Towards Understanding Small Objects in Indian Driving Situations

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Umamahesh Janapareddi
2019900058
umamahesh.j@research.iiit.ac.in

Advisor: Dr. Jawahar C.V.

INTERNATIONAL INSTITUTE OF
INFORMATION TECHNOLOGY
HYDERABAD

International Institute of Information Technology Hyderabad
500 032, India

March 2025

International Institute of Information Technology Hyderabad
Hyderabad, India

CERTIFICATE

This is to certify that work presented in this thesis proposal titled ***Towards Understanding Small Objects in Indian Driving Situations*** by *Umamahesh Janpareddi* has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Advisor: Dr. Jawahar C.V.

Abstract

In Indian urban and rural driving scenarios, small objects are pervasive and often crucial for safe navigation. These objects can include pedestrians crossing roads, children playing near streets, cyclists, stray animals, as well as small vehicles like scooters and motorbikes. Additionally, traffic signs, signal lights, potholes, and road markings (such as lane dividers or zebra crossings) are often small in size but essential for driving decisions. In such contexts, missing or inaccurately segmenting these small objects can lead to critical errors in detection, causing accidents or delays in the vehicle's decision-making process. Automated understanding of such objects need detection and segmentation to start with.

Semantic Segmentation is a critical task in computer vision with a wide range of applications. The objective is to partition an image—a collection of pixels—into distinct labeled regions, each corresponding to specific objects or parts of the scene. This process is crucial for scene understanding and enables the localization of objects within the image. Over time, significant progress has been made in semantic segmentation, especially with the advent of deep learning. The advances in this area have revolutionized computer vision, pushing beyond traditional methods and achieving remarkable improvements in performance.

When discussing semantic segmentation, we often focus on datasets, the objects within those datasets, and their corresponding segmentations. While many datasets exist for road scenarios, particularly those representing Western road conditions, there is relatively little research on road conditions specific to India. One notable exception is the Indian Driving Dataset (IDD), a dataset specifically designed for semantic segmentation of Indian road scenarios.

Road and driving datasets typically contain objects of varying sizes within each class label. These objects can be broadly categorized into three types: small, medium, and large. The importance of segmentation is well understood across several domains such as medical imaging, autonomous vehicles, aerial imagery, robotics, surveillance, and industrial automation. However, one of the most challenging problems in segmentation is the segmentation of small objects. Small object segmentation is particularly difficult due to factors such as (i) the limited number of pixels representing small objects, (ii) class imbalance during training, and (iii) the inherent challenges posed by small object representations. These factors hinder the performance of deep learning architectures, making it harder for modern techniques to accurately handle small objects.

This issue becomes evident when evaluating deep learning models, as performance metrics tend to be significantly worse for small objects. In this study, we delve into the challenges of segmenting small objects and explore potential strategies for improving algorithm design for this task. It is widely ac-

knowledged that loss functions play a key role in segmentation performance, alongside the architecture of deep learning models. Indeed, the segmentation of small objects is highly influenced by the choice of loss function during model training.

Various small objects commonly encountered in road scenarios are highlighted in this study, emphasizing their importance in Indian road environments. Objects such as pedestrians, traffic signs, and traffic lights—despite their small size—are crucial for tasks like autonomous navigation and driver assistance systems. The Indian Driving Dataset (IDD) offers a valuable resource, containing a wide range of small objects across different class labels, some of which dominate specific categories. These objects are captured and annotated in real-world, unstructured environments, making this dataset particularly relevant for segmentation tasks in Indian road scenarios. Therefore, our focus in this study is primarily on detection and segmentation of small objects within the IDD dataset and explore the impact of loss functions.

Contents

List of Figures

List of Tables

List of Related Publications

[P1] Umamahesh Janapareddi, C.V. Jawahar **"Semantic Segmentation of Indian Road Scene Images with a Focus on Small Objects"**, in proceedings of *2nd International Conference on Artificial Intelligence, Computing Technologies, Internet of Things (IoT) and Data Analytics* , 2024.

Chapter 1

Introduction

Detection and segmentation of objects in road scenes is crucial for computer vision algorithms involved in tasks such as road surveillance, navigation, intelligent transportation and autonomous driving. Detection and segmentation helps in localizing the objects of interest in a natural image. In road scenes, often small objects are extremely valuable. They could be missed in on a quick scan of the image, but could be costly if you miss to detect such objects (such as pedestrians). Occurrence of small objects in the road scenes needs specific attention. Small objects arise due to small physical objects, or low resolution of the objects. This could also be due to other factors such as occlusion, background clutter, and variations in lighting conditions. All these make the detection and segmentation challenging.

1.1 Object Detection and Semantic Segmentation

In the field of computer vision, Semantic Segmentation and Object Detection are two foundational tasks that help machines interpret and understand visual information. Both tasks are crucial for enabling computers to perform high-level analysis of images and videos, making them essential components in applications such as autonomous driving, medical imaging, surveillance systems, robotics, and many more.

Object detection refers to the task of identifying and localizing objects in an image. Unlike simple image classification, where the goal is to assign a single label to an entire image, object detection involves recognizing individual objects within an image and determining their locations. This is typically achieved by drawing bounding boxes around the detected objects and classifying them into predefined categories, such as people, cars, animals, traffic signs, etc.

The main challenge in object detection lies in not just recognizing the objects but also localizing them accurately. This means that for each object, we need to predict both the class label (e.g., "dog," "car," or "person") and the bounding box coordinates that specify its position within the image (usually represented by the top-left and bottom-right corners).

Object detection algorithms have evolved significantly over the years, with the advent of deep learning leading to major breakthroughs. Early methods used handcrafted features and sliding windows to search for objects in images, but this approach was computationally expensive and often inaccurate. The introduction of Convolutional Neural Networks (CNNs) and more advanced architectures such as Region Proposal Networks (RPNs) and Faster R-CNN has drastically improved the efficiency and accuracy of object detection systems.

One of the most popular object detection models today is YOLO (You Only Look Once), which divides the image into a grid and predicts bounding boxes and class labels for each grid cell in a single forward pass. Another notable method is SSD (Single Shot MultiBox Detector), which, like YOLO, can predict multiple bounding boxes and object classes in one step but with a focus on multi-scale features.

Despite significant progress, challenges remain, particularly when it comes to detecting small objects, handling occlusions (where objects overlap or are partially hidden), and achieving high accuracy in real-time applications.

Semantic segmentation is a more detailed and pixel-level approach to understanding an image. The goal of semantic segmentation is to assign a class label to each pixel in an image. This means that instead of just drawing a bounding box around an object, the system segments the image into regions where each region corresponds to a specific class of object.

For example, in a street scene, semantic segmentation might label all pixels corresponding to cars as "car," all pixels corresponding to pedestrians as "pedestrian," and so on. This pixel-level understanding provides a much more precise view of the image and is important for tasks like scene understanding, autonomous driving, and medical image analysis, where exact boundaries of objects matter.

The key challenge in semantic segmentation is to classify each pixel correctly while considering the spatial relationships between pixels. Traditional methods, before the rise of deep learning, relied on techniques like thresholding, edge detection, and region growing. However, with the advent of CNNs, semantic segmentation has seen significant improvement.

Today, CNN-based architectures like Fully Convolutional Networks (FCNs), U-Net, and DeepLab are widely used for semantic segmentation. These models are designed to process an image as a whole and predict pixel-wise class labels. FCNs, for example, replace fully connected layers in traditional CNNs with convolutional layers to make the network capable of generating output maps with the same spatial dimensions as the input image.

While CNN-based architectures have greatly advanced the state of semantic segmentation, challenges persist, particularly in handling fine details, small objects, and complex boundaries. For example, semantic segmentation can struggle with accurately segmenting objects with irregular shapes or distinguishing between similar objects (e.g., distinguishing between two types of cars that appear visually similar).

In short, object detection and semantic segmentation are two critical tasks in computer vision that help machines understand and interpret images. Object detection focuses on identifying and localizing objects in an image, while semantic segmentation assigns a class label to every pixel. Both tasks have evolved significantly with the advent of deep learning, and models like YOLO, Faster R-CNN, U-Net, and DeepLab [1]have pushed the boundaries of what is possible in these domains. Despite the significant progress made, challenges remain, particularly in the detection of small objects, handling occlusions, and achieving real-time performance. As computer vision continues to advance, object detection and semantic segmentation will remain at the core of many cutting-edge applications.

1.2 Focus on Small Objects

Small objects, despite their size, have a disproportionate impact on driving safety and efficiency. Detecting and understanding these small objects is crucial for autonomous driving systems and advanced driver assistance systems. From pedestrians and cyclists to traffic signs and debris, small objects present significant challenges but are essential for safe navigation and effective traffic management. As the technology continues to evolve, focusing on improving small object detection will be key to achieving safer, smarter roads for both human and autonomous drivers.

In driving scenarios, the ability to detect and accurately identify small objects plays a crucial role in ensuring safety, efficiency, and reliability of autonomous vehicles (AVs) and advanced driver assistance systems (ADAS). While large objects, such as cars and trucks, are easily detectable and form the primary focus of most driving-related algorithms, small objects—such as pedestrians, traffic signs, debris, potholes, and other obstacles—are equally critical for safe navigation and proper decision-making.

A small object is one that appears tiny within the video or image frame, like a distant para-glider or soccer ball, often due to being tracked from afar. It may also refer to objects that are physically smaller in the real world. As per the MS-COCO metric evaluation, small objects are those with an area of 32×32 pixels or less, a threshold generally accepted for datasets involving common objects [2] in the given image frame size.

To localize the object in the image, we need to segment the pixels. Semantic segmentation is a computer vision task that involves partitioning an image into segments, where each pixel is assigned a label corresponding to a particular object class. Unlike object detection, which provides bounding boxes around objects, semantic segmentation offers a pixel-level understanding of visual content, making it crucial for applications requiring precise localization such as autonomous driving, medical imaging, and scene understanding.

Early methods for semantic segmentation primarily relied on traditional techniques such as region-based algorithms and graph cuts. However, the emergence of deep learning has brought substantial progress to this field. Convolutional Neural Networks (CNNs) have played a pivotal role, with architectures like Fully Convolutional Networks (FCNs) leading the way in end-to-end training for segmentation tasks [3]. Subsequent advancements, including U-Net [4], tailored for biomedical image segmentation, and DeepLab [1], which introduced atrous convolution to capture multi-scale context, have further enhanced the performance and accuracy of semantic segmentation.

The performance of deep learning algorithms depends on (a) the network architecture, (b) the data used for training, and (c) the loss function employed during training. Researchers have highlighted that the choice of loss functions plays a crucial role in semantic segmentation [5] [6]. However, their studies do not specifically address the effects on small objects. While there is research available on deep learning-based small object detection [7], similar work on small object segmentation is notably lacking. This indicates that small object segmentation has received less attention compared to small object detection. Furthermore, small object segmentation has not been explored in the Indian context. To address this gap, we utilize the India Driving Dataset [8] to study segmentation in the Indian setting.

1.3 Small Objects commonly seen in Road Situations

Addressing perception challenges, such as detection and segmentation, on Indian roads is crucial for developing accurate surveillance and road safety solutions, extending beyond autonomous driving. The following are examples of objects frequently observed in road scenes. Many of these objects appear small when viewed from a distance, highlighting the importance of segmenting them effectively.

- Vehicle registration numbers
- Vehicles at a distance
- Pedestrians at a distance
- Sharp turns at a distance
- Zebra crossing at a distance
- Damages to the road (eg: potholes) at a distance
- Depth of water filled potholes and open man-holes at a distance
- Breaking traffic rules at a distance
- Road blockade at a distance
- Small road markings
- Traffic sign boards
- Helmets
- Seat belts
- Small and sharp objects on the road
- Traffic lights at a distance.

The perception of small objects in road scenarios depends on factors such as the camera used and external conditions like scene clutter. Small objects are better understood in the context of individual object instances rather than broader object categories. Key objects in autonomous driving include pedestrians, animals, riders, motorcycles, autorickshaws, cars, trucks, and buses. Accurate detection and segmentation of these objects are essential to prevent them from becoming obstacles or causing accidents. Traffic signs and lights are equally important for safety and communicating regulations.

Despite its importance, small object semantic segmentation has received limited attention in the past. This task is particularly challenging due to (i) the limited number of pixels representing small objects and (ii) class imbalance during the learning process.

1.4 Practical Issues on Roads Due to Small Objects

When a small object is encountered on the road, it may avoid detection, leading to the following issues.

- In autonomous driving, lack of detection may lead to accidents, while delay in detection may affect smoothness in manoeuvrability.
- In traffic density detection, lack of detection causes errors in the frequency of the detected traffic. This would affect the downstream tasks involving density detection.
- Road surveillance systems are affected when objects are not detected due to small size.
- Traffic offenders, when not detected due to smallness in size due to a large distance in detection, affect the performance of the authorities involved.
- Traffic control and regulation becomes a problem due to lack of or delay in detection of small objects.

1.5 Small Objects in Indian Driving Scenes

Cars, pedestrians, and traffic signs are typical examples of small objects extracted in segmentation tasks. However, in the Indian driving scenario, additional objects also fall into the small-object category. A relevant dataset for this context is the Indian Driving Dataset (IDD) [8], a unique dataset that facilitates the detection and segmentation of objects in unstructured environments.

The IDD has gained significant attention from researchers globally, as perception tasks like autonomous driving on Indian roads are considered extremely challenging. Baseline results on this dataset have highlighted the domain gap and the relative complexity of segmentation in the Indian context compared to datasets from Europe (e.g., Cityscapes) and the USA (e.g., BDD).

Typically, objects with fewer than 1000 pixels are classified as small objects. In IDD, small objects include persons, animals, riders, motorcycles, autorickshaws, cars, trucks, buses, traffic signs, traffic lights, poles, and vegetation.

Several datasets have been introduced to address semantic and instance segmentation challenges, such as Pascal VOC [9], MS COCO [10], and SUN [11, 12]. For semantic segmentation with a focus on autonomous navigation, notable datasets include Cityscapes [13], KITTI [14], CamVid [15, 16], Leuven [17], and the Daimler Dataset [18]. ADE20K [19] is a more recent addition targeting general scene parsing tasks. Recently, the Mapillary Vistas dataset [20], emphasizing street view imagery, and the Berkeley Deep Drive Dataset [21], designed for autonomous navigation, have also been released.

IDD bears a closer resemblance to Cityscapes, as it is captured using calibrated cameras that avoid distortions. In contrast, BDD100K relies on dashboard cameras mounted inside vehicles, often resulting in internal glass reflections and distortions from rain. Additionally, a significant portion of BDD100K comprises nighttime images. On the other hand, the Mapillary Vistas dataset includes images captured with diverse cameras, such as smartphones, offering varied perspectives of roads and their surroundings. IDD presents novelty in terms of ambiguous road boundaries (Figure 1.1), diversity of vehicles and pedestrians (Figure 1.2), extensive use of information boards (Figure 1.3), and diversity of ambient conditions (Figure 1.4) in addition to small objects.

Traditional segmentation algorithms often relied on clustering or graph-cut-based methods [22]. In recent years, deep learning-based semantic segmentation algorithms have achieved significant success. These algorithms are trained on a large number of pixel-level labeled images. During the training process, a loss function (or objective function) is employed, and backpropagation, often with suitable modifications like regularization, is used to optimize the model. The unique characteristic of IDD lies in its focus on driving scenarios within less structured environments.It is contended that this better aligns with the requirements of autonomous navigation across vast regions of the world, such as Asia, South America, and Africa.

The detection of small objects [7] is inherently challenging, and the same applies to segmentation. Firstly, distinguishing small objects from generic background clutter is difficult, which poses challenges for standard detectors that rely on ”objectness” due to the significant increase in the number of potential object locations. Secondly, as an image passes through a typical CNN architecture, the activations for small objects shrink with each pooling layer, making their detection harder. Lastly, there is a lack of large publicly available datasets specifically designed for small objects [23].

1.6 This Work

1.6.1 Scope

This work aims to explore and gain a deeper understanding of the critical role that small objects play in road scene understanding, with a particular focus on the unique challenges presented by Indian roads. Small objects in road scenes—such as pedestrians, cyclists, animals, and small traffic signs—are often overlooked by conventional object detection models due to their size and the complexities of their interactions with the environment. In the context of Indian roadways, where traffic is dense, unpredictable, and includes a variety of road users, from rickshaws to two-wheelers to stray animals, detecting these small objects becomes even more challenging. Moreover, the infrastructure on Indian roads is often inconsistent, with poorly maintained signage, irregular road markings, and the absence of structured lanes, all of which make it difficult for both human drivers and autonomous systems to track smaller ob-

Figure 1.1: Ambiguous Road Boundaries: not all datasets distinguish between the road and possible unsafe drivable area on both sides of the road [8].

jects effectively. By focusing on understanding how small objects impact road scene interpretation, this research aims to enhance the effectiveness of autonomous vehicles (AVs) and driver assistance systems (ADAS) operating in these environments. The ultimate goal is to improve safety, reduce accidents, and enable more reliable decision-making in real-time road scenarios, ensuring that small but critical objects are detected and processed correctly in the complex and diverse landscape of Indian road conditions.

In this work, we study the critical role of loss functions in the context of road scene understanding, particularly in the domain of object detection and semantic segmentation for autonomous driving and driver assistance systems. Loss functions play a pivotal role in training deep learning models, as they define how the model's predictions are penalized when they deviate from the ground truth. In road scene understanding tasks, where the goal is to accurately detect and segment objects in complex environments, the choice of loss function can significantly influence the performance of the model. For instance, in detecting small objects, such as pedestrians, cyclists, and traffic signs, traditional loss functions like cross-entropy or mean squared error may not adequately capture the nuances of these objects due to their smaller size and sparse pixel coverage. This could lead to poor detection accuracy or missed small objects in crowded or cluttered environments. We explore how different loss functions, such as focal loss or dice loss, can help address these challenges by placing more emphasis on harder-to-detect small objects or those with high class imbalance. Additionally, we consider how loss functions can be tailored to account for the unique characteristics of road scenes, including occlusions, background clut-

(a)

(b)

Figure 1.2: Diversity of vehicles, pedestrians and animals:(a) (Top left) A herd of buffaloes on the road at dusk and (b) (Top left) Many motorbikes with multiple riders which are less structured [8]

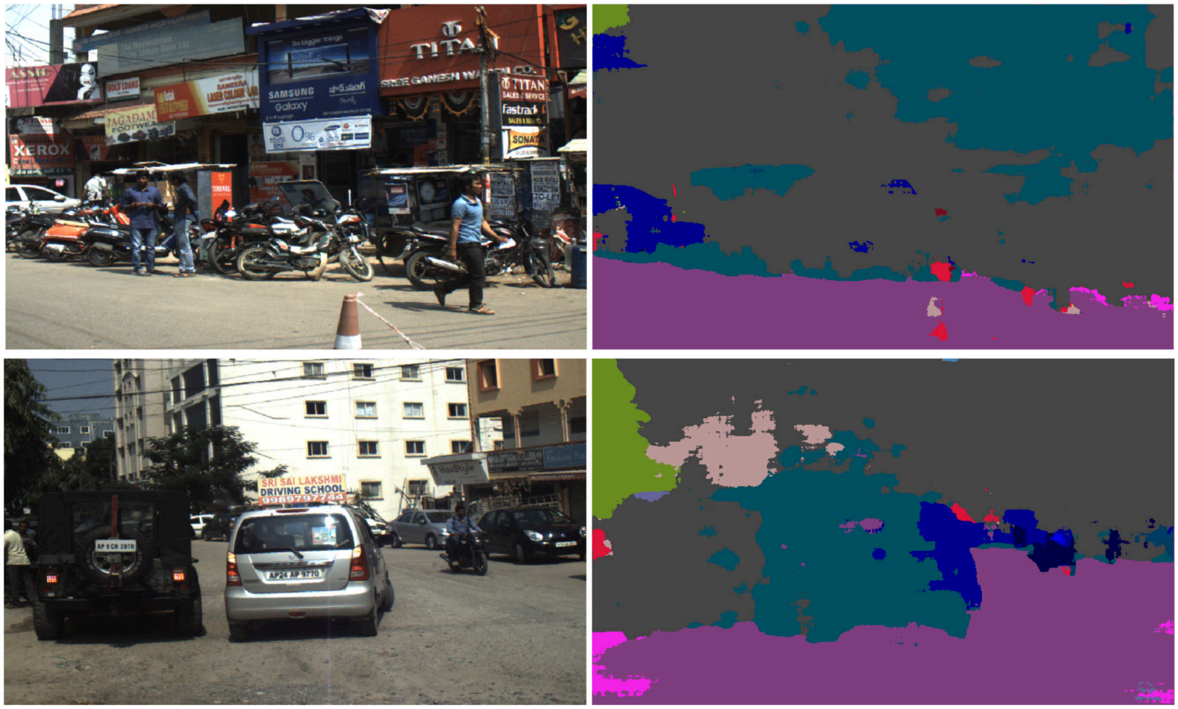


Figure 1.3: Use of information boards: (Top Left) An array of billboards indicating the shops. (Bottom left) A vehicle with a billboard of a driving school [8].

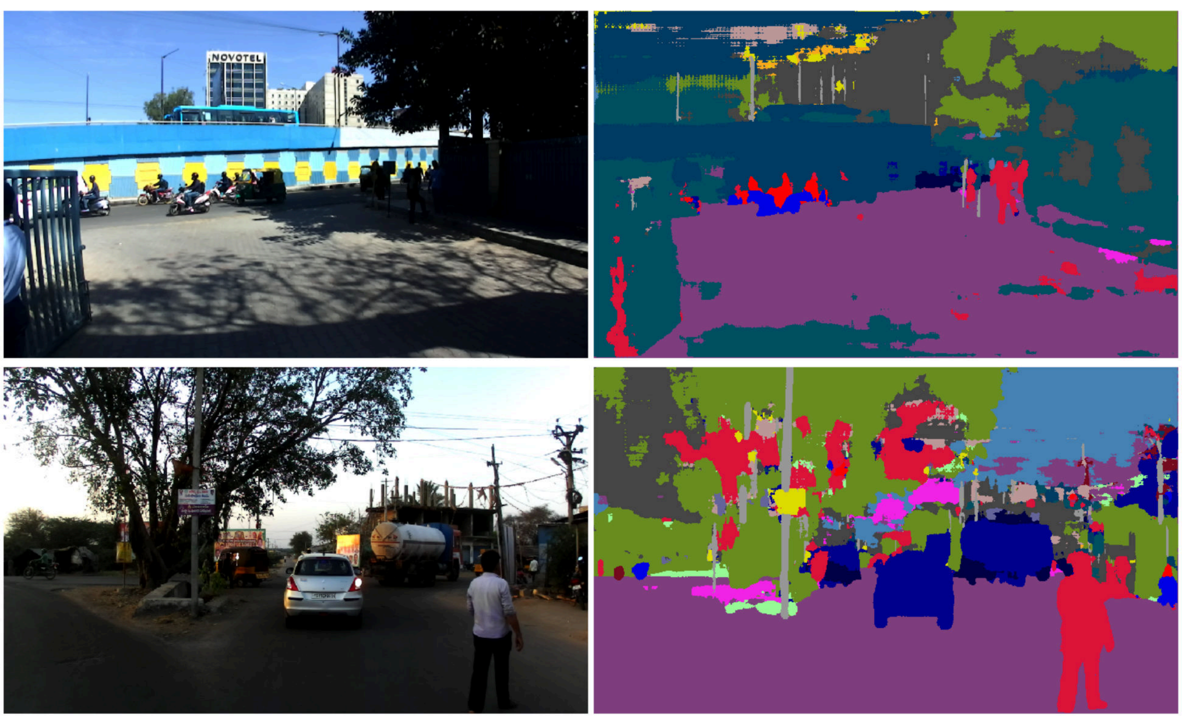


Figure 1.4: Diversity of ambient conditions:(Top left)Heavy shadows in the image or (Bottom left) Low light conditions [8].

ter, and variations in lighting and weather conditions. By understanding how loss functions interact with deep learning models in the context of road scene understanding, we aim to improve model robustness and performance, particularly in dynamic real-world driving scenarios. The insights gained from this study could contribute to developing more reliable, accurate, and efficient systems for both autonomous vehicles and advanced driver assistance systems (ADAS).

1.6.2 Contributions

The following are the technical contributions from the author as part of the thesis:

- Understand and appreciate the role of small objects in Road scene understanding. Various small objects present in road situations have been identified. Such identification is new within itself. This is specially carried out in the Indain road context.

- Pratical issues on roads due to small objects have been identified. This identification is highly reasonable as they drive towards the importance of small object detection.

- A comprehensive understanding of loss function is a step towards solving this problem. Loss functions necessary for semantic segmentation have been studied in detail and summarized.

- Small objects are less studied which is proven by the fact that small object detection was studied but segmenation is a field less studied relatively. Hence Indian Driving Dataset(IDD) has been analyzed with respect to small objects to bring forth the distribution of small objects in the dataset.

- Sematic Segmenation of Indian Driving Dataset(IDD) has been studied with respect to various loss functions.

1.6.3 Organization of the Thesis

- Chapter 2 discusses the related work and research directions in object detection and semantic segmentation. Also discuss the related works on small objects.

- Chapter 3 gives an idea and talks about several loss functions used in semantic segmentation and detection.

- Chapter 4 discusses the analysis of small objects in Indian Driving Dataset, architecture used for semantic segmentation, results, plots and inferences. It also talks about object detection, architecture used, results and plots.

- Finally, Chapter 5 concludes this thesis by summarizing the results obtained thus far. It provides an overview of the work conducted and presents the envisioned structure of the proposed thesis.

Chapter 2

Related Work

2.1 Understanding Road Scenes

AI algorithms used for autonomous driving and driver assistance systems need to interpret road scenes with a high level of accuracy to ensure safe and reliable operation. This involves a combination of computer vision, sensor fusion, deep learning, and robotic control systems to process and understand complex environments in real-time. Camera and computer vision are key to this progress. Driver Assistance systems need to understand (i) Traffic Signs (ii) Lane Detection and Tracking (iii) Broader perception of the surrounding and finally efficient execution.

Two fundamental problems of interest in this space are:

- Object Detection
- Semantic Segmentation

We discuss these in detail in the next section.

2.2 Object Detection and Semantic Segmentation

2.2.1 Object Detection

Object detection has become a central task in computer vision, and today it is carried out using a variety of deep learning models that can identify and locate objects within an image or video stream. The field has evolved significantly with advancements in deep learning and neural networks, especially with Convolutional Neural Networks (CNNs) and more specialized architectures for object detection.

Process involved some amount of data preparation and model building. The first step is gathering labeled data (images with bounding box annotations around objects of interest). Datasets like COCO [10], Pascal VOC [9], and ImageNet [24] are commonly used. Before feeding images into a model, preprocessing steps like resizing, normalization (scaling pixel values), and augmentation (e.g., random flipping, rotation) are often used to enhance model generalization.

Most state-of-the-art object detection models rely on deep learning architectures, particularly CNNs, with some models utilizing specialized designs to handle both localization and classification tasks simultaneously. Popular architectures include Region-based CNNs (R-CNN) [25] and Variants. R-CNN (2014) was one of the earliest breakthroughs, using selective search to propose candidate regions (potential objects), which were then passed through. Fast R-CNN [26] improved the original R-CNN by using a region of interest (RoI) pooling layer, which made the detection process faster by eliminating the need to run CNNs independently on each region proposal. Faster R-CNN [27] introduced a Region Proposal Network (RPN) that generates region proposals directly, eliminating the need for selective search, which greatly improved speed.

YOLO (2016) [28]: Unlike R-CNN-based approaches, YOLO frames object detection as a single regression problem. It divides an image into a grid and predicts bounding boxes and class probabilities directly for each grid cell. This makes YOLO extremely fast compared to earlier methods. YOLOv3, YOLOv4, YOLOv5 [29], and YOLOv7: Subsequent versions of YOLO improved accuracy and speed, with YOLOv4 being known for balancing real-time performance and high accuracy. YOLOv5, although not officially from the original YOLO authors, gained popularity due to its ease of use and excellent performance. YOLOv8: The latest iteration, continuing to refine and optimize the architecture for performance on various hardware. SSD (2016) [30] is another fast, end-to-end object detection model. It uses multiple convolutional feature maps to detect objects of different sizes. SSD is particularly well-suited for real-time applications and trades off some accuracy for speed.

The general pipeline for object detection today can be broken down into several steps. Input Image: The image is passed through a deep learning model (typically a CNN-based architecture). Feature Extraction: The model uses convolutional layers to extract feature maps from the input image. These feature maps represent the spatial hierarchy of patterns in the image. Region Proposals or Direct Predictions: Depending on the architecture, the network either generates candidate bounding boxes (e.g., using an RPN) or directly predicts bounding boxes and class scores. Bounding Box Regression and Classification: For each region or grid cell, the model predicts: Class of the object Coordinates of the bounding box (x, y, width, height) Non-Maximum Suppression (NMS): To reduce redundant detections and keep the best ones, non-maximum suppression is used to discard overlapping bounding boxes with lower confidence scores. Final Detection Output: The final output consists of the predicted bounding boxes, associated class labels, and confidence scores. There are many advanced techniques using transformers (DETR [31], ViTs [32]) today.

After detecting objects, several post-processing techniques may be used to refine results. Bounding Box Refinement: Fine-tuning bounding boxes based on IoU (Intersection over Union). Class Filtering: Some models use confidence thresholds to filter out detections with low probabilities.

Evaluation Object detection models are evaluated using metrics like: Mean Average Precision (mAP): A metric that averages precision at different recall levels. It considers both the accuracy of the detected bounding boxes and the correctness of the object class. Intersection over Union (IoU): Measures the overlap between predicted and ground truth bounding boxes. Typically, an IoU threshold (like 0.5) is used to determine whether a detection is a ”true positive.”

It is widely accepted that these are some of the challenges in object detection today.

- **Small Object Detection:** Detecting objects that occupy a small portion of the image is still a challenge.
- **Occlusions:** Objects partially obscured by other objects can be difficult to detect.
- **Real-time Performance:** Balancing detection accuracy with the need for real-time processing, especially in applications like autonomous driving or video surveillance.
- **Generalization:** Ensuring that models generalize well across different environments and conditions (e.g., different lighting, angles, or backgrounds).

Popular object detection methods leverage sophisticated deep learning architectures, with YOLO, Faster R-CNN, and SSD being some of the most widely used models. Advances in model design, such as the use of transformers (DETR) and efficient architectures (EfficientDet [33]), continue to push the boundaries in terms of accuracy, speed, and application flexibility. Object detection is a rapidly evolving field with a range of applications in areas like autonomous driving, healthcare, retail, and security.

2.2.2 Semantic Segmentation

Semantic segmentation involves assigning a class label to each pixel in an image, essentially segmenting the image into regions of different objects or classes. It is a critical task in computer vision with applications in fields like autonomous driving, medical imaging, robotics, and satellite imagery. Over the years, semantic segmentation has been transformed by advancements in deep learning and Convolutional Neural Networks (CNNs), with some models now achieving near-human levels of performance.

Encoder-Decoder Architectures are popular for this. Early segmentation models were often based on fully convolutional networks (FCNs), but more recent models use encoder-decoder architectures, which provide more precise pixel-wise predictions. The encoder extracts deep features from the input image using layers of convolution and pooling. The output is a compact, high-level feature map that represents the image.The decoder gradually upsamples these feature maps to the original image resolution and makes pixel-wise predictions. The encoder-decoder structure allows the model to capture both local context (fine-grained details) and global context (overall image structure).

Fully Convolutional Networks (FCNs) have been one of the early success stories in this area. FCNs replace fully connected layers with convolutional layers in traditional CNNs, making the architecture capable of handling images of any size. FCNs perform pixel-wise classification by directly predicting the class of each pixel. FCNs use transposed convolutions (or deconvolutions) to upsample the output feature maps back to the original resolution of the input image.

U-Net is a key architecture. U-Net is an encoder-decoder network that is specifically designed for image segmentation in biomedical applications. It includes skip connections between the encoder and decoder to help preserve spatial information. This allows the model to perform better on small objects and fine details. The feature maps from the encoder (which are spatially smaller) are concatenated with the corresponding feature maps from the decoder (which are larger), helping to recover fine-grained spatial information.

Loss Functions for Semantic Segmentation

- Cross-Entropy Loss
- Dice Loss
- IoU (Intersection over Union) Loss
- Tversky Loss

We pay special attention to the loss functions in this area and the next chapter presents our understanding of Loss functions.

Semantic segmentation today is dominated by deep learning-based models that combine powerful CNN architectures, multi-scale feature learning, and advanced techniques like transformers and attention mechanisms. These models are capable of producing pixel-wise accurate segmentations, even for complex scenes with varying object sizes and challenging conditions (e.g., small objects, occlusions).

As the field continues to evolve, we see an increasing use of transformer-based models (e.g., Swin Transformer [34]), self-supervised learning, and hybrid models that combine CNNs with transformers for even more robust and accurate segmentation.

2.3 Small Objects: Detection and Segmentation

The problem of small object detection is relevant in various fields, with direct applications including: enhancing the classification and detection of elements in satellite images, improving video surveillance through the analysis of footage captured by security cameras positioned at high vantage points, and boosting the detection of pedestrians or traffic. This article specifically focuses on the latter application. With the proposed solution, it will be possible to detect a greater number of objects and improve the accuracy of each detection, all without the need to retrain the model. This makes the solution particularly beneficial for automated traffic control systems.

While significant progress has been made in the field of object detection in natural scenes—especially with the advent of deep convolutional networks (ConvNets)—the problem remains far from solved, particularly when it comes to detecting small objects. Small objects are defined as those that occupy less than 1-10% of the image area. This issue is especially relevant in a variety of challenging real-world applications today, such as detecting pedestrians, traffic signs, cars on roads, and objects in aerial imagery.

Detecting small objects is inherently difficult for several reasons. First, it's challenging to distinguish small objects from background clutter, making it hard for many standard detectors that rely on "objectness" to perform well. The increased number of potential object locations in the background further complicates detection. Second, as small objects pass through standard CNN architectures like VGG16, their activations shrink with each pooling layer. For example, an object that is 32×32 pixels will be reduced to just 1 pixel after passing through the block5 pool layer of VGG16, making it easy for these activations to be overlooked. Third, many small objects have simple shapes that are difficult to break down into smaller parts, whereas popular CNN-based detectors excel at learning hierarchical features. Finally, there is a lack of large publicly available datasets specifically for small objects. While datasets like MS COCO and VOC2012 contain instances of small objects, there are no dedicated large-scale datasets for small object detection. Furthermore, much of the prior research has focused on datasets with larger objects. For instance, the mean Average Precision (mAP) for state-of-the-art end-to-end detectors on a dataset like PASCAL VOC is 76.3% but on datasets that primarily feature small objects, the mAP drops to just 27%.

The challenge of detecting small objects is highlighted by the vast search space, background clutter, and the weakening of object signals as they pass through standard convolutional layers. For instance, the mouse (in the green box) is a small object, but it's difficult to detect among other similarly sized objects in the image.

In recent years, one popular approach to object detection has been the use of object proposal techniques, where several candidate regions are suggested and then classified by a deep CNN model [3]. These methods have proven effective because the features learned by deep CNNs are far more discriminative than hand-crafted features. Unlike earlier methods that used dense sliding windows to exhaustively scan the image for potential object regions, newer algorithms that rely on low-level cues generate fewer, more sparse proposals. This shift led to the development of Region Proposal Networks (RPN) [1], which not only generate better proposals but also significantly speed up the detection process by sharing the convolutional layers between the proposal and detection networks.

In our work, we adopt a similar strategy: an RPN generates proposals, which are then classified by a deep CNN. However, we observed that the standard RPN, as used in the widely adopted Faster R-CNN framework [1], struggles to detect small objects due to the large size of its anchor boxes. To address this, we carefully study the impact of anchor box size for a given dataset. Our findings demonstrate that optimizing the anchor box size can significantly improve performance, outperforming other existing methods for small object detection.

Before the rise of deep learning, object detection tasks, such as vehicle detection in aerial imagery, relied heavily on handcrafted features. With the advent of deep learning, Convolutional Neural Networks (CNNs) have taken over the task of learning discriminative features, greatly improving detection performance.

Recently, several approaches have emerged that do not rely on region proposals. For instance, YOLO (You Only Look Once) divides the image into a grid and predicts class labels and bounding boxes for each grid cell. Another notable method is Single Shot Detection , which proposes fixed boxes of various scales where objects are likely to appear, and evaluates the presence of objects for each of these boxes during testing. However, it has been shown that proposal-based methods tend to outperform proposal-free methods in terms of both recall and accuracy. Popular methods for proposal generation, such as those described in references, use low- and mid-level features to generate proposals. Recently, using deep networks for proposal generation has gained popularity. For example, Deepbox, re-ranks proposals generated by Edgeboxes, while DeepProposal, applies an inverse cascade that works backward from the final to the initial convolutional layers of the CNN. The Region Proposal Network (RPN) can share convolutional layers with the classifier network, using anchor boxes of various scales and aspect ratios that slide across the feature map from the last convolutional layer. This RPN functions as an attention mechanism, directing the detector to focus on relevant areas of the image.

The performance of Faster R-CNN was improved by introducing the RoIPooling layer, which maps images of varying sizes to a fixed-dimensional feature map. Faster R-CNN consists of two components: the RPN, which feeds into Fast R-CNN for final detection. All these methods predict both bounding boxes and the probability of class membership for each object. Sharing weights between the proposal network and the detector has been shown to significantly reduce runtime while improving performance.

Several subsequent detectors have been built upon the Faster R-CNN framework. However, many of them only briefly address the challenge of detecting small objects. Many recent methods such as focus on adapting the Fast R-CNN architecture for tasks like logo, face, and pedestrian detection, all of which involve small object detection.

Small Object RCNN is perhaps the first work to focus specifically on the problem of small object detection. This paper introduces a small dataset, an evaluation metric, and provides a baseline score. It suggests modifications to the Region Proposal Network, demonstrating improvements in recall and mean average precision. Notably, it recommends using smaller anchor box sizes and attaching them to the conv43 layer rather than conv53 of VGG16. The authors also argue that the RoI pooling layer may not preserve enough information for small objects and therefore suggest adhering to the RCNN framework. In this work, we extend these ideas and propose a more efficient approach for achieving comparable performance in an end-to-end pipeline.

The disparity between low and high-resolution spaces means that a single model may not work effectively across all object sizes, as most large datasets are heavily biased toward larger, high-resolution objects.Many recent papers have tackled challenges in activity recognition for low-resolution videos by mapping both low- and high-resolution inputs to a common space. They also highlights that working with low-resolution data is more challenging due to its sensitivity to small translations. Meanwhile, high-resolution data to help learn filters for the low-resolution domain. In contrast, our approach takes advantage of the fact that classifiers perform well in high-resolution spaces, and we leverage super-resolution techniques to transfer classification tasks to a network that has been pretrained on high-resolution images. The tasks of image denoising and image super-resolution have been well explored in recent years, with numerous deep learning approaches.

Chapter 3

Loss Functions for Detection and Segmentation

3.1 Role of Loss Functions

Loss functions play a crucial role in the training process of deep learning models. They serve as the measure of how well a model's predictions match the actual target values (ground truth). The loss function computes a scalar value that represents the discrepancy between the predicted and true labels. The goal during training is to minimize this loss, which, in turn, improves the model's ability to make accurate predictions. The loss function provides feedback to the network about its performance on the training data. This error is propagated back through the network during backpropagation, helping the model learn better feature representations. The loss function is essential for guiding optimization algorithms, such as gradient descent or its variants (Adam, SGD, etc.), to adjust the model's parameters (weights). During each iteration of training, the model computes the gradient of the loss with respect to the model parameters, which tells the optimizer how to update the parameters in order to reduce the error.

A single loss function does not meet all the needs. Some of the special challenges are:

- Class Imbalance. In tasks where some classes are underrepresented, loss functions like Weighted Cross-Entropy Loss or Focal Loss are used to put more emphasis on the minority class to prevent bias toward the majority class.

- Small Object Detection: In tasks like object detection (e.g., small object segmentation), specialized loss functions such as Dice Loss or IoU (Intersection over Union) Loss are used to handle small object segmentation challenges. These loss functions help focus on fine-grained accuracy where the objects are hard to detect and their boundaries are more critical.

- Segmentation Loss: For semantic segmentation tasks, Dice Loss and IoU Loss are commonly used, as they focus on the overlap between predicted and true segmentations, which is especially useful for tasks involving pixel-level precision.

In modern deep learning, the role of loss functions cannot be overstated. They directly influence how a model learns and how well it generalizes to new data. The choice of loss function must align with the task at hand—whether it's classification, regression, segmentation, or another domain. In addition, loss functions can be designed to address specific challenges like class imbalance, small object detection, and regularization. The development and refinement of loss functions continue to be an active area of research, as they are key to improving the performance, stability, and effectiveness of deep learning models across a wide range of applications.

Semantic Segmentation is a popular task in Deep Learning area of Computer Vision. A loss function in a Deep Learning model is one that quantifies the error between the predicted and actual values of the model. The following ar some important loss functions used in semantic segmentation. [5] [6]

3.2 Examples of Loss Functions

3.2.1 Binary Cross-Entropy

Cross-entropy is defined as a measure of the difference between two probability distributions for a given random variable or set of events. It is widely used for classification objective, and as segmentation in pixel-level classification it works well. Binary Cross-Entropy (Figure 3.1) is defined as [6]:

$$L_{BCE(y,\hat{y})} = -(ylog(\hat{y}) + (1-y)log(1-\hat{y}))$$

Here, $\hat{y}$ is the predicted value by the prediction model.

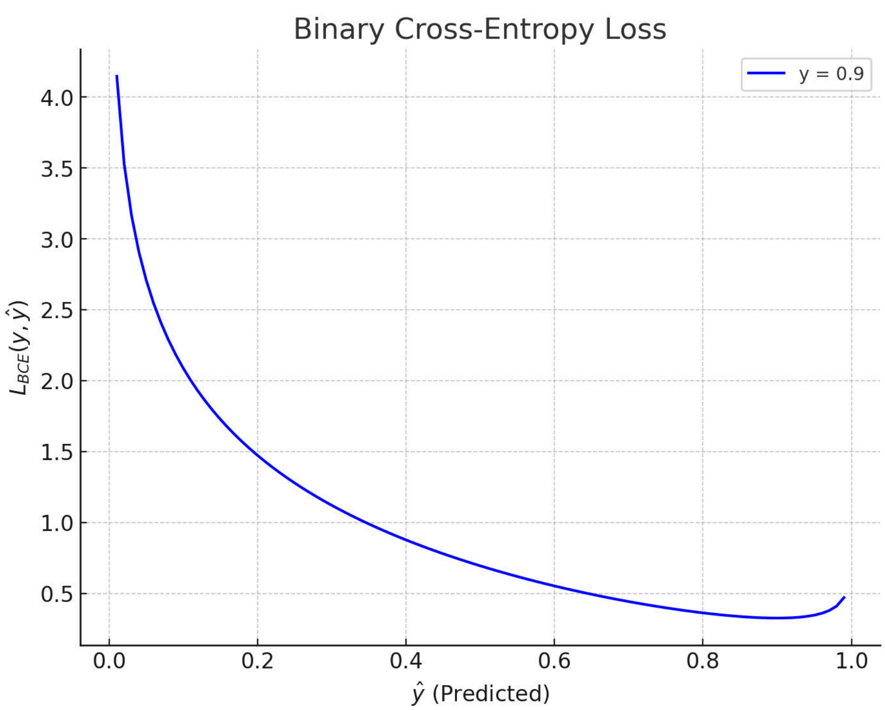


Figure 3.1: Grpah of Binary cross entropy loss

3.2.2 Weighted Binary Cross-Entropy

Weighted Binary Cross Entropy (WCE) is a modified version of binary cross entropy where positive examples are assigned a specific weight or coefficient. This approach is particularly useful when dealing with imbalanced or skewed data. The Weighted Cross Entropy (Figure 3.2) can be defined as:

$$L_{W-BCE(y,\hat{y})} = -(\beta y log(\hat{y}) + (1-y)log(1-\hat{y}))$$

β value can be used to tune false negatives and false positives.If you want to reduce the number of false negatives then set $\beta > 1$, similarly to decrease the number of false positives, set $\beta < 1$ [6]. The idea behind choosing $\beta > 1 \quad and \quad \beta < 1$ is that, since it is multiplied with the probability value y the effective value considered during training is βy instead of y and the predicted probability, $\hat{y}$, gets adjusted accordingly affecting the FNs and FPs.

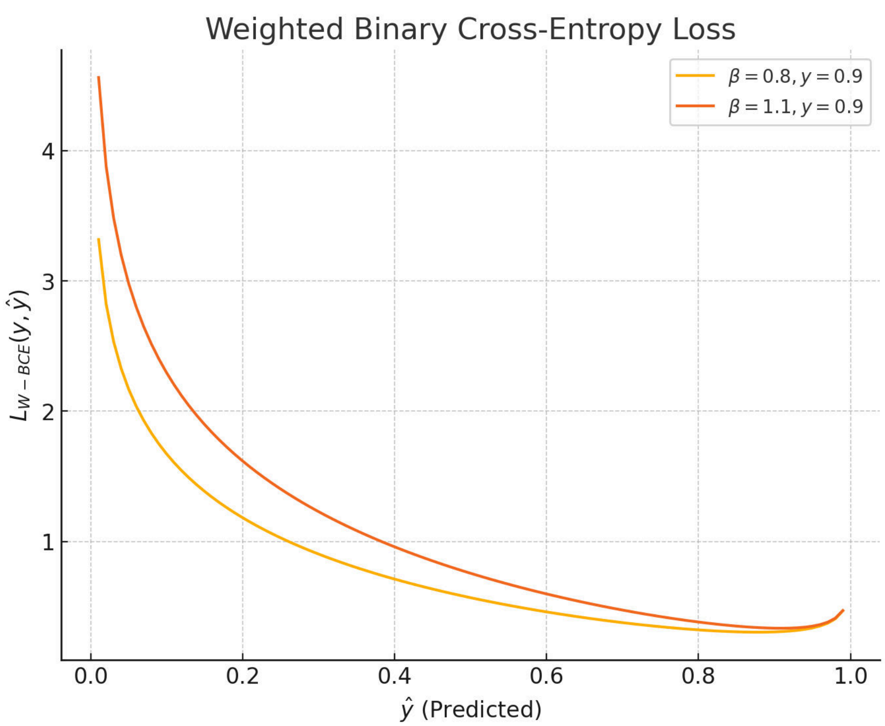


Figure 3.2: Grpah of Weighted Binary cross entropy loss

3.2.3 Balanced Cross Entropy

Balanced Cross Entropy (BCE) is similar to Weighted Cross Entropy, with the key difference being that it assigns weights to both positive and negative examples [8]. The Balanced Cross Entropy (Figure 3.3)can be defined as follows:

$$L_{BCE(y,\hat{y})} = -(\beta y log(\hat{y}) + (1-\beta)(1-y)log(1-\hat{y}))$$

Here, β is defined as $1 - \frac{y}{H*W}$ [6].

The idea behind the choice of β is similar to the one behind weighted binary cross-entropy. Since $1 - \beta$ is also present in the negative sample side of the loss, it has the same effect that β does. $H * W$ includes the size of the image in β.

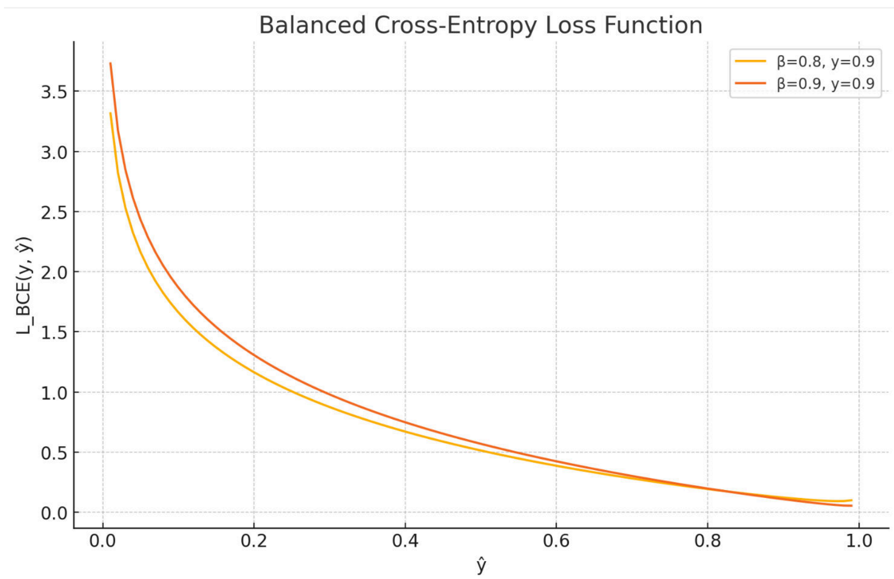


Figure 3.3: Grpah of Balanced cross entropy loss

3.2.4 Focal Loss

Focal Loss (FL) is another variation of Binary Cross-Entropy that reduces the impact of easy examples, allowing the model to focus more on learning hard examples. It is particularly effective in handling highly imbalanced class scenarios. To understand how Focal Loss is formulated, we will first examine the binary cross-entropy loss and then explore how Focal Loss is derived from it.

$$CE = \begin{cases} -log(p) \quad if \quad y = 1 \\ -log(1-p), otherwise \end{cases}$$

To make convenient notation, Focal Loss defines the estimated probability of class as:

$$p_t = \begin{cases} p \quad if \quad y = 1 \\ 1-p, otherwise \end{cases}$$

Therefore, now Cross-Entropy can be written as,

$$CE(p, y) = CE(p_t) = -log(p_t)$$

Focal Loss (Figure 3.4)introduces a modulating factor, $(1 - p_t)^\gamma$, to reduce the weight of easy examples and concentrate training on harder negatives, as shown below:

$$FL(p_t) = -\alpha_t (1 - p_t)^\gamma log(p_t)$$

Here, $\gamma > 0$. Similarly, α generally range from [0,1], It can be set by inverse class frequency or treated as a hyperparameter [6].
The idea behind the loss is that lesser values of p_t give higher loss. Misclassified positive samples are introduced again as hard negatives.

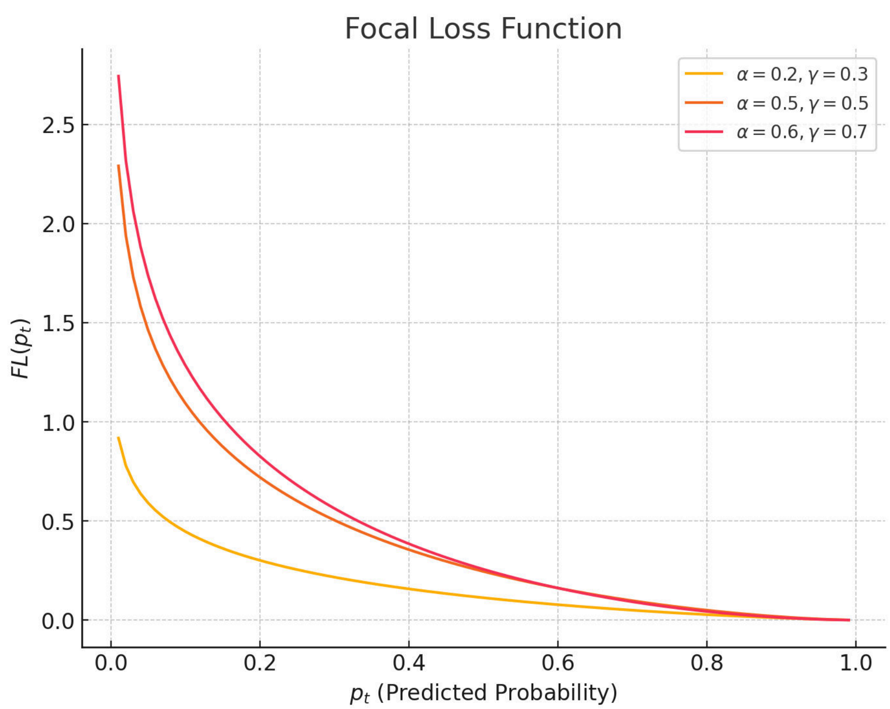


Figure 3.4: Grpah of Focal loss

3.2.5 Dice Loss

The Dice coefficient is a popular metric in the computer vision community for measuring the similarity between two images. It has since been adapted into a loss function, referred to as Dice Loss 3.5.

$$DL(y, \hat{p}) = 1 - \frac{2y\hat{p} + 1}{y + \hat{p} + 1}$$

Here, 1 is added in numerator and denominator to ensure that the function is not undefined in edge case scenarios such as when $y = \hat{p} = 0$ [6].
The idea behind the loss is the one behind choosing a harmonic mean, where the harmonic mean has a value closer to the lower value and hence the minimization of loss moves it closer to the larger value.

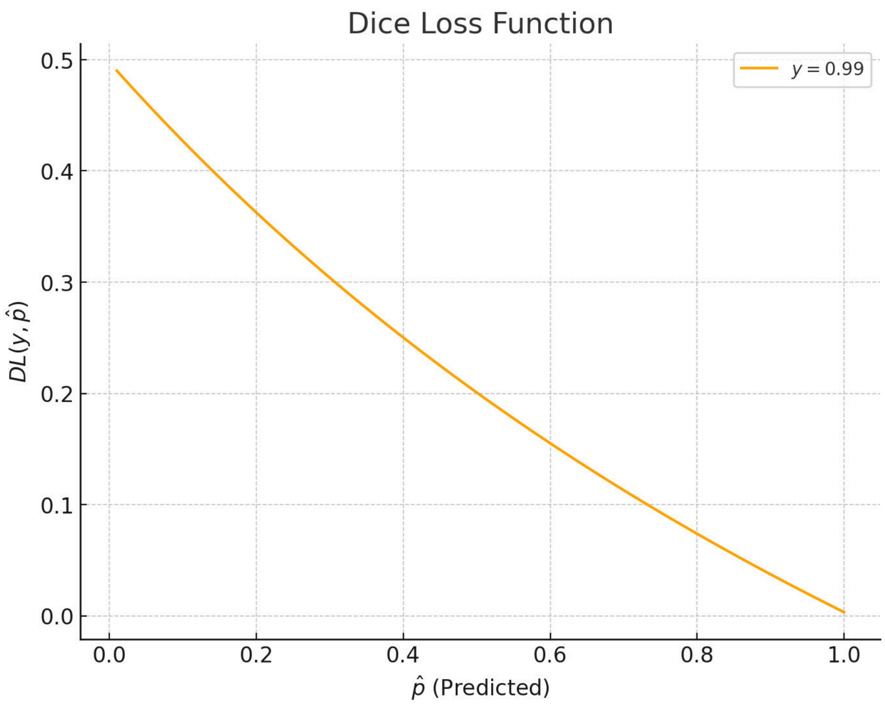


Figure 3.5: Grpah of Dice loss

3.2.6 Tversky Loss

The Tversky Index (TI) is a generalized form of the Dice coefficient. It incorporates weights for false positives (FP) and false negatives (FN) using a coefficient β.

$$TI(p, \hat{p}) = \frac{p\hat{p}}{p\hat{p} + \beta(1-p)\hat{p} + (1-\beta)p(1-\hat{p})}$$

Here, when β = 1/2, It can be solved into regular Dice coefficient. Similar to Dice Loss, Tversky loss (Figure 3.6) can also be defined as [6]:

$$TL(p, \hat{p}) = 1 - \frac{p\hat{p}}{p\hat{p} + \beta(1-p)\hat{p} + (1-\beta)p(1-\hat{p})}$$

The idea behind the loss is that the weight β helps in focussing on FPs and FNs.

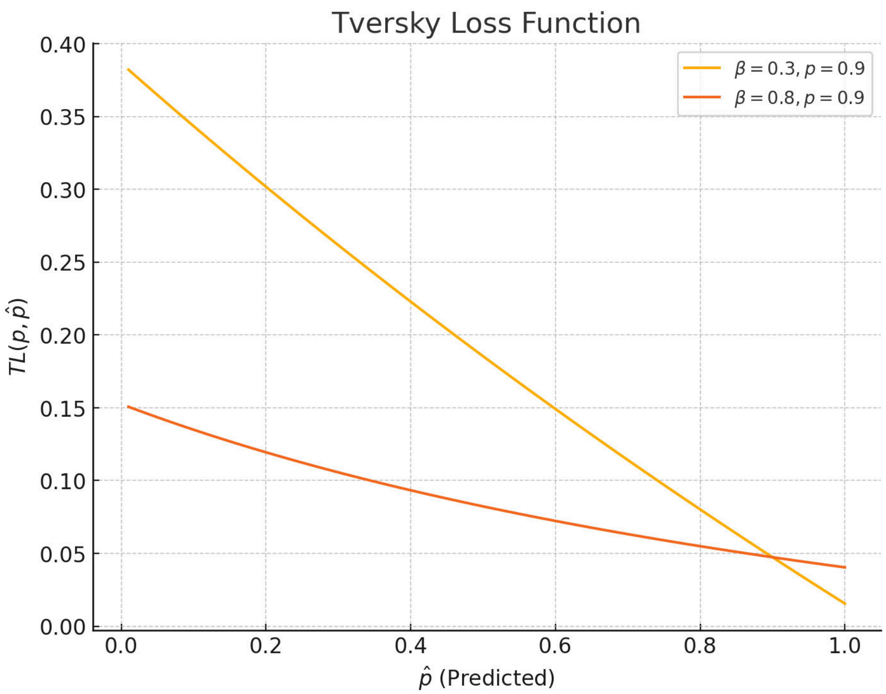


Figure 3.6: Tversky loss

3.2.7 Focal Tversky Loss

Similar to Focal Loss, which emphasizes hard examples by reducing the weight of easy or common ones, Focal Tversky Loss also focuses on learning hard examples, such as those with small regions of interest (ROIs), using a γ coefficient, as shown below:

$$FTL = \sum_c (1 - TI_c)^{\gamma}$$

here, TI indicates Tversky Index, and γ can range from [1,3] [6].
As the name suggests, the Focal Tversky Loss (Figure 3.7)is the combination of Focal Loss and Tversky Loss.

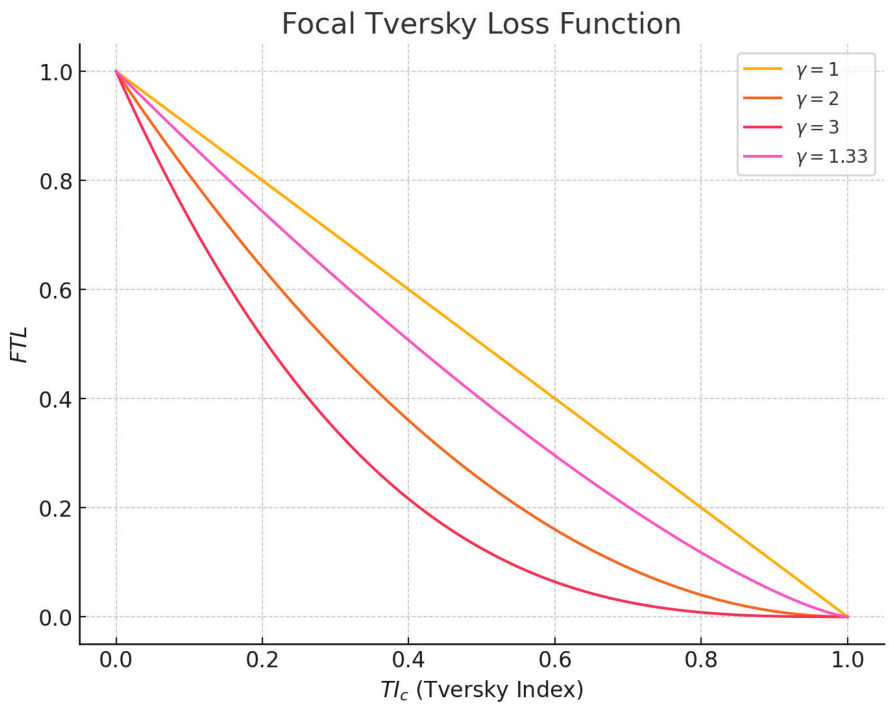


Figure 3.7: Focal Tversky loss

3.2.8 Sensitivity Specificity Loss

Similar to the Dice Coefficient, Sensitivity and Specificity are commonly used metrics for evaluating segmentation predictions. This loss function addresses the class imbalance problem by incorporating a weighting parameter w. The loss (Figure 3.8) is defined as [6] [5]:

$$SSL = 1 - (w * sensitivity + (1 - w) * specificity)$$

where,

$$sensitivity = \frac{TP}{TP + FN}$$

$$specificity = \frac{TN}{TN + FP}$$

The idea behind this loss is to weigh the TPs and TNs. The denominators TP+FN and TN+FP, which are Total positive ground truths and Total negative ground truths, help in maintaining the loss between 0 and 1.

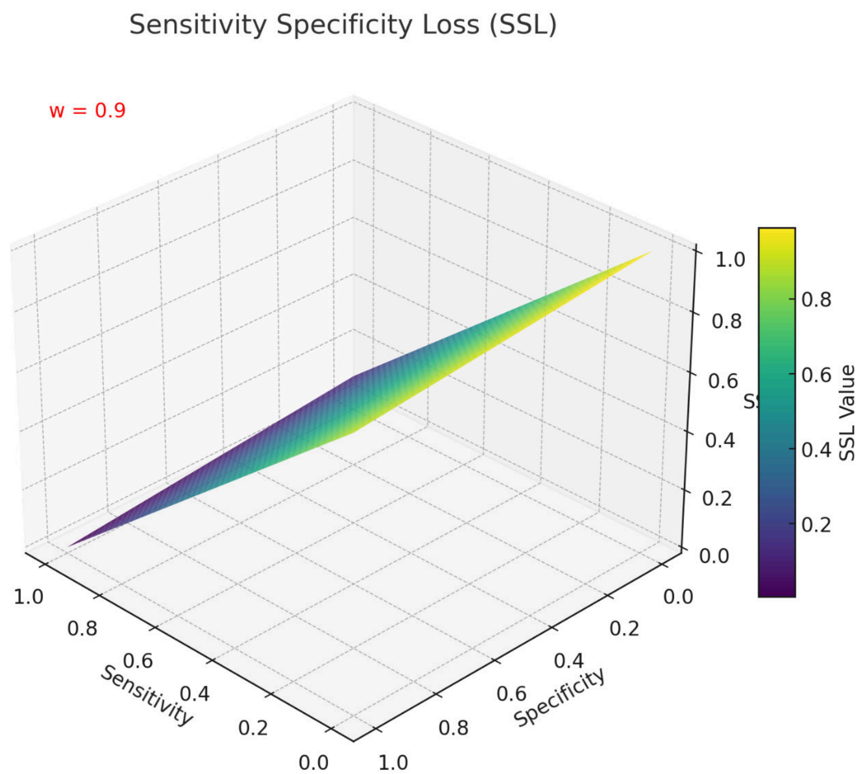


Figure 3.8: Sensitivity Specificity loss

3.2.9 Shape-aware Loss

As the name implies, Shape-aware Loss considers the shape of objects during segmentation. While most loss functions operate at the pixel level, Shape-aware Loss computes the average point-to-curve Euclidean distance between the points on the predicted segmentation boundary and the ground truth. This distance is then used as a coefficient to the cross-entropy loss function. It is defined as follows:

$$E_i = D(\hat{C}, C_{GT})$$

$$L_{shape-aware} = -\sum_i CE(y, \hat{y}) - \sum iE_iCE(y, \hat{y})$$

Using E_i the network learns to produce a prediction masks similar to the training shapes [6].

3.2.10 Combo Loss

Combo Loss (Figure 3.9) is defined as a weighted sum of Dice Loss and a modified cross-entropy. It aims to combine the benefits of Dice Loss in handling class imbalance with the ability of cross-entropy to smooth curves. The Dice Loss in this case is taken as the Dice Coefficient itself. It is defined as [6]:

$$L_{m-bce} = -\frac{1}{N}\sum_{i}\beta(ylog(\hat{y})) + (1-\beta)(1-y)log(1-\hat{y})$$

$$CL(y,\hat{y}) = \alpha L_{m-bce} - (1-\alpha)DiceCoefficient(y,\hat{y})$$

The idea behind the loss is that focussing on FPs and FNs(modified cross entropy) helps in moving the shape of the segmented object closer to ground truth. Dice Loss helps with class imbalance and small objects.

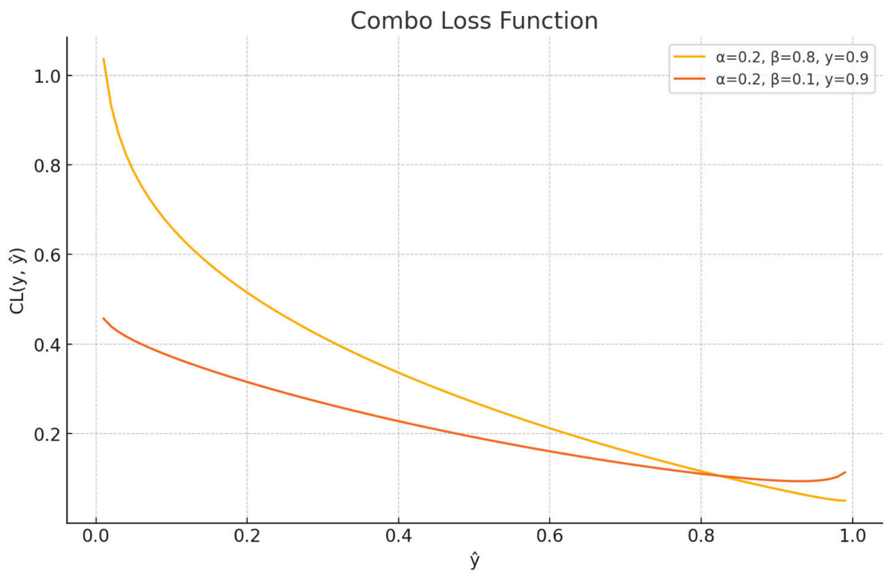


Figure 3.9: Combo Loss

3.2.11 Exponential Logarithmic Loss

Exponential Logarithmic Loss (Figure 3.10), like Combo Loss, combines weighted cross-entropy loss and Dice coefficient to address the class imbalance issue. The key difference is that Exponential Logarithmic Loss applies logarithmic and exponential transformations to Dice Coefficient and exponential transformation to weighted cross-entropy loss before combining them. This approach provides flexibility in controlling the model's focus on easy or hard pixels. The proposed loss function is defined as follows:

$$L_{Exp-Log} = w_{Dice}L_{Exp-Log-DiceCoefficient} + w_{Cross}L_{Exp-WCE}$$

where $L_{Exp-Log-DiceCoefficient}$ is the exponential logarithmic Dice Coefficient loss and $L_{Exp-WCE}$ is the exponential weighted cross-entropy loss:

$$L_{Exp-Log-DiceCoefficient} = (-log(DiceCoefficient))^{\gamma_{Dice}}$$

$$L_{Exp-WCE} = (\beta * (-ylog(\hat{y}))^{\gamma_{Cross}} + (1-\beta) * (-(1-y)log(1-\hat{y}))^{\gamma_{Cross}}$$

Here γ_{Dice} and γ_{WCE} can be used to control the focus of the loss function. Specifically, with $\gamma > 1$, the loss focuses more on hard-to-classify pixels and vice versa [6] [5].

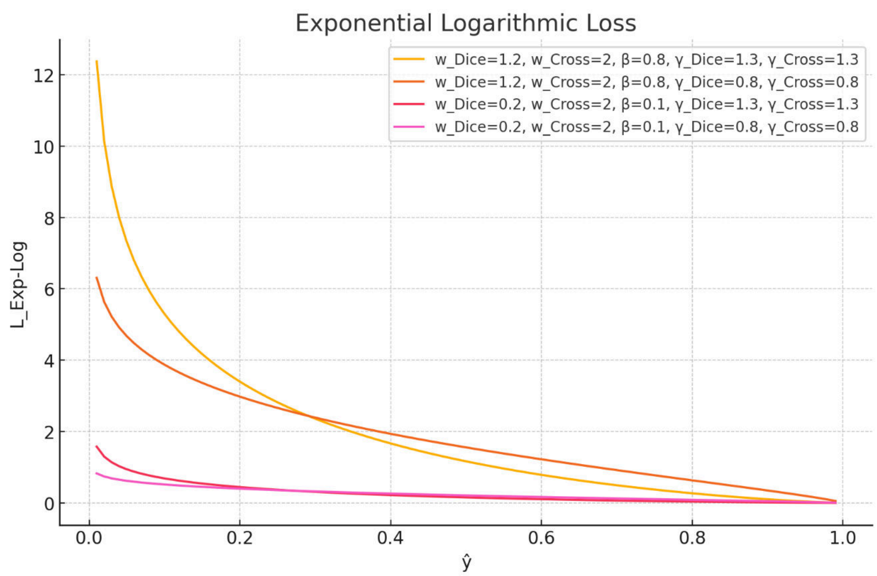


Figure 3.10: Exponential Logarithmic Loss

3.2.12 Distance Map Derived Cross- Entropy Loss

Many semantic segmentation models experience performance degradation at object boundaries. A simple approach to address this issue is to apply a higher penalty for segmentation errors at these boundary regions. To achieve this, distance maps are used. A distance map has the same dimensions as the image, where each pixel is assigned the smallest distance to a boundary pixel. The inverse of the distance map, ϕ, is then used as a weight for the cross-entropy loss, ensuring that pixels near the boundary receive a higher weight, while those farther from the boundary are given a lower weight [5].

$$L_{DMCE(y,t,\phi)} = -\sum_{n=1}^{N}(1+\phi_n)log(t_n, y_n)$$

t_n and y_n are one hot encoded vector and generated class probabilities for the n^{th} pixel.
The constant 1 is added to ϕ to avoid the vanishing gradient problem.

3.2.13 Hausdorff Distance Loss

Hausdorff Distance (HD) is a widely used evaluation metric in medical image segmentation. It is defined for pairs of sets and measures the maximum distance between a point in one set and the nearest point in the other set, capturing the worst-case scenario. In this context, two non-empty point sets, denoted as X and Y, are considered, with a distance measure $d(x, y)$ between points. $x \in X$ and $y \in Y$, typically using metrics such as Euclidean or Manhattan distances. The Hausdorff distance is defined as:

$$HD(X, Y) = max(d_h(X, Y), d_h(Y, X))$$

$$d_h(X, Y) = max_{x \in X} min_{y \in Y} d_{(x,y)}$$

$$d_h(Y, X) = max_{y \in Y} min_{x \in X} d_{(y,x)}$$

In image segmentation, Hausdorff Distance (HD) is calculated between the boundaries of the predicted and ground truth masks. While HD is a commonly used metric, it has certain drawbacks. Unlike other metrics that assess overall segmentation performance, HD focuses solely on the largest error, making it highly sensitive to outliers. As a result, optimizing exclusively to minimize the largest error can lead to instability in the algorithm and unreliable results. Additionally, minimizing only the largest segmentation error may reduce overall performance, particularly for compound shapes, which are prevalent in medical imaging. This is because, although the model may perform well on most of the image, it could experience significant errors in a few exceptionally challenging regions [5].

3.2.14 Correlation Maximized Structural Similarity Loss

Many semantic segmentation loss functions primarily focus on classification errors at the pixel level, often neglecting the structural information at the pixel level. Structural Similarity Loss (SSL) achieves a high positive linear correlation between the ground truth map and the predicted map. The approach is divided into three steps: Structure Comparison, Cross-Entropy weight coefficient determination, and mini-batch loss definition.

In the Structure Comparison step, the authors calculate the ecoefficient, which measures the degree of linear correlation between the ground truth and the predicted map:

$$e = \left| \frac{y - \mu_y + C_4}{\sigma_y + C_4} - \frac{p - \mu_p + C_4}{\sigma_p + C_4} \right|$$

Here, C4 is stability factor set to 0.01 as an empirical observed value. μ_y and σ_y are the local mean and standard deviation of the ground truth y respectively.

After calculating the degree of correlation, e is used as coefficient for cross entropy loss function, defined as:

$$f_{n,c} = \mathbf{1} * (e_{n,c} > \beta e_{max})$$

Using this coefficient function, we can define SSL loss as:

$$Loss_{ssl}(y_{n,c}, p_{n,c}) = e_{n,c} f_{n,c} L_{CE}(y_{n,c}, p_{n,c})$$

and finally for mini-batch loss calculation, The SSL can be defined as:

$$L_{ssl} = \frac{1}{M} \sum_{n=1}^{N} \sum_{c=1}^{C} Loss_{ssl}(y_{n,c}, p_{n,c})$$

where, M is $\sum_{n=1}^{N} \sum_{c=1}^{C} f_{n,c}$. Using above formula, loss function will automatically abandon those pixel level predictions, which doesn't show correlation in terms of structure [6].

3.3 Discussions

The choice of loss function plays a pivotal role in the performance of deep learning models for object detection and semantic segmentation. It is important to consider task-specific aspects (classification vs. regression), handle class imbalance, address small object challenges, and ensure alignment with evaluation metrics. A careful selection of loss functions that suit the specific characteristics of the dataset and task can significantly improve the accuracy, robustness, and efficiency of the model.

Object Detection: Involves both classification (identifying what object is present) and regression (predicting the location of the object through bounding boxes). Therefore, object detection loss functions must account for both tasks simultaneously. A combination of classification loss (e.g., cross-entropy loss) and regression loss (e.g., Smooth L1 loss or IoU loss) is commonly used.

Semantic Segmentation: This involves classifying each pixel in an image, so the loss function should focus on pixel-level classification. Typically uses pixel-wise classification loss such as cross-entropy loss or Dice loss.

3.3.1 Handling Class Imbalance

Both object detection and semantic segmentation tasks can suffer from class imbalance, where certain classes (e.g., background or small objects) dominate, and the model may become biased toward predicting the majority class.

Focal Loss: A modified version of cross-entropy loss that down-weights well-classified examples and focuses more on hard-to-detect objects. This is commonly used in object detection to handle class imbalance.

Weighted Cross-Entropy Loss: A weighted version of cross-entropy to assign higher weights to underrepresented classes (such as small objects or foreground classes) in semantic segmentation.

Dice Loss and IoU Loss: Both these losses focus on improving the overlap between predicted and true object regions, which is useful when class imbalance is severe in segmentation tasks.

Chapter 4

Detection and Segmentation

4.1 Semantic Segmentation

4.1.1 Methods

In this work, we employ a CNN-based semantic segmentation architecture. Successive convolutional layers function as an encoder, progressively compressing the information, which is then decoded into a segmented image. Additional skip connections are incorporated to enhance feature recovery. Such architectures are widely known as U-Net. U-Net (Figure 2) has been utilized as the backbone, with a Fully Convolutional Network (FCN) serving as both the auxiliary and decode heads. Our motivation for using this architecture stems from its widespread success and popularity in semantic segmentation tasks.

Table 4.1: mIoU for Cross Entropy Loss, Weighted Cross Entropy Loss and Dice Loss (Please zoom for clearly reading the table)

Loss Type	road	parking	drivable fallback	sidewalk	rail track	non-drivable fallback	person	animal	rider	motorcycle
WCE loss	91.66	0	54.72	51.14	nan	45.65	53.15	0	64.38	68.03
CE loss	91.35	0	53.09	51.25	nan	35.7	47.06	0	60.38	64.58
Dice loss	90.8	0	53.32	43.74	nan	45.05	0	0	55.56	67.39
Loss Type	bicycle	autorickshaw	car	truck	bus	caravan	trailer	train	vehicle fallback	curb
WCE loss	0	73.01	83.38	52.78	54.99	0	na	na	10.04	62.44
CE loss	0	67.48	81.2	54.38	58.75	0	na	na	0.03	58.33
Dice loss	0	65.56	81.15	52.9	46.53	0	na	na	0	61.9
Loss Type	wall	fence	guard rail	billboard	traffic sign	traffic light	pole	polegroup	obs-str-bar-fallback	building
WCE loss	40.61	19.07	48.7	51.91	53.84	0	45.12	20.08	29.64	56.2
CE loss	40.65	12.5	44.63	49.72	10.6	0	43.08	0	27.05	50.55
Dice loss	41.32	0	0	52.42	0	0	40.61	0	28.79	55.45
Loss Type	bridge	tunnel	vegetation	sky	fallback background	unlabeled	ego vehicle	rectification border	out of roi	license plate
WCE loss	38.69	0	85.04	95.69	0	0	0	na	na	na
CE loss	5.79	0	83.82	95.5	0	0	0	na	na	na
Dice loss	0	0	84.67	94.56	0	0	0	na	na	na

Table 4.2: Accuracies for losses on the validation set of IDD

Loss	Iterations	aAcc	IoU	mAcc
Cross Entropy Loss	160000	84.7400	34.9300	42.5600
Weighted Cross Entropy Loss	320000	85.7000	39.7000	48.9500
Dice Loss	416000	84.3400	31.2300	38.1900

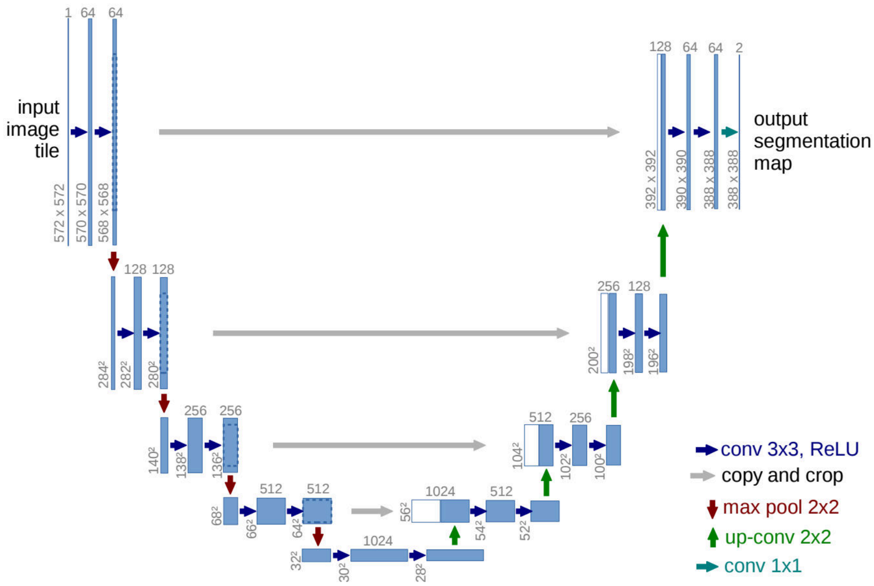


Figure 4.1: U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations [4].

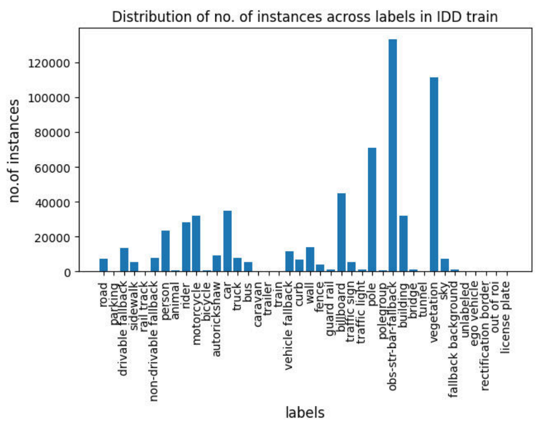
The three loss functions considered are Cross Entropy Loss, Weighted Cross Entropy Loss and Dice Loss. These Loss functions are discussed in chapter 3.

We analyze the presence and distribution of small objects within the IDD dataset, starting with the object and category distributions. The distribution of objects across labels in the IDD dataset is illustrated in figures 4.2 to 4.5. Additionally, we examine the pixel distribution across various categories. Figures 4.4 and 4.5 depict the instance and pixel distributions across labels for small, medium, and large object size categories in the IDD dataset.

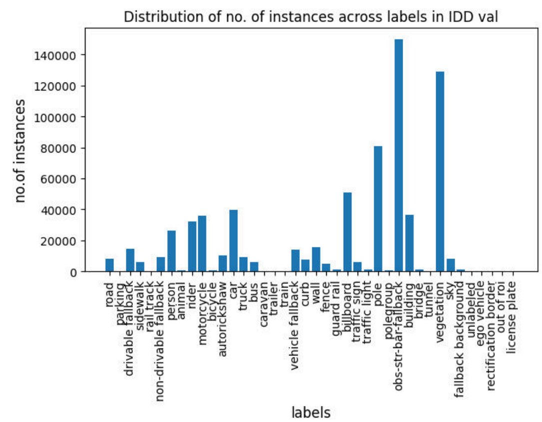
The choice of a loss function can significantly influence the results obtained from a dataset. This choice often depends on both the characteristics of the dataset and the specific problem being addressed. In the case of semantic segmentation of small objects, both factors are crucial. To systematically evaluate performance, we experiment with multiple loss functions, starting with Cross Entropy Loss and Weighted Cross Entropy Loss. Cross Entropy Loss is suited for datasets with balanced class distributions, while Weighted Cross Entropy Loss is more effective for datasets with class imbalances. Additionally, we examine the performance of another widely used loss function — Dice Loss.

4.1.2 Experiments

Training has been conducted using the Indian Driving Dataset (IDD) with U-Net as the backbone and Fully Convolutional Network (FCN) as both the decode head and auxiliary head. The IDD provides

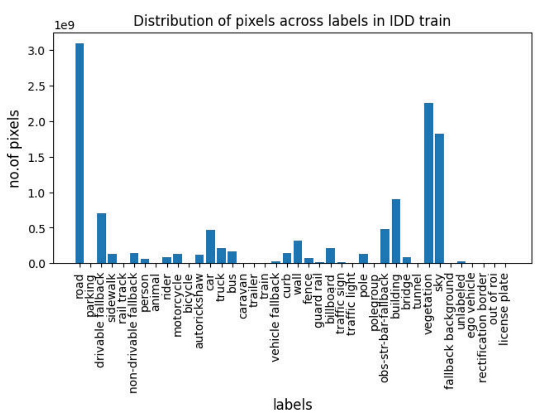


(a)

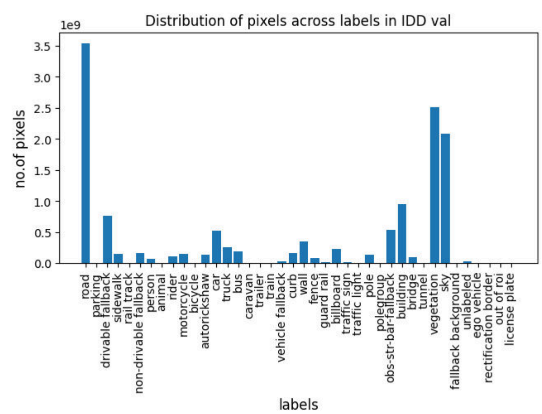


(b)

Figure 4.2: (a) Distribution of instances across labels in IDD train (b) Distribution of instances across labels in IDD val



(a)



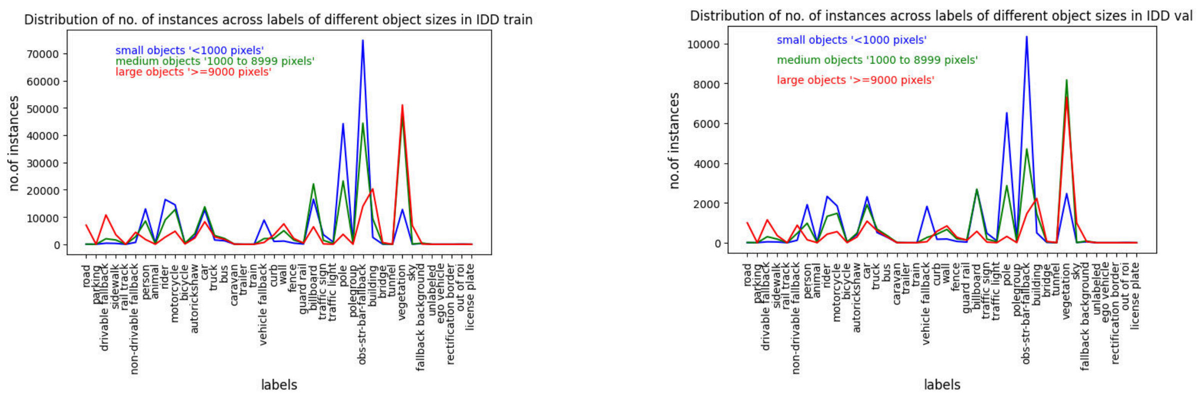
(b)

Figure 4.3: (a) Distribution of pixels across labels in IDD train (b) Distribution of pixels across labels in IDD val

pixel-level labeled images for training. The images in the dataset come in two resolutions: 1920×1080 and 1280×720. Experiment has been conducted with three different loss functions—Cross Entropy Loss, Weighted Cross Entropy Loss, and Dice Loss—and observations have been documented.

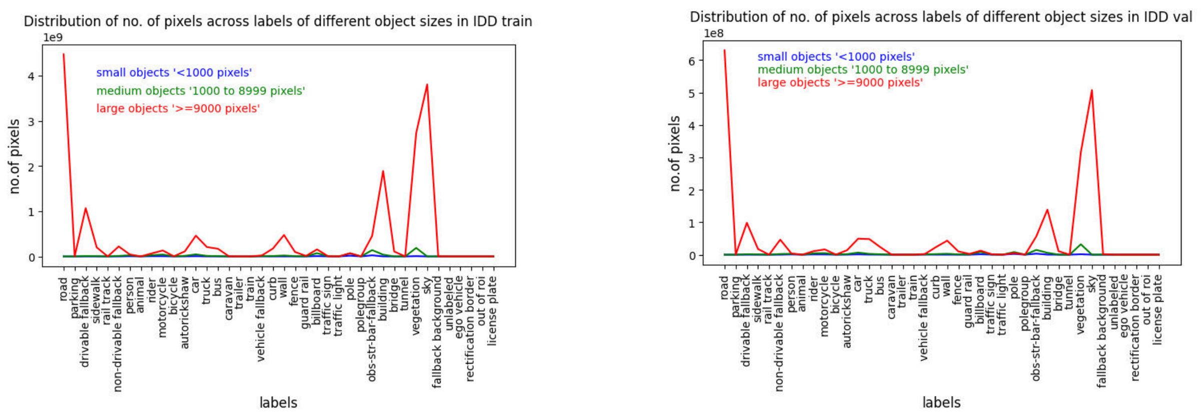
To ensure uniform image dimensions during training, all images are resized to a standard size of 1024×512, with a crop size of 512×256. The training utilizes the SGD optimizer, with an initial learning rate of 0.01, a momentum coefficient of 0.9, and a weight decay coefficient of 0.0005. A polynomial learning rate schedule is applied, with a minimum learning rate of 0.0001 and a power factor of 0.9.

The performance graphs for the three loss functions—Cross Entropy Loss, Weighted Cross Entropy Loss, and Dice Loss—are shown in Figures 4.6 to 4.8. Objective performance is evaluated using the



(a) (b)

Figure 4.4: (a) Distribution of instances across labels and Objsize in IDD train (b) Distribution of instances across labels and Objsize in IDD val



(a) (b)

Figure 4.5: (a) Distribution of pixels across labels and Objsize in IDD train (b) Distribution of pixels across labels and Objsize in IDD val

mean Intersection over Union (mIoU). Table 4.1 presents the mIoUs for all 40 classes in the IDD dataset, while Table 4.2 summarizes aggregate accuracy, mean IoU, and mean accuracy for the three loss functions

4.1.3 Overview of Figures

The small objects of significance in the IDD dataset [8] include persons, animals, riders, motorcycles, autorickshaws, cars, trucks, buses, traffic signs, and traffic lights. Below is a brief summary of the results presented in the study:

- Figure 4.2: Shows the distribution of the number of instances of labels in the training and validation datasets across all labels.

- Figure 4.3: Shows the distribution of the number of pixels of labels in the training and validation datasets across all labels.

- Figure 4.4: Shows the distribution of the number of instances of labels in the training and validation datasets for small, medium, and large object sizes.

- Figure 4.5: Shows the distribution of the number of pixels of labels in the training and validation datasets for small, medium, and large object sizes.

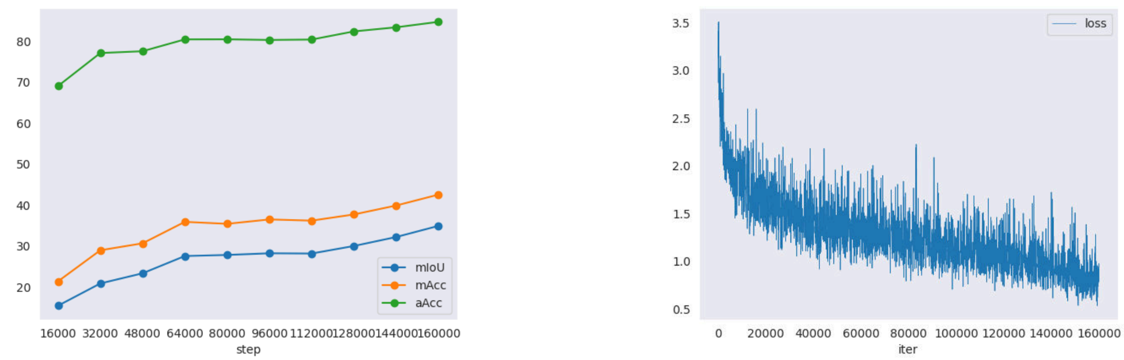
- Figures 4.6, 4.7, 4.8: Display the accuracies and losses for the three loss functions (Cross Entropy Loss, Weighted Cross Entropy Loss, and Dice Loss) across iterations.

- Figures 4.9, 4.10, 4.11: Contain three example images from the IDD dataset along with the predictions generated using Dice Loss for these images.

4.1.4 Discussion of Results

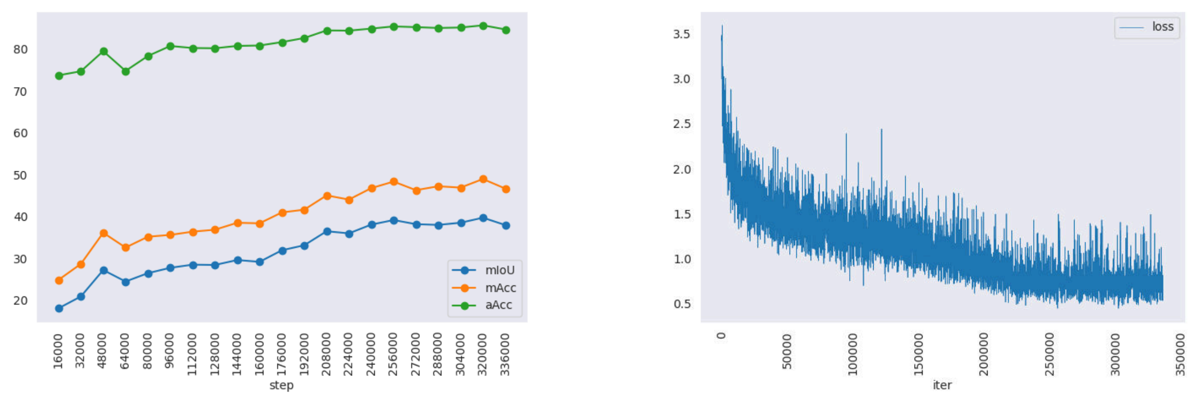
The three loss functions—Weighted Cross Entropy Loss, Cross Entropy Loss, and Dice Loss—perform similarly well on the rider, motorcycle, autorickshaw, car, truck, and bus classes. This can be attributed to the distribution of these classes as observed in Figures 4.2, 4.3, 4.4, 4.5. These categories have a significant number of instances and pixels in the dataset (Figures 4.2, 4.3). While these classes are well-represented in smaller object sizes (Figure 4.4), they contribute fewer total pixels when considered as smaller objects (Figure 4.5).

Weighted cross entropy loss, and cross entropy loss perform well on the person class but Dice loss gives a 0 mIoU. Cross entropy loss does not perform as well as weighted cross entropy Loss on traffic sign. The three losses perform equally poorly with a 0 mIoU on animal, bicycle, and traffic light classes. The reason why Dice Loss performs poorly with a 0 mIoU on person class may be observed from the Figures 4.2, 4.3 where the instances are considerable but the total pixels are less. Since the total pixels are less in person class, the Dice loss performs poorly as it is a Region level loss and in the initial stages of training, any wrong prediction of the region due to a lesser total number of pixels for the class may cause the future stages of training to be affected. On the contrary, cross entropy loss and weighted cross entropy loss are pixel level losses, and hence they have a chance for moving towards the correct class prediction at each pixel level giving a much better mIoU for the person class. A similar reason, may be attributed to the poor mIoU for dice loss over traffic sign class. Relatively poorer performance of cross entropy loss over weighted cross entropy loss can be attributed to the imbalance in the dataset. Animal, bicycle and traffic light classes have poor mIoUs for all the three losses in discussion as they have a lesser number of instances and total pixels(see Figures 4.2, 4.3).



(a) (b)

Figure 4.6: (a) Accuracies Plot for Cross Entropy Loss for 160000 iterations (b) Loss Plot for Cross Entropy Loss for 160000 iterations



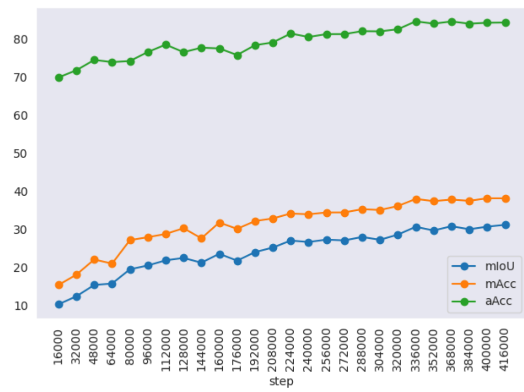
(a) (b)

Figure 4.7: (a) Accuracies Plot for Weighted Cross Entropy Loss for 336000 iterations (b) Loss Plot for Weighted Cross Entropy Loss for 336000 iterations

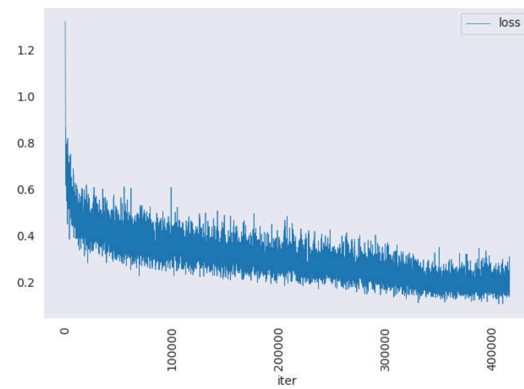
Figures 4.9, 4.10 4.11 illustrate predictions using Dice Loss for three sample images. Notably, the person class is often misclassified, whereas the rider class—also part of the broader human category—is segmented with a higher mIoU. The disparity in performance between the rider class and the person class, despite both belonging to the same category, raises an intriguing question. Understanding why Dice Loss achieves better segmentation for the rider class compared to the person class could be considered for future research.

Our analysis and experiments yield the following key observations:

- Indian outdoor environments, particularly road settings, contain a significant number of small objects, making the problem uniquely challenging.



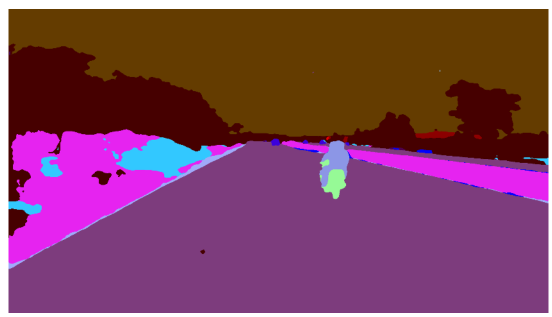
(a)



(b)

Figure 4.8: (a) Accuracies Plot for Dice Loss for 416000 iterations (b) Loss Plot for Dice Loss for 416000 iterations

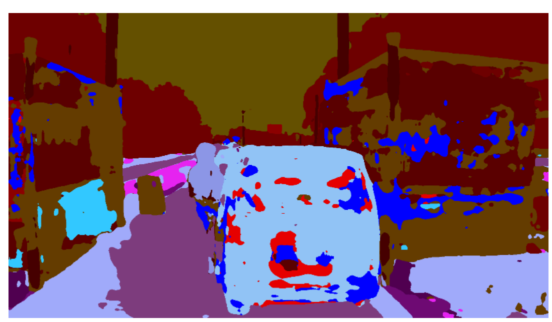
(a)



(b)

Figure 4.9: I Qualitative result of Dice Loss- Image and Prediciton

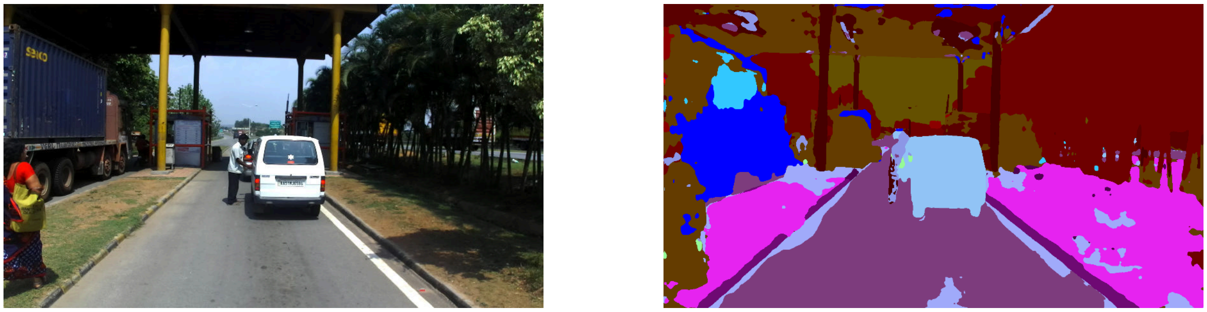
(a)



(b)

Figure 4.10: II Qualitative result of Dice Loss- Image and Prediciton

- Semantic segmentation of small objects is notably difficult, with average performance (mIoU) being significantly lower compared to larger objects.



(a) (b)

Figure 4.11: III Qualitative result of Dice Loss- Image and Prediciton

- In the context of driving and road safety, small objects hold critical importance, warranting dedicated future efforts to improve their segmentation.
- The performance of semantic segmentation is highly influenced by the choice of loss function, highlighting the need for further research into designing specialized loss functions or strategies to achieve robust segmentation of road scenes.

4.2 Object Detection

4.2.1 Methods

We employ a CNN-based object detection architecture. We employ the Faster-RCNN [27] (Figure 4.12) model which has two modules - RPN(Region Proposal Network) module which proposes regions and Fast-RCNN [26]module which uses the proposed regions.

The backbone considered is Resnet-50 [12]. Resnet-50, a residual network, uses residual connections which are identity mappings on a plain network. A Feature Pyramid Network(FPN) [35] is used as a neck.

The loss functions considered are Cross Entropy Loss for classification and L1 loss for regression.

4.2.2 Experiments

Training has been conducted using the Indian Driving Dataset (IDD) with Resnet-50 as the backbone and Faster-RCNN for the object detection task. The IDD detection provides bounding box level images for training. The images in the dataset come in different resolutions like 1920×1080 and 1280×964. Experiment has been conducted with Cross Entrpoy loss and L1 loss as classification and regression losses respectively —and observations have been documented.

In the image data loading during training, scale factors of 2048x800 and 2048x1024 are used. The training utilizes the SGD optimizer, with an initial learning rate of 2e-05, a momentum coefficient of

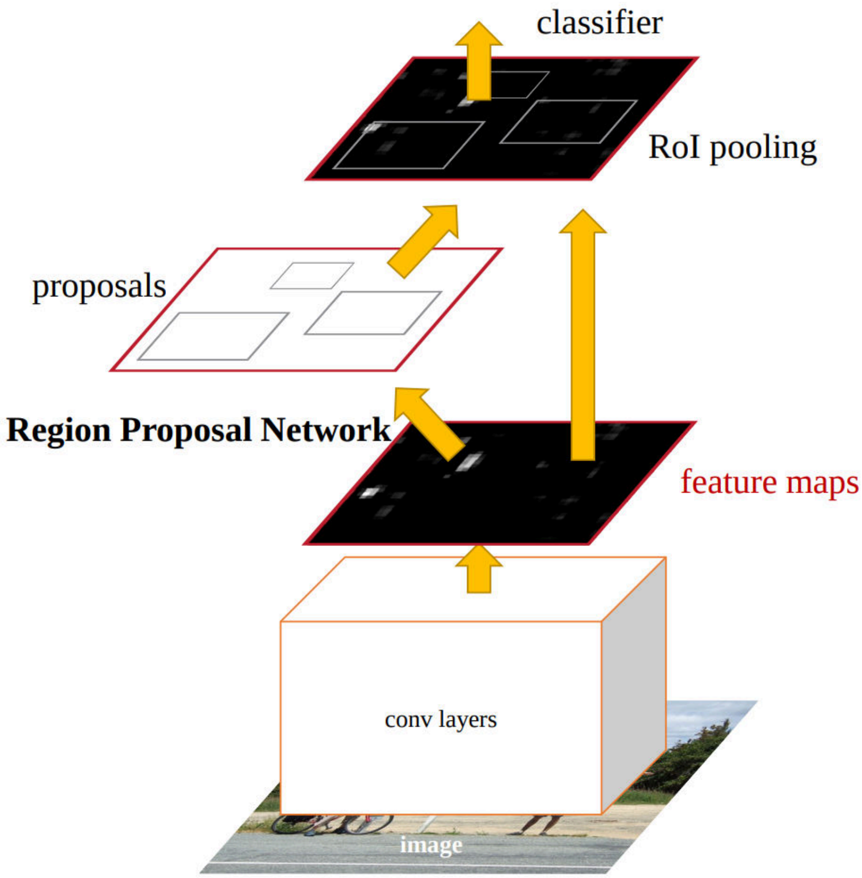


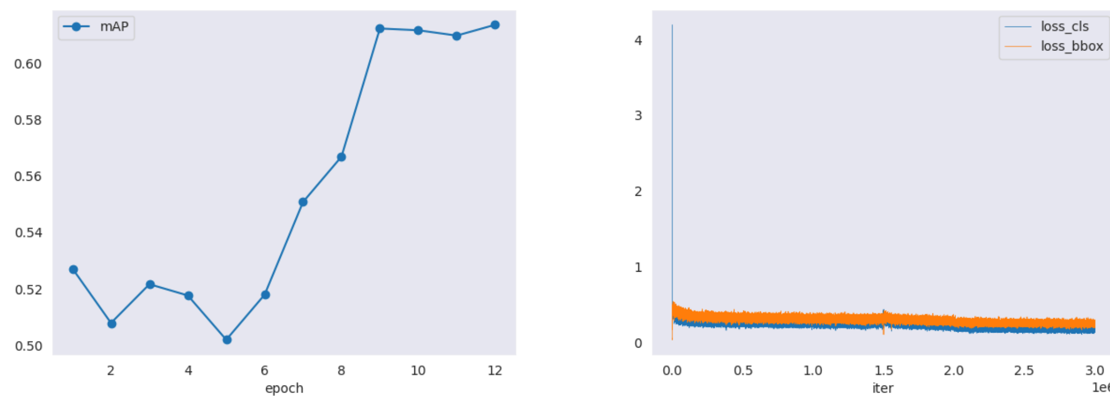
Figure 4.12: Faster R-CNN is a single, unified network for object detection [27].

0.9, and a weight decay coefficient of 0.0005. A linear learning rate schedule is applied with a start factor of 0.001, and a multistep learning rate is applied at epochs 8 and 11.

The performance graph for the experiment using Cross Entropy Loss andL1 loss for classification and regression respectively—is shown in Figure 4.14. Objective performance is evaluated using the mean Average Precision (mAP). Table 4.3 presents the recall, AP and mAP for all 11 classes in the IDD dataset for the parameters and losses specified.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 4.13: Resnet set of architectures. Resnet-50 can be seen [12].



(a) (b)

Figure 4.14: (a) mAP plot for 12 epochs (b) Loss Plot for Cross Entopy loss and L1 loss for 3e6 iterations

class	gts	dets	recall	ap
person	18078	50287	0.798	0.648
animal	1460	3515	0.453	0.332
rider	24518	48949	0.755	0.670
motorcycle	25489	46178	0.817	0.758
bicycle	569	2127	0.698	0.542
autorickshaw	7782	17615	0.875	0.782
car	24844	48170	0.832	0.766
truck	7078	19228	0.824	0.703
bus	4916	10908	0.853	0.764
traffic sign	4287	7948	0.563	0.434
traffic light	919	1452	0.430	0.351
mAP				0.614

Table 4.3: Evaluation results on IDD detection (gts = ground truths, dets = detections).

4.2.3 Discussion of Results

The loss functions-Cross entropy loss and L1 loss for classification and regression-perform well on all the 8 classes of IDD detection dataset but do not perform well on the three classes-animal, traffic sign and traffic light classes. This can probably be attributed to an imbalance in the dataset with respect to these three classes. The imbalance can be due to lesser instances. This imbalance can also be attribuited to lesser pixels bringing them under the ambit of smaller objects. However, the exact reasons can be known only after an analysis of the IDD detection dataset similar to the IDD segmentation dataset which can be considered for future work.

Chapter 5

Conclusion

5.1 Summmary

In this thesis, study was conducted on small objects in road situations and specifically on small objects in Indian Driving Dataset and their semantic segmentation. It also talks about Indian Driving Dataset and object detection.

Chapter 1 small objects commonly seen in road situations and practical issues on roads due to small objects and small objects in Indian Driving scenes. Contributions of the thesis have been highlighted.

Chapter 2 talks about related work corresponding to object detection, segmentation and small objects in detection and segmentation.

Chapter 3 discusses in detail the various loss functions. Three loss functions have been adopted from the extensive list of the discussed loss functions for study on small objects and Indian Driving Dataset segmentaion. One loss function from the extensive list has been adopted for IDD detection dataset.

Chapter 4 discusses the experiments conducted, qualitative and quantitative results on IDD segmentation dataset. Plots analyzing the Indian Driving Dataset have been discussed. The results of the three losses have been discussed. The reasons for performance and non-performance of losses have been discussed. The images inferred using a particular loss (Dice Loss) have been discussed conveying the reason behind the performance of that loss. The reason behind varying accuracies of the three losses across certain classes has been discussed. Further, the experiments conducted and quantitative results of the IDD detection dataset have been discussed. Probable reasons for non-performance across three classes have also been discussed.

5.2 Future Directions

There are many possible future directions for this thesis. We list some of them here:

- Design of novel loss functions that can help in segmentation and detection of small objects.
- Development of novel algorithms and architectures for solving perception problems (detection, segmentation, pose estimation etc.) of small objects.

- Development of a practical driver assistance system that pay special attention to small objects on the Indian road scenes.

Bibliography

[1] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[2] B. Mirzaei, H. Nezamabadi-pour, A. Raoof, and R. Derakhshani, "Small object detection and tracking: A comprehensive review," *Sensors*, vol. 23, no. 15, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/15/6887

[3] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[4] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds. Cham: Springer International Publishing, 2015, pp. 234–241.

[5] R. Azad, M. Heidary, K. Yilmaz, M. Hüttemann, S. Karimijafarbigloo, Y. Wu, A. Schmeink, and D. Merhof, "Loss functions in the era of semantic segmentation: A survey and outlook," 2023. [Online]. Available: https://arxiv.org/abs/2312.05391

[6] S. Jadon, "A survey of loss functions for semantic segmentation," in *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. IEEE, Oct. 2020. [Online]. Available: http://dx.doi.org/10.1109/CIBCB48159.2020.9277638

[7] Y. Liang, Y. Han, and F. Jiang, "Deep learning-based small object detection: A survey," in *Proceedings of the 8th International Conference on Computing and Artificial Intelligence*, ser. ICCAI '22. New York, NY, USA: Association for Computing Machinery, 2022, p. 432–438. [Online]. Available: https://doi.org/10.1145/3532213.3532278

[8] G. Varma, A. Subramanian, A. Namboodiri, M. Chandraker, and C. V. Jawahar, "Idd: A dataset for exploring problems of autonomous navigation in unconstrained environments," 2018. [Online]. Available: https://arxiv.org/abs/1811.10200

[9] M. Everingham, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge." *Int. J. Comput. Vis.*, vol. 88, no. 2, pp. 303–338, 2010. [Online]. Available: http://dblp.uni-trier.de/db/journals/ijcv/ijcv88.html#EveringhamGWWZ10

[10] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 740–755.

[11] S. Song, S. P. Lichtenberg, and J. Xiao, "Sun rgb-d: A rgb-d scene understanding benchmark suite," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 567–576.

[12] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," 06 2010, pp. 3485–3492.

[13] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[14] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3061–3070.

[15] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009, video-based Object and Event Analysis. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167865508001220

[16] G. J. Brostow, J. Shotton, J. Fauqueur, and R. Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Computer Vision – ECCV 2008*, D. Forsyth, P. Torr, and A. Zisserman, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 44–57.

[17] B. Leibe, N. Cornelis, K. Cornelis, and L. Van Gool, "Dynamic 3d scene analysis from a moving vehicle," 06 2007.

[18] M. Cordts, T. Rehfeld, M. Enzweiler, U. Franke, and S. Roth, "Tree-structured models for efficient multi-cue scene labeling," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, pp. 1–1, 01 2016.

[19] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 5122–5130.

[20] G. Neuhold, T. Ollmann, S. R. Bulò, and P. Kontschieder, "The mapillary vistas dataset for semantic understanding of street scenes," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5000–5009, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:5753855

[21] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," 05 2018.

[22] M. Y. Choong, W. L. Khong, W. Y. Kow, L. Angeline, and K. T. K. Teo, "Graph-based image segmentation using k-means clustering and normalised cuts," in *2012 Fourth International Conference on Computational Intelligence, Communication Systems and Networks*, 2012, pp. 307–312.

[23] H. Krishna and C. Jawahar, "Improving small object detection," in *2017 4th IAPR Asian Conference on Pattern Recognition (ACPR)*, 2017, pp. 340–345.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[25] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 11 2013.

[26] R. Girshick, "Fast r-cnn," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448.

[27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, 06 2015.

[28] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[29] P. Jiang, D. Ergu, F. Liu, Y. Cai, and B. Ma, "A review of yolo algorithm developments," *Procedia Computer Science*, vol. 199, pp. 1066–1073, 2022, the 8th International Conference on Information Technology and Quantitative Management (ITQM 2020, 2021): Developing Global Digital Economy after COVID-19. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050922001363

[30] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Computer Vision – ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham: Springer International Publishing, 2016, pp. 21–37.

[31] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, *End-to-End Object Detection with Transformers*, 11 2020, pp. 213–229.

[32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 10 2020.

[33] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10 778–10 787, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:208175544

[34] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, " Swin Transformer: Hierarchical Vision Transformer using Shifted Windows ," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2021, pp. 9992–10 002. [Online]. Available: https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.00986

[35] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.