THE UNIVERSITY OF CHICAGO

OBJECT DETECTION AND PANOPTIC SEGMENTATION THROUGH LIKELIHOOD OPTIMIZATIONS

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE DIVISION OF THE PHYSICAL SCIENCES
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

DEPARTMENT OF STATISTICS

BY
ANGZHI FAN

CHICAGO, ILLINOIS
JUNE 2024

In dedication to my family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I wish to express my deepest gratitude to my advisor, Professor Yali Amit, for accepting me as his student and for his meticulous guidance throughout my doctoral studies over the years. It has been a great pleasure collaborating with him. In our weekly meetings, he consistently listens attentively to the details of my reports and offers insightful feedback. His profound knowledge, patience, and support have left an indelible impression on me. I sincerely appreciate the time and energy he has dedicated to my academic development.

I extend my gratitude to the other members of my dissertation committee, namely Professor Greg Shakhnarovich and Professor Victor Veitch, for their insightful suggestions regarding my research projects.

I express my gratitude to my friends, classmates and office mates at the University of Chicago, particularly those within the Department of Statistics, such as Yuhan Liu, Wei Kuang, Irina Cristali, Zehao Niu, Yi Wei, Yi Wang, Lijia Zhou, Wanrong Zhu, Yuwei Cheng, Xiaohan Zhu, Jiacheng Wang, Qing Yan, Changji Xu, Yuguan Wang, Yeo Jin Jung, and others. Additionally, I extend my thanks to the faculty and staff members of the Department of Statistics. Their companionship throughout my doctoral program has contributed significantly to a rewarding and enriching experience.

Last but not least, I would like to thank my parents for their unwavering support. The foundational education they provided during my formative years continues to exert a profound influence on my present self.

ABSTRACT

This thesis focuses on two pivotal subjects within the domain of Computer Vision: object detection and panoptic segmentation. Fueled by deep neural networks, substantial advancements have been witnessed in these fields in recent years. Many efforts in object detection and panoptic segmentation rely on feed-forward approaches, lacking a probabilistic interpretation. In response to this, the present thesis puts forth three innovative algorithms: the Detection Selection Algorithm, the Detection Selection Algorithm with Mask, and the Maximizing the Posterior for Panoptic Segmentation Algorithm. The initial algorithm is tailored for object detection, while the latter two are specifically devised for panoptic segmentation.

These three algorithms are rooted in three distinct probabilistic frameworks. Notwithstanding, they still depend on feed-forward models like Faster R-CNN and Mask R-CNN to generate raw object detections and instance segmentations. Given an image and a hypothesis regarding object configuration and latent codes, the probabilistic frameworks define their respective likelihoods. The primary objective of these algorithms is to identify a configuration hypothesis that maximizes these likelihoods. They employ greedy search procedures to mitigate computational complexity. These three algorithms differ in their approaches to maximizing likelihoods, with some maximizing a log joint probability and another maximizing a posterior probability.

The computation of likelihoods necessitates auxiliary tools, including Deep Generative Models that capture the distribution of object appearances. In the case of these three algorithms, we employ the Variational Autoencoder, VAE with flow prior, and Generative Latent Flow, respectively. To conduct inference on the distribution of latent codes, Single Reconstruction Algorithms are designed. Additionally, Whole Reconstruction Algorithms are introduced to amalgamate the probability model of individual objects into a comprehensive probability model for the entire image. They necessitate occlusion relationship reasoning methods to identify the visible components of objects. Experimental results demonstrate

that our algorithms yield improvements in tasks such as object counting and enhancement of Panoptic Quality scores. This thesis aims to showcase the potency of probabilistic modeling in the world of contemporary machine learning.

CHAPTER 1

INTRODUCTION

1.1 Motivation and Overview

Within the realm of artificial intelligence, object detection and panoptic segmentation emerge as two significant tasks that have attracted considerable attention. Object detection finds wide-ranging applications in real-world scenarios, including medical image processing, object tracking, facial recognition, and more. On the other hand, panoptic segmentation, a relatively recent research field, offers a more comprehensive understanding of image scenes compared to object detection and has demonstrated successful applications in autonomous driving.

In the pre-deep-learning era of computer vision, probabilistic modeling approaches, such as the POP model Amit and Trouvé [2007], enjoyed popularity in object detection and classification. However, with the advent of deep learning, mainstream research in object detection and panoptic segmentation has shifted towards a more data-driven paradigm, providing greater flexibility in modeling but sacrificing interpretability.

In contrast to many deep learning models characterized by pure feed-forward network architectures, the algorithms proposed in this thesis incorporate online optimizations. These algorithms are grounded in probabilistic frameworks with clear probability interpretations, setting them apart from previous works. Despite the current lack of emphasis on probabilistic modeling, our experiments in this thesis demonstrate its efficacy in the post-processing of object detection and panoptic segmentation. Through the ensuing chapters, we aim to elucidate the enhancements that probabilistic modeling can bring to machine learning tasks.

The rest of this chapter is structured as follows: Section 1.2 provides a comprehensive review of various research fields pertinent to our algorithms, encompassing Deep Generative Models (DGMs), depth estimation, occlusion relationship reasoning, object detection,

instance and semantic segmentation, and panoptic segmentation. Following this, Section 1.3 delineates the principal contributions made by this thesis.

Chapter 2 furnishes an in-depth examination of three baseline models: Faster R-CNN Ren et al. [2015], Mask R-CNN He et al. [2017], and the POP model Amit and Trouvé [2007]. Faster R-CNN serves as the baseline for the Detection Selection Algorithm (DSA) introduced in Chapter 3, within the context of the object detection task. Mask R-CNN, employed as the baseline model for panoptic segmentation, is juxtaposed with our Detection Selection Algorithm with Mask (DSAM) and Maximizing the Posterior for Panoptic Segmentation Algorithm (MPPS), expounded upon in Chapters 4 and 5 respectively. The Patchwork of Parts (POP) model, devoid of neural network training, motivates our probabilistic modeling of images.

Detection Selection Algorithm (DSA), as elaborated in Chapter 3, serves as a post-processing method tailored for object detection algorithms such as Faster R-CNN. The chapter introduces Faster R-CNN-OC, which performs occlusion relationship reasoning but lacks generalization ability. Additionally, this chapter introduces two auxiliary tools for DSA: the Single Reconstruction Algorithm for DSA (DSASR) and the Whole Reconstruction Algorithm for DSA (DSAWR). The object detection problem is then formulated within a probabilistic framework. Finally, it introduces DSA, the main algorithm in this chapter and shows some related experiments.

Detection Selection Algorithm with Mask (DSAM), detailed in Chapter 4, functions as a post-processing method for instance segmentation models like Mask R-CNN. DSAM is designed to enhance the quality of panoptic segmentation, wherein everything apart from the specifically addressed objects is denoted by a generic "background" semantic label. Chapter 4 establishes a distinct probabilistic framework to address potential visual clutter and colored backgrounds that do not appear in the previous chapter. For occlusion relationship reasoning, MiDaS Lasinger et al. [2019], a depth estimation tool, is employed. Two auxiliary algorithms,

DSAMSR and DSAMWR, are introduced, specifically tailored for the main algorithm DSAM. Experimental results for DSAM are also presented in Chapter 4.

Maximizing the Posterior for Panoptic Segmentation Algorithm (MPPS), outlined in Chapter 5, represents another post-processing method for instance segmentation models. Operating within a probabilistic framework distinct from that of DSAM, MPPS utilizes a different Deep Generative Model. Comparative experiments in Chapter 5 reveal that MPPS demonstrates slightly superior performance to DSAM. The chapter also enumerates potential directions for enhancing MPPS.

1.2 Related Works

1.2.1 Deep Generative Models

Most Deep Generative Models (DGMs) are crafted with the primary objective of acquiring a probability distribution from input data while concurrently possessing the capacity to generate novel samples based on the acquired distribution. Exemplary instances of DGMs encompass Variational Auto-encoders (VAEs) Kingma and Welling [2013], Rezende et al. [2014], Burda et al. [2015], Auto-regressive Models Larochelle and Murray [2011], Uria et al. [2014], Van den Oord et al. [2016], Normalizing Flows Dinh et al. [2014, 2016], Kingma and Dhariwal [2018], Deep Energy-based Models Du and Mordatch [2019], Welling and Teh [2011], Generative Adversarial Networks (GANs) Goodfellow et al. [2014], Radford et al. [2015], Arjovsky et al. [2017] and Diffusion Models Ho et al. [2020], Xu et al. [2023]. The family of DGMs can be subdivided into two categories: explicit models and implicit models. Explicit models explicitly formulate the likelihood function or provide an approximation of the likelihood function for the given data. Conversely, implicit models refrain from explicitly specifying the likelihood and instead focus on training a model to sample from it.

One of the most preeminent explicit models is the Variational Auto-encoders (VAEs)

Kingma and Welling [2013], which optimizes a variational lower bound to solve the computational intractability of the marginal likelihood. The encoder of VAE predicts a posterior distribution within the latent space whereas the decoder maps latent code to the image space. VAE with flow prior Huang et al. [2017] represents a more sophisticated iteration of the conventional VAE framework, as they integrate a flow prior within the latent space. Generative Latent Flow (GLF) Xiao et al. [2019] employs an auto-encoder to acquire latent representations and adds an invertible flow model to map the latent representation to gaussian noise. Notably, GLF Xiao et al. [2019] mitigates the over-regularization issue observed in VAE Kingma and Welling [2013], Huang et al. [2017]. While VAE with flow prior Huang et al. [2017] provides an estimate of the marginal likelihood, it is noteworthy that GLF Xiao et al. [2019] does not offer such an approximation. Similar to GLF, Latent Diffusion Models (LDMs) Rombach et al. [2022] employ diffusion models (DMs) to acquire the latent representation subsequent to dimension reduction through pre-trained autoencoders. This enhancement accelerates the training of DMs by a minimum factor of 2.7 and yields noteworthy improvements in FID scores Heusel et al. [2017].

1.2.2 Monocular Depth Estimation

The technique of estimating depth from a single image, known as Monocular Depth Estimation Ming et al. [2021], Eigen et al. [2014], Lasinger et al. [2019], Kim et al. [2022], yields an estimated pixel-level depth map, proving valuable in diverse applications, including but not limited to autonomous driving. Recent advancements Eigen et al. [2014], Lasinger et al. [2019], Kim et al. [2022] facilitated by Convolutional Neural Networks (CNNs) have notably improved the speed and accuracy of Monocular Depth Estimation. Global-Local Path Networks Kim et al. [2022] incorporates self-attention Vaswani et al. [2023] in its encoder and introduces a Selective Feature Fusion (SFF) module in the decoder to effectively merge global and local features. MiDaS Lasinger et al. [2019] develops a training strategy involving

multiple data sources and trained with a dataset derived from 3D movies. The outcomes of MiDaS Lasinger et al. [2019] has been assessed through zero-shot cross-dataset transfer, revealing its capacity for generalization to previously unseen data.

1.2.3 Object Detection

The application of neural networks in object detection has attracted much attention. Some object detection algorithms rely on region proposal generation. For example, Fast R-CNN Girshick [2015b] and Faster R-CNN Ren et al. [2015]. Some are single-stage methods, including Single Shot MultiBox Detector (SSD) Liu et al. [2015] and You Only Look Once (YOLO) Redmon et al. [2015]. All these object detection methods predict class probabilities and bounding box locations. Faster R-CNN Ren et al. [2015] also predicts the objectness score which represent the confidence of a detection.

Post-processing is an important step to remove false positive detections in all object detection algorithms. One type of popular post-processing methods is Non-maximum Suppression (NMS) and its variants. The paper Efficient Non-Maximum Suppression Neubeck and Van Gool [2006] proposed several algorithms to accelerate NMS. A recent review paper Gong et al. [2021] summarized five NMS techniques: Soft-NMS Bodla et al. [2017b], Softer-NMS He et al. [2018], IOU-Guided NMS Jiang et al. [2018], Adaptive NMS Liu et al. [2019b] and DIoU-NMS Zheng et al. [2020].

These five methods emphasize local information as opposed to optimizing a global objective function. Among them, Softer-NMS He et al. [2018], IOU-Guided NMS Jiang et al. [2018] and Adaptive NMS Liu et al. [2019b] require modifying the detection model or adding additional modules. Instead of setting a threshold to suppress highly overlapping bounding boxes, in each step Soft-NMS Bodla et al. [2017b] decreases the detection score by a factor that depends on the IoU. Distance-IoU (DIoU) Zheng et al. [2020] takes into account the distance between the centers of bounding boxes. The idea of DIoU can be used in NMS and

in designing IoU-related loss functions.

Another type of post-processing method defines a global objective functions and uses some search procedure to choose the final detections. Examples are a Bayesian model for face detection Zaytseva and Vitrià [2012], HS-NMS Song et al. [2019] , probabilistic faster R-CNN Yi et al. [2021] and Patchwork of Parts (POP) Models Amit and Trouvé [2007]. Probabilistic faster R-CNN Yi et al. [2021] trains Gaussian Mixture Models (GMM) on heights and widths of region proposals Girshick [2015b] and uses GMM to calculate the likelihood for each region proposal. The Bayesian model in Zaytseva and Vitrià [2012] first uses a kernel smoother on the face hypotheses to estimate the prior distribution, then uses face templates to estimate face likelihood, and use MCMC to get a stable face distribution from the posterior distribution. The POP Model, as detailed in the work by Amit et al. Amit and Trouvé [2007], acquires templates in the form of probability arrays grounded in edge features. Subsequently, it establishes reference points to augment the initial rigid model into a deformable model while retaining the capability to compute likelihood. The model engages in detection and occlusion order reasoning by finding the highest posterior.

1.2.4 Occlusion Relationship Reasoning

Understanding the occlusion relationship between objects is called Occlusion Relationship Reasoning. MT-ORL Feng et al. [2021] can predict object boundary maps and occlusion orientation maps and requires corresponding ground truths in order to train. A recent work Yuan et al. [2021] performs occlusion relationship reasoning by pixel-level competition for conflict areas in segmentation.

1.2.5 Semantic and Instance Segmentation

Leveraging neural networks, both semantic segmentation Minaee et al. [2021], Long et al. [2015], Noh et al. [2015] and instance segmentation Minaee et al. [2021], He et al. [2017],

Chen et al. [2019] have undergone substantial advancements in recent years. In the context of semantic segmentation, the image serves as input and the model predicts a label for each pixel within the image. Semantic segmentation involves the assignment of object labels at the pixel level but lacks the concept of object instance identification. On the other hand, instance segmentation is focused on the identification of object instances and their corresponding segments, and it exclusively assigns labels to pixels located within the predicted object masks. Notably, in contrast to object detection methods such as Faster R-CNN Ren et al. [2015], instance segmentation He et al. [2017] not only provides object labels and locations but also furnishes object masks.

A powerful model for instance segmentation is Mask R-CNN He et al. [2017]. Both Mask R-CNN He et al. [2017] and Faster R-CNN Ren et al. [2015] are categorized as two-stage detectors: The first stage employs a Region Proposal Network (RPN) to generate object proposals. The RPN operates on the convolutional feature map by mapping each sliding window into lower-dimensional features and predicts box-regression and box-classification through two fully-connected layers. The box-regression layer predicts a rectangular region boundary and the box-classification layer estimates the probability of an object's presence within the sliding window. To cater to diverse object sizes, RPN simultaneously predicts box-regression and box-classification for several anchors with varying scales and aspect ratios. The second stage of Faster R-CNN Ren et al. [2015] involves a Fast R-CNN Girshick [2015b] module that conducts classification and a more precise box-regression. In the case of Mask R-CNN He et al. [2017], the second stage extends beyond the Fast R-CNN Girshick [2015b] module by incorporating a parallel mask branch dedicated to predicting the object mask.

Researchers have investigated the reconstruction of the invisible parts and the prediction of the entire mask of an object including its invisible parts. The latter is often called amodal instance segmentation. SeGAN Ehsani et al. [2018] jointly predicts invisible masks and generates invisible parts of objects under the GAN Goodfellow et al. [2014] framework.

One work Qi et al. [2019] uses Multi-Level Coding to guide invisible mask prediction by multi-branch features.

In the realm of semantic segmentation, Fully Convolutional Networks (FCNs) Long et al. [2015] have showcased the capability of end-to-end convolutional networks and devised a methodology to integrate coarse and fine predictions. Models based on the encoder-decoder architecture, such as DeConvNet Noh et al. [2015], also show promising outcomes.

1.2.6 Panoptic Segmentation

Panoptic Segmentation Kirillov et al. [2019b], Li and Chen [2022], Elharrouss et al. [2021], Chuang et al. [2023] is a task wherein each pixel is assigned both a semantic label and an instance ID. It diverges from semantic segmentation, which lacks the concept of instance ID, and differs from instance segmentation, which allows for possible overlap between object masks. Panoptic Quality (PQ) metric Kirillov et al. [2019b] stands as a widely used evaluation measure. PQ is the product of segmentation quality (SQ) and recognition quality (RQ), where SQ quantifies the average IoU in true positives, while $RQ = \frac{|TP|}{|TP|+0.5|FP|+0.5|FN|}$ imposes penalties on false positives and false negatives. Within the context of Panoptic Segmentation Kirillov et al. [2019b], *thing class* refers to a category encompassing countable objects such as cars and people, while a *stuff class* comprises amorphous objects like sky and road. For any given stuff class, there exists at most 1 segment corresponding to that class within the image. In Panoptic Quality (PQ) metric Kirillov et al. [2019b], thing classes and stuff classes are treated with equal weight. In contrast, a modified version of Panoptic Quality Porzi et al. [2019] distinguishes between them by exempting the requirement of IoU > 0.5 in the matching between predicted and ground truth stuff segments. To address the issue of the possibly overlapping object masks in detections, a NMS-like fusion procedure was used in early panoptic segmentation research Kirillov et al. [2019b]. This NMS-like fusion procedure sorts instance proposals based on their objectness scores and subsequently assigns

pixels to instance proposals in a greedy fashion.

OCFusion Lazarow et al. [2020] identifies the suboptimal nature of the previously mentioned NMS-like fusion procedure and enhances it by incorporating an "occlusion head" into Mask R-CNN to perform binary classifications on occlusion relationships. The training of the occlusion head necessitates ground truth amodal instance masks and ground truth panoptic segmentation. In addition to OCFusion, the development of novel network modules for panoptic segmentation has garnered considerable attention. Panoptic FPN Kirillov et al. [2019a] extends Mask R-CNN He et al. [2017] by adding a semantic segmentation branch subsequent to a shared Feature Pyramid Network (FPN). EPSNet Chang et al. [2021] introduces a protohead to generate prototype masks for the whole image and a cross-layer attention (CLA) as fusion module. EfficientPS Mohan and Valada [2021] features a 2-way FPN as backbone. The end-to-end Occlusion Aware Network (OANet) Liu et al. [2019a] proposes a spatial ranking module that applies large kernel convolution to generate a ranking score map for addressing occlusion challenges. Pixel Consensus Voting (PCV) Wang et al. [2020] reformulates conventional offset regression as a classification problem and employs dilated deconvolution before aggregating the results into a voting heatmap. A Query Filter is implemented at peak regions of the voting heatmap so as to deduce object masks. Pixel Consensus Voting (PCV) Wang et al. [2020] draws inspiration from Generalized Hough transform and is not constructed within the dominant R-CNN framework Girshick et al. [2014], Girshick [2015b], Ren et al. [2015], He et al. [2017].

1.3 Main Contributions

The primary contributions of this thesis encompass the three probabilistic frameworks and their aforementioned algorithms: Detection Selection Algorithm (DSA), Detection Selection Algorithm with Mask (DSAM), and Maximizing the Posterior for Panoptic Segmentation Algorithm (MPPS). The conceptual novelty of this thesis arises from the integration of

probabilistic frameworks, marking a departure from prior work. Furthermore, the three probabilistic frameworks employed are distinctive in their individual characteristics.

An additional contribution lies in the formulation of auxiliary Whole Reconstruction Algorithms accompanying DSA, DSAM, and MPPS. These algorithms are designed to amalgamate information pertaining to individual objects, thereby facilitating the computation of probabilities for the entire image.

CHAPTER 2

BASELINE MODELS

In this chapter, a detailed exposition of three baseline models is provided: Faster R-CNN Ren et al. [2015], Mask R-CNN He et al. [2017], and POP model Amit and Trouvé [2007]. Serving as a baseline model for the Detections Selection Algorithm (DSA) discussed in Chapter 3, Faster R-CNN is expounded upon in Section 2.1. Mask R-CNN, delineated in Section 2.2, serves as the baseline model for both the Chapter 4 and Chapter 5. The POP model, discussed in Section 2.3, serves as the inspirational basis for our proposals in Chapter 3, 4 and 5. Although rooted in likelihood comparisons, notable distinctions exist between the POP model and our proposed models.

2.1 Faster R-CNN

Faster R-CNN Ren et al. [2015] incorporates Fast R-CNN Girshick [2015b] as one of its constituent elements, and Fast R-CNN, in turn, is founded on R-CNN Girshick et al. [2014]. The acronym R-CNN denotes "Regions with CNN features". R-CNN employs selective search Uijlings et al. [2013] to generate region proposals. These proposals, characterized by arbitrary shapes, are then resized to 227×227 before being input into AlexNet Krizhevsky et al. [2012] to yield 4096-dimensional feature vectors. Subsequently, R-CNN assesses these feature vectors using category-specific linear Support Vector Machines (SVMs), thus facilitating classification. Notably, the implementation of R-CNN necessitates pre-training AlexNet with image-level annotations and domain-specific fine-tuning on the designated dataset. Additionally, R-CNN can be further augmented through bounding-box regression, enhancing the accuracy of predictions regarding the location and scales of bounding boxes.

The Fast Region-based Convolutional Network (Fast R-CNN) Girshick [2015b] outperforms R-CNN in terms of both speed and detection accuracy. Fast R-CNN receives a convo-

lutional feature map and object proposals as input, as depicted in Figure 2.1. The convolutional feature map is derived from a deep detection network, such as VGG16 Simonyan and Zisserman [2014]. For the generation of object proposals, Fast R-CNN utilizes traditional object detectors like selective search Uijlings et al. [2013] or DPM Felzenszwalb et al. [2009].

The region of interest (RoI) associated with an object proposal, regardless of its size, is partitioned into an $H \times W$ grid of nearly uniformly-sized sub-windows, where H and W represent predetermined hyperparameters. All RoIs take on rectangle shapes, facilitating the division process. Standard max-pooling is executed for each sub-window and feature channel, constituting the *RoI pooling layer* depicted in Figure 2.1. This layer ensures that RoIs of arbitrary sizes are converted to the same size, enabling a fully connected (FC) layer to transform the max-pooled RoIs into fixed-sized RoI feature vectors. The final stage involves two tasks: classification and "bbox regression". In the classification task, softmax probabilities are predicted for K object classes along with a "background" class. The term "bbox regression" pertains to bounding box regression.

A predicted bounding box is defined by a tuple $P^u = (P^u_x, P^u_y, P^u_w, P^u_h)$, where (P^u_x, P^u_y) denotes the coordinates of the bounding box center, and P^u_w and P^u_h represent the width and height of the bounding box. These predicted bounding boxes are class-specific, with the class label denoted as the variable u. In contrast, the ground truth bounding box is denoted as $G = (G_x, G_y, G_w, G_h)$ and does not require a class label. The bounding box regression aims at predicting $t^u_* = (t^u_x, t^u_y, t^u_w, t^u_h)$, where

$$t^u_x = (G_x - P^u_x)/P^u_w$$

$$t^u_y = (G_y - P^u_y)/P^u_h$$

$$t^u_w = \log(G_w/P^u_w)$$

$$t^u_h = \log(G_h/P^u_h).$$

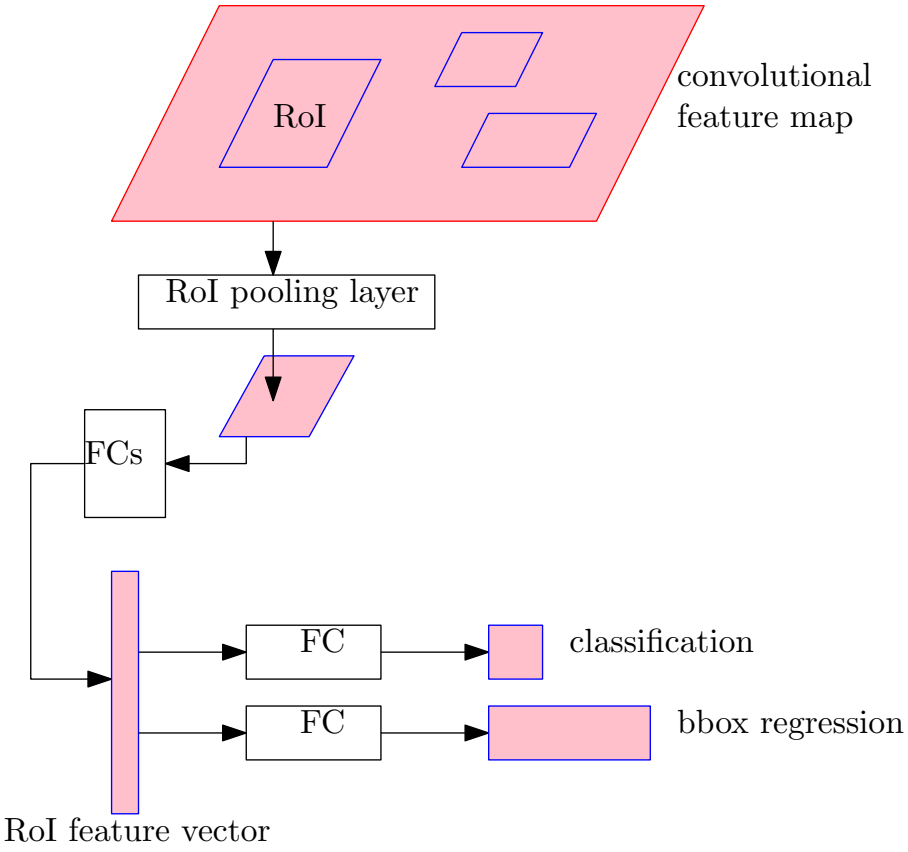


Figure 2.1: Fast R-CNN network architecture.

While it is possible to establish bounding box regression targets for each object class, excluding the background, the loss of bounding box regression is exclusively assessed for the true object class. This loss is then combined with the classification loss to formulate a multi-task loss. The training process of Fast R-CNN is characterized by a single-stage approach involving the utilization of the multi-task loss, making it considerably faster than R-CNN.

To elucidate the connections between Fast R-CNN and Faster R-CNN Ren et al. [2015], we depict the network architectures of Faster R-CNN in Figure 2.2, wherein one of its components is identical to Fast R-CNN. Faster R-CNN continues to generate softmax classification probabilities and bounding box regression outputs for each object proposal. However, in contrast to Fast R-CNN, Faster R-CNN takes the raw image as input and employs its Region Proposal Network (RPN) to generate object proposals. Within the entire Faster R-CNN

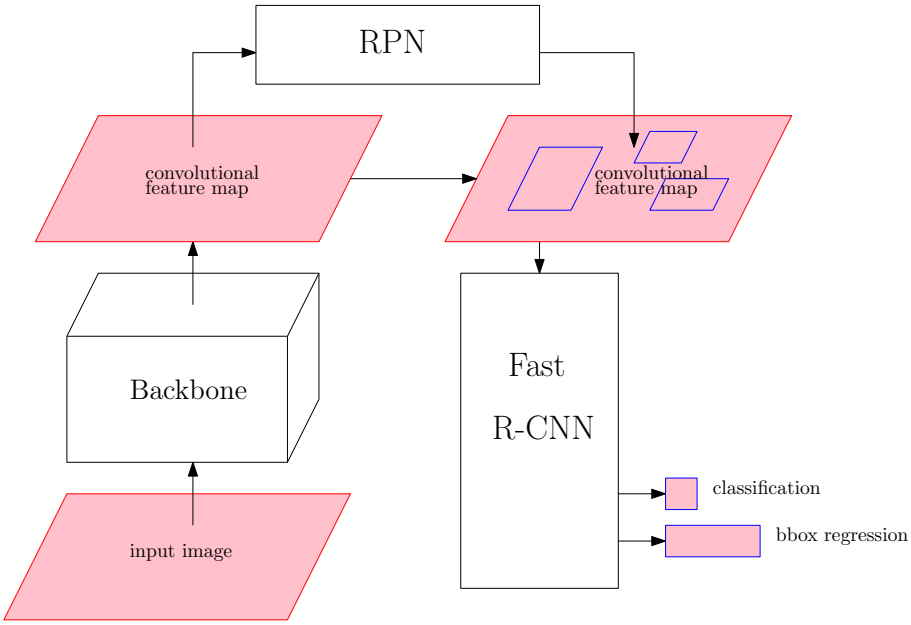


Figure 2.2: Faster R-CNN network architecture.

framework, RPN can be conceptualized as the "attention mechanism", guiding the Fast R-CNN component on where to focus. Notably, RPN and the Fast R-CNN component share the same convolutional feature map produced by the common *Backbone* detection network.

The Region Proposal Network (RPN) is constructed based on sliding windows applied to the convolutional feature map. At each position of the sliding window, there are, by default, 9 *anchors* consisting of 3 scales and 3 aspect ratios. RPN takes the content within the sliding window as input and employs two parallel fully connected layers to generate two distinct types of outputs:

- RPN predicts the *objectness score*, representing the probability of being an object rather than background, for each anchor.
- For each anchor, RPN conducts bounding box regression for its 4 coordinates.

Owing to the substantial overlap among the sliding windows and their anchors, post-processing is imperative to mitigate redundancy in the object proposals generated by RPN. By default, non-maximum suppression (NMS) is employed for this purpose. NMS follows a

greedy approach: iteratively selecting the object proposal with the highest objectness score and discarding any other object proposal with Intersection-over-Union (IoU) greater than a specified threshold (e.g., 0.7) concerning the previously selected object proposal. This process repeats until all remaining object proposals are chosen. Subsequently, the remaining object proposals are ranked based on their objectness scores, and only the top-N (e.g., $N = 300$) proposals are input into the Fast R-CNN module for classification and more precise bounding box regression. The superior performance of Faster R-CNN over Fast R-CNN, in terms of both speed and accuracy, is predominantly attributed to its RPN module.

Detection benchmarks such as PASCAL VOC 2007 Everingham et al. [2007] and the Microsoft COCO object detection dataset Lin et al. [2014] demonstrate the remarkable capability of Faster R-CNN. The PASCAL VOC 2007 dataset comprises approximately 5000 images for training and validation, along with an additional 5000 images for testing. This dataset encompasses 20 diverse object classes, ranging from animals to vehicles. The Microsoft COCO dataset is substantially larger, consisting of 80k, 40k, and 20k images for training, validation, and testing, respectively. Microsoft COCO includes 80 distinct object classes. The backbone network incorporates VGG16 Simonyan and Zisserman [2014] and ZF net Matthew Zeiler and Rob [2014], with the primary evaluation metric being mean Average Precision (mAP).

2.2 Mask R-CNN

Mask R-CNN He et al. [2017] serves as an extension to Faster R-CNN, specifically tailored for instance segmentation. In contrast to Faster R-CNN, Mask R-CNN employs the same RPN for generating object proposals but incorporates an additional branch within its Fast R-CNN module to predict object masks. Notably, for object mask prediction, fully convolutional networks (FCN) Long et al. [2015] are utilized, diverging from the approach of employing fully connected layers for classification and bounding box regression. The use of FCN ensures

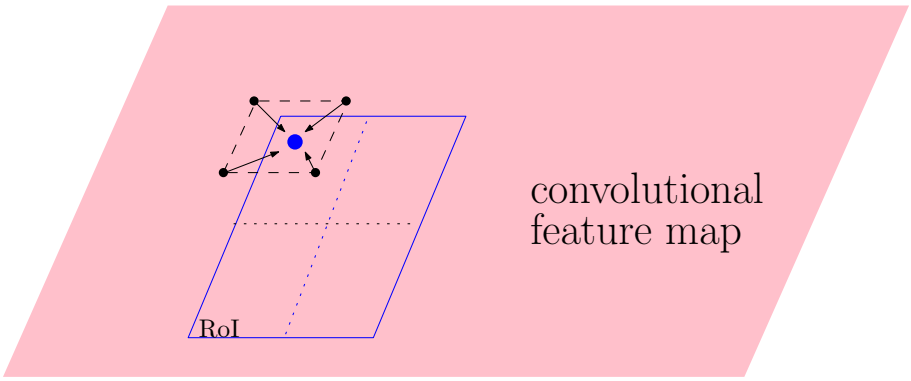


Figure 2.3: RoIAlign layer.

the preservation of spatial correspondence between Region of Interest (RoI) features and the original image, a characteristic not maintained by fully connected layers.

An additional noteworthy improvement introduced by Mask R-CNN is its RoIAlign layer. In the process of extracting RoI features from the convolutional feature map, Faster R-CNN utilizes RoIPool Girshick [2015b]. However, as the coordinates of bounding boxes may not be integers, RoIPool employs *quantization* on continuous coordinates. Quantization involves rounding a continuous coordinate to its nearest integer, leading to potentially detrimental small translations in the extracted feature map. In contrast, the RoIAlign layer, proposed by Mask R-CNN, replaces quantization with bilinear interpolation Jaderberg et al. [2015]. Similar to RoIPool, the RoIAlign layer necessitates the conversion of arbitrary-sized RoIs into fixed-sized $m \times m$ *RoI bins* before executing max-pooling.

Figure 2.3 illustrates the functioning of the RoIAlign layer. An RoI, represented by the blue box, is subdivided into 2×2 RoI bins. The RoIAlign layer systematically samples four points within each RoI bin, with the blue point being one of them. The feature value for the blue point is determined through bilinear interpolation from the four surrounding black points, which are all valid integer coordinates within the convolutional feature map. Subsequently, the feature values for the four blue points within each RoI bin are aggregated either by their maximum or their mean. By default, 7×7 RoI bins are employed. For a given RoI, Mask R-CNN predicts object masks for each object class, yet only the loss

corresponding to the ground truth class contributes to its training.

2.3 POP model

The Patchwork of Parts (POP) model Amit and Trouvé [2007] constitutes a deformable template model designed for tasks such as classification, object detection, and multi-object configurations, achieved through likelihood comparison. The POP model is constructed upon coarse binary oriented edge features Amit and Geman [1999], chosen for their resilience to intensity variations. Edges are detected across the 8 regular orientations and extended to encompass their 3×3 neighborhoods, a strategy employed to enhance robustness against local deformations.

A 2d lattice L is defined on the given image. For orientation e among those 8 orientations, a binary random variable $X_e(x), x \in L$ indicates if a edge of orientation e, or its 3×3 spread ones, is present at pixel x. The POP model first establishes a *rigid* model, which assumes conditionally independence of $X_e(x)$ across different pixel x given their marginal probabilities. For an object class centered at the origin, a probability array $(p_e(y))_{y \in \mathbb{Z}^2}$ is defined as $p_e(y) = \mathbb{P}(X_e(y) = 1)$ within the object support

$$S(0) = \{y \in \mathbb{Z}^2 : \max_e p_e(y) \geq \rho\},$$

where ρ is a fixed constant. Here, the pixel y is defined on $\mathbb{Z}^2$, representing the infinite 2d plane. Outside the object support $S(0)$, the rigid model still assumes $\mathbb{P}(X_e(y) = 1) = p_{e,bgd} > 0$, where $p_{e,bgd}$ denotes a homogeneous "background probability". The rigid model allows for a shift of the object, and during the inference phase, the rigid model is expressed as follows:

$$\mathbb{P}(X_e(x) = 1|r) = p_e(x - r), \ x \in L, \tag{2.1}$$

where the variable r represents the shift.

The rigid model is generalized by a *deformable* model, where n reference points y_i, $i = 1, 2, ..., n$ are defined. Corresponding to reference point y_i there is a shift v_i. Considering the overall translation r, the reference point y_i is displaced to a new location $z_i = y_i + r + v_i$. Then the deformable model performs kernel smoothing around each z_i

$$\mathbb{P}(X_e(x) = 1|\theta) = \frac{\sum_{i=1}^{n} p_e(x - r - v_i)\mathcal{K}(x - z_i)}{\sum_{i=1}^{n} \mathcal{K}(x - z_i)}, \; x \in L, \tag{2.2}$$

where $\theta = (r, v_1, v_2, ..., v_n)$ and $\mathcal{K}(\cdot)$ is a non-negative kernel. By default, $\mathcal{K}(\cdot)$ is selected to be an indicator function $\mathbf{1}_W(\cdot)$, where W represents a square centered around the origin. Owing to this kernel choice, only a square neighborhood surrounding each reference point, which is called a *part*, contributes to the right hand side of Equation 2.2. Consequently, the deformable model, which amalgamates the parts around the reference points, is referred to as the Patchwork of Parts (POP) model.

The deformable model maintains the assumption of conditional independence of $X_e(x)$ given their marginal probabilities and a background probability $p_{e,bgd}$. When considering the scenario where only one object is assumed to exist in the image, the probability of the entire image $X = \{X_e(x)|x \in L, e = 1, 2, ..., 8\}$ is computed as

$$\begin{aligned} \mathbb{P}(X|\theta) = &\prod_{x \in S(\theta)} \prod_{e} \mathbb{P}(X_e(x) = 1|\theta)^{X_e(x)} \mathbb{P}(X_e(x) = 0|\theta)^{1 - X_e(x)} \\ &\times \prod_{x \notin S(\theta)} \prod_{e} p_{e,bgd}^{X_e(x)} (1 - p_{e,bgd})^{1 - X_e(x)}, \end{aligned} \tag{2.3}$$

where $S(\theta) = \{x \in L : \max_e p_e(x) \geq \rho\}$ is the instantiated object support. A "background model" involves the assumption that every location in the image corresponds to the background. The likelihood ratio between the deformable model and the background model is

expressed as

$$\frac{\mathbb{P}(X|\theta)}{\mathbb{P}(X|background)} = \prod_{x\in S(\theta)} \prod_{e} (\frac{\mathbb{P}(X_e(x)=1|\theta)}{p_{e,bgd}})^{X_e(x)} \times (\frac{1-\mathbb{P}(X_e(x)=1|\theta)}{1-p_{e,bgd}})^{1-X_e(x)}. \tag{2.4}$$

The overall shift r is assumed to follow a uniform prior distribution, and the prior on $v = (v_1, v_2, ..., v_n)$ is a multivariate Gaussian with zero means. Additionally, the POP model accommodates multiple subclasses within each class. For each class $c = 1, 2, ..., C$ and subclass $m = 1, ..., M_c$, the prior density of θ is denoted as $f_{c,m}(\theta)$. Consequently, the probability of the image X modeled under class c is given by

$$\mathbb{P}_c(X) = \sum_{m=1}^{M_c} \mathbb{P}_c(m) \int \mathbb{P}_{c,m}(X|\theta) f_{c,m}(\theta) d\theta, \tag{2.5}$$

where $\mathbb{P}_c(m)$ is the prior for subclass m and $\mathbb{P}_{c,m}(X|\theta)$ is the probability defined in Equation 2.3 under class c and subclass m.

If there are k objects in the image, sorted based on their occlusion sequence from the most visible to the least visible, they can be represented as

$$\mathbf{I} = (c_i, m_i, \theta_i)_{i=1,2,...,k}.$$

For simplicity, let S_i denote the object support of the i-th object, and $T_i = \cup_{j=1}^{i} S_j$. Then, the visible part of the i-th object is given by $S_i \setminus T_{i-1}$, reflecting its occlusion sequence. The likelihood ratio in Equation 2.4 can be extended to

$$\begin{aligned}\frac{\mathbb{P}(X|\mathbf{I})}{\mathbb{P}(X|background)} = &\prod_{i=1}^{k} \prod_{e} \prod_{x\in S_i\setminus T_{i-1}} (\frac{\mathbb{P}_{c_i,m_i}(X_e(x)=1|\theta_i)}{p_{e,bgd}})^{X_e(x)} \\ &\times (\frac{1-\mathbb{P}_{c_i,m_i}(X_e(x)=1|\theta_i)}{1-p_{e,bgd}})^{1-X_e(x)}.\end{aligned} \tag{2.6}$$

Assuming independence between r and v, the term $f_{c,m}(\theta) = f_{c,m}((r, v))$ is proportional to the prior on v due to the uniform distribution of r. In the context of the classification task for an isolated object, the assumption is made that $r = 0$. The determination of the class $\hat{c}$ entails maximizing the posterior

$$\hat{c} = \underset{c}{\operatorname{argmax}} \max_{1 \leq m \leq M_c} \max_{v} \mathbb{P}_{c,m}(X|(0, v)) f_{c,m}((0, v)), \tag{2.7}$$

where an approximate solution can be attained through iterative maximization with respect to v_i. Regarding detection, a declaration of detection at location r is made if the maximized posterior at that location surpasses a predetermined threshold τ_c. The maximized posterior is defined as

$$\mathcal{J}(r) = \max_{1 \leq m \leq M_c} \max_{v} \mathbb{P}_{c,m}(X|(r, v)) f_{c,m}((r, v)). \tag{2.8}$$

In the context of multi-object configurations, the POP model makes the assumption that the number of objects, denoted as k, is known. The POP model maximize a posterior

$$\mathbb{P}(\mathbf{I}|X) \propto h(r_1, r_2, ..., r_k) \prod_{i=1}^{k} g_{c_i,m_i}(v_i) \frac{\mathbb{P}(X|\mathbf{I})}{\mathbb{P}(X|background)}, \tag{2.9}$$

where $\frac{\mathbb{P}(X|\mathbf{I})}{\mathbb{P}(X|background)}$ is calculated in Equation 2.6, $h(r_1, r_2, ..., r_k)$ is the prior on the shifts of the objects, and $g_{c_i,m_i}(v_i)$ is the prior on v_i under class c_i and subclass m_i. To mitigate computational complexity in this task, it is common to employ either greedy iterations or dynamic programming. The POP model necessitates only minimal training sets and, in contrast to numerous purely feed-forward methods, remains applicable even when the test set significantly differs from the training set.

CHAPTER 3

DETECTION SELECTION ALGORITHM FOR OBJECT DETECTION

3.1 Motivation

Object detection Zou et al. [2023], Amit [2002] stands out as a paramount task within the realm of computer vision. The majority of object detection algorithms yield predictions in the form of bounding boxes, specifying the center, height, and width for each. Concurrently, these algorithms engage in the classification of the object contained within each bounding box. Certain object detection algorithms, such as Faster R-CNN Ren et al. [2015], go a step further by furnishing an objectness score. This score quantifies the level of confidence associated with the presence of an object within the corresponding bounding box.

Typically, object detection algorithms begin by generating excessive detections, subsequently employing post-processing techniques such as Non-maximum Suppression (NMS) to curtail this surplus. NMS operates by retaining the most promising detections via local comparisons. To execute NMS effectively, a NMS-threshold $N_t \in [0, 1]$ is utilized, determining the point at which less promising neighboring bounding boxes should be suppressed. However, following the NMS process, the remaining bounding boxes might not necessarily exhibit high objectness scores, often resulting in a surplus of bounding boxes exceeding the actual count of objects. In Faster R-CNN Ren et al. [2015], the top-N bounding boxes subsequent to NMS are designated as the final detections. Nevertheless, given that the number of objects within an image is frequently unknown, the parameter N is generally set to a value larger than the actual count of objects.

In contrast to Non-maximum Suppression (NMS), Soft-NMS Bodla et al. [2017b] does not directly eliminate less promising neighboring bounding boxes. Instead, it diminishes their objectness scores based on the Intersection-over-Union (IoU). Subsequent to the application

of Soft-NMS, it remains necessary to impose constraints, such as a maximum allowable number of boxes or a lower bound threshold on the objectness scores of detections. In the context of the Soft-NMS Bodla et al. [2017b] paper, the practice involves utilizing the top 400 detections per image on MS-COCO.

To find the correct number of objects and labels in the image, one natural idea is to use a threshold T on the objectness scores. Bounding boxes with scores surpassing the threshold T are selected as the final detections. Ideally, the threshold T is established through validation set analysis. However, the robustness of this approach is contingent upon the validation set possessing a distribution comparable to that of the test set. A disparity in distribution between the validation and test sets can render the outcome highly sensitive to the chosen threshold T, potentially resulting in considerable degradation of model performance.

Our work proposes a novel post processing method for object detection algorithms building on the work in Amit and Trouvé [2007]. The core concept involves identifying the most probable "interpretation" of an image, where an interpretation denotes an ordered subset of detections organized by occlusion. Each object class is characterized by a generative model, mapping a low-dimensional latent space to the image space. The pixel values are assumed to be independently Gaussian conditioned on the latent variables. The generative model serves to offer both a reconstruction of the object image and a region of object support. The log-likelihood of the entire image conditional on the objects, their locations and the values of the latent variables is the sum of the log-likelihoods of the individual objects *on their visible parts*. This underscores the significance of occlusion ordering, given that objects positioned behind are only visible outside the support regions of those in the forefront.

Optimizing over an ordered subset of detections proves computationally prohibitive, leading us to adopt a greedy search strategy. Leveraging the objectness scores provided by Faster R-CNN facilitates this process. Additionally, we augment Faster R-CNN with an extra branch, denoted Faster R-CNN-OC, which furnishes an occlusion score within the

range of 0 to 1. A higher occlusion score indicates precedence for an object positioned in front of another with a lower occlusion score. These outputs play a pivotal role in informing the greedy search for the most plausible interpretation, as delineated below. In summary, the Detections Selection Algorithm (DSA) entails a greedy search across ordered subsets of detections to identify the subset with the highest likelihood, utilizing both objectness and occlusion scores supplied by the Faster R-CNN-OC.

For the sake of simplicity, this chapter assumes that the image I is composed of objects situated against a pure black background. The examination of real-life images featuring diverse backgrounds is deferred to Chapter 4. The detection process involves the application of an algorithm, such as Faster R-CNN-OC, to obtain detections. The detections are represented by $\{\mathbf{det}_i = (score_i, bb_i, occ_i, cls_i)\}_{i=1}^{N}$. For each detection $\mathbf{det}_i$, $score_i$ is the objectness score defined in Faster R-CNN Ren et al. [2015], bb_i is the bounding box, cls_i is the classification result, and occ_i is the occlusion score obtained from the occlusion branch of Faster R-CNN-OC. We denote by $I[bb_i]$ the image restricted to the bounding box bb_i. If the objects in $\mathbf{det}_i$ and $\mathbf{det}_j$ overlap, and $occ_i < occ_j$, the object in $\mathbf{det}_i$ is predicted to be occluded in the overlapping area by the object in $\mathbf{det}_j$. Typically there are false positive detections among all detections $\{(score_i, bb_i, cls_i)\}_{i=1}^{N}$ produced by the Faster R-CNN-OC. In other words, only a subset of $\{(score_i, bb_i, cls_i)\}_{i=1}^{N}$ is correct. Our goal is to find the ordered subset $\{\mathbf{det}_{i_j} = (score_{i_j}, bb_{i_j}, occ_{i_j}, cls_{i_j})\}_{j=1}^{k}$ which yields the best interpretation of the image, namely the lowest negative log likelihood (NLL). But, trying every possible ordered subset of $\{\mathbf{det}_i\}_{i=1}^{N}$ is computational prohibitive. Our proposed Detections Selection Algorithm (DSA) greedily selects the detections when processing them according to their objectness scores from high to low, taking into account the occlusion scores to identify the visible parts of each object.

The first component of our method is a Single Reconstruction Algorithm for DSA (DSASR) tasked with reconstructing an entire object based on its visible components. This nomen-

clature, DSASR, is assigned to this algorithm due to its specialized design for integration into our Detection Selection Algorithm (DSA). In this context, we adopt a decoder architecture reminiscent of Variational Autoencoders (VAEs). Diverging from traditional VAEs, the reconstruction process is optimized over latent variables rather than relying on variables predicted by an encoder. This modification is crucial due to the inherent limitation of not always observing the entirety of an object. The reconstruction loss during latent code optimization, quantified as the negative log-likelihood (NLL), is exclusively computed for the visible portion. Adapting the encoder to varying visible inputs seems prohibitive to us. The Single Reconstruction Algorithm for DSA takes a bounding box and its associated information, as well as reconstructions of previous objects in the sequence as input, and returns the whole appearance of the hypothesized object for that bounding box. A byproduct of the Single Reconstruction Algorithm for DSA (DSASR) is that it performs amodal instance segmentation using the reconstruction.

The second component of our method is the Whole Reconstruction Algorithm for DSA (DSAWR). It consolidates outcomes from the DSASR applied to the current sequence of selected objects. This amalgamation provides the negative log-likelihood (NLL) for the entire image data given this sequence.

The final component - Detection Selection Algorithm (DSA), systematically explores detections supplied by the Faster R-CNN-OC. The search operates in a greedy fashion, ordered based on their objectness scores. At each iteration, a detection is added and reconstructed from its visible part, computed as the complement within its bounding box, excluding the union of supports of previously reconstructed detections with higher occlusion scores. If our defined loss function of the complete reconstruction diminishes, the additional object is incorporated; otherwise, it is omitted. Additionally, we conduct a one-step backward search to assess the potential reduction in the loss by removing a previously incorporated object while simultaneously introducing a new object. A fixed penalty term is introduced for each

added object, equivalent to an exponential prior on the number of objects. Eventually, the decrease in loss function attributable to an additional object is offset by the object penalty, leading to termination of the search.

Our idea to evaluate the likelihoods of an image under various object hypotheses is inspired by the POP model Amit and Trouvé [2007]. However, it is essential to note distinctions in our approach. While the POP model employs a deformable template for object modeling, we utilize a more flexible decoder structure. Additionally, the POP model necessitates the determination of occlusion ordering as part of its optimization process, whereas we leverage the output of the Faster R-CNN-OC to acquire occlusion ordering. Furthermore, our algorithm is applicable to images featuring 3D objects, in contrast to the POP model, which is confined to 2D objects.

The main contribution of this chapter is the DSA, DSASR, DSAWR algorithms. Our primary objective is to ascertain the precise count of objects and their corresponding labels within an image, predicated on achieving the lowest loss through comprehensive reconstruction. Several ancillary outcomes emanate from this pursuit:

- The Faster R-CNN-OC incorporates an occlusion ordering mechanism. Remarkably, our findings indicate that training the Faster R-CNN-OC solely on object pairs yields outstanding detections for scenes featuring multiple objects, accompanied by highly reliable occlusion scores.

- The Single Reconstruction Algorithm for DSA (DSASR) reconstructs imperceptible portions of objects, thereby offering a straightforward solution for amodal instance segmentation.

- The Whole Reconstruction Algorithm for DSA (DSAWR) furnishes a means to generate an image based on several hypothesized objects and their respective locations.

The subsequent sections of this chapter are structured as follows: The introduction of

Faster R-CNN-OC is presented in Section 3.2. The probabilistic framework and the delineation of three algorithms are expounded upon in Section 3.3, 3.4, 3.5, and 3.6. Section 3.7 provides an account of our dataset, while Section 3.8 details the experiments conducted. Our code is accessible on the GitHub repository at the following URL: `https://github.com/angzhifan/DSA_research`

3.2 Faster R-CNN-OC

Faster R-CNN Ren et al. [2015] is a detection framework which uses a Region Proposal Network (RPN) to generate region proposals. Subsequently, the Fast-RCNN Girshick [2015a] module is employed for bounding box regression and classification for each region proposal. The RPN yields an objectness score for each bounding box, where a higher score signifies a more confident detection.

We introduce an additional branch named the occlusion branch to Faster R-CNN, situated in parallel with the regression branch and the classification branch inside the Fast-RCNN module. The resulting model, denoted as Faster R-CNN-OC, is illustrated in Figure 3.1. The occlusion branch is composed of a fully-connected layer, and the output of this layer undergoes a sigmoid function to yield an occlusion score ranging between 0 and 1. As previously mentioned, when two objects overlap, the one with the higher occlusion score is anticipated to be visible in the overlapping area. During the inference phase, the comparison of occlusion scores is sufficient for determining the occlusion sequence. One approach to training the occlusion branch involves providing pairs of overlapping objects and assigning an occlusion score of 0 to the occluded object and 1 to the occluding object. Further details will be expounded upon in Section 3.8.

The occlusion branch has demonstrated consistent and reliable outcomes in our experiments. Its training was conducted on images featuring two objects, and we observed robust generalization performance to images containing multiple objects. Additional experiments

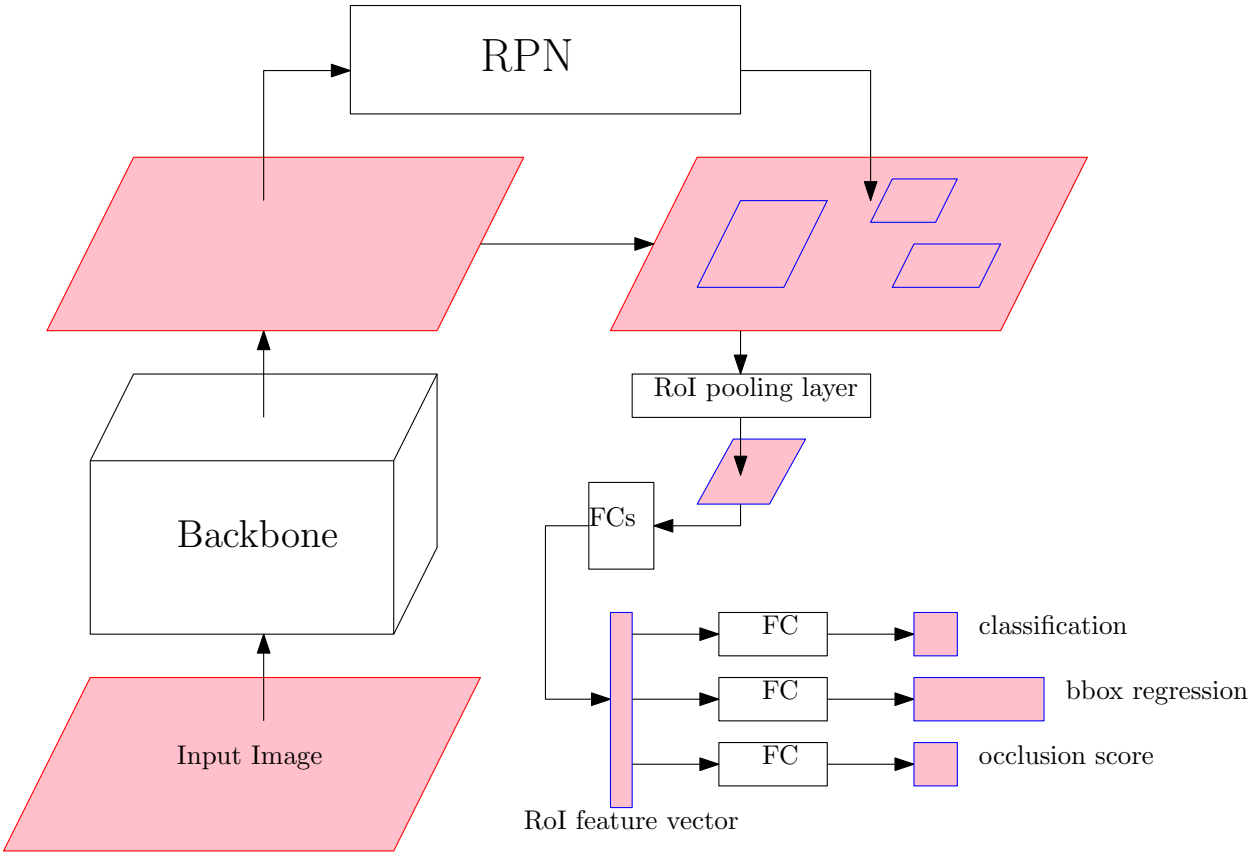


Figure 3.1: Faster R-CNN-OC network architecture.

pertaining to this aspect are detailed in Section 3.8.1.

3.3 Single Reconstruction Algorithm for DSA (DSASR)

Our Single Reconstruction Algorithm for DSA (DSASR) is based on a generative model structured similarly to a decoder in a Variational Autoencoder (VAE) Kingma and Welling [2013]. In the training phase of a complete VAE, the objective is to maximize the variational evidence lower bound (ELBO) on the marginal likelihood

$$\mathcal{L}(\theta, \phi; x) = \mathbf{E}_{q_\phi(z|x)}(\log p_\theta(x|z)) - \mathbf{D}_{KL}(q_\phi(z|x)||p(z)) \tag{3.1}$$

where the symbols ϕ and θ represent the parameters of the encoder and decoder, respectively, while x denotes the target image segment. For simplicity, assume $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{N_z})$, $x|z \sim \mathcal{N}(m_{\theta,z}, \sigma^2\mathbf{I})$ and $q_\phi(z|x)$ is the density of $\mathcal{N}(\mu_x, \Gamma_x)$, where $m_{\theta,z}$ is the output of the decoder given latent code z, μ_x is a vector, and Γ_x is a diagonal matrix

$$\Gamma_x = \begin{bmatrix} \tau_{x,1}^2 & & \\ & \ddots & \\ & & \tau_{x,N_z}^2 \end{bmatrix} \tag{3.2}$$

with $\tau_{x,i} > 0$ for $i = 1, 2, ..., N_z$.

During the inference phase, when a portion of the object is potentially occluded, the reconstruction process relies solely on the visible part. This circumstance poses challenges for the encoder in predicting μ_x and Γ_x. To address this, both during training and inference, we optimize over μ_x and Γ_x. Consequently, the VAE encoder is omitted from the training process, and only the decoder is trained. Post-training, we can determine μ_x and Γ_x for an incomplete object. Subsequently, by passing $z = \mu_x$ to the decoder, the complete appearance of the object can be reconstructed.

In the initial part of this section, an assumption is made that the target bounding box shares the same dimensions as the images used for training the decoder. However, in practice, the target bounding boxes may vary in size. To address this discrepancy, we employ the concept of the parameterised sampling grid of Spatial Transformer Networks Jaderberg et al. [2015]. This approach is elaborated upon in the latter half of this section.

3.3.1 Fixed-size Reconstructions

The decoder is trained utilizing the methodology of stochastic variational inference (SVI) Hoffman et al. [2013]. Instead of relying on the encoder to predict μ_x and Γ_x, these variables are updated through a fixed number of optimization steps employing gradient descent.

Subsequently, μ_x and Γ_x are held constant while updating the decoder parameters θ. This iterative process continues until convergence. Notably, our decoder is trained on fixed-size images within the training dataset, where each image contains a singular object, and separate models are trained for each class.

During the reconstruction process for a designated target bounding box $\tilde{x}$, with the visible pixel set denoted as V, we perform optimization on $\mu_{\tilde{x}}$ and $\Gamma_{\tilde{x}}$. This optimization aims to maximize Equation 3.1 based on the visible segment, while maintaining a fixed decoder configuration. To be more specific, Equation 3.1 becomes

$$\begin{aligned}
\mathcal{L}(\theta, \phi; \tilde{x}, V) =& \mathbf{E}_{q_\phi(z|\tilde{x})}(\log p_{\theta,V}(\tilde{x}|z)) - \mathbf{D}_{KL}(q_\phi(z|\tilde{x})||p(z)) \\
=& \mathbf{E}_{\mathcal{N}(\mu_{\tilde{x}},\Gamma_{\tilde{x}})}(\log p_{\theta,V}(\tilde{x}|z)) - \mathbf{D}_{KL}(\mathcal{N}(\mu_{\tilde{x}},\Gamma_{\tilde{x}})||\mathcal{N}(\mathbf{0},\mathbf{I}_{N_z})) \\
=& \mathbf{E}_{\mathcal{N}(\mu_{\tilde{x}},\Gamma_{\tilde{x}})}(-\frac{|V|}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i\in V}(m_{\theta,z,i} - \tilde{x}_i)^2) \\
& - (\frac{\|\mu_{\tilde{x}}\|_2^2 + \sum_{i=1}^{N_z}\tau_{\tilde{x},i}^2 - N_z}{2} - \sum_{i=1}^{N_z}\log\tau_{\tilde{x},i})
\end{aligned} \tag{3.3}$$

then we can get our reconstruction as

$$\hat{\tilde{x}} = m_{\theta,z_{\tilde{x}}^*} \tag{3.4}$$

where the variable $z_{\tilde{x}}^*$ is set equal to $\mu_{\tilde{x}}$, and the notation $m_{\theta,z}$ is defined earlier in this section. As the decoder is trained using complete objects, the resulting output encompasses an entire object.

3.3.2 Arbitrary-size Reconstructions

Our target image corresponds to the visible segment of everything enclosed within the bounding box. However, the dimensions of the bounding box typically do not align with the size

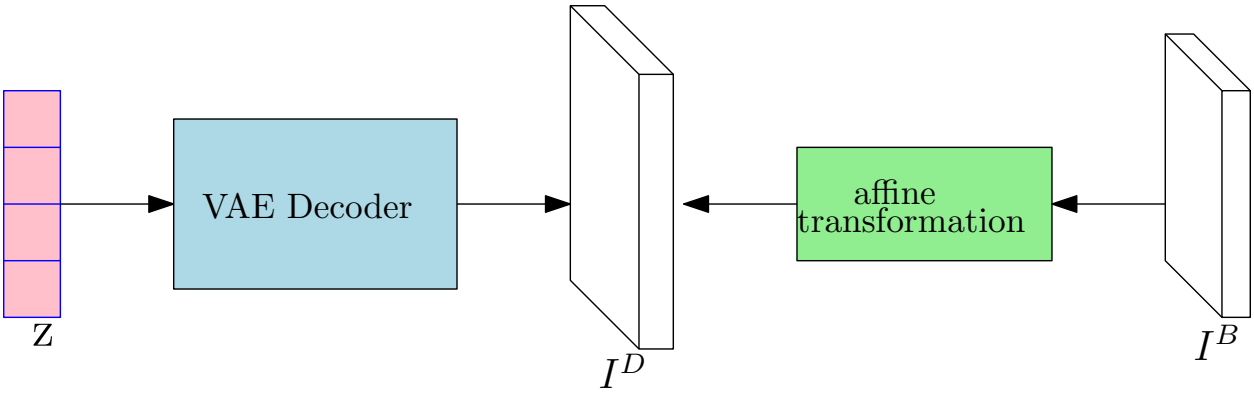


Figure 3.2: Decoder architecture.

of the training images for the Variational Autoencoder (VAE). To address this misalignment, we adopt the concept of the parameterized sampling grid by the Spatial Transformer Network Jaderberg et al. [2015]. A similar concept is also presented in Gregor et al. [2015].

If our designated target image box is I_{i_j}, acquired by cropping the full image within the bounding box region bb_{i_j}, we presume the existence of an affine transformation between I_{i_j} and the output of the decoder. For any coordinate (x^b, y^b) within our reconstruction of the target image box, after the affine transformation we get

$$\begin{pmatrix} x^d \\ y^d \end{pmatrix} = \begin{pmatrix} s_x & 0 & t_x s_x \\ 0 & s_y & t_y s_y \end{pmatrix} \begin{pmatrix} x^b \\ y^b \\ 1 \end{pmatrix} \tag{3.5}$$

and for each channel, the coordinate (x^b, y^b) in our reconstruction should have the same image value as coordinate (x^d, y^d) in the decoder output. As shown in Figure 3.2, if our reconstruction is I^B and the decoder output is I^D, $I^B(m, n, c)$ and $I^D(m, n, c)$ is the pixel value at coordinate (m, n) and channel c in image I^B and I^D respectively, then

$$I^D(x_i^d, y_i^d, c) = I^B(x_i^b, y_i^b, c). \tag{3.6}$$

For simplicity, and to avoid too much flexibility, we do not consider rotation in the affine

transformation. The shearing parameters s_x, s_y represent scaling in the x and y axis. We assume an isotropic scaling and fix $s_x = s_y = d/L$, where $(d, d, 3)$ is the VAE training image size and L is the maximum between the height $H_{bb_{i_j}}$ and width $W_{bb_{i_j}}$ of the target bounding box bb_{i_j}

$$L = \max(H_{bb_{i_j}}, W_{bb_{i_j}}).$$

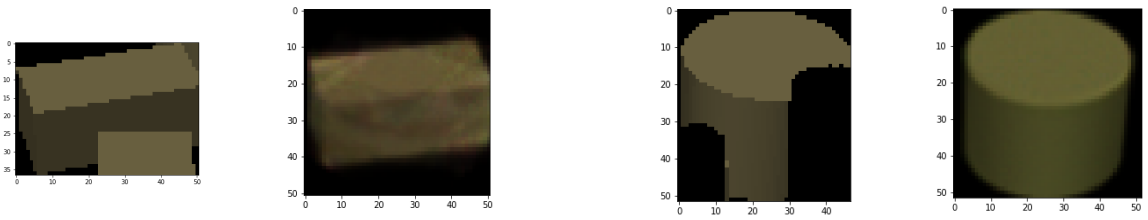
The t_x, t_y translation parameters are kept free. The coordinate (x^b, y^b) can be any integer coordinate in our reconstruction. However, after the transformation, we get the corresponding (x^d, y^d), which may not be integers. We utilize the bilinear sampling kernel in the Spatial Transformer Network Jaderberg et al. [2015] to interpolate for coordinate (x^d, y^d). The bilinear sampling kernel is formulated as

$$I^B(x, y, c) = \sum_{m=1}^{d} \sum_{n=1}^{d} I^D(m, n, c) \max(0, 1 - |x - m|) \max(0, 1 - |y - n|) \tag{3.7}$$

In this case, our reconstruction is an $L \times L \times 3$ image, so we need to crop a $H_{bb_{i_j}} \times W_{bb_{i_j}} \times 3$ region at the center of the $L \times L \times 3$ image to get our reconstruction for the target image bounding box.

Given the presence of the affine transformation and bilinear sampling kernel, gradients retain the ability to back-propagate from the target image to the latent code. Through this process, we assess the disparity between the target image and our reconstruction for the designated bounding box. Consequently, we obtain the latent codes along with t_x and t_y using gradient descent.

In summary, for target image box I_{i_j}, we fix s_x, s_y but optimize μ_{i_j}, τ_{i_j} and (t_x, t_y), where μ_{i_j} and τ_{i_j} are the variables in the posterior distribution $z|I_{i_j}, V_{i_j} \sim \mathcal{N}(\mu_{i_j}, \Gamma_{i_j})$. Given μ_{i_j}, Γ_{i_j}, (t_x, t_y) and (s_x, s_y), we use $z = \mu_{i_j}$, pass it to the decoder to get a $d \times d \times 3$ decoder output, and use the affine transformation and bilinear sampling kernel to get a $L \times L \times 3$ reconstruction called R_{i_j}. This procedure from z, (t_x, t_y) and (s_x, s_y) to R_{i_j} is represented



(a) (b) (c) (d)

Figure 3.3: Example of the DSASR. (a), (c) - target images with occluded parts in black, (b), (d) - reconstructions R_{ij} on the $L \times L$ grid.

by the function "Decoder(z, (t_x, t_y), (s_x, s_y))" in Algorithm 1.

In addition to R_{i_j}, Algorithm 1 also yields $bb^*_{i_j}$, a bounding box of dimensions $L \times L \times 3$ centered at the midpoint of the target bounding box bb_{i_j}. Given that R_{i_j} has dimensions $L \times L \times 3$, when situating R_{i_j} on the entire image domain, it should extend to the larger bounding box $bb^*_{i_j}$ rather than bb_{i_j}. Our single reconstruction R_{i_j} represents an object on a pure black background. The object's support is defined as all pixels with magnitudes exceeding a predefined threshold t_0, referred to as the "occlusion threshold." As Algorithm 1 is designed to reconstruct a solitary object based on a single bounding box, it is aptly termed the Single Reconstruction Algorithm.

An illustration of the Single Reconstruction Algorithm for DSA is presented in Figure 3.3, with a chosen parameter $d = 50$. In Figure 3.3, image (a) depicts the cropped target image box containing clutter. Image (b) represents the $L \times L \times 3$ single reconstruction corresponding to the target image in (a). Analogously, image (d) serves as the single reconstruction for the target image in (c), with the distinction that (c) is now a partially occluded image.

3.4 Whole Reconstruction Algorithm for DSA (DSAWR)

The Whole Reconstruction Algorithm for DSA (DSAWR), see Algorithm 2, is employed to amalgamate the single reconstructions of a sequentially ordered subset of detections on a blank background called *Canvas*, identical in size to the image, in accordance with their

Algorithm 1: Single Reconstruction Algorithm for DSA (DSASR)

Input: Cropped target image $I_{i_j} = I[bb_{i_j}]$, V_{i_j}: the coordinates of the visible pixels in I_{i_j}, the detection $\mathbf{det}_{i_j} = (score_{i_j}, bb_{i_j}, occ_{i_j}, cls_{i_j})$, VAE training image size $(d, d, 3)$, N_{iter}, N_z and σ

$L \leftarrow max(H_{I_{i_j}}, W_{I_{i_j}})$;

$\mu_{i_j} \leftarrow zeros(N_z)$; `/* ` N_z `is the dimension of the latent code */`

$(\log \tau_{i_j,1}, \log \tau_{i_j,2}, ..., \log \tau_{i_j,N_z}) \leftarrow zeros(N_z)$; `/* ` $(\tau^2_{i_j,1}, \tau^2_{i_j,2}, ..., \tau^2_{i_j,N_z})$ `is the diagonal of covariance matrix` Γ_{i_j} `*/`

$(t_x, t_y) \leftarrow (0, 0)$; `/* translation parameters */`

$(s_x, s_y) \leftarrow (d/L, d/L)$; `/* shearing parameters, fixed */`

for $j = 1$ **to** N_{iter} **do**

$z \leftarrow$ sampled from $\mathcal{N}(\mu_{i_j}, \Gamma_{i_j})$;

$R_{i_j} \leftarrow$ Decoder(z, (t_x, t_y), (s_x, s_y)) ; `/* ` R_{i_j} `has size` $(L, L, 3)$ `*/`

$R^{(bb)}_{i_j} \leftarrow$ Cropped region of size $(H_{I_{i_j}}, W_{I_{i_j}}, 3)$ at the center of R_{i_j} ;

$Loss \leftarrow \mathbb{D}_{KL}(\mathcal{N}(\mu_{i_j}, \Gamma_{i_j}) || \mathcal{N}(0, \mathbf{I}_{N_z})) + \frac{1}{2\sigma^2} \sum_{x \in V_{i_j}} (R^{(bb)}_{i_j,x} - I_{i_j,x})^2$;

Update μ_{i_j}, $(\log \tau_{i_j,1}, \log \tau_{i_j,2}, ..., \log \tau_{i_j,N_z})$ and (t_x, t_y) based on gradients of $Loss$;

end

$R_{i_j} \leftarrow$ Decoder(μ_{i_j}, (t_x, t_y), (s_x, s_y)) ;

$bb^*_{i_j} \leftarrow$ an $L \times L \times 3$ bounding box which centers at the center of bounding box bb_{i_j};

Output: Single reconstruction R_{i_j}, parameters μ_{i_j}, Γ_{i_j} and the inferred bounding box $bb^*_{i_j}$

respective occlusion scores. As previously mentioned, if the support regions of two objects intersect, the object visible in the overlapping area is determined by the one with the higher occlusion score.

In Algorithm 2, we iterate through the ordered subset of detections. If the single reconstruction for the current detection has not been computed, the Single Reconstruction Algorithm for DSA is implemented. This involves determining which pixels within the current bounding box are visible, taking into account all pre-computed single reconstructions with higher occlusion scores: Objects with superior occlusion scores are placed on the background. After that, the blank segment on the background is assumed to remain visible.

As mentioned in the Single Reconstruction Algorithm for DSA (DSASR) in Section 3.3, a pre-determined threshold t_0 is used to determine which pixels constitute the support of the object in its single reconstruction. Only those pixels falling within the support region adopt the values of the reconstruction; the remaining pixels remain blank.

Figure 3.4 illustrates an example of Algorithm 2. The original image featuring 5 objects is presented in (a). Within (a), there are 6 detected bounding boxes labeled at their lower-right corners from 0 to 5, arranged in descending order of their objectness scores. Images (b), (c), and (d) portray the whole reconstruction canvases corresponding to the selected detections $\{0, 1, 2\}$, $\{0, 1, 2, 3\}$, and $\{0, 1, 2, 3, 4\}$ respectively. The single reconstructions from Figure 3.3 are utilized in this context. Moving from (b) to (c), the cylinder within bounding box 3 becomes occluded by the two cuboids in bounding boxes 1 and 2. Image (c) in Figure 3.3 represents an incomplete cylinder due to retaining only the visible portion.

It is noteworthy that the single reconstruction has dimensions $(L, L, 3)$ and is typically larger than the size of the target bounding box. Nevertheless, we do not confine the single reconstruction within the boundaries of the target bounding box. In other words, if the support region of the object extends beyond the target bounding box, we still include those pixels in the canvas during the Whole Reconstruction Algorithm for DSA. Our rationale

Algorithm 2: Whole Reconstruction Algorithm for DSA (DSAWR)

Input: A subset of detection results $\{\mathbf{det}_{i_j} = (score_{i_j}, bb_{i_j}, occ_{i_j}, cls_{i_j})\}_{j=1}^{k}$, reconstructions hashmap *ReconDict*, occlusion threshold t_0

Sort $\{\mathbf{det}_{i_j}\}_{j=1}^{k}$ according to occ_{i_j} from high to low;

$Canvas \leftarrow zeros(H, W, 3)$; `/* (H, W, 3) is the image size */`

for $j = 1$ **to** k **do**

  if *ReconDict*$[i_j]$ *doesn't exist* **then**

    $I_{i_j} \leftarrow I[bb_{i_j}]$; `/* I is the image and` $I[bb_{i_j}]$ `is obtained by cropping the image at bounding box` bb_{i_j} `*/`

    $V_{i_j} \leftarrow$ The coordinates of blank pixels in $Canvas[bb_{i_j}]$; `/* "blank pixel" is a pixel with value` $(0, 0, 0)$ `*/`

    $(R_{i_j}, \mu_{i_j}, \Gamma_{i_j}, bb^*_{i_j}) \leftarrow DSASR(I_{i_j}, V_{i_j}, \mathbf{det}_{i_j})$;

    $ReconDict[i_j] \leftarrow (R_{i_j}, \mu_{i_j}, \Gamma_{i_j}, bb^*_{i_j})$;

  end

  $(R_{i_j}, \mu_{i_j}, \Gamma_{i_j}, bb^*_{i_j}) \leftarrow ReconDict[i_j]$;

  l_{i_j}, r_{i_j} - coordinates of upper left hand corner of $bb^*_{i_j}$;

  for *pixel* x, y *in* R_{i_j} **do**

    if $Canvas[l_{i_j} + x, r_{i_j} + y]$ *is* $(0, 0, 0)$ *and the magnitude of* $R_{i_j}[x, y]$ *is above* t_0 **then**

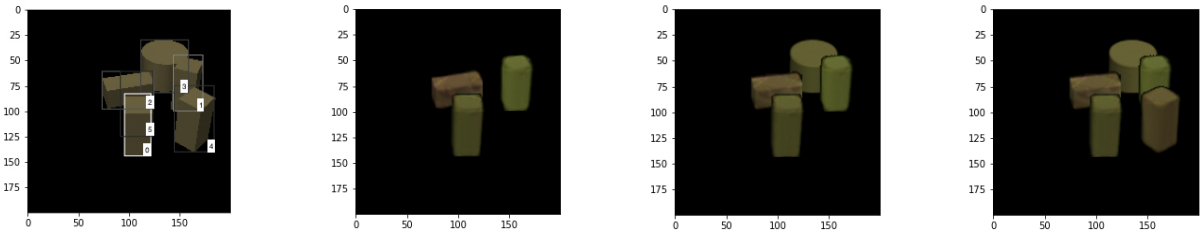
      $Canvas[l_{i_j} + x, r_{i_j} + y] \leftarrow R_{i_j}[x, y]$;

    end

  end

end

Output: Whole reconstruction *Canvas*, reconstructions hashmap *ReconDict*



(a) (b) (c) (d)

Figure 3.4: Example of DSAWR.

for this approach is that if the reconstructed object genuinely exists, its portion outside the target bounding box should also be considered.

3.5 Probabilistic Framework

In our approach, an effective interpretation of the image is expected to fulfill two primary objectives. Firstly, we aim to achieve a high-quality reconstruction of the entire image based on our chosen detections. Secondly, we seek to prevent the selection of redundant detections. These twin objectives serve as the motivation for the ensuing probabilistic framework.

Suppose a detection algorithm yields detection results $\{\mathbf{det}_i = (score_i, bb_i, occ_i, cls_i)\}_{i=1}^{N}$, and $\{\mathbf{det}_{i_j} = (score_{i_j}, bb_{i_j}, occ_{i_j}, cls_{i_j})\}_{j=1}^{k}$ is a subset used to interpret the image. For the quantity of detections k within the subset, we posit a prior distribution $p_K(k) \propto e^{-\lambda_0 k}$ for $k \geq 0$. For each detection $\mathbf{det}_i$, its latent code z_i adheres to a Gaussian prior $z_i \sim \mathcal{N}(0, \mathbf{I}_{N_z})$. We adopt a non-informative prior for each detection $\mathbf{det}_i = (score_i, bb_i, occ_i, cls_i)$, rendering the prior for objects $\{\mathbf{det}_{i_j}\}_{j=1}^{k}$ equal to the prior for the number of objects $p_K(k)$, or equivalently, $p(\{\mathbf{det}_{i_j}\}_{j=1}^{k}) = p_K(k)$.

Given $\{\mathbf{det}_{i_j} = (score_{i_j}, bb_{i_j}, occ_{i_j}, cls_{i_j})\}_{j=1}^{k}$ and $\{z_{i_j}\}_{j=1}^{k}$, we assume the distribution of the hypothesized image follows a Gaussian distribution with a uniform variance σ^2 across all pixels. Additionally, we assume pixel-level independence given the mean. The mean of this Gaussian distribution is stipulated to be the reconstruction generated by the canvas of

the Whole Reconstruction Algorithm for DSA (DSAWR).

Therefore, if the image is I, the log marginal likelihood of the interpretation is

$$\begin{aligned}
&\log p(I, \{\mathbf{det}_{i_j}\}_{j=1}^k) \\
&= \log p_K(k) + \log p(I|\{\mathbf{det}_{i_j}\}_{j=1}^k) \\
&= \log p_K(k) + \log \int\int\cdots\int p(I|\{z_{i_j}\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) \prod_{j=1}^k p(z_{i_j}) dz_{i_1} dz_{i_2} \cdots dz_{i_k},
\end{aligned} \tag{3.8}$$

which is intractable in terms of computation. Using the well known variational approximation

$$\begin{aligned}
&\log p(I|\{\mathbf{det}_{i_j}\}_{j=1}^k) \\
=&\mathbf{E}_{q_\phi(\{z_{i_j}\}_{j=1}^k|I,\{\mathbf{det}_{i_j}\}_{j=1}^k)} \log \frac{p(I, \{z_{i_j}\}_{j=1}^k|\{\mathbf{det}_{i_j}\}_{j=1}^k)}{q_\phi(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k)} \\
&+ \mathbf{D}_{KL}(q_\phi(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k)||p(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k)) \\
=&\mathbf{E}_{q_\phi(\{z_{i_j}\}_{j=1}^k|I,\{\mathbf{det}_{i_j}\}_{j=1}^k)} \log p(I|\{z_{i_j}\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) \\
&- \mathbf{D}_{KL}(q_\phi(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k)||p(\{z_{i_j}\}_{j=1}^k|\{\mathbf{det}_{i_j}\}_{j=1}^k)) \\
&+ \mathbf{D}_{KL}(q_\phi(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k)||p(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k))
\end{aligned} \tag{3.9}$$

and

$$p(\{z_{i_j}\}_{j=1}^k|\{\mathbf{det}_{i_j}\}_{j=1}^k) = p(\{z_{i_j}\}_{j=1}^k), \tag{3.10}$$

we drop the last Kullback–Leibler divergence term and use

$$\begin{aligned}
&\mathbf{E}_{q_\phi(\{z_{i_j}\}_{j=1}^k|I,\{\mathbf{det}_{i_j}\}_{j=1}^k)} \log p(I|\{z_{i_j}\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) \\
&\qquad - \mathbf{D}_{KL}(q_\phi(\{z_{i_j}\}_{j=1}^k|I, \{\mathbf{det}_{i_j}\}_{j=1}^k)||p(\{z_{i_j}\}_{j=1}^k))
\end{aligned} \tag{3.11}$$

to approximate $\log p(I|\{\mathbf{det}_{i_j}\}_{j=1}^k)$.

As in previous sections, $I_{i_j} = I[bb_{i_j}]$ denotes the cropped image from I at bounding box

bb_{i_j}. Without loss of generality, we can assume $\{\mathbf{det}_{i_j}\}_{j=1}^{k}$ is sorted by occlusion scores from high to low. Then we use the following approximation

$$q_\phi(\{z_{i_j}\}_{j=1}^{k}|I,\{\mathbf{det}_{i_j}\}_{j=1}^{k}) \approx \prod_{j=1}^{k} q_\phi(z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j}) \tag{3.12}$$

where V_{i_j} denotes the visible pixels in the bounding box of object i_j taking into account the union of supports of reconstructions i_r, where $r = 1,\ldots,j-1$ (see Algorithm 2), and $q_\phi(z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j})$ is the posterior distribution of z_{i_j} given cropped image I_{i_j}, V_{i_j} and detection $\mathbf{det}_{i_j}$. Using $z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j} \sim \mathcal{N}(\mu_{i_j},\Gamma_{i_j})$, equation 3.11 can be approximated by

$$\begin{aligned}
&\mathbf{E}_{\prod_{j=1}^{k} q_\phi(z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j})} \log p(I|\{z_{i_j}\}_{j=1}^{k},\{\mathbf{det}_{i_j}\}_{j=1}^{k}) \\
&\quad - \sum_{j=1}^{k} \mathbf{D}_{KL}(q_\phi(z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j})||p(z_{i_j})) \\
=&\mathbf{E}_{\prod_{j=1}^{k} q_\phi(z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j})} \log p(I|\{z_{i_j}\}_{j=1}^{k},\{\mathbf{det}_{i_j}\}_{j=1}^{k}) - \sum_{j=1}^{k} \mathbf{D}_{KL}(\mathcal{N}(\mu_{i_j},\Gamma_{i_j})||\mathcal{N}(0,\mathbf{I}_{N_z})) \\
=&\mathbf{E}_{\prod_{j=1}^{k} q_\phi(z_{i_j}|I_{i_j},V_{i_j},\mathbf{det}_{i_j})} \log p(I|\{z_{i_j}\}_{j=1}^{k},\{\mathbf{det}_{i_j}\}_{j=1}^{k}) \\
&\quad - \sum_{j=1}^{k} \Big(\frac{\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - N_z}{2} - \sum_{t=1}^{N_z} \log \tau_{i_j,t}\Big)
\end{aligned} \tag{3.13}$$

In this way, we approximate the log marginal likelihood as

$$
\begin{aligned}
&\log p(I, \{\mathbf{det}_{i_j}\}_{j=1}^k) \\
\approx &\log p_K(k) + \mathbf{E}_{\prod_{j=1}^k q_\phi(z_{i_j}|I_{i_j}, V_{i_j}, \mathbf{det}_{i_j})} \log p(I|\{z_{i_j}\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) \\
&- \sum_{j=1}^k \left(\frac{\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - N_z}{2} - \sum_{t=1}^{N_z} \log \tau_{i_j,t}\right) \\
\approx &\log p_K(k) + \log p(I|\{z_{i_j}^*\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) - \sum_{j=1}^k \frac{\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - N_z}{2} + \sum_{j=1}^k \sum_{t=1}^{N_z} \log \tau_{i_j,t}
\end{aligned}
\tag{3.14}
$$

where $z_{i_j}^*$ is sampled from $\mathcal{N}(\mu_{i_j}, \Gamma_{i_j})$. In the Single Reconstruction Algorithm for DSA (DSASR), minimizing the loss gives us μ_{i_j}, Γ_{i_j} and (t_x, t_y). If the whole reconstruction using $\{z_{i_j}^*\}_{j=1}^k$ and $\{\mathbf{det}_{i_j}\}_{j=1}^k$ is $I_{\{i_1,\ldots,i_k\}}$, then

$$
\log p(I|\{z_{i_j}^*\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) = -\frac{|I|}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|I - I_{\{i_1,\ldots,i_k\}}\|_{vec,2}^2 \tag{3.15}
$$

It is crucial to underscore that Equation (3.15) furnishes a log-likelihood for the entire image. The output *Canvas* of the Whole Reconstruction Algorithm for DSA (DSAWR) yields the union of the supports of the selected objects in the image as the collection of all non-zero pixels. The complement of this set is regarded as the background, and the hypothesized distribution at each background pixel, based on the aforementioned equations, is simply $N(0, \sigma^2)$ in each channel.

3.6 Detection Selection Algorithm (DSA)

By our assumptions, $p_K(k) = \frac{e^{-\lambda_0 k}}{e^{\lambda_0}/(e^{\lambda_0}-1)}$ for $k \geq 0$. Based on our probabilistic framework, if $\{i_1, ..., i_k\}$ are the indices of the selected detections, which are used as an interpretation of the image, we have

$$
\begin{aligned}
&\log p(I, \{\mathbf{det}_{i_j}\}_{j=1}^k) \\
&\approx \log p_K(k) + \log p(I|\{z_{i_j}^*\}_{j=1}^k, \{\mathbf{det}_{i_j}\}_{j=1}^k) - \sum_{j=1}^{k} \frac{\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - N_z}{2} + \sum_{j=1}^{k}\sum_{t=1}^{N_z} \log \tau_{i_j,t} \\
&= \log \frac{e^{-\lambda_0 k}}{e^{\lambda_0}/(e^{\lambda_0}-1)} - \frac{|I|}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|I - I_{\{i_1,...,i_k\}}\|_{vec,2}^2 \\
&\quad - \sum_{j=1}^{k} \frac{\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - N_z}{2} + \sum_{j=1}^{k}\sum_{t=1}^{N_z} \log \tau_{i_j,t}
\end{aligned}
\tag{3.16}
$$

where $I_{\{i_1,...,i_k\}}$ is the whole reconstruction given by $\{z_{i_j}^*\}_{j=1}^k$ and $\{\mathbf{det}_{i_j}\}_{j=1}^k$, $|I|$ is the cardinality of image I. Dropping some constants in Equation 3.16, our loss function is defined as

$$
L = \|I - I_{\{i_1,...,i_k\}}\|_{vec,2}^2 + \lambda k + \sigma^2 \sum_{j=1}^{k} [\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - 2\sum_{t=1}^{N_z} \log \tau_{i_j,t}], \tag{3.17}
$$

where $\lambda = 2\sigma^2\lambda_0$ can be regarded as a penalty on the number of selected boxes k.

If there are N detections in total, it is impossible to enumerate and evaluate all possible ordered subsets $\{i_1, ..., i_k\}$. So we propose a Detection Selection Algorithm (DSA), see Algorithm 3, to find a good subset in polynomial time.

In Algorithm 2 (DSAWR), we processed the selected detections based on their occlusion scores to identify the visible pixels for each bounding box. However, it is possible for some

detections of very low quality to exhibit high occlusion scores. As higher objectness scores indicate more confident detections, and more confident detections are more likely to be included in our final selection, in Algorithm 3 we process $\{\mathbf{det}_{i_j}\}_{j=1}^{k}$, the subset of detections, according to their objectness scores arranged from high to low. At each step, when provided with a subset of detections, we feed it into Algorithm 2, where the detections are reordered based on occlusion scores to generate the whole reconstruction and calculate the loss function defined in Equation 3.17.

We use S to represent the currently selected detections, which is $\emptyset$ in the beginning. If selecting $S \cup \{\mathbf{det}_i\}$ yields smaller loss than with S, we prefer interpretation $S \cup \{\mathbf{det}_i\}$ to S. But we also consider the case when there is a $\mathbf{det}_j$, $j < i$, which has significant overlap with $\mathbf{det}_i$. It is possible that $\mathbf{det}_i$ is the correct detection and $\mathbf{det}_j$ isn't. So we select the detection $j < i$ with highest intersection-over-union (IoU) with i, and consider the interpretation $(S \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}$. Thus, we compare S, $S \cup \{\mathbf{det}_i\}$ and $(S \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}$, the one which has the smallest loss is used as the new S. Then we move on to the next detection in the objectness score ordering.

Therefore in Algorithm 3 (DSA), $\mathbf{det}_j$ is chosen as the previously selected detection which has the highest IoU with $\mathbf{det}_i$. If there is no previous selected detection, or if all previously selected detections have zero IoU with $\mathbf{det}_i$, then $\mathbf{det}_j$ doesn't exist and we don't need to consider $(S \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}$. In this case we simply set $L_{i,2} = \infty$ for that index i in Algorithm 3 so that $(S \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}$ can't be selected.

Our Detections Selection Algorithm (DSA) greedily chooses subsets of detections to minimize Equation 3.17. The initial term in Equation 3.17 incentivizes DSA to opt for the interpretation that yields superior reconstruction performance. Occasionally, selecting duplicated detections may result in nearly identical reconstruction loss. The penalty λ imposed on the number of boxes is essential to steer DSA away from such scenarios. Once all detections have undergone processing, the final selected detections $\{\mathbf{det}_i\}_{i \in S_N}$ are chosen as our

Algorithm 3: Detection Selection Algorithm (DSA)

Input: Image I, detection results $\{\mathbf{det}_i = (score_i, bb_i, occ_i, cls_i)\}_{i=1}^{N}$ sorted by $score_i$ from high to low, the penalty $\lambda \geq 0$, assumed variance $\sigma^2 > 0$, latent dimension N_z

$S_0 \leftarrow \emptyset$;

$ReconDict \leftarrow \{\}$;

$L_0 \leftarrow \infty$;

for $i = 1$ **to** N **do**

$(I_{i,1}, ReconDict) \leftarrow DSAWR(S_{i-1} \cup \{\mathbf{det}_i\}, ReconDict)$;

$L_{i,1} \leftarrow \|I - I_{i,1}\|_2^2 + \lambda |S_{i-1} \cup \{\mathbf{det}_i\}| + \sum_{i_j:\mathbf{det}_{i_j} \in S_{i-1} \cup \{\mathbf{det}_i\}} \sigma^2 [\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - 2\sum_{t=1}^{N_z} \log \tau_{i_j,t}]$; `/* |·| is the cardinality of the set */`

if $S_{i-1} = \emptyset$ *or* $\max_{\boldsymbol{det}_{j_1} \in S_{i-1}} IoU(\boldsymbol{det}_{j_1}, \boldsymbol{det}_i) = 0$ **then**

$L_{i,2} \leftarrow \infty$;

else

$\mathbf{det}_j = \operatorname{argmax}_{\mathbf{det}_{j_1} \in S_{i-1}} IoU(\mathbf{det}_{j_1}, \mathbf{det}_i)$;

$(I_{i,2}, ReconDict) \leftarrow DSAWR((S_{i-1} \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}, ReconDict)$;

$L_{i,2} \leftarrow \|I - I_{i,2}\|_2^2 + \lambda |(S_{i-1} \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}| + \sum_{i_j:\mathbf{det}_{i_j} \in (S_{i-1} \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}} \sigma^2 [\|\mu_{i_j}\|_2^2 + \sum_{t=1}^{N_z} \tau_{i_j,t}^2 - 2\sum_{t=1}^{N_z} \log \tau_{i_j,t}]$;

end

$L_i \leftarrow \min(L_{i-1}, L_{i,1}, L_{i,2})$;

if $L_i = L_{i-1}$ **then**

$S_i \leftarrow S_{i-1}$;

else if $L_i = L_{i,1}$ **then**

$S_i \leftarrow S_{i-1} \cup \{\mathbf{det}_i\}$;

else

$S_i \leftarrow (S_{i-1} \setminus \{\mathbf{det}_j\}) \cup \{\mathbf{det}_i\}$;

end

end

Output: Selected detections S_N

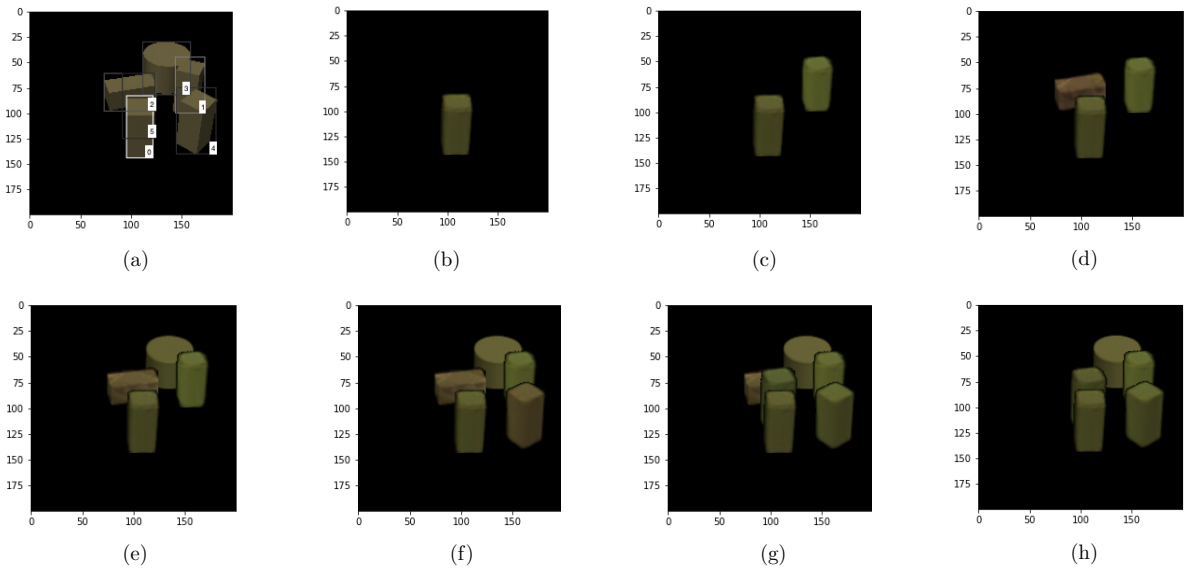


Figure 3.5: Example of Detection Selection Algorithm (DSA).

interpretation for the image in Algorithm 3.

In Figure 3.5, in image (a) we show the original image with five actual objects and 6 detections indexed from 0 to 5, ordered according to their objectness score, same as in Figure 3.4. $N = 6$ and the last bounding box is redundant. As stated in Algorithm 3, we start from $S_0 = \emptyset$. In the first step, we only consider detection $\{0\}$, which has highest objectness score. The loss is 1307.8, so we have $S_1 = \{0\}$, as shown in (b). Next we move on to bounding box 1 as the next highest objectness score. Because no bounding box has an intersection with bounding box 1, in the second step we only consider the ordered set $\{0, 1\}$, (there is no possible detection to omit). The loss is 982.5, which is better than 1307.8, so $S_2 = \{0, 1\}$. Its canvas is shown in (c). In the third step bounding box 0 has the largest IoU with bounding box 2, so we compare both $\{0, 1, 2\}$ and $\{1, 2\}$ and select $S_3 = \{0, 1, 2\}$ in (d). Its loss is 713.9. Similarly we compare $\{0, 1, 2, 3\}$ and $\{0, 2, 3\}$ and select $S_4 = \{0, 1, 2, 3\}$ in (e) which gives us loss 356.3. In step 5, $\{0, 1, 2, 3, 4\}$ and $\{0, 2, 3, 4\}$ are compared and $S_5 = \{0, 1, 2, 3, 4\}$ with loss 151.6 is selected. Finally, since bounding box 2 has the largest IoU with bounding box 5, we process $\{0, 1, 2, 3, 4, 5\}$ and $\{0, 1, 3, 4, 5\}$, which gives us losses 165.2 and 253.0

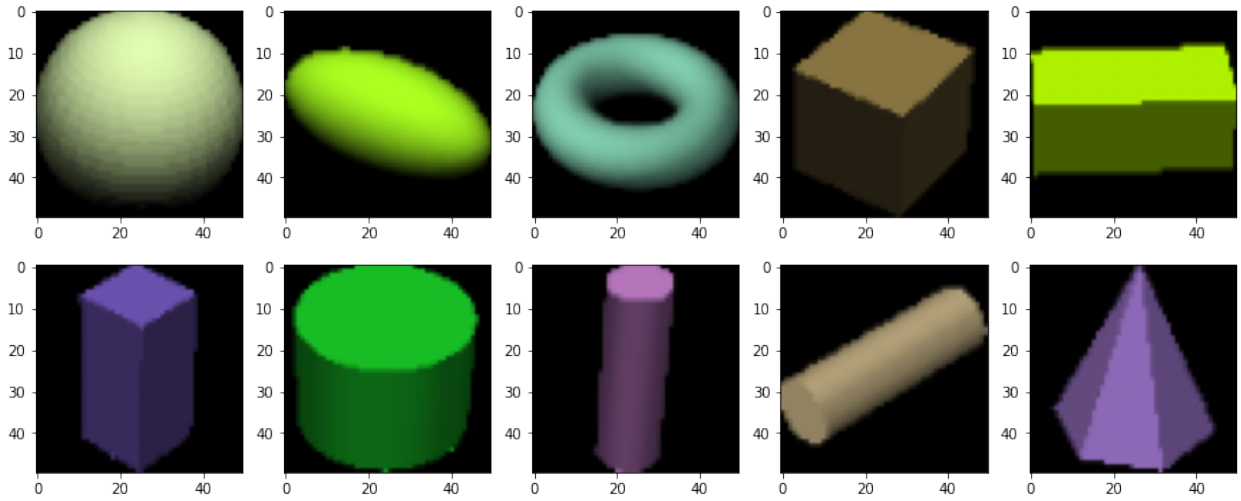


Figure 3.6: 10 classes of objects.

respectively, and their canvases are shown in images (g) and (h). But neither $\{0, 1, 2, 3, 4, 5\}$ or $\{0, 1, 3, 4, 5\}$ has a smaller loss than S_5, so $S_6 = S_5 = \{0, 1, 2, 3, 4\}$. Therefore, ultimately we interpret the image by bounding boxes $\{0, 1, 2, 3, 4\}$, which means there are 5 objects and our predicted labels are the corresponding labels of the bounding boxes $\{0, 1, 2, 3, 4\}$. Some more experiments about DSA are shown in Section 3.8.2.

3.7 Dataset

Our synthesized datasets encompass 10 object classes, denoted from class 1 through class 10. The 10 classes include the following objects, as illustrated in Figure 3.6: sphere, ellipsoid, torus, regular cube, lying thin cuboid, standing thin cuboid, regular cylinder, standing thin cylinder, lying thin cylinder, and cone. Irrespective of the object classes, each image features random colors (r, g, b) assigned to all objects, where $0 \leq r, g, b \leq 1$ are uniformly chosen, subject to the constraint $r + g + b \geq 1$. Objects are set to have the same color to enhance the difficulty of detection.

In all our datasets for this chapter, objects are randomly positioned on a pure black background. We employ the Python package 'pyvista' for image generation. Each image

is illuminated by three lights with fixed directions and intensities set at $0.5, 0.5, 0.2$ respectively. The camera position and focus remain constant, while the objects are allowed to rotate horizontally from 0 to 360 degrees. As depicted in Figure 3.6, classes $1, 3, 7, 8$ exhibit rotational invariance, whereas the others do not.

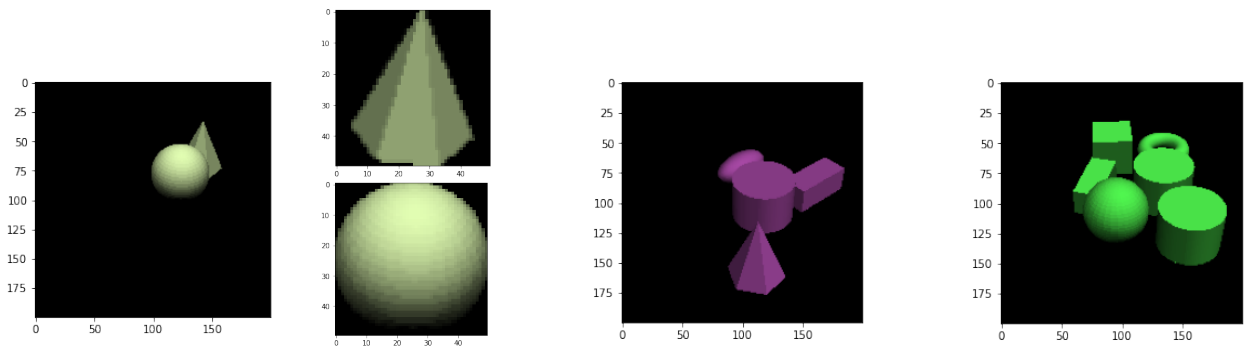
3.7.1 Training sets

To train the Faster R-CNN model or the Faster R-CNN-OC model in Section 3.2, we generate paired occluded objects on a 200×200 black background as our training set. These objects are positioned on an invisible floor. Rejection sampling is employed to ensure that, despite being occluded, the two objects won't be in close proximity to each other, and each object has at least 200 visible pixels. The classes of the two paired objects are selected from the 10 classes we have, ensuring that each class appears exactly 1000 times in the dataset. Thus, we have a total of $1000 \times 10/2 = 5000$ images for Faster R-CNN and Faster R-CNN-OC.

For the Single Reconstruction Algorithm for DSA (DSASR) mentioned in Section 3.3, we train a VAE decoder. In generating pairs of objects for the Faster R-CNN and Faster R-CNN-OC training images, we use the same individual objects as our decoder training data. However, these objects are isolated, centered in 50×50 images, and re-scaled to maximize their size within the 50×50 images. This resizing is achieved through the parameterized sampling grid technique discussed in Section 3.3.2. In Figure 3.7, image (a) represents a training image for Faster R-CNN and Faster R-CNN-OC, and the two images in (b) depict the corresponding training data for the decoder. We have a total of $5000 \times 2 = 10000$ images for the decoder.

3.7.2 Validation and test sets

To assess the performance of our post-processing method DSA, we utilize a validation set and a test set. Both datasets consist of 200×200 images, with each image containing between



(a) (b) (c) (d)

Figure 3.7: Datasets.

3 and 7 objects. The challenge of post-processing tends to increase with a higher number of objects.

Both the validation set and test set for post-processing comprise 500 images each. We ensure that every object has a minimum of 200 visible pixels, and the objects are sufficiently spaced apart. For the validation set, there are 250 images with 3 objects and 250 images with 4 objects. An example of a validation image is shown in image (c) of Figure 3.7. In the test set, we have $150, 150, 200$ images with $5, 6, 7$ objects, respectively. Image (d) in Figure 3.7 is an example from the test set containing 6 objects.

3.8 Experiments

As elucidated in Section 3.7, both the Faster R-CNN and our Faster R-CNN-OC model are trained using 5000 paired occluded objects, which are further split into 4000 training images and 1000 validation images. Both models are trained with a batch size of 10, utilizing default Faster R-CNN parameters and no pre-training. The training process halts upon observing 10 consecutive epochs with no improvement in the validation loss. The Faster R-CNN and Faster R-CNN-OC models completed training at epoch 79 and 64 respectively. In the occlusion branch, during training, the "occlusion scores" of the upper object and the lower object are set to be 1.0 and 0.0 respectively.

For each class, a separate decoder is trained with a latent dimension of 10. The decoder architecture comprises one hidden layer with 300 units fully connected to a layer with 7500 units corresponding to the $50 \times 50 \times 3$ output. ReLU nonlinearity is applied after the hidden layer, and a sigmoid nonlinearity is used after the final layer. The first 8000 images are employed for training the decoder, while the remaining 2000 are reserved for testing. The decoder undergoes 400 epochs of training, with decoder parameter updates occurring once after every 10 optimization iterations of the latent code. Fixed values are set for σ (0.1) and batch size (100). Adam optimizers are utilized, and the learning rates for updating decoder parameters and latent code are 0.0001 and 0.01.

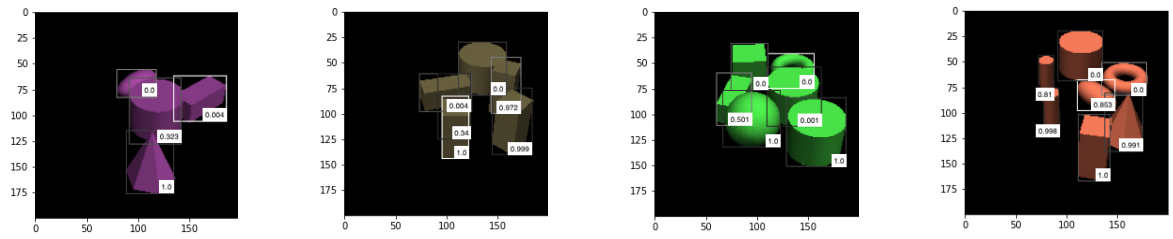
In the full Detection Selection Algorithm (DSA), detections with an objectness score less than 0.25 are discarded, as such detections would ultimately likely be rejected by Algorithm 3, thereby avoiding unnecessary computational costs.

3.8.1 Occlusion Scores

The Faster R-CNN-OC model is trained on pairs of objects; however, it demonstrates effective generalization when tested on scenarios involving three or more overlapping objects. Given that the primary emphasis of our work is not on occlusion relationship reasoning, we present selected results in Figure 3.8. The predicted occlusion score is depicted at the lower right corner for each bounding box. It is important to note that, for clarity, we only display the top few bounding boxes following the application of Non-maximum Suppression (NMS) by Faster R-CNN-OC.

3.8.2 DSA Accuracies

Frequently, the evaluation of detection quality involves the use of mAP (mean Average Precision). However, when establishing the accurate count of objects is crucial, precision at $recall = 1$ assumes utmost significance. In this work we use two types of accuracies as our



(a) (b) (c) (d)

Figure 3.8: Example of Predicted Occlusion Scores.

evaluation metrics:

- The precent of images where the correct number of boxes is chosen.
- The percent of images with the correct number of boxes and correct predicted labels.

As an illustration, consider an image containing one object of class-1 and two objects of class-2. If our prediction comprises two objects of class-1 and one object of class-2, we would be deemed correct according to the first evaluation metric but incorrect under the second evaluation metric. Evidently, the second evaluation metric exhibits a more stringent criterion.

In Table 3.1, we present a comparison of three post-processing methods applied to Faster R-CNN: NMS, Soft-NMS, and DIoU-NMS. The parameters T_{boxes} and T_{labels} are both thresholds. In the first segment of the table, we determine the optimal threshold T_{boxes} through a grid search over the range $0.01, 0.02, ..., 0.99$ as a threshold on the validation set. Detections above the chosen threshold for each method are considered final detections. Similarly, T_{labels} aims to maximize the accuracy of labels in the validation set. Notably, the chosen T_{boxes} and T_{labels} are quite close to each other. The thresholds for Soft-NMS are lower because Soft-NMS decreases objectness scores. For comparison, in the second segment of Table 3.1, we fix the thresholds to be 0.5. Due to these lower thresholds, the accuracies of NMS decrease drastically.

The accuracies in Table 3.1 are calculated on the test set based on the thresholds es-

Table 3.1: NMS, Soft-NMS and DIoU-NMS with Faster R-CNN

Methods	T_{boxes}	T_{labels}	**Accuracy for Boxes**	**Accuracy for Labels**
NMS	0.91	0.91	0.962 (0.0086)	0.962 (0.0086)
Soft-NMS	0.69	0.69	0.950 (0.0097)	0.950 (0.0097)
DIoU-NMS	0.91	0.91	0.940 (0.0106)	0.940 (0.0106)
NMS	0.5	0.5	0.772 (0.0188)	0.772 (0.0188)
Soft-NMS	0.5	0.5	0.946 (0.0101)	0.946 (0.0101)
DIoU-NMS	0.5	0.5	0.934 (0.0111)	0.934 (0.0111)

timated from the validation set. Because the validation set and test set have different distributions, the thresholds may not be optimal. The numbers in the parenthesis are the estimated standard deviations using $\sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$, where $\hat{p}$ is the average accuracy and $n = 500$ is the number of test samples. Accuracy for boxes and accuracy for labels represent the first and second evaluation metrics mentioned earlier. It is observed that, for each method, the accuracy for boxes and labels are the same, indicating that the predicted labels are typically correct.

Given a set of detections generated by an object detection algorithm, the described Detections Selection Algorithm (DSA) lacks the ability to reduce False Negatives but can be employed to minimize False Positives. We apply DSA after Non-Maximum Suppression (NMS) or DSA after Soft-NMS to refine the detections produced by the Faster R-CNN-OC. The NMS threshold is set at 0.5. In Table 3.2, DSA after NMS is referred to as "NMS+DSA", and DSA after Soft-NMS is denoted as "Soft-NMS+DSA".

The penalty parameters λ_{boxes} and λ_{labels} are also chosen using the validation set to maximize the two evaluation metrics, respectively. We experimented with values such as $10, 20, 30, 40, 50$. In case of ties, the median value is selected. Table 3.2 demonstrates noticeable improvements over the original NMS or Soft-NMS results presented in Table 3.1.

Particularly, NMS+DSA exhibits significant performance improvement.

Table 3.2: NMS+DSA and Soft-NMS+DSA on Faster R-CNN-OC

Methods	λ_{boxes}	λ_{labels}	**Accuracy for Boxes**	**Accuracy for Labels**
NMS+DSA	15	15	0.980 (0.0063)	0.980 (0.0063)
Soft-NMS+DSA	20	20	0.982 (0.0059)	0.980 (0.0063)

3.8.3 Recovering False Negatives

We conducted a simple experiment to see how DSA might be extended to recover missed detections of NMS or soft-NMS. We rotated each test image by 10 degrees. This minor perturbation significantly reduces the accuracy of the detection algorithms. For example soft-NMS yields 0.90 for number of boxes and 0.652 for proportion of images with correct labels. Just observing the results it is clear that the small rotation leads the faster R-CNN to label many instances of class 8 - the upright cylinder as class 9. So we added a minor hack in the code, where any time a box is labeled 9, we also run the Whole Reconstruction Algorithm for DSA (DSAWR) on exactly the same input except that the new box is labeled 8 instead of 9 and then compare the NLL's. Furthermore, we introduce a new variable α in the decoder for rotation in addition to the translation variable so that the decoder optimization is over $\mu, \Gamma, (t_x, t_y), \alpha$. This yielded a significant improvement of 0.906 proportion of images with correct number of detected boxes and 0.856 proportion of images with all labels correct. In Figure 3.9 we show the different whole reconstructions produced without and with the likelihood comparison between class 9 and class 8. This experiment points to the possibility of recovering from distribution shifts by extending the free parameters of the decoder as well as entertaining more than one class label for each detected box.

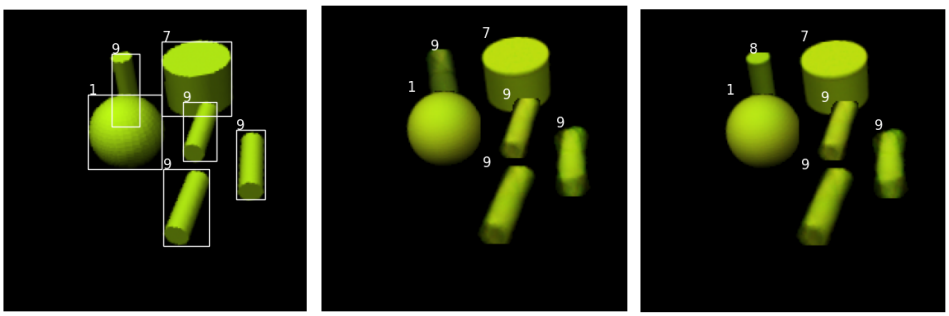


Figure 3.9: Left: Faster R-CNN output on rotated image, Middle: Whole reconstruction without class 8 competition, Right: Whole reconstruction with class 8 competition.

3.8.4 Enlarged Objects

Another noteworthy experiment involves rescaling all objects in the test set by the same proportion of $\frac{10}{9}$, while maintaining the training and validation sets unchanged. This is achieved by cropping a 180×180 region containing all the objects and enlarging it to create a 200×200 image.

Table 3.3: NMS, Soft-NMS and DIoU-NMS with Faster R-CNN on Enlarged Objects

Methods	T_{boxes}	T_{labels}	**Accuracy for Boxes**	**Accuracy for Labels**
NMS	0.91	0.91	0.886 (0.0142)	0.880 (0.0145)
Soft-NMS	0.69	0.69	0.916 (0.0124)	0.908 (0.0129)
DIoU-NMS	0.91	0.91	0.886 (0.0142)	0.874 (0.0148)

Table 3.4: NMS+DSA and Soft-NMS+DSA on Enlarged Objects with Faster R-CNN-OC

Methods	λ_{boxes}	λ_{labels}	**Accuracy for Boxes**	**Accuracy for Labels**
NMS+DSA	15	15	0.988 (0.0049)	0.982 (0.0059)
Soft-NMS+DSA	20	20	0.964 (0.0083)	0.960 (0.0088)

The outcomes are presented in Table 3.3 and Table 3.4. The results indicate that DSA brings about highly significant enhancements. Notably, NMS+DSA demonstrates particu-

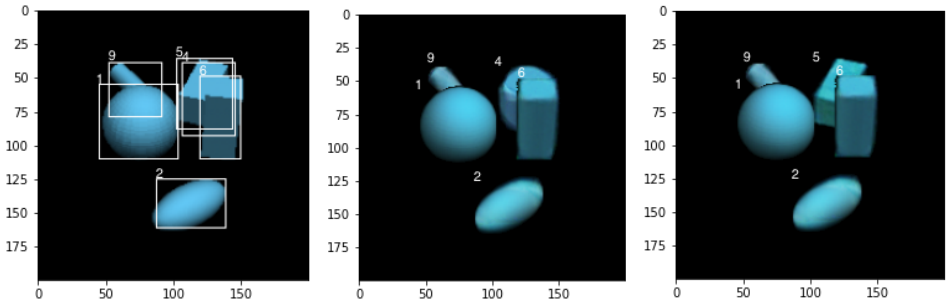


Figure 3.10: Left: Faster R-CNN output , Middle: Whole reconstruction of top 5 bounding boxes, Right: Whole reconstruction of top 4 and the 6th bounding boxes.

larly favorable performance. Transitioning from standard objects to the enlarged objects, the accuracies of NMS decrease from $0.962, 0.962$ to $0.886, 0.880$, whereas the accuracies of NMS+DSA don't decrease.

Due to the introduction of objects with different sizes in the new images, the objectness scores become less reliable for Faster R-CNN. Consequently, NMS, Soft-NMS, and DIoU-NMS exhibit diminished performance. However, our Single Reconstruction Algorithm for DSA (DSASR) demonstrates the capability to handle bounding boxes of various scales, allowing our DSA to effectively operate on enlarged objects.

Figure 3.10 illustrates why we need to compare $(S\backslash\{\mathbf{det}_j\})\cup\{\mathbf{det}_i\}$ with S and $S\cup\{\mathbf{det}_i\}$ in the Detection Selection Algorithm (DSA). In Figure 3.10, an object of class 5 is predicted in two different bounding boxes by Faster R-CNN as class 4 and class 5 with objectness scores 0.89 and 0.82 respectively. Thus, the class 4 object is processed before the class 5 object in the DSA. The whole reconstruction by the top 5 detections in terms of objectness scores yields loss 430.36. It selects the wrong bounding box of class 4. In the next step, DSA considers dropping the bounding box of class 4 and adding the bounding box of class 5. The loss decreases to 376.92, and it gives us the right interpretation.

3.9 Conclusion

In this chapter, we have introduced the Detection Selection Algorithm (DSA) along with several complementary algorithms designed to ascertain the precise number of objects and their corresponding labels in an image. DSA is employed subsequent to NMS or similar techniques. The framework is likelihood-based, involving comparisons among image interpretations, specifically ordered sequences of instantiated objects. The probabilistic framework offers a global evaluation of any interpretation and takes into account the relationships between the different objects. Notable byproducts of DSA include determining the occlusion sequence of objects, reconstructing the invisible parts of objects, and generating images based on a given set of hypothesized objects.

We note that most network models used today in image processing are fully feed-forward. The input passes through the network and produces the output. This works well when there is ample training data and when the distribution of the test data is the same as that of the training data, i.e. no distributional shift. However such methods are quite sensitive to distributional shifts as demonstrated in the experiments above, and it appears to us that in certain settings adjusting to such shifts without retraining necessitates an online optimization procedure that can accommodate the modified distribution, and in particular using global likelihood based reasoning. A full probabilistic model is the most principled way to achieve this, albeit at a significant computational cost. Our greedy algorithm implements only one-step back search, only inspecting the detection with highest overlap. More extensive searches could be implemented exploring a wider range of ordered subsets of the detections, again, at a higher computational cost.

To extend the DSA, DSASR and DSAWR to real-world images with colored background and clutter leads to Chapter 4.

CHAPTER 4

DETECTION SELECTION ALGORITHM WITH MASK FOR PANOPTIC SEGMENTATION

4.1 Motivation

Panoptic segmentation, as explored in various studies Kirillov et al. [2019b], Li and Chen [2022], Elharrouss et al. [2021], Chuang et al. [2023], has recently garnered significant attention. This research field involves the assignment of semantic labels to pixels, along with unique object instance IDs, thereby facilitating a comprehensive understanding of the image. Diverging from instance segmentation, where predicted object masks may overlap, panoptic segmentation ensures a non-overlapping assignment of labels to every pixel. Therefore, adaptation is required to apply instance segmentation to panoptic segmentation tasks Kirillov et al. [2019b]. The assessment of panoptic segmentation quality typically relies on metrics such as Panoptic Quality (PQ) Kirillov et al. [2019b] or modified PQ Porzi et al. [2019]. Images are partitioned into distinct categories, namely *thing* and *stuff* classes, where the former comprises countable objects, and the latter encompasses amorphous elements such as grass, road, and sky. Both thing and stuff classes contribute to Panoptic Quality (PQ) scores. The Panoptic Quality score can be seen as the multiplication of two quantities, one is the F1 score Van Rijsbergen [1979] and the other is the averaged IoU of those matched pieces in panoptic segmentation.

Instance segmentation methods, exemplified by Mask R-CNN He et al. [2017], which detects objects and predicts object masks, can be integrated into panoptic segmentation when considering everything other than the objects as a *stuff* class. This panoptic segmentation can be achieved with a NMS-like procedure Kirillov et al. [2019b] which greedily assign pixels to detections according to their confidence scores from high to low. This NMS-like procedure has been pointed out to be suboptimal Lazarow et al. [2020], because detections which

have higher confidence scores don't necessarily occlude less confident detections. Numerous advancements in panoptic segmentation, as evidenced by studies such as Lazarow et al. [2020], Kirillov et al. [2019a], Mohan and Valada [2021], Lazarow et al. [2020], have been realized through innovative network architectures and novel training objectives. However, the question arises: Is it possible to enhance panoptic segmentation quality without altering the fundamental architecture of instance segmentation model?

Addressing this query affirmatively, in this chapter we introduce our Detection Selection Algorithm with Mask (DSAM). DSAM enhances Panoptic Segmentation quality by leveraging both a trained detection model and a trained deep generative model. As an illustrative example in this chapter we use Mask R-CNN He et al. [2017] and VAE with flow prior Huang et al. [2017] as our detection model and deep generative model. Notably, DSAM serves as an extension of the Detection Selection Algorithm (DSA) Fan et al. [2023]. DSA, characterized as a greedy algorithm, aims to select the optimal set of detections for interpreting the image. Operating under the assumption of a pure black background, DSA exclusively involves object detections without accompanying object masks. Its design aims to surpass traditional techniques like Non-maximum Suppression (NMS) and Soft-NMS Bodla et al. [2017a]. Executing DSA requires three essential tools: Faster R-CNN-OC, Single Reconstruction Algorithm for DSA (DSASR), and Whole Reconstruction Algorithm for DSA (DSAWR). For a more in-depth exploration of these concepts, refer to Chapter 3.

Diverging from the Detection Selection Algorithm (DSA), our Detection Selection Algorithm with Mask (DSAM) is tailored to address real-life images characterized by colored backgrounds populated with various clutters and irrelevant objects. In our approach, we uniformly categorize all such stuff classes and irrelevant objects as a singular entity referred to as the "background", and treat it as a *stuff* class. Much like DSA, DSAM incorporates an occlusion relationship reasoning algorithm, a single reconstruction algorithm, and a whole reconstruction algorithm as incidental outcomes. However, in DSAM, the occlusion

relationship reasoning algorithm relies on depth estimation rather than Faster R-CNN-OC. Additionally, we have made slight modifications to enhance the efficiency of both the DSASR in Section 3.3 and the DSAWR in Section 3.4 and call the modified algorithms DSAMSR and DSAMWR respectively. In addition to the "selection" and "discarding" operations in DSA, in DSAM we have another "designation as background" operation. This is needed in DSAM because in many cases the detection algorithm recognizes clutters or non-objects as objects. In DSA the image has no clutters so "designation as background" operation is less important. The evaluation of our method is conducted using the Panoptic Quality (PQ) metric. The primary contributions of this chapter encompass:

- We establish three operations for each detection: "selection", "discarding", and "designation as background". Following the implementation of DSAM, detections subjected to the "selection" operation are retained and subsequently utilized in panoptic segmentation.

- We formulate a likelihood optimization framework, elucidated in Section 4.2, to construct a loss function employed within our Detection Selection Algorithm with Mask (DSAM).

- We propose DSAM, a method that sequentially determines one of three operations for each detection based on likelihood comparisons.

In the subsequent discussion, we delineate our likelihood framework, which is presented in Section 4.2. Following this, in Section 4.3 we elaborate on our new occlusion relationship reasoning method, founded on depth estimation as an alternative to an occlusion branch in Faster R-CNN. Furthermore, Sections 4.4, 4.5, and 4.6 detail the Single Reconstruction Algorithm for DSAM (DSAMSR), Whole Reconstruction Algorithm for DSAM (DSAMWR), and DSAM, respectively. Section 4.7 presents the experimental results, and, ultimately, we draw this chapter to a conclusion in Section 4.8.

4.2 Probabilistic Framework

In contrast to the entirely black background utilized in DSA Fan et al. [2023], DSAM, in its application, confronts real-world images characterized by diverse colored backgrounds, encompassing various forms of visual clutter and miscellaneous object classes.

Assuming that our occlusion relationship reasoning algorithm and the Mask R-CNN He et al. [2017] propose detections in the form of $\{\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)\}_{i=1}^{N}$, where $score_i$, bb_i, $mask_i$, cls_i are objectness (confidence) score, predicted bounding box, predicted object mask and predicted label by Mask R-CNN, and occ_i is the predicted occlusion score detailed in Section 4.3. We execute one of the three operations on each detection: either "selection", "discarding", or "designation as background". The sets S, D and B represent the indices of operations corresponding to "selection", "discarding", and "designation as background" respectively. It is imperative to emphasize that henceforth in this chapter we designate I^* to be the original image and the variable I to represent the original image constrained within the union of all predicted masks in the detections, denoted as $\cup_{i=1}^{N} mask_i$. The pixels subject to analysis by our algorithm encompass precisely the entirety enclosed within the union $\cup_{i=1}^{N} mask_i$. As an illustrative example, in Figure 4.1, image (a) represents the original image denoted as I^*, whereas the non-zero segment of image (b) corresponds to the variable I.

Diverging from the probabilistic framework in DSA Fan et al. [2023], our new probability model computes $p(I, S, D, B, \{\mathbf{det}_i\}_{i=1}^{N})$, representing the joint probability of having $I, \{\mathbf{det}_i\}_{i=1}^{N}$, selected detections S, discarded detections D and detections designated as backgrounds B. It is crucial to emphasize that our set I encompasses pixels within the complete union $\cup_{i=1}^{N} mask_i$, irrespective of the specific choice of D. In other words, we assign detections into S, D, B to explain I but the choice of S, D, B doesn't affect I itself. It is stipulated that, by definition, $S \cap D = D \cap B = B \cap S = \emptyset$ and $S \cup D \cup B = \{i\}_{i=1}^{N}$. The

(a)

(b)

Figure 4.1: (a)- the original image I^*, (b)- the original image constrained within the union of all predicted masks, denoted as I.

logarithm of the joint likelihood can be computed as follows:

$$\log p(I, S, D, B, \{\mathbf{det}_i\}_{i=1}^N) = \log p(S, D, B, \{\mathbf{det}_i\}_{i=1}^N) + \log p(I|S, D, B, \{\mathbf{det}_i\}_{i=1}^N), \tag{4.1}$$

where

$$\begin{aligned} &\log p(I|S, D, B, \{\mathbf{det}_i\}_{i=1}^N) \\ &= \log \int \cdots \int p(I|\{z_i\}_{i=1}^N, S, D, B, \{\mathbf{det}_i\}_{i=1}^N) p(\{z_i\}_{i=1}^N|S, D, B, \{\mathbf{det}_i\}_{i=1}^N) dz_1 \cdots dz_N. \end{aligned} \tag{4.2}$$

In the aforementioned formula, it is assumed that latent code z_i corresponding to the object

in detection $\mathbf{det}_i$ follows a prior distribution. Additionally, a decoder maps z_i to the image space. For any given set $\{z_i\}_{i=1}^N, S, D, B$, the mean of the Gaussian conditional distribution $I|\{z_i\}_{i=1}^N, S, D, B, \{\mathbf{det}_i\}_{i=1}^N$ is determined, and the observed pixels are conditionally independent in the Gaussian conditional distribution $I|\{z_i\}_{i=1}^N, S, D, B, \{\mathbf{det}_i\}_{i=1}^N$ with a consistent variance denoted by σ^2. This framework simplifies the computation of the likelihood.

It is important to note that $\{z_i\}_{i\in D}$, the latent codes for the discarded detections, does not impact the interpretation of the image, as the detections $\{\mathbf{det}_i\}_{i\in D}$ have been discarded. In the probability $p(I|S, D, B, \{\mathbf{det}_i\}_{i=1}^N)$, latent codes for objects in sets S and B are used to interpret the content of I. In instances where the pixels in I are not accounted for by objects in sets S and B, a value of $(0, 0, 0)$ is utilized for explanation. This approach mirrors the scenario where detections in set D are disregarded, and only sets S and B are utilized to explain I, denoted as $p(I|S, B, \{\mathbf{det}_i\}_{i=1}^N)$. Therefore, we make the assumption that

$$p(I|S, D, B, \{\mathbf{det}_i\}_{i=1}^N) = p(I|S, B, \{\mathbf{det}_i\}_{i=1}^N).$$

The proportion of pixels in image I unaccounted for by sets S and B is minimal, given that explanations from sets S or B generally result in a more accurate alignment with the image compared to $(0, 0, 0)$. A more precise alignment with the image typically leads to a reduced loss, which is subsequently defined in this section. Instances where a detection is assigned to set D typically arise when pixels within its object mask have already been accounted for by another detection. As a result, the DSAM outlined in Section 4.6 tends to avoid leaving pixels unaccounted for by S and B.

Throughout this chapter, we use the symbol $\mathbf{D}_{KL}$ to represent the Kullback–Leibler divergence

$$\mathbf{D}_{KL}(p(\cdot)||q(\cdot)) = \int p(x) \log \frac{p(x)}{q(x)} dx.$$

In alignment with the methodology employed in DSA Fan et al. [2023], we employ the subsequent variational approximation:

$$
\begin{aligned}
&\log p(I|S,D,B,\{\mathbf{det}_i\}_{i=1}^N) = \log p(I|S,B,\{\mathbf{det}_i\}_{i=1}^N) \\
=&\mathbf{E}_{q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N))} \log \frac{p(I,\{z_i\}_{i\in S\cup B}|S,B,\{\mathbf{det}_i\}_{i=1}^N)}{q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)} \\
&+\mathbf{D}_{KL}(q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)||p(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)) \\
=&\mathbf{E}_{q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)} \log p(I|\{z_i\}_{i\in S\cup B},S,B,\{\mathbf{det}_i\}_{i=1}^N) \\
&-\mathbf{D}_{KL}(q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)||p(\{z_i\}_{i\in S\cup B}|S,B,\{\mathbf{det}_i\}_{i=1}^N)) \\
&+\mathbf{D}_{KL}(q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)||p(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)) \\
\approx&\mathbf{E}_{q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)} \log p(I|\{z_i\}_{i\in S\cup B},S,B,\{\mathbf{det}_i\}_{i=1}^N) \\
&-\mathbf{D}_{KL}(q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)||p(\{z_i\}_{i\in S\cup B}|S,B,\{\mathbf{det}_i\}_{i=1}^N)),
\end{aligned}
\tag{4.3}
$$

where $q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)$ represents the posterior distribution derived from our generative models while $p(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N)$ is the true posterior distribution. Specifically, we train separate generative models for each object class as well as for the background. For clarity, we denote ϕ_c as the generative model trained on class c and ϕ_{bg} for the generative model trained on background pieces. We additionally posit the independence of the posterior distributions of z_i across different detections

$$
q_\phi(\{z_i\}_{i\in S\cup B}|I,S,B,\{\mathbf{det}_i\}_{i=1}^N) = \prod_{i\in S} q_{\phi_{cls_i}}(z_i|I_i) \prod_{i\in B} q_{\phi_{bg}}(z_i|I_i), \tag{4.4}
$$

where I_i denotes the "image context", which is the square image segment surrounding bb_i. In Figure 4.2, the predicted bounding boxes are represented by the blue boxes, and their corresponding image contexts are depicted by the content of the yellow boxes.

Figure 4.2: Example of 3 image contexts surrounding 3 bounding boxes.

Another assumption that we make is

$$p(\{z_i\}_{i\in S\cup B}|S, B, \{\mathbf{det}_i\}_{i=1}^{N}) = \prod_{i\in S} p_{\theta_{cls_i}}(z_i) \prod_{i\in B} p_{\theta_{bg}}(z_i), \tag{4.5}$$

which means z_i's are independent in their generation. In modeling each prior distribution, we employ normalizing flows denoted as $f_\theta : \mathcal{Z} \to \mathcal{E}$. Assume $z_i|I_i \sim \mathcal{N}(\mu_i, \Gamma_i)$ for $i \in S\cup B$, where Γ_i is a $N_z \times N_z$ diagonal matrix with diagonal elements $\tau_{i,t}^2$, $t = 1, 2, ..., N_z$, the KL divergence can be derived as

$$\begin{aligned}&\mathbf{D}_{KL}(q_\phi(z_i|I_i)||p_\theta(z_i))\\ &= -\frac{N_z}{2}(1+\log(2\pi)) - \sum_{t=1}^{N_z}\log\tau_{i,t} - \mathbf{E}_{q_\phi(z_i|I_i)}\log p_{\mathcal{E}}(f_\theta(z)) - \mathbf{E}_{q_\phi(z_i|I_i)}\log|det\left(\frac{\partial f_\theta(z)}{\partial z}\right)|.\end{aligned} \tag{4.6}$$

The computational details are elucidated in Equation 4.16. Under the aforementioned assumptions, Equation 4.3 can be further streamlined to

$$
\begin{aligned}
&\mathbf{E}_{\prod_{i\in S} q_{\phi_{cls_i}}(z_i|I_i)\prod_{i\in B} q_{\phi_{bg}}(z_i|I_i)} \log p(I|\{z_i\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N) \\
&- \sum_{i\in S} \mathbf{D}_{KL}(q_{\phi_{cls_i}}(z_i|I_i)||p_{\theta_{cls_i}}(z_i)) - \sum_{i\in B} \mathbf{D}_{KL}(q_{\phi_{bg}}(z_i|I_i)||p_{\theta_{bg}}(z_i)) \\
=&\mathbf{E}_{\prod_{i\in S} q_{\phi_{cls_i}}(z_i|I_i)\prod_{i\in B} q_{\phi_{bg}}(z_i|I_i)} \log p(I|\{z_i\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N) \\
&+ \sum_{i\in S\cup B} [\frac{N_z}{2}(1+\log(2\pi)) + \sum_{t=1}^{N_z} \log \tau_{i,t}] \\
&+ \sum_{i\in S} [\mathbf{E}_{q_{\phi_{cls_i}}(z_i|I_i)} \log p_{\mathcal{E}}(f_{\theta_{cls_i}}(z_i)) + \mathbf{E}_{q_{\phi_{cls_i}}(z_i|I_i)} \log |det \left(\frac{\partial f_{\theta_{cls_i}}(z_i)}{\partial z_i}\right)|] \\
&+ \sum_{i\in B} [\mathbf{E}_{q_{\phi_{bg}}(z_i|I_i)} \log p_{\mathcal{E}}(f_{\theta_{bg}}(z_i)) + \mathbf{E}_{q_{\phi_{bg}}(z_i|I_i)} \log |det \left(\frac{\partial f_{\theta_{bg}}(z_i)}{\partial z_i}\right)|].
\end{aligned}
\tag{4.7}
$$

The first term in equation 4.7 can be approximated as

$$
\begin{aligned}
&\mathbf{E}_{\prod_{i\in S} q_{\phi_{cls_i}}(z_i|I_i)\prod_{i\in B} q_{\phi_{bg}}(z_i|I_i)} \log p(I|\{z_i\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N) \\
&\approx \log p(I|\{z_i^*\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N),
\end{aligned}
\tag{4.8}
$$

where z_i^*'s are drawn from the predicted posterior distribution $z_i|I_i \sim \mathcal{N}(\mu_i, \Gamma_i)$. The set $\{z_i^*\}_{i\in S\cup B}$, S, B, $\{\mathbf{det}_i\}_{i=1}^N$ uniquely determines the distribution of pixels on the object masks associated with each detection in S and B. Employing the occlusion ordering method outlined in Section 4.3, we aggregate the means of the distributions pertaining to all detections in S and B to derive the whole reconstruction output $Canvas$, denoted as $Canvas$ and explicated in Section 4.5. Plugging in $Canvas$,

$$
\log p(I|\{z_i^*\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N) = -\frac{|I|}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|I - Canvas\|_{vec,2}^2 \tag{4.9}
$$

where $|I|$ signifies the aggregate count of pixels within I, equivalent to the total number of

pixels encompassed by the union of all masks $\cup_{i=1}^{N} mask_i$. According to equations 4.1, 4.3, 4.7, 4.8 and 4.9, and under the assumption of a uniform prior $p(S, D, B, \{\mathbf{det}_i\}_{i=1}^{N})$, the task of maximizing $\log p(I, S, D, B, \{\mathbf{det}_i\}_{i=1}^{N})$ is equivalently transformed into the minimization of our defined loss function

$$
\begin{aligned}
L = \|I - Canvas\|_{vec,2}^2 - 2\sigma^2 \sum_{i \in S \cup B} [\frac{N_z}{2}(1 + \log(2\pi)) + \sum_{t=1}^{N_z} \log \tau_{i,t}] \\
- 2\sigma^2 \sum_{i \in S} [\mathbf{E}_{q_{\phi_{cls_i}}(z_i|I_i)} \log p_{\mathcal{E}}(f_{\theta_{cls_i}}(z_i)) + \mathbf{E}_{q_{\phi_{cls_i}}(z_i|I_i)} \log |det \left(\frac{\partial f_{\theta_{cls_i}}(z_i)}{\partial z_i} \right)|] \\
- 2\sigma^2 \sum_{i \in B} [\mathbf{E}_{q_{\phi_{bg}}(z_i|I_i)} \log p_{\mathcal{E}}(f_{\theta_{bg}}(z_i)) + \mathbf{E}_{q_{\phi_{bg}}(z_i|I_i)} \log |det \left(\frac{\partial f_{\theta_{bg}}(z_i)}{\partial z_i} \right)|].
\end{aligned}
\tag{4.10}
$$

The purpose of DSAM, as elucidated in Section 4.6, is to minimize the loss function L defined in equation 4.10.

4.3 Occlusion Relationship Reasoning by MiDaS

Due to the potentially overlapping nature of object masks in instance segmentation He et al. [2017], it becomes imperative to address the intricacies of reasoning about occlusion relationships. Our approach involves a direct method for occlusion relationship reasoning: initially, we execute a depth estimation algorithm MiDaS Lasinger et al. [2019] on the entire image. Subsequently, we compute the "occlusion scores" by averaging the estimated values over each predicted object mask.

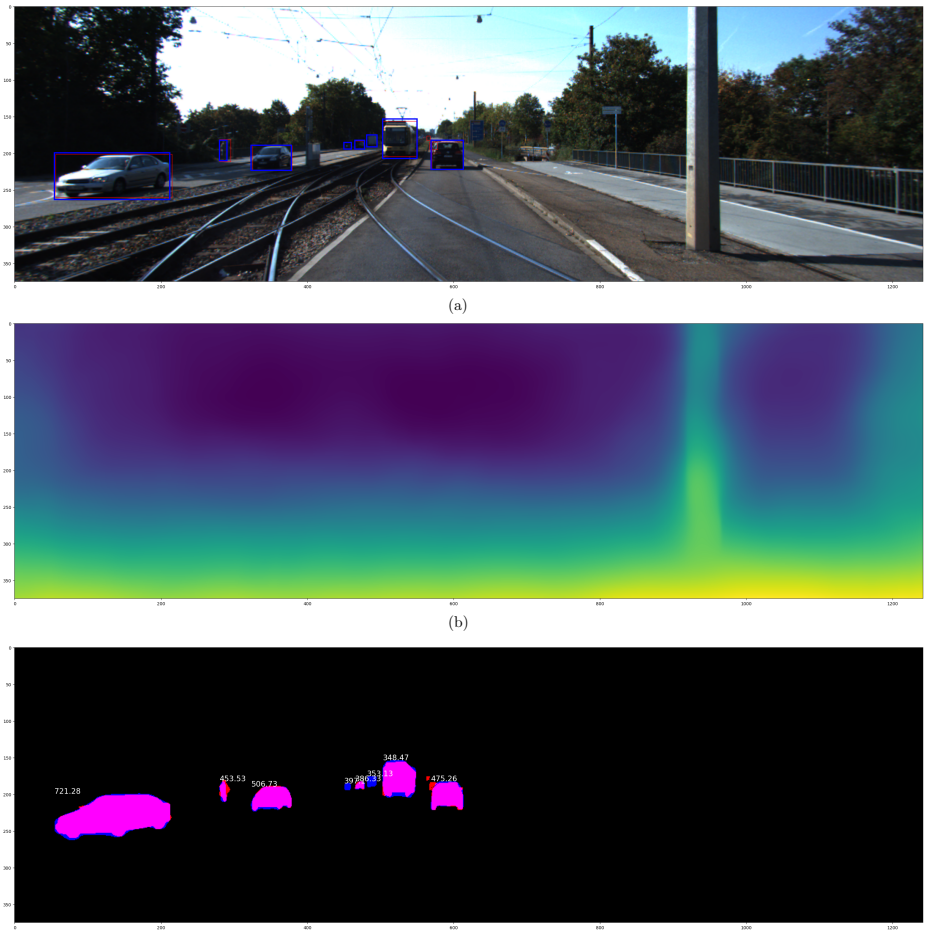
The MiDaS models are retrieved from the "torch.hub". Three distinct MiDaS models are available, namely "MiDaS_small", "DPT_Hybrid" and "DPT_Large". These models are arranged in ascending order of accuracy, yet in descending order of inference speed. In the MiDaS model dedicated to occlusion relationship reasoning, our selection is the "DPT_Hybrid" model. The predictions generated by MiDaS are expected to possess identical height and width dimensions as the original image, with the channel count reduced to 1. If the predic-

tions of MiDaS are represented as $MiDaS$, the occlusion score occ_i is defined as follows:

$$occ_i = \frac{\sum_{a \in mask_i} MiDaS_a}{\sum_{a \in mask_i} 1}.$$

Here, $mask_i$ denotes the predicted object mask of $\mathbf{det}_i$, and the variable a in the equation iterates through each pixel within $mask_i$. The notation $MiDaS_a$ signifies the predicted value by MiDaS at pixel a. MiDaS predicts the relative inverse depth. In the course of depth estimation by MiDaS, higher value indicates a closer region, therefore we operate under the assumption that object masks with higher occlusion scores can effectively obscure those with lower occlusion scores. Compared to the Faster R-CNN-OC described in Section 3.2, the occlusion relationship reasoning method based on MiDaS in this section is much more flexible and has far better generalization abilities because Faster R-CNN-OC is only trained with paired objects.

An illustrative instance is presented in Figure 4.3. Image (a), sized at $(375, 1242, 3)$, serves as our original image, with ground truth bounding boxes and predicted bounding boxes depicted in red and blue, respectively. The *stuff* classes include road, trees and sky etc., but we treat them as one *stuff* class called "background". Image (b) displays the results of relative inverse depth estimation for image (a) by MiDaS, characterized by dimensions $(375, 1242, 1)$, where a lighter color signifies a higher predicted value. While acknowledging that this relative inverse depth estimation may not be flawless, we consider it adequate for our occlusion relationship reasoning. Image (c) portrays the computed occlusion scores for each predicted object mask at the upper left corner, with ground truth object masks in red, predicted ones in blue, and their overlapping areas in pink.



(a)

(b)

(c)

Figure 4.3: Example of our occlusion relationship reasoning by MiDaS. (a) - original image with predictions in blue boxes and ground truths in red boxes, (b) - relative inverse depth estimation from MiDaS, (c) - predicted object masks in blue, ground truth object masks in red, their overlaps in pink, and occlusion scores of object masks at the upper left corner of corresponding object masks.

4.4 Single Reconstruction Algorithm for DSAM (DSAMSR)

Variational auto-encoders (VAEs) Kingma and Welling [2013] represent a category of Deep Generative Models (DGMs) that postulate the existence of a latent code z within the latent space and posit a conditional distribution $x|z$. Owing to the computational challenges associated with the log marginal likelihood

$$\log p(x) = \log \int p_\eta(x|z)p(z)dz \tag{4.11}$$

where η is the decoder parameters, VAEs instead maximize a variational evidence lower bound objective (ELBO)

$$\mathcal{L}(\eta, \phi; x) = \mathbf{E}_{q_\phi(z|x)}(\log p_\eta(x|z)) - \mathbf{D}_{KL}(q_\phi(z|x)||p(z)). \tag{4.12}$$

A full VAE comprises an encoder and a decoder, with their parameters represented as ϕ and η in Equation 4.12. VAEs can undergo training using the reparameterization trick as outlined in Rezende et al. [2014]. Following the training of a VAE, it becomes possible to sample z from the prior distribution $p(z)$ and transmit it to the decoder for the generation of new samples. Due to the fact that

$$\log p(x) = \mathcal{L}(\eta, \phi; x) + \mathbf{D}_{KL}(q_\phi(z|x)||p_\eta(z|x)), \tag{4.13}$$

$\mathcal{L}(\eta, \phi; x)$ can be a good approximation to $\log p(x)$ if $\mathbf{D}_{KL}(q_\phi(z|x)||p_\eta(z|x))$ is small.

During the maximization of Equation 4.12, it's commonly assumed that $p_\eta(x|z)$ adheres to a normal distribution $\mathcal{N}(m_{\theta,z}, \sigma^2\mathbf{I})$, where $m_{\theta,z}$ signifies the output of the decoder when provided with the latent code z, and $\mathbf{I}$ represents the identity matrix. Let the density $q_\phi(z|x)$ of the posterior distribution $z|x$ be Gaussian, specifically $\mathcal{N}(\mu_x, \Gamma_x)$, where μ_x is a N_z-dimensional vector and Γ_x is a $N_z \times N_z$ diagonal matrix

$$\Gamma_x = \begin{bmatrix} \tau_{x,1}^2 & & \\ & \ddots & \\ & & \tau_{x,N_z}^2 \end{bmatrix} \tag{4.14}$$

and $\tau_{x,t} > 0$, $t = 1, 2, ..., N_z$. If the prior $p(z)$ is $\mathcal{N}(\mathbf{0}, \mathbf{I}_{N_z})$, then

$$\mathbf{D}_{KL}(q_\phi(z|x)||p(z)) = \frac{\|\mu_x\|_2^2 + \sum_{t=1}^{N_z} \tau_{x,t}^2 - N_z}{2} - \sum_{t=1}^{N_z} \log \tau_{x,t}. \tag{4.15}$$

If a normalizing flow $f_\theta : \mathcal{Z} \to \mathcal{E}$ is assumed as the prior $p_\theta(z)$, it implies

$$\log p_\theta(z) = \log p_{\mathcal{E}}(f_\theta(z)) + \log |det \left(\frac{\partial f_\theta(z)}{\partial z} \right)|,$$

where $p_{\mathcal{E}}(\cdot)$ is the probability density function of gaussian noise $\mathcal{N}(\vec{0}, \mathbf{I}_{N_z})$. Then

$$\begin{aligned}
&\mathbf{D}_{KL}(q_\phi(z|x)||p_\theta(z)) = \mathbf{E}_{q_\phi(z|x)} \log q_\phi(z|x) - \mathbf{E}_{q_\phi(z|x)} \log p_\theta(z) \\
=&\mathbf{E}_{q_\phi(z|x)} \log q_\phi(z|x) - \mathbf{E}_{q_\phi(z|x)} \log p_{\mathcal{E}}(f_\theta(z)) - \mathbf{E}_{q_\phi(z|x)} \log |det \left(\frac{\partial f_\theta(z)}{\partial z} \right)| \\
=&-\frac{N_z}{2}(1 + \log(2\pi)) - \sum_{t=1}^{N_z} \log \tau_{x,t} - \mathbf{E}_{q_\phi(z|x)} \log p_{\mathcal{E}}(f_\theta(z)) - \mathbf{E}_{q_\phi(z|x)} \log |det \left(\frac{\partial f_\theta(z)}{\partial z} \right)|.
\end{aligned} \tag{4.16}$$

Deviating from the Single Reconstruction Algorithm for DSA (DSASR) discussed in Chapter 3, the present section introduces our Single Reconstruction Algorithm for DSAM (DSAMSR). Notably, our DSAMSR excludes the latent code optimization scheme, opting instead for predictions from the encoder, primarily due to its significantly faster performance. In Chapter 3, object reconstruction occasionally relies on incomplete pieces. It's challenging for the encoder to predict a reliable latent code if the input is incomplete. Consequently, latent code optimization becomes a necessity in Chapter 3. To transition to encoder predictions in the current chapter, we extract image contexts from the training images to facilitate the training of our Deep Generative Model. Despite potential occlusions and the presence of clutter from other objects and backgrounds within these image contexts, we still employ

them in model training. This decision is rooted in the absence of an assumption that clean object representations are available from unobstructed object images. To address issues related to clutter and occlusion, during training our encoder takes the entire image context as input, while the reconstruction loss by the decoder is solely evaluated on the annotated ground truth object masks. This training approach enables our Deep Generative Model to concentrate on the object itself, disregarding extraneous clutter.

As outlined in Algorithm 4, for the detection $\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)$ our DSAMSR utilizes the trained generative model to reconstruct the image context while simultaneously estimating $\mathbf{E}_{q_{\phi_c}(z|x)} \log p_{\mathcal{E}}(f_{\theta_c}(z))$ and $\mathbf{E}_{q_{\phi_c}(z|x)} \log |det \left(\frac{\partial f_{\theta_c}(z)}{\partial z} \right)|$ through sample averages. The estimation of the sample average is conducted through the drawing of 100 samples, a practice aimed at mitigating the variance of the estimation. It is pertinent to note that the parameters ϕ_c and θ_c are specific to class label c. The class label c may take on either the value of the predicted label cls_i or bg. This distinction is crucial for determining whether the image context represents a background segment during the execution of DSAM. Further elaboration on this matter is provided in Section 4.6.

An illustrative example is presented in Figure 4.4. Image (a) represents the image context, sized at $(158, 158, 3)$, surrounding a target bounding box. Utilizing the generative model trained with the category *cars*, the single reconstruction is depicted in image (b), exhibiting a squared error loss of 777.1679 when compared to image (a), with estimated *latentLL* and *logD* values in Algorithm 4 equal to -96.5977 and -87.3579 respectively. In contrast, the single reconstruction using the generative model trained with background pieces is illustrated in image (c), featuring a squared error loss of 984.1468 when compared to image (a), *latentLL* of -102.5404, and *logD* of -94.3730. Evidently, the single reconstruction in image (b) surpasses that in image (c) in terms of squared error loss, *latentLL*, and *logD*.

The generative model, trained with background pieces, undergoes training as if "background" were a distinct class label. Due to the occasional misidentification of background

Algorithm 4: Single Reconstruction Algorithm for DSAM (DSAMSR)

Input: Image context I_i, the detection $\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)$, generative model parameters η, ϕ, θ, generative model training image size $(d, d, 3)$ and N_z, type of operation 'S' or 'B'

$I_i \leftarrow$ resize I_i to $(d, d, 3)$;

if *operation = 'S'* **then**
| $c = cls_i$

else
| $c = bg$

end

$\mathcal{N}(\mu_i, \Gamma_i) \leftarrow$ posterior distribution predicted by the encoder of ϕ_c ; `/* covariance matrix` Γ_i `is diagonal with diagonal elements` $(\tau^2_{i,1}, \tau^2_{i,2}, ..., \tau^2_{i,N_z})$ `*/`

$z_i^* \leftarrow$ sampled from $\mathcal{N}(\mu_i, \Gamma_i)$;

$R_i \leftarrow$ output from the decoder of η_c using latent code z_i^* ;

$R_i \leftarrow$ resize R_i to the original image context size ;

$latentLL_i \leftarrow$ sample average of $\log p_{\mathcal{E}}(f_{\theta_c}(z))$ by drawing 100 z's from $\mathcal{N}(\mu_i, \Gamma_i)$;

$logD_i \leftarrow$ sample average of $\log |det \left(\frac{\partial f_{\theta_c}(z)}{\partial z} \right)|$ by drawing 100 z's from $\mathcal{N}(\mu_i, \Gamma_i)$;

Output: Single reconstruction R_i, and μ_i, Γ_i, $latentLL_i$, $logD_i$, $mask_i$

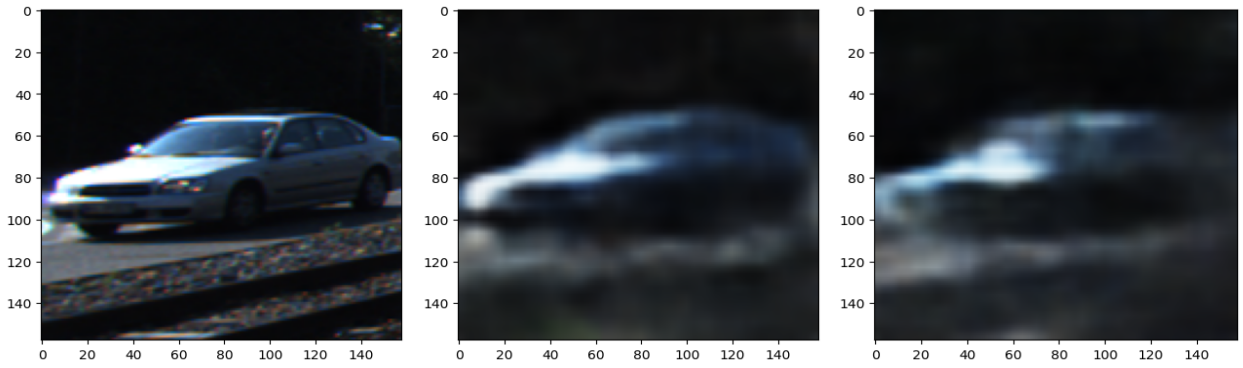


Figure 4.4: Example of the DSAMSR. (a) - image context, (b) - single reconstruction using the predicted label, (c) - single reconstruction by treating it as background.

pieces as objects by the base instance segmentation algorithm such as Mask R-CNN He et al. [2017], our DSAM and DSAMWR involve a comparison of losses for the predicted class label and "background" for each detection. In cases where the class label is accurately specified, the predicted posterior distribution $\mathcal{N}(\mu_i, \Gamma_i)$ typically aligns with the common regions in the latent space for that class. Consequently, we anticipate obtaining a satisfactory single reconstruction, along with *latentLL* and *logD*. If the class label is not correctly specified, we may observe superior outcomes in terms of single reconstruction, *latentLL*, and *logD* for the "background" class. Our defined loss aids in determining which scenario is more probable. Further details are provided in Section 4.5 and 4.6.

4.5 Whole Reconstruction Algorithm for DSAM (DSAMWR)

Much like the Whole Reconstruction Algorithm for DSA (DSAWR) in Section 3.4, our new Whole Reconstruction Algorithm for DSAM (DSAMWR), as summarized in Algorithm 5, initiates with a pure black *Canvas* of the same dimensions as the original image. However, in the context of DSAM, the original image does not necessarily feature a pure black background. To address this discrepancy, our algorithm permits the selection of certain detections as background pieces, which is called "designation as background", and incorporates

their single reconstructions as the "background" class (e.g., image (c) in Figure 4.4) onto the *Canvas*.

Collectively, we have three operations on detections: "selection", "discarding", and "designation as background". During each processing step in DSAM, we opt for one of these three operations for the designated detection. The guideline for determining the operations is postponed to Section 4.6. The "discarding" operation entails the rejection of the detection. The "selection" operation implies that the detection is deemed accurate, and its single reconstruction, using its predicted label (e.g., image (b) in Figure 4.4), is integrated into the *Canvas* by our DSAMWR algorithm. The "designation as background" operation signifies that the detection is considered incorrect, but its single reconstruction as the "background" class (e.g., image (c) in Figure 4.4) is utilized in our DSAMWR. It is important to note that the sets S and B represent all the detections under the "selection" and "designation as background" operations, respectively. Our DSAMWR requires all the single reconstructions of the detections in S and B, as illustrated in Algorithm 5.

It is worth noting that, irrespective of the specific content of sets S and B, the image context for a detection $\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)$ remains constant. Consequently, if a detection $\mathbf{det}_i$ has undergone DSAMSR previously, we can store the outcomes of its single reconstruction. This elucidates the presence of a "reconstructions hashmap" denoted as *ReconDict* in Algorithm 5. The key of the *ReconDict* encompasses two components: the index of the detection and the operation employed in that specific single reconstruction.

Examples of the single reconstructions are depicted in Figure 4.4. However, when incorporating a single reconstruction onto the *Canvas* in Algorithm 5, we confine the reconstruction within its respective object mask $mask_i$. The predicted object mask $mask_i$, with values ranging from 0 to 1 at each pixel across the entire image, is binarized by truncating it at the threshold of 0.5. Following the binarization process, pixels with a value of 1 are considered to represent the object's support. The consideration of occlusion sequences in the

single reconstructions demands meticulous attention. To establish the occlusion sequence, we employ the occlusion scores introduced in Section 4.3. Prior to placing single reconstructions on the *Canvas*, we arrange detections $\{\mathbf{det}_i\}_{i \in S \cup B}$ in descending order based on their occlusion scores. As outlined in Algorithm 5, we commence with those possessing higher occlusion scores and fill in the remaining blank pixels using detections with lower occlusion scores. This guarantees that objects with higher occlusion scores will occlude those with lower occlusion scores.

For the purpose of likelihood comparison introduced in Section 4.2, DSAMWR also computes the loss function. The loss function in DSAMWR aligns with the definition in Equation 4.10. Given any S and B, DSAMWR yields the loss and the reconstructions hashmap. Since DSAM in Section 4.6 utilizes DSAMWR multiple times, retaining the reconstructions hashmap and reusing the single reconstructions can assist in reducing computational costs. Moreover, the loss function provided by DSAMWR is utilized in the comparison of those three operations by DSAM.

Algorithm 4 (DSAMSR) mandates the target image context to be a perfect square. In cases where this requirement is not met, Algorithm 5 (DSAMWR) automatically substitutes $\mu_i = \vec{0}$, $\Gamma_i = \mathbf{I}_{N_z}$, $latentLL_i = 0$, and $logD_i = 0$, using the target image context itself as the single reconstruction. In such instances, the detection is automatically included in the set S by the DSAM algorithm in Section 4.6. This offers a straightforward solution for image contexts that are not perfect squares. However, for a more meticulous analysis, latent code optimization similar to that described in Section 3.3 can be applied here.

Illustrated in Figure 4.5, Algorithm 5 positions single reconstructions on a blank *Canvas*, with each single reconstruction confined to the region outlined by its predicted object mask. In the lower right corner of Figure 4.5, the single reconstruction for that detection is created by directly applying the original image, as described earlier. The majority of the *Canvas* remains blank because we only reconstruct and analyze the pixels within the union

Algorithm 5: Whole Reconstruction Algorithm for DSAM (DSAMWR)

Input: The original image I^*, the image constrained within the union of all predicted object masks I, detection results $\{\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)\}_{i=1}^{N}$, the sets S, B, reconstructions hashmap $ReconDict$, assumed variance $\sigma^2 > 0$, latent dimension N_z

Sort $\{\mathbf{det}_i\}_{i \in S \cup B}$ according to occ_i from high to low ;

$Canvas \leftarrow zeros(H, W, 3)$; `/* (H, W, 3) is the image size */`

$L \leftarrow 0$; `/* loss defined in equation 4.10 */`

for $i \in S \cup B$ **do**

  if $i \in S$ **then**

    operation = 'S'

  else

    operation = 'B'

  end

  if $ReconDict[i, operation]$ *doesn't exist* **then**

    $l \leftarrow$ maximum between the height and width of bb_i ;

    $bb_i^* \leftarrow$ the $l \times l$ square centered at the center of bb_i ;

    if bb_i^* *isn't completely within the image borders* **then**

      $ReconDict[i, operation] \leftarrow (I^*[bb_i], \vec{0}, \mathbf{I}_{N_z}, 0, 0)$; `/* I*[bb_i] is obtained by cropping the image at bounding box bb_i */`

    else

      $I_i \leftarrow I^*[bb_i^*]$;

      $(R_i, \mu_i, \Gamma_i, latentLL_i, logD_i) \leftarrow DSAMSR(I_i, \mathbf{det}_i, operation)$;

      $R_i \leftarrow$ crop a box of size bb_i at the center of R_i ;

      $ReconDict[i, operation] \leftarrow (R_i, \mu_i, \Gamma_i, latentLL_i, logD_i)$;

    end

  end

  $(R_i, \mu_i, \Gamma_i, latentLL_i, logD_i) \leftarrow ReconDict[i, operation]$;

  $L \leftarrow L - 2\sigma^2[\frac{N_z}{2}(1 + \log(2\pi)) + \sum_{t=1}^{N_z} \log \tau_{i,t} + latentLL_i + logD_i]$;

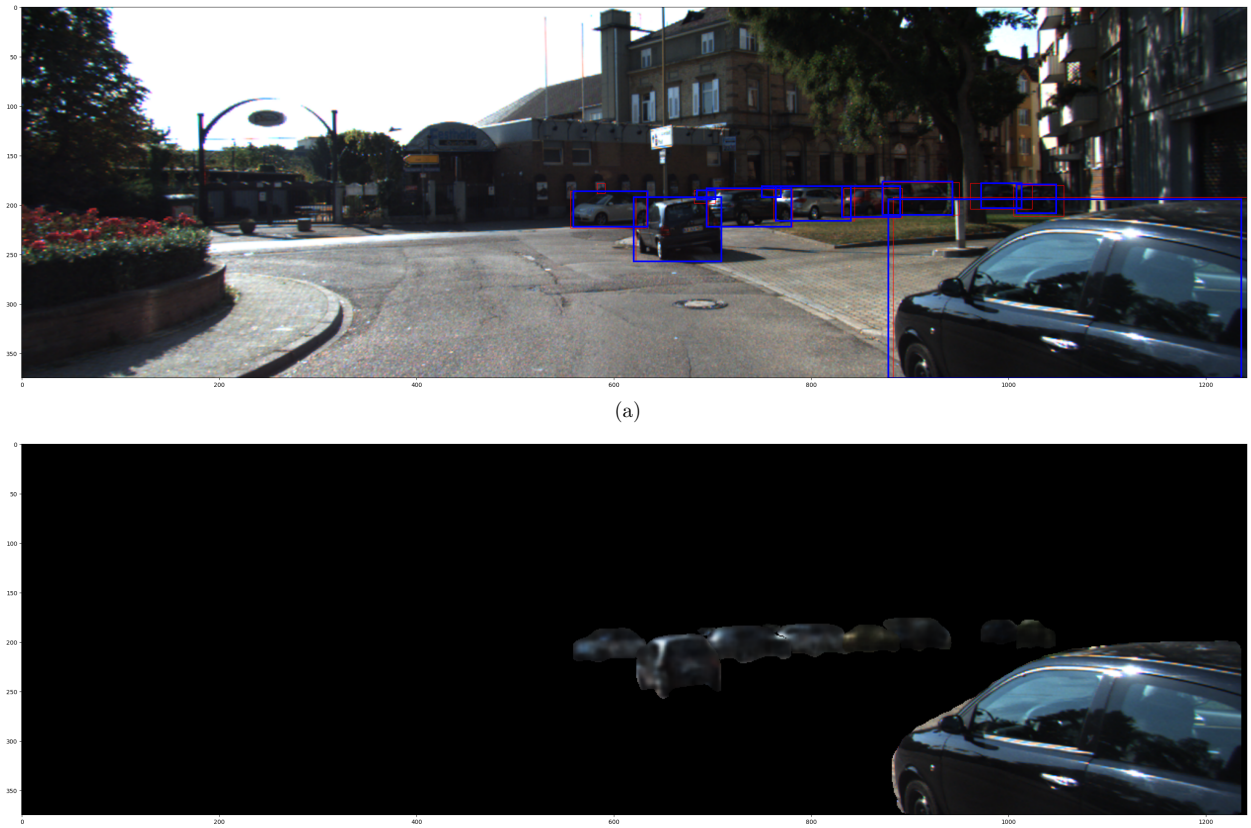
  Fill in the blank pixels in $Canvas[mask_i]$ by the corresponding values in R_i ;

  `/* "blank pixel" is a pixel with value` $(0, 0, 0)$ `*/`

end

$L \leftarrow L + \|I - Canvas\|_{vec,2}^2$;

Output: Loss L, reconstructions hashmap $ReconDict$



(a)

(b)

Figure 4.5: Example of the DSAMWR. (a) - original image with bounding boxes, (b) - The *Canvas* by the DSAMWR given a specific S and B.

$\cup_{i=1}^{N} mask_i$. The loss, as defined in Section 4.2, relies solely on the union $\cup_{i=1}^{N} mask_i$.

4.6 Detection Selection Algorithm with Mask (DSAM)

4.6.1 DSAM

Our Detection Selection Algorithm with Mask (DSAM) operates as a greedy algorithm that allocates detections to one of the three sets: S, D, or B, corresponding to three operations "selection", "discarding", and "designation as background". As depicted in Algorithm 6, the algorithm commences with empty sets S_0, D_0, B_0, and an empty "reconstructions hashmap" *ReconDict*. The processing of detection results adheres to objectness scores, arranged in

descending order. When processing detection $\mathbf{det}_i$, the algorithm compares the losses associated with the following three scenarios:

- $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1} \cup \{i\}, B_{i-1}$
- $S_i, D_i, B_i \leftarrow S_{i-1} \cup \{i\}, D_{i-1}, B_{i-1}$
- $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1}, B_{i-1} \cup \{i\}$

Among which the first case assumes that $\mathbf{det}_i$ is discarded, the second case assumes that $\mathbf{det}_i$ is selected as an object with its predicted label, and the third case assumes that $\mathbf{det}_i$ represents a background piece. At each time, the sets S_i, D_i, B_i are input into the Whole Reconstruction Algorithm for DSAM (DSAMWR), and DSAMWR assists in computing the loss as defined in Equation 4.10. The set S_i, D_i, B_i is determined based on the scenario with the smallest loss.

In Algorithm 6, three losses are compared for a detection $\mathbf{det}_i$: L_{i-1}, $L_{i,1}$ and $L_{i,2}$. Because the loss given by Algorithm 5 (DSAMWR) only depends on the detections in S and B, the two cases $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1} \cup \{i\}, B_{i-1}$ and $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1}, B_{i-1}$ yield exactly the same loss, and the latter is already computed as L_{i-1}. The loss $L_{i,1}$ represents the case $S_i, D_i, B_i \leftarrow S_{i-1} \cup \{i\}, D_{i-1}, B_{i-1}$ and the loss $L_{i,2}$ is calculated under $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1}, B_{i-1} \cup \{i\}$. In the event of ties, we prioritize L_{i-1}, followed by $L_{i,1}$, and lastly $L_{i,2}$.

In contrast to the Detection Selection Algorithm (DSA) discussed in Chapter 3, DSAM differs primarily by incorporating a "designation as background" operation. DSAM delegates the majority of computational tasks to DSAMWR and, due to computational complexity issues, does not involve a one-step back search.

An illustrative example is presented in Figure 4.6, where three detections are labeled as boxes 1, 2, and 3. Since the image context of box 1 extends beyond the image boundary, the image itself is employed as the single reconstruction, similar to the scenario depicted

Algorithm 6: Detection Selection Algorithm with Mask (DSAM)

Input: Detection results $\{\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)\}_{i=1}^{N}$ sorted by $score_i$ from high to low, assumed variance $\sigma^2 > 0$, latent dimension N_z

$S_0, D_0, B_0 \leftarrow \emptyset, \emptyset, \emptyset$;

$ReconDict \leftarrow \{\}$;

$L_0 \leftarrow \infty$;

for $i = 1$ **to** N **do**

  $(L_{i,1}, ReconDict) \leftarrow DSAMWR(S_{i-1} \cup \{i\}, D_{i-1}, B_{i-1}, ReconDict)$;

  $(L_{i,2}, ReconDict) \leftarrow DSAMWR(S_{i-1}, D_{i-1}, B_{i-1} \cup \{i\}, ReconDict)$;

  $L_i \leftarrow \min(L_{i-1}, L_{i,1}, L_{i,2})$;

  if $L_i = L_{i-1}$ **then**

    $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1} \cup \{i\}, B_{i-1}$;

  else if $L_i = L_{i,1}$ **then**

    $S_i, D_i, B_i \leftarrow S_{i-1} \cup \{i\}, D_{i-1}, B_{i-1}$;

  else

    $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1}, B_{i-1} \cup \{i\}$;

  end

end

Output: S_N, D_N, B_N

in the lower right corner of Figure 4.5. DSAM starts with the detection in box 1. In the first scenario, $S_1, D_1, B_1 \leftarrow \{\}, \{1\}, \{\}$, resulting in a loss of 14515.563. In the second scenario, $S_1, D_1, B_1 \leftarrow \{1\}, \{\}, \{\}$, and the third scenario, $S_1, D_1, B_1 \leftarrow \{\}, \{\}, \{1\}$, both yield a loss of 87.9987, with Algorithm 6 selecting $S_1, D_1, B_1 \leftarrow \{1\}, \{\}, \{\}$. Similarly, in the second step, Algorithm 6 compares $S_2, D_2, B_2 \leftarrow \{1\}, \{2\}, \{\}$, $S_2, D_2, B_2 \leftarrow \{1, 2\}, \{\}, \{\}$, and $S_2, D_2, B_2 \leftarrow \{1\}, \{\}, \{2\}$, ultimately selecting $S_2, D_2, B_2 \leftarrow \{1, 2\}, \{\}, \{\}$. In the third step, Algorithm 6 chooses $S_3, D_3, B_3 \leftarrow \{1, 2\}, \{\}, \{3\}$. Consequently, after image processing, boxes 1 and 2 are employed for panoptic segmentation. Despite boxes 2 and 3 being relatively small, Algorithm 4 resizes their image context to a fixed scale, as demonstrated in images (b) and (c) in Figure 4.6, enabling analysis on boxes 2 and 3.

4.6.2 from DSAM to Panoptic Segmentation

After the execution of DSAM, three sets, namely S, D, and B, are obtained. However, further processing is necessary to generate panoptic segmentation results. Panoptic segmentation involves the assignment of pixel-level semantic and instance ID labels. For each object detected by Mask R-CNN He et al. [2017], the predicted object mask and predicted class label are available. Once again, we apply binarization to the predicted object mask using a threshold of 0.5. The detections within the set S determined by DSAM are retained and utilized in panoptic segmentation. For instance, Figure 4.7 illustrates the panoptic segmentation produced by DSAM for the image (b) in Figure 4.5, with distinct instance IDs represented by different colors.

The objects in set S are arranged in descending order based on their occlusion scores. Object masks with higher occlusion scores are assigned first, followed by objects with lower occlusion scores. This process resembles an NMS-like procedure Kirillov et al. [2019b], with the distinction that our objects are ordered according to occlusion, whereas their objects are sorted by confidence scores. Once all the semantic and instance ID labels are assigned,

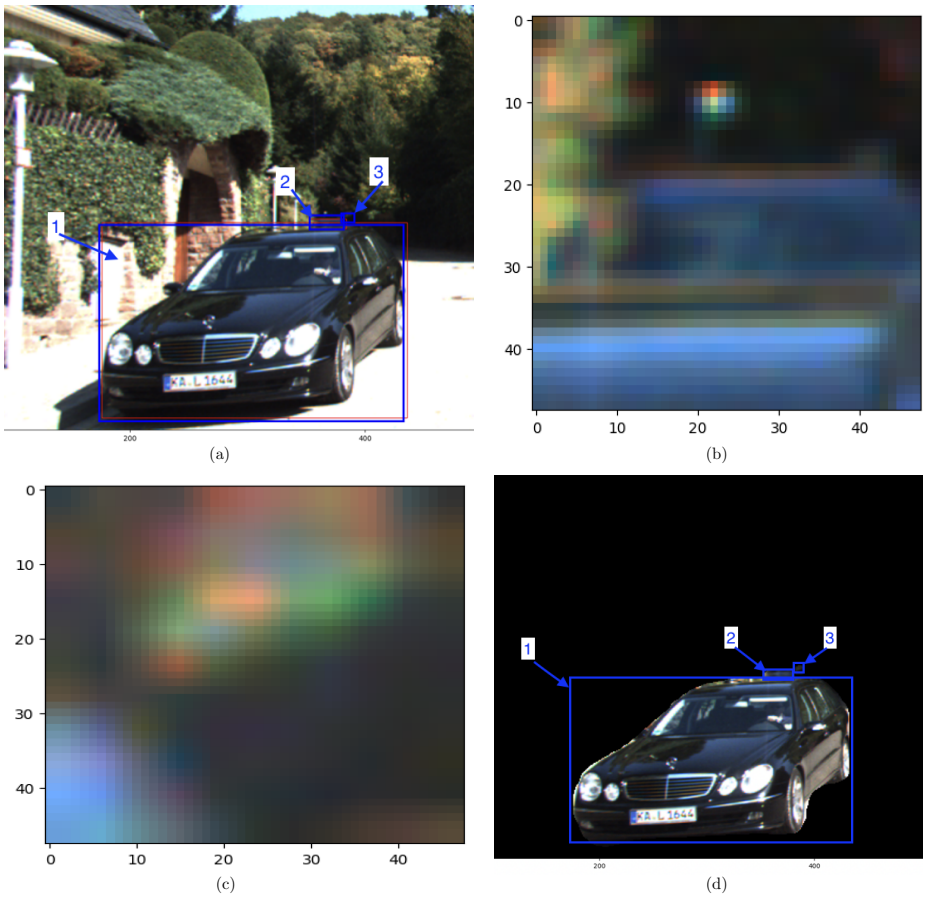


Figure 4.6: Example of DSAM. (a) - original image with bounding boxes, (b) - The resized image context of box 2, (c) - The resized image context of box 3, (d) - The *Canvas* of DSAMWR by $S_3, D_3, B_3 \leftarrow \{1, 2\}, \{\}, \{3\}$.

any remaining unlabeled pixels are designated as belonging to a *stuff* class referred to as "background," as discussed in Section 4.1.

4.7 Dataset and Experiments

The KITTI INStance dataset (KINS) Qi et al. [2019] has been generated by annotating 14,991 KITTI images Geiger et al. [2012] with amodal instance masks, inmodal instance masks, relative occlusion orderings, amodal bounding boxes, inmodal bounding boxes, and object categories. These images, sourced from street scenes, exhibit dimensions approximately equal to $(375, 1242, 3)$. The selection of this dataset is predicated on its comprehensive annotations.

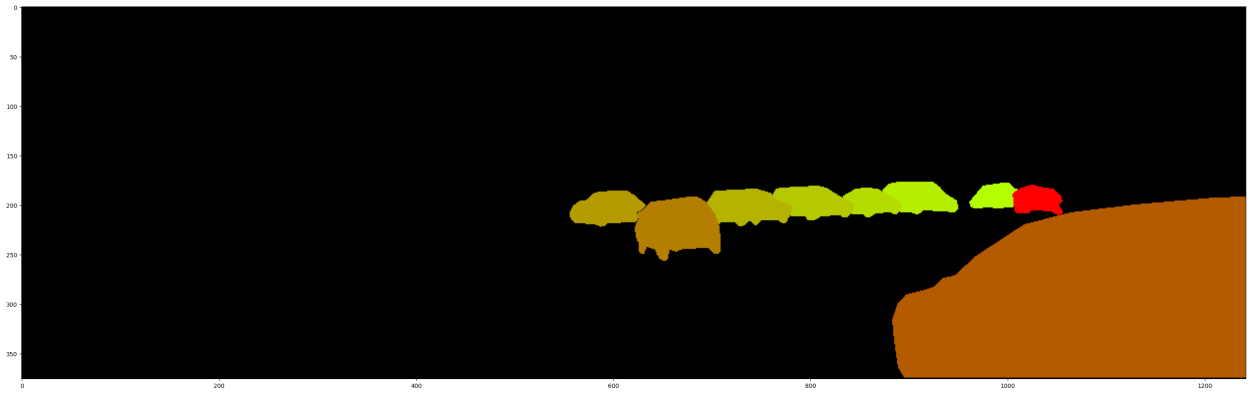


Figure 4.7: Example of Panoptic Segmentation by DSAM.

Comprising a total of 7,474 images allocated for training and 7,517 for testing, the dataset encompasses eight distinct categories: 'cyclist', 'pedestrian', 'person-sitting', 'car', 'tram', 'truck', 'van', and 'misc'. The first three categories fall under the general category 'people', while the remaining five are classified as 'vehicle'. The 'misc'. class encompasses diverse ambiguous vehicles that defy classification under other vehicle categories. On average, there are 12.53 objects per image. Addressing the occlusion aspect, 53.6% of objects experience partial occlusion, with an average occlusion ratio of 31.7%. Notably, these eight categories demonstrate a marked imbalance, loosely adhering to Zipf's law Piantadosi [2014], wherein the 'car' and 'truck' classes emerge as the most and least frequent, respectively. For a more in-depth exploration, please consult Qi et al. [2019].

The open sourced KINS dataset is available on `https://github.com/qqlu/Amodal-Instance-Segmentation-through-KINS-Dataset`. It furnishes comprehensive data for training instance segmentation models such as Mask R-CNN He et al. [2017]. Our implementation of Mask R-CNN undergoes training using the KINS training set, initialized with default weights pre-trained on the COCO dataset Lin et al. [2014]. Notably, we employ a batch size of 1 and set the train-validation split ratio at 8:2. The optimization process involves Stochastic Gradient Descent (SGD) with a learning rate of 0.005, momentum of 0.9, and weight decay set to 0.0005. Training ceases if no improvement in validation loss is

observed for five consecutive epochs, resulting in termination at epoch 21.

In the context of training our Deep Generative Model, image contexts undergo cropping when both dimensions of their inmodal bounding boxes measure at least 30 units in length. Subsequently, we rescale all cropped contexts to dimensions of $48 \times 48 \times 3$. Concurrently, we collect the annotated inmodal object masks and their corresponding ground truth class labels. This process results in a total of 49,594 samples, and the distribution of their classes is summarized in Table 4.1. Notably, the class 'person-sitting' is not represented among the samples, as there are no annotated bounding boxes for 'person-sitting' in either the training or testing images within the aforementioned open-source dataset. These object classes exhibit a pronounced imbalance, with the most prevalent category being 'car'.

Table 4.1: Class distribution of DGM training set

	cyclist	**pedestrian**	**personsitting**	**car**	**tram**	**truck**	**van**	**misc**
number	1751	5433	0	30674	889	484	2878	7485
ratio(%)	3.53	10.95	0	61.85	1.79	0.98	5.80	15.09

Conversely, for the 'background' class, we acquire training images for the Deep Generative Model (DGM) through random cropping. For an entire image with dimensions $(H, W, 3)$, we uniformly sample a size l from the range $Uniform[10, \min(H, W)]$, determining the dimensions of the bounding box as $l \times l$. Subsequently, we ascertain the upper-left corner by uniformly sampling integers $x_1 \sim Uniform[0, H - l]$ and $y_1 \sim Uniform[0, W - l]$. The original image is then cropped at the bounding box defined by its size and upper-left corner to yield a background piece. Given that the total number of valid annotated boxes in the KINS training set is 95,456, we sample an equivalent number of background pieces. The count of sampled background pieces in each image corresponds to the number of annotated boxes within that particular image.

As detailed in Table 4.1, certain object classes such as 'tram' and 'truck' exhibit a limited

number of samples, potentially insufficient for training a large Deep Generative Model. Consequently, we opt to implement a strategy wherein certain network architectures are shared among classes, reducing the number of parameters that require training for each individual object class. We propose a partially shared Deep Generative Model network architecture, outlined in Figure 4.8. Within this illustration, $q_\phi(z|x)$ denotes the posterior distribution of latent code z given an image x, and f_{θ_c} represents the normalizing flow specific to class c. This architecture is designed based on the Variational Autoencoder (VAE) with a flow prior, as expounded upon in Section 4.4. Notably, the encoder and decoder components are shared, whereas the normalizing flows are class-specific. We adhere to the assumptions outlined in Section 4.4. Combining Equation 4.12 and 4.16, for each class c, we maximize the variational evidence lower bound objective (ELBO)

$$\begin{aligned}\mathcal{L}(\eta, \phi, \theta_c; x) =& \mathbf{E}_{q_\phi(z|x)}(\log p_\eta(x|z)) + \mathbf{E}_{q_\phi(z|x)} \log p_{\mathcal{E}}(f_{\theta_c}(z)) \\ &+ \mathbf{E}_{q_\phi(z|x)} \log |det \left(\frac{\partial f_{\theta_c}(z)}{\partial z} \right)| - \mathbf{E}_{q_\phi(z|x)} \log q_\phi(z|x),\end{aligned} \tag{4.17}$$

where ϕ, η, θ_c denotes the parameters for encoder, decoder and normalizing flow for class c respectively. It is noteworthy that the computation of $p_\eta(x|z)$ is as follows:

$$\log p_\eta(x|z) = -\frac{|x|}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \times \frac{|x| \sum_{a \in mask} \|x_a - D_\eta(z)_a\|_2^2}{\sum_{a \in mask} 3}, \tag{4.18}$$

where the symbol $|x|$ denotes the cardinality of the image x, while σ^2 represents the variance as defined in Section 4.2. The decoding result of latent code z is denoted by $D_\eta(z)$, and the 3-dimensional value at pixel a is represented by x_a or $D_\eta(z)_a$. Additionally, the variable *mask* corresponds to the ground truth mask. Equation 4.18 is tantamount to utilizing the mean square error specifically at the object mask to characterize the mean square error across the entire reconstruction. This approach enables our model to prioritize the accuracy of object predictions while disregarding extraneous details or clutter.

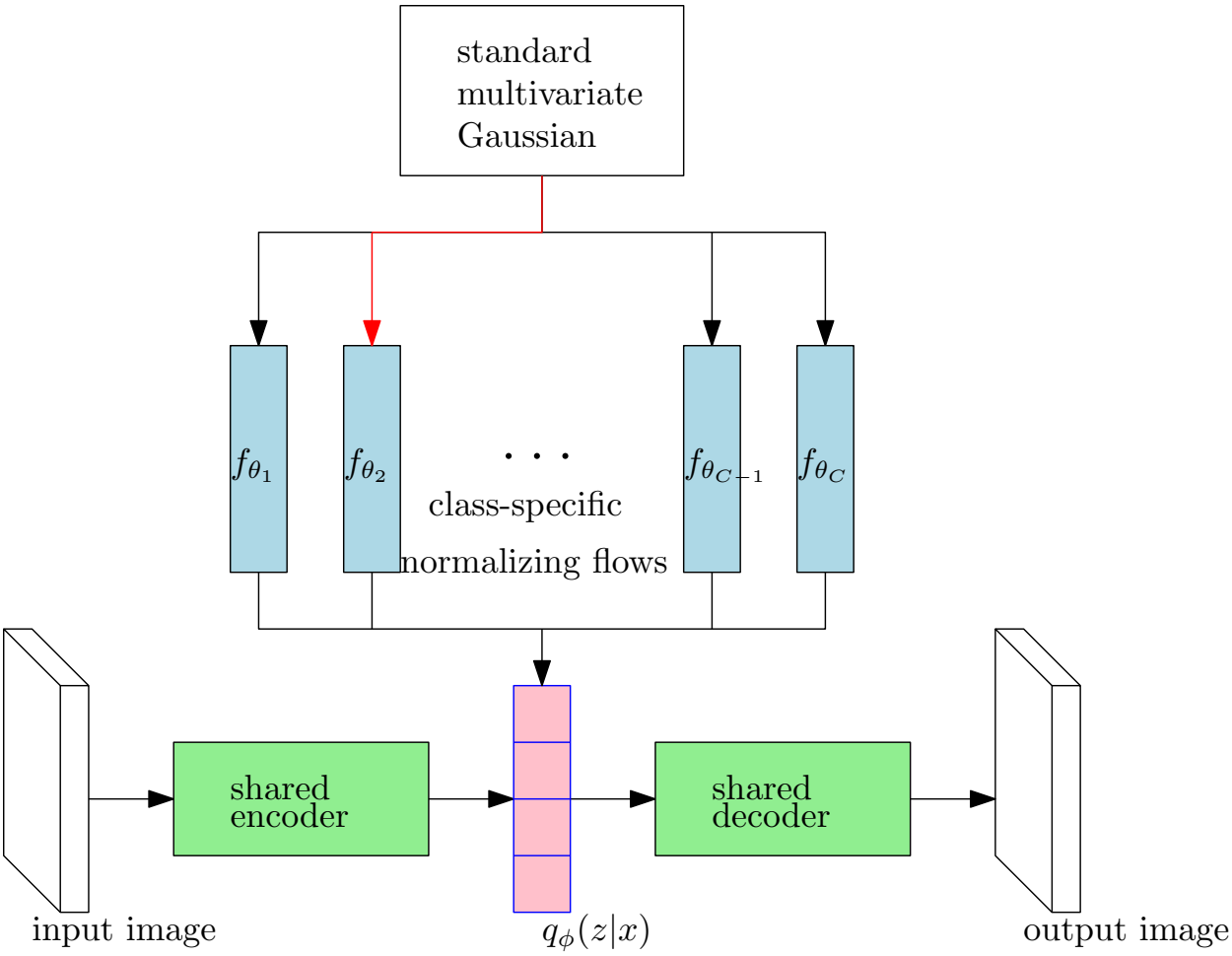


Figure 4.8: Our Deep Generative Model.

Similar to the approach adopted in Generative Latent Flow (GLF) Xiao et al. [2019], we employ the generator from InfoGAN Chen et al. [2016], which encompasses deconvolution layers, as our decoder. Correspondingly, the encoder utilizes a reversed network architecture involving convolution layers. The combined parameters for our encoder and decoder amount to 38,275,843 trainable parameters. For each class, our normalizing flow consists of four flow blocks, collectively contributing 1,335,808 parameters. The properties of the flow block, characterized by invertibility and computational efficiency, are delineated in Xiao et al. [2019]. The latent space dimension is fixed at 64. Employing a batch size of 256 and setting $\sigma = 0.05$, we utilize the Adam optimizer with a learning rate of 0.001 for training the encoder

and decoder, while the normalizing flow is trained with a learning rate of $1e-5$. The model undergoes training for a duration of 200 epochs.

To comprehensively assess our Deep Generative Model, we evaluate the variational evidence lower bound (ELBO) and its constituent components for each object class, as detailed in Table 4.2. These components, denoted as 'recon_ll', 'gauss_ll', 'log_jacob', and 'entropy', correspond to the four terms on the right-hand side of Equation 4.17, arranged sequentially. The values presented in Table 4.2 are estimates derived from a model trained 200 epochs, calculated as averages across the training set.

The ELBOs are predominantly influenced by the 'recon_ll' term, a consequence of the factors $|x| = 6912$ and $\frac{1}{2\sigma^2} = 200$ in Equation 4.18. The 'recon_ll' exhibits considerable variation across classes, reflecting diverse reconstruction challenges. While employing distinct σ^2 values for various classes could potentially address this discrepancy, we opt for a uniform σ^2 to prioritize simplicity, considering that our primary focus lies beyond refining the Deep Generative Model itself.

The combined influence of 'gauss_ll' and 'log_jacob' represents the log likelihood of latent codes. Notably, higher likelihoods are assigned to classes with larger sample sizes. In contrast, less frequent classes such as 'tram' and 'truck' exhibit the lowest 'gauss_ll' values. To optimize the ELBO objective across all training samples, the model tends to allocate popular classes to high probability regions.

Table 4.3 presents a summary of the variational evidence lower bounds (ELBOs) under the assumption of potentially mis-specified labels. The image data for all object classes is sourced from the KINS test set. The row names signify the specified labels, while the column names denote the corresponding data categories. Notably, bold text emphasizes the highest ELBO within each column. For conciseness, we exclude the 'tram' and 'truck' classes due to their limited sample sizes.

The table reveals instances where the model misclassifies 'cyclist' and 'van' as the predom-

Table 4.2: Four components of the ELBO

	cyclist	**pedestrian**	**car**	**tram**	**truck**	**van**	**misc**
recon_ll	-21955.9	-21141.3	-21783.1	-17036.5	-19893.6	-33794.0	-10139.5
gauss_ll	-133.7	-92.0	-99.6	-198.6	-269.3	-111.4	-86.6
log_jacob	-27.7	-51.5	-39.5	-16.7	-7.8	-52.6	-31.6
entropy	-111.6	-116.8	-125.5	-122.4	-125.3	-130.1	-101.7
ELBO	-22229.0	-21401.6	-22047.8	-17374.1	-20296.0	-34088.1	-10359.5

inant 'car' class, a trend observed in Table 4.2 where the 'car' class generally exhibits higher 'gauss_ll' plus 'log_jacob'. However, the model demonstrates proficiency in distinguishing between objects and background. Specifically, for object data, the background model yields the lowest ELBO, while for background data, generative models trained on objects yield notably low ELBO values.

Table 4.3: ELBO under mis-specified labels

	cyclist	**pedestrian**	**car**	**van**	**misc**	**background**
cyclist	-67872.7	-67542.5	-52524.6	-79234.7	-31236.9	-98431.7
pedestrian	-67952.1	**-67489.3**	-52467.	-79205.8	-31188.3	-98049.5
car	**-67821.7**	-67504.2	**-52457.1**	**-79142.3**	-31199.	-98407.9
van	-67885.5	-67501.9	-52489.1	-79181.5	**-31179.8**	-99836.1
misc	-67849.8	-67497.	-52479.5	-79159.4	-31202.1	-98973.6
background	-70228.4	-86666.3	-67137.6	-82507.5	-51675.7	**-24435.2**

The results obtained from both the NMS-like procedure Kirillov et al. [2019b] and our DSAM, assessed on KINS test images, are delineated in Table 4.4. Detections with objectness scores exceeding 0.1 are preserved for analysis. In the context of the table, the 'Baseline' row signifies the utilization of the NMS-like procedure independently with Mask R-CNN He et al.

[2017] output. Conversely, the 'with DSAM' row denotes the application of our DSAM to select a subset of detections from the same Mask R-CNN output, followed by the subsequent execution of the NMS-like procedure to generate panoptic segmentation.

Performance evaluation is conducted across four scenarios:

- Using all ground truth and predicted objects.
- Employing only ground truth and predicted objects with annotated or predicted object masks not less than 100 pixels.
- Similar to the previous scenario, with the threshold raised to 400 pixels.
- Similar to the previous scenario, with the threshold further raised to 900 pixels.

The progression from the first to the fourth scenario involves an increase in the average size of objects, rendering the task less challenging. As evident in Table 4.4, DSAM consistently enhances the Panoptic Quality (PQ) scores across all four scenarios. The improvement is particularly noteworthy in more challenging scenarios, where the task complexity increases.

Table 4.4: DSAM PQ with Objectness Scores Ordering

PQ	**all objects**	≥ 100 **pixels**	≥ 400 **pixels**	≥ 900 **pixels**
Baseline	0.411198	0.444078	0.582399	0.654408
with DSAM	0.479953	0.506556	0.607400	0.670859

The process, starting from our DSAM outcomes and leading to panoptic segmentation, can be executed utilizing occlusion score ordering, as elucidated in Section 4.6.2, while keeping the remaining configurations unchanged. The corresponding results are presented in Table 4.5. Once more, our DSAM demonstrates substantial enhancements across all four scenarios. The source code employed in our experiments has been made accessible at the following URL: `https://github.com/angzhifan/DSAM`. Should we refrain from removing

any objects, regardless of their sizes, in the absence of DSAM, the mean processing duration per image stands at approximately 3 seconds. Conversely, in the presence of DSAM, this average processing time escalates to approximately 34 seconds. However, when restricting consideration to objects possessing a minimum of 900 pixels in their object masks, the average processing time is at around 8 seconds without DSAM. In contrast, with DSAM, the average processing time experiences a more modest increase, reaching approximately 16 seconds.

Table 4.5: DSAM PQ with Occlusion Ordering

PQ	**all objects**	≥ 100 **pixels**	≥ 400 **pixels**	≥ 900 **pixels**
Baseline	0.388117	0.420399	0.560606	0.636605
with DSAM	0.471164	0.497567	0.597729	0.662387

4.8 Conclusion

In this chapter, the core algorithm under development is the Detection Selection Algorithm with Mask (DSAM), expounded in Section 4.6. DSAM serves as a post-processing technique designed for instance segmentation models such as Mask R-CNN He et al. [2017]. Analogous to Chapter 3, there exist two supplementary algorithms for DSAM: the Single Reconstruction Algorithm for DSAM (DSAMSR) outlined in Section 4.4, and the Whole Reconstruction Algorithm for DSAM (DSAMWR) detailed in Section 4.5. DSAM categorizes each detection into one of three operations: "selection", "discarding", or "designation as background". Only the detections subjected to the "selection" operation are preserved for panoptic segmentation purposes. The determination of these operations is contingent upon minimizing a loss function derived from an approximated log joint probability, as elucidated in Section 4.2. The post-processing outcomes of DSAM are subsequently applied to the panoptic segmenta-

tion task and assessed through the Panoptic Quality (PQ) scores Kirillov et al. [2019b]. The experiments conducted in Section 4.7 substantiate that DSAM yields significant improvements in panoptic segmentation quality. The enhancements observed in DSAM results come from the judicious selection and removal of detections, with the assignment of object labels dependent upon the underlying baseline instance segmentation models. This phenomenon is clearly illustrated in Table 4.3, wherein the VAE with flow prior encounters difficulty in effectively distinguishing between various object classes.

In contrast to the Detection Selection Algorithm (DSA) and its associated methodologies discussed in Chapter 3, DSAM operates within a significantly more intricate context, characterized by a broader array of object appearances and organizational variations. Given the constraints of the Faster-RCNN-OC outlined in Chapter 3 regarding occlusion relationship reasoning, we opt for the MiDaS Lasinger et al. [2019] package, mentioned in Section 4.3, as a more versatile depth estimation tool. DSAM distinguishes itself from other endeavors aimed at enhancing panoptic segmentation quality by not necessitating alterations or retraining of the instance segmentation model. Rather, it takes the instance segmentation model's output as input and applies post-processing to it. In contrast to much of the related research in panoptic segmentation that overlooks probabilistic interpretations, DSAM is firmly grounded in a probabilistic framework. Its fundamental concept relies on likelihood comparisons, with the learning of object distributions executed through Deep Generative Models. DSAM enhances traditional likelihood reasoning methods, such as the POP model Amit and Trouvé [2007], by integrating contemporary deep learning techniques.

CHAPTER 5

MAXIMIZING THE POSTERIOR FOR PANOPTIC SEGMENTATION

5.1 Motivation

In Chapter 4, our approach within the DSAM framework involves the utilization of Deep Generative Models for modeling the probability distributions associated with objects. The selection of these Deep Generative Models should adhere to the following constraints:

- Be able to calculate or approximate $p(x)$, the likelihood of the image segment x.
- Process an encoder and a decoder, as there are instances where training involves loss on incomplete images, such as image segments confined to an object mask.
- Ensure efficient and rapid evaluation of the likelihood or approximated likelihood, and is fast to sample new images from the model.

The first and third constraints facilitates the computation of joint likelihood, as delineated in the probabilistic framework given in Section 4.2. The second constraint allows the Deep Generative Model to encode the entire image context while assessing the reconstruction loss solely within the confines of the object mask. This methodology serves to improve the modeling of the appearance of objects.

Among the Deep Generative Models deliberated upon in Section 1.2.1, Generative Adversarial Networks (GANs) Goodfellow et al. [2014], Radford et al. [2015], Arjovsky et al. [2017] fail to adhere to the first and third constraints, as they do not explicitly provide likelihood. A majority of Auto-regressive Models Larochelle and Murray [2011], Uria et al. [2014], Van den Oord et al. [2016], Normalizing Flows Tabak and Turner [2013], Dinh et al. [2014, 2016], Kingma and Dhariwal [2018], and Deep Energy-based Models Du and Mordatch

[2019], Welling and Teh [2011] do not satisfy the second constraint. While Diffusion Models Ho et al. [2020], Xu et al. [2023] fulfill the first two constraints, they fall short of meeting the third requirement due to their sluggish computational speed. Variational Auto-encoders (VAEs) Kingma and Welling [2013], Rezende et al. [2014], Burda et al. [2015] emerge as the most suitable choice for our objectives, as they satisfy all three specified constraints.

For a given input image x, the Variational Auto-encoder (VAE) utilizes an encoder to predict a posterior distribution $q_\phi(z|x)$ within the latent space, where ϕ denotes the parameters of the encoder. The aggregated approximate posterior, expressed as

$$q_\phi(z) = \mathbf{E}_{x\sim p_x} q_\phi(z|x),$$

captures the marginal distribution of the latent code z under p_x, and p_x represents the distribution of the training images. Conversely, a decoder, parameterized by η, maps any latent code z in the latent space to its corresponding image x in the image space. Following the training of a VAE, the sampling procedure involves drawing a latent code z from a predefined prior distribution $p(z)$ and subsequently passing it to the decoder to generate an image.

To align the distribution of generated images with that of training images, the Variational Auto-encoder (VAE) penalizes $\mathbf{D}_{KL}(q_\phi(z|x)||p(z))$ within its variational evidence lower bound objective (ELBO), as depicted in Equation 4.12. It has been established Hoffman and Johnson [2016] that the ELBO objective can be reformulated as the "average reconstruction" minus index-code mutual information and further subtracting $\mathbf{D}_{KL}(q_\phi(z)||p(z))$.

However, the enforced alignment between $q_\phi(z)$ and $p(z)$ may compromise the "average reconstruction" performance of the decoder, particularly when $p(z)$ is assumed to be a simplistic distribution like factorial Gaussians. Consequently, in Chapter 4, DSAM employs VAE with a flow prior Huang et al. [2017], an enhanced version of VAE. This variant incorporates a learnable flow model Rezende and Mohamed [2015], Kingma et al. [2016] to

parameterize the prior in VAE. With this enhancement, the prior is no longer constrained to factorial Gaussians, and it possesses the capability to model complex distributions. As elaborated in Equation 4.12 and 4.16, the VAE with flow prior maximizes a variational evidence lower bound objective (ELBO) defined as

$$\begin{aligned}\mathcal{L}(\eta, \phi, \theta; x) =& \mathbf{E}_{q_\phi(z|x)}(\log p_\eta(x|z)) + \mathbf{E}_{q_\phi(z|x)} \log p_{\mathcal{E}}(f_\theta(z)) \\ &+ \mathbf{E}_{q_\phi(z|x)} \log |det \left(\frac{\partial f_\theta(z)}{\partial z} \right)| - \mathbf{E}_{q_\phi(z|x)} \log q_\phi(z|x),\end{aligned} \tag{5.1}$$

where f_θ is the normalizing flow parameterized by θ and $p_{\mathcal{E}}(\cdot)$ is the density of standard multivariate Gaussian.

In this chapter, we employ a distinct type of Deep Generative Model known as Generative Latent Flow (GLF) Xiao et al. [2019]. GLF functions through a deterministic auto-encoder (AE) combined with a normalizing flow Tabak and Turner [2013], Tabak and Vanden-Eijnden [2010]. In contrast to the VAE encoder, which predicts a posterior distribution $q_\phi(z|x)$, the AE encoder predicts a deterministic latent code z for the image x. The decoder component of AE operates similarly to that of VAE, with AE serving as a mapping between the image space and the latent space.

Similar to VAE with a flow prior, the normalizing flow in GLF also operates within the latent space. The AE reduces the dimensionality from the image space to the latent space, thereby reducing the computational complexity of the normalizing flow. GLF is trained to maximize the following objective:

$$\mathcal{G}(\eta, \phi, \theta) = \mathbf{E}_{x \sim p_x}(\log p(x|D_\eta(E_\phi(x)))) + \mathbf{E}_{x \sim p_x} \log p_{\mathcal{E}}(f_\theta(z)) + \mathbf{E}_{x \sim p_x} \log |det \left(\frac{\partial f_\theta(z)}{\partial z} \right)|, \tag{5.2}$$

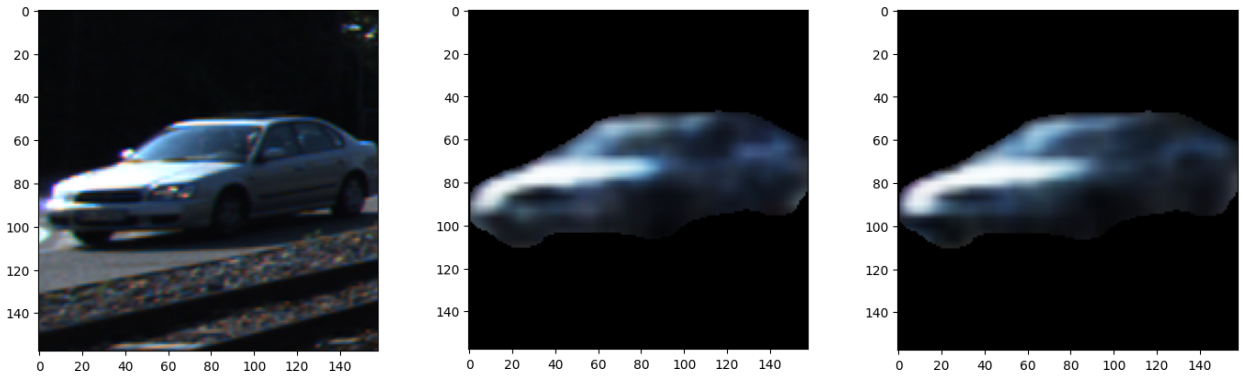
where $E_\phi(x)$ is the encoding result of x and $D_\eta(E_\phi(x))$ is the decoding result of $E_\phi(x)$. The probability $p(x|D_\eta(E_\phi(x)))$ represents the likelihood of x given the decoding result $D_\eta(E_\phi(x))$, typically modeled as a Gaussian density conditioned on the mean $D_\eta(E_\phi(x))$.

The latent code $z = E_\phi(x)$. The training process for GLF involves a two-stage procedure designed to prevent degeneracy in the latent codes, where we initially train the Auto-encoder (AE) parameters η and ϕ using only the first term in Equation 5.2. Subsequently, we proceed to train the normalizing flow parameters θ using Equation 5.2 with η and ϕ held fixed. A single-stage training approach for GLF, on the other hand, would involve simultaneously training η, ϕ, and θ using Equation 5.2. However, this simultaneous training is contingent upon preventing gradients of $\log p(x|D_\eta(E_\phi(x)))$ from influencing the gradients on θ.

In comparison to the ELBO presented in Equation 5.1, the GLF objective outlined in Equation 5.2 lacks the final entropy term, thereby circumventing over-regularization. Experimental assessments, gauged by the Fréchet Inception Distance (FID) Heusel et al. [2017], indicate that GLF outperforms the VAE with a flow prior significantly.

An illustrative example is presented in Figure 5.1, where images (b) and (c) depict comparisons of the reconstructions of the image context (a) by VAE with flow prior and GLF. The training procedures for both models are detailed in Sections 4.7 and 5.4. Notably, in images (b) and (c), only the reconstructions within the predicted object mask are displayed. Within this delineated region, image (b) exhibits a squared error loss of 184.7, whereas image (c) has a squared error loss of 159.1 in comparison to the corresponding region in image (a). This observation aligns with the notion that the GLF model possesses a superior capacity to reconstruct the appearance of objects.

The transition from the VAE with a flow prior to the Generative Latent Flow (GLF) necessitates the establishment of a new probabilistic framework. Specifically, we depart from the utilization of a variational approximation of the marginal likelihood as described in Section 4.2, opting instead for Maximum a Posteriori Probability (MAP) estimation explained in Section 5.2. This also entails the development of algorithms tailored explicitly for GLF. The ensuing discussions in this chapter will expound upon these aspects.



(a) (b) (c)

Figure 5.1: Single Reconstructions. (a) - image context, (b) - single reconstruction by VAE with flow prior, (c) - single reconstruction by GLF.

5.2 Probabilistic Framework

Similar to Chapter 4, this section initiates the establishment of our probabilistic framework before introducing our methodology in Section 5.3. The foundational problem setups remain consistent: we address real-life images characterized by diverse colored backgrounds, potential clutters, and the presence of irrelevant objects. Employing an instance segmentation algorithm such as Mask R-CNN He et al. [2017] results in the detection of objects for all object classes and their associated predicted masks. A catch-all "background" class is defined to encompass elements not included in any object classes. By combining the occlusion scores defined in Section 4.3, we derive detections and their corresponding attributes $\{\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)\}_{i=1}^{N}$ in the same manner as detailed in Chapter 4. Our analysis is confined to I, representing the original image I^* constrained to the union of predicted object masks $\cup_{i=1}^{N} mask_i$.

For each detection, one of three operations is executed: "selection", "discarding", and "designation as background". The sets S, D, and B denote the indices of detections that undergo these respective operations. Assuming a uniform prior $p(S, D, B, \{\mathbf{det}_i\}_{i=1}^{N})$ on S,

D, B and $\{\mathbf{det}_i\}_{i=1}^N$, the posterior probability

$$\begin{aligned}
&p(\{z_i\}_{i\in S\cup B}, S, D, B | I, \{\mathbf{det}_i\}_{i=1}^N) \\
=&p(I, \{z_i\}_{i\in S\cup B}, S, D, B, \{\mathbf{det}_i\}_{i=1}^N)/p(I, \{\mathbf{det}_i\}_{i=1}^N) \\
=&p(S, D, B, \{\mathbf{det}_i\}_{i=1}^N) \times p(\{z_i\}_{i\in S\cup B} | S, D, B, \{\mathbf{det}_i\}_{i=1}^N) \\
&\times p(I | \{z_i\}_{i\in S\cup B}, S, D, B, \{\mathbf{det}_i\}_{i=1}^N)/p(I, \{\mathbf{det}_i\}_{i=1}^N) \\
\propto&p(\{z_i\}_{i\in S\cup B} | S, D, B, \{\mathbf{det}_i\}_{i=1}^N) \times p(I | \{z_i\}_{i\in S\cup B}, S, D, B, \{\mathbf{det}_i\}_{i=1}^N),
\end{aligned} \tag{5.3}$$

where the latent code z_i dictates the appearance of the object within the object mask in detection $\mathbf{det}_i$ conditioned on its class label. The density $p(I | \{z_i\}_{i\in S\cup B}, S, D, B, \{\mathbf{det}_i\}_{i=1}^N)$ is assumed to follow a Gaussian distribution with isotropic variance and a mean determined by $\{z_i\}_{i\in S\cup B}$, S, D, B and $\{\mathbf{det}_i\}_{i=1}^N$. Similar to the approach in Section 4.2, for pixels in I not accounted for by objects in $S \cup B$, we attribute a value of $(0, 0, 0)$ to explain them. This methodology ensures that $\{z_i\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N$ determines the interpretation of I, therefore

$$p(I | \{z_i\}_{i\in S\cup B}, S, D, B, \{\mathbf{det}_i\}_{i=1}^N) = p(I | \{z_i\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N). \tag{5.4}$$

Conditioned on $S, D, B, \{\mathbf{det}_i\}_{i=1}^N$, independence between $\{z_i\}_{i\in S\cup B}$ is assumed, and

$$p(\{z_i\}_{i\in S\cup B} | S, D, B, \{\mathbf{det}_i\}_{i=1}^N) = \prod_{i\in S} p(z_i | cls_i) \times \prod_{i\in B} p(z_i | bg), \tag{5.5}$$

where $p(z_i | cls_i)$ and $p(z_i | bg)$ are assessed by the Deep Generative Model associated with the class label cls_i and the "background" class, respectively. In the case of a discarded detection, where its content is assumed not to exist, we do not evaluate its probability. If the chosen Deep Generative Model is a traditional Variational Autoencoder (VAE), then $p(z_i | cls_i)$ and $p(z_i | bg)$ are the densities of the Gaussian prior evaluated at z_i. However, given that our

Deep Generative Model is GLF in this chapter, $p(z|cls_i)$ should be calculated as

$$p(z|cls_i) = p_{\mathcal{E}}(f_{\theta_{cls_i}}(z)) \times |det\left(\frac{\partial f_{\theta_{cls_i}}(z)}{\partial z}\right)|,$$

where the parameterization of the normalizing flow model trained on class cls_i is denoted by θ_{cls_i}, and $p_{\mathcal{E}}(\cdot)$ represents the density of standard Gaussian noise. The probability $p(z_i|bg)$ can be computed in a similar manner.

With the set $\{\mathbf{det}_i\}_{i=1}^N$ provided, the objective of this chapter is to identify an optimal set of S, D, B that enhances panoptic segmentation. The determination of S, D, B involves an approximate maximization of the posterior probability $p(\{z_i\}_{i\in S\cup B}, S, D, B|I, \{\mathbf{det}_i\}_{i=1}^N)$ as discussed in Equation 5.3. Considering Equation 5.5, $\{z_i\}_{i\in S\cup B}$ given S, B and $\{\mathbf{det}_i\}_{i=1}^N$ requires determination. However, the computational expense associated with optimizing $\{z_i\}_{i\in S\cup B}$ to identify the set that maximizes the posterior is prohibitive. This limitation motivates us to predict the optimal $\{z_i\}_{i\in S\cup B}$ using the encoder of our Deep Generative Model. Taking GLF as an example, for any $i \in S$, we employ

$$\hat{z_i} = E_{\phi_{cls_i}}(x_i),$$

where x_i corresponds to the "image context", as defined in Section 4.2, and $E_{\phi_{cls_i}}(x_i)$ represents the encoding outcome generated by the encoder of the GLF specifically trained for class cls_i. Likewise, for $i \in B$, we utilize $\hat{z_i} = E_{\phi_{bg}}(x_i)$, denoting the encoding result produced by the GLF trained on background pieces. Substituting $\{\hat{z_i}\}_{i\in S\cup B}$ into Equations 5.3, 5.4, and 5.5, the objective becomes finding S, D, B that maximizes

$$p(I|\{\hat{z_i}\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N) \times \prod_{i\in S} p(\hat{z_i}|cls_i) \times \prod_{i\in B} p(\hat{z_i}|bg).$$

Following a similar approach to the DSAMWR presented in Section 4.5, we consolidate

the appearances of objects determined by $\{\hat{z}_i\}_{i\in S\cup B}$ onto a *Canvas*, with additional details provided in Section 5.3. We make the assumption of an isotropic Gaussian density

$$\log p(I|\{\hat{z}_i\}_{i\in S\cup B}, S, B, \{\mathbf{det}_i\}_{i=1}^N) = -\frac{|I|}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\|I - Canvas\|_{vec,2}^2, \tag{5.6}$$

where the quantity $|I|$ represents the number of pixels in I, and σ^2 is a pre-defined pixel-level variance. Utilizing the equation above, we can alternatively frame our problem as identifying S, D, B that minimizes the defined loss, expressed as

$$\begin{aligned} L =& \|I - Canvas\|_{vec,2}^2 - 2\sigma^2 \sum_{i\in S} \log p(\hat{z}_i|cls_i) - 2\sigma^2 \sum_{i\in B} \log p(\hat{z}_i|bg) \\ =& \|I - Canvas\|_{vec,2}^2 - 2\sigma^2 \sum_{i\in S} (\log p_{\mathcal{E}}(f_{\theta_{cls_i}}(\hat{z}_i)) + \log|\, det \left(\frac{\partial f_{\theta_{cls_i}}(z)}{\partial z}\bigg|_{z=\hat{z}_i} \right)|) \\ & - 2\sigma^2 \sum_{i\in B} (\log p_{\mathcal{E}}(f_{\theta_{bg}}(\hat{z}_i)) + \log|\, det \left(\frac{\partial f_{\theta_{bg}}(z)}{\partial z}\bigg|_{z=\hat{z}_i} \right)|). \end{aligned} \tag{5.7}$$

Given that our underlying concept behind this probabilistic framework is to maximize the posterior probability $p(\{z_i\}_{i\in S\cup B}, S, D, B|I, \{\mathbf{det}_i\}_{i=1}^N)$, we henceforth refer to our method in this chapter as Maximizing the Posterior for Panoptic Segmentation.

5.3 Method

The methodology pertaining to this chapter closely aligns with the principles underlying DSAMSR, DSAMWR, and DSAM as detailed in Sections 4.4, 4.5, and 4.6. The distinction between the methodologies elucidated in Chapter 4 and the current chapter lies in their respective probabilistic frameworks. In Chapter 4, a variational approximation of the log marginal likelihood is employed, whereas in this chapter, emphasis is placed on loosely maximizing a posterior density.

Concomitant with the variations in probabilistic frameworks, Chapter 4 employs the

Variational Autoencoder (VAE) with a flow prior Huang et al. [2017] for modeling object distributions. Conversely, the present chapter adopts the Generative Latent Flow (GLF) approach Xiao et al. [2019]. Consequently, disparities arise in the reconstruction of image contexts and the formulation of associated losses. The loss function in this chapter, articulated in Equation 5.7, diverges from the one presented in Chapter 4 (Equation 4.10), notably by omitting the entropy term.

This section expounds upon the methodology tailored for the current probabilistic framework, designed to accommodate the intricacies of the new generative model and the associated loss function.

Algorithm 7: Single Reconstruction Algorithm for MPPS (MPPSSR)

Input: Image context I_i, the detection $\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)$, GLF parameters η, ϕ, θ, GLF training image size $(d, d, 3)$ and N_z, type of operation 'S' or 'B'

$I_i \leftarrow$ resize I_i to $(d, d, 3)$;

if *operation = 'S'* **then**
| $c = cls_i$

else
| $c = bg$

end

$\hat{z_i} \leftarrow E_{\phi_c}(I_i)$, the encoding result of I_i by the encoder ϕ_c ;

$R_i \leftarrow D_{\eta_c}(\hat{z_i})$, the decoding result of latent code $\hat{z_i}$ by the decoder η_c ;

$R_i \leftarrow$ resize R_i to the original image context size ;

$latentLL_i \leftarrow \log p_{\mathcal{E}}(f_{\theta_c}(\hat{z_i}))$;

$logD_i \leftarrow \log |\, det \left(\frac{\partial f_{\theta_c}(z)}{\partial z} \Big|_{z=\hat{z_i}} \right) |$;

Output: Single reconstruction R_i, and $latentLL_i$, $logD_i$, $mask_i$

In this chapter, a Single Reconstruction Algorithm for Maximizing the Posterior for Panoptic Segmentation (MPPSSR) has been devised. This algorithm, outlined in detail in

Algorithm 7, is responsible for computing both the reconstructions of image contexts and the likelihood of latent codes. MPPSSR operates in a manner akin to the Single Reconstruction Algorithm for DSAM (DSAMSR) elucidated in Section 4.4, where both algorithms perform the reconstruction of image contexts and evaluate the log likelihood using the flow model.

Notably, MPPSSR distinguishes itself from DSAMSR in several aspects. In MPPSSR, the sampling of latent code z_i^* from the posterior distribution predicted by the encoder is unnecessary, owing to the deterministic nature of the autoencoder employed by the Generative Latent Flow (GLF). The optimal maximization of the posterior density, as delineated in Equation 5.3, necessitates the optimization of latent codes $\{z_i\}_{i \in S \cup B}$. However, as explicated in Section 5.2, we adopt the encoding output from the GLF encoder as the latent code in order to economize on computational time. Additionally, MPPSSR omits the need to retain μ_i and Γ_i, as it no longer calculates the entropy of the posterior distribution of latent codes. The variables μ_i and Γ_i correspond exactly to the definitions provided in Section 4.4.

Similarly, within this section, we introduce the Whole Reconstruction Algorithm for Maximizing the Posterior for Panoptic Segmentation (MPPSWR) to replace DSAMWR as outlined in Section 4.5. Two notable distinctions exist between MPPSWR and DSAMWR. Firstly, MPPSWR relies on MPPSSR, as opposed to DSAMSR. Secondly, for each detection in $S \cup B$, DSAMWR augments the loss L by

$$-2\sigma^2[\frac{N_z}{2}(1 + \log(2\pi)) + \sum_{t=1}^{N_z} \log \tau_{i,t} + latentLL_i + logD_i]$$

, whereas MPPSWR adds

$$-2\sigma^2(latentLL_i + logD_i)$$

to the loss L. The discrepancy

$$\frac{N_z}{2}(1 + \log(2\pi)) + \sum_{t=1}^{N_z} \log \tau_{i,t} = -\int p_{\mathcal{N}(\mu_i,\Gamma_i)}(x) \log p_{\mathcal{N}(\mu_i,\Gamma_i)}(x) dx$$

represents the entropy of $\mathcal{N}(\mu_i, \Gamma_i)$. Despite the close resemblance between MPPSWR and DSAMWR, we opt to explicate the specifics of MPPSWR in Algorithm 8 to prevent any potential confusion.

Concluding this section, we introduce a greedy algorithm analogous to DSAM, denoted as the Maximizing the Posterior for Panoptic Segmentation Algorithm (MPPS). The details of this greedy algorithm are elucidated in Algorithm 9. Both the previously mentioned MPPSSR and MPPSWR are specifically crafted to execute subtasks within the broader framework of MPPS. The operational procedure of MPPS closely mirrors that of DSAM, with the key distinction being that MPPS relies on MPPSWR instead of DSAMWR. Notably, MPPS exhibits a notable computational advantage, being approximately 20% faster than DSAM, attributed to its elimination of the need for latent code sampling.

5.4 Experiments

In this section, we replicate the experiments detailed in Section 4.7, with the modification that we substitute MPPS and related methods in place of DSAM. Although the Generative Latent Flow (GLF) model lacks a variational evidence lower bound (ELBO), we illustrate the decomposition of its objective in Table 5.1. The objective of GLF is presented in Equation 5.2, and we denote the three terms in the equation, from left to right, as 'recon_ll,' 'gauss_ll,' and 'log_jacob.'

Given the transition from the VAE with a flow prior in the preceding chapter to GLF in the current chapter, it becomes imperative to train GLFs for various object classes. We adopt a partially shared model architecture akin to the one illustrated in Figure 4.8. However, in this case, the posterior distribution $q_\phi(z|x)$ in Figure 4.8 degenerates to a single latent code within the latent space. The GLF models are collectively trained with a latent space dimension set at 64, and the number of flow blocks in their normalizing flows is established at one. The determination of the number of flow blocks is guided by the MPPS performance on

Algorithm 8: Whole Reconstruction Algorithm for MPPS (MPPSWR)

Input: The original image I^*, the image constrained within the union of all predicted object masks I, detection results $\{\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)\}_{i=1}^{N}$, the sets S, B, reconstructions hashmap $ReconDict$, assumed variance $\sigma^2 > 0$, latent dimension N_z

Sort $\{\mathbf{det}_i\}_{i \in S \cup B}$ according to occ_i from high to low ;

$Canvas \leftarrow zeros(H, W, 3)$; `/* (H, W, 3) is the image size */`

$L \leftarrow 0$; `/* loss defined in equation 4.10 */`

for $i \in S \cup B$ **do**

  if $i \in S$ **then**

    operation = 'S'

  else

    operation = 'B'

  end

  if $ReconDict[i, operation]$ *doesn't exist* **then**

    $l \leftarrow$ maximum between the height and width of bb_i ;

    $bb_i^* \leftarrow$ the $l \times l$ square centered at the center of bb_i ;

    if bb_i^* *isn't completely within the image borders* **then**

      $ReconDict[i, operation] \leftarrow (I^*[bb_i], 0, 0)$; `/* I*[bb_i] is obtained by cropping the image at bounding box bb_i */`

    else

      $I_i \leftarrow I^*[bb_i^*]$;

      $(R_i, latentLL_i, logD_i) \leftarrow MPPSSR(I_i, \mathbf{det}_i, operation)$;

      $R_i \leftarrow$ crop a box of size bb_i at the center of R_i ;

      $ReconDict[i, operation] \leftarrow (R_i, latentLL_i, logD_i)$;

    end

  end

  $(R_i, latentLL_i, logD_i) \leftarrow ReconDict[i, operation]$;

  $L \leftarrow L - 2\sigma^2(latentLL_i + logD_i)$;

  Fill in the blank pixels in $Canvas[mask_i]$ by the corresponding values in R_i ;

  `/* "blank pixel" is a pixel with value` $(0, 0, 0)$ `*/`

end

$L \leftarrow L + \|I - Canvas\|_{vec,2}^2$;

Output: Loss L, reconstructions hashmap $ReconDict$

Algorithm 9: Maximizing the Posterior for Panoptic Segmentation Algorithm (MPPS)

Input: Detection results $\{\mathbf{det}_i = (score_i, occ_i, bb_i, mask_i, cls_i)\}_{i=1}^{N}$ sorted by $score_i$ from high to low, assumed variance $\sigma^2 > 0$, latent dimension N_z

$S_0, D_0, B_0 \leftarrow \emptyset, \emptyset, \emptyset$;

$ReconDict \leftarrow \{\}$;

$L_0 \leftarrow \infty$;

for $i = 1$ **to** N **do**

  $(L_{i,1}, ReconDict) \leftarrow MPPSWR(S_{i-1} \cup \{i\}, D_{i-1}, B_{i-1}, ReconDict)$;

  $(L_{i,2}, ReconDict) \leftarrow MPPSWR(S_{i-1}, D_{i-1}, B_{i-1} \cup \{i\}, ReconDict)$;

  $L_i \leftarrow \min(L_{i-1}, L_{i,1}, L_{i,2})$;

  if $L_i = L_{i-1}$ **then**

    $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1} \cup \{i\}, B_{i-1}$;

  else if $L_i = L_{i,1}$ **then**

    $S_i, D_i, B_i \leftarrow S_{i-1} \cup \{i\}, D_{i-1}, B_{i-1}$;

  else

    $S_i, D_i, B_i \leftarrow S_{i-1}, D_{i-1}, B_{i-1} \cup \{i\}$;

  end

end

Output: S_N, D_N, B_N

a small validation set extracted from the KINS training set. GLF employs a reduced number of flow blocks due to the comparatively less chaotic nature of its latent codes, making them more amenable to learning by the flow model. The avoidance of an excessively potent flow model is crucial to mitigate potential overfitting issues. Consequently, the autoencoder of GLF encompasses 38,210,243 parameters, while the normalizing flow comprises 333,952 parameters. The batch size, optimizer, and number of epochs remain consistent with the specifications outlined in Section 4.7.

Table 5.1: GLF - Three components of its objective

	cyclist	**pedestrian**	**car**	**tram**	**truck**	**van**	**misc**
recon_ll	-19292.6	-18042.7	-18848.8	-14421.1	-16960.9	-28933.6	-8479.1
gauss_ll	-243.1	-162.0	-100.0	-311.5	-327.0	-265.0	-110.4
log_jacob	-7.3	-27.1	-49.5	-3.9	-3.2	-15.1	-32.8
objective	-19543.1	-18231.9	-18998.2	-14736.5	-17291.2	-29213.8	-8622.3

As depicted in Table 5.1, the GLF objective is predominantly influenced by the term 'recon_ll.' Notably, the 'gauss_ll' for the 'car' category attains the highest value, a phenomenon attributed to the widespread prevalence of the 'car' object class. A comparative analysis between Table 4.2 and Table 5.1 reveals that the 'recon_ll' of GLF exhibits superior performance, signifying enhanced reconstruction capabilities. This superiority can be attributed to the less constrained nature of GLF, as opposed to the VAE with a flow prior, which is subject to regulation by the entropy of the posterior distribution and the flow likelihood. In contrast, GLF lacks these two regulatory factors.

As a parallel to Table 4.3, we present the GLF objective when subjected to potentially mis-specified labels in Table 5.1. In comparison to Table 4.3, the values in Table 5.1 are higher. It additionally demonstrates that the GLF trained on the 'car' category attains comparatively high objectives. The table suggests that while discerning between various

object classes may pose challenges for the trained GLFs, they do exhibit proficiency in distinguishing between background images and object images.

Table 5.2: GLF objective under mis-specified labels

	cyclist	**pedestrian**	**car**	**van**	**misc**	**background**
cyclist	-64760.1	-64905.9	-49991.3	-76837.6	-29711.	-96769.7
pedestrian	-64697.5	-64836.1	-49879.9	-76735.	-29604.3	-97168.2
car	**-64615.6**	**-64796.9**	**-49856.4**	-76649.5	-29563.6	-96952.1
van	-64745.3	-64893.	-49957.8	-76785.6	-29648.3	-95097.4
misc	-64673.6	-64807.1	-49863.3	**-76631.8**	**-29552.**	-97364.5
background	-71169.	-86457.4	-66696.4	-82606.8	-51913.3	**-24074.**

Finally, we apply our MPPS to the panoptic segmentation task and compare its performance with that of the Detection Selection Algorithm with Mask (DSAM) and baseline NMS-like procedure described in Kirillov et al. [2019b]. The Panoptic Quality (PQ) scores for these approaches are presented in Tables 5.3 and 5.4. The panoptic segmentations in Table 5.3 are derived by arranging the selected detections in descending order based on their objectness scores, while in Table 5.4, the ordering is based on occlusion scores from high to low. The row designated as "Baseline" pertains to the Panoptic Quality (PQ) scores attained through the application of the NMS-like procedure on the output of Mask R-CNN He et al. [2017]. These results align with those presented in Section 4.7. The row labeled "with DSAM" corresponds to the PQ scores achieved by DSAM, same as the results reported in Section 4.7. Conversely, the row labeled "with MPPS" encompasses the PQ scores obtained using MPPS for post-processing. The results indicate a slight superiority of MPPS over DSAM across all eight scenarios.

Table 5.3: MPPS PQ with Objectness Scores Ordering

PQ	**all objects**	≥ 100 **pixels**	≥ 400 **pixels**	≥ 900 **pixels**
Baseline	0.411198	0.444078	0.582399	0.654408
with DSAM	0.479953	0.506556	0.607400	0.670859
with MPPS	0.483950	0.510529	0.610217	0.672629

Table 5.4: MPPS PQ with Occlusion Ordering

PQ	**all objects**	≥ 100 **pixels**	≥ 400 **pixels**	≥ 900 **pixels**
Baseline	0.388117	0.420399	0.560606	0.636605
with DSAM	0.471164	0.497567	0.597729	0.662387
with MPPS	0.474265	0.500368	0.599472	0.663313

5.5 Conclusion and Future Work

This chapter introduces the Maximizing the Posterior for Panoptic Segmentation Algorithm (MPPS), which is showed to deliver slightly superior performance compared to the Detection Selection Algorithm with Mask (DSAM) expounded upon in Chapter 4. Both MPPS and DSAM are oriented towards enhancing panoptic segmentation quality. They share a common objective of selecting a subset of detections generated by the instance segmentation model, with only these chosen detections being utilized in panoptic segmentation. The selection process for both algorithms involves opting for one of three operations: "selection", "discarding", or "designation as background". Their decisions regarding the choice of operation are guided by likelihood comparisons.

Nonetheless, MPPS deviates significantly from DSAM. The likelihood framework of DSAM is derived from the maximization of $\log p(I, S, D, B, \{\mathbf{det}_i\}_{i=1}^N)$, whereas that of MPPS is constructed based on maximizing $p(\{z_i\}_{i\in S\cup B}, S, D, B|I, \{\mathbf{det}_i\}_{i=1}^N)$, a posterior probability. The former necessitates an integration of latent codes z_i, while the latter incorpo-

rates latent codes as a part of its maximization, albeit both employ certain approximations to facilitate computation. Discrepancies in their probabilistic frameworks necessitate the utilization of distinct Deep Generative Models. DSAM employs the VAE with flow prior, whereas MPPS opts for GLF. GLF, characterized by less regulation, demonstrates superior reconstruction performance. This disparity is evident in our experiments detailed in Section 5.4, where MPPS exhibits slightly superior results compared to DSAM.

A future direction that may worth exploring involves the consideration of Latent Diffusion Models (LDMs) Rombach et al. [2022] as a substitute for GLF within the MPPS framework. LDM and GLF share similarities, employing a deterministic autoencoder to map from the image space to a latent space of lower dimensions. While GLF incorporates a normalizing flow model in the latent space, LDM employs Diffusion Models (DMs), which have demonstrated exceptional capabilities in image synthesis and have attained state-of-the-art performance across various tasks. By integrating a more potent generative model, there is potential to enhance image reconstruction and likelihood estimation abilities, consequently improving the overall performance of MPPS.

An intriguing avenue for enhancing MPPS involves the conversion of images to a convolutional feature space using pre-trained convolutional backbones. As exemplified in GLF Xiao et al. [2019], training with perceptual loss Johnson et al. [2016] has yielded substantial improvements in FID scores. The implementation of perceptual loss involves transforming images into a feature space through convolutional networks and computing the loss within that space. A similar strategy can be applied to MPPS, wherein MPPS operates on the transformed feature space instead of the original image space. This adaptation enables MPPS to harness the capabilities of powerful pre-trained backbone models such as VGG Simonyan and Zisserman [2014], thereby mitigating visual clutter.

REFERENCES

Yali Amit. *2D Object Detection and Recognition: Models, Algorithms, and Networks*. The MIT Press, 08 2002. ISBN 9780262267090. doi:10.7551/mitpress/1006.001.0001. URL https://doi.org/10.7551/mitpress/1006.001.0001.

Yali Amit and Donald Geman. A computational model for visual selection. *Neural Computation*, 11:1691–1715, 1999. URL https://api.semanticscholar.org/CorpusID:7784637.

Yali Amit and Alain Trouvé. Pop: Patchwork of parts models for object recognition. *International Journal of Computer Vision*, 75(2):267–282, 2007.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein gan, 2017. URL https://arxiv.org/abs/1701.07875.

Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S. Davis. Soft-nms – improving object detection with one line of code, 2017a.

Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S Davis. Soft-nms–improving object detection with one line of code. In *Proceedings of the IEEE international conference on computer vision*, pages 5561–5569, 2017b.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders, 2015. URL https://arxiv.org/abs/1509.00519.

Chia-Yuan Chang, Shuo-En Chang, Pei-Yung Hsiao, and Li-Chen Fu. *EPSNet: Efficient Panoptic Segmentation Network with Cross-layer Attention Fusion*, page 689–705. Springer International Publishing, 2021. ISBN 9783030695255. doi:10.1007/978-3-030-69525-5_41. URL http://dx.doi.org/10.1007/978-3-030-69525-5_41.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016.

Xinlei Chen, Ross Girshick, Kaiming He, and Piotr Dollár. Tensormask: A foundation for dense object segmentation, 2019.

Yuelong Chuang, Shiqing Zhang, and Xiaoming Zhao. Deep learning-based panoptic segmentation: Recent advances and perspectives. *IET Image Processing*, 2023.

Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation, 2014. URL https://arxiv.org/abs/1410.8516.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2016. URL https://arxiv.org/abs/1605.08803.

Yilun Du and Igor Mordatch. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

Kiana Ehsani, Roozbeh Mottaghi, and Ali Farhadi. Segan: Segmenting and generating the invisible. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6144–6153, 2018.

David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network, 2014.

Omar Elharrouss, Somaya Al-Maadeed, Nandhini Subramanian, Najmath Ottakath, Noor Almaadeed, and Yassine Himeur. Panoptic segmentation: A review, 2021.

M Everingham, L Van Gool, CKI Williams, J Winn, and A Zisserman. The pascal visual object classes challenge 2007 (voc2007) results http://www. pascal-network. org/challenges. In *VOC/voc2007/workshop/index. html*, 2007.

Angzhi Fan, Benjamin Ticknor, and Yali Amit. Detection selection algorithm: A likelihood based optimization method to perform post processing for object detection, 2023.

Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2009.

Panhe Feng, Qi She, Lei Zhu, Jiaxin Li, Lin Zhang, Zijian Feng, Changhu Wang, Chunpeng Li, Xuejing Kang, and Anlong Ming. Mt-orl: Multi-task occlusion relationship learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9364–9373, 2021.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

Ross Girshick. Fast r-cnn, 2015a.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.

Ross B. Girshick. Fast R-CNN. *CoRR*, abs/1504.08083, 2015b. URL `http://arxiv.org/abs/1504.08083`.

Meiling Gong, Dong Wang, Xiaoxia Zhao, Huimin Guo, Donghao Luo, and Min Song. A review of non-maximum suppression algorithms for deep learning target detection. In *Seventh Symposium on Novel Photoelectronic Detection Technology and Applications*, volume 11763, pages 821–828. SPIE, 2021.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014. URL `https://arxiv.org/abs/1406.2661`.

George D. Greenwade. The Comprehensive Tex Archive Network (CTAN). *TUGBoat*, 14 (3):342–351, 1993.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *International conference on machine learning*, pages 1462–1471. PMLR, 2015.

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.

Yihui He, Xiangyu Zhang, Marios Savvides, and Kris Kitani. Softer-nms: Rethinking bounding box regression for accurate object detection. *arXiv preprint arXiv:1809.08545*, 2(3): 69–80, 2018.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Matthew D Hoffman and Matthew J Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. In *Workshop in Advances in Approximate Bayesian Inference, NIPS*, volume 1, 2016.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Chin-Wei Huang, Ahmed Touati, Laurent Dinh, Michal Drozdzal, Mohammad Havaei, Laurent Charlin, and Aaron Courville. Learnable explicit density for continuous latent space and variational inference, 2017.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, page 2017–2025, Cambridge, MA, USA, 2015. MIT Press.

Borui Jiang, Ruixuan Luo, Jiayuan Mao, Tete Xiao, and Yuning Jiang. Acquisition of localization confidence for accurate object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–799, 2018.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016.

Doyeon Kim, Woonghyun Ka, Pyungwhan Ahn, Donggyu Joo, Sehwan Chun, and Junmo Kim. Global-local path networks for monocular depth estimation with vertical cutdepth, 2022.

Diederik P. Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions, 2018. URL https://arxiv.org/abs/1807.03039.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Durk P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. *Advances in neural information processing systems*, 29, 2016.

Alexander Kirillov, Ross Girshick, Kaiming He, and Piotr Dollár. Panoptic feature pyramid networks, 2019a.

Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9404–9413, 2019b.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 29–37. JMLR Workshop and Conference Proceedings, 2011.

Katrin Lasinger, René Ranftl, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *CoRR*, abs/1907.01341, 2019. URL http://arxiv.org/abs/1907.01341.

Justin Lazarow, Kwonjoon Lee, Kunyu Shi, and Zhuowen Tu. Learning instance occlusion for panoptic segmentation, 2020.

Xinye Li and Ding Chen. A survey on deep learning-based panoptic segmentation. *Digit. Signal Process.*, 120(C), jan 2022. ISSN 1051-2004. doi:10.1016/j.dsp.2021.103283. URL https://doi.org/10.1016/j.dsp.2021.103283.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

Huanyu Liu, Chao Peng, Changqian Yu, Jingbo Wang, Xu Liu, Gang Yu, and Wei Jiang. An end-to-end network for panoptic segmentation, 2019a.

Songtao Liu, Di Huang, and Yunhong Wang. Adaptive nms: Refining pedestrian detection in a crowd. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6459–6468, 2019b.

Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015. URL `http://arxiv.org/abs/1512.02325`.

Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015.

D Matthew Zeiler and Fergus Rob. Visualizing and understanding convolutional neural networks. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings*. ECCV, 2014.

Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3523–3542, 2021.

Yue Ming, Xuyang Meng, Chunxiao Fan, and Hui Yu. Deep learning for monocular depth estimation: A review. *Neurocomputing*, 438:14–33, 2021.

Rohit Mohan and Abhinav Valada. Efficientps: Efficient panoptic segmentation, 2021.

Alexander Neubeck and Luc Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation, 2015.

Steven T Piantadosi. Zipf's word frequency law in natural language: A critical review and future directions. *Psychonomic bulletin & review*, 21:1112–1130, 2014.

Lorenzo Porzi, Samuel Rota Bulò, Aleksander Colovic, and Peter Kontschieder. Seamless scene segmentation, 2019.

Lu Qi, Li Jiang, Shu Liu, Xiaoyong Shen, and Jiaya Jia. Amodal instance segmentation with kins dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3014–3023, 2019.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015. URL `https://arxiv.org/abs/1511.06434`.

Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015. URL `http://arxiv.org/abs/1506.02640`.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 91–99, Cambridge, MA, USA, 2015. MIT Press.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models, 2014. URL `https://arxiv.org/abs/1401.4082`.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

Claude E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948. doi:10.1002/j.1538-7305.1948.tb01338.x.

Jiajun Shen and Yali Amit. Deformable classifiers. *Quarterly of Applied Mathematics*, 77 (2):207–226, 2019.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Yanan Song, Quan-Ke Pan, Liang Gao, and Biao Zhang. Improved non-maximum suppression for object detection using harmony search algorithm. *Applied Soft Computing*, 81: 105478, 2019.

Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

Esteban G Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104: 154–171, 2013.

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *International Conference on Machine Learning*, pages 467–475. PMLR, 2014.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, Alex Graves, et al. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

C Van Rijsbergen. Information retrieval: theory and practice. In *Proceedings of the Joint IBM/University of Newcastle upon Tyne Seminar on Data Base Systems*, volume 79, 1979.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Haochen Wang, Ruotian Luo, Michael Maire, and Greg Shakhnarovich. Pixel consensus voting for panoptic segmentation, 2020.

Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011.

Zhisheng Xiao, Qing Yan, and Yali Amit. Generative latent flow, 2019.

Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. Ufogen: You forward once large scale text-to-image generation via diffusion gans, 2023.

Dewei Yi, Jinya Su, and Wen-Hua Chen. Probabilistic faster r-cnn with stochastic region proposing: Towards object detection and recognition in remote sensing imagery. *Neurocomputing*, 459:290–301, 2021.

Xiaoding Yuan, Adam Kortylewski, Yihong Sun, and Alan Yuille. Robust instance segmentation through reasoning about multi-object occlusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11141–11150, 2021.

Ekaterina Zaytseva and Jordi Vitrià. A search based approach to non maximum suppression in face detection. In *2012 19th IEEE International Conference on Image Processing*, pages 1469–1472. IEEE, 2012.

Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 12993–13000, 2020.

Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023.