 4

[38] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. KDD*, pages 226–231, 1996. 97

[39] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision (IJCV)*, 88(2):303–338, June 2010. 26, 30, 147

[40] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 44, 46

[41] Dave Ferguson and Anthony Stentz. Using interpolation to improve path planning: The field D* algorithm. *J Field Robot*, 23:79–101, 2006. 90

[42] Paul Filitchkin. *Visual Terrain Classification For Legged Robots*. PhD thesis, University of California, Santa Barbara, 2011. 90

[43] Roland W Fleming. Material Perception. *Annual Reviews*, 3:365–88, 2017. 40

[44] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015. 32, 118

[45] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. 32, 90, 94, 118

[46] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015. xiii, 133, 134

[47] Alberto Garcia-Garcia, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez, Pablo Martinez-Gonzalez, and Jose Garcia-Rodriguez. A survey on deep learning techniques for image and video semantic segmentation. *Applied Soft Computing*, 70:41–65, 2018. 41

[48] Leon A Gatys, Alexander S Ecker, Matthias Bethge, Aaron Hertzmann, and Eli Shechtman. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3985–3993, 2017. 87

[49] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased

towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018. 50, 61, 63, 65, 68, 69

[50] Robert Geirhos, Carlos RM Temme, Jonas Rauber, Heiko H Schütt, Matthias Bethge, and Felix A Wichmann. Generalisation in humans and deep neural networks. In *Advances in neural information processing systems*, pages 7538–7550, 2018. 64, 65

[51] Ishaan Gulrajani and David Lopez-Paz. In search of lost domain generalization. *arXiv preprint arXiv:2007.01434*, 2020. 65, 133

[52] Saurabh Gupta, Pablo Arbelaez, and Jitendra Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 564–571, 2013. 19

[53] Timo Hackel, N. Savinov, L. Ladicky, J. Wegner, K. Schindler, and M. Pollefeys. SEMANTIC3D.NET: A new large-scale point cloud classification benchmark. *ISPRS Annals*, 4(1):91–98, 2016. 93

[54] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. A survey on instance segmentation: state of the art. *International Journal of Multimedia Information Retrieval*, pages 1–19, 2020. 41

[55] Yutao Han, Hubert Lin, Jacopo Banfi, Kavita Bala, and Mark Campbell. Deepsemantichppc: Hypothesis-based planning over uncertain semantic point clouds. *ICRA*, 2020. 7, 89

[56] Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Chris Callison-Burch, and Jeffrey P Bigham. A data-driven analysis of workers' earnings on amazon mechanical turk. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, page 449. ACM, 2018. 22

[57] Kotaro Hara, Abigail Adams, Kristy Milland, Saiph Savage, Benjamin V Hanrahan, Jeffrey P Bigham, and Chris Callison-Burch. Worker demographics and earnings on amazon mechanical turk: An exploratory analysis. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*, page LBW1217. ACM, 2019. 23

[58] Mohammed Hassanin, Salman Khan, and Murat Tahtali. Visual affordance and function understanding: A survey. *arXiv preprint arXiv:1807.06775*, 2018. 11, 32

[59] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. 2

[60] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019. 62, 64, 66

[61] Dan Hendrycks, Norman Mu, Ekin D Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019. 63, 65

[62] Benjamin Hepp, Debadeepta Dey, Sudipta N. Sinha, Ashish Kapoor, Neel Joshi, and Otmar Hilliges. Learn-to-score: Efficient 3d scene exploration by predicting view utility. In *Proc. ECCV*, pages 437–452, 2018. 90

[63] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 3

[64] Seunghoon Hong, Donghun Yeo, Suha Kwak, Honglak Lee, and Bohyung Han. Weakly supervised semantic segmentation using web-crawled videos. *arXiv preprint arXiv:1701.00352*, 2017. 14

[65] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 30

[66] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. *Cornell University arXiv Institution: Ithaca, NY, USA*, 2017. 14

[67] Eric Huang, Haoqi Zhang, David C Parkes, Krzysztof Z Gajos, and Yiling Chen. Toward automatic task design: a progress report. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 77–85. ACM, 2010. 17, 23

[68] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017. 65, 69, 128

[69] Alexander Ivanov and Mark E. Campbell. Uncertainty constrained robotic exploration: An integrated exploration planner. *IEEE T Contr Syst T*, 27:146–160, 2016. 89

[70] Xin Jin, Cuiling Lan, Wenjun Zeng, Zhibo Chen, and Li Zhang. Style normalization and restitution for generalizable person re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3143–3152, 2020. 65

[71] Yongcheng Jing, Yezhou Yang, Zunlei Feng, Jingwen Ye, Yizhou Yu, and Mingli Song. Neural style transfer: A review. *IEEE transactions on visualization and computer graphics*, 2019. 50, 61, 63, 69

[72] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019. 56

[73] Martin Kemp et al. The science of art: Optical themes in western art from brunelleschi to seurat. 1990. 42

[74] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Proc. NIPS*, pages 5574–5584, 2017. 90, 94

[75] Anna Khoreva, Rodrigo Benenson, Jan Hendrik Hosang, Matthias Hein, and Bernt Schiele. Simple does it: Weakly supervised instance and semantic segmentation. In *CVPR*, volume 1, page 3, 2017. 14

[76] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proc. CVPR*, pages 9404–9413, 2019. 93, 98, 141

[77] Dmytro Kotovenko, Artsiom Sanakoyeu, Sabine Lang, and Bjorn Ommer. Content and style disentanglement for artistic style transfer. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4422–4431, 2019. 65

[78] Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105, 2012. 1

[79] Philipp Krüsi, Paul Furgale, Michael Bosse, and Roland Siegwart. Driving on point clouds: Motion planning, trajectory optimization, and terrain assessment in generic nonplanar environments. *J Field Rob*, 34(5):940–984, 2017. xii, xvii, 90, 91, 92, 93, 100, 108, 109

[80] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018. 44

[81] Nikolay Laptev, Jason Yosinski, Li Erran Li, and Slawek Smyl. Time-series extreme event forecasting with neural networks at uber. In *International Conference on Machine Learning*, number 34, pages 1–5, 2017. 33

[82] Steven M. LaValle. Rapidly-exploring random trees: A new tool for path planning. 1998. 92

[83] Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, New York, NY, USA, 2006. 89

[84] Hoang Le, Long Mai, Brian Price, Scott Cohen, Hailin Jin, and Feng Liu. Interactive boundary prediction for object selection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 18–33, 2018. 13, 14

[85] Ann-Sophie Lehmann. Fleshing out the body: The'colours of the naked'in workshop practice and art theory, 1400-1600. *Nederlands Kunsthistorisch Jaarboek*, 59:86, 2008. 46

[86] Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific reports*, 7(1):17816, 2017. 33, 118

[87] Victor Lempitsky and Andrew Zisserman. Learning to count objects in images. In *Advances in neural information processing systems*, pages 1324–1332, 2010. 32

[88] Anat Levin, Dani Lischinski, and Yair Weiss. A closed-form solution to natural image matting. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):228–242, 2008. 13

[89] Anat Levin, Alex Rav-Acha, and Dani Lischinski. Spectral matting. *IEEE*

transactions on pattern analysis and machine intelligence, 30(10):1699–1712, 2008. 13

[90] Boyi Li, Felix Wu, Ser-Nam Lim, Serge Belongie, and Kilian Q Weinberger. On feature normalization and data augmentation. *arXiv preprint arXiv:2002.11102*, 2020. 70

[91] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017. 50, 65, 68, 125

[92] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M Hospedales. Episodic training for domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1446–1455, 2019. 65

[93] Yanghao Li, Naiyan Wang, Jiaying Liu, and Xiaodi Hou. Demystifying neural style transfer. *arXiv preprint arXiv:1701.01036*, 2017. 63

[94] Zhengqi Li and Noah Snavely. Cgintrinsics: Better intrinsic image decomposition through physically-based rendering. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–387, 2018. 19

[95] Di Lin, Jifeng Dai, Jiaya Jia, Kaiming He, and Jian Sun. Scribblesup: Scribble-supervised convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3159–3167, 2016. x, 14, 29, 30

[96] Hubert Lin, Paul Upchurch, and Kavita Bala. Block annotation: Better image annotation with sub-image decomposition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5290–5300, 2019. 5, 10, 45

[97] Hubert Lin, Mitchell van Zuijlen, Sylvia C Pont, Maarten WA Wijntjes, and Kavita Bala. What can style transfer and paintings do for model robustness? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11028–11037, 2021. 6, 7, 62

[98] Hubert Lin, Mitchell Van Zuijlen, Maarten WA Wijntjes, Sylvia C Pont, and Kavita Bala. Insights from a large-scale database of material depictions in paintings. In *International Conference on Pattern Recognition*, pages 531–545. Springer, 2021. 5, 6, 41

[99] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014. 11, 13, 22, 26, 44, 45, 46, 147

[100] Zhiqiu Lin, Jin Sun, Abe Davis, and Noah Snavely. Visual chirality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12295–12303, 2020. 41

[101] Huan Ling, Jun Gao, Amlan Kar, Wenzheng Chen, and Sanja Fidler. Fast interactive object annotation with curve-gcn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5257–5266, 2019. 45

[102] Aishan Liu, Xianglong Liu, Chongzhi Zhang, Hang Yu, Qiang Liu, and Junfeng He. Training robust deep neural networks via adversarial noise propagation. *arXiv preprint arXiv:1909.09034*, 2019. 62, 63, 65, 83

[103] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015. 56

[104] Khairul Azmi Mahadhir, Shing Chiang Tan, Cheng Yee Low, Roman Dumitrescu, Adam Tan Mohd Amin, and Ahmed Jaffar. Terrain classification for track-driven agricultural robots. In *Proc. SYSINT*, pages 775 – 782, 2014. 90

[105] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens Van Der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European conference on computer vision (ECCV)*, pages 181–196, 2018. 3

[106] Utkarsh Mall, Kevin Matzen, Bharath Hariharan, Noah Snavely, and Kavita Bala. Geostyle: Discovering fashion trends and events. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 411–420, 2019. 41

[107] Pascal Mamassian. Ambiguities and conventions in the perception of visual art. *Vision Research*, 48(20):2143–2153, 2008. 6, 40, 82

[108] Pascal Mamassian. Ambiguities and conventions in the perception of visual art. *Vision Research*, 48(20):2143–2153, 2008. 51

[109] R. Manduchi, A. Castano, A. Talukder, and L. Matthies. Obstacle detection and terrain classification for autonomous off-road navigation. *Auton Robot*, 18(1):81–102, 2005. 90

[110] Kevis-Kokitsi Maninis, Sergi Caelles, Jordi Pont-Tuset, and Luc Van Gool. Deep extreme cut: From extreme points to object segmentation. *arXiv preprint arXiv:1711.09081*, 2017. x, 13, 25, 45, 47

[111] Pat Marion, Peter R Florence, Lucas Manuelli, and Russ Tedrake. A pipeline for generating ground truth labels for real rgbd data of cluttered scenes. *arXiv preprint arXiv:1707.04796*, 2017. 13

[112] Toshihiko Matsuura and Tatsuya Harada. Domain generalization using a mixture of multiple latent domains. In *AAAI*, pages 11749–11756, 2020. 65, 133, 134

[113] Kevin Matzen, Kavita Bala, and Noah Snavely. Streetstyle: Exploring world-wide clothing styles from millions of photos. *arXiv preprint arXiv:1706.01869*, 2017. 41

[114] M.H. van Eikema Hommes. The contours in the paintings of the oranjezaal, huis ten bosch. 2005. 45

[115] Branislav Mičušĺík and Jana Košecká. Semantic segmentation of street scenes by superpixel co-occurrence and 3d geometry. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 625–632. IEEE, 2009. 32

[116] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 891–898, 2014. 11, 20, 30

[117] B. Nagy and A. Kelly. Trajectory generation for car-like robots using cubic curvature polynomials. In *Proc. FSR*, 2001. 93

[118] Gerhard Neuhold, Tobias Ollmann, S Rota Bulo, and Peter Kontschieder. The Mapillary Vistas dataset for semantic understanding of street scenes. In *International Conference on Computer Vision (ICCV)*, pages 22–29, 2017. 13

[119] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter

Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *ICCV*, pages 5000–5009, 2017. 11, 22

[120] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 1(10):e3, 2016. 83

[121] Michael W. Otte, Scott G. Richardson, Jane Mulligan, and Gregory Z. Grudic. Path planning in image space for autonomous robot navigation in unstructured environments. *J Field Robot*, 26:212–240, 2009. 90

[122] Erwin Panofsky. *Perspective as symbolic form*. Princeton University Press, 1927/2020. 42

[123] Dim P. Papadopoulos, Jasper R. R. Uijlings, Frank Keller, and Vittorio Ferrari. Extreme clicking for efficient object annotation. *International Journal of Computer Vision*, 2017. x, 43, 46, 47

[124] Dim P Papadopoulos, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari. Extreme clicking for efficient object annotation. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 4940–4949. IEEE, 2017. 13

[125] George Papandreou, Liang-Chieh Chen, Kevin P Murphy, and Alan L Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1742–1750, 2015. 14, 29, 30

[126] Deepak Pathak, Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional multi-class multiple instance learning. *arXiv preprint arXiv:1412.7144*, 2014. 29, 30

[127] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019. 50, 56, 79

[128] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017. 79

[129] Maurice Henri Pirenne. *Optics, painting & photography*. Cambridge University Press. 42

[130] Carol Pottasch. Frans van mieriss painting technique as one of the possible sources for willem beurss treatise on painting. *Art & Perception*, pages 1 – 17, 13 Jun. 2020. 42

[131] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proc. CVPR*, pages 918–927, 2018. 93

[132] Charles Ruizhongtai Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. CVPR*, pages 77–85, 2017. 93

[133] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proc. NIPS*, pages 5099–5108, 2017. 93

[134] Xuebin Qin, Shida He, Zichen Zhang, Masood Dehghan, and Martin Jagersand. Bylabel: A boundary based semi-automatic image annotation tool. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1804–1813. IEEE, 2018. 14

[135] Alexander J Quinn and Benjamin B Bederson. Human computation: a survey and taxonomy of a growing field. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 1403–1412. ACM, 2011. 20

[136] Stephan Rabanser, Stephan Günnemann, and Zachary C Lipton. Failing loudly: An empirical study of methods for detecting dataset shift. *arXiv preprint arXiv:1810.11953*, 2018. 3

[137] Tal Remez, Jonathan Huang, and Matthew Brown. Learning to segment via cut-and-paste. *CoRR*, abs/1803.06414, 2018. 14

[138] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 47

[139] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM transactions on graphics (TOG)*, volume 23, pages 309–314. ACM, 2004. 13, 45

[140] Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Bjorn Ommer.

A style-aware content loss for real-time hd style transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 698–714, 2018. 65, 80, 128

[141] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4510–4520, 2018. 30

[142] Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE international conference on computer vision*, pages 618–626, 2017. 52

[143] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and Service Robotics*, 2017. 105

[144] Lavanya Sharan, Ce Liu, Ruth Rosenholtz, and Edward H. Adelson. Recognizing materials using perceptually inspired features. *International Journal of Computer Vision*, 103(3):348–371, 2013. 65, 67

[145] Tong Shen, Guosheng Lin, Chunhua Shen, and Ian Reid. Bootstrapping the performance of webly supervised semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1363–1371, 2018. 14

[146] Lu Sheng, Ziyi Lin, Jing Shao, and Xiaogang Wang. Avatar-net: Multi-scale zero-shot style transfer by feature decoration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8242–8250, 2018. 65

[147] Zhiyuan Shi, Yongxin Yang, Timothy M Hospedales, and Tao Xiang. Weakly-supervised image annotation and segmentation with objects and attributes. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2525–2538, 2017. 14

[148] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019. 4

[149] Nathan Somavarapu, Chih-Yao Ma, and Zsolt Kira. Frustratingly simple domain generalization via image stylization. *arXiv preprint arXiv:2006.11207*, 2020. 71

[150] Chunjin Song, Zhijie Wu, Yang Zhou, Minglun Gong, and Hui Huang. Etnet: Error transition network for arbitrary style transfer. *arXiv preprint arXiv:1910.12056*, 2019. 65, 69, 128

[151] Shuran Song, Fisher Yu, Andy Zeng, Angel X Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. *Proceedings of 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 19

[152] Gregory J. Stein, Christopher Bradley, and Nicholas Roy. Learning over subgoals for efficient navigation of structured, unknown environments. In *Proc. CORL*, pages 213–222, 2018. 89

[153] David Summers. Conventions in the history of art. *New literary history*, 13(1):103–125, 1981. 82

[154] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017. 3

[155] Jan Svoboda, Asha Anoosheh, Christian Osendorfer, and Jonathan Masci. Two-stage peer-regularized feature recombination for arbitrary image style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13816–13825, 2020. 65, 69, 128

[156] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *International Conference on Learning Representations (ICLR)*, 2014. 2

[157] Wei Ren Tan, Chee Seng Chan, Hernán E Aguirre, and Kiyoshi Tanaka. Artgan: Artwork synthesis with conditional categorical gans. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3760–3764. IEEE, 2017. 69

[158] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016. 67

[159] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 56

[160] Paul Upchurch, Daniel Sedra, Andrew Mullen, Haym Hirsh, and Kavita Bala. Interactive consensus agreement games for labeling images. In *AAAI Conference on Human Computation and Crowdsourcing (HCOMP)*, October 2016. 13

[161] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018. 51

[162] Mitchell JP Van Zuijlen, Hubert Lin, Kavita Bala, Sylvia C Pont, and Maarten WA Wijntjes. Materials in paintings (mip): An interdisciplinary dataset for perception, art history, and computer vision. *Plos one*, 16(8):e0255109, 2021. 5, 6, 40, 41, 43, 68, 127

[163] Mitchell JP van Zuijlen, Sylvia C Pont, and Maarten WA Wijntjes. Painterly depiction of material properties. *Journal of vision*, 20(7):7–7, 2020. 45

[164] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 51

[165] Krzysztof Walas. Terrain classification and negotiation with a walking robot. *J Intell Robot Syst*, 78:401–423, 2015. 90

[166] Haohan Wang, Xindi Wu, Zeyi Huang, and Eric P Xing. High-frequency component helps explain the generalization of convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8684–8694, 2020. 83

[167] Huan Wang, Yijun Li, Yuehai Wang, Haoji Hu, and Ming-Hsuan Yang. Collaborative distillation for ultra-resolution universal style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1860–1869, 2020. 65

[168] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Wenjun Zeng, and Tao Qin. Generalizing to unseen domains: A survey on domain generalization. *arXiv preprint arXiv:2103.03097*, 2021. 2

[169] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 87

[170] Yisen Wang, Xingjun Ma, James Bailey, Jinfeng Yi, Bowen Zhou, and Quanquan Gu. On the convergence and robustness of adversarial training. In *ICML*, volume 1, page 2, 2019. 65

[171] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018. 93

[172] Xiu-Shen Wei, Jianxin Wu, and Quan Cui. Deep learning for fine-grained image analysis: A survey. *arXiv preprint arXiv:1907.03069*, 2019. 51

[173] Peter Welinder, Steve Branson, Pietro Perona, and Serge J Belongie. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, pages 2424–2432, 2010. 13

[174] John White. *The birth and rebirth of pictorial space*. Cambridge, MA, 1957. 42

[175] "Lisa Wiersma". "colouring material depiction in flemish and dutch baroque art theory". *"Art & Perception"*, pages "1 – 23", "22 Apr. 2020". 42

[176] M. W. A. Wijntjes, C. Spoiala, and H. de Ridder. Thurstonian scaling and the perception of painterly translucency. *Art & Perception*, pages 1 – 24, 04 Sep. 2020. 42

[177] WikiArt. Wikiart: Visual art encyclopedia. In *wikiart.org*. 69

[178] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2, 2019. 47

[179] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas Huang. Deep grabcut for object selection. *arXiv preprint arXiv:1707.00243*, 2017. 13

[180] Ning Xu, Brian Price, Scott Cohen, Jimei Yang, and Thomas S Huang. Deep interactive object selection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–381, 2016. 13

[181] Ning Xu, Brian L Price, Scott Cohen, and Thomas S Huang. Deep image matting. In *CVPR*, volume 2, page 4, 2017. 13

[182] Jingkang Yang, Kaiyang Zhou, Yixuan Li, and Ziwei Liu. Generalized out-of-distribution detection: A survey. *arXiv preprint arXiv:2110.11334*, 2021. 3

[183] Mang Ye, Jianbing Shen, Gaojie Lin, Tao Xiang, Ling Shao, and Steven CH Hoi. Deep learning for person re-identification: A survey and outlook. *arXiv preprint arXiv:2001.04193*, 2020. 41

[184] J. Yen. Finding the k shortest loopless paths in a network. *Manage Sci*, 17(11):712–716, 1971. 98

[185] Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer. A fourier perspective on model robustness in computer vision. *Advances in Neural Information Processing Systems*, 32:13276–13286, 2019. 84

[186] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018. 11, 13, 24, 25

[187] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6023–6032, 2019. 63, 65

[188] Lishi Zhang, Chenghan Fu, and Jia Li. Collaborative annotation of semantic objects in images with multi-granularity supervisions. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 474–482. ACM, 2018. 14

[189] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-time user-guided image colorization with learned deep priors. *ACM Transactions on Graphics (TOG)*, 9(4), 2017. 32

[190] Yinda Zhang, Shuran Song, Ersin Yumer, Manolis Savva, Joon-Young Lee, Hailin Jin, and Thomas Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 19

[191] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 98, 141

[192] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, volume 1, page 4. IEEE, 2017. 5, 11, 33, 34, 45, 93

[193] Kaiyang Zhou, Ziwei Liu, Yu Qiao, Tao Xiang, and Chen Change Loy. Domain generalization: A survey. *arXiv preprint arXiv:2103.02503*, 2021. 2

[194] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. In *Proc. CVPR*, pages 4490–4499, 2018. 93

[195] Aleksandar Zlateski, Ronnachai Jaroensri, Prafull Sharma, and Frédo Durand. On the importance of label quality for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1479–1487, 2018. 14