TOWARD THE GENERALIZATION OF VISION MODELS

by

Yutong Bai

A dissertation submitted to Johns Hopkins University

in conformity with the requirements for the degree of

Doctor of Philosophy

Baltimore, Maryland

March, 2024

Abstract

In the field of computer vision, the ability to generalize is essential for ensuring robust and reliable performance across diverse datasets and real-world scenarios. Generalization refers to a computer vision model's capability to accurately interpret and process new, unseen data beyond the specific examples it was trained on. Real-world visual data can vary widely in terms of lighting conditions, backgrounds, viewpoints, occlusions, and other factors, making generalization crucial for effective performance. A computer vision system with strong generalization ability can identify objects, recognize patterns, and make accurate predictions even when presented with variations it has not encountered during training. Achieving robust generalization necessitates careful algorithm design, sufficient training data that captures real-world variability, regularization techniques to prevent overfitting, and robust evaluation methodologies. Without robust generalization, computer vision models may perform well on training data but fail to generalize to new, unseen data, limiting their real-world applicability.

In this dissertation, I will present my research efforts in enhancing the generalization ability of vision models from several perspectives. Firstly, i focus on designing vision algorithms that can generalize to novel viewpoints and occlusions, tackling challenges commonly encountered in real-world scenarios. Secondly, I investigate the architectural disparities between vision transformers and convolutional neural networks (CNNs) to better understand their impact on generalization, particularly concerning out-of-distribution data and adversarial attacks. Thirdly, I propose innovative self-supervised learning algorithms, Point-Level Region Contrast for Object Detection Pre-Training, aimed at providing strong backbone features for generalization across various downstream tasks. Additionally, I introduce efficient masked autoencoding techniques for enhanced representation learning. Finally, I propose Sequential Modeling Enables Scalable Learning for Large Vision Models, a methodology designed to effectively leverage large volumes of real-world visual data, thereby further enhancing generalization capabilities. Through these contributions, I aim to advance the field towards more generalized and robust vision models with broad applicability across diverse real-world scenarios.

Thesis Readers

Dr. Alan L. Yuille (Primary Advisor)
Bloomberg Distinguished Professor
Department of Computer Science
Johns Hopkins University

Dr. Alexei Efros
Professor
Department of Electrical Engineering and Computer Sciences
UC Berkeley

Dr. Tianmin Shu
Assistant Professor
Department of Computer Science
Johns Hopkins University

Acknowledgments

Pursuing a PhD has been the most arduous journey of my life. Beyond the realm of research, I faced a maelstrom of challenges: visa restrictions, covid, separation, depression, chronic pain, a car crash, and even the loss of loved ones. It felt as though Murphy's Law was in full effect—if something could go wrong, it seemed destined to.

Through these trials, my family has been my bedrock, providing reasons to persevere even in the darkest times. I am deeply grateful for their unwavering support and love.

I owe a tremendous debt of gratitude to my mentors, whose guidance and tutelage rescued me from despair. My advisor, Alan Yuille, was the first to supervise me, offering me the opportunity to delve into research and showing me the breadth of real challenges in computer vision beyond benchmarks. Alyosha Efros, by example, imparted to me the essence of research purity and meticulously trained me in the rigors of scientific inquiry. Jitendra Malik noticed my struggle under the immense pressure of research, reminding me that our work targets problems, not people. Jianbo Shi provided invaluable care and guidance during times of personal confusion, spending many hours helping me navigate through my troubles and rebuild my confidence. Trevor Darrell recognized the unique pressures faced by female researchers like myself and aided me in regaining my footing.

My friends—Yihui Ma, Qiuyue Wang, Qihang Yu, Yidong Shi, Chenhao Ling, Yu Sun, Amir Bar, Karttikeya Mangalam, Xinyang Geng, Lisa Dunlap, Himanshu Singh, and Konpat Preechakul — have been pillars of understanding and unconditional support. In moments of sorrow and hardship, they stood by my side without hesitation.

Lastly, I wish to give myself a minor acknowledgement for surviving these trials. This journey has taught me resilience, and I thank myself for the strength to continue fighting and thriving amidst adversity.

To everyone who has been a part of this journey: your support has not only sustained me but has also transformed me. Thank you for believing in me, for lifting me up, and for being the light in my darkest days.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Generalization is one of the key goals in computer vision research. It refers to a model's capability to perform well on new, unseen data that is different from the data it was trained on. This is important because real-world visual data can be extremely varied, with changes in lighting, viewpoints, occlusions, and environments.

In this thesis, I first look at supervised algorithms that allow object detection models to generalize to unseen viewpoints (Chapter 2) and be robust to occlusions (Chapter 3). We also explore the role of architecture choice, and surprisingly find (Chapter 4) that the specific architecture is not the most crucial factor for generalization.

Since self-supervised representation learning has shown some level of generalization capability across downstream tasks (Chapter 5), I then dive deeper into developing strong representation learning algorithms (Chapter 6).

However, in all previous representation learning approaches, features were encoded independently for each image, resulting in a gap when transferring to downstream tasks after fine-tuning.

In this sense, the algorithm is never allowed to think on the fly by following hints. To leap over this gap, I reformulate visual recognition as a "visual sentence completion" task in (Chapter 7). It integrates all types and sources of visual data, including human-labeled and naturally co-occurring data, into a sequential form. Recognition is formulated as a think-on-the-fly in-context prediction task, where the next image predicted could be a segmentation mask or an imagined result of an action. This unified framework is designed to improve generalization since it combines all possible generative and discriminative tasks into one common framework during the learning stage. We are not learning to remember image labels in isolation but learning to follow hints in a visual sentence and thus think on the fly.

In conclusion, the road toward generalization is full of pitfalls, dead ends, and false starts. The final discovery emerges from the sad realization that true generalization does not come from algorithms or architectures, and training on ImageNet data give limited generalization ability. Accepting this allows me to focus on the data itself: train with diverse things and make the prediction as hard as possible.

1.1 Generalize to Novel Viewpoints for Semantic Part Detection

Detecting semantic parts of an object is a challenging task in computer vision, particularly because it is hard to construct large annotated datasets due to the difficulty of annotating semantic parts. In Chapter 2, we present an approach that learns from a small dataset of annotated semantic parts captured from a limited range of viewpoints, yet generalizes to detect parts from a much broader range of unseen viewpoints.

Our matching-based algorithm establishes accurate spatial correspondences between image pairs, transferring part annotations from one image to another. Specifically, we match training images to a virtual 3D object model and use clustering to annotate the model's parts. This annotated 3D model synthesizes part-labeled images across many viewpoints, which can be matched to test images to detect parts in novel views. Despite being simple and parameter-efficient, our algorithm outperforms conventional deep nets, particularly excelling at novel viewpoints on the VehicleSemanticPart car subset.

1.2 Contrastive Learning for Robust Keypoint Detection

Chapter 3 introduces CoKe, a contrastive learning framework tailored for keypoint detection, which not only enhances robustness to occlusion but also fosters finer-grained understanding of object structures, bolstering generalization capabilities.

Unlike other vision tasks, keypoint detection requires representing and detecting multiple annotated keypoints per input image. Our approach extends contrastive learning to handle this multi-keypoint scenario, allowing the loss to discriminate keypoint features from each other and background cues. CoKe offers two key benefits: leveraging contrastive learning's power for keypoint tasks and achieving robustness to occlusions by detecting keypoints independently rather than holistically. Our framework employs technical innovations like a clutter bank for non-keypoint features, a keypoint bank storing prototypical keypoint representations to approximate the contrastive loss between keypoints, and a cumulative moving average update to learn these prototypes while training the feature extractor. Across diverse datasets (PASCAL3D+, MPII, ObjectNet3D), CoKe performs favorably against alternative keypoint detectors, even for human poses. Notably, it exhibits exceptional robustness to partial occlusions and previously unseen object poses.

1.3 Are Transformers More Robust Than CNNs?

Chapter 4 transitions to a broader comparative analysis of vision models. Our research explores the robustness and generalization capabilities of transformers and CNNs. By conducting fair comparisons and delving into the intricacies of adversarial robustness and generalization, we aim to elucidate the strengths and weaknesses of these foundational architectures, offering valuable insights for the advancement of robust vision systems.

Transformers have emerged as a powerful tool for visual recognition, demonstrating competitive performance across a wide range of visual benchmarks. Recent works have argued that Transformers are significantly more robust than Convolutional Neural Networks (CNNs). Surprisingly, however, we find that these conclusions are drawn from unfair experimental settings, where Transformers and CNNs are compared at different scales and with distinct training frameworks. In this work, we aim to provide the first fair and in-depth comparisons between Transformers and CNNs, focusing on evaluations of robustness. With our unified training setup, we first challenge the previous belief that Transformers outperform CNNs in measuring adversarial robustness. More surprisingly, we find that CNNs can easily match the robustness of Transformers against adversarial attacks if they properly adopt the training recipes of Transformers. Regarding generalization to out-of-distribution samples, we show that pre-training on large-scale external datasets is not a fundamental requirement for Transformers to achieve better performance than CNNs. Moreover, our ablations suggest that such stronger generalization is largely benefited by the Transformer's self-attention-like architectures themselves, rather than by other training setups.

1.4 Point-Level Region Contrast for Object Detection Pre-Training

In Chapter 5, the focus shifts towards pre-training methodologies for object detection, with the introduction of point-level region contrast. This self-supervised approach harnesses contrastive learning to improve both localization and recognition, thereby enhancing generalization across multiple tasks and datasets.

This approach is motivated by the two key factors in detection: localization and recognition. While accurate localization favors models that operate at the pixel- or point-level, correct recognition typically relies on a more holistic, region-level view of objects. Incorporating this perspective in pre-training, our approach performs contrastive learning by directly sampling individual point pairs from different regions. Compared to an aggregated representation per region, our approach is more robust to the change in input region quality and further enables us to implicitly improve initial region assignments via online knowledge distillation during training. Both advantages are important when dealing with imperfect regions encountered in the unsupervised setting.

Experiments show point-level region contrast improves on state-of-the-art pre-training methods for object detection and segmentation across multiple tasks and datasets, and we provide extensive ablation studies and visualizations to aid understanding.

1.5 Masked Autoencoders Enable Efficient Knowledge Distillers

As we find in Chapter 5, self-supervised learning has shown generalization ability among many downstream tasks. Therefore, in this work, we hope to keep empowering the representation learning algorithm by developing an efficient way for distilling a pre-trained model, which enables strong downstream performance and generalization under a very low budget.

The approach is simple: in addition to optimizing the pixel reconstruction loss on masked inputs, we minimize the distance between the intermediate feature map of the teacher model and that of the student model. This design leads to a computationally efficient knowledge distillation framework, given 1) only a small visible subset of patches is used, and 2) the (cumbersome) teacher model only needs to be partially executed, *i.e.*, forward propagate inputs through the first few layers, for obtaining intermediate feature maps.

Compared to directly distilling fine-tuned models, distilling pre-trained models substantially improves downstream performance. For example, by distilling the knowledge from an MAE pre-trained ViT-L into a ViT-B, our method achieves 84.0% ImageNet top-1 accuracy, outperforming the baseline of directly distilling a fine-tuned ViT-L by 1.2%. More intriguingly, our method can robustly distill knowledge from teacher models even with extremely high masking ratios: *e.g.*, with 95% masking ratio where merely TEN patches are visible during distillation, our ViT-B competitively attains a top-1 ImageNet accuracy of 83.6%; surprisingly, it can still secure 82.4% top-1 ImageNet accuracy by aggressively training with just FOUR visible patches (98% masking ratio).

1.6 Sequential Modeling Enables Scalable Learning for Large Vision Models

In Chapter 7, we introduce a novel sequential modeling approach that enables learning a Large Vision Model (LVM) without making use of any linguistic data. To do this, we define a common format, “visual sentences", in which we can represent raw images and videos as well as annotated data sources such as semantic segmentations and depth reconstructions without needing any meta-knowledge beyond the pixels. Once this wide variety of visual data (comprising 420 billion tokens) is represented as sequences, the model can be trained to minimize a cross-entropy loss for next token prediction. By training across various scales of model architecture and data diversity, we provide empirical evidence that our models scale effectively. Many different vision tasks can be solved by designing suitable visual prompts at test time.

This novel sequential modeling approach builds upon our previous findings on enhancing generalization, robustness, and representation learning for vision models. By representing all data modalities as sequences in a common format, LVM can leverage self-supervised learning objectives like next token prediction to learn powerful representations that generalize effectively across diverse visual tasks and conditions.

1.7 Relevant Publications

Parts of this present dissertation has been previously published in conference proceedings, as follows:

- Chapter 2 - Bai, Yutong, Qing Liu, Lingxi Xie, Weichao Qiu, Yan Zheng, and Alan

L. Yuille. "Semantic part detection via matching: Learning to generalize to novel viewpoints from limited training data." In Proceedings of the IEEE/CVF International Conference on Computer Vision 2019 (13).

- Chapter 3 - Bai, Yutong, Angtian Wang, Adam Kortylewski, and Alan Yuille. "Coke: Contrastive learning for robust keypoint detection." In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision 2023 (15).

- Chapter 4 - Bai, Yutong, Jieru Mei, Alan L. Yuille, and Cihang Xie. "Are transformers more robust than cnns?." Advances in neural information processing systems 2021 (14).

- Chapter 5 - Bai, Yutong, Xinlei Chen, Alexander Kirillov, Alan Yuille, and Alexander C. Berg. "Point-level region contrast for object detection pre-training." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2022 (10).

- Chapter 6 - Bai, Yutong, Zeyu Wang, Junfei Xiao, Chen Wei, Huiyu Wang, Alan L. Yuille, Yuyin Zhou, and Cihang Xie. "Masked autoencoders enable efficient knowledge distillers." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2023 (16).

- Chapter 7 - Bai, Yutong, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan Yuille, Trevor Darrell, Jitendra Malik, and Alexei A. Efros. "Sequential modeling enables scalable learning for large vision models." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2024 (12).

Chapter 2

Semantic Part Detection via Matching: Learning to Generalize to Novel Viewpoints from Limited Training Data

As discussed earlier, achieving good generalization ability is a key goal in computer vision research. This means models should perform well on new, unseen data that is different from what they were trained on. In this chapter, we explore specific methods to improve the generalization capabilities of vision models. We start by tackling the problem of detecting semantic parts (e.g. wheels, headlights) of objects from limited training data covering only a narrow range of viewpoints.

2.1 Introduction

Object detection has been a long-lasting challenge in computer vision which has attracted a lot of research attention (80)(82). Recently, with the development of deep networks, this research area has been dominated by a strategy which starts by extracting a number of regional proposals and then determines if each of them belongs to a specific set of object

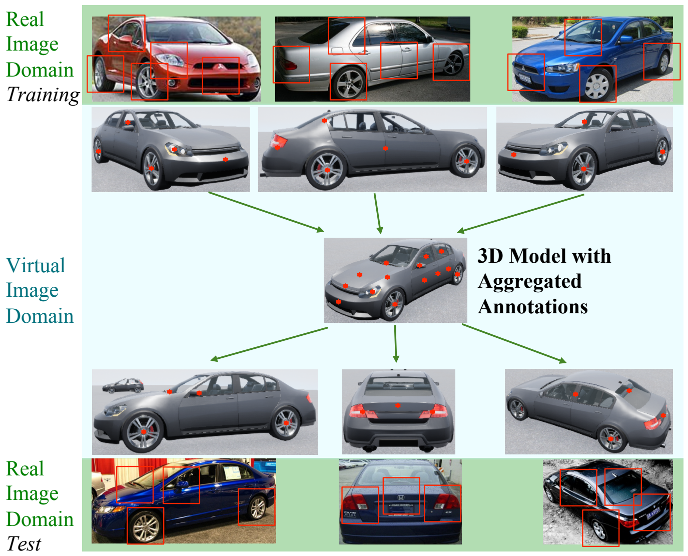


Figure 2.1: The flowchart of our approach (best viewed in color). The key module is the matching algorithm which finds spatial correspondence between images with similar viewpoints. This enables us to match the training data to a virtual 3D model and thereby annotate it. The virtual 3D model can then be used to detect the semantic parts on images in the test set, by re-using the matching algorithm.

classes (90)(205)(60)(160)(203). The success of these approaches (80)(151) motivates researchers to address the more challenging task of understanding the objects at finer level and, in particular, to parse it into semantic parts, which we define to be those components of an object with semantic meaning and can be verbally described (314). A particular challenge is that annotating semantic parts is very time-consuming and much more difficult than annotating objects, which makes it harder to directly apply deep networks to this task.

In this paper, we address the problem of semantic part detection when there is a small amount of training data and when the object is seen from a limited range of viewpoints. Our strategy is to define a matching algorithm which finds correspondences between images of the same object seen from roughly the same viewpoint. This can be used to match the training images to the rendered images of a virtual 3D model enabling us to annotate the semantic parts of the 3D model. The same matching algorithm can then be used to find correspondences between the virtual 3D model and the test images, which enables us to detect the semantic parts of the test images, even though they may correspond to viewpoints not seen in the training set. The overall framework is illustrated in Figure 2.1.

Mathematically, we have a training set $\mathcal{D}$, in which each image $\mathbf{I}_n$ has a viewpoint $\mathbf{p}_n$ and a semantic part set $\mathcal{S}_n$. Given a testing image $\mathbf{I}^*$ and the corresponding viewpoint $\mathbf{p}^*$ (provided by ground-truth or some state-of-the-art approaches such as (233)), the goal is to predict a set of matches for each training sample, *i.e.*, $\mathcal{M}_n = \mathbf{M}(\mathbf{I}^*, \mathbf{I}_n)$, so that $\mathcal{S}_n$ is transplanted to $\mathcal{S}_n^*$ according to $\mathcal{M}$ and finally all these transplanted sets are summarized into the final prediction $\mathcal{S}^*$. In this pipeline, the key component is the matching algorithm $\mathbf{M}(\cdot,\cdot)$. For simplicity, we assume that $\mathbf{M}(\cdot,\cdot)$ only deals with two images with similar viewpoints (which could be found by an off-the-shelf viewpoint detector). In order to combine information from the training images with different viewpoints, we introduce a viewpoint transfer function which converts each training image $\mathbf{I}_n$ with viewpoint $\mathbf{p}_n$ into $\mathbf{I}_n'$ with viewpoint $\mathbf{p}^*$, following which $\mathbf{M}(\mathbf{I}^*, \mathbf{I}_n')$ is computed. the matching algorithm, this is implemented by introducing a virtual model $\mathbb{I}$ (*e.g.*, represented by point clouds) with 3D semantic part annotations $\mathbb{S}$ assigned to it. A graphics algorithm is used for rendering $(\mathbf{I}_n^\circ, \mathcal{S}_n^\circ) = \mathbf{g}(\mathbb{I}, \mathbb{S} \mid \mathbf{p}_n)$. $\mathbb{S}$ is a hidden variable, so we apply a clustering-based algorithm to estimate it in the training stage. An additional benefit of this strategy is that the costly matching algorithm $\mathbf{M}(\cdot,\cdot)$ is executed only once in the testing stage.

The major contribution of this work is to provide a simple and intuitive algorithm for semantic part detection which works using little training data and can generalize to novel viewpoints. It is an illustration of how virtual data can be used to reduce the need for time-consuming semantic part annotation. Experiments are performed on the VehicleSemanticPart (VSP) dataset (259), which is currently the largest corpus for semantic part detection. In the *car* subclass, our approach achieves significantly better performance than standard end-to-end methods such as Faster R-CNN (205) and DeepVoting (314). The advantages become even bigger when the amount of training data is limited.

The remainder of this paper is organized as follows. Section 2.2 briefly reviews the prior literature, and Section 2.3 presents our framework. After experiments are shown in Section 2.4, we conclude this work in Section 2.5.

2.2 Related Work

In the past years, deep learning (143) has advanced the research and applications of computer vision to a higher level. With the availability of large-scale image datasets (65) as well as powerful computational device, researchers designed very deep neural networks (141)(223)(231) to accomplish complicated vision tasks. The fundamental idea of deep learning is to organize neurons (the basic units that perform specified mathematical functions) in a hierarchical manner, and tune the parameters by fitting a dataset. Based on some learning algorithms to alleviate numerical stability issues (178)(228)(128), researchers developed deep learning in two major directions, namely, increasing the depth of the network towards higher recognition accuracy (110)(125)(124), and transferring the pre-trained models to various tasks, including feature extraction (70)(202), object detection (90)(89)(205), semantic segmentation (162)(37), pose estimation (180), boundary detection (285), *etc*.

For object detection, the most popular pipeline, in the context of deep learning, involves first extracting a number of bounding-boxes named regional proposals (3)(249)(205), and then determining if each of them belongs to the target class (90)(89)(205)(60)(160)(203). To improve spatial accuracy, the techniques of bounding-box regression (131) and non-maximum suppression (122) were widely used for post-processing. Boosted by high-quality visual features and end-to-end optimization, this framework significantly outperforms the conventional deformable part-based model (82) which were trained on top of handcrafted features (62). Despite the success of this framework, it still suffers from weak explainability, as both object proposal extraction and classification modules were black-boxes, and thus easily confused by occlusion (314) and adversarial attacks (279). There were also research efforts of using mid-level or high-level contextual cues to detect objects (259) or semantic parts (314). These methods, while being limited on rigid objects such as vehicles, often benefit from better transferability and work reasonably well on partially occluded data (314).

Another way of visual recognition is to find correspondence between features or images, so that annotations from one (training) image can be transplanted to another (testing) image (103)(136)(183)(234)(248). This topic was noticed in the early age of vision (184) and later built upon handcrafted features (169)(123)(294). There were efforts in introducing semantic information into matching (153), and also improving the robustness against noise (166). Recently, deep learning has brought a significant boost to these problems by improving both features (202)(317) and matching algorithms (73)(102)(320)(126), while a critical part of these frameworks still lies in end-to-end optimizing deep networks.

Training a vision system requires a large amount of data. To alleviate this issue, researchers sought for help from the virtual world, mainly because annotating virtual data

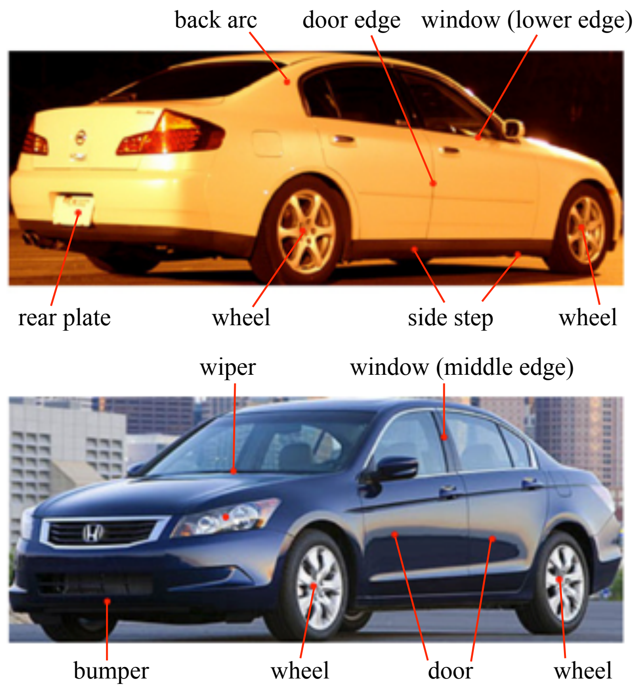


Figure 2.2: Two examples of annotated semantic parts in the class *car*. For better visualization, we only show part of annotation.

is often easy and cheap (193). Another solution is to perform unsupervised or weakly-supervised training with consistency that naturally exists (320)(94)(324). This paper investigates both of these possibilities.

2.3 Our Approach

2.3.1 Problem: Semantic Part Detection

The goal of this work is to detect *semantic parts* on an image. We use $\mathcal{P}$ to denote the image-independent set of semantic parts, each element in which indicates a verbally describable components of an object (259). For example, there are tens of semantic parts in the class *car*, including *wheel*, *headlight*, *licence plate*, *etc*. Two *car* examples with semantic parts annotated are shown in Figure 2.2.

Let the training set $\mathcal{D}$ contain N images, and each image, $\mathbf{I}_n$, has a width of W_n and a height of H_n. A set $\mathcal{S}_n$ of M_n semantic parts are annotated for each image, and each semantic part appears as a bounding box $\mathbf{b}_{n,m}$ and a class label $s_{n,m} \in \{1, 2, \ldots, |\mathcal{P}|\}$, where m is the index.

2.3.2 Framework: Detection by Matching

We desire a function $\mathcal{S} = \mathbf{f}(\mathbf{I}; \boldsymbol{\theta})$ which receives an image $\mathbf{I}$ and outputs the corresponding semantic part set $\mathcal{S}$. In the context of deep learning, researcher designed end-to-end models (205) which starts with an image, passes the signal throughout a series of layers, and outputs the prediction in an encoded form. With ground-truth annotation $\mathcal{S}_n$, a loss function $\mathcal{L}_n$ is computed between $\mathcal{S}'_n = \mathbf{f}(\mathbf{I}_n; \boldsymbol{\theta})$ and $\mathcal{S}_n$, and the gradient of $\mathcal{L}_n$ with respect to $\boldsymbol{\theta}$ is computed in order to update $\boldsymbol{\theta}$ accordingly. Despite their success, these approaches often require a large number of annotations to avoid over-fitting, yet their ability of explaining the prediction is relatively weak. DeepVoting (314) went a step further by simplifying the high-level inference layers as well as reducing the number of parameters in them, so that the prediction can be partly explained by a feature voting process. However, as we shall see in experiments, it still suffers significant accuracy drop in the scenario of few training data.

This paper investigates the problem of semantic part detection from another perspective. Instead of directly optimizing $\mathbf{f}(\cdot)$, we adopt an indirect approach which assumes that semantic part detection can be achieved by finding semantic correspondence between two (or more) images. This is to say, if a training image $\mathbf{I}_n$ is annotated with a set $\mathcal{S}_n$ of semantic parts, and we know that $\mathbf{I}_n$ is densely matched to a testing image $\mathbf{I}^*$, then we can transplant $\mathcal{S}_n$ to $\mathcal{S}^*$ by projecting each element of $\mathcal{S}_n$ onto the corresponding element of $\mathcal{S}^*$ by applying a spatially-constrained mapping function.

This approach suffers from a major drawback. For every testing image $\mathbf{I}^*$, there needs to be at least one training image which has a very similar viewpoint of $\mathbf{I}^*$. The argument is two fold. First, semantic parts that appear in an image vary among different viewpoints, so we shall not expect the annotation to transfer between two images with large viewpoint difference. Second, towards simplicity, we aim at using a relatively simple algorithm, *e.g.*, the max-clique algorithm, for image matching. However, in the scenarios with few (*e.g.*, tens of) training images, it is not realistic to expect the existence of such training samples.

Our solution, as well as the main contribution of this work, is to introduce auxiliary cues by borrowing information from those training images with clearly different viewpoints. These auxiliary cues are named viewpoint consistency, which suggests that when an object and its semantic parts are observed from one viewpoint, they should be predictable under another viewpoint, *e.g.*, the viewpoint of the testing image. Therefore, our idea is to transfer each training image to the same viewpoint of the testing image with semantic parts preserved, so that we can make prediction and transplant semantic parts with a light-weighted feature matching algorithm. In what follows, we formulate this idea mathematically and solve it using an efficient algorithm.

2.3.3 The Matching Algorithm

We start with defining a generation function $\mathbf{g}(\cdot)$ which takes a source (training) image $\mathbf{I}_n$ as well as its semantic part annotation $\mathcal{S}_n$, refers to the source and target viewpoints $\mathbf{p}_n$ and $\mathbf{p}^*$, and outputs a transferred image and the corresponding semantic parts:

$$\left(\mathbf{I}_n', \mathcal{S}_n'\right) = \mathbf{g}(\mathbf{I}_n, \mathcal{S}_n \mid \mathbf{p}_n, \mathbf{p}^*). \tag{2.1}$$

Throughout the remaining part, a prime in superscript indicates elements in a generated image. Next, by assuming that $\mathbf{I}^*$ and $\mathbf{I}_n'$ have the same viewpoint[1], we build a regional feature matching between them, denoted by $\mathcal{M}_n^* = \mathbf{M}(\mathbf{I}^*, \mathbf{I}_n')$. We assume that both images are represented by a set of regional features, each of which describes the appearance of a patch. In the context of deep learning, this is often achieved by extracting mid-level neural responses from a pre-trained deep network (259)(317). Although it is possible to fine-tune the network with an alternative head for object detection (314), we do not take this option in order to preserve the model's explainability[2]. Let an image $\mathbf{I}_n$ have a set $\mathcal{V}_n$ consisting of L_n regional features, the l-th of which is denoted by $\mathbf{v}_{n,l}$. We assume that all these feature vectors have a fixed length, *e.g.*, all of them are 512-dimensional vectors corresponding to specified positions at the *pool-4* layer of VGGNet (223)(259). Each $\mathbf{v}_{n,l}$ is also associated with a 2D coordinate $\mathbf{u}_{n,l}$.

[1]We expect $\mathbf{I}^*$ and $\mathbf{I}_n'$ to have exactly the same viewpoint, though the matching algorithm $\mathbf{I}^*$ has the true viewpoint of $\mathbf{p}^*$ while $\mathbf{I}_n'$ is generated by the estimated $\mathbf{p}^*$, so the actual viewpoints of these two images can be slightly different, *e.g.*, (233) reported a medium viewpoint prediction error of less than 10° on the *car* subclass.

[2]We follow the argument that low-level features, *e.g.*, edges, basic geometric shapes, *etc.*, can be automatically learned by deep networks on a large dataset, but in order to achieve explainability, the high-level inference part should be assigned with clear meanings, *e.g.*, visual word counting or feature voting.

Then, $\mathcal{M}_n^*$ takes the form of:

$$\mathcal{M}_n^* = \mathbf{M}\big(\mathbf{I}^*, \mathbf{I}_n'\big) = \left\{\left(\mathbf{v}_{l_w}^*, \mathbf{u}_{l_w}^*, \mathbf{v}_{n,l_{n,w}}', \mathbf{u}_{n,l_{n,w}}'\right)\right\}_{w=1}^{W}, \tag{2.2}$$

which implies that a feature vector $\mathbf{v}_{l_w}^*$ with a coordinate $\mathbf{u}_{l_w}^*$ on $\mathbf{I}^*$ is matched to a feature $\mathbf{v}_{n,l_{n,w}}'$ at with a coordinate $\mathbf{u}_{n,l_{n,w}}'$ on $\mathbf{I}_n'$. l_w and $l_{n,w}$ are the matched feature indices on $\mathbf{I}^*$ and $\mathbf{I}_n'$, respectively. We use both unary and binary relationship terms to evaluate the quality of $\mathcal{M}_n^*$ in terms of appearance and spatial consistency. The penalty function of $\mathcal{M}_n^*$, $\alpha(\mathcal{M}_n^*)$, is defined as:

$$\begin{aligned}\alpha(\mathcal{M}_n^*) = \lambda_1 \cdot \sum_{w=1}^{W} \left|\mathbf{v}_{l_w}^* - \mathbf{v}_{n,l_{n,w}}'\right|^2 + \\ \lambda_2 \cdot \sum_{1 \leqslant w_1 < w_2 \leqslant W} \left|\Delta\mathbf{u}_{l_{w_1},l_{w_2}}^* - \Delta\mathbf{u}_{n,l_{n,w_1},l_{n,w_2}}'\right|^2,\end{aligned} \tag{2.3}$$

where Δ denotes the oriented distance between two features, *i.e.*, $\Delta\mathbf{u}_{l_{w_1},l_{w_2}}^* = \mathbf{u}_{l_{w_1}}^* - \mathbf{u}_{l_{w_2}}^*$ and $\Delta\mathbf{u}_{n,l_{n,w_1},l_{n,w_2}}' = \mathbf{u}_{n,l_{n,w_1}}' - \mathbf{u}_{n,l_{n,w_2}}'$. Thus, the first term on the right-hand side measures the similarity in appearance of the matched patches, and the second term measures the spatial consistency of the connection between all patch pairs.

Based on $\mathcal{M}_n^*$, we can compute a coordinate transformation function $\mathbf{T}(\cdot)$ mapping the bounding box of each semantic part of $\mathbf{I}_n'$ to the corresponding region on $\mathbf{I}^*$:

$$\mathbf{b}_{n,m}^* = \mathbf{T}\big(\mathbf{b}_{n,m}' \mid \mathcal{M}_n^*\big), \quad s_{n,m}^* = s_{n,m}' = s_{n,m}, \tag{2.4}$$

where m is the index of a semantic part.

2.3.4 Optimization with Multiple Viewpoints

After the annotations of all training images are collected and transplanted to $\mathbf{I}^*$, we apply the final term named cross-viewpoint consistency to confirm that these annotations align with each other. For simplicity, we denote $\mathbf{b}_m^*$ as the m-th semantic part transferred to $\mathbf{I}^*$, regardless of its source image index n. For all pairs (m_1, m_2) with the same semantic part index, *i.e.*, $s_{m_1}^* = s_{m_2}^*$, we compute the intersection-over-union (IOU) between the two bounding boxes and penalize those pairs with similar but clearly different positions:

$$\beta(\mathbf{b}^*, \mathbf{s}^*) = \lambda_3 \cdot \sum_{\substack{s_{m_1}^* = s_{m_2}^* \\ \mathrm{IOU}\left(\mathbf{b}_{m_1}^*, \mathbf{b}_{m_2}^*\right) \geqslant \tau}} 1 - \mathrm{IOU}\left(\mathbf{b}_{m_1}^*, \mathbf{b}_{m_2}^*\right), \tag{2.5}$$

where we set $\tau = 0.5$, *i.e.*, two annotations are considered to be different instances if their IOU is smaller than 0.5.

The final solution $\mathbf{b}^*, \mathbf{s}^*$ comes from minimizing the sum of the matching penalty defined in Eqn (2.3) and the consistency penalty defined in Eqn (2.5):

$$Q(\mathbf{b}^*, \mathbf{s}^* \mid \mathbf{I}^*, \mathcal{D}) = \sum_{n=1}^{N} \alpha(\mathcal{M}_n^*) + \beta(\mathbf{b}^*, \mathbf{s}^*). \tag{2.6}$$

In this overall loss function, three modules can be optimized, namely, the generator $\mathbf{g}(\cdot)$, the matching function $\mathbf{M}(\cdot, \cdot)$ and the coordinate transformation function $\mathbf{T}(\cdot)$.

Directly optimizing Eqn (2.6) is computationally intractable due to two reasons. First, it is not a convex function. Second, Eqn (2.6) involves matching $\mathbf{I}^*$ to all training images, which makes it time consuming the matching algorithm. This subsection provides a practical solution, which deals with the first issue with iteration, and the second using a 3D model, while the possibility of designing other algorithm to optimize this generalized objective function is preserved.

The major motivation comes from accelerating computation. We use a simplified version of Eqn (2.1) which assumes that all images $\mathbf{I}_n'$ and annotations $\mathcal{S}_n'$ are generated by the same model and thus identical to each other:

$$\left(\mathbf{I}',\mathcal{S}'\right) \doteq \left(\mathbf{I}_1',\mathcal{S}_1'\right) = \ldots = \left(\mathbf{I}_N',\mathcal{S}_N'\right) = \mathbf{R}(\mathbb{I},\mathbb{S} \mid \mathbf{p}^*). \tag{2.7}$$

In practice, $\mathbb{I}$ is a 3D virtual model (*e.g.*, a point cloud) and $\mathbb{S}$ indicates the vertices in $\mathbb{I}$ that correspond to semantic parts. Provided a viewpoint $\mathbf{p}^*$, $\mathbf{R}(\cdot \mid \mathbf{p}^*)$ is a rendering algorithm (*e.g.*, UnrealCV (193)) that generates a 2D image $\mathbf{I}'$ from $\mathbb{I}$ with $\mathbb{S}$ projected onto the image as $\mathcal{S}'$. Thus, we relate each training data $\left(\mathbf{I}_n,\mathcal{S}_n\right)$ to the testing data $\left(\mathbf{I}^*,\mathcal{S}^*\right)$ by:

$$\left(\mathbf{I}_n^\circ,\mathcal{S}_n^\circ\right) = \mathbf{R}(\mathbb{I},\mathbb{S} \mid \mathbf{p}_n) = \mathbf{R}\left(\mathbf{R}^{-1}(\mathbf{I}^*,\mathcal{S}^* \mid \mathbf{p}^*) \mid \mathbf{p}_n\right), \tag{2.8}$$

and then computing the relationship between $\left(\mathbf{I}_n,\mathcal{S}_n\right)$ and $\left(\mathbf{I}_n^\circ,\mathcal{S}_n^\circ\right)$, *i.e.*, using Eqns (2.3) and (2.5) accordingly.

Finally, we assume that $\mathbb{I}$ and $\mathbb{S}$ do not change with testing data. Under this assumption, Eqn (2.6) is partitioned into two individual parts, namely, a training stage which optimizes $\mathbb{I}$ and $\mathbb{S}$, and a testing stage which infers $\mathcal{S}^*$, *i.e.*, $\mathbf{b}^*$ and $\mathbf{s}^*$. Though the first part is often time-consuming, it is executed only once and does not slow down testing.

2.3.4.1 Training: Optimizing the Hidden Variables

It is difficult to construct $\mathbb{I}$ directly by optimization, so pre-define a model set $\mathcal{I}$ and each $\mathbb{I}$ is an instance in it. We purchase $K = 5$ high-resolution models from the Unreal

Engine Marketplace, and enumerate through the set to achieve the best matching to each $\mathbf{I}_n$ individually. We denote $k_n \in \{1, 2, \ldots, K\}$ as the index of the model that $\mathbf{I}_n$ corresponds to. Given a specified $\mathbb{I}_k$, we first render it into $\mathbf{I}_n^\circ = \mathbf{R}(\mathbb{I}_k \mid \mathbf{p}_n)$, and then extract the feature sets $\mathcal{V}_n$ and $\mathcal{V}_n^\circ$ with L_n and L_n° elements, respectively. the matching algorithm, this is achieved by rescaling each image so that the short axis of the object is 224-pixel long (259), followed by feeding it into a pre-trained 16-layer VGGNet (223) and extracting all 512-dimensional feature vectors at the *pool-4* layer. All feature vectors are ℓ_2-normalized.

Now, Eqn (2.6), in the training stage, is simplified as:

$$Q(\{\mathbb{I}_k, \mathcal{S}_k\}, \{k_n\} \mid \mathcal{D}) = \sum_{n=1}^{N} \alpha\left(\mathcal{M}_{n,k_n}^\circ\right) + \sum_{k=1}^{K} \beta(\mathbf{b}_k^\circ, \mathbf{s}_k^\circ), \tag{2.9}$$

where $\mathcal{M}_{n,k_n}^\circ$ is the matching between $\mathbf{I}_n^\circ$ and $\mathbf{I}'_{k_n}$, and $\alpha\left(\mathcal{M}_{n,k_n}^\circ\right)$ and $\beta\left(\mathbf{b}_k^\circ, \mathbf{s}_k^\circ\right)$ are the corresponding losses to Eqns (2.3) and (2.5). Note that both of these terms depend on hidden variables $\{k_n\}_{n=1}^{N}$, *i.e.*, $\alpha\left(\mathcal{M}_{n,k_n}^\circ\right)$ computes the loss from $\mathbf{I}_n^\circ$ to $\mathbf{I}'_{k_n}$, and $\beta\left(\mathbf{b}_k^\circ, \mathbf{s}_k^\circ\right)$ only sums up the data that are assigned to $\mathbb{I}_k$. We use an iterative algorithm to optimize them. The starting point is that all k_n are randomly sampled from $\{1, 2, \ldots, K\}$.

In the **first-step** of each iteration, we fix all k_n as well as $\alpha\left(\mathcal{M}_{n,k_n}^\circ\right)$ and minimize $\beta\left(\mathbf{b}_k^\circ, \mathbf{s}_k^\circ\right)$ for each $\mathbb{I}_k$, which is equivalent to finding the optimal $\mathcal{S}_k$. Note that each $\mathbb{I}_k$ is equipped with an individual $\mathcal{S}_k$. So, we collect semantic parts from all images that are assigned to $\mathbb{I}_k$ and optimize Eqn (2.5). As an approximation, each collected $\mathbf{b}_m^\circ$ is considered a candidate if at least $\lfloor \mu \cdot N_k \rfloor$ boxes, including itself, have the same semantic part index as well as an IOU of at least τ with it, where μ is a hyper-parameter and N_k is the number of training images assigned to $\mathbb{I}_k$. The average of all the overlapping boxes to a candidate forms a semantic part of $\mathcal{S}_k$. This algorithm is able to filter out false positives (*e.g.*, those generated by incorrect matching) because these samples are mostly isolated. Also, the true positives are averaged towards more accurate localization.

In the **second-step**, we instead fix all $\beta\left(\mathbf{b}_k^{\circ}, \mathbf{s}_k^{\circ}\right)$, minimize each $\alpha\left(\mathcal{M}_{n,k_n}^{\circ}\right)$ by enumerating $k \in \{1, 2, \ldots, K\}$ and finding the best solution meanwhile updating k_n. Again, this is done in an approximate manner. For each of the $L_n \times L_n^{\circ}$ feature pairs (l, l°), we compute the ℓ_2 distance between $\mathbf{v}_{n,l}$ and $\mathbf{v}_{n,l^{\circ}}^{\circ}$ and use a threshold ξ to filter them. On the survived features , we further enumerate all quadruples $(l_1, l_2, l_1^{\circ}, l_2^{\circ})$ with l_1 matched to l_1° and l_2 matched to l_2°, compute $\Delta\mathbf{u}_{l_1,l_2}$ and $\Delta\mathbf{u}_{l_1^{\circ},l_2^{\circ}}^{\circ}$, and again filter them with a threshold ζ. Finally, we apply the Bron-Kerbosch algorithm to find the max-clique on both images that are matched with each other. the matching algorithm, the hyper-parameters ξ and ζ do not impact performance.

2.3.4.2 Testing: Fast Inference

In the testing stage, all $\mathbb{I}_k$ and $\mathbb{S}_k$ are fixed, so the viewpoint consistency term vanishes and the goal becomes:

$$Q(\mathbf{b}^*, \mathbf{s}^* \mid \mathbf{I}^*, \{\mathbb{I}_k, \mathbb{S}_k\}) = \min_k \alpha(\mathcal{M}_k^*). \tag{2.10}$$

Thus, we enumerate over all possible $\mathbb{I}_k$ and find the best solution. There are no new components introduced in this part – the feature matching process is executed, based on which a coordinate transformation function is built and transplants $\mathbb{S}_k$ into $\mathcal{S}^*$ which obtains the desired results.

Compared to Eqn (2.6) that computes feature matching between the testing image and all N training images, Eqn (2.10) only performs the computation for each of the K models. Most often, we have $K \ll N$ and so this strategy saves a large amount of computation at the testing stage.

2.3.5 Implementation Details

The rendering function $\mathbf{g}(\cdot)$ is implemented with standard rasterization in a game engine. We place the 3D model in regular background with *road* and *sky*, and use two directional light sources to reduce shadow in the rendered images (this improves image matching performance).

The transformation of the semantic part annotations is learned using their nearby matched features. For each semantic part, a weighted average of its neighboring features' relative translation is applied to the annotation, where the weights are proportional to the inverse of the 2-D Euclidean distances between the semantic part and the features in the source image.

The basis of our approach is the feature vectors extracted from a pre-trained deep network. However, these features, being computed at a mid-level layer, often suffer a lower resolution in the original image plane. For example, the *pool-4* features of VGGNet (223) used in this paper have a spatial stride of 16, which leads to inaccuracy in feature coordinates and, consequently, transformed locations of semantic parts. To improve matching accuracy,

Approach	16 Training Samples				32 Training Samples				64 Training Samples			
	L0	**L1**	**L2**	**L3**	**L0**	**L1**	**L2**	**L3**	**L0**	**L1**	**L2**	**L3**
Faster R-CNN (205)	31.64	15.57	10.51	8.35	43.05	20.72	14.81	11.95	**55.43**	29.03	19.05	13.75
DeepVoting (314)	33.28	18.09	13.92	10.26	37.66	21.91	16.35	12.12	49.23	30.01	21.86	15.52
Ours	**45.87**	**31.32**	**26.52**	**15.77**	**50.16**	**34.27**	**28.25**	**17.67**	52.23	**35.11**	**29.08**	**18.04**

Table 2.1: Semantic part detection accuracy (by mAP, %) of different approaches using different number of training samples. **L0** through **L3** indicate occlusion levels, with **L0** being non-occlusion and **L3** the heaviest occlusion.

we apply a hierarchical way of extracting regional features. The idea is to first use higher-level (*e.g.*, *pool-4*) features for semantic matching, and then fine-tune the matching using lower-level (*e.g.*, *pool-3*) features which have a smaller spatial stride for better alignment.

2.3.6 Discussions

Compared to prior methods on object detection (205) or parsing (314), our approach enjoys a higher explainability as shown in experiments (see Figure 2.5). Here, we inherit the argument that low-level or mid-level features can be learned by deep networks as they often lead to better local (297) or regional (202) descriptions, but the high-level inference stage should be semantically meaningful so that we can manipulate either expertise knowledge or training data for improving recognition performance or transferring the pipeline to other tasks. Moreover, our approach requires much fewer training data to be optimized, and applies well in novel viewpoints which are not seen in training data.

The training process of our approach can be largely simplified if we fix $\mathbb{S}$, *e.g.*, manually labeling semantic parts on each 3D model $\mathbb{I}$. However, the amount of human labor required increases as the complexity of annotation as well as the number of 3D models. Our approach serves as a good balance – the annotation on each 2D image can be transferred to different 3D models. In addition, 2D images are often annotated by different users, which provide complementary information by crowd-sourcing (65). Therefore, learning 3D models from the ensemble of 2D annotations is a safer option.

2.4 Experiments

2.4.1 Settings and Baselines

We perform experiments on the VehicleSemanticPart (VSP) dataset (259). We choose *sedan*, a prototype of *car*, which aligns with the purchased 3D models, and investigate whether our approach generalizes to other prototypes. There are 395 training and 409 testing images, all come from the Pascal3D+ dataset (274), and the authors manually labeled 39 semantic parts covering a large fraction of the surface of each *car* (examples in Figure 2.2). There are 9 semantic parts related to *wheel*, 1 at the center and other 8 around the rim. We only consider the center one as the others are less consistent in annotation. Moreover, we did not investigate other classes due to the difficulty in defining 3D models, *i.e.*, *airplane*, *bus*, and *train* images suffer substantial intra-class appearance variation, and *bike* and *motorbike* are not perfectly rigid. The same setting was used in some prior work (186)(93)(145), which focused on broad applications of *car* parsing.

We use the ground-truth azimuth angle to categorize training images into 8 bins, centered at $0^\circ, 45^\circ, \ldots, 315^\circ$, respectively. We randomly sample $N' \in \{2, 4, 8\}$ images in each bin, leading to three training sets with 16, 32 and 64 images, which are much smaller than the standard training set (395 images). In the testing stage, we provide the ground-truth viewpoint and bounding-box for each image, which often requires less than 3 seconds to annotate – in comparison, labeling all semantic parts costs more than one minute. We add quantization noise to the ground-truth azimuth and polar angles by assigning them into bins with fixed widths. This is to reduce the benefit of the algorithm in using very accurate viewpoints. In addition, following the same setting of (314), different levels of occlusion are added to each testing image.

The competitors of our approach include DeepVoting (314), a recent approach towards explainable semantic part detection, as well as Faster R-CNN (205) (also used in (314)), an end-to-end object detection algorithm. Other approaches (*e.g.*, (259) and (258)) are not listed as they have been verified weaker than DeepVoting.

2.4.2 Quantitative Results

Results are summarized in Table 2.1. One can observe that our approach outperforms both DeepVoting and Faster R-CNN, especially in the scenarios of (i) fewer training data and (ii) heavier occlusion.

2.4.2.1 Impact of the Amount of Training Data

One major advantage of our method is the ability to learn from just a few training samples by preserving the 3D geometry consistency for images from different viewpoints. As shown in Table 2.1, when using as less as 16 training images, which provides no more than 6 training samples for most semantic parts, our method still gives reasonable predictions and outperforms other baseline method by a large margin. By increasing the training sample number, our method also benefits from learning more accurate annotations on the 3D model, resulting to higher mAP in the 2D detection task. By contrast, both Faster R-CNN and DeepVoting fail easily given small number of training.

Since the viewpoint is provided for each testing image, we also design a naive overlaying algorithm which transfers semantic parts between two images with similar viewpoints, *i.e.*, each semantic part is transferred to the same relative position within the bounding box. With 16 training samples, it produces an mAP of 23.74%, about 50% lower than our approach. This reveals the effectiveness of using feature matching to guide coordinate transformation.

2.4.2.2 Ability of Dealing with Occlusion

To evaluate how robust our method is to occlusion, we apply the models learned from the occlusion-free dataset to images with different levels of occlusion. Compared to DeepVoting (314), which learns the spatial relationship between semantic parts and their characteristic deep features in occlusion-free settings, our method directly models the spatial relationship of parts by projecting them to the 3D space, and then matches them back to the occluded testing. On light occlusion, our method consistently beats the baseline methods. In the cases of heavier occlusion, due to the deficiency of accurately matched features, the performance of our method deteriorates. As expected, Faster R-CNN lacks the ability of dealing with occlusion and its performance drops quickly, while DeepVoting is affected less.

It is interesting to see that the robustness to fewer training data and occlusion is negatively related to the number of extra parameters. For example, DeepVoting has less than 10% parameters compared to Faster R-CNN, and our approach, being a stage-wise one, only requires some hyper-parameters to be set. This largely alleviates the risk of over-fitting (to small datasets) and the difficulty of domain adaptation (to occlusion data).

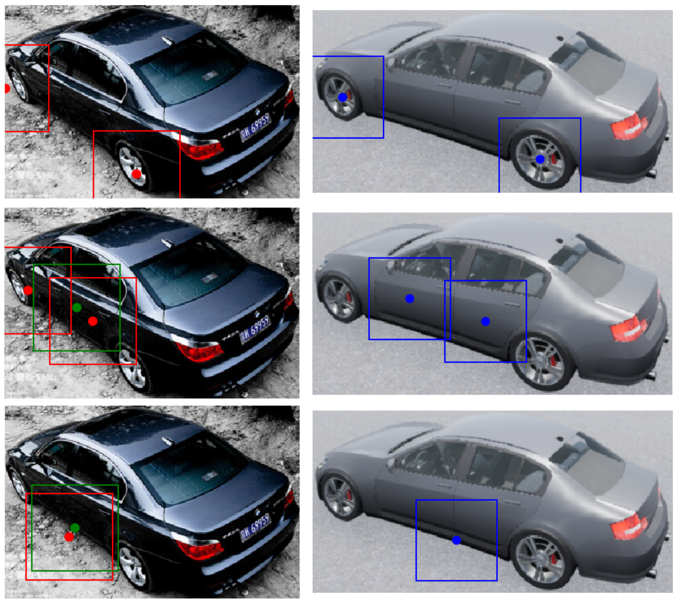


Figure 2.3: Example of predicting on unseen viewpoint. Left: test image with transferred annotation (red) and ground truth (green). Note that no ground truth for wheels is given for the current image, which is an annotation error. Right: viewpoint matched synthetic image with semantic part annotations learned from training data.

2.4.2.3 Predicting on Unseen Viewpoints

To show that our approach has the ability of working on unseen viewpoints, we train the models using *sedan* images with various azimuth angle and 0° elevation angle, then test them on *sedan* with elevation angle equal or larger than 10°. The results are shown in Table 2.2. Our method maintains roughly the same mAP as tested on all viewpoints, while

Approach	# Training Samples		
	16	32	64
Faster R-CNN	16.02	21.80	19.91
DeepVoting	8.59	27.71	33.82
Ours	**48.96**	**50.86**	**49.54**

Table 2.2: Semantic part detection accuracy (by mAP, %) of different approaches on unseen viewpoints.

FasterRCNN and DeepVoting deteriorate heavily. In Figure 2.3, we show predictions made by our method on one sample with unseen viewpoint (elevation angle equals 20°). The predicted locations are very close to the annotated ground truth, and may help to fix the annotation error in the dataset.

2.4.2.4 Transfer across Different Prototypes

In order to evaluate how sensitive our method is to the car prototype used during training (and 3D reconstruction), we transfer the *sedan* model with semantic parts to other prototypes of cars. Results are summarized in Table 2.3. As expected, our method generalizes fine to prototypes with similar appearance (e.g., *SUV*). For *minivan* and *hatch-back*, due to the variation of their 3D structures and semantic part definitions, the performance drops more. Similar results are observed from Faster R-CNN and DeepVoting, and DeepVoting seems slightly more robust to the prototypes.

2.4.3 Qualitative Studies

2.4.3.1 Viewpoint Consistency in Training

In Figure 2.4, we show examples on how viewpoint consistency improves the stability of the training stage. Although we have applied 2-D geometry coherence as one of the criteria

Approach	*sedan*	*SUV*	*mini-van*	*hatch-back*
Faster R-CNN	43.05	41.16	32.18	30.04
DeepVoting	37.66	37.18	30.67	31.62
Ours	**50.16**	**44.39**	**39.41**	**34.91**

Table 2.3: Semantic part detection accuracy (by mAP, %) of different approaches on other car prototypes. All models are trained using 32 *sedan* images.

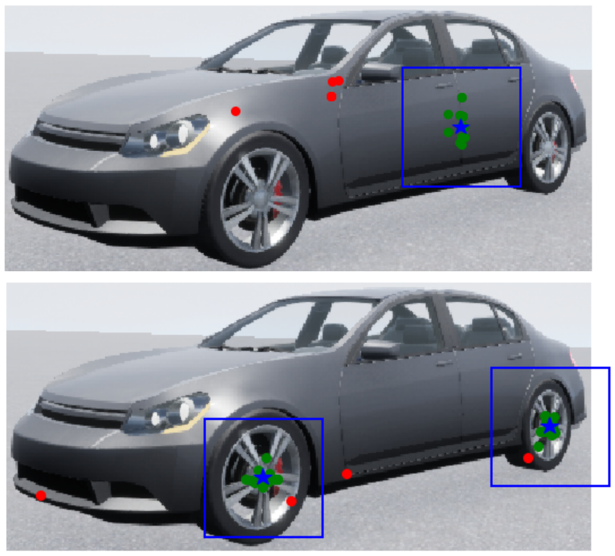


Figure 2.4: Two examples of how viewpoint consistency improves the semantic part annotation on 3D model. The red circles represent incorrectly transferred semantic part annotations that get eliminated during our aggregation process using 3-D geometry constraints. The green circles are the reasonable annotations that are used to get the final annotation for the targeted semantic part, which is represented by the blue stars with bounding-boxes.

during matching individual training samples to their viewpoint-paired synthetic images, it is possible to get wrong matched features at inaccurate positions. Therefore, the semantic part annotations transferred from an individual training image could be far off the ground truth area (e.g., outliers shown by the red circles). With viewpoint consistency, the incorrect annotations are eliminated during aggregation, and our method stably outputs the right position for the targeted semantic parts (e.g., final annotation shown by blue stars) based on the reasonably transferred annotations (e.g., inliers shown by green circles).

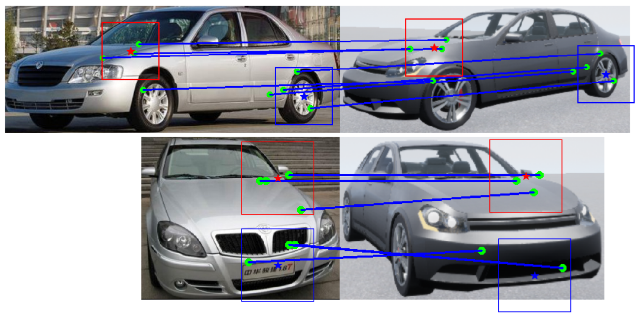


Figure 2.5: Interpret semantic part detection. Stars with bounding-boxes represent the semantic parts that are transferred from the synthetic image (right) to the testing image (left), based on transformation learned from features (green circles, matched features are linked by red lines) in their neighborhood.

2.4.3.2 Interpreting Semantic Part Detection

Next, we provide some qualitative results to demonstrate the explainability of our approach. In Figure 2.5, we show examples of how we locate the semantic parts in two image pairs. Each pair includes one testing image and its viewpoint-matched synthetic image.The star represents the location of the semantic parts in each image (learned from training in the synthetic images and got transferred to in the testing images), and the color represent their identity. The transformation is learned using nearby matched features, which are shown by green circles (matched features are linked by red lines). For better visualization, we only display the nearest three features for each semantic part in the figure. This explains what features are used to transfer the annotation from synthetic images to testing images, and helps us understand what is going on during the inference process.

2.5 Conclusions

In this paper, we present a novel framework for semantic part detection. The pipeline starts with extracting regional features and applying robust matching algorithms to find correspondence between images with similar viewpoints. To deal with the problem of limited training data, an additional consistency loss term is added, which measures how semantic part annotations transfer across different viewpoints. By introducing a 3D model as well as its viewpoints as hidden variables, we can optimize the loss function using an iterative algorithm. In the testing stage, we directly apply the same algorithms to match the semantic parts from the 3D model back to each 2D image, and achieve high efficiency in the testing stage. Experiments are performed to detect semantic parts of *car* in the VSP dataset. Our approach works especially well with very few (*e.g.*, tens of) training images, on which other competitors (205)(314) often heavily over-fit the data and generalize badly.

Our approach provides an alternative solution to object parsing, which has three major advantages: (i) it can be trained on a limited amount of data and generalized to unseen viewpoints; (ii) it can be trained on a subset of viewpoints and then transferred to novel ones; and (iii) it can be assisted by virtual data in both training and testing. However, it still suffers from the difficulty of designing parameters, which is the common weakness of stepwise methods compared to the end-to-end learning methods. In other words, a lot of work is yet undone towards the balance of abilities of learning and explanation.

Researchers believe that 3D is the future direction of computer vision. In the intending research, we will try to learn one or more 3D models directly from 2D data, or allow the 3D model to adjust slightly to fit 2D data. More importantly, it is an intriguing yet challenging topic to generalized this idea to non-rigid objects, which will largely extend its area of application.

Chapter 3

CoKe: Contrastive Learning for Robust Keypoint Detection

In this part of the work, we focus on making keypoint detection more robust. Keypoint detection involves locating important landmarks or keypoints on an object. We introduce a new method called CoKe that uses contrastive learning, which learns representations by bringing similar examples close together and pushing dissimilar ones apart, and it enhances the robustness and generalization of keypoint detection across diverse real-world data.

3.1 Introduction

Semantic keypoints, such as the joints of a human body, provide concise abstractions of visual objects in terms of their shape and pose. Accurate keypoint detections are of central importance for many visual understanding tasks, including viewpoint estimation (189), human pose estimation (29), action recognition (171), feature matching (163), image classification (309), and 3D reconstruction (132). There are many diverse approaches for keypoint detection. Common approaches include the application of a regression loss

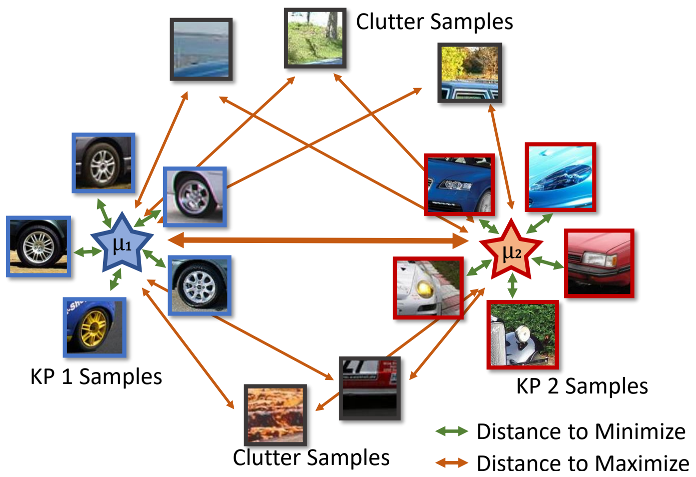


Figure 3.1: Intuition behind our approach. Image patches depict feature representations of two different keypoints (blue and red border) and background clutter (grey border). The star shapes illustrate the average representations μ_1 and μ_2 of the corresponding keypoint features. Our approach learns a representation space such that the following three distances are optimized: (1) The distance between features of the same keypoint is small, i.e. they cluster tightly around their mean. (2) The distance between the keypoint clusters is maximized. (3) The distance between the clutter features and the keypoint centers is maximized.

(180; 133), a classification loss (109), or combinations of either of those with a 3D geometric model of the object (321). In recent years, work in contrastive learning has led to major advances in representation learning (39; 106; 174) demonstrating benefits over classical losses, such as cross-entropy, e.g. in terms of robustness and data efficiency (134). However, most works on contrastive learning in computer vision focus on the task of image classification, and it remains unclear how contrastive learning can be applied for keypoint detection.

In this paper, we introduce a contrastive learning framework for keypoint detection (CoKe). Keypoint detection differs from other visual tasks where contrastive learning has been applied, such as face recognition (213) or unsupervised learning (106), because the input is a set of images in which multiple keypoints are annotated. To enable the contrastive learning of keypoint detectors, we need to represented and detected keypoints independently, such that a contrastive loss can make the keypoint features different from each other and from the background. This is very different from current popular approaches to keypoint detection, such as stacked hourglass networks (**?**), which attempt to detect all keypoints jointly. Our approach has two benefits: It enables us to exploit the power of contrastive learning for keypoint detection, and by detecting each keypoint independently the detection can become more robust to occlusion compared to holistic methods (as shown in our experiments).

Our CoKe framework introduces several technical innovations. Specifically, we found that the contrastive learning of keypoint representations requires the optimization of three types of distances in the feature space (Figure 3.1):

(1) The distance between features of the same keypoint should be small. But the

computational cost to calculate the distance between all features of the same keypoint is quadratic in their number. We instead reduce the computational cost to be linear by introducing an average prototypical representation for each keypoint (stars in Figure 3.1).

(2) The distance between features of different keypoints should be large. The number of distance comparisons between features of different keypoints is combinatorial in the number of keypoints and training images. To manage this computational burden, we introduce a *keypoint bank* which stores the prototypical representations of all keypoints and allows for an efficient computation of the distance between the prototypes of different keypoints (Figure 3.1 bold orange arrow).

(3) The distance between keypoint features and features from the background clutter (Figure 3.1 grey squares) should be large to reduce false-positive detections. However, most of the features in an image are clutter features and it is not feasible to compute the distance to all of them. Therefore, we introduce a *clutter bank* that keeps track of clutter features that are spatially close to keypoint features and hence are most difficult to be distinguished from.

The proposed approximations enable an efficient contrastive learning of keypoint detectors. We evaluate CoKe on several datasets including PASCAL 3D+, MPII, and ObjectNet3D. We observe that CoKe performs on par, and often even better, compared to SOTA related work (Stacked Hourglass Networks, MSS-Net (133)) and to approaches that use additional supervision in terms of detailed 3D object geometries (StarMap (321)). These results are remarkable as CoKe works well on all these datasets while, for example, the best results on MPII are often achieved by architectures that are specialized for human keypoint detection. We also observe that when compared to related work, CoKe is exceptionally robust to partial occlusion and unseen object poses. Our main contributions are:

1. We introduce a framework for keypoint detection that benefits from the power of contrastive learning.

2. CoKe performs very well on various keypoint detection datasets for rigid and articulated objects.

3. CoKe achieves exceptional robustness to partial occlusion and previously unseen object poses.

3.2 Related Work

Keypoint Detection. Keypoint detection, is a widely studied problem in computer vision. Popular applications are, e.g., the detection of human joints (29; 180; 239; 240) or distinct locations on rigid objects (270; 247; 189; 321). Early approaches relied on local descriptors (96; 211; 297; 52; 83) that are distinctive and invariant (165). While approaches using local descriptors have proven to be robust to occlusion and background clutter, they were outperformed by deep learning approaches that were trained end-to-end (180). Toshev et al. (240) first trained a deep neural network for 2D human pose regression and Li et al. (146) extended this approach to 3D. Starting from the work of Tompson et al. (239), regression-based approaches to keypoint detection became very popular. They perform keypoint detection by regressing a heatmap representation. These approaches achieve a particularly good performance at detecting the joints of both articulated and rigid objects (180), because they can implicitly leverage the structural information between keypoint to resolve locally ambiguous keypoint detections. Tulsiani et al. (247), proposed to integrate the structural information between keypoints explicitly by integrating 2D and 3D models, which inspired a number of follow-up works, in particular for rigid objects (321; 247; 189).

Supervised Contrastive Learning. Contrastive learning, originates from Metric Learning (35; 268; 206) and involves the learning of a representation space by optimizing the similarities of sample pairs in this space. Intuitively, supervised contrastive learning aims to reduce the distance of feature representations of the same class, while increasing the distance between samples from different classes. Popular examples use pairs of samples for loss computation (101), triplets (213), or N-Pair tuples (225).

Recently, contrastive learning has attracted attention from the research community in self-supervised learning (39; 106; 272; 174; 113). The main difference in the self-supervised learning setting is that positive examples are usually generated using data augmentations(57) or co-occurrence (121; 235) of a query sample, whereas negative examples are chosen as other images in the same mini-batch.

While most of the supervised contrastive learning (134) focuses on learning a holistic representation of the complete image, in this paper, we target a more fine-grained task - keypoint detection. Keypoints are localized image patterns and therefore require the learning of local feature embeddings. The main challenge is that local image patterns can be highly ambiguous (e.g. the front and back tire of a car) and therefore require a contrastive learning framework that can learn to disambiguate local representations, while at the same time being able to learn a distinct representation that can be localized accurately.

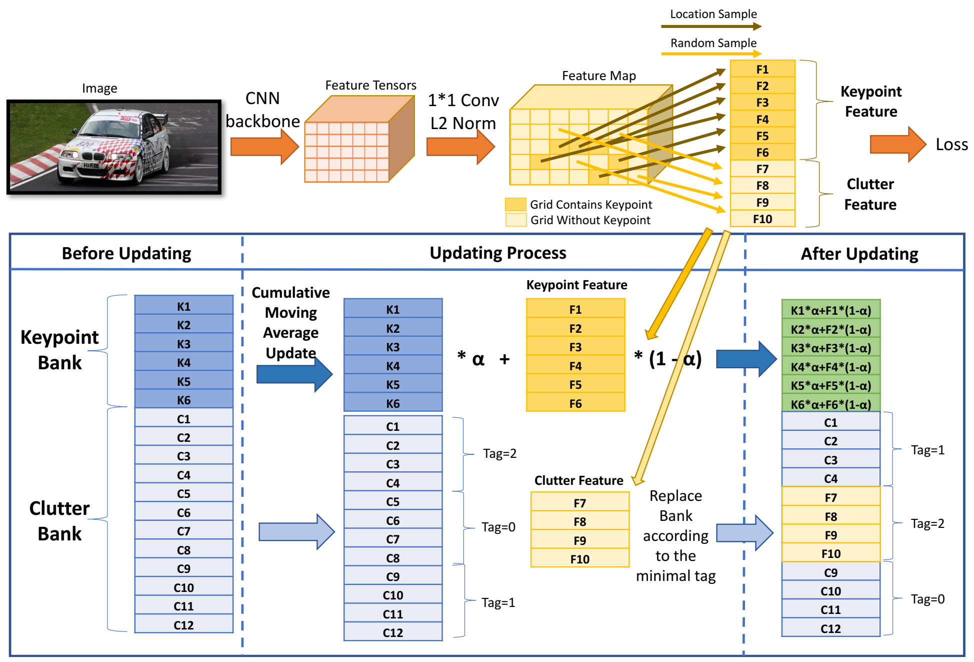


Figure 3.2: Illustration the Keypoint and Clutter Bank Updating Process. First, a feature map for the input image is extracted. After a dimensionality reduction and L_2 normalization, we retrieve the keypoint features (F1-F6) and randomly selected clutter features (F7-F10). The Keypoint Bank is updated using a cumulative moving average update. The Clutter Bank is updated by replacing the oldest features in the Clutter Bank with (F7-F10) based on the time tag.

3.3 CoKe: Contrastive Keypoint Learning

In this section, we present our framework for contrastive keypoint learning, then discuss the intuition underlying our approach and the training pipeline and how to perform keypoint detection during inference.

3.3.1 Training CoKe

We use Φ to denote the feature extractor. Given an input image $\mathbf{I}^i$, it computes the feature map $\Phi(\mathbf{I}^i) = \mathbf{F}^i \in \mathbb{R}^{H \times W \times D}$, where i is the image index in the training data

$\{\mathbf{I}^i | i \in \{1, \dots, N\}\}$. Using the keypoint annotation we retrieve the corresponding keypoint features $\mathbf{f}_k^i \in \mathbb{R}^D$ for the keypoints $k \in \mathbb{K}$ from the feature map $\mathbf{F}^i$. Similarly, we can (randomly) select a non-keypoint location as clutter point and retrieve a set of clutter features $\{\mathbf{f}_c^i \in \mathbb{R}^D | c \in \{1, \dots, C\}\}$. We define the distance between two features as $d(\cdot, \cdot)$. During training we learn a feature extractor that optimizes the following distances in the feature space:

(1) One objective for the feature extractor during training is to **minimize the distance between features of the same keypoint** (i.e. the intra-keypoint distance) across all training images, by minimizing the objective:

$$D_{intra}(\mathbf{f}_k^i) = \sum_{j=1}^{N} d(\mathbf{f}_k^i, \mathbf{f}_k^j) \approx d(\mathbf{f}_k^i, {}^{-}_{k}). \tag{3.1}$$

However, as we described in the introduction it is unpractical to compute the distance of a keypoint's feature vector from one image $\mathbf{f}_k^i$ to the corresponding vectors in all other training images $j \in \{1, \dots, N\}$. To resolve this computational problem, we define a prototypical keypoint feature ${}^{-}_{k}$, that represents the average feature of keypoint k. Instead of computing the full objective, we approximate it as simply the distance to the corresponding average representation $d(\mathbf{f}_k^i, {}^{-}_{k})$. We store the prototypical features of all keypoints $\{\mu_k\}$ in a *keypoint bank* and update them during training.

(2) The second objective for the feature extractor is to **maximize the distance between features of different keypoints** (i.e. the inter-keypoint distance). This requires computing the distance between the feature representations of one particular keypoint k and all other

keypoints $k^{'}$ over all training images:

$$D_{inter}(\mathbf{f}_k^i) = \sum_{k' \in K \setminus \{k\}} \sum_{j=1}^{N} d(\mathbf{f}_k^i, \mathbf{f}_{k'}^j) \approx \sum_{k' \in K \setminus \{k\}} d(\mathbf{f}_k^i, \text{¯}_{k'}). \tag{3.2}$$

We approximate this objective by instead computing the distance to the prototypes of the respective keypoint features from the keypoint bank.

(3) The third objective for the feature extractor is to **maximize the distance between the keypoint features and all clutter features**. In the best case, this involves computing the distance between a keypoint feature $\mathbf{f}_k^i$ to every clutter feature in all training images. To avoid the computation of this large amount of distances, we instead approximate this objective by storing a subset of all clutter features $\{\text{`}_c, c \in \mathbb{C}\}$ in a *clutter bank*. Which allows us to approximate the full objective with

$$D_{clutter}(\mathbf{f}_k^i) \approx \sum_{c \in \mathbb{C}} d(\mathbf{f}_k^i, \text{`}_c). \tag{3.3}$$

These three approximations make it feasible to optimize the overall objective with a feasible computational load. As the parameters of the feature extractor will change during learning, the clutter features $\mathbf{f}_c$ in the clutter bank as well as the prototypes ¯_k need to be updated. To achieve this, we follow an EM-type optimization process. First, we initialize the clutter bank by randomly sampling clutter features from the training data, and initialize the prototypes by computing the average feature for every keypoint across the training data $\mu_k = \frac{\sum_{i=1}^{N} \mathbf{f}_k^i}{N}$. Using these initial estimates, we can compute the overall objective and train the feature extractor. While training the feature extractor, we update the clutter and keypoint bank. We perform those updates in an alternating manner.

3.3.1.1 Keypoint and Clutter Bank Update

Figure 3.2, illustrates the process of updating the keypoint prototypes and the clutter bank during training.

Keypoint Bank Update. Computing the prototypical keypoint features $\bar{\ }_k$ while learning the feature extractor is challenging, because we want to avoid to re-compute the prototypes over all training images $\mu_k = \frac{\sum_{i=1}^{N} \mathbf{f}_k^i}{N}$ after every gradient step. Instead, we approximate the sample mean via a cumulative moving average. Specifically, we update $\bar{\ }_k$ with a training batch of size m using:

$$\bar{\ }_k \leftarrow \bar{\ }_k * \alpha + \frac{\sum_{i=0}^{m} \mathbf{f}_k^i}{m} * (1 - \alpha). \tag{3.4}$$

Clutter Bank Update. The clutter bank contains a limited number C of clutter features $\{\grave{\ }_c, c \in \mathbb{C}\}$. In practice the size of the clutter bank depends on the availability of GPU memory and we observe that the larger the bank, the better the training performance (see experiments section). We update the clutter bank during training by replacing the oldest clutter features with newly extracted features from the current training batch based on a time tag that indicates how long features were stored in the bank.

3.3.1.2 Feature Extractor Training

When training the feature extractor with SGD, we freeze the keypoint bank and clutter bank in every gradient step and use them to compute the gradient update on the weights of the feature extractor. To compute the distance between two feature vectors, we use the L2 distance:

$$d(\mathbf{f}_a, \mathbf{f}_b) = (\mathbf{f}_a - \mathbf{f}_b)^2 = 2 * (1 - \mathbf{f}_a \cdot \mathbf{f}_b). \tag{3.5}$$

The last step in above equation leverages that all features in our model are L2 normalized. From Eq. 3.5 observe that we can minimize $D_{intra}(\mathbf{f}_k^i)$ by maximizing $\mathbf{f}_k^i \cdot \text{¯}_k$. Similarly, we can maximize $D_{inter}(\mathbf{f}_k^i)$ and $D_{clutter}(\mathbf{f}_k^i)$ by minimizing $\{\mathbf{f}_k^i \cdot \text{¯}_{k'} | \forall k' \in K \setminus \{k\}\}$ and $\{\mathbf{f}_k^i \cdot \text{`}_c \, | \forall c \in \mathbb{C}\}$ respectively. To optimize those terms simultaneously, we use a non-parametric softmax as our loss function. Thus, the loss for each keypoint feature is calculated as:

$$\mathcal{L}(\mathbf{f}_k^i, \{\mu_k\}, \{\text{`}_c\}) = \frac{e^{\mathbf{f}_k^i \cdot \text{¯}_k}}{\sum_{k' \in K} e^{\mathbf{f}_k^i \cdot \text{¯}_{k'}} + \sum_{c' \in C} e^{\mathbf{f}_k^i \cdot \text{`}_{c'}}}, \tag{3.6}$$

where $\{\mu_k\}$ is the keypoint bank and $\{\theta_c\}$ is the clutter bank.

Clutter Sampling Loss. A technical problem is that the features in the clutter bank are copied from a large number of training images, and therefore it is not practical to calculate gradient directly w.r.t. $\{\theta_c\}$, and therefore the feature extractor is not optimized w.r.t. the clutter features. This technical limitation makes the optimization w.r.t. the clutter distance $D_{clutter}(\mathbf{f}_k^i)$ converge slowly, especial when the clutter bank is large. To make the training more efficient, we propose a clutter sampling loss, which uses the sampled clutter features $\mathbf{f}_c^i$ from the current training batch and directly maximizes the clutter to keypoint distance:

$$\mathcal{L}(\mathbf{f}_c^i, \{\mu_k\}) = \sum_{k' \in K} \mathbf{f}_c^i \cdot \text{¯}_{k'}. \tag{3.7}$$

Thus, the final loss for training the feature extractor becomes:

$$\mathcal{L}(\mathbf{F}^i,\{\mu_k\},\{\text{`}_c\})=\sum_{k\in\mathbb{K}} \mathcal{L}(\mathbf{f}_k^i, \{\mu_k\}, \{\text{`}_c\})+\sum_{c\in\mathbb{C}} \mathcal{L}(\mathbf{f}_c^i, \{\mu_k\}). \tag{3.8}$$

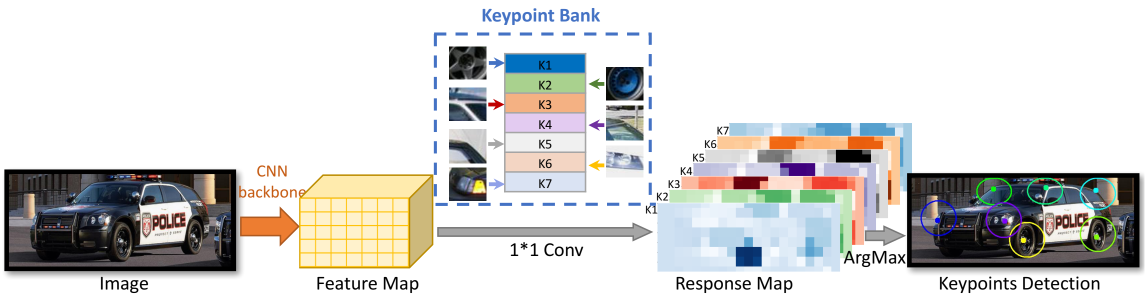


Figure 3.3: Keypoint detection with CoKe. We first extract feature representation at different positions of input image. Each keypoint in the Keypoint Bank has an individual representation used as a convolution kernel to compute a response map. The location of maximum response is used as prediction result. Colored boxes show ground truth and dots the prediction result.

3.3.2 Inference with the CoKe Model

In the following, we describe the detection process for keypoints of a single category, but note that this can be trivially extended to multiple categories. Figure 3.3 illustrates the inference process with the CoKe. To locate the predicted the keypoint position p_k for keypoint k on the feature map of a test image $\mathbf{I}^i$, we apply the following steps:

Extract a feature map $\Phi(\mathbf{I}^i) = \mathbf{F}^i$ using the trained feature extractor Φ. Use the prototypes $^{-}_{k}, k \in \mathbb{K}$ to compute the per pixel feature distance $d(\mathbf{f}, ^{-}_{k})$ for all feature vectors $\mathbf{f} \in \mathbf{F^i}$ and store the detection scores in the output $\mathbf{S} \in \mathbb{R}^{H \times W \times K}$. For each keypoint, select the position p with the highest detection score in $\mathbf{S}$. Project p back to the original image coordinates.

3.4 Experiments

In this section, we experimentally evaluate CoKe and compare it to related works. We first describe the experimental setup, compare CoKe to related work on several diverse datasets, and study their robustness to partial occlusion. Then discuss qualitative results with ablation studies.

3.4.1 Experimental Setup

Evaluation Protocol. Evaluation is done using the standard Percentage of Correct Keypoints (PCK) metric which reports the percentage of detections that fall within a normalized distance of the ground truth. we use PCK=0.1 for PASCAL3D+, ObjectNet3D and OccludedPASCAL3D+ datasets following the standard experimental protocol. For MPII, we use PCKh=0.5 as the evaluation metric. Distance is normalized by a fraction of the head size (PCKh). We follow the common protocol of evaluating each object category by computing the average accuracy on the visible keypoints over all the test images.

Training Setup. We use the standard train-val-test split for all the datasets. We use a batch size of 64 for training. For each image, we randomly select a group of 20 clutter points. The full clutter bank consists of 1024 such groups. We choose the clutter features to be within two pixels distance to the keypoint annotation in the feature map. We use the non-parametric softmax (272) to calculate the similarity between features and banks. The temperature parameter(120) that controls the concentration level of the distribution is set to $\tau = 0.7$.

PASCAL3D+ Dataset. The dataset contains 12 man-made object categories with totally training 11045 images and 10812 evaluation images. Different from previous works (321; 247), we use all images for evaluation, including occluded and truncated ones.

MPII Dataset. MPII Human Pose (4) consists of images taken from a wide range of human activities with a challenging array of articulated poses. The keypoint visibility is

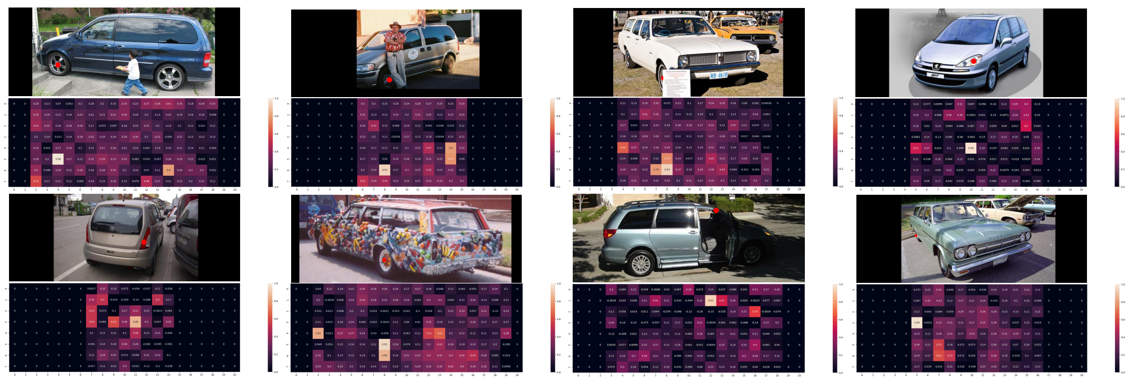


Figure 3.4: Eight examples of CoKe-Res50's representation visualization. For each sub-figure, top is the original image with keypoint annotation, labeled with red dot. Bottom is the response map, predicted by CoKe-Res50. It is worth noting that how all keypoints are detected accurately despite the difficulty from false positives, occlusions, rare viewpoints, different domain, irregular appearance and irregular state and rare viewpoints.

annotated, enabling us to report numbers for the full dataset as well as partially occluded humans.

ObjectNet3D Dataset. ObjectNet3D consists of common daily life objects and is notably more difficult compared to PASCAL3D+ as it contains more rare viewpoints, shapes and truncated objects with occlusion. We test on nine categories, chosen based on their high annotation accuracy, as some categories in ObjectNet3D have low quality annotations due to the complexity of this dataset.

OccludedPASCAL3D+ Dataset. While it is important to evaluate algorithms on real images of partially occluded objects, simulating occlusion enables us to quantify the effects of partial occlusion more accurately. We use an analogous dataset with artificial occlusion for keypoint detection proposed in (257) for object detection. It contains all 12 classes of the PASCAL3D+ dataset at various levels of occlusion. The dataset has a total of 3 occlusion levels, with Lv.1: 20-40%, Lv.2: 40-60% and Lv.3: 60-80% of the object area being occluded.

3.4.2 Performance on Various Datasets

PASCAL3D+ and OccludedPASCAL3D+. Table 3.1 shows the keypoint detection results on the PASCAL3D+ dataset for CoKe models learned from three different backbones: ResNet-50 (110), Stacked-Hourglass-Network and Res-UNet (313). We also show the performance of StarMap (321) as reported in the original paper. Note that StarMap uses 3D models as additional supervision to jointly reason about the relative position of the keypoints. The performance of CoKe with all backbones is constantly high. The highest performance is achieved with the most recently developed architecture Res-UNet. When compared to the original Stacked-Hourglass-Network trained with a regression loss (SHG) we can clearly observe a large gain in performance. Most notably, the performance difference is very prominent for strong occlusion. We believe CoKe is more robust to occlusion, because of two reasons: 1) It is actively optimized to discriminate between keypoint features and clutter features based on their local representations only, which might help to reduce the effective receptive field size and hence reduces the negative effects of occlusions. 2) CoKe stores the clutter representation from a large number of images, and hence can better learn to distinguish keypoints from clutter. Overall, our results clearly highlight that CoKe is very competitive with related work, while being highly robust to partial occlusion.

In Table 3.2, we also perform an experiment to demonstrate additional robustness to **unseen object poses**. We divide the azimuth pose in the *car* category of Pascal3D+ into 4 bins, front, back, left and right. We train CoKe on the front and back subsets, and test on seen (7305 images) and unseen (3507 images) poses separately. Table 3.2 shows that CoKe is much more robust to unseen poses compared to SHG.

PASCAL3D+					
Occlusion Level	Lv.0	Lv.1	Lv.2	Lv.3	Avg
SHG	68.0	46.5	43.2	39.9	49.4
MSS-Net	68.9	46.6	42.9	39.6	49.5
StarMap	78.6	-	-	-	-
CoKe-Res50	77.0	67.6	**59.9**	53.4	64.4
CoKe-SHG	78.3	66.3	58.4	52.3	63.8
CoKe-ResUnet	**80.3**	**68.5**	59.1	**54.0**	**65.5**

Table 3.1: Keypoint detection results on PASCAL3D+ under different levels of partial occlusion (Lv.0:0%,Lv.1:20-40%,Lv.2:40-60%,Lv.3:60-80% of objects are occluded, L0 is the original dataset). CoKe outperforms baseline models and is highly robust to partial occlusion.

PASCAL3D+		
	Seen-Pose	Unseen-Pose
CoKe-SHG	94.0	**84.0**
SHG	**94.2**	73.6

Table 3.2: Robustness of keypoint detectors to unseen poses. For the *car* category, we separate PASCAL3D+ training and testing set into 4 bins using azimuth annotations. We train CoKe on the front and back subsets, and test under both seen and unseen poses.

MPII. We compare CoKe learned from the SHG backbone to a number of related works on MPII in Table 3.3. CoKe-SHG is again highly competitive and outperforms related work by a small but significant margin. Table 3.4 compares CoKe-SHG and SHG for both occluded and non-occluded keypoint detection on the MPII dataset. We can observe that CoKe achieves significantly higher results on occlusion scenarios compared with SHG.

ObjectNet3D. We compare SHG and a CoKe-SHG on ObjectNet3D in Table 3.5. CoKe-SHG outperforms SHG by a large margin on every category. Since ObjectNet3D dataset contains much more challenging scenarios, we claim that CoKe is more robust.

MPII	
RecurrentPose(19)	88.1
PoseMachines(267)	88.5
DeeperCut(127)	88.5
PartHeatmap(27)	89.7
SHG(180)	90.9
DualPathNetworks(181)	91.2
PoseRegression(27)	91.2
CoKe-SHG	**91.4**

Table 3.3: Keypoint detection results on MPII compared with regression based algorithms.

MPII		
	Full	Occluded
SHGs	90.1	84.3
CoKe-SHG	**91.4**	**86.7**

Table 3.4: Keypoint detection results on MPII of CoKe-SHG and SHG on both original and challenging scenarios.

In summary, we observe that CoKe is a general purpose framework that constantly achieves a very high performance across a wide range of backbone architectures and for a range of datasets with very different characteristics, while also being highly robust to occlusion.

3.4.3 Qualitative Results

Visualization of Keypoint Detection Maps. We visualize the part detection maps from CoKe-Res50 in Figure 3.4. Images are selected from the *car* category in PASCAL3D+ which has complex changes in pose, illumination and object structure. Note how all keypoints are detected accurately despite the difficulty from false positives, occlusions, rare viewpoints, different domain, irregular appearance. We can observe that CoKe is robust in

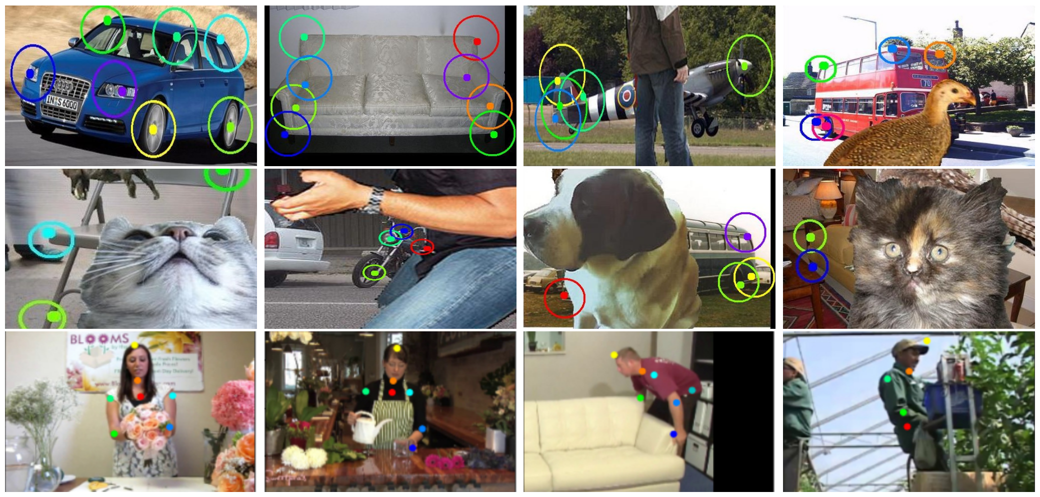


Figure 3.5: Qualitative detection results under different levels of partial occlusion for artificially occluded objects from PASCAL3D+ (a-d) and humans from MPII (e). The dots visualize the detection result of CoKe. The colored circles in (a-d) indicate the ground-truth position within PCK=0.1, and in (e) indicate ground-truth position within PCKh=0.5. Note how CoKe is very robust even under strong occlusion.

these challenging scenarios.

Visualization of Detection Results under Occlusion. We visualize qualitative results in Figure 3.5. Overall, the illustrations demonstrate the robustness of CoKe to partial occlusions. Any keypoints that are not in the vicinity of occluders are correctly detected and not affected by the occlusion. Furthermore, keypoints that are partially occluded (e.g. the wheel) can still be located robustly, although the detections tend to move away from the occluder. Importantly, we do not observe false positive detections at locally ambiguous keypoints. This demonstrates that CoKe leverages the receptive field to disambiguate keypoints, while still being able to localize individual keypoints accurately.

Inference Time and Memory Consumption. During inference, CoKe-Res50 (params: 23M, acc: 77%) takes 0.01s per image, while SHGs (params: 25M, acc: 68%) needs 0.06s.

ObjectNet3D					
	coffee	dryer	kettle	jar	wash
SHG	31.0	35.1	32.2	41.9	33.9
CoKe-SHG	**36.4**	**37.6**	**37.8**	**45.2**	**36.1**
	can	calc	eyegls	guitar	mean
SHG	66.8	52.3	44.6	45.8	42.62
CoKe-SHG	**70.2**	**57.7**	**51.3**	**49.6**	**46.88**

Table 3.5: Keypoint Detection results on ObjectNet3D+ (273).

For the memory consumption, CoKe-Res50 needs 715MB, SHGs 786MB when batch size equals to 1. CoKe has an advantage of the inference time while maintaining the competence of the memory consumption.

3.4.4 Ablation Study

Comparison with Other Losses. In Table 3.6, we ablate the effect of the contrastive learning compared to other common losses on different backbone architectures. Specifically, we compare to a regression loss as used in Stacked-Hourglass-Networks and a supervised classification loss. The classification baseline enables us to compare the effect of the contrastive learning with a cross-entropy loss. We implement keypoint detection as a classification problem by using the features at the ground truth keypoint locations in the training data as instances of different classes to train the backbone with the cross-entropy objective. During testing, we use the average representations from the training set as part detector, similar as in the CoKe pipeline. The main difference to the CoKe training is hence that the contrastive loss and the clutter bank are not used. From the results, we observe that our representation learning formulation of keypoint detection constantly outperforms other losses by a large margin.

PASCAL3D+			
Backbone	Reg	Class	CoKe
SHGs	68.0	67.8	**78.3**
Res50	68.9	69.3	**77.0**
ResUnet	69.7	70.2	**80.3**

Table 3.6: Ablation study of CoKe training strategy in comparison of standard regression or classification training loss.

Occlusion Level	Lv.0	Lv.1	Lv.2	Lv.3
No clutter	79.3	75.4	71.8	65.8
Image-specific clutter	92.8	82.7	76.5	69.2
Clutter Bank (64 groups)	93.0	83.6	**80.1**	**73.3**
Clutter Bank (256 groups)	94.3	84.3	77.7	71.0
Clutter Bank (1024 groups)	**95.5**	**85.9**	79.0	70.6
Clutter Bank w/o clutter sampling loss	94.2	83.1	76.8	68.0

Table 3.7: Ablation study on PASCAL3D+ with different settings for contrastive learning: no clutter features, image-specific clutter features (using 20 features from the same image as clutter examples), our proposed Clutter Bank with different number of groups (each group contains 20 features) and deactivating clutter sampling loss. Note the benefit of using clutter features in general, and in particular using a large clutter bank, as well as the importance of the clutter sampling loss.

Clutter Bank Mechanism In Table 3.7, we study the influence of the clutter features and the clutter sampling loss on the contrastive learning result. In particular, the table shows the keypoint detection results for CoKe-Res-UNet on the car category of the PASCAL3D+ dataset. We observe that the performance decreases significantly when no clutter features are used during training. An extension of this basic setup is to use image-specific clutter features but without maintaining the features in a Clutter Bank. In particular, we select the clutter features from the same image from which the keypoint features are sampled using the same hard negative sampling mechanism as in our standard setup. From the results we observe that using the image-specific clutter features increases the performance significantly. However, the best performance is achieved using our proposed Clutter Bank mechanism. In particular, the results show a general trend that the more features we store in the bank, the higher the performance becomes. Notably, lower occlusion scenarios benefit from a larger clutter bank while for stronger occlusion a smaller clutter bank is more beneficial. Finally, our ablation shows that explicitly regularizing the clutter to be distinct from the Keypoint Bank using the clutter sampling loss is highly beneficial.

Cumulative Moving Average Update We also study the importance of the cumulative moving average update. Here we provide another possible method: average approximation. In particular, we calculate an average over the whole dataset for $\{`\}$ each 10 epochs. We report the result using CoKe-Res-UNet on the car category of the PASCAL3D+ dataset. We observe a deduction of keypoint detection accuracy as L0: 94.2 $\rightarrow$ 88.7, L1: 83.1 $\rightarrow$ 80.0, L0: 76.8 $\rightarrow$ 75.0, L0: 68.0 $\rightarrow$ 68.9. Conclusively, the best performance is achieved using our proposed cumulative moving average update mechanism.

3.5 Conclusion

In this paper, we study keypoint detection from the perspective of contrastive learning. Our contrastive keypoint learning framework (CoKe) makes several efficient approximations to enable the contrastive representation learning for keypoint detection: (i) A clutter bank to approximate non-keypoint features; (ii) a keypoint bank that stores prototypical representations of keypoints, to approximate the within-keypoint distance; and (iii) a cumulative moving average update to learn the keypoint prototypes while training the feature extractor. Our experiments on several diverse datasets (PASCAL3D+, MPII, ObjectNet3D) show that CoKe is very general and performs well for rigid as well as articulated objects.

Compared to related work, CoKe achieves higher performance, while also being much more robust to partial occlusion and unseen object poses.

Chapter 4

Are Transformers More Robust Than CNNs?

As we continue our investigation into the generalization ability of vision models, we now pivot towards a comparative analysis between transformers and CNNs. By conducting fair comparisons and delving into the intricacies of adversarial robustness and generalization, this chapter aims to compare the strengths and weaknesses of these two architectures, providing some insights for the advancement of robust vision systems.

4.1 Introduction

Convolutional Neural Networks (CNNs) have been the widely-used architecture for visual recognition in recent years (140; 224; 230; 111; 125). It is commonly believed the key to such success is the usage of the convolutional operation, as it introduces several useful inductive biases (*e.g.*, translation equivalence) to models for benefiting object recognition. Interestingly, recent works alternatively suggest that it is also possible to build successful recognition models without convolutions (198; 316; 21). The most representative work in this direction is Vision Transformer (ViT) (71), which applies the pure self-attention-based architecture to sequences of images patches and attains competitive performance on the challenging ImageNet classification task (209) compared to CNNs. Later works (161; 260) further expand Transformers with compelling performance on other visual benchmarks, including COCO detection and instance segmentation (151), ADE20K semantic segmentation (318).

The dominion of CNNs on visual recognition is further challenged by the recent findings that Transformers appear to be *much more robust* than CNNs. For example, Shao *et al.* (220) observe that the usage of convolutions may introduce a negative effect on models' adversarial robustness, while migrating to Transformer-like architectures (*e.g.*, the Conv-Transformer hybrid model or the pure Transformer) can help secure models' adversarial robustness. Similarly, Bhojanapalli *et al.* (22) report that, if pre-trained on sufficiently large datasets, Transformers exhibit considerably stronger robustness than CNNs on a spectrum of out-of-distribution tests (*e.g.*, common image corruptions (114), texture-shape cue conflicting stimuli (87)).

Though both (22) and (220) claim that Transformers are preferable to CNNs in terms of robustness, we find that such conclusion cannot be strongly drawn based on their existing experiments. Firstly, Transformers and CNNs are not compared at the same model scale, *e.g.*, a small CNN, ResNet-50 (25 million parameters), by default is compared to a much larger Transformer, ViT-B (86 million parameters), for these robustness evaluations. Secondly, the training frameworks applied to Transformers and CNNs are distinct from each other (*e.g.*, training datasets, number of epochs, and augmentation strategies are all different), while little efforts are devoted on ablating the corresponding effects. In a nutshell, due to these inconsistent and unfair experiment settings, *it remains an open question whether Transformers are truly more robust than CNNs*.

To answer it, in this paper, we aim to provide the first benchmark to fairly compare Transformers to CNNs in robustness evaluations. We particularly focus on the comparisons between Small Data-efficient image Transformer (DeiT-S) (241) and ResNet-50 (111), as they have similar model capacity (*i.e.*, 22 million parameters *vs*. 25 million parameters) and achieve similar performance on ImageNet (*i.e.*, 76.8% top-1 accuracy *vs*. 76.9% top-1 accuracy[1]). Our evaluation suite accesses model robustness in two ways: 1) adversarial robustness, where the attackers can actively and aggressively manipulate inputs to approximate the worst-case scenario; 2) generalization on out-of-distribution samples, including common image corruptions (ImageNet-C (114)), texture-shape cue conflicting stimuli (Stylized-ImageNet (87)) and natural adversarial examples (ImageNet-A (116)).

With this unified training setup, we present a completely different picture from previous ones (220; 22). Regarding adversarial robustness, we find that Transformers actually are no more robust than CNNs—if CNNs are allowed to properly adopt Transformers' training recipes, then these two types of models will attain similar robustness on defending against both perturbation-based adversarial attacks and patch-based adversarial attacks. While for generalization on out-of-distribution samples, we find Transformers can still substantially outperform CNNs even without the needs of pre-training on sufficiently large (external) datasets. Additionally, our ablations show that adopting Transformer's self-attention-like architecture is the key for achieving strong robustness on these out-of-distribution samples, while tuning other training setups will only yield subtle effects here. We hope this work can

[1]Here we apply the general setup in (243) for the ImageNet training. We follow the popular ResNet's standard to train both models for 100 epochs. Please refer to Section 4.3.1 for more training details.

serve as a useful benchmark for future explorations on robustness, using different network architectures, like CNNs, Transformers, and beyond (238; 154).

4.2 Related Works

Vision Transformer. Transformers, invented by Vaswani *et al.* in 2017 (253), have largely advanced the field of natural language processing (NLP). With the introduction of self-attention module, Transformer can effectively capture the non-local relationships between all input sequence elements, achieving the state-of-the-art performance on numerous NLP tasks (296; 61; 26; 67; 194; 195).

The success of Transformer on NLP also starts to get witnessed in computer vision. The pioneering work, ViT (71), demonstrates that the pure Transformer architectures are able to achieve exciting results on several visual benchmarks, especially when extremely large datasets (*e.g.*, JFT-300M (229)) are available for pre-training. This work is then subsequently improved by carefully curating the training pipeline and the distillation strategy to Transformers (241), enhancing the Transformers' tokenization module (302), building multi-resolution feature maps on Transformers (161; 260), designing parameter-efficient Transformers for scaling (305; 244; 290), *etc*. In this work, rather than focusing on furthering Transformers on standard visual benchmarks, we aim to provide a fair and comprehensive study of their performance when testing out of the box.

Robustness Evaluations. Conventional learning paradigm assumes training data and testing data are drawn from the same distribution. This assumption generally does not hold, especially in the real-world case where the underlying distribution is too complicated to be covered in a (limited-sized) dataset. To properly access model performance in the wild, a set of robustness generalization benchmarks have been built, *e.g.*, ImageNet-C (114), Stylized-ImageNet (87), ImageNet-A (116), *etc*. Another standard surrogate for testing model robustness is via adversarial attacks, where the attackers deliberately add small perturbations or patches to input images, for approximating the worst-case evaluation scenario (232; 95). In this work, both robustness generalization and adversarial robustness are considered in our robustness evaluation suite.

Concurrent to ours, both Bhojanapalli *et al.* (22) and Shao *et al.* (220) conduct robustness comparisons between Transformers and CNNs. Nonetheless, we find their experimental settings are unfair, *e.g.*, models are compared at different capacity (22; 220) or are trained under distinct frameworks (220). In this work, our comparison carefully align the model capacity and the training setups, which draws completely different conclusions from the previous ones.

4.3 Settings

4.3.1 Training CNNs and Transformers

Convolutional Neural Networks. ResNet (111) is a milestone architecture in the history of CNN. We choose its most popular instantiation, *ResNet-50* (with 25 million parameters), as the default CNN architecture. To train CNNs on ImageNet, we follow the standard recipe of (97; 197). Specifically, we train all CNNs for a total of 100 epochs, using momentum-SGD optimizer; we set the initial learning rate to 0.1, and decrease the learning rate by 10$\times$ at the 30-th, 60-th, and 90-th epoch; no regularization except weight decay is applied.

Vision Transformer. ViT (71) successfully introduces Transformers from natural language processing to computer vision, achieving excellent performance on several visual benchmarks compared to CNNs. In this paper, we follow the training recipe of DeiT (241), which successfully trains ViT on ImageNet without any external data, and set *DeiT-S* (with 22 million parameters) as the default Transformer architecture. Specifically, we train all Transformers using AdamW optimizer (164); we set the initial learning rate to 5e-4, and apply the cosine learning rate scheduler to decrease it; besides weight decay, we additionally adopt three data augmentation strategies (*i.e.*, RandAug (58), MixUp (307) and CutMix (303)) to regularize training (otherwise DeiT-S will attain significantly lower ImageNet accuracy due to overfitting (45)).

Note that different from the standard recipe of DeiT (which applies 300 training epochs by default), we hereby train Transformers only for a total of 100 epochs, *i.e.*, same as the setup in ResNet. We also remove {Erasing, Stochastic Depth, Repeated Augmentation}, which were applied in the original DeiT framework, in this basic 100 epoch schedule, for preventing over-regularization in training. Such trained DeiT-S yields 76.8% top-1 ImageNet accuracy, which is similar to the ResNet-50's performance (76.9% top-1 ImageNet accuracy).

4.3.2 Robustness Evaluations

Our experiments mainly consider two types of robustness here, *i.e.*, robustness on adversarial examples and robustness on out-of-distribution samples.

Adversarial Examples, which are crafted by adding human-imperceptible perturbations or small-sized patches to images, can lead deep neural networks to make wrong predictions.

In addition to the very popular PGD attack (167), our robustness evaluation suite also contains: A) AutoAttack (56), which is an ensemble of diverse attacks (*i.e.*, two variants of PGD attack, FAB attack (55) and Square Attack (5)) and is parameter-free; and B) Texture Patch Attack (TPA) (291), which uses a predefined texture dictionary of patches to fool deep neural networks.

Recently, several benchmarks of **out-of-distribution samples** have been proposed to evaluate how deep neural networks perform when testing out of the box. Particularly, our robustness evaluation suite contains three such benchmarks: A) ImageNet-A (116), which are real-world images but are collected from challenging recognition scenarios (*e.g.*, occlusion, fog scene); B) ImageNet-C (114), which is designed for measuring model robustness against 75 distinct common image corruptions; and C) Stylized-ImageNet (87), which creates texture-shape cue conflicting stimuli by removing local texture cues from images while retaining their global shape information.

4.4 Adversarial Robustness

In this section, we investigate the robustness of Transformers and CNNs on defending against adversarial attacks, using ImageNet validation set (with 50,000 images). We consider both perturbation-based attacks (*i.e.*, PGD and AutoAttack) and patch-based attacks (*i.e.*, TPA) for robustness evaluations.

4.4.1 Robustness to Perturbation-Based Attacks

Following (220), we first report the robustness of ResNet-50 and DeiT-S on defending against AutoAttack. We verify that, when applying with a small perturbation radius $\epsilon =$

0.001, DeiT-S indeed achieves higher robustness than ResNet-50, *i.e.*, 22.1% *vs*. 17.8% as shown in Table 4.1.

However, when increasing the perturbation radius to 4/255, a more challenging but standard case studied in previous works (217; 269; 278), *both models will be circumvented completely*, *i.e.*, 0% robustness on defending against AutoAttack. This is mainly due to that both models are not adversarially trained (95; 167), which is an effective way to secure model robustness against adversarial attacks, and we will study it next.

Table 4.1: Performance of ResNet-50 and DeiT-S on defending against AutoAttack, using ImageNet validation set. We note both models are completely broken when setting perturbation radius to 4/255.

	Clean	Perturbation Radius	
		0.001	4/255
ResNet-50	76.9	17.8	0.0
DeiT-S	76.8	22.1	0.0

4.4.1.1 Adversarial Training

Adversarial training (95; 167), which trains models with adversarial examples that are generated on-the-fly, aims to optimize the following min-max framework:

$$\arg\min_{\theta} \mathbb{E}_{(x,y)\sim\mathbb{D}}\left[\max_{\epsilon\in\mathbb{S}} L(\theta, x+\epsilon, y)\right], \tag{4.1}$$

where $\mathbb{D}$ is the underlying data distribution, $L(\cdot,\cdot,\cdot)$ is the loss function, θ is the network parameter, x is a training sample with the ground-truth label y, ϵ is the added adversarial perturbation, and $\mathbb{S}$ is the allowed perturbation range. Following (283; 269), the adversarial training here applies *single-step PGD* (PGD-1) to generate adversarial examples (for lowering training cost), with the constrain that maximum per-pixel change $\epsilon = 4/255$.

Adversarial Training on Transformers. We apply the setup above to adversarially train both ResNet-50 and DeiT-S. However, surprisingly, this default setup works for ResNet-50 but will collapse the training with DeiT-S, *i.e.*, the robustness of such trained DeiT-S is merely around 4% when evaluating against PGD-5. We identify the issue is over-regularization—when combining strong data augmentation strategies (*i.e.*, RangAug, Mixup and CutMix) with adversarial attacks, the yielded training samples are too hard to be learnt by DeiT-S.

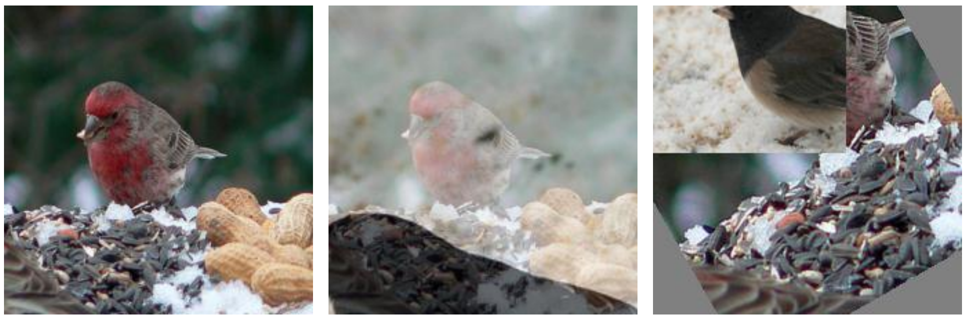

(a) Epoch = 0 (b) Epoch = 4 (c) Epoch = 9

Figure 4.1: The illustration of the proposed augmentation warm-up strategy. At the beginning of adversarial training (from epoch=0 to epoch=9), we progressively increase the augmentation strength.

To ease this observed training difficulty, we design a curriculum of the applied augmentation strategies. Specifically, as shown in Figure 4.1, at the first 10 epoch, we progressively enhance the augmentation strength (*e.g.*, gradually changing the distortion magnitudes in RandAug from 1 to 9) to warm-up the training process. Our experiment verifies this curriculum enables a successful adversarial training—DeiT-S now attains around 44% robustness (boosted from around 4%) on defending against PGD-5.

Transformers with CNNs' Training Recipes. Interestingly, an alternative way to address the observed training difficulty is directly adopting CNN's recipes to train Transformers (220), *i.e.*, applying M-SGD with step decay learning rate scheduler and removing strong data augmentation strategies (like Mixup). Though this setup can stabilize the adversarial training process, it significantly hurts the overall performance of DeiT-S—the clean accuracy drops to 59.9% (**-6.6%**), and the robustness on defending against PGD-100 drops to 31.9% (**-8.4%**).

One reason for this degenerated performance is that strong data augmentation strategies are not included in CNNs' recipes, therefore Transformers will be easily overfitted during training (45). Another key factor here is the incompatibility between the SGD optimizer and Transformers. As explained in (157), compared to SGD, adaptive optimizers (like AdamW) are capable of assigning different learning rates to different parameters, resulting in consistent update magnitudes even with unbalanced gradients. This property is crucial for enabling successful training of Transformers, given the gradients of attention modules are highly unbalanced.

CNNs with Transformers' Training Recipes. As shown in Table 4.2, adversarially trained ResNet-50 is less robust than adversarially trained DeiT-S, *i.e.*, 32.26% *vs*. 40.32% on defending against PGD-100. It motivates us to explore whether adopting Transformers' training recipes to CNNs can enhance CNNs' adversarial training. Interestingly, if we directly apply AdamW to ResNet-50, the adversarial training will collapses. We also explore the possibility of adversarially training ResNet-50 with strong data augmentation strategies (*i.e.*, RandAug, Mixup and CutMix). However, we find ResNet-50 will be overly regularized in adversarial training, leading to very unstable training process, sometimes may even collapse completely.

Though Transformers' optimizer and augmentation strategies cannot improve CNNs' adversarial training, we find *Transformers' choice of activation functions matters*. Unlike the widely-used activation function in CNNs is ReLU, Transformers by default use GELU (115). As suggested in (278), ReLU significantly weakens adversarial training due to its non-smooth nature; replacing ReLU with its smooth approximations (*e.g.*, GELU, SoftPlus) can strengthen adversarial training. We verify that by replacing ReLU with Transformers' activation function (*i.e.*, GELU) in ResNet-50. As shown in Table 4.2, adversarial training now can be significantly enhanced, *i.e.*, ResNet-50 + GELU substantially outperforms its ReLU counterpart by 8.01% on defending against PGD-100. Moreover, we note the usage of GELU enables ResNet-50 to match DeiT-S in adversarial robustness, *i.e.*, 40.27% *vs*. 40.32% for defending against PGD-100, and 35.51% *vs*. 35.50% for defending against AutoAttack, *challenging the previous conclusions (22; 220) that Transformers are more robust than CNNs on defending against adversarial attacks*.

Table 4.2: The performance of ResNet-50 and DeiT-S on defending against adversarial attacks (with $\epsilon = 4$). After replacing ReLU with DeiT's activation function GELU in ResNet-50, its robustness can match the robustness of DeiT-S.

	Activation	Clean Acc	PGD-5	PGD-10	PGD-50	PGD-100	AutoAttack
ResNet-50	ReLU	66.77	38.70	34.19	32.47	32.26	26.41
	GELU	67.38	44.01	40.98	40.28	40.27	35.51
DeiT-S	GELU	66.50	43.95	41.03	40.34	40.32	35.50

4.4.2 Robustness to Patch-Based Attacks

We next study the robustness of CNNs and Transformers on defending against patch-based attacks. We choose Texture Patch Attack (TPA) (291) as the attacker. Note that different from typical patch-based attacks which apply monochrome patches, TPA additionally optimizes the pattern of the patches to enhance attack strength. By default, we set the number of attacking patches to 4, limit the largest manipulated area to 10% of the whole image area, and set the attack mode as the non-targeted attack. For ResNet-50 and DeiT-S, we do not consider adversarial training here as their vanilla counterparts already demonstrate non-trivial performance on defending against TPA.

Interestingly, as shown in Table 4.3, though both models attain similar clean image accuracy, DeiT-S substantially outperforms ResNet-50 by 28% on defending against TPA. We conjecture such huge performance gap is originated from the differences in training setups; more specifically, it may be resulted by the fact DeiT-S by default use strong data augmentation strategies while ResNet-50 use none of them. The augmentation strategies like CutMix already naïvely introduce occlusion or image/patch mixing during training, therefore are potentially helpful for securing model robustness against patch-based adversarial attacks.

To verify the hypothesis above, we next ablate how strong augmentation strategies in DeiT-S (*i.e.*, RandAug, Mixup and CutMix) affect ResNet-50's robustness. We report the results in Table 4.4. Firstly, we note all augmentation strategies can help ResNet-50 achieve stronger TPA robustness, with improvements ranging from +4.6% to +32.7%. Among all these augmentation strategies, CutMix stands as the most effective one to secure model's TPA robustness, *i.e.*, CutMix alone can improve TPA robustness by 29.4%. Our best model is obtained by using both CutMix and RandAug, reporting 52.4% TPA robustness, which is even stronger than DeiT-S (47.7% TPA robustness). This observation still holds by using

Table 4.3: Performance of ResNet-50 and DeiT-S on defending against Texture Patch Attack.

Architecture	Clean Acc	Texture Patch Attack
ResNet-50	76.9	19.7
DeiT-S	76.8	**47.7**

stronger TPA with 10 patches (increased from 4), *i.e.*, ResNet-50 now attains 34.5% TPA robustness, outperforming DeiT-S by 5.6%. *These results suggest that Transformers are also no more robust than CNNs on defending against patch-based adversarial attacks.*

Table 4.4: Performance of ResNet-50 trained with different augmentation strategies on defending against Texture Patch Attack. We note 1) all augmentation strategies can improve model robustness, and 2) CutMix is the most effective augmentation strategy to secure model robustness.

Augmentations			Clean Acc	Texture Patch Attack
RandAug	MixUp	CutMix		
			76.9	19.7
✓			77.5	24.3 (+4.6)
	✓		75.9	31.5 (+11.8)
		✓	77.2	49.1 (+29.4)
✓	✓		75.7	31.7 (+12.0)
✓		✓	76.7	**52.4 (+32.7)**
	✓	✓	77.1	39.8 (+20.1)
✓	✓	✓	76.4	48.6 (+28.9)

4.5 Robustness on Out-of-distribution Samples

In addition to adversarial robustness, we are also interested in comparing the robustness of CNNs and Transformers on out-of-distribution samples. We hereby select three datasets, *i.e.*, ImageNet-A, ImageNet-C and Stylized ImageNet, to capture the different aspects of out-of-distribution robustness.

4.5.1 Aligning Training Recipes

We first provide a direct comparison between ResNet-50 and DeiT-S with their default training setup. As shown in Table 4.5, we observe that, *even without pretraining on (external) large scale datasets*, DeiT-S still significantly outperforms ResNet-50 on ImageNet-A (+9.0%), ImageNet-C (+9.9) and Stylized-ImageNet (+4.7%). It is possible that such performance gap is caused by the differences in training recipes (similar to the situation we observed in Section 4.4), which we plan to ablate next.

Table 4.5: DeiT-S shows stronger robustness generalization than ResNet-50 on ImageNet-C, ImageNet-A and Stylized-ImageNet. Note the results on ImageNet-C is measured by mCE (lower is better).

Architecture	ImageNet ↑	ImageNet-A ↑	ImageNet-C ↓	Stylized-ImageNet ↑
ResNet-50	76.9	3.2	57.9	8.3
ResNet-50*	76.3	4.5	55.6	8.2
DeiT-S	76.8	**12.2**	**48.0**	**13.0**

A fully aligned version. A simple baseline here is that we completely adopt the recipes of DeiT-S to train ResNet-50, denoted as ResNet-50*. Specifically, this ResNet-50* will be trained with AdamW optimizer, cosine learning rate scheduler and strong data augmentation strategies. Nonetheless, as reported in Table 4.5, ResNet-50* only marginally improves ResNet-50 on ImageNet-A (+1.3%) and ImageNet-C (+2.3), which is still much worse than DeiT-S on robustness generalization.

It is possible that completely adopting the recipes of DeiT-S overly regularizes the training of ResNet-50, leading to suboptimal performance. To this end, we next seek to discover the "best" setups to train ResNet-50, by ablating learning rate scheduler (step decay *vs*. cosine decay), optimizer (M-SGD *vs*. AdamW) and augmentation strategies (RandAug, Mixup and CutMix) progressively.

Step 1: aligning learning rate scheduler. It is known that switching learning rate scheduler from step decay to cosine decay improves model accuracy on clean images (20). We additionally verify that such trained ResNet-50 (second row in Table 4.6) attains slightly better performance on ImageNet-A (+0.1%), ImageNet-C (+1.0) and Stylized-ImageNet (+0.1%). Given the improvements here, we will use cosine decay by default for later ResNet training.

Step 2: aligning optimizer. We next ablate the effects of optimizers. As shown in the third row in Table 4.6, switching optimizer from M-SGD to AdamW weakens ResNet training, *i.e.*, it not only decreases ResNet-50's accuracy on ImageNet (-1.0%), but also hurts ResNet-50's robustness generalization on ImageNet-A (-0.2%), ImageNet-C (-2.4) and Stylized-ImageNet (-0.3%). Given this degenerated performance, we stick to M-SGD for later ResNet-training.

Table 4.6: The robustness generalization of ResNet-50 trained with different learning rate schedulers and optimizers. Nonetheless, compared to DeiT-S, all the resulted ResNet-50 show worse generalization on out-of-distribution samples.

	Optimizer-LR Scheduler	ImageNet ↑	ImageNet-A ↑	ImageNet-C ↓	Stylized-ImageNet ↑
ResNet-50	SGD-Step	76.9	3.2	57.9	8.3
	SGD-Cosine	77.4	3.3	56.9	8.4
	AdamW-Cosine	76.4	3.1	59.3	8.1
DeiT-S	AdamW-Cosine	76.8	**12.2**	**48.0**	**13.0**

Step 3: aligning augmentation strategies. Compared to ResNet-50, DeiT-S additionally applied RandAug, Mixup and CutMix to augment training data. We hereby examine whether these augmentation strategies affect robustness generalization. The performance of ResNet-50 trained with different combinations of augmentation strategies is reported in Table 4.7. Compared to the vanilla counterpart, nearly all the combinations of augmentation strategies can improve ResNet-50's generalization on out-of-distribution samples. The best performance is achieved by using RandAug + Mixup, outperforming the vanilla ResNet-50 by 3.0% on ImageNet-A, 4.6 on ImageNet-C and 2.4% on Stylized-ImageNet.

Comparing ResNet with the "best" training recipes to DeiT-S. With the ablations above,

Table 4.7: The robustness generalization of ResNet-50 trained with different combinations of augmentation strategies. We note applying RandAug + Mixup yields the best ResNet-50 on out-of-distribution samples; nonetheless, DeiT-S still significantly outperforms such trained ResNet-50.

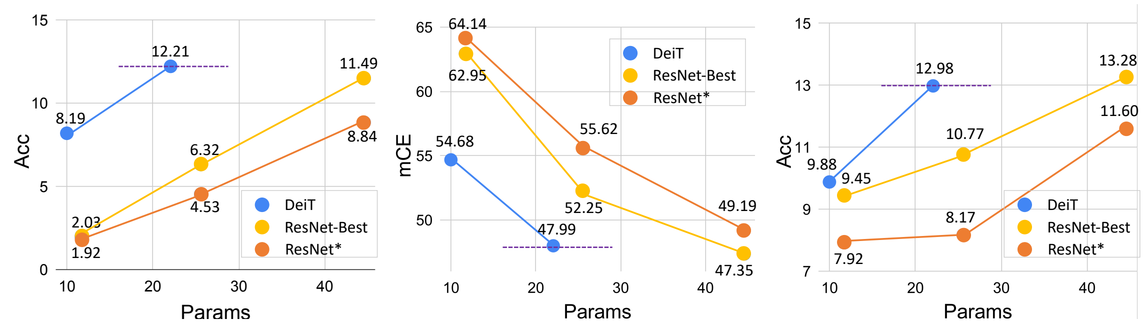
Architecture	Augmentation Strategies			ImageNet ↑	ImageNet-A ↑	ImageNet-C ↓	Stylized-ImageNet↑
	RandAug	MixUp	CutMix				
ResNet-50				77.4	3.3	56.9	8.4
	✓	✓		75.7	**6.3**	**52.3**	**10.8**
	✓		✓	76.7	6.3	56.3	7.1
		✓	✓	77.1	6.1	55.1	8.8
	✓	✓	✓	76.4	5.5	54.0	9.1
DeiT-S	✓	✓	✓	**76.8**	**12.2**	**48.0**	**13.0**

we can conclude that the "best" training recipes for ResNet-50 (denoted as ResNet-50-Best) is by applying M-SGD optimizer, scheduling learning rate using cosine decay, and augmenting training data using RandAug and Mixup. As shown in the second row of Table 4.7, ResNet-50-Best attains 6.3% accuracy on ImageNet-A, 52.3 mCE on ImageNet-C and 10.8% accuracy on Stylized-ImageNet.

Nonetheless, interestingly, we note DeiT-S still shows much stronger robustness generalization on out-of-distribution samples than our "best" ResNet-50, *i.e.*, +5.9% on ImageNet-A, +4.3 on ImageNet-C and +2.2% on Stylized-ImageNet. *These results suggest that the differences in training recipes (including the choice of optimizer, learning rate scheduler and augmentation strategies) is not the key for leading the observed huge performance gap between CNNs and Transformers on out-of-distribution samples.*

Model size. To further validate that Transformers are indeed more robust than CNNs on out-of-distribution samples, we hereby extend the comparisons above to other model sizes. Specifically, we consider the comparison at a smaller scale, *i.e.* ResNet-18 (12 million parameters) *vs*. DeiT-Mini (10 million parameters, with embedding dimension = 256 and number of head = 4). For ResNet training, we consider both the fully aligned recipe version (denoted as ResNet*) and the “best” recipe version (denoted as ResNet-Best). Figure 4.2 shows the main results. Similar to the comparison between ResNet-50 and DeiT-S, DeiT-Mini also demonstrates much stronger robustness generalization than ResNet-18* and ResNet-18-Best.

We next study DeiT and ResNet at a more challenging setting—comparing DeiT to a much larger ResNet on robustness generalization. Surprisingly, we note in both cases, DeiT-Mini *vs*. ResNet-50 and DeiT-S *vs*. ResNet-101, DeiTs are able to show similar, sometimes even superior, performance than ResNets. For example, DeiT-S beats the nearly 2× larger ResNet-101* (22 million parameters *vs*. 45 million parameters) by 3.37% on ImageNet-A, 1.20 on ImageNet-C and 1.38% on Stylized-ImageNet. All these results further corroborate that Transformers are much more robust than CNNs on out-of-distribution samples.



(a) ImageNet-A (b) ImageNet-C (c) Stylized-ImageNet

Figure 4.2: By comparing models at different scales, DeiT consistently outperforms ResNet* and ResNet-Best by a large margin on ImageNet-A, ImageNet-C and Stylized-ImageNet.

4.5.2 Distillation

In this section, we make another attempt to bridge the robustness generalization gap between CNNs and Transformers—we apply knowledge distillation to let ResNet-50 (student model) directly learn from DeiT-S (teacher model). Specifically, we perform soft distillation (120), which minimizes the Kullback-Leibler divergence between the softmax of the teacher model and the softmax of the student model; we adopt the training recipe of DeiT during distillation.

Main results. We report the distillation results in Table 4.8. Though both models attain similar clean image accuracy, the student model ResNet-50 shows much worse robustness generalization than the teacher model DeiT-S, *i.e.*, the performance is decreased by 7.0% on ImageNet-A, 6.2 on ImageNet-C and 3.2% on Stylized-ImageNet. This observation is counter-intuitive as student models typically achieve higher performance than teacher models in knowledge distillation.

However, interestingly, if we switch the roles of DeiT-S and ResNet-50, the student model DeiT-S is able to significantly outperforms the teacher model ResNet-50 on out-of-distribution samples. As shown in the third row and the fourth row in Table 4.8, the improvements are 6.4% on ImageNet-A, 6.3 on ImageNet-C and 3.7% on Stylized-ImageNet. *These results arguably suggest that the strong generalization robustness of DeiT is rooted in the architecture design of Transformer that cannot be transferred to ResNet via neither training setups or knowledge distillation.*

Table 4.8: The robustness generalization of ResNet-50, DeiT-S and their distilled models.

Distillation	Architecture	ImageNet ↑	ImageNet-A ↑	ImageNet-C ↓	Stylized-ImageNet ↑
Teacher	DeiT-S	76.8	12.2	48.0	13.0
Student	ResNet-50*-Distill	76.7	5.2 (-7.0)	54.2 (+6.2)	9.8 (-3.2)
Teacher	ResNet-50*	76.3	4.5	55.6	8.2
Student	DeiT-S-Distill	76.2	10.9 (+6.4)	49.3 (-6.3)	11.9 (+3.7)

4.5.3 Hybrid Architecture

Following the discussion in Section 4.5.2, we hereby ablate whether incorporating Transformer's self-attention-like architecture into model design can help robustness generalization. Specifically, we create a hybrid architecture (named Hybrid-DeiT) by directly feeding the output of res_4 block in ResNet-18 into DeiT-Mini, and compare its robustness generalization to ResNet-50 and DeiT-Small. Note that under this setting, these three models are at the same scale, *i.e.*, hybrid-DeiT (21 million parameters) *vs*. ResNet-50 (25 million parameters) *vs*. DeiT-S (22 million parameters). We apply the recipe of DeiT to train these three models.

Main results. We report the robustness generalization of these three models in Figure 4.3. Interestingly, with the introduction of Transformer blocks, Hybrid-DeiT is able to achieve better robustness generalization than ResNet-50, *i.e.*, +1.1% on ImageNet-A and +2.5% on Stylized-ImageNet, *suggesting Transformer's self-attention-like architectures is essential for boosting performance on out-of-distribution samples*. We additionally compare this hybrid architecture to the pure Transformer architecture. As expected, Hybrid-DeiT attains lower robustness generalization than DeiT-S, as shown in Figure 4.3.

4.5.4 300-Epoch Training

As mentioned in Section 4.3.1, we by default train all models for only 100 epochs. This is a standard setup in training CNNs (97; 197), but not typical in training Transformers (243; 161). To rule out the possibility of introducing negative effects in shortening training length, we lastly ablate the 300-epoch setup, *i.e.*, we directly borrow the default setup in (243) to train both ResNet and DeiT.

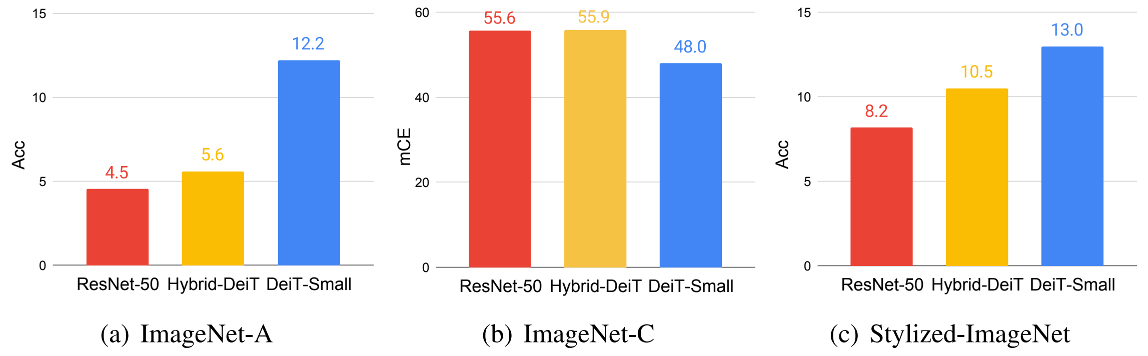


Figure 4.3: The robustness generalization of ResNet-50, DeiT-S and Hybrid-DeiT. We note introducing Transformer blocks into model design benefits generalization on out-of-distribution samples.

As reported in Table 4.9, DeiT-S substantially outperforms ResNet-50 by 10.4% on ImageNet-A, 7.5 on ImageNet-C and 5.6 on Stylized-ImageNet. Nonetheless, we argue that such comparison is less interesting and even unfair—DeiT-S already beats ResNet-50 by 1.8% on ImageNet classification, therefore it is expected that DeiT-S will also show stronger performance than ResNet-50 on ImageNet-A, ImageNet-C and Stylized-ImageNet.

Table 4.9: The robustness generalization of ResNet-50 and DeiT-S under the 300-epoch training setup. We note DeiT-S shows stronger performance than ResNet-50 on both clean images and out-of-distribution samples.

Architecture	ImageNet ↑	ImageNet-A ↑	ImageNet-C ↓	Stylized-ImageNet ↑
ResNet-50	78.1	8.8	50.3	9.5
DeiT-S	79.9	**19.2**	**42.8**	**15.1**

To make the setup fairer (*i.e.*, comparing the robustness of models that have similar accuracy), we now compare DeiT-S to the much larger ResNet-101 (*i.e.*, 22 million parameters *vs*. 45 million parameters). The results are shown in Table 4.10. We observe that though both models achieve similar accuracy on ImageNet, DeiT-S demonstrates much stronger robustness generalization than ResNet-101. This observation can also holds for bigger Transformers and CNNs, *e.g.*, DeiT-B can consistently outperforms ResNet-200 on

ImageNet-A, ImageNet-C and Stylized- ImageNet, despite they attain similar clean image accuracy (*i.e.*, 81.8% *vs*. 82.1%).

Table 4.10: The robustness generalization of ResNet and DeiT under the 300-epoch training setup. Though both models attain similar clean image accuracy, DeiTs show much stronger robustness generalization than ResNets.

Architecture	ImageNet ↑	ImageNet-A ↑	ImageNet-C ↓	Stylized-ImageNet ↑
ResNet-101	80.2	17.6	45.8	11.9
DeiT-S	79.9	**19.2**	**42.8**	**15.1**
ResNet-200	82.1	23.8	40.8	13.6
DeiT-B	81.8	**27.9**	**38.0**	**17.9**

In summary, in this 300-epoch training setup, we can draw the same conclusion as the one in the 100-epoch training setup, *i.e.*, *Transformers are truly much more robust than CNNs on out-of-distribution samples*. In addition, we note this conclusion is further corroborated in concurrent works (306; 188; 281; 322; 179), where a range of additional out-of-distribution tasks/datasets are tested. We refer interested readers to their papers for details.

4.6 Conclusion

With the recent success of Transformer in visual recognition, researchers begin to study its robustness compared with CNNs. While recent works suggest that Transformers are much more robust than CNNs, their comparisons are not fair in many aspects, *e.g.*, training datasets, model scales, training strategies, *etc*. This motivates us to provide a fair and in-depth comparisons between CNNs and Transformers, focusing on adversarial robustness and robustness on out-of-distribution samples. With our unified training setup, we found that Transformers are no more robust than CNNs on adversarial robustness. By properly adopting Transformer's training recipes, CNNs can achieve similar robustness as Transformers on defending against both perturbation-based adversarial attacks and patch-based adversarial attacks. While regarding generalization on out-of-distribution samples (*e.g.*, ImageNet-A, ImageNet-C and Stylized ImageNet), we find Transformer's self-attention-like architectures is the key. We hope this work would shed lights on the understanding of Transformer, and help the community to fairly compare robustness between Transformers and CNNs.

Chapter 5

Point-Level Region Contrast for Object Detection Pre-Training

Continuing our exploration of techniques aimed at enhancing generalization capabilities, we delve into the realm of pre-training approaches for object detection. Through the introduction of point-level region contrast, our research focuses on empowering vision models with generalizable representation learning capabilities across different downstream tasks.

5.1 Introduction

Un-/self-supervised learning – in particular contrastive learning (113; 107; 40) – has recently arisen as a powerful tool to obtain visual representations that can potentially benefit from an unlimited amount of *unlabeled* data. Promising signals are observed on important tasks like object detection (152). For example, MoCo (107) shows convincing improvement on VOC (79) over supervised pre-training by simply learning to discriminate between images as holistic instances (74) on the ImageNet-1K dataset (210). Since then, numerous

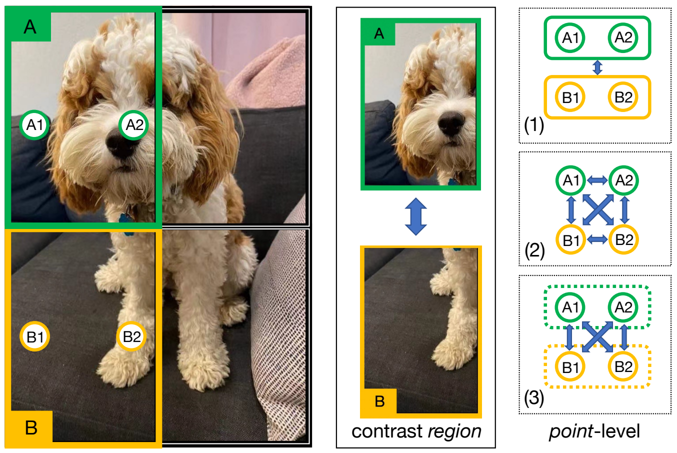


Figure 5.1: For intra-image contrastive learning, samples of a feature map can be aggregated and then compared between regions (1), compared directly between all samples (2), or only compared directly between samples in different regions (3). We call (3) **point-level region contrast**, it allows both learning at the point-level to help localization, and at the region-level to help holistic object recognition – two crucial aspects for object detection.

pre-text tasks that focus on *intra-image* contrast have been devised specifically for object detection as the downstream transfer task (262; 286; 112). While there has been steady progress, state-of-the-art detectors (2) still use weights from supervised pre-training (*e.g.*, classification on ImageNet-22K (66)). The full potential of unsupervised pre-training for object detection is yet to be realized.

Object detection requires both accurate *localization* of objects in an image and correct *recognition* of their semantic categories. These two sub-tasks are tightly connected and often reinforce each other in successful detectors (168). For example, region proposal methods (249; 325; 6) that first narrow down candidate object locations have enabled R-CNN (91) to perform classification on rich, *region-level* features. Conversely, today's dominant paradigm for object instance segmentation (109) first identifies object categories along with their coarse bounding boxes, and later uses them to compute masks for better localization at the *pixel-level*.

With this perspective, we hypothesize that to learn a useful representation for object detection, it is also desirable to balance recognition and localization by leveraging information at various levels during pre-training. Object recognition in a scene typically takes place at the region-level (91; 205). To support this, it is preferable to maintain a conceptually coherent 'label' for each region, and learn to contrast pairs of regions for representation learning. On the other hand, for better localization, the model is preferred to operate at the pixel-, or *'point-level'* (137; 49), especially when an initial, unsupervised, assignment of pixels to regions (*i.e.*, segmentation) is sub-optimal (see Fig. 5.1 for an example). To our knowledge, existing methods in this frontier can be lacking in either of these two aspects (to be discussed in Sec. 7.2).

In this paper, we present a self-supervised pre-training approach that conceptually contrasts at the region-level while operating at the point-level. Starting from MoCo v2 (42) as an image-level baseline, we introduce the notion of 'regions' by dividing each image into a non-overlapping grid (112). Treating rectangular regions on this grid as separate instances, we can define the task of intra-image discrimination on top of the existing inter-image one (74) and pre-train a representation with contrastive objectives. Deviating from the common practice that aggregates features for contrastive learning (107; 40; 112), we directly operate at the point-level by sampling multiple points from each region, and contrasting point pairs individually across regions (see Fig. 5.1, right column for illustrations).

The advantage of operating at the point-level is two-fold, both concerning dealing with *imperfect* regions as there is no ground-truth. First, such a design can be more *robust* to the change in region quality, since feature aggregation can cause ambiguities when the regions are not well localized (*e.g.*, in Fig. 5.1, both regions of interest can mean ‘a mixture of dog and couch’), whereas individual points still allow the model to see distinctions. Second and perhaps more importantly, it can enable us to *bootstrap* (100) for potentially better regions during the training process. This is because any segmentation can be viewed as a hard-coded form of *point affinities* – 1 for point pairs within the same region and 0 otherwise; and a natural by-product of contrasting point pairs is soft point affinities (values between 0 and 1) that *implicitly* encode regions. By viewing the momentum encoder as a ‘teacher’ network, we can formulate the problem as knowledge distillation one (120; 30), and improving point affinities (and thus implicitly regions) online in the same self-supervised fashion.

Empirically, we applied our approach to standard pre-training datasets (ImageNet-1K (66) and COCO train set (152)), and transferred the representation to multiple downstream datasets: VOC (79), COCO (for both object detection and instance segmentation), and Cityscapes (54) (semantic segmentation). We show strong results compared to state-of-the-art pre-training methods which use image-level, point-level, or region-level contrastive learning. Moreover, we provide extensive ablation studies covering different aspects in design, and qualitatively visualize the point affinities learned through knowledge distillation.

While we are yet to showcase improvements on larger models, longer training schedules, stronger augmentations (88), and bigger pre-training data for object detection, we believe our explorations on the pre-training design that better balances recognition and localization can inspire more works in this direction.

5.2 Related Work

Self-supervised learning. Supervised learning/classification (110; 210) has been the dominant method for pre-training representations useful for downstream tasks in computer vision. Recently, contrastive learning (272; 113; 235; 107; 40; 75) has emerged as a promising alternative that pre-trains visual representations *without* class labels or other forms of human annotations – a paradigm commonly referred as 'self-supervised learning'. By definition, self-supervised learning holds the potential of scaling up pre-training to huge models and billion-scale data. As a demonstration, revolutionary progress has already been made in fields like natural language processing (67; 196; 24) through scaling. For computer vision, such a moment is yet to happen. Nonetheless, object detection as a fundamental task in computer vision is a must-have benchmark to test the transferability of pre-trained representations (91).

Contrastive learning. Akin to supervised learning which maps images to class labels, contrastive learning maps images to separate vector embeddings, and attracts positive embedding pairs while dispels negative pairs. A key concept connecting the two types of learning is instance discrimination (74), which models each image as its *own* class. Under this formulation, two augmentations of the same image is considered as a positive pair, while different images form negative pairs. Interestingly, recent works show that negative pairs are not required to learn meaningful representations (100; 44) for reasons are yet to be understood. Regardless, all these frameworks treat each image as a single instance and use aggregated (*i.e.*, pooled) features to compute embeddings. Such a classification-oriented design largely ignores the internal structures of images, which could limit their application to object detection that performs dense search *within* an image (205; 160; 150).

Point-level contrast. Many recent works (262; 286; 159; 284; 191) have realized the above limitation, and extended the original idea from contrasting features between whole images to contrasting features at points. Different ways to match points as pairs have been explored. For example, (262) selects positive pairs by ranking similarities among all points in the latent space; (286) defines positive pairs by spatial proximity; (159) jointly matches a set of features at points to another set via Sinkhorn-Knopp algorithm (59), designed to maximize the set-level similarity for sampled features. However, we believe directly contrasting features at arbitrary points over-weights localization, and as a result misses a more global view of the entire object that can lead to better *recognition*.

Region-level contrast. Closest to our paper is the most recent line of work that contrasts representations at the region-level (112; 280; 266; 282; 277; 207). Specifically, images are divided into regions of interest, via either external input (112; 266; 282), or sliding windows (277), or just random sampling (280; 207). Influenced by image-level contrastive learning, most approaches represent each region with a single, aggregated vector embedding for loss computation and other operations, which we argue – and show empirically – is detrimental for *localization* of objects.

5.3 Approach

In this section we detail our approach: point-level region contrast. To lay the background and introduce notations, we begin by reviewing the formulation of MoCo (107).

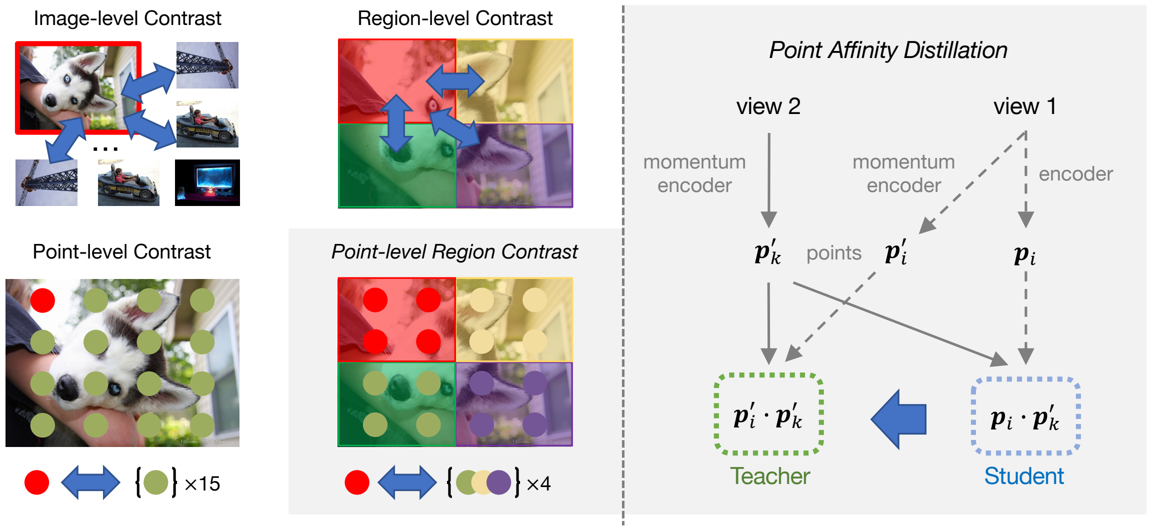


Figure 5.2: Illustration of **point-level region contrast** (Sec. 5.3.2), which also enables **point affinity distillation** (Sec. 5.3.3). On the left we show four different types of contrastive learning methods, including image-level, region-level, point-level and our point-level region contrast. On the right we show point affinity distillation with one pair of points.

5.3.1 Background: Momentum Contrast

As the name indicates, MoCo (107; 42) is a contrastive learning framework (185; 40) that effectively uses momentum encoders to learn representations. Treating each image as a single *instance* to discriminate against others, MoCo operates at the image-level (see Fig. 6.1 top left corner).

Image-level contrast. While the original model for instance discrimination (74) literally keeps a dedicated weight vector for each image in the dataset (on ImageNet-1K (210) it would mean more than one million vectors), modern frameworks (272; 40) formulate this task as a contrastive learning one which only requires online computation of embedding vectors per-image and saves memory. Specifically MoCo, two parallel encoders, f^E and

f^M, take two augmented views (and $'$) for each image in a batch, and output two ℓ_2-normalized embeddings and $'$. Here f^E denotes the base encoder being trained by gradient updates as in normal supervised learning, and f^M denotes the momentum encoder that keeps updated by exponential moving average on the base encoder weights. Then image-level contrastive learning is performed by enforcing similarity on views from the same image, and dissimilarity on views from different images, with the commonly used InfoNCE objective (185):

$$\mathcal{L}_{\mathrm{m}} = -\log \frac{\exp(\cdot'/\tau)}{\sum_j \exp(\cdot'_j/\tau)}, \tag{5.1}$$

Where τ is the temperature, other images (and self) are indexed by j. In MoCo, other images are from the momentum bank (272), which is typically much smaller in size compared to the full dataset.

It is important to note that in order to compute the embedding vectors (and $'$), a pooling-like operation is often used in intermediate layers to aggregate information from all spatial locations in the 2D image. This is inherited from the practice in supervised learning, where standard backbones (*e.g.*, ResNet-50 (110)) average-pool features before the classification task.

5.3.2 Point-Level Region Contrast

As discussed above, image-level contrast is classification oriented. Next, we discuss our designs in point-level region contrast, which are more fit for the tasks of object detection.

Regions. Region is a key concept in state-of-the-art object detectors (205; 109). Through region-of-interest pooling, object-level recognition (*i.e.*, classifying objects into pre-defined categories) are driven by region-level features. Different from detector training, ground-truth object annotations are not accessible in self-supervised learning. Therefore, we simply introduce the notion of regions by dividing each image into a non-overlapping, $n \times n$ grid (112). We treat the rectangular regions on this grid as separate instances, which allows inter-image contrast and *intra-image* contrast to be jointly performed on pairs of regions. Now, each augmentation is paired with masks, and each mask denotes the corresponding region under the same geometric transformation as with which it shares resolution. Note that due to randomly resized cropping (107), some masks can be empty. Therefore, we randomly sample N=16 valid masks $\{_n\}$ ($n \in \{1, \ldots, N\}$) (with repetition) as regions to contrast, following the design of (112).

Grid regions are the simplest form of the spatial heuristic that nearby pixels are likely belong to the same object (112). More advanced regions (249; 6), or even ground-truth segmentation masks (used for analysis-only) (152) can be readily plugged in our method to potentially help performance, but it comes at the expense of more computation costs, potential risk of bias (34) or human annotation costs. Instead, we focus on improving training strategies and just use grids for our explorations.

Point-level. Given the imperfect regions, our key insight is to operate at the point-level. Intuitively, pre-training by contrasting regions can help learn features that are discriminative enough to tell objects apart as *holistic* entities, but they can be lacking in providing low-level cues for the exact *locations* of objects. This is particularly true if features that represent regions are aggregated over all pertinent locations, just like the practice in image-level contrast. Deviating from this, we directly sample multiple points from each region, and contrast point pairs individually across regions *without* pooling.

Formally, we sample P points per mask $_n$, and compute point-level features $_i$ ($i\in\{1,\ldots,N\times P\}$) for contrastive learning. Each $_i$ comes with an indicator for its corresponding region, $_i$. To accommodate this, we modify the encoder architecture so that the *spatial* dimensions are kept all the way till the output.[1] The final feature map is up-sampled to a spatial resolution of $R\times R$ via interpolation. Then our point-level, region contrastive loss is defined as:

$$\mathcal{L}_{\mathrm{c}} = -\frac{1}{C}\sum_{i=k}\log\frac{\exp({}_i\cdot{}'_k/\tau)}{\sum_j \exp({}_i\cdot{}'_j/\tau)}, \tag{5.2}$$

where j loops over points from regions in the same image (intra-), or over points from other images (inter-). C is a normalization factor for the loss which depends on the number of positive point pairs. An illustrative case (for n=2 and P=4) is shown in Fig. 6.1.

5.3.3 Point Affinity Distillation

Operating at the point level enables us to bootstrap (311) and not be restricted by the pre-defined regions. This is because according to Eq. (5.2), the only place the pre-defined regions matter is in the indicators $_i$, which provides a *hard* assignment from points to regions. When $_i{=}_k$, it means the probability of $_i$ and $_k$ coming from the same region is 1, otherwise it is 0.

On the other hand, the InfoNCE loss (185) (Eq. (5.1)) used for contrastive learning computes *point affinities* as a natural by-product which we define as:

$$_{ik'}(\tau) := \frac{\exp({}_i\cdot{}'_k/\tau)}{\sum_j \exp({}_i\cdot{}'_j/\tau)}. \tag{5.3}$$

Note that $_{ik'}(\tau)$ is a pairwise term controlled by two indexes i and k', and the additional $'$ indicates the embeddings participating is computed by the momentum encoder. For

[1]An additional projector MLP is introduced in MoCo v2 (42) following SimCLR (40), we convert the MLP into 1×1 convolution layers.

example ${}_{i'k'}(\tau)$ means both embeddings are from the momentum encoder f^M. Point affinities offer *soft*, implicit assignment from points to regions, and an explicit assignment can be obtained via clustering (*e.g.* k-means). In this sense, they arguably provide more *complete* information about which point pairs belong to the same region.

The Siamese architecture (44) of self-supervised learning methods like MoCo presents a straightforward way to bootstrap and obtain potentially better regions. The momentum encoder f^M itself can be viewed as a 'teacher' which serves as a judge for the quality of f^E (30). From such an angle, we can formulate the problem as a knowledge distillation one (120), and use the outputs of f^M to supervise the point affinities that involve f^E via cross entropy loss:

$$\mathcal{L}_{\mathrm{a}} = -\sum_{i,k} {}_{i'k'}(\tau_t) \log_{ik'}(\tau_s), \tag{5.4}$$

where τ_t and τ_s are temperatures for the teacher and the student, respectively. We call this 'point affinity distillation'. There are other possible ways to distill point affinities from the momentum encoder (see Sec. 5.4.5.2), we choose the current design trading off speed and accuracy.

On the other hand, we note that the pooling operation does *not* back-propagate gradients to the coordinates (only to the features) by default. Therefore, it is less straightforward to morph regions along with training by contrasting aggregated region-level features (112; 266; 282).

method	# of epochs	Pascal VOC			COCO detection			COCO segmentation			Cityscapes
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}	mIoU
Scratch	-	33.8	60.2	33.1	26.4	44.0	27.8	29.3	46.9	30.8	65.3
Supervised	200	54.2	81.6	59.8	38.2	58.2	41.2	33.3	54.7	35.2	73.7
MoCo (107)	200	55.9	81.5	62.6	38.5	58.3	41.6	33.6	54.8	35.6	75.3
SimCLR (40)	1000	56.3	81.9	62.5	38.4	58.3	41.6	-	-	-	75.8
MoCo v2 (42)	800	57.6	82.7	64.4	39.8	59.8	43.6	36.1	56.9	38.7	76.2
InfoMin (237)	200	57.6	82.7	64.6	39.0	58.5	42.0	-	-	-	75.6
DetCo (280)	200	57.8	82.6	64.2	39.8	59.7	43.0	34.7	56.3	36.7	76.5
InsLoc (292)	800	58.4	83.0	65.3	39.8	59.6	42.9	34.7	56.3	36.9	-
PixPro (286)	200	58.8	83.0	66.5	40.0	59.3	43.4	34.8	-	-	76.8
DetCon (112)	200	-	-	-	40.5	-	-	36.4	-	-	76.5
SoCo (266)	200	59.1	83.4	65.6	40.4	**60.4**	43.7	34.9	56.8	37.0	76.5
Ours	200	**59.4**	**83.6**	**67.1**	**40.7**	**60.4**	**44.7**	**36.9**	**57.4**	**39.6**	**77.0**

Table 5.1: Main results with ImageNet-1K pre-training. From left to right, we show transfer performance on 4 tasks: VOC (07+12) detection (79), COCO object detection (152); COCO instance segmentation and Cityscapes semantic segmentation (54). From top to down, we compare our approach with 3 other setups: i) no pre-training (*i.e.*, scratch); ii) general pre-training with supervised learning or *inter-image* contrastive learning; iii) object detection oriented pre-training with additional *intra-image* contrast. Our point-level region contrast pre-training shows consistent improvements across different tasks under fair comparisons.

5.3.4 Overall Loss Function

We jointly perform point-level region contrast learning (Sec. 5.3.2) and point affinity distillation (Sec. 5.3.3) controlled by a balance factor α:

$$\mathcal{L}_{\mathrm{p}} = \alpha\mathcal{L}_{\mathrm{c}} + (1-\alpha)\mathcal{L}_{\mathrm{a}}. \tag{5.5}$$

Here, $\mathcal{L}_{\mathrm{c}}$ offers an initialization of regions to contrast with, whereas $\mathcal{L}_{\mathrm{a}}$ bootstraps (100) from data, regularizes learning and alleviates over-fitting to the initial imperfect region assignments. This is how these two terms interact and benefit each other – a common practice for knowledge distillation with additional ground-truth labels.

Finally, our point-level loss is added to the original MoCo loss for joint optimization, controlled by another factor β:

$$\mathcal{L} = \beta\mathcal{L}_{\mathrm{p}} + (1 - \beta)\mathcal{L}_{\mathrm{m}}, \tag{5.6}$$

which does not incur extra overhead for backbone feature computation. Note that all the loss terms we have defined above are focused on a single image for explanation clarity, the full loss is averaged over all images.

5.4 Experiments

In this section we perform experiments. For our main results, we pre-train on ImageNet-1K or COCO, and transfer the learned representations to 4 downstream tasks. We then conduct analysis by: 1) visualizing the learned point affinities with a quantitative evaluation metric using VOC ground-truth masks, 2) presenting evidence that point-level representations are effective and more robust to region-level ones when the mask quality degenerates; and 3) ablating different point affinity distillation strategies in our approach. More analysis on various hyper-parameters and more visualizations are found in the appendix.

5.4.1 Pre-Training Details

We either pre-train on ImageNet-1K (210) or COCO (152), following standard setups (112; 262).

ImageNet-1K setting. Only images from the training split are used, which leads to 1.28 million images for ImageNet-1K. We pre-train the model for 200 epochs.

It is worth noting that we build our approach on the default, *asymmetric* version of MoCo v2 (42), which is shown to roughly compensate for the performance of pre-training with *half* the length using *symmetrized* loss (44) – both setups share the same amount of compute in this case.

COCO setting. Only images from the training split (`train2017`) are used, which leads to 118k for COCO. We pre-train with 800 *COCO* epochs, not ImageNet epochs.

Hyper-parameters and augmentations. We use a 4×4 grid and sample N=16 valid masks per view following (112). P=16 points are sampled per region. The up-sampled resolution of the feature map R is set to 64. We use a teacher temperature τ_t of 0.07 and student temperature τ_s of 0.1, with 30 epochs as a warm-up stage where no distillation is applied. The balancing ratios for losses are set as α=0.5 and β=0.7. For optimization hyper-parameters (*e.g.* learning rate, batch size *etc.*) and augmentation recipes we follow MoCo v2 (42). We follow the same strategy in DetCon (112) to sample region pairs through random crops, and skip the loss computation for points when views share no overlapping region, which happens rarely in practice.

method	# of epochs	Pascal VOC			COCO detection			COCO segmentation			Cityscapes
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}	mIoU
Scratch	-	33.8	60.2	33.1	29.9	47.9	32.0	32.8	50.9	35.3	63.5
MoCo v2 (42)	800	54.7	81.0	60.6	38.5	58.1	42.1	34.8	55.3	37.3	73.8
BYOL (100)	800	-	-	-	37.9	57.5	40.9	-	-	-	-
Self-EMD (159)	800	-	-	-	38.5	58.3	41.6	-	-	-	-
PixPro (286)	800	56.5	81.4	62.7	39.0	58.9	43.0	35.4	56.2	38.1	75.2
Ours	800	**57.1**	**82.1**	**63.8**	**39.8**	**59.6**	**43.7**	**35.9**	**56.9**	**38.6**	**75.9**

Table 5.2: Main results with COCO pre-training. Same as ImageNet-1K, from left to right, we show the performance on 4 tasks: VOC (07+12) detection, COCO detection; COCO instance segmentation and Cityscapes semantic segmentation. From top to down, we compare with training from scratch and pre-training with self-supervision. For COCO pre-training, our method shows significant improvements.

5.4.2 Downstream Tasks

We evaluate feature transfer performance on four downstream tasks: object detection on VOC (79), object detection and instance segmentation on COCO (152), and semantic segmentation on Cityscapes (54).

VOC. PASCL VOC is the default dataset to evaluate self-supervised pre-training for object detection. We follow the setting introduced in MoCo (107), namely a Faster R-CNN detector (205) with the ResNet-50 *C4* backbone, which uses the *conv4* feature map to produce object proposals and uses the *conv5* stage for proposal classification and bounding box regression. In fine-tuning, we synchronize all batch normalization layers across devices. Training is performed on the combined set of `trainval2007` and `trainval2012`. For testing, we report AP, AP50 and AP75 on the `test2007` set. Detectron2 (271) is used.

COCO. On COCO we study both object bounding box detection and instance segmentation. We adopt Mask R-CNN (109) with ResNet-50 *C4* as the backbone and head. Other setups are the same as VOC. Detectron2 is again used. We follow the standard $1\times$ schedule for fine-tuning, which is 90k iterations for COCO.

Cityscapes. On Cityscapes we evaluate semantic segmentation, a task that also relies on good localization and recognition. We follow the previous settings (107; 286), where a FCN-based structure is used (162). The classification is obtained by an additional 1×1 convolutional layer.

5.4.3 Main Results

ImageNet-1K pre-training. Tab. 5.1 compares our point-level region contrast to previous state-of-the-art unsupervised pre-training approaches on 4 downstream tasks, which all require dense predictions. We compare with four categories of methods: 1) training from scratch, *i.e*. learning the network from random initialization; 2) ImageNet-1K supervised pre-training; 3) general self-supervised pre-training, including MoCo, MoCo v2, SimCLR and InfoMin. Those are under their reported epochs; 4) Task-specific pre-training, including DetCo (280), PixPro (286), DenseCL (262) and DetCon (112). We report the numbers with 200-epoch pre-training. It is worth noting that we adopt the asymmetric network structure (42), *i.e*. each view is only used once per iteration. For this reason, we denote PixPro (100-epoch reported in (262)) and SoCo (266) as 200 epochs since the loss is symmetrized there. DetCon (112) uses pre-defined segmentation masks acquired by off-the-shelf algorithms. We also compared with it under the same number of epochs.

It shows consistent improvement on every tasks compared with prior arts under this *fair* comparison setting on VOC object detection, COCO object detection, COCO instance segmentation and Cityscapes semantic segmentation.

COCO pre-training. Tab. 5.2 compares our method to previous state-of-the-art unsupervised pre-training approaches on COCO. We evaluate the transferring ability to the same 4 downstream tasks used for ImageNet-1K pre-training, and on all of them we show significant improvements. Different from ImageNet-1K, COCO images have more objects per-image on average, thus our point-level region contrast is potentially more reasonable and beneficial in this setting.

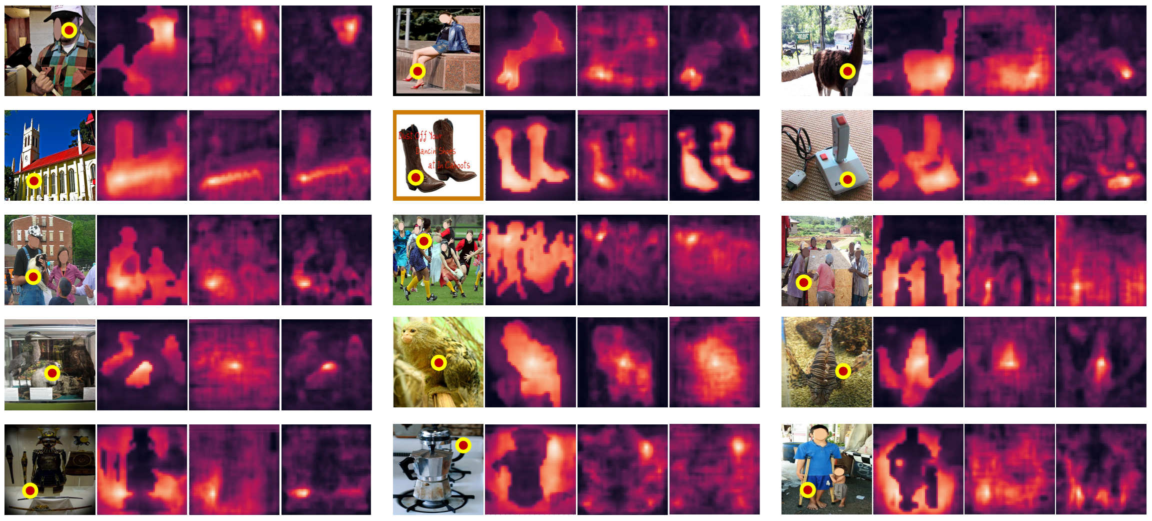


Figure 5.3: Point affinity visualizations. In total we show 15 groups of examples. In each group from left to right we show the original image with the selected point (denoted by red circle); three affinity maps calculated from the point to the rest of the image with the output of i) our point-level region contrast; ii) region-level contrast; and iii) MoCo v2 (image-level contrast). In rows from top to down, we show 5 categories of picked points: i) single non-rigid objects, ii) single rigid objects, iii) multiple objects, iv) objects in chaotic background and v) background stuff. Brighter colors in the affinity map denote more similar points. Best viewed in color and zoomed in.

5.4.4 Visualization of Point Affinities

In order to provide a more intuitive way to show the effectiveness of our method, we visualize the point affinities after pre-training in Fig. 5.3.

The images are randomly chosen from the validation set of ImageNet-1K. We follow the previous experimental setting to pre-train 200 epochs on ImageNet-1K. We then resize all images to 896×896, and interpolate the corresponding feature map of *Res5* from (28×28) to 56×56 for higher resolution. For each image, we first pick one point (denoted with a red circle), then calculate the point affinity (in terms of cosine similarity) from the last-layer output feature representation of this point to all the others within the same image.

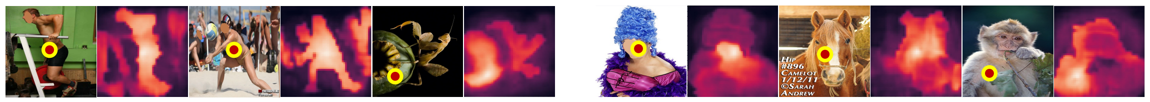


Figure 5.4: Point affinity (failures). We present two kinds of failure cases for our method: under-segmentation (left) and over-segmentation (right). For each kind we show 3 pairs of images, using the same visualization technique in Fig. 5.3. See text for details.

In addition, we also compare it with the visualizations from MoCo v2 and a region-level contrast variant of our method to analyse the improvement. The region-level contrast variant is implemented using MoCo v2 framework with grid regions (same as ours), with an AP of 58.2 on VOC. In Fig. 5.3, from top to down we show 15 different groups of examples which (row-wise) represent 5 categories of picked points: single non-rigid objects, single rigid objects, multiple objects, objects in chaotic background, and background. Within each group, from left to right we show the point affinity of our method, region-level contrast, and the MoCo v2 baseline. Brighter colors on the feature map denote more similar points.

Observations. For original MoCo, its final global pooling operation intuitively causes a loss in 2D spatial information, since everything is compressed into a single vector for representations. Therefore, when tracing back, the salient regions usually only cover certain closely-connected small area around the picked point. For the region-level contrast baseline, its salient regions can expand to a larger area, but the area is quite blurry and hard to tell the boundaries. For objects (shown in row 1-3), although all three methods show some localization capabilities, ours often predicts sharper and more clear boundaries, indicating a better understanding of the localization of objects. Row 4 shows the objects in chaotic environments, which is hard to recognize even with human eyes. Except for foreground objects, we also test the ability on the background stuff (row 5). It is interesting to see that even for background, ours can still distinguish it with foreground objects.

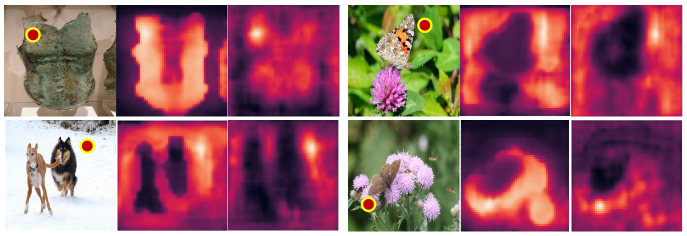


Figure 5.5: Point affinity *with* or *without* affinity distillation. In each of the 4 groups, we show (from left to right) the original image, ours *with* point affinity distillation and ours *without*. As can be seen, the distillation loss plays a key role in capturing object boundaries, as shown in the 4 groups of examples.

random	supervised	image-	region-	ours
15.3	22.9	33.1	33.8	**52.0**

Table 5.3: Quantitative metric to compare VOC visualizations from different pre-training methods. Our point-level region contrast outperforms all baselines ranging from random, supervised pre-training and self-supervised pre-training at various levels.

Failure cases. We also give some failure cases from our model in Fig. 5.4. On the left we show *under-segmentation*, where a segment contains more objects than it should be. For example, in the first image, both the man and the running machine have higher similarity to the chosen point. On the contrary, on the right we show *over-segmentation*, where a segment does not cover the entire object. For example, the face of the woman has higher similarity to the chosen point, while the clothes and wig have lower similarities – ideally they should all belong to the same person. We believe this is reasonable in our unsupervised setting: without definition of object classes, the model can at best form groups using low-level cues such as textures or colors; therefore, it can miss semantic-level grouping of objects.

Affinity distillation helps localization. We visualize our method with or without point affinity distillation in Fig. 5.5. We find the distillation loss plays a key role in capturing object boundaries for better localization.

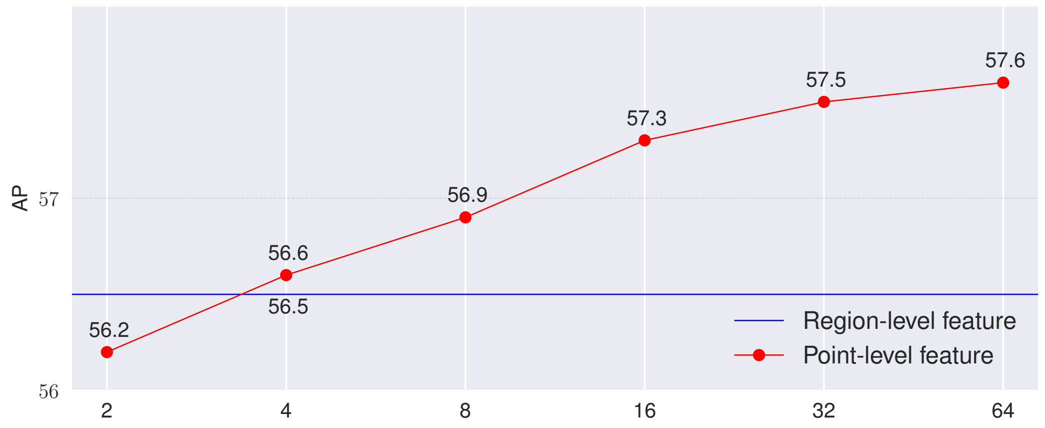


Figure 5.6: Point-level *vs*. region-level features. We check how many points are needed to match a region-level representation when pre-trained on ImageNet-1K. Along the horizontal axis the number of points increases from 2 to 64 for point-level features. The pre-trained representation can already match region-level features (blue line) in VOC AP with only 4 points.

Quantitative Metric. We quantitatively evaluate the visualizations from different pre-training methods on VOC `val2007`. For each ground-truth object, we pick its center point and calculate the similarity from this point to the rest of the image with the pre-trained model to generate segmentation masks. We pick a threshold to keep 80% of the entire affinity map. The mask is then benchmarked with Jaccard similarity, defined as the intersection over union (IoU) between the predicted mask and the ground-truth one.

Our baselines are: random (no pre-training), supervised (on ImageNet-1K), MoCo v2 (image-level) and region-level. The results are summarized in Tab. 5.3. As expected, point-level region contrast significantly outperforms others.

5.4.5 Main Ablation Studies

For our main ablation analysis, we begin with point-level contrastive learning in Sec. 5.4.5.1, showing its effectiveness to represent regions and robustness to inferior initial regions compared to a region-based counterpart. Then we discuss and compare possible point affinity distillation strategies in Sec. 5.4.5.2. More ablations are found in the appendix. Throughout this section, we pre-train for 100 epochs on ImageNet-1K and 400 COCO epochs on COCO.

5.4.5.1 Point-Level *vs*. Region-Level

We first design experiments to show the motivation and effectiveness of introducing point-level operations to region-level contrast. We conduct two experiments.

First is to see how many points are needed to match the pooled region-level features. We pre-train on ImageNet for 100 epochs *without point affinity loss* for fair comparisons, and report results with VOC object detection transfer. As shown in Fig. 5.6, we find with only *4* points per-region, its AP (56.6) is already better than region-level contrast (56.5). Interestingly, more point-level features continue to benefit performance even up to 64 points, which suggests that the pooled, region-level features are not as effective as point-level ones for object detection pre-training.

Second, we add back the point affinity loss and compare the robustness of our full method against contrast learning with aggregated region-level features (112). For this experiment, we pre-train on COCO as COCO is annotated with ground-truth object boxes/-masks. P=16 points are used per-region and evaluation is also performed with VOC object detection. In Fig. 5.7 we gradually decrease the region quality, from highest (ground-truth mask), to lowest (2×2 grid) with ground-truth box and 4×4 grid in-between. Not only does point-level region contrast perform better than region-level contrast, the gap between the two *increases* as the region quality degenerates from left to right. This confirms that our

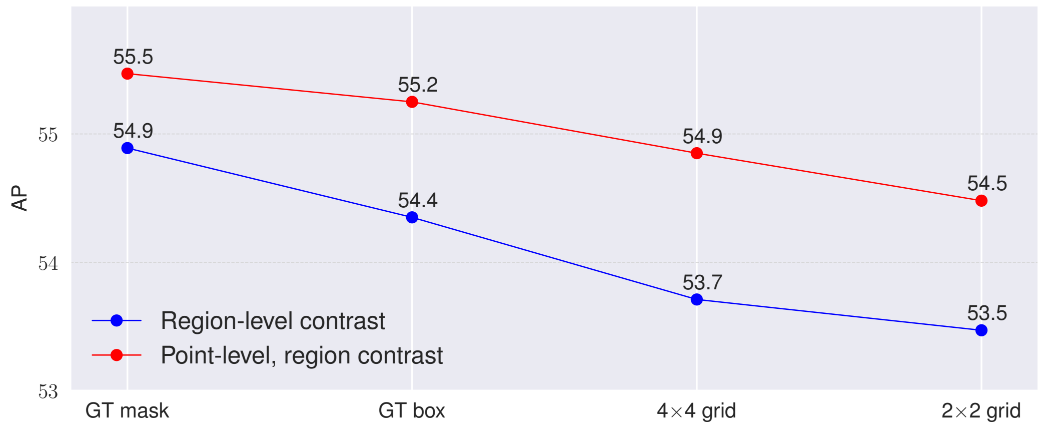


Figure 5.7: Region quality *vs*. AP comparison between our point-level region contrast (red) and region-level contrast with pooled-features, pre-trained on COCO. Along the horizontal axis the region quality degenerates: ground truth masks, ground truth bounding box, 4×4 grid and 2×2 grid. Our method is consistently better and is more resilient to the degeneration of region qualities.

method is more *robust* to initial region assignments and can work with all types of regions.

5.4.5.2 Point Affinity Distillation Strategies

For point affinity distillation, there are three possible strategies: 1) $_{i'k'}$ as teacher (see Eq. (5.3) for its definition), $_{ik'}$ as student (default); 2) $_{ik'}$ as teacher, $_{ik}$ as student; 3) $_{i'k'}$ as teacher, $_{ik}$ as student, which requires an *extra* forward pass with momentum encoders.

Strategy 1) achieves 58.0 AP. Switching to strategy 2) slightly degenerates AP to 57.6, and strategy 3) yields the same AP as 1) while requiring extra computations. Therefore we set 1) as our default setting.

5.5 Conclusion

Balancing recognition and localization, we introduced point-level region contrast, which performs self-supervised pre-training by directly sampling individual point pairs from different regions. Compared to other contrastive formulations, our approach can learn both inter-image and intra-image distinctions, and is more resilient to imperfect unsupervised regions assignments. We empirically verified the effectiveness of our approach on multiple setups and showed strong results against state-of-the-art pre-training methods for object detection. We hope our explorations can provide new perspective and inspirations to the community.

Chapter 6

Masked Autoencoders Enable Efficient Knowledge Distillers

Next, based on the finding that representation learning can show certain level of generalization ability among downstream tasks. We keep pushing for stronger representation learning algorithm, and propose DMAE.

6.1 Introduction

Following the success in the natural language processing (253; 67), the Transformer architecture is showing tremendous potentials in computer vision (72; 242; 245; 11; 300), especially when they are pre-trained with a huge amount of unlabelled data (17) with self-supervised learning techniques(108; 10). Masked image modeling, which trains models to predict the masked signals (either as raw pixels or as semantic tokens) of the input image, stands as one of the most powerful ways for feature pre-training. With the most recent representative work in this direction, masked autoencoder (MAE) (104), we are now able to efficiently and effectively pre-train high-capacity Vision Transformers (ViTs) with strong

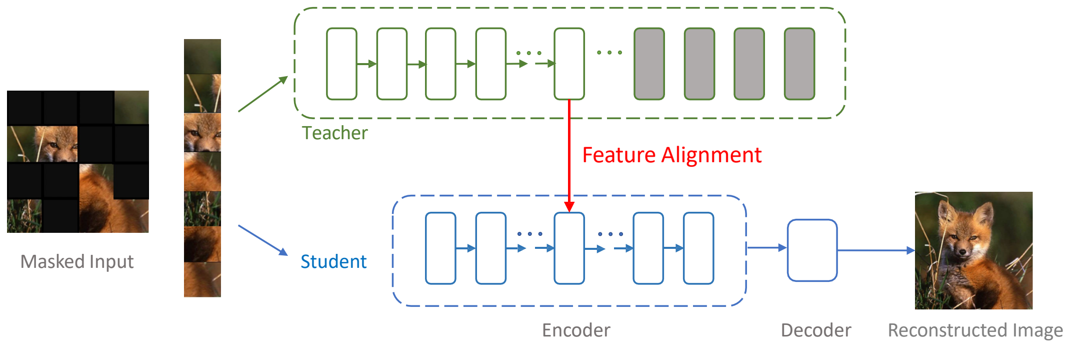


Figure 6.1: Illustration of the distillation process in DMAE. There are two key designs. Firstly, following MAE, we hereby only take visible patches as inputs and aim to reconstruct the masked ones. Secondly, knowledge distillation is achieved by aligning the intermediate features between the teacher model and the student model. Note the gray blocks denote the dropped high-level layers of the teacher model during distillation.

feature representations, leading to state-of-the-art solutions for a wide range of downstream visual tasks.

In this paper, we are interested in applying knowledge distillation (120), which is one of the most popular model compression techniques, to transfer the knowledge from these strong but cumbersome ViTs into smaller ones. In contrast to prior knowledge distillation works (120; 312; 173), the teacher considered here is a pre-trained model whose predictions do not necessarily reveal the fine-grained relationship between categories; therefore, typical solutions like aligning the soft/hard logits between the teacher model and the student model may no longer remain effective. Moreover, after distilling the pre-trained teacher model, these student models need an extra round of fine-tuning to adapt to downstream tasks. These factors turn distilling pre-trained models seemingly a less favorable design choice in terms of both performance and computational cost.

Nonetheless, surprisingly, we find by building upon MAE, the whole distillation framework can efficiently yield high-performance student models. There are two key designs.

Firstly, we follow MAE to let the encoder exclusively operate on a small visible subset of patches and to employ a lightweight decoder for pixel reconstruction. Whereas rather than using the "luxury" setups in MAE, we show aggressively *simplifying pre-training from 1600 epochs to 100 epochs* and *pushing masking ratio from 75% to 95%* suffice to distill strong student models. Secondly, instead of aligning logits, we alternatively seek to match the intermediate feature representation; this enables the cumbersome teacher model to only forward propagate inputs through the first few layers, therefore, reducing computations. We note applying *L1 norm for distance measure* is an essential recipe for ensuring a successful intermediate feature alignment.

We name this distilling MAE framework as DMAE. Compared to the traditional knowledge distillation framework where the teacher is a fine-tuned model, DMAE is more efficient and can train much stronger student models at different capacities. For example, by setting ViT-B as the student model, while the baseline of distilling a fine-tuned ViT-L achieves 82.8% top-1 ImageNet accuracy, DMAE substantially boosts the performance to 84.0% (+1.2%) top-1 ImageNet accuracy, at an even lower training cost (*i.e.*, 195 GPU hours *vs*. 208 GPU hours, see Table 6.9). More intriguingly, we found that DMAE allows for robust training with extremely highly masked images—even with TEN visible patches (*i.e.*, 95% masking ratio), ViT-B can competitively attain a top-1 ImageNet accuracy of 83.6%; this masking ratio can further be aggressively pushed to 98% (FOUR visible patches) where DMAE still help ViT-B secure 82.4% top-1 ImageNet accuracy. We hope this work can benefit future research on efficiently unleashing the power of pre-train models.

6.2 Related Work

Knowledge distillation (KD) is a popular model compression technique that allows models to achieve both strong performances of large models and fast inference speed of small models. The first and seminal KD approach, proposed in (120), transfers the "dark knowledge" via minimizing the KL divergence between the soft logits of the teacher model and that of the student model. From then on, many advanced KD methods have been developed, which can be categorized into two branches: logits distillation (84; 312; 50; 173; 315; 276) and intermediate representation distillation (208; 135; 118; 117; 236). Our DMAE belongs to the second branch, as it minimizes the distance between latent features of the teacher model and those of the student model.

The first feature-based distillation method is FitNets (208). In addition to aligning logits, FitNets requires the student model to learn an intermediate representation that is predictive of the intermediate representations of the teacher network. Heo et al. (117) re-investigates the design of feature distillation and develops a novel KD method to create a synergy among various aspects, including teacher transform, student transform, distillation feature position, and distance function. CRD (236) incorporates contrastive learning into KD to capture correlations and higher-order output dependencies. Unlike these existing works, our DMAE is the first to consider applying KD to extra information from self-supervised pre-trained models.

Masked image modeling (MIM) helps models acquire meaningful representations by reconstructing masked images. The pioneering works are built on denoising autoencoders (256) and context encoders (187). Following the success of BERT in natural language (67), and also with the recent trend of adopting Transformer (253) to computer vision (72), there has emerged a set of promising works on applying MIM for self-supervised visual pre-training. BEiT (17) first successfully adopts MIM to ViT pre-training by learning to predict visual tokens. MaskFeat (264) finds that learning to reconstruct HOG features enables effective visual representation learning. SimMIM (287) and MAE (104) both propose to directly reconstruct the pixel values of the masked image patches. Our work is built on MAE and finds that MAE enables the whole distillation framework to be efficient and effective.

6.3 Approach

6.3.1 Masked Autoencoders

Our method is built upon MAE, a powerful autoencoder-based MIM approach. Specifically, the MAE encoder first projects unmasked patches to a latent space, which are then fed into the MAE decoder to help predict pixel values of masked patches. The core elements in MAE include:

Masking. MAE operates on image tokens, *i.e.*, the image needs to be divided into non-overlapping patches. A random small subset of those patches will be kept for the MAE encoder, and the rest will be set as the predicting target of the MAE decoder. Typically, a high masking ratio (*e.g.*,75%) is applied, preventing models from taking shortcuts (*e.g.*, simply extrapolating missing pixels based on neighbors) in representation learning.

MAE encoder. The MAE encoder is a standard ViT architecture except that it only operates on those unmasked patches. This design largely reduces the computation cost of encoders.

MAE decoder. In addition to the encoded features of unmasked patches (from MAE encoder), the MAE decoder receives mask tokens as input, a learned vector shared across all missing positions. The mask token is only used during pre-training, allowing independent decoder design. Particularly, MAE adopts a lightweight decoder for saving computations.

Reconstruction. Different from BEiT (17) or MaskFeat (264), MAE directly reconstructs image pixel values. The simple mean squared error is applied to masked tokens for calculating loss.

Note that, other than distillation-related operations, the whole pre-training and fine-tuning process in this paper exactly follow the default setup in MAE, unless specifically mentioned. Interestingly, compared to MAE, our DMAE robustly enables a much more efficient pre-training setup, *e.g.*, 100 (*vs*. 1600) training epochs and 95% (*vs*. 75%) masking ratio.

6.3.2 Knowledge Distillation

MAE demonstrates extraordinary capabilities in learning high-capacity models efficiently and effectively. In this work, we seek to combine knowledge distillation with the MAE framework, to efficiently acquire small and fast models with similar performance as those powerful yet cumbersome models. The most straightforward approach is directly applying existing knowledge distillation methods, like the one proposed in DeiT (242), to a fine-tuned MAE model. However, we empirically find that this approach hardly brings in improvements. In addition, this approach fails to leverage the special designs in MAE for reducing computations, *e.g.*, only feeding a small portion of the input image to the encoder. To this end, we hereby study an alternative solution: directly applying knowledge distillation at the pre-training stage.

Since there are no categorical labels in MAE pre-training, distilling logits can hardly help learn semantically meaningful representations. We, therefore, resort to distilling the intermediate features. This idea is first developed in FitNets (208), and inspired a set of followups for advancing knowledge distillation (135; 118; 117; 236). Concretely, we first extract the features from the specific layers of the student model; after feeding such features into a small project head, the outputs will be asked to mimic the features from the corresponding layers of the teacher model. In practice, the projection head is implemented by simple fully connected layer, which addresses the possible feature dimension mismatch between teacher models and student models.

Formally, let $\mathbf{x} \in \mathbb{R}^{3HW \times 1}$ be the input pixel RGB values and $\mathbf{y} \in \mathbb{R}^{3HW \times 1}$ be the predicted pixel values, where H denotes image height and W denotes image width. The MAE reconstruction loss L_{MAE} can be written as

$$L_{MAE} = \frac{1}{\Omega\left(\mathbf{x}_M\right)} \sum_{i \in M} \left(\mathbf{y}_i - \mathbf{x}_i\right)^2 . \tag{6.1}$$

where M denotes the set of masked pixels, $\Omega(.)$ is the number of elements, and i is the pixel index.

Let $\mathbf{z}_l^S, \mathbf{z}_l^T \in \mathbb{R}^{LC \times 1}$ be the features extracted from the lth layer of the student model and the teacher model, respectively, where L denotes the patch numbers, and C denotes the channel dimension. We use $\sigma()$ to denote the projection network function. Our feature alignment distillation loss L_{Dist} can be written as

$$L_{Dist} = \sum_l \frac{1}{\Omega\left(\mathbf{z}_l^T\right)} \sum_i \left\| \sigma\left(\mathbf{z}_l^S\right)_i - \mathbf{z}_{l,i}^T \right\|_1 . \tag{6.2}$$

# of Layers	Layer Location	Student Aligned Layer Index	ImageNet Top-1 Acc (%)
Single	Bottom	3	82.6
	Middle	6	83.6
		9	**84.0**
	Top	12	83.4
Multiple	Middle+Top	6+12	**84.2**

Table 6.1: The effects of feature alignment location. We hereby test with 5 different layer locations, where the top layers refer to those closer to network outputs. For single layer feature alignment, features from the $\frac{3}{4}$ depth of the model leads to the best ImageNet top-1 accuracy. This performance is even comparable to the setup of aligning multiple layers.

The final loss used in pre-training is a weighted summation of MAE reconstruction loss L_{MAE} and the feature alignment distillation loss L_{Dist}, controlled by the hyperparameter α:

$$L = L_{MAE} + \alpha \times L_{Dist}. \tag{6.3}$$

The framework of DMAE is summarized in Figure 6.1. Following MAE, DMAE also takes masked inputs and performs the pretext task of masked image modeling. Besides, the corresponding features are aligned between the teacher model and the student model. It is worthy of highlighting that DMAE is an efficient knowledge distiller: 1) it only operates on a tiny subset of visible patches *i.e.*, a high masking ratio is applied; and 2) aligning intermediate layer features reduces the computation cost of (cumbersome) teacher model. In the next section, we extensively compare our method with three baselines: the original MAE without any distillation, DeiT-style distillation, and feature alignment distillation in the supervised setting.

6.4 Experiments

6.4.1 Implementation Details

Following MAE (104), we first perform self-supervised pre-training on ImageNet-1k (65). Unless otherwise mentioned, the teacher models are public checkpoints released from the official MAE implementations. For pre-training, we train all models using AdamW optimizer (164), with a base learning rate of 1.5e-4, weight decay of 0.05, and optimizer momentum $\beta_1, \beta_2 = 0.9, 0.95$. We use a total batch size of 4096, and pre-train models for 100 epochs with a warmup epoch of 20 and a cosine learning rate decay schedule. We by default use the masking ratio of 75%; while the ablation study shows that our method can robustly tackle extremely high masking ratios. *After pre-training, the teacher model is dropped, and we exactly follow the default setups in MAE to fine-tune the student model on ImageNet*.

For feature alignments, we choose to align the features from the $\frac{3}{4}$ depth of both the student model and the teacher model, which we find delivers decent results for all model sizes tested. For example, with a 24-layer ViT-L as the teacher model and a 12-layer ViT-B as the student model, features from the 9th layer of ViT-B are aligned with the features from the 18th layer of ViT-L. We set $\alpha = 1$ in Eq. 6.3 to balance the tradeoff between MAE reconstruction loss L_{MAE} and the feature alignment distillation loss L_{Dist} in pre-training.

6.4.2 Analysis

We first provide a detailed analysis of how to set distillation-related parameters in DMAE. Specifically, we set the teacher model as an MAE pre-trained ViT-L (from MAE official

Teacher Layer (relative position)	ImageNet Top-1 Acc (%)
Middle	84.0
Top	83.3
Bottom	82.1

Table 6.2: The analysis of aligning order on ImageNet classification top-1 accuracy (%).

GitHub repository, attaining 85.9% top-1 ImageNet accuracy after fine-tuning), and set the student model as a randomly initialized ViT-B. We analyze the following six factors:

Where to align. In Table 6.1 we first check the effect of feature alignment location on model performance. We observe that shallower features are less favored: *e.g.*, the 3rd layer alignment under-performs all other settings. We speculate this is due to the learning process of ViTs—images are much noisier and less semantic than texts, ViTs will first group the raw pixels in the bottom layers (closer to the input), which is harder to transfer. While features from $\frac{3}{4}$ depth (*i.e.*, the 9th layer in ViT-B) achieves the best performance, a simple rule-of-thumb which we find fits all model scales in our experiments. We adopt this design choice in all other experiments. It is also worth mentioning that simply aligning multiple layers has no clear advantage over aligning features from $\frac{3}{4}$ depth (84.2% *vs*. 84.0%); we, therefore, stick to the $\frac{3}{4}$ depth setting, which is more efficient.

Aligning order. We next test the importance of alignment ordering on model performance. Specifically, by fixing the layer location in the student model (*i.e.*, the middle layer in our experiment), we then align it to different layers of the teacher model. As shown in Table 6.2, we observe that when the aligned layers are in the same relative position (*e.g.*, middle to middle), the student model can achieve the best performance.

Masking ratio. MAE reveals that the masking ratio in masked image modeling could be surprisingly high (75%). The hypothesis is that by learning to reason about the gestalt of the

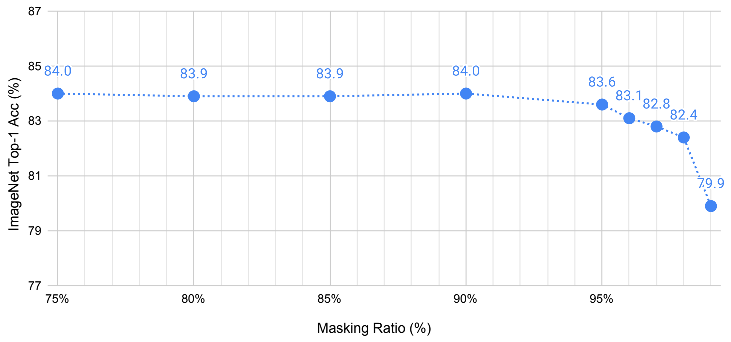


Figure 6.2: DMAE allows an extremely high masking ratio. From left to right, we increase the masking ratio from the basic 75% to the extreme 99%. We note that our DMAE competitively attains 83.6% top-1 ImageNet accuracy with 95% masking ratio (TEN visible patches), and still secures 82.4% top-1 ImageNet accuracy even by learning with FOUR visible patches (98% masking ratio).

Projection Head	ImageNet Top-1 Acc (%)
Linear	84.0
2-layer MLP	84.0
3-layer MLP	83.8

Table 6.3: Projection Head. A simple fully-connected layer works best. We choose this as the default setting.

Decoder Depth	ImageNet Top-1 Acc (%)
2	83.7
4	83.8
8	84.0

Table 6.4: Decoder Depth. A deeper decoder (slightly) improves the pre-trained representation quality.

missing objects and scenes, which cannot be done by extending lines or textures because of the high masking ratio, the model is also learning useful representations. Interestingly, we find that when combining MAE and knowledge distillation, an even much higher masking ratio is possible, as shown in Figure 6.2.

Firstly, it is interesting to note that, compared to the typical 75% masking ratio setting, further raising the masking ratio to 90% comes at no performance drop, *i.e.*, both attain

	Loss Design	ImageNet Top-1 Acc (%)
Loss Choice	L1 with $\alpha = 1$	84.0
	L2 with $\alpha = 1$	83.3
Loss Ratio	L1 with $\alpha = 0.5$	83.9
	L1 with $\alpha = 1$	84.0
	L1 with $\alpha = 2$	84.0
	L1 with $\alpha = 4$	84.0

Table 6.5: Loss Function. From the first block, we observe that the L1 distance yields significantly higher performance than the L2 distance. From the second block, we observe that the hyperparameter α has little influence on the representation quality of DMAE.

Method	Pre-training epochs	Supervised training / fine-tuning epochs	ImageNet Top-1 Acc (%)
MAE-B	100	100	81.6
MAE-B	1600	100	83.6
DeiT-B	-	100	76.8
DeiT-B	-	300	81.8
DeiT-B-Soft Distillation	-	100	77.5
DeiT-B-Hard Distillation	-	100	78.3
CRD(236)	-	100	81.9
SRRL(295)	-	100	82.2
Dear-KD(41)	-	100	82.4
Supervised Feature Alignment	-	100	82.8
DMAE-B	100	100	84.0

Table 6.6: DMAE shows stronger performance than all three kinds of baselines: MAE, supervised model, and other existing advanced distillation strategies.

84.0% top-1 ImageNet accuracy. Next, even with an extremely large masking ratio like 98% (only FOUR visible patches), DMAE still beats the 100-epoch MAE baseline that uses a masking ratio of 75% (second row in Table 6.6), by a non-trivial-margin (82.4% *vs*. 81.6%). These results suggest that, with the assistance of distilled knowledge from the teacher model, the student model can make better use of visible patches, even at a very limited amount, for representation learning.

Projection head. The goal of the proposed feature alignment distillation is to encourage the student model to learn features that are predictive of features from a stronger teacher model. To that end, a small projection head is employed on features from the student model, to 1) project them onto a space of the same dimension as the hidden dimension of the teacher model, and 2) provide extra flexibility for feature alignment. We ablate the choice of this projection network, as shown in Table 6.3. We can observe that applying a simple fully connected layer already performs the best among other choices.

Decoder depth. In Table 6.4 we analyze the effect of decoder depth. Similar to MAE (104), the final performance gets (slightly) increased with a deeper decoder. We choose a decoder depth of 8 as the default setting as in (104). Note that a decoder depth of 2 is also a competitive choice—compared to an 8-depth decoder, it significantly reduces the computation cost while only marginally sacrificing the accuracy by 0.3%.

Loss designs. Table 6.5 ablates the loss design. While SimMIM (287) shows that L1 distance and L2 distance lead to similar performance, ours suggests that L1 distance exhibits a clear advantage over L2 distance, *i.e.*, +0.7% improvement. Furthermore, we note DMAE is quite robust to the specific value of the hyperparameter α, which controls the relative importance of the distillation loss over the reconstruction loss. Based on these results, we choose L1 in Eq. 6.2 for distance measure and set $\alpha = 1$ in Eq. 6.3 for the rest experiments.

6.4.3 Comparison with Baselines.

In Table 6.6, we compare the performance of our DMAE with various baselines:

MAE. The MAE baselines are presented in the first block of Table 6.6. If MAE is also asked to pre-train for only 100 epochs, DMAE can substantially outperform this baseline by 2.4%

(from 81.6% to 84.0%). When comparing to a much stronger but more computationally expensive MAE baseline with 1600 pre-training epochs, we note DMAE still beats it by 0.4%.

Supervised model. The second block in Table 6.6 demonstrates the effectiveness of DMAE compared with models trained under the supervision of categorical labels, which requires a much longer training time, *i.e.*, +2.2% compared to the DeiT 300 epochs supervised training.

Other distillation strategies. We next compare DMAE with other distillation methods. We consider DeiT-style logit-based distillation (242), CRD (236), SRRL (295), Dear-KD (41), and feature alignment distillation. Note for these baselines, one significant difference from DMAE is that the student model here directly distills knowledge from a supervisely fine-tuned teacher model. Moreover, to make these baselines more competitive, the teacher models will first be MAE pre-trained and then fine-tuned on ImageNet-1k.

The results are shown in the third block of Table 6.6. Firstly, we can observe that the DeiT-style logit-based distillation, either soft or hard, even hurts the student models' performance. This phenomenon potentially suggests that such a distillation strategy may not fit teacher models of ViT architectures. For other baselines, we note that feature alignment distillation performs the best; but this is still worse than DMAE (82.8% *vs.* 84.0%), indicating the importance and effectiveness of distilling knowledge from a pre-trained teacher model.

Student Model	Teacher Model	ImageNet Top-1 Acc (%)	
		MAE	DMAE
Base	Large	81.6	84.0 (+2.4)
Small	Large	77.4	80.1 (+2.7)
	Base	77.4	79.3 (+1.9)
Tiny	Base	66.6	70.0 (+3.4)

Table 6.7: Across different model sizes, DMAE shows consistent improvements compared with MAE.

6.4.4 Scaling to Different Model Sizes

We test DMAE with different model sizes, listed in Table 6.7. For a fair comparison, both methods only pre-train models for 100 epochs. DMAE shows consistent improvement compared to MAE across different model sizes. With *only one middle layer* feature alignment, DMAE brings an additional improvement of +2.4% with ViT-B, +2.7% with ViT-Small, and +3.4% with ViT-Tiny. In addition, we are interested in the following two cases:

Same teacher model, different student model. As shown in the first two lines in Table 6.7, we find that when using ViT-L as the teacher model, both ViT-B and ViT-S benefit from the distillation, demonstrating a clear advantage over the MAE baseline. We argue that the ability to effectively generalize to cases where an even smaller student model is desirable, especially for those computation-constrained real-world applications.

Same student model, different teacher model. As shown in Table 6.7, with a ViT-S as the student model, enlarging the teacher model from ViT-B to ViT-L further boosts the accuracy by 0.8% (from 79.3% to 80.1%). This result suggests that DMAE can effectively distill knowledge from the teacher models at different scales.

Method	Pre-training epochs	Supervised training / fine-tuning epochs	IN-1% Top-1 Acc(%)	IN-10% Top-1 Acc(%)
MAE-B	100	100	33.9	65.0
MAE-B	1600	100	49.6	72.8
DeiT-B	-	100	-	-
DeiT-B-Soft Distillation	1600	100	36.0	66.4
DeiT-B-Hard Distillation	1600	100	37.3	67.3
Supervised Feature Alignment	1600	100	34.2	67.6
DMAE-B	100	100	50.3	73.4

Table 6.8: DMAE demonstrates much stronger performance than all other baselines when training data is limited. Note that the DeiT-B baseline is unable to converge because of data insufficiency.

Model	Training Cost (GPU Hours)			ImageNet
	Pre-training	Fine-tuning	Overall	Top-1 Acc(%)
MAE-B-100-epoch	78h	112h	190h	81.6
MAE-B-1600-epoch	1248h	112h	1360h	83.6
DeiT-B-Soft Distillation	-	213h	213h	77.5
DeiT-B-Hard Distillation	-	213h	213h	78.3
Supervised Feature Alignment	-	208h	208h	82.8
DMAE-B	83h	112h	195h	**84.0**

Table 6.9: Computational cost comparisons among DMAE and other baselines. The training cost is measured by A5000 GPU hours. We note the proposed DMAE maintains a similar (or even cheaper) training cost than others, while achieving much higher top-1 ImageNet accuracy.

6.4.5 Limited Training Data

In certain real-world applications, data could be hard to acquire because of high data collection and labeling costs or due to privacy concerns. Leveraging models pre-trained on large-scale unlabeled datasets for fine-tuning when only a small dataset of downstream tasks is available becomes a promising solution. Here we seek to test the potential of our DMAE in this data-scarce scenario. We strictly follow (39) to sample 1% or 10% of the labeled ILSVRC-12 training datasets in a class-balanced way. We set MAE pre-trained ViT-L as the teacher model and a randomly initialized ViT-B as the student model. In addition, we compare DMAE with three kinds of baselines described in Section 6.4.3: MAE, supervised model, and other distillation strategies, and similarly, set the teacher model to be a ViT-L that first pre-trained with MAE and then fine-tuned on ImageNet-1k. Note that since DMAE has full access to the 100% ImageNet dataset (without labels) during pre-training, to ensure a fair and competitive comparison, *we initialize all the baselines as the 1600-epoch MAE pre-trained model on ImageNet*.

Table 6.8 shows that DMAE largely surpasses all other baselines. For example, when only 10% ImageNet data is available for supervised training or fine-tuning, DMAE outperforms the MAE pre-trained baseline by 8.4% (*i.e.*, 73.4% *vs*. 65.0%). DMAE also significantly outperforms other distillation strategies, with an improvement ranging from 5.8% to 7.0%. We note this accuracy gap is even larger when only 1% ImageNet is available, demonstrating the data efficiency of DMAE.

6.4.6 Generalization to Other Methods

Lastly, we provide preliminary results of integrating DMAE into other self-supervised training frameworks, including DINO (31) and MoCo-V3 (47). Unlike MAE, which belongs to masked image modeling, DINO and MoCo-V3 are contrastive learning-based methods. Still, as shown in Table 6.10, without further hyperparameter tuning, DMAE effectively shows non-trivial improvements on top of both DINO (+1.3%) and MoCo-v3 (+1.4%), demonstrating the potential of distilling pre-trained models (rather than fine-tuned models as in most existing knowledge distillation frameworks).

	w/o DMAE	w/ DMAE
DINO	80.9	**82.2 (+1.3)**
MoCo-V3	81.1	**82.5 (+1.4)**

Table 6.10: DMAE effectively improves other self-supervised pre-training frameworks (including DINO and MoCo-v3) on ImageNet classification top-1 accuracy (%).

6.4.7 Generalization to Downstream tasks

Following ViT-Det(149), we conduct downstream fine-tuning on the COCO dataset, where MAE/DMAE pre-trained ViT-Base model is adopted as the plain backbone of Mask-RCNN. We train the model on the `train2017` split and evaluate it on the `val2017` split. We report results on bounding box object detection (AP^{box}) and instance segmentation (AP^{mask}), shown in Table 6.11. Our observations indicate that DMAE also demonstrates strong potential in downstream tasks like detection and segmentation.

method	AP^{box}	AP^{mask}
MAE	51.2	45.5
DMAE	**53.4**	**46.9**

Table 6.11: DMAE effectively improves the downstream tasks as well, including object detection and segmentation.

6.5 Discussion

In this section, we present a quantitative evaluation of the computational cost of DMAE compared to baseline methods. Additionally, we conduct a standard deviation analysis to assess the stability of our DMAE approach. Finally, we propose an advanced fine-tuning recipe for applying DMAE to smaller ViT models.

6.5.1 Computational Costs

In Table 6.9, we provide a quantitative evaluation of the computational cost, which is tested on a single NVIDIA A5000 GPU. We can observe that, without a significantly increase in training hours, DMAE substantially outperforms the MAE-100 baseline by +2.4% on ImageNet; this result even exceeds the MAE-1600 baseline, which causes 7x GPU hours than MAE-100 baseline. We additionally provide the actual GPU hours of other baselines, and find that the proposed DMAE stands as the most efficient one, meanwhile achieving the best top-1 ImageNet accuracy.

6.5.2 Standard Deviation Analysis

In the above experiments, we kept the same random seed. Following MAE, we perform the statistical analysis for DMAE by changing the random seeds. In Table 6.12, from top to down, we show three aligning settings; and from left to right, we show the results with the default seed, the average accuracy with three randomly sampled seeds, and their standard deviation, respectively. From these results, we could conclude that our DMAE can bring in statistically stable improvements.

Pos	Acc(%)	Avg	Standard Deviation
Bottom (3th)	82.6	82.50	0.20
Mid (6th)	83.6	83.67	0.08
Top (9th)	84.0	84.03	0.10

Table 6.12: Standard deviation analysis for DMAE. From top to bottom, we show three aligning settings, the model performance with the default seed, the average performance with three randomly sampled seeds, and their standard deviations, respectively. 'Pos' denotes the student distillation position. Acc denotes ImageNet Top-1 accuracy. 'Avg' denotes the average value over three times.

((**a**)) ViT-Tiny

ViT-Tiny	Top-1 Acc (%)
MAE	70.1
DeiT	74.5
DMAE	**74.9**

((**b**)) ViT-Small

ViT-Small	Top-1 Acc (%)
MAE	80.0
DeiT	81.2
DMAE	**82.2**

Table 6.13: Weaker augmentation and regularization helps smaller ViT models, during finetuning for both MAE and DMAE on ImageNet classification top-1 accuracy (%)

6.5.3 Smaller ViT Models.

Since the original MAE paper does not offer specialized recipes for ViT-Small and Tiny, we by default use the recipe for ViT-Base to fine-tune these smaller ViTs. However, this recipe includes strong regularization and augmentation techniques that might lead to over-regularization for the smaller ViTs. To address this issue, we experiment with a modified recipe with weaker augmentation and regularization by removing MixUp, CutMix, and Stochastic Depth. Results on ImageNet in Table 6.13 show that DMAE not only maintains its advantage over MAE, but also outperforms DeiT with this new recipe, highlighting its effectiveness.

6.6 Conclusion

Self-supervised pre-training has demonstrated great success for those exponentially growing models in the natural language domain. Recently, the rise of MAE shows that a similar paradigm also works for the computer vision domain, and now the development of vision models may embark on a similar trajectory as in the language domain. Yet, it is often desirable to have a well-balanced model between performance and speed in real-world applications. This work is a small step towards unleashing the potential of knowledge distillation, a popular model compression technique, within the MAE framework. Our DMAE is a simple, efficient, and effective knowledge distillation method: feature alignment during MAE pre-training. Extensive experiments on multiple model scales demonstrate the effectiveness of our approach. Moreover, an intriguing finding is that it allows for a masking ratio even higher than the already large one used in MAE (*i.e.*, 75%). We have also validated the effectiveness of our DMAE when in the small-data regime. We hope this work can benefit future research in knowledge distillation with pre-trained models.

Chapter 7

Sequential Modeling Enables Scalable Learning for Large Vision Models

In our final exploration, we delve into the data part. By introducing a novel sequential modeling approach, we aim to tackle the challenges posed by vast amounts of visual data efficiently. This chapter represents a crucial advancement towards handling the complexities of real-world visual data at scale, paving the way for scalable and versatile vision systems capable of addressing diverse challenges.

7.1 Introduction

Large language models (LLMs) such as GPT (25) and LLaMA (246) have taken the world by storm. What would it take to build a Large Vision Model (LVM)? From the animal world, we know that visual competences are not dependent on language. In particular, many experiments have shown that the visual world of non-human primates is remarkably similar to that of humans. So while the space of vision-language models such as LLaVA (155) is interesting and worthwhile to pursue, in this paper we seek an answer to a different question

– how far can we go from pixels alone?

The key features of contemporary LLMs that we seek to emulate in LVMs are: 1) scaling in the presence of big data, and 2) flexible specification of tasks through prompting (in-context learning). How do we achieve this? As usual, there are three main components that must be specified:

Data: We want to exploit all the remarkable diversity in visual data. First of all, just raw unannotated images and videos. Next, we want to exploit the variety of annotated visual data sources that have been produced over the last couple of decades – semantic segmentations, depth reconstructions, keypoints, multiple views of 3D objects, among others. We define a common format, "visual sentences", in which to represent these different annotations without needing any meta-knowledge beyond the pixels. The total size of our training dataset is 1.64 billion images/frames.

Architecture: We use a large transformer architecture (3 billion parameters) trained on visual data represented as sequence of tokens, using a learned tokenizer that maps each image to a string of 256 vector-quantized tokens.

Loss function: We draw inspiration from the natural language community, where masked token modeling has given way to sequential autoregressive prediction. Once images/videos/annotated images can all be represented as sequences, we can train the model to minimize the cross-entropy loss for predicting the next token.

With this extremely simple design, we demonstrate some noteworthy behaviors:

- Appropriate scaling behavior as one increases model size and data size.
- Many different vision tasks can now be "solved" by designing suitable prompts at

test time. While the results don't show as high performance as bespoke, specifically-trained models, the fact that so many tasks are all addressed by a single vision model is quite encouraging.

- We see a clear benefit of the amount of unsupervised data on the performance on various standard vision tasks.
- We see a hint of an ability for general visual reasoning – handling out-of-distribution data, and performing novel tasks. But further investigation is needed.

7.2 Related Work

Pretrained Vision Models. The value of using pretrained models (such as ImageNet-pretrained AlexNet (141)) has been demonstrated as far back as 2015 in R-CNN (92), and it has since become standard practice in computer vision. Self-supervised pretraining was proposed as a way to vastly increase the amount of data available for pretraining (68; 310; 182; 187; 43; 100). Unfortunately, this was not very successful, likely because the CNN-based architectures of that time did not have enough capacity to absorb the data. With the introduction of Transformers (254), which have much higher capacity, researchers revisited self-supervised pretraining, and showed that transformer-based masked image reconstruction approaches, such as BEiT (17), MAE (104), SimMIM (287), perform vastly better than their CNN-based counterparts (187). Yet, despite their recent successes, current pretrained vision-only models have had trouble scaling up to the really large datasets, such as LAION (214).

Multi-task Learning and In-context Learning. From the classic one-model-per-task setups, computer vision is slowly moving toward a single model performing multiple different tasks. Various multi-task learning approaches (304; 138; 69; 216; 129) exist but they are typically limited to a fixed, pre-defined number of tasks. More recently, methods inspired by in-context learning in LLMs forgo any notion of tasks and instead let the model infer the task directly from the input prompt. For example, Visual Prompting (18; 261) takes in a task input/output example pair and a query image at test time, concatenates them into a single 2-by-2 image, and uses inpainting to generate the desired output. But, since the inpainting is performed using a variant of MAE (104), the same problems with scaling are inherited by these approaches.

Auto-regressive Visual Models. The idea of using auto-regressive models for synthesizing visual data goes back at least 70 years. Inspired by Shannon's use of N-grams to synthesize language (218; 219), a number of works, starting with Attneave's seminal 1954 paper (8), applied this idea to sequentially synthesizing pixels (85; 192; 77; 119), image patches (76), video frames (212; 200), and motion capture data (7; 139; 144). As deep models became popular, newer works replaced N-grams with RNNs or CNNs for pixel synthesis (251; 250). Most recently, transformer-based autoregessive visual generation methods have been proposed (38; 301; 78; 298), and, combined with language, have demonstrated impressive image synthesis results, e.g. Parti (299).

7.3 Data

"Data! Data! Data! I can't make bricks without clay!" – SHERLOCK HOLMES

The key requirement of any Large Pre-trained Model is that it must be trained on vast

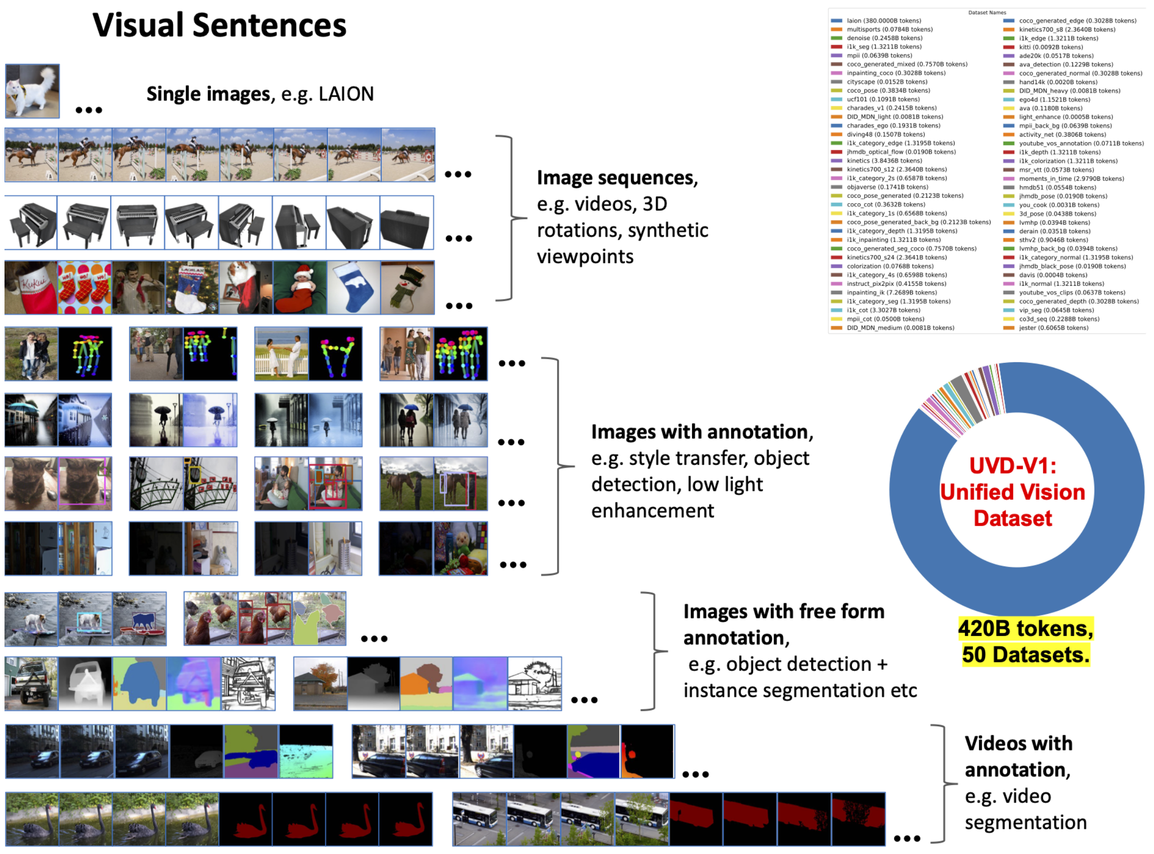

Visual Sentences
Single images, e.g. LAION
Image sequences, e.g. videos, 3D rotations, synthetic viewpoints
Images with annotation, e.g. style transfer, object detection, low light enhancement
Images with free form annotation, e.g. object detection + instance segmentation etc
Videos with annotation, e.g. video segmentation
Dataset Names
laion (380.0000B tokens)
multisports (0.0784B tokens)
denoise (0.2458B tokens)
i1k_seg (1.3211B tokens)
mpii (0.0639B tokens)
coco_generated_mixed (0.7570B tokens)
inpainting_coco (0.3028B tokens)
cityscape (0.0152B tokens)
coco_pose (0.3834B tokens)
ucf101 (0.1091B tokens)
charades_v1 (0.2415B tokens)
DID_MDN_light (0.0081B tokens)
charades_ego (0.1931B tokens)
diving48 (0.1507B tokens)
i1k_category_edge (1.3195B tokens)
jhmdb_optical_flow (0.0190B tokens)
kinetics (3.8436B tokens)
kinetics700_s12 (2.3640B tokens)
i1k_category_2s (0.6587B tokens)
objaverse (0.1741B tokens)
coco_pose_generated (0.2123B tokens)
coco_cot (0.3632B tokens)
i1k_category_1s (0.6568B tokens)
coco_pose_generated_back_bg (0.2123B tokens)
i1k_category_depth (1.3195B tokens)
i1k_inpainting (1.3211B tokens)
coco_generated_seg_coco (0.7570B tokens)
kinetics700_s24 (2.3641B tokens)
colorization (0.0768B tokens)
i1k_category_4s (0.6598B tokens)
instruct_pix2pix (0.4155B tokens)
inpainting_ik (7.2689B tokens)
i1k_category_seg (1.3195B tokens)
i1k_cot (3.3027B tokens)
mpii_cot (0.0500B tokens)
DID_MDN_medium (0.0081B tokens)
coco_generated_edge (0.3028B tokens)
kinetics700_s8 (2.3640B tokens)
i1k_edge (1.3211B tokens)
kitti (0.0092B tokens)
ade20k (0.0517B tokens)
ava_detection (0.1229B tokens)
coco_generated_normal (0.3028B tokens)
hand14k (0.0020B tokens)
DID_MDN_heavy (0.0081B tokens)
ego4d (1.1521B tokens)
ava (0.1180B tokens)
light_enhance (0.0005B tokens)
mpii_back_bg (0.0639B tokens)
activity_net (0.3806B tokens)
youtube_vos_annotation (0.0711B tokens)
i1k_depth (1.3211B tokens)
i1k_colorization (1.3211B tokens)
msr_vtt (0.0573B tokens)
moments_in_time (2.9790B tokens)
hmdb51 (0.0554B tokens)
jhmdb_pose (0.0190B tokens)
you_cook (0.0031B tokens)
3d_pose (0.0438B tokens)
lvmhp (0.0394B tokens)
derain (0.0351B tokens)
sthv2 (0.9046B tokens)
lvmhp_back_bg (0.0394B tokens)
i1k_category_normal (1.3195B tokens)
jhmdb_black_pose (0.0190B tokens)
davis (0.0004B tokens)
i1k_normal (1.3211B tokens)
youtube_vos_clips (0.0637B tokens)
coco_generated_depth (0.3028B tokens)
vip_seg (0.0645B tokens)
co3d_seq (0.2288B tokens)
jester (0.6065B tokens)
UVD-V1: Unified Vision Dataset
420B tokens, 50 Datasets.

Figure 7.1: Visual sentences allow us to format diverse vision data into the unified structure of image sequences.

amounts of data. For language models, very large and very diverse datasets are fairly easy to obtain. For instance, the popular Common Crawl repository (1) contains 250 billion web pages spanning the entire Web, is extremely diverse, and includes "natural demonstrations" like language translations, question answering, etc. In computer vision, we are still very far from having a data source of comparable size and diversity. One of the central contributions of our work is the first step toward curating such a dataset that we call *Unified Vision Dataset v1 (UVDv1)*. To assemble it, we leverage many different sources of visual data: (1) unlabelled images, (2) images with visual annotations, (3) unlabelled videos, (4) videos with visual annotations, and (5) 3D synthetic objects. The unlabeled images, which represent over 80% of our data, capture a huge cross-section of our visual world, and provide the required diversity, at the cost of lower quality. Images with annotations have a much more constrained distribution, but are usually of higher quality. Video data is even more constrained (typically, to human-centric activities), but is an invaluable source of temporal data. Renderings of 3D synthetic objects are the lowest in diversity but can provide valuable hints about the behavior of 3D structures. Importantly, UVDv1 is a purely visual dataset, with no non-visual meta-data (e.g. text) included. All together, UVDv1 contains 1.64 billion images.

Another important difference from Large Language Models is that language data has a natural, unified one-dimensional structure for all the data – a stream of text. Unfortunately, this is not the case for visual data, with different sources all having different structures. In this work we propose *visual sentence* as a unified unit of visual data, which enables us to train scalable models from a diverse set sources. A visual sentence is simply a sequence containing one or more images followed by an end-of-sentence (EOS) token. Figure 7.1 shows how the various data sources are partitioned into visual sentences. In particular:

Single images. A single image itself represents the simplest form of a visual sentence – {image, EOS}. We use the filtered subset of 1.49 billion images (263) from the LAION 5B (215) dataset. This is by far the largest part of our data, comprising 88.5%.

Image sequences. A sequence of images is a natural form of visual sentence. We create such sequences by sourcing video data from a wide range of existing datasets (226; 190; 142; 28; 175; 176; 204; 222; 98; 63; 32; 288; 289; 170; 148; 147; 221; 177; 99). Visual sentences of 16 frames are formed by randomly sampling the videos at three different strides (10, 20, and 30). In addition, we utilize synthetic 3D objects from the Objaverse Dataset (64) to generate object-centric multiview sequences for a variety of objects. For each object, we sample one radius length between the object center and the camera from 1.5 to 2.2, and sample one constant elevation from -45 degrees to 45 degrees, then traverse different views of the object by changing the azimuth with a step length of 15 degrees and render 24 views. We rendered 42000 such sequences in total for training and 8000 for testing. Finally, we can also represent images belonging to the same semantic category as being (part of) a sequence. We use categories from ImageNet, concatenating together groups of images (2,4,8, or 16) from the same category into a 16-image long visual sentences.

Images with annotations. To handle different types of image annotations in a uniform way, we choose to represent all annotations as images. Some data types, e.g. semantic segmentation maps (319), edge maps (227), depth (199) and normal images (9), are already represented this way. For others we apply tailored methods for each specific annotation type: 1) Object Detection: We create annotations by overlaying a color-coded bounding box around each object, following the methodology in (36); 2) Human Pose: Human skeletons are rendered in pixel space, adhering to the OpenPose format, utilizing MMPose (53); 3) Depth Estimation, Surface Normal, and Edge Detection: given ImageNet and COCO images, we generate annotations in line with the protocols from (158). 3) Style Transfer (23), De-rain (308), De-noise (255), Low Light Enhancement (265), and Stereo Datasets (86): These are all represented as image pairs (e.g. input/output). 4) Colorization: We convert ImageNet images to greyscale, producing image pairs. 5) Inpainting: The process involves randomly adding black-colored boxes in images to simulate corruption, resulting in image pairs. For all the above annotation types, we can create visual sentences by concatenating 8 image pairs of the same annotation type into a 16-image visual sentence. For datasets containing k different annotations for the same image we use a different approach: for each set of $1 + k$ images (input plus k annotations), we randomly select m elements, where $m \leq n+1 \leq 16$. These m-tuples are then concatenated to form visual sequences.

Image sequences with annotations. When converting annotated video data (VIPSeg (172), Hand14K (81), AVA (177), JHMDB (130)) to visual sentences, we apply two complementary strategies. The first is similar to how we treat image data with paired annotations: each visual sentence is constructed by concatenating frames with their annotations – {frame1,annot1,frame2,annot2,...}. The second method involves grouping multiple frames followed by their corresponding annotations – {frame1,frame2,annot1,annot2,...}.

We present a detailed summary of all the data sources, annotation type and data statistics of UVDv1 in the Appendix.

7.4 Approach

In this section, we describe the design of our autoregressive Large Vision Model. Unlike text data, which naturally exhibits discrete sequential structure, it is not straightforward to model image pixels in visual sentences. In this work, we take a two-stage approach: 1) train a large visual tokenizer (which operates on individual images) to convert each image into a sequence of visual tokens; 2) train an autoregressive transformer model on visual sentences, each represented as a sequence of tokens. We summarize our approach in Figure 7.2.

7.4.1 Image Tokenization

While the visual sentences exhibit a sequence structure between consecutive images, we don't have such natural sequence structure within an image. Therefore, in order to apply a transformer model to images, prior works typically do one of the following: either divide the image into patches in scan-line order, and treat that as a sequence (71), or use a pre-trained image tokenizer, such as VQVAE (252) or VQGAN (78), to cluster image features into a grid of discrete tokens, which, again, are turned into a sequence in scan-line order. We adopt the latter approach since the discrete categorical output from a model naturally forms a probabilistic distribution that one can easily sample from, enabling flexible conditional generation of new images within a visual sentence.

Specifically, we employ semantic tokens generated by a VQGAN model, a concept introduced by Esser et al (78). This framework consists of an encoding and a decoding mechanism, featuring a quantization layer that assigns input images to a sequence of discrete tokens from an established codebook. Our encoders and decoders are constructed purely with convolutional layers. The encoder is equipped with several downsampling modules to contract the spatial dimension of the input, whereas the decoder is fitted with an equivalent series of upsampling modules to restore the image to its initial size. For a given image, our VQGAN tokenizer produces 256 discrete tokens.

It is important to note that our tokenizer operates on individual images independently, rather than on the entire visual sentence at once. This independence allows us to decouple the tokenizer training from the downstream Transformer model so that the tokenizer can be trained on a dataset of single images without having to consider the distribution of visual sentences.

Implementation Details: We adopt an off-the-shelf VQGAN architecture from (33). We follow the exact configuration in (33), which uses a downsampling factor of $f = 16$ and codebook size 8192. This means that for an image of size 256×256, our VQGAN tokenizer produces $16 \times 16 = 256$ tokens where each can take 8192 different values. We found that using the results of an ImageNet pre-trained tokenizer did not generalize well beyond ImageNet images. Therefore, we train our own tokenizer on a 1.5B subset of the LAION 5B dataset (215).

7.4.2 Sequence Modeling of Visual Sentences

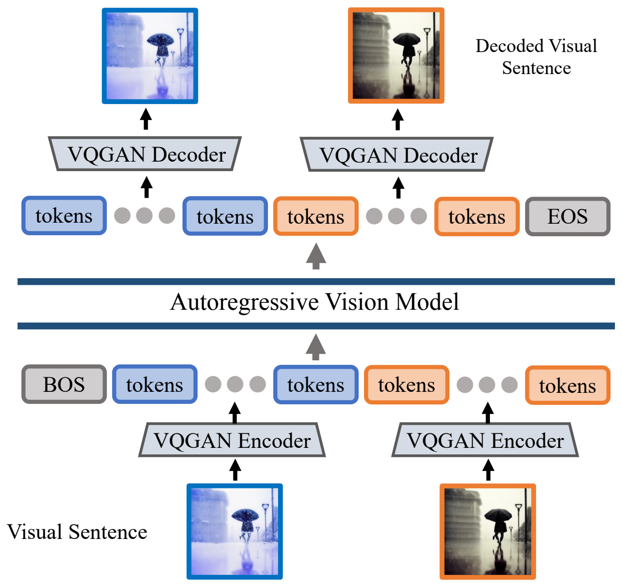


Figure 7.2: Architecture of LVM. We first convert individual images from a visual sentence into discrete tokens using a VQGAN encoder. The resulting tokens from all images are then concatenated into a 1D sequence, and fed into an autoregressive Transformer model to predict the next token in the sequence. The predicted visual tokens are decoded into images using the VQGAN decoder.

After converting images into discrete tokens with VQGAN, we treat our visual sentence as a unified sequence by concatenating the discrete tokens from multiple images into a 1D sequence. Importantly, all visual sentences are treated equally – we do not make use of any special tokens to indicate particular tasks or formats. We train a causal Transformer model with the next token prediction objective using a cross-entropy loss, similar to the standard approach for language models (25). Training the model the same way on all visual sentences enables the model to infer the relation between images from context instead of from task- or format-specific tokens. This gives the model an opportunity to generalize to other, unseen visual sentence structures.

Implementation Details: After tokenizing each image in a visual sentence into 256 tokens, we concatenate them to form a 1D sequence of tokens. On top of the sequences of visual tokens, our Transformer model is virtually the same as an autoregressive language model, so we adopt the Transformer architecture of LLaMA (246), a popular open-source language model with widely available implementations. We use a context length of 4096 tokens, which can fit 16 images under our VQGAN tokenizer. Similar to language models, we add a [BOS] (begin of sentence) token to the beginning of each visual sentence and an [EOS] (end of sentence) token to the end, and use sequence concatenation (51) during training time to improve efficiency. We train our model on our entire UVDv1 dataset (420 billion tokens) using one epoch (simple epoch training is standard in language models to avoid potential overfitting). We train 4 models with different numbers of parameters: 300 million, 600 million, 1 billion and 3 billion, following the same training configurations. We provide the detailed training hyperparameters in Appendix **??**.

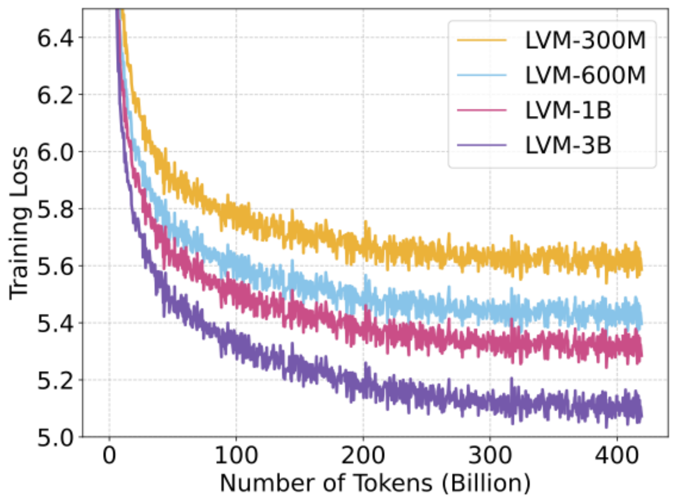


Figure 7.3: Training loss for the 300M, 600M, 1B, and 3B models. All models are trained on 420B tokens, which correspond to 1.64B images. The training scales well with model sizes.

7.4.3 Inference by Visual Prompting

Since the autoregressive Transformer in our model outputs a probability distribution of the next token conditioned on previous tokens, we can easily sample from this distribution to generate new visual tokens that complete a visual sentence. To use the model for downstream tasks, one can construct a partial visual sentence that defines a task at test time, and apply the model to generate the output. This is similar to in-context learning in language models (26) or visual prompting in computer vision (119; 18).

7.5 Experimental Results and Analysis

In this section, we evaluate the scaling abilities of our trained model, as well as its ability to understand and answer a range of diverse prompted tasks.

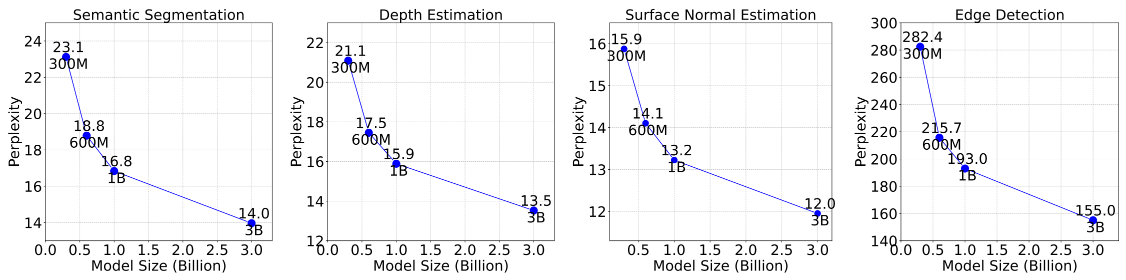


Figure 7.4: Larger LVMs perform better on downstream tasks. We evaluate LVM models of varying sizes on 4 different downstream tasks, following the 5 shot setting on the ImageNet validation set and report the perplexity. We find that perplexity decreases with larger models across all tasks, indicating the strong scalability of LVM.

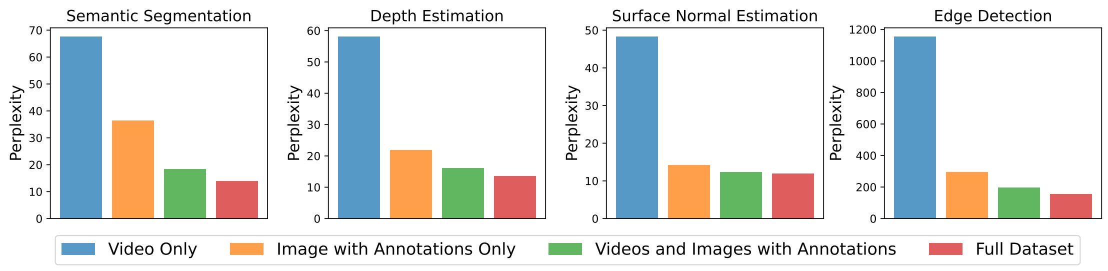


Figure 7.5: We evaluate the perplexity of 4 models trained on different subsets of our datasets on multiple tasks using the ImageNet validation set. All models are 3B parameters and all evaluations are conducted in the 5-shot setting. We can see that the model benefits from each single images, videos and annotations, demonstrating the importance of our training dataset diversity.

7.5.1 Scalability

We investigate the scaling behavior of our model in terms of the training loss and downstream task performance as we increase the model size as well as the number of tokens seen during training.

Training loss. We first inspect the training loss of LVM with different parameter sizes, which we present in Figure 7.3. Since all our models are trained for only one epoch on the dataset, the model sees a given data sample just once, and therefore the training loss

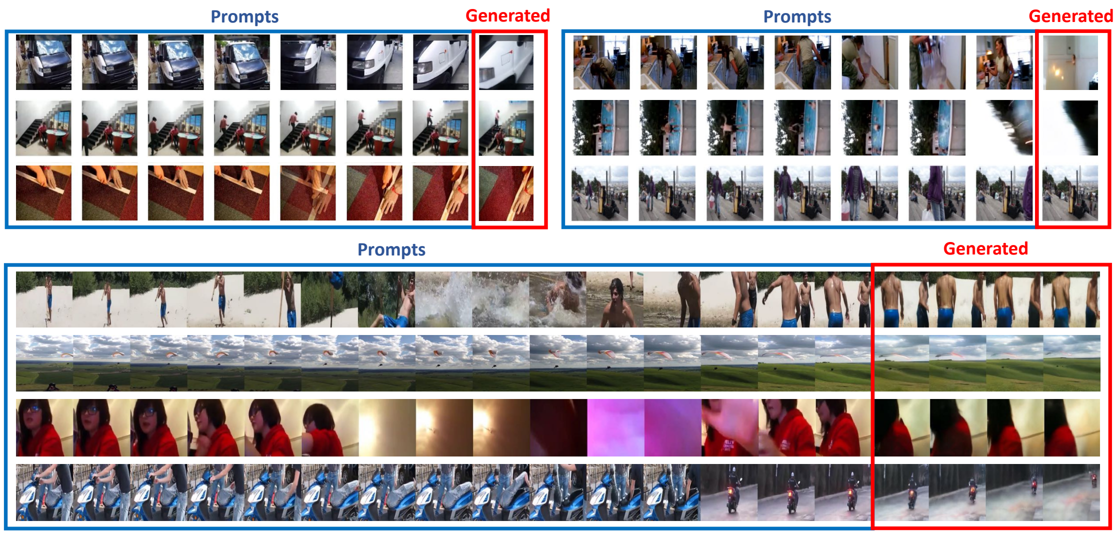


Figure 7.6: Frame predictions. LVM predicts the next frame (marked in red) given previous video frames as prompt. The results reveal that the LVM can predict the video frames while considering dynamic objects and camera motion.

at any point during training is very similar to the validation loss. One can observe that as training progresses: 1) the training loss (perplexity) of the models, regardless of their size, continues to decrease; 2) as we increase the size of the model (parameter count), the loss decreases faster. These observations indicate that LVM shows strong scalability behavior with both larger models and more data.

Scalability on downstream benchmarks. While the LVM overall loss scales well during training, there is no guarantee that the better overall model would also perform better on a given specific downstream task. Therefore, we evaluate different sizes of models on 4 downstream tasks: semantic segmentation, depth estimation, surface normal estimation, and edge detection. We evaluate these tasks on the ImageNet validation set and generate all the annotations using the corresponding method described in Sec. 3. For each task, we give 5 pairs consisting of the inputs and corresponding ground-truth annotations as well as the query image as input prompt and evaluate the perplexity of the ground-truth annotation under our model's prediction of the next 256 tokens (one image). We report the results in Figure 7.4. We see that larger models indeed attain lower perplexity across all tasks, showcasing that our scalable overall performance does transfer to a range of downstream tasks.

Dataset ablation. While LVM attains better performance with larger models and more data, it is natural to ask whether each data component we collect in UVDv1 helps. To answer this question, we conduct an ablation study on our dataset by training several 3B models on subsets of our dataset, and compare their performances on downstream tasks. We use the same 4 downstream tasks and settings as before and present the results in Figure 7.5. We observe that each data component contributes positively to the downstream tasks. LVM not only benefits from larger data, but also improves with more diversity in the dataset, which includes both annotated and unsupervised image and video data.

7.5.2 Sequential Prompting

We begin with the most intuitive and straightforward approach to visually prompt the LVM: sequential reasoning. Here the prompt construction is very simple: we present the model with a sequence of 7 images and ask it to predict the next image (256 tokens).

Video frame prediction. The most direct task for sequential prompting is video prediction. Figure 7.6 presents several next frame prediction examples, prompted by sequences from the Kinetics-700 validation set. At the top, 7 frame prompts (blue border) are followed by the predicted frame (red border). We observe a certain degree of inferential ability regarding spatial positioning, viewpoint and object understanding. Perplexity of prediction

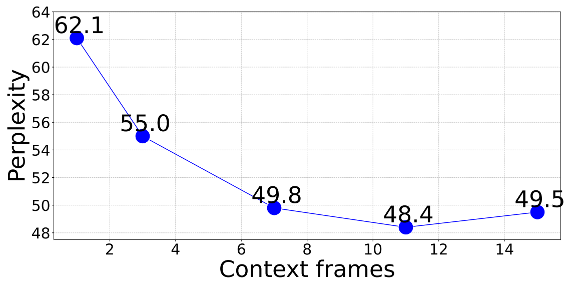


Figure 7.7: Longer context helps model understand better.

on Kinetics val set is 49.8. The last 4 rows show predictions with longer context (15 frames) and a longer prediction (4 frames).

Rotation and Category prediction. The same type of simple sequential prompting can be used in other ways as well. For example, Figure **??** shows how prompting the model with a sequence of 3D rotations of a synthetic object around an arbitrary axis allows it to predict further rotation. Or we can think of a list of items of a given category as a sequence and predict other ideas in that same category, as shown in Figure **??**. Note that, while the system was trained on groups of images from the same ImageNet category, here the prompt consists of sketches, which have not been seen in any annotated data.

Context length analysis. Next we ask how much temporal context is required to accurately predict the subsequent frame? We assessed the model's frame generation perplexity when prompted with a context of varying lengths (1 to 15 frames). As Figure 7.7 shows, on the Kinetics-700 val set, we see a clear improvement in perplexity from 1 to 11 frames after which it stabilizes (from 62.1 $\rightarrow$ 48.4).

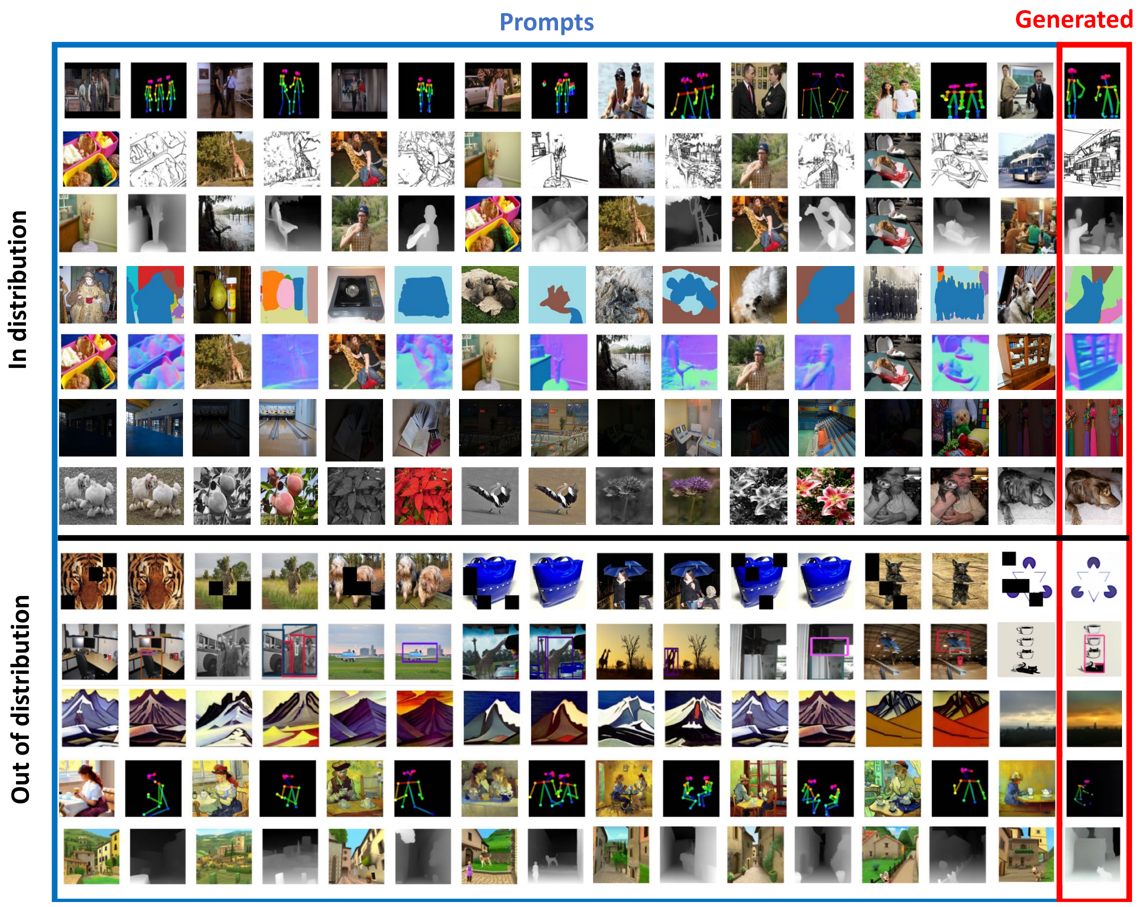


Figure 7.8: In and out of distribution prompting examples. Every row is a prompt that contains a sequence of images interleaved with annotations, followed by a query. The last image is predicted by the model (marked in red). The last 5 rows show examples where the query image is out of distribution (painting, sketch, etc) for the task it was trained for.

7.5.3 Analogy Prompting

Our study progresses by evaluating a more complex prompting structure, which we call ‘Analogy Prompting’. This method challenges the model to comprehend analogies of arbitrary length and complexity, thereby testing its advanced interpretative abilities.

Qualitative Results. Figure 7.8 shows a sampling of qualitative results with analogy

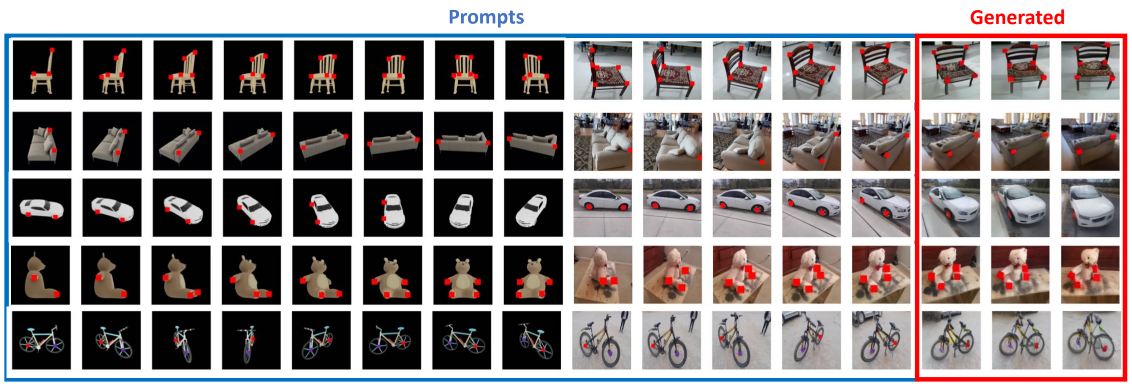


Figure 7.9: Task Compositionality. Examples of prompts that combine two different tasks – object rotation and keypoint tracking.

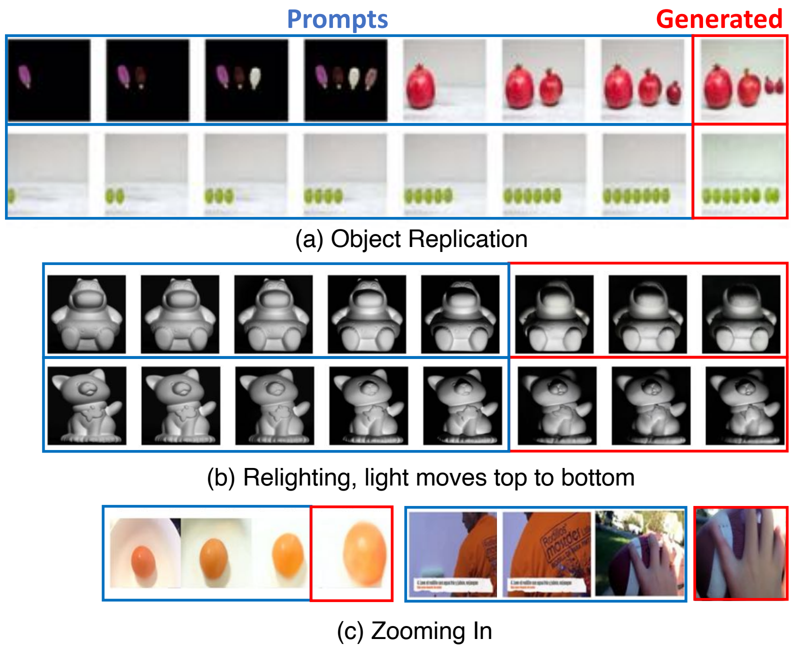


Figure 7.10: Miscellaneous Prompting. A variety of simple vision tasks, such as object replication (top), relighting (middle), and zooming in (bottom), can be simply specified via a suitably chosen visual sentence prompt that expresses the task to the LVM.

prompting on a number of tasks. The prompts consist of a sequence of 14 images giving examples of various tasks, followed by a 15th query image. Given each prompt, the next image predicted is the result. The top part of the figure shows several example prompts defining tasks that were part of the training set (but these actual images were never seen at training). The bottom part of the figure demonstrates generalization to tasks never shown at training. See the Appendix for many more qualitative examples.

Unseen Tasks and Dataset. We present the results for keypoint detection on Pascal 3D+ (275), evaluated using the standard Percentage of Correct Keypoints (PCK) metric with a of threshold 0.1. Remarkably, LVM achieves a PCK of 81.2 without training on this dataset, demonstrating impressive generalization capabilities. In comparison, we show some existing task-specific model: StackedHourglass (180) scores 68.0 PCK, MSS-Net (133) achieves 68.9 PCK, and StarMap (321) registers 78.6 PCK.

Comparison with Visual Prompting. The closest approach to ours that also allows for defining arbitrary tasks is Visual Prompting (18). In Table 7.1, we compare various visual prompting models on few-shot segmentation, object detection, and colorization tasks. Note that our sequential LVM beats previous approaches on almost all tasks.

Task Compositing. Figure 7.9 demonstrates compositing several tasks together within a single prompt. Here, we demonstrate the rotation task together with the novel keypoint correspondence task and ask the model to continue the pattern. The model is able to successfully combine these two at test ti me, demonstrating some degree of compositionality.

7.5.4 Miscellaneous Prompts

Here we try to see how far we can push our model by offering it various prompts it has not seen before. Figure 7.10 shows a few such prompts that happened to work reasonably well. Figure 7.11 shows some prompts which are not easily describable by words – these are the

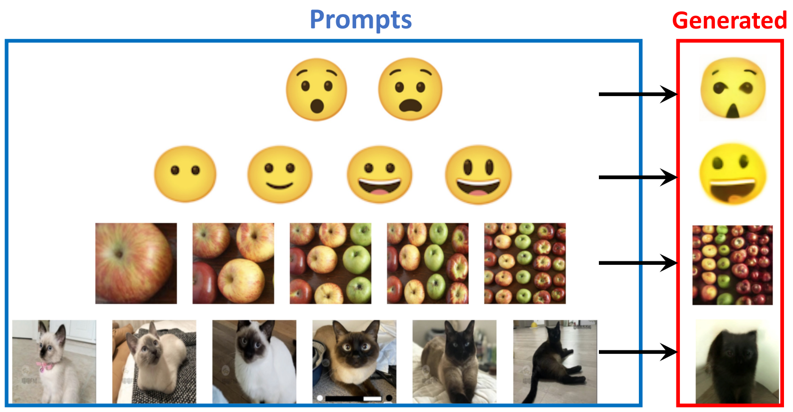


Figure 7.11: What comes next? Tasks that are not always easily describable in language.

Model	Foreground Segmentation ↑				Single Object Detection ↑				Colorization ↓	
	Split 0	Split 1	Split 2	Split 3	Split 1	Split 2	Split 3	Split 4	MSE	LPIPS
MAE (IN-1k)	1.92	6.76	3.85	4.57	1.37	1.98	1.62	1.62	1.13	0.87
MAE-VQGAN (IN-1k)	2.22	7.07	5.48	6.28	3.34	3.21	2.80	2.80	3.31	0.75
MAE (CVF)	17.42	25.70	18.64	16.53	5.49	4.98	5.24	5.84	**0.43**	0.55
MAE-VQGAN (CVF)	27.83	30.44	26.15	24.25	24.19	25.20	25.36	25.23	0.67	**0.40**
Ours	**48.94**	**51.29**	**47.66**	**50.82**	**48.25**	**49.60**	**50.08**	**48.92**	0.51	0.46

Table 7.1: Comparison with Visual Prompting (18). For Foreground Segmentation and Single Object Detection, we report the *mIOU* score. For Colorization, we report the *MSE* and *LPIPS*.

type of tasks where LVMs might eventually outshine LLMs.

In Figure 7.13, we show initial qualitative results on a typical visual reasoning question as found on non-verbal human IQ tests (Raven's Progressive Matrices (201)). With considerable squinting, one could imagine the LVM having a latent ability for grasping abstract visual patterns and applying the grasped pattern to extrapolate the shown visual sequence. This exciting result warrants further study. Therefore we performed perplexity analysis on 10 classic Raven 5-way multiple-choice Matrices, choosing the answer with lowest perplexity under our model. It shows that our results are better than chance. And in order

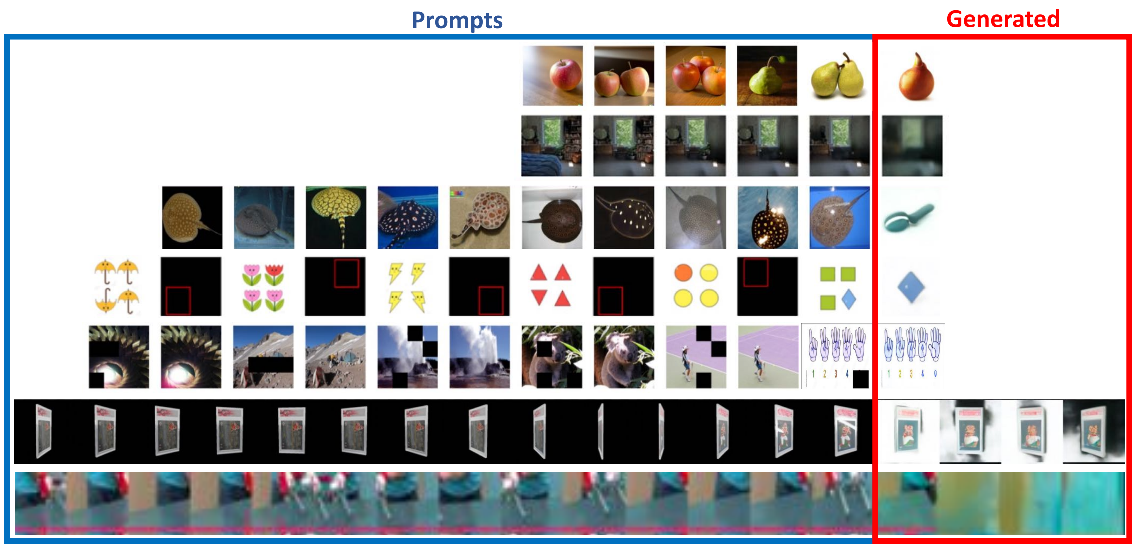


Figure 7.12: Failure Cases. This figure illustrates seven examples of failure cases: (1) Task confusion: the model interprets the counting task as style transfer, resulting in a pear-shaped apple. (2) Task entanglement: the model perceives the prompt as high frequencies from the entire image, rather than just from some parts of image (objects). (3) Wrong instance: the model correctly identifies rotation but mistakenly generates a brush instead of a manta. (4) Outlier detection versus generation: rather than detecting an outlier, the model directly generates it. (5) Digits: the model fails to represents digit sequences. (6) Tokenizer: poor performance with some out-of-distribution synthetic data generation (wrong background). (7) Sequence degeneration: sometimes frame predictions go astray.

to further understand this process, we also generated 900 new Raven-like synthetic tests to study individual factors: color, shape, and size following (18). As shown in Table 7.2, LVM has shown better understanding in all three aspects compared to chance.

	Raven IQ matrices	shape	color	size
chance	20%	33%	33%	50%
Ours	**30%**	**45%**	**42%**	**94%**

Table 7.2: Raven's Progressive Matrices Perplexity and Synthetic Resaoning Tasks.

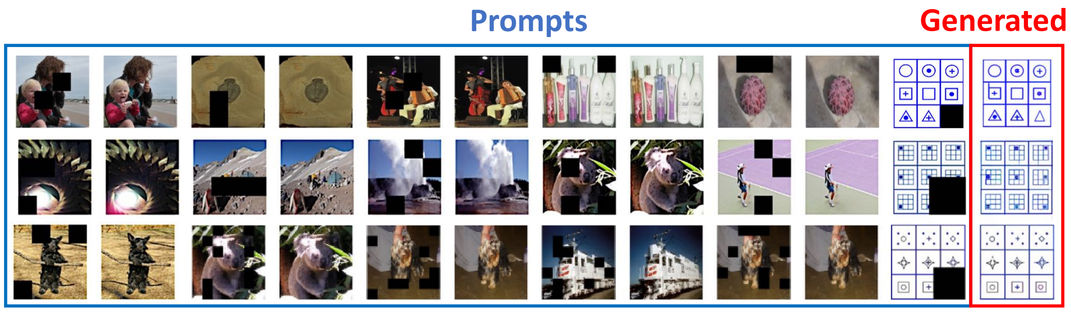


Figure 7.13: "Sparks of AGI?" 🤨 We prompt LVM with a masked reasoning visual sentence to infer the solution for non-verbal reasoning questions which are prevalent in IQ tests (masked image, second from the right). We find that the model often infers and applies the abstract visual pattern correctly. So, we graciously hand over to you, our gentle reader, the task of pondering whether our modest LVM also exhibits the much-vaunted 'Sparks of AGI'.

7.6 Limitations

Figure 7.12 shows some typical failure cases of the current model. One common element, the use of visual prompt to define a task is often under-constrained (more so than in language, since images are very high-dimensional), or the requested task might be beyond the capabilities of the current system. Other, more mundane failures involve issues with the tokenizer and lack of high-quality video training data.

Limited computing resources placed severe constraints that prevented us from exploring a range of intriguing problems, including the impact of different data sets and detailed ablation studies. It is important to note that, despite this being one of the biggest vision models to date, it is still rather small in comparison with modern Large Language Models. Therefore, the question of emergence and true generalization in Large Vision Models remains wide open and ripe for further study.

Chapter 8

Future Work

The key goals of my future research are achieving higher levels of general intelligence directly from raw sensory experience, akin to how humans learn. This potentially involves continuous adaptation for better generalization in dynamic real-world settings over time with multi-modal input.

8.1 Life-long Learning

An important direction for future work is enabling life-long learning for vision models (46). Traditional training assumes static datasets and finite training periods (105; 31), limiting adaptability as environments and data distributions evolve over time. Future research should focus on life-long learning techniques that allow models to incrementally update their knowledge from new data streams while retaining prior learned skills.

Strategies are needed to mitigate catastrophic forgetting, dynamically adapt tokens and prioritize important input samples for further training via meta-learning and model introspection. While the visual sentence model from this thesis was trained for just one epoch, the future vision is for models to continually learn and improve over their lifetimes through constant interaction with a stream of real-world data.

8.2 Multi-modal Learning

In contrast to existing work where language has dominated multi-modal training (156; 323), we are interested in investigating whether any intelligent behavior can be learned directly from sensory data alone. This line of inquiry has significant potential for embodied AI systems that must operate in real-world environments based solely on raw sensory inputs like vision, audio, and haptics(48; 293).

Our visual sentence framework provides a promising direction by representing all visual modalities, including video, audio, depth, and more, as sequential tokens that can be consumed by a single multi-modal encoder. By training on next-token prediction across these unified "visual sentences", the model learns multi-modal representations grounded in the low-level signals rather than relying on symbolic language representations.

Key objectives of this multi-modal learning approach include developing better perception models that can fuse multiple sensor streams, enabling tight sensor-action loops for control in embodied agents, and ultimately acquiring broad general intelligence from raw sensory experience akin to how humans learn.

Bibliography

[1] Common crawl repository. https://commoncrawl.org/.

[2] Lvis challenge 2021. https://www.lvisdataset.org/challenge_2021. Accessed: 2021-11-16.

[3] B. Alexe, T. Deselaers, and V. Ferrari. Measuring the objectness of image windows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2189–2202, 2012.

[4] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[5] Maksym Andriushchenko, Francesco Croce, Nicolas Flammarion, and Matthias Hein. Square attack: a query-efficient black-box adversarial attack via random search. In *ECCV*, 2020.

[6] Pablo Arbeláez, Jordi Pont-Tuset, Jonathan T Barron, Ferran Marques, and Jitendra Malik. Multiscale combinatorial grouping. In *CVPR*, 2014.

[7] Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Trans. Graph.*, 21(3):483–490, jul 2002.

[8] Fred Attneave. Some informational aspects of visual perception. *Psychological*

review, 61(3):183, 1954.

[9] Gwangbin Bae, Ignas Budvytis, and Roberto Cipolla. Estimating and exploiting the aleatoric uncertainty in surface normal estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13137–13146, 2021.

[10] Yutong Bai, Xinlei Chen, Alexander Kirillov, Alan Yuille, and Alexander C Berg. Point-level region contrast for object detection pre-training. In *CVPR*, 2022.

[11] Yutong Bai, Haoqi Fan, Ishan Misra, Ganesh Venkatesh, Yongyi Lu, Yuyin Zhou, Qihang Yu, Vikas Chandra, and Alan Yuille. Can temporal information help with contrastive self-supervised learning? *arXiv preprint arXiv:2011.13046*, 2020.

[12] Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. Sequential modeling enables scalable learning for large vision models. *arXiv preprint arXiv:2312.00785*, 2023.

[13] Yutong Bai, Qing Liu, Lingxi Xie, Weichao Qiu, Yan Zheng, and Alan L Yuille. Semantic part detection via matching: Learning to generalize to novel viewpoints from limited training data. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 7535–7545, 2019.

[14] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are transformers more robust than cnns? *Advances in Neural Information Processing Systems*, 34:26831–26843, 2021.

[15] Yutong Bai, Angtian Wang, Adam Kortylewski, and Alan Yuille. Coke: Contrastive learning for robust keypoint detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 65–74, 2023.

[16] Yutong Bai, Zeyu Wang, Junfei Xiao, Chen Wei, Huiyu Wang, Alan L Yuille, Yuyin Zhou, and Cihang Xie. Masked autoencoders enable efficient knowledge

distillers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24256–24265, 2023.

[17] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. In *ICLR*, 2021.

[18] Amir Bar, Yossi Gandelsman, Trevor Darrell, Amir Globerson, and Alexei Efros. Visual prompting via image inpainting. *Advances in Neural Information Processing Systems*, 35:25005–25017, 2022.

[19] Vasileios Belagiannis and Andrew Zisserman. Recurrent human pose estimation. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 468–475. IEEE, 2017.

[20] Irwan Bello, William Fedus, Xianzhi Du, Ekin D Cubuk, Aravind Srinivas, Tsung-Yi Lin, Jonathon Shlens, and Barret Zoph. Revisiting resnets: Improved training and scaling strategies. In *NeurIPS*, 2021.

[21] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *ICCV*, 2019.

[22] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. *arXiv preprint arXiv:2103.14586*, 2021.

[23] Tim Brooks, Aleksander Holynski, and Alexei A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023.

[24] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris

Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. 2020.

[25] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[26] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[27] Adrian Bulat and Georgios Tzimiropoulos. Human pose estimation via convolutional part heatmap regression. In *European Conference on Computer Vision*, pages 717–732. Springer, 2016.

[28] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 961–970, 2015.

[29] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE Conference on*

Computer Vision and Pattern Recognition, pages 7291–7299, 2017.

[30] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *ICCV*, 2021.

[31] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *CVPR*, 2021.

[32] Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. *arXiv preprint arXiv:1907.06987*, 2019.

[33] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. *arXiv preprint arXiv:2202.04200*, 2022.

[34] Neelima Chavali, Harsh Agrawal, Aroma Mahendru, and Dhruv Batra. Object-proposal evaluation protocol is 'gameable'. In *CVPR*, 2016.

[35] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. Large scale online learning of image similarity through ranking. In *Iberian Conference on Pattern Recognition and Image Analysis*, pages 11–14. Springer, 2009.

[36] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tianheng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[37] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and

fully connected crfs. In *International Conference on Learning Representations*, 2016.

[38] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR, 2020.

[39] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[40] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[41] Xianing Chen, Qiong Cao, Yujie Zhong, Jing Zhang, Shenghua Gao, and Dacheng Tao. Dearkd: Data-efficient early knowledge distillation for vision transformers. In *CVPR*, 2022.

[42] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[43] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *arXiv preprint arXiv:2011.10566*, 2020.

[44] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *CVPR*, 2021.

[45] Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pretraining or strong data augmentations. *arXiv preprint arXiv:2106.01548*, 2021.

[46] Xinlei Chen, Abhinav Shrivastava, and Abhinav Gupta. Neil: Extracting visual knowledge from web data. In *Proceedings of the IEEE international conference on*

computer vision, pages 1409–1416, 2013.

[47] Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *ICCV*, 2021.

[48] Ziyang Chen, Shengyi Qian, and Andrew Owens. Sound localization from motion: Jointly learning sound direction and camera rotation. *International Conference on Computer Vision (ICCV)*, 2023.

[49] Bowen Cheng, Omkar Parkhi, and Alexander Kirillov. Pointly-supervised instance segmentation. *arXiv preprint arXiv:2104.06404*, 2021.

[50] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In *CVPR*, 2019.

[51] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[52] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. *arXiv preprint arXiv:1606.03558*, 2016.

[53] MMPose Contributors. Openmmlab pose estimation toolbox and benchmark. https://github.com/open-mmlab/mmpose, 2020.

[54] 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 *CVPR*, pages 3213–3223, 2016.

[55] Francesco Croce and Matthias Hein. Minimally distorted adversarial examples with a fast adaptive boundary attack. In *ICML*, 2020.

[56] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *ICML*, 2020.

[57] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[58] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical data augmentation with no separate search. In *NeurIPS*, 2020.

[59] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. 2013.

[60] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks. In *Advances in Neural Information Processing Systems*, 2016.

[61] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *ACL*, 2019.

[62] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition*, 2005.

[63] Pradipto Das, Chenliang Xu, Richard F Doell, and Jason J Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2634–2641, 2013.

[64] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, pages 13142–13153, 2023.

[65] J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition*, 2009.

[66] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[67] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[68] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[69] Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *ICCV*, 2017.

[70] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *International Conference on Machine Learning*, 2014.

[71] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

[72] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[73] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision*, 2015.

[74] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *NeurIPS*, 2014.

[75] Debidatta Dwibedi, Yusuf Aytar, Jonathan Tompson, Pierre Sermanet, and Andrew Zisserman. With a little help from my friends: Nearest-neighbor contrastive learning of visual representations. *arXiv preprint arXiv:2104.14548*, 2021.

[76] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *Proceedings of SIGGRAPH 2001*, 2001.

[77] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, Corfu, Greece, September 1999.

[78] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[79] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *IJCV*, 88(2):303–338, 2010.

[80] 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*, 88(2):303–338, 2010.

[81] Alireza Fathi, Xiaofeng Ren, and James M Rehg. Learning to recognize objects in

egocentric activities. In *CVPR 2011*, pages 3281–3288. IEEE, 2011.

[82] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010.

[83] Peter R Florence, Lucas Manuelli, and Russ Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. *arXiv preprint arXiv:1806.08756*, 2018.

[84] Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In *ICML*, 2018.

[85] David Donovan Garber. *Computational Models for Texture Analysis and Texture Synthesis*. PhD thesis, University of Southern California, Image Processing Institute, 1981.

[86] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[87] 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. In *ICLR*, 2018.

[88] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *CVPR*, 2021.

[89] R. Girshick. Fast r-cnn. In *Computer Vision and Pattern Recognition*, 2015.

[90] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and*

Pattern Recognition, 2014.

[91] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[92] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Region-based convolutional networks for accurate object detection and segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 38(1):142–158, 2015.

[93] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and continuous pose estimation. *Image and Vision Computing*, 30(12):923–933, 2012.

[94] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *Computer Vision and Pattern Recognition*, 2017.

[95] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015.

[96] Nicolas Gourier, Daniela Hall, and James L Crowley. Facial features detection robust to pose, illumination and identity. In *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, volume 1, pages 617–622. IEEE, 2004.

[97] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[98] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter Yianilos, Moritz Mueller-Freitag, et al. The" something something" video database for learning

and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[99] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[100] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. 2020.

[101] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE, 2006.

[102] B. Ham, M. Cho, C. Schmid, and J. Ponce. Proposal flow. In *Computer Vision and Pattern Recognition*, 2016.

[103] K. Han, R. S. Rezende, B. Ham, K. Y. K. Wong, M. Cho, C. Schmid, and J. Ponce. Scnet: Learning semantic correspondence. In *International Conference on Computer Vision*, 2017.

[104] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[105] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B.

Girshick. Masked autoencoders are scalable vision learners. *CoRR*, abs/2111.06377, 2021.

[106] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. *arXiv preprint arXiv:1911.05722*, 2019.

[107] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[108] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[109] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[110] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

[111] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[112] Olivier J Hénaff, Skanda Koppula, Jean-Baptiste Alayrac, Aaron van den Oord, Oriol Vinyals, and João Carreira. Efficient visual pretraining with contrastive detection. In *ICCV*, 2021.

[113] Olivier J Hénaff, Aravind Srinivas, Jeffrey De Fauw, Ali Razavi, Carl Doersch, SM Eslami, and Aaron van den Oord. Data-efficient image recognition with contrastive predictive coding. *arXiv preprint arXiv:1905.09272*, 2019.

[114] Dan Hendrycks and Thomas G Dietterich. Benchmarking neural network robustness to common corruptions and surface variations. In *ICLR*, 2018.

[115] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[116] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *CVPR*, 2021.

[117] Byeongho Heo, Jeesoo Kim, Sangdoo Yun, Hyojin Park, Nojun Kwak, and Jin Young Choi. A comprehensive overhaul of feature distillation. In *CVPR*, 2019.

[118] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge transfer via distillation of activation boundaries formed by hidden neurons. In *AAAI*, 2019.

[119] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340, 2001.

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

[121] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[122] J. H. Hosang, R. Benenson, and B. Schiele. Learning non-maximum suppression. In *International Conference on Computer Vision*, 2017.

[123] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):504–511, 2013.

[124] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In *Computer Vision and Pattern Recognition*, 2018.

[125] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. In *Computer Vision and Pattern Recognition*, 2017.

[126] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *Computer Vision and Pattern Recognition*, 2017.

[127] Eldar Insafutdinov, Leonid Pishchulin, Bjoern Andres, Mykhaylo Andriluka, and Bernt Schiele. Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In *European Conference on Computer Vision*, pages 34–50. Springer, 2016.

[128] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015.

[129] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

[130] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *International Conf. on Computer Vision (ICCV)*, pages 3192–3199, Dec. 2013.

[131] B. Jiang, R. Luo, J. Mao, T. Xiao, and Y. Jiang. Acquisition of localization confidence for accurate object detection. In *European Conference on Computer Vision*, 2018.

[132] Angjoo Kanazawa, Shubham Tulsiani, Alexei A Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 371–386, 2018.

[133] Lipeng Ke, Ming-Ching Chang, Honggang Qi, and Siwei Lyu. Multi-scale structure-aware network for human pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 713–728, 2018.

[134] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362*, 2020.

[135] Jangho Kim, SeongUk Park, and Nojun Kwak. Paraphrasing complex network: Network compression via factor transfer. *NeurIPS*, 2018.

[136] S. Kim, D. Min, B. Ham, S. Jeon, S. Lin, and K. Sohn. Fcss: Fully convolutional self-similarity for dense semantic correspondence. In *Computer Vision and Pattern Recognition*, 2017.

[137] Alexander Kirillov, Yuxin Wu, Kaiming He, and Ross Girshick. Pointrend: Image segmentation as rendering. In *CVPR*, 2020.

[138] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6129–6138, 2017.

[139] Lucas Kovar, Michael Gleicher, and Frederic Pighin. Motion Graphs. In *Proceedings of SIGGRAPH '02*, San Antonio, TX, 2002.

[140] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[141] A. Krizhevsky, I. Sutskever, and G. E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[142] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *2011 International conference on computer vision*, pages 2556–2563. IEEE, 2011.

[143] Y. LeCun, Y. Bengio, and G. E. Hinton. Deep learning. *Nature*, 521(7553):436, 2015.

[144] Jehee Lee, Jinxiang Chai, Paul Reitsma, Jessica K. Hodgins, and Nancy Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics (TOG)*, 21(3):491 – 500, July 2002.

[145] C. Li, M. Z. Zia, Q. H. Tran, X. Yu, G. D. Hager, and M. Chandraker. Deep supervision with shape concepts for occlusion-aware 3d object parsing. In *Computer Vision and Pattern Recognition*, 2017.

[146] Sijin Li and Antoni B Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, pages 332–347. Springer, 2014.

[147] Yixuan Li, Lei Chen, Runyu He, Zhenzhi Wang, Gangshan Wu, and Limin Wang. Multisports: A multi-person video dataset of spatio-temporally localized sports actions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13536–13545, 2021.

[148] Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 513–528, 2018.

[149] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. Exploring plain vision transformer backbones for object detection. In *ECCV*, 2022.

[150] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss

for dense object detection. In *ICCV*, 2017.

[151] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014.

[152] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[153] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):978–994, 2011.

[154] Hanxiao Liu, Zihang Dai, David R So, and Quoc V Le. Pay attention to mlps. *arXiv preprint arXiv:2105.08050*, 2021.

[155] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[156] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[157] Liyuan Liu, Xiaodong Liu, Jianfeng Gao, Weizhu Chen, and Jiawei Han. Understanding the difficulty of training transformers. In *EMNLP*, 2020.

[158] Shikun Liu, Linxi Fan, Edward Johns, Zhiding Yu, Chaowei Xiao, and Anima Anandkumar. Prismer: A vision-language model with an ensemble of experts. *arXiv preprint arXiv:2303.02506*, 2023.

[159] Songtao Liu, Zeming Li, and Jian Sun. Self-emd: Self-supervised object detection without imagenet. *arXiv preprint arXiv:2011.13677*, 2020.

[160] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg. Ssd:

Single shot multibox detector. In *European Conference on Computer Vision*, 2016.

[161] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021.

[162] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition*, 2015.

[163] Jonathan L Long, Ning Zhang, and Trevor Darrell. Do convnets learn correspondence? In *Advances in neural information processing systems*, pages 1601–1609, 2014.

[164] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[165] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[166] J. Ma, W. Qiu, J. Zhao, Y. Ma, A. L. Yuille, and Z. Tu. Robust l2e estimation of transformation for non-rigid registration. *IEEE Transactions on Signal Processing*, 63(5):1115–1129, 2015.

[167] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *ICLR*, 2018.

[168] Jitendra Malik, Pablo Arbeláez, João Carreira, Katerina Fragkiadaki, Ross Girshick, Georgia Gkioxari, Saurabh Gupta, Bharath Hariharan, Abhishek Kar, and Shubham Tulsiani. The three r's of computer vision: Recognition, reconstruction and reorganization. *Pattern Recognition Letters*, 72:4–14, 2016.

[169] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from

maximally stable extremal regions. *Image and Vision Computing*, 22(10):761–767, 2004.

[170] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.

[171] Ross Messing, Chris Pal, and Henry Kautz. Activity recognition using the velocity histories of tracked keypoints. In *2009 IEEE 12th international conference on computer vision*, pages 104–111. IEEE, 2009.

[172] Jiaxu Miao, Xiaohan Wang, Yu Wu, Wei Li, Xu Zhang, Yunchao Wei, and Yi Yang. Large-scale video panoptic segmentation in the wild: A benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

[173] Seyed Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant. In *AAAI*, 2020.

[174] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6707–6717, 2020.

[175] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfreund, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):502–508, 2019.

[176] Mathew Monfort, Bowen Pan, Kandan Ramakrishnan, Alex Andonian, Barry A

McNamara, Alex Lascelles, Quanfu Fan, Dan Gutfreund, Rogério Schmidt Feris, and Aude Oliva. Multi-moments in time: Learning and interpreting models for multi-action video understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9434–9445, 2021.

[177] Naila Murray, Luca Marchesotti, and Florent Perronnin. Ava: A large-scale database for aesthetic visual analysis. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2408–2415. IEEE, 2012.

[178] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *International Conference on Machine Learning*, 2010.

[179] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. In *NeurIPS*, 2021.

[180] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, 2016.

[181] Guanghan Ning and Zhihai He. Dual path networks for multi-person human pose estimation. *arXiv preprint arXiv:1710.10192*, 2017.

[182] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, pages 69–84. Springer, 2016.

[183] D. Novotnỳ, D. Larlus, and A. Vedaldi. Anchornet: A weakly supervised network to learn geometry-sensitive features for semantic matching. In *Computer Vision and Pattern Recognition*, 2017.

[184] M. Okutomi and T. Kanade. A multiple-baseline stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353–363, 1993.

[185] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with

contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[186] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *Computer Vision and Pattern Recognition*, 2009.

[187] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016.

[188] Sayak Paul and Pin-Yu Chen. Vision transformers are robust learners. In *AAAI*, 2022.

[189] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017.

[190] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016.

[191] Pedro O Pinheiro, Amjad Almahairi, Ryan Y Benmaleck, Florian Golemo, and Aaron Courville. Unsupervised learning of dense visual representations. *arXiv preprint arXiv:2011.05499*, 2020.

[192] Kris Popat and Rosalind W. Picard. Novel cluster-based probability model for texture synthesis, classification, and compression. In *Proc. SPIE Visual Comm. and Image Processing*, 1993.

[193] W. Qiu and A. L. Yuille. Unrealcv: Connecting computer vision to unreal engine. In *European Conference on Computer Vision*, 2016.

[194] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya

Sutskever. Improving language understanding by generative pre-training. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

[195] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[196] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[197] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *CVPR*, 2020.

[198] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jonathon Shlens. Stand-alone self-attention in vision models. In *NeurIPS*, 2019.

[199] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021.

[200] MarcAurelio Ranzato, Arthur Szlam, Joan Bruna, Michael Mathieu, Ronan Collobert, and Sumit Chopra. Video (language) modeling: a baseline for generative models of natural videos. *arXiv preprint arXiv:1412.6604*, 2014.

[201] John C Raven. Standardization of progressive matrices, 1938. *British Journal of Medical Psychology*, 1941.

[202] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition*,

2014.

[203] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Computer Vision and Pattern Recognition*, 2016.

[204] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10901–10911, 2021.

[205] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, 2015.

[206] Oren Rippel, Manohar Paluri, Piotr Dollar, and Lubomir Bourdev. Metric learning with adaptive density discrimination. *arXiv preprint arXiv:1511.05939*, 2015.

[207] Byungseok Roh, Wuhyun Shin, Ildoo Kim, and Sungwoong Kim. Spatially consistent representation learning. In *CVPR*, 2021.

[208] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550*, 2014.

[209] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015.

[210] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[211] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2016.

[212] Arno Schodl, Richard Szeliski, David H. Salesin, and Irfan Essa. Video textures. In *Proceedings of SIGGRAPH '00*, pages 489–498, 2000.

[213] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[214] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems*, 35:25278–25294, 2022.

[215] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. Laion-5b: An open large-scale dataset for training next generation image-text models, 2022.

[216] Ozan Sener and Vladlen Koltun. Multi-task learning as multi-objective optimization. *Advances in neural information processing systems*, 31, 2018.

[217] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *NeurIPS*, 2019.

[218] Claude E. Shannon. A mathematical theory of communication. *Bell Sys. Tech. Journal*, 27, 1948.

[219] Claude E Shannon. Prediction and entropy of printed english. *Bell system technical journal*, 30(1):50–64, 1951.

[220] Rulin Shao, Zhouxing Shi, Jinfeng Yi, Pin-Yu Chen, and Cho-Jui Hsieh. On the adversarial robustness of visual transformers. *arXiv preprint arXiv:2103.15670*, 2021.

[221] Gunnar A Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. Charades-ego: A large-scale dataset of paired third and first person videos. *arXiv preprint arXiv:1804.09626*, 2018.

[222] Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 510–526. Springer, 2016.

[223] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[224] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[225] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Advances in neural information processing systems*, pages 1857–1865, 2016.

[226] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[227] X. Soria, E. Riba, and A. Sappa. Dense extreme inception network: Towards a robust cnn model for edge detection. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1912–1921, Los Alamitos, CA, USA, mar 2020. IEEE Computer Society.

[228] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[229] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *ICCV*, 2017.

[230] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[231] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al. Going deeper with convolutions. In *Computer Vision and Pattern Recognition*, 2015.

[232] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014.

[233] R. Szeto and J. J. Corso. Click here: Human-localized keypoints as guidance for viewpoint estimation. In *International Conference on Computer Vision*, 2017.

[234] J. Thewlis, H. Bilen, and A. Vedaldi. Unsupervised learning of object landmarks by factorized spatial embeddings. In *International Conference on Computer Vision*, 2017.

[235] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive multiview coding.

arXiv preprint arXiv:1906.05849, 2019.

[236] Yonglong Tian, Dilip Krishnan, and Phillip Isola. Contrastive representation distillation. In *ICLR*, 2020.

[237] Yonglong Tian, Chen Sun, Ben Poole, Dilip Krishnan, Cordelia Schmid, and Phillip Isola. What makes for good views for contrastive learning. In *NeurIPS*, 2020.

[238] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. Mlp-mixer: An all-mlp architecture for vision. *arXiv preprint arXiv:2105.01601*, 2021.

[239] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–656, 2015.

[240] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[241] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[242] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, pages 10347–10357. PMLR, 2021.

[243] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers &amp; distillation through attention. In *International Conference on Machine Learning*, volume 139, pages 10347–10357, July 2021.

[244] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *ICCV*, 2021.

[245] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *CVPR*, 2021.

[246] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[247] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1510–1519, 2015.

[248] N. Ufer and B. Ommer. Deep semantic feature matching. In *Computer Vision and Pattern Recognition*, 2017.

[249] J. R. R. Uijlings, K. E. A. Van De Sande, T. Gevers, and A. W. M. Smeulders. Selective search for object recognition. *International Journal of Computer Vision*, 104(2):154–171, 2013.

[250] Aaron van den Oord, Nal Kalchbrenner, Lasse Espeholt, koray kavukcuoglu, Oriol Vinyals, and Alex Graves. Conditional image generation with pixelcnn decoders. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in*

Neural Information Processing Systems, volume 29. Curran Associates, Inc., 2016.

[251] Äaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel recurrent neural networks. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1747–1756, New York, New York, USA, 20–22 Jun 2016. PMLR.

[252] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[253] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[254] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[255] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML ’08, page 1096–1103, 2008.

[256] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *JMLR*, 11(12), 2010.

[257] Angtian Wang, Yihong Sun, Adam Kortylewski, and Alan Yuille. Robust object detection under occlusion with context-aware compositionalnets. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, 2020.

[258] J. Wang, C. Xie, Z. Zhang, J. Zhu, L. Xie, and A. L. Yuille. Detecting semantic parts on partially occluded objects. In *British Machine Vision Conference*, 2017.

[259] J. Wang, Z. Zhang, C. Xie, V. Premachandran, and A. Yuille. Unsupervised learning of object semantic parts from internal states of cnns by population encoding. *arXiv preprint arXiv:1511.06855*, 2015.

[260] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, 2021.

[261] Xinlong Wang, Wen Wang, Yue Cao, Chunhua Shen, and Tiejun Huang. Images speak in images: A generalist painter for in-context visual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6830–6839, 2023.

[262] Xinlong Wang, Rufeng Zhang, Chunhua Shen, Tao Kong, and Lei Li. Dense contrastive learning for self-supervised visual pre-training. *arXiv preprint arXiv:2011.09157*, 2020.

[263] Ryan Webster, Julien Rabin, Loic Simon, and Frederic Jurie. On the de-duplication of laion-2b, 2023.

[264] Chen Wei, Haoqi Fan, Saining Xie, Chao-Yuan Wu, Alan Yuille, and Christoph Feichtenhofer. Masked feature prediction for self-supervised visual pre-training. In *CVPR*, 2022.

[265] Chen Wei, Wenjing Wang, Wenhan Yang, and Jiaying Liu. Deep retinex decomposition for low-light enhancement. *arXiv preprint arXiv:1808.04560*, 2018.

[266] Fangyun Wei, Yue Gao, Zhirong Wu, Han Hu, and Stephen Lin. Aligning pretraining

for detection via object-level contrastive learning. 2021.

[267] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[268] Kilian Q Weinberger and Lawrence K Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10(2), 2009.

[269] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. In *ICLR*, 2020.

[270] Jiajun Wu, Tianfan Xue, Joseph J Lim, Yuandong Tian, Joshua B Tenenbaum, Antonio Torralba, and William T Freeman. Single image 3d interpreter network. In *European Conference on Computer Vision*, pages 365–382. Springer, 2016.

[271] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[272] Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018.

[273] Yu Xiang, Wonhui Kim, Wei Chen, Jingwei Ji, Christopher Choy, Hao Su, Roozbeh Mottaghi, Leonidas Guibas, and Silvio Savarese. Objectnet3d: A large scale database for 3d object recognition. In *European Conference on Computer Vision*, pages 160–176. Springer, 2016.

[274] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, 2014.

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

[276] Junfei Xiao, Longlong Jing, Lin Zhang, Ju He, Qi She, Zongwei Zhou, Alan Yuille, and Yingwei Li. Learning from temporal gradient for semi-supervised action recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3252–3262, 2022.

[277] Tete Xiao, Colorado J Reed, Xiaolong Wang, Kurt Keutzer, and Trevor Darrell. Region similarity representation learning. In *ICCV*, 2021.

[278] Cihang Xie, Mingxing Tan, Boqing Gong, Alan Yuille, and Quoc V Le. Smooth adversarial training. *arXiv preprint arXiv:2006.14536*, 2020.

[279] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. L. Yuille. Adversarial examples for semantic segmentation and object detection. In *International Conference on Computer Vision*, 2017.

[280] Enze Xie, Jian Ding, Wenhai Wang, Xiaohang Zhan, Hang Xu, Peize Sun, Zhenguo Li, and Ping Luo. Detco: Unsupervised contrastive learning for object detection. In *CVPR*, 2021.

[281] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. In *NeurIPS*, 2021.

[282] Jiahao Xie, Xiaohang Zhan, Ziwei Liu, Yew Ong, and Chen Change Loy. Unsupervised object-level representation learning from scene images. *Advances in Neural Information Processing Systems*, 34, 2021.

[283] Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc Le. Self-training with noisy

student improves imagenet classification. In *CVPR*, 2020.

[284] Saining Xie, Jiatao Gu, Demi Guo, Charles R Qi, Leonidas Guibas, and Or Litany. Pointcontrast: Unsupervised pre-training for 3d point cloud understanding. In *ECCV*, 2020.

[285] S. Xie and Z. Tu. Holistically-nested edge detection. In *International Conference on Computer Vision*, 2015.

[286] Zhenda Xie, Yutong Lin, Zheng Zhang, Yue Cao, Stephen Lin, and Han Hu. Propagate yourself: Exploring pixel-level consistency for unsupervised visual representation learning. In *CVPR*, 2021.

[287] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *CVPR*, 2022.

[288] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016.

[289] Ning Xu, Linjie Yang, Yuchen Fan, Dingcheng Yue, Yuchen Liang, Jianchao Yang, and Thomas Huang. Youtube-vos: A large-scale video object segmentation benchmark. *arXiv preprint arXiv:1809.03327*, 2018.

[290] Fuzhao Xue, Ziji Shi, Yuxuan Lou, Yong Liu, and Yang You. Go wider instead of deeper. *arXiv preprint arXiv:2107.11817*, 2021.

[291] Chenglin Yang, Adam Kortylewski, Cihang Xie, Yinzhi Cao, and Alan Yuille. Patchattack: A black-box texture-based attack with reinforcement learning. In *ECCV*, 2020.

[292] Ceyuan Yang, Zhirong Wu, Bolei Zhou, and Stephen Lin. Instance localization for

self-supervised detection pretraining. In *CVPR*, 2021.

[293] Fengyu Yang, Jiacheng Zhang, and Andrew Owens. Generating visual scenes from touch. *International Conference on Computer Vision (ICCV)*, 2023.

[294] H. Yang, W. Y. Lin, and J. Lu. Daisy filter flow: A generalized discrete approach to dense correspondences. In *Computer Vision and Pattern Recognition*, 2014.

[295] Jing Yang, Brais Martinez, Adrian Bulat, Georgios Tzimiropoulos, et al. Knowledge distillation via softmax regression representation learning. In *ICLR*, 2021.

[296] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019.

[297] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. Lift: Learned invariant feature transform. In *European Conference on Computer Vision*, 2016.

[298] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

[299] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation, 2022.

[300] Qihang Yu, Yingda Xia, Yutong Bai, Yongyi Lu, Alan L Yuille, and Wei Shen. Glance-and-gaze vision transformer. In *NeurIPS*, 2021.

[301] Yingchen Yu, Fangneng Zhan, Rongliang Wu, Jianxiong Pan, Kaiwen Cui, Shijian Lu, Feiying Ma, Xuansong Xie, and Chunyan Miao. Diverse image inpainting with bidirectional and autoregressive transformers. In *Proceedings of the 29th ACM*

International Conference on Multimedia, pages 69–78, 2021.

[302] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet. In *ICCV*, 2021.

[303] 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 *ICCV*, 2019.

[304] Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *CVPR*, pages 3712–3722, 2018.

[305] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. *arXiv preprint arXiv:2106.04560*, 2021.

[306] Chongzhi Zhang, Mingyuan Zhang, Shanghang Zhang, Daisheng Jin, Qiang Zhou, Zhongang Cai, Haiyu Zhao, Shuai Yi, Xianglong Liu, and Ziwei Liu. Delving deep into the generalization of vision transformers under distribution shifts. *arXiv preprint arXiv:2106.07617*, 2021.

[307] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ICLR*, 2018.

[308] He Zhang and Vishal M Patel. Density-aware single image de-raining using a multi-stream dense network. In *CVPR*, 2018.

[309] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[310] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In

ECCV. Springer, 2016.

[311] Xiao Zhang and Michael Maire. Self-supervised visual representation learning from hierarchical grouping. *arXiv preprint arXiv:2012.03044*, 2020.

[312] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *CVPR*, 2018.

[313] Zhengxin Zhang, Qingjie Liu, and Yunhong Wang. Road extraction by deep residual u-net. *IEEE Geoscience and Remote Sensing Letters*, 15(5):749–753, 2018.

[314] Z. Zhang, C. Xie, J. Wang, L. Xie, and A. L. Yuille. Deepvoting: An explainable framework for semantic part detection under partial occlusion. In *Computer Vision and Pattern Recognition*, 2018.

[315] Borui Zhao, Quan Cui, Renjie Song, Yiyu Qiu, and Jiajun Liang. Decoupled knowledge distillation. *arXiv preprint arXiv:2203.08679*, 2022.

[316] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020.

[317] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Object detectors emerge in deep scene cnns. 2015.

[318] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *CVPR*, 2017.

[319] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127(3):302–321, 2019.

[320] T. Zhou, P. Krahenbuhl, M. Aubry, Q. Huang, and A. A. Efros. Learning dense correspondence via 3d-guided cycle consistency. In *Computer Vision and Pattern Recognition*, 2016.

[321] Xingyi Zhou, Arjun Karpur, Linjie Luo, and Qixing Huang. Starmap for category-agnostic keypoint and viewpoint estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 318–334, 2018.

[322] Chen Zhu, Wei Ping, Chaowei Xiao, Mohammad Shoeybi, Tom Goldstein, Anima Anandkumar, and Bryan Catanzaro. Long-short transformer: Efficient transformers for language and vision. In *NeurIPS*, 2021.

[323] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models. *arXiv preprint arXiv:2304.10592*, 2023.

[324] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision*, 2017.

[325] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.

Vita

Yutong Bai is completing her Ph.D. degree of Computer Science at the Johns Hopkins University, under the supervision of Bloomberg Distinguished Professor Alan L. Yuille. Yutong received her B.S. degree in Software Engineering from Northwestern Polytechnical University in 2019. Yutong's research interests lie in the fields of computer vision, deep learning, and self-supervised learning.