Nicolas Abboud
202473009

Semantic Scene Understanding in Autonomous Robotics

Project Report
MECH798M

Table of Contents:

Abstract-

This project report outlines the development and evaluation of a semantic segmentation model for enhancing scene understanding in autonomous robotics. The study utilizes deep learning techniques, particularly convolutional neural networks, to enable robust semantic scene understanding. This research leverages a synthetic dataset of common assets in an industrial robotics environment to demonstrate the challenges of training models exclusively on synthetic data. The findings demonstrate the effectiveness of the developed model in handling synthetic images while acknowledging the limitations faced with real-world data, suggesting a future direction that involves integrating multi-source data to enhance model generalizability and robustness.
The project implementation is hosted on this public repository: github.com/8bit-nyk/my-semantic-segmentation/

Introduction

Semantic scene understanding and segmentation are crucial technologies in autonomous robotics, enabling robots to perceive and interact with their environments intelligently. This capability is essential for applications such as autonomous driving, aerial surveillance, and robotic navigation in complex, dynamic settings. In this domain, there has been a significant shift towards utilizing deep learning methods for enhanced accuracy and efficiency. The evolution of these approaches ranges from traditional feature extraction techniques to the latest advancements in deep learning-based methods, emphasizing computational efficiency to meet the real-time processing requirements of autonomous systems.

In this report we present a scientific endeavor to experiment with semantic segmentation in order to understand and integrate semantic understanding within a monocular visual SLAM (Simultaneous Localization and Mapping) system and advance research on perception and mapping in autonomous robotics and digital twin applications. We begin with a brief introduction on semantic image segmentation and its evolution with the advent of deep-learning approaches. Furthermore, we present the methodology, experimentation and findings accrued by training and deploying a basic semantic image segmentation model.

Semantic segmentation refers to the process of partitioning an image into distinct regions associated with specific labels. Unlike object detection, which identifies bounding boxes around objects, semantic segmentation classifies each pixel of an image, resulting in a pixel-wise dense prediction map. This technique allows for a fine-grained

Figure 1- Example of a pixel-wise classification mask

understanding of the scene, which is pivotal in applications where precise spatial understanding and localization are required.

Initially, scene understanding relied on parts-based representations for object description and rule-based image processing techniques, involving thresholding, k-means clustering, and conditional random fields.[1] Traditional feature representation methods like Scale-Invariant Feature Transform (SIFT), GIST, and Latent Semantic Analysis (LSA) were prominent. These methods, while foundational, lacked the ability to learn from data and adapt to complex scenarios instead they focused on extracting low-level and middle-level visual features to achieve semantic classification through feature modeling [2].

The introduction of neural networks brought significant improvements. Early neural network applications in scene understanding were limited but set the stage for more complex architectures. The rise of Convolutional Neural Networks (CNNs) marked a pivotal transition to deep learning-based methodologies, which could adaptively extract features and perform various image recognition tasks such as object detection or semantic segmentation. Unlike traditional methods that required manually designed features, CNNs learn higher-level abstract representations of data automatically, significantly improving performance in scene understanding tasks [2] [3]. Fully Convolutional Networks (FCNs) accelerated the shift from image classification to pixel-level segmentation by replacing fully connected layers with convolutional layers to output spatial maps instead of classification scores, which was a significant milestone for scene understanding tasks. Deep learning models, especially those leveraging CNNs, have dominated the field, with architectures like U-Net and SegNet specifically designed for semantic segmentation, offering improved performance by effectively capturing spatial hierarchies and context. Recently, unsupervised learning has garnered attention for its potential to alleviate the need for large, annotated datasets. Techniques like mutual information maximization combined with adversarial regularization have shown promise in unsupervised image segmentation, aiming to cluster semantically similar pixels or superpixels without explicit supervision. This approach not only aids in computational efficiency but also opens new possibilities for learning from unlabeled data [4].

For real-time applications in autonomous robotics the need for computational efficiency is paramount. Deep learning methods have evolved to incorporate strategies that reduce computational overhead while maintaining or even enhancing the accuracy of scene understanding tasks. This includes the use of lighter network architectures, efficient data processing pipelines, and optimizing model training to leverage limited computational resources effectively. In summary, the journey from traditional feature extraction methods to advanced deep learning approaches has significantly enriched the field of semantic scene understanding in autonomous robotics. The focus on computational efficiency, coupled with the exploration of unsupervised learning techniques, signifies a progressive shift towards creating more autonomous, reliable, and efficient robotic systems capable of complex scene interpretation.

Methodology

Overview

Deep learning based semantic segmentation models input images along with their corresponding semantic labels into a convolutional neural network. The network initially processes these inputs through a series of down-sampling steps, such as convolutions and pooling layers. This allows the network to learn a one-dimensional latent representation of the input image, which captures the essential features needed for effective segmentation. Subsequently, the network up-samples this one-dimensional representation back to the original input size using bi-linear interpolation or similar techniques, aiming to reconstruct the spatial dimensions necessary for pixel-level classification. Throughout the learning process, the model is optimized by minimizing the discrepancies between the predicted segmentation maps and the actual labeled images, thus ensuring the model accurately captures the semantics of the scenes.

Data Evolution

In deep-learning based segmentation models the features extracted typically evolve from general to specific as the data moves through the network. Below is a detailed description of how data evolves through such networks:

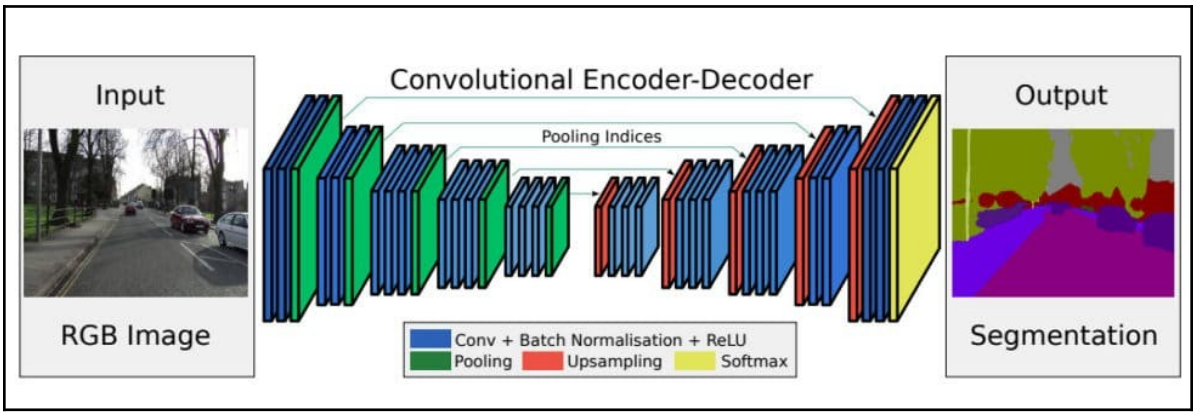


Figure 2- Overview of the flow of data in a basic semantic segmentation model.

The input to a semantic segmentation network is generally a standard color image (RGB), where each pixel in the image serves as input data with three feature values (red, green, blue intensities). The first few layers of the network generally extract low-level features such as edges, colors, and textures. These are achieved through the application of various learned filters to the input image, producing a set of feature maps. These feature maps still closely resemble the original input image in layout, where each point in the feature map corresponds to a region in the input image. Interspersed among the convolutional layers, pooling layers reduce the spatial dimensions (width and height) of the input volume for the next convolutional layer. They are used to decrease the computational load, memory usage, and number of parameters.

Pooling layers also help in making the detection of features invariant to scale and orientation changes.

As we move deeper into the network, the filters begin to detect higher-level features like shapes and specific textures. These features become more abstract and less visually interpretable compared to those in earlier layers. The spatial dimensions of the feature maps reduce further due to additional pooling layers, concentrating the information into more abstract representations. In the deepest layers, the network begins to assemble the more abstract features into high-level features, which might correspond to specific objects or parts of objects in the scene. For example, these might be features that specifically identify cars, trees, or roads in the context of autonomous driving.

For semantic segmentation, the spatial resolution of the output needs to match the original image size. Therefore, networks often include upsampling layers toward the end. These layers increase the dimensions of the feature maps. Additionally, in architectures like U-Net, skip connections are used to concatenate feature maps from earlier layers with the upsampled feature maps. This helps to preserve spatial hierarchies and details that are crucial for pixel-level classification. The final layers of a semantic segmentation network typically consist of 1x1 convolutions that reduce the number of feature maps to the number of classes in the segmentation task. This is followed by a SoftMax or sigmoid activation layer that assigns each pixel in the output image a probability of belonging to a certain class.

Throughout this process, the input image is transformed from a raw pixel format to a complex feature-rich representation that captures the presence of different semantic regions within the image along with their corresponding boundaries. Each layer's output can be viewed as a more refined understanding of the image, where the network learns to pay attention to increasingly abstract and relevant patterns.

Deep Learning Models

Overview

The field of semantic scene understanding has seen the development of various models, each with its unique approach to tackling the challenges of image segmentation and classification. We present an overview of some key models:

Fully Convolutional Networks (FCNs) were among the first deep learning models to be applied successfully to semantic segmentation. Unlike traditional CNNs, which contain fully connected layers, FCNs consist entirely of convolutional layers. This design allows FCNs to take input images of arbitrary sizes and output segmentation maps corresponding to the input dimensions. FCNs achieve this by replacing fully connected layers with convolutional layers, enabling the network to retain spatial information throughout the network.

U-Net is well-regarded for its effectiveness in image segmentation, featuring a U-shaped architecture. It has a contracting path to capture context and a symmetrically expanding path that enables precise localization. U-Net utilizes skip connections between corresponding layers in the contracting and expanding paths, which helps in recovering fine-grained details in the segmentation map.

Mask-RCNN builds on the Faster-RCNN framework by adding a branch for predicting segmentation masks on each region of interest, parallel to the existing branch for classification and bounding box regression. This design enables Mask-RCNN to perform both object detection and instance segmentation simultaneously otherwise know as panoptic segmentation.

DeepLab models utilize dilated convolutions to capture multi-scale information and broaden the field of view of filters. It incorporates an encoder-decoder structure, enhancing its ability to capture sharper object boundaries. The model also employs spatial pyramid pooling to effectively segment objects at multiple scales.

PSPNet (Pyramid Scene Parsing Network) addresses the need for effective global contextual information in scene parsing tasks. It introduces a pyramid pooling module that aggregates context information at different scales and locations. It is particularly adept at scene parsing tasks where integrating information from different scales is crucial to maintain the consistency and accuracy of the segmentation.

SegNet is known for its efficiency in terms of computational cost and memory usage, making it suitable for applications that require real-time processing. It uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear upsampling in the decoder, efficiently using the network parameters and reducing the model size.

Recent years have seen the application of **Transformer** models, originally developed for natural language processing, to scene understanding. These models, such as **ViT (Vision Transformer)**, have shown promising results in capturing global dependencies without being bound to the local receptive fields of CNNs. The integration of attention mechanisms in transformers has allowed for better context capture and focus on relevant parts of the scene for more detailed understanding. This shift towards using attention is seen as pivotal in handling the complexities of scene understanding. Moreover, combining CNNs with Transformers has produced state-of-the-art results in various scene understanding tasks. These hybrid models leverage both local features captured by CNNs and global context captured by Transformers, providing a comprehensive understanding of scenes.

Proposed Model

Given our focus on autonomous robots and their need for robust and efficient scene understanding capabilities, we propose the use of models like EfficientNet or MobileNets; they provide a good balance between accuracy and performance, making them suitable for deployment in robots with limited processing capabilities.

EfficientNet is a family of convolutional neural networks that systematically scales up CNNs in a more structured manner. The core idea behind EfficientNet is to scale the network depth, width, and resolution uniformly using a simple yet effective compound coefficient. This approach, based on a fixed set of scaling rules, optimizes the trade-offs between the different dimensions through which CNNs can be scaled, such as the depth of the layers, the width of the channels, or the resolution of the input images. This method allows EfficientNet to achieve much higher accuracy and efficiency than other CNNs, making it highly effective for tasks that require processing large amounts of visual data with limited computational resources.

Our model employs a popular CNN based approach, the **Unet++** architecture as the base model, however we modify the encoder backbone to an EfficientNet variant creating a more efficient and scalable model. The model inputs the 3 color channels (RGB) representing an image and outputs, in the last layer of the network, the number of classes in the dataset. For a binary segmentation case, this last layer would typically be a single channel with a sigmoid activation. Whereas, for multi-class segmentation, it's a SoftMax over the number of specified classes. When considering an appropriate loss function for segmentation, common choices are Cross-Entropy Loss or Dice Loss for their ability to handle different aspects of segmentation effectively. Cross-Entropy Loss is favored for its robustness in dealing with multi-class problems and class imbalance, while Dice Loss is preferred for its emphasis on improving the overlap between the predicted and actual segments, making it ideal for binary segmentation tasks. Additionally, an Adam optimizer is generally employed in this case for its adaptive learning rates and efficiency with sparse gradients. Moreover we experimented with fine tuning two hyper-parameters, the learning rate and batch size, using a simple grid search algorithm.

A detailed description of the implementation procedure follows:

Implementation

Dataset Description and Visualization

COCO dataset

When initially planning the project, we were going to use the **COCO (Common Objects in Context)** dataset, which is extensively recognized for its rich annotations and variety in image content, making it an ideal choice for training and evaluating models aimed at semantic scene understanding. However, we changed course after submitting the project milestone to a more concise and interesting dataset. The details will be presented shortly after we present a quick overview of COCO.

The COCO dataset is an extensive dataset for image recognition used for tasks like object detection, semantic segmentation, dense pose estimation, and panoptic segmentation. It

Figure 3- Images of different object classes extracted from the COCO dataset. The segmentation masks are overlayed over each image.

comprises over 330,000 images featuring complex everyday scenes with common objects in their natural context. Each image is labeled with 80 object categories and 5 descriptions of the scene. Since COCO is so extensive, diverse and huge, training on the whole dataset would have been a needlessly time-consuming endeavor especially when using modest hardware. Additionally, the popularity of the use of this dataset across different image recognition tasks means that such training has been exhaustively done for various types of research. Hence we decided to move to a more niche dataset that has potential for future meaningful contribution to the existing body of research.

IdealWorks Synthetic Dataset

The dataset used for this project was available to us through the collaboration of the Vision and Robotics Lab at AUB with Idealworks GMBH. The dataset used is a small subset of the **Synthetic Object Recognition Dataset for Industries (SORDI)** generative AI dataset, developed by idealworks in partnership with NVIDIA Corp. and BMW group [5,6]. SORDI is the world's largest reference dataset for artificial intelligence in manufacturing, consisting of more than 800,000 photorealistic synthetic images of production resources in 80 classes and includes objects of particular relevance in the core technologies of automotive engineering and logistics.

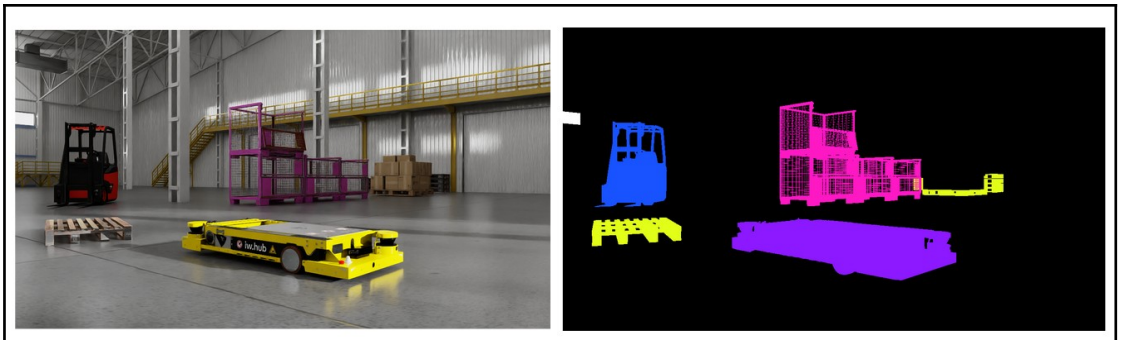
At our disposal we had a small subset of this extensive generative dataset that consists of 7 classes of objects belonging to an industrial automotive environment such as palette, dolly, stillage, forklifts, an industrial mobile robot known as the "iwhub" developed by idealworks and its power charging station. Each class contains 1000 data points consisting of RGB images, semantic masks and annotations (for semantic segmentation), bounding box coordinates and annotations (for object detection).

In the first stage of this project, we implement a preliminary binary classification model that classifies the iwhub mobile robot from its background. For that purpose, we only use the 1000 data points corresponding to the iwhub class as our dataset.

Dataset Structure and Preparation

The dataset comprised 1,000 sets of images, masks, and corresponding labels. Each set consisted of:

- RGB images.
- Mask images with colored masks for various classes.
- JSON files containing label information, detailing the presence of classes within each image and their corresponding color codes.



{"(0, 0, 0, 0)": {"class": "BACKGROUND"}, "(0, 0, 0, 255)": {"class": "UNLABELLED"}, "(25, 82, 255, 255)": {"class": "forklift"}, "(226, 255, 25, 255)": {"class": "pallet"}, "(255, 25, 197, 255)": {"class": "stillage"}, "(140, 25, 255, 255)": {"class": "iwhub"}}

Figure 4- Example of one datapoint consisting of an RGB image, its segmentation mask, and the class annotation.

Data Manipulation and Handling

The raw pixel values of images serve as the feature space for the model hence the data must be manipulated accordingly. We developed custom data loaders using PyTorch to handle the loading of images and their corresponding masks and labels. Transformations such as resizing, normalization, and data augmentation (flipping, rotation) were applied to prepare the images for training. As for the class label data, custom functions were implemented to convert RGB masks to binary masks for simple binary classification case (1 represents the object and 0 represents the background) and convert to class indexing for multi-classification case.

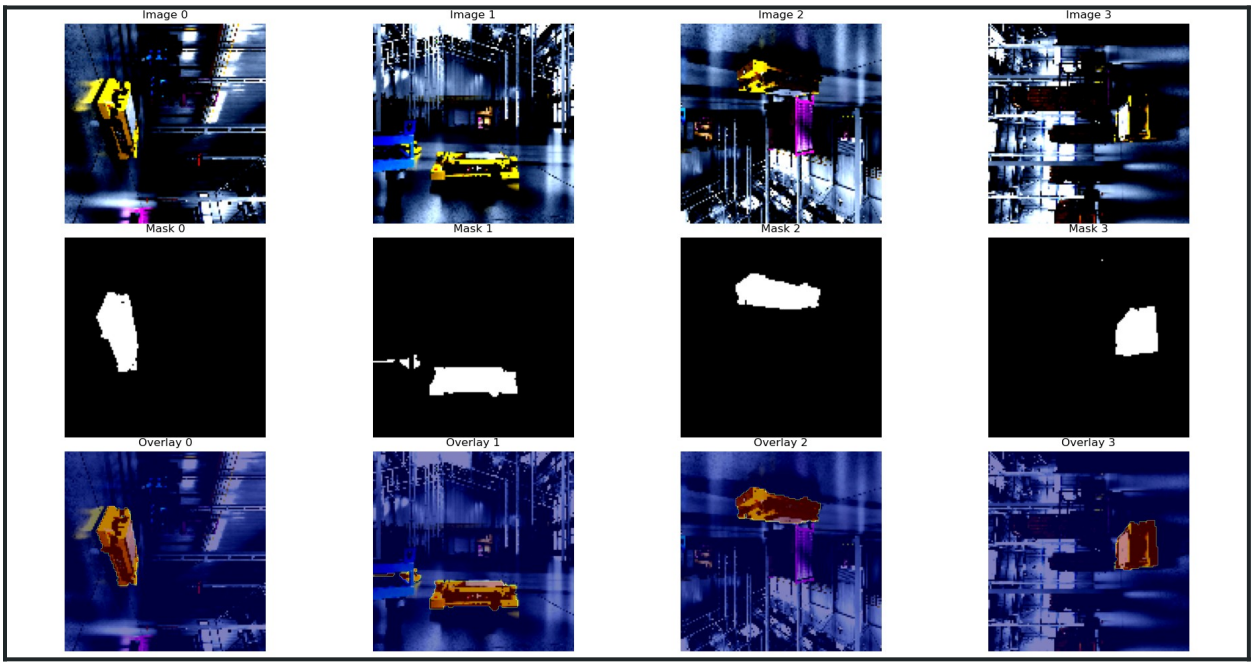


Figure 5- Images showing images, masks and overlays during data augmentation.

The data was split into training, validation, and test sets with a distribution of 80% for training and 20% for validation. Further, the training set was split into sub-training and test sets using a similar ratio to ensure thorough evaluation and avoid data leakage.

Model Training and Validation

As mentioned , a variant of U-Net architecture with a pre-trained EfficientNet backbone was employed due to its efficacy in handling semantic segmentation tasks. The model was trained over 20 and 40 epochs using the binary cross-entropy loss function for binary classification tasks, along with the Adam optimizer. The training process involved feeding batches of images and masks into the model, calculating loss, and updating the model weights accordingly. Post each training epoch, the model is evaluated on the validation set to monitor the performance and avoid overfitting.

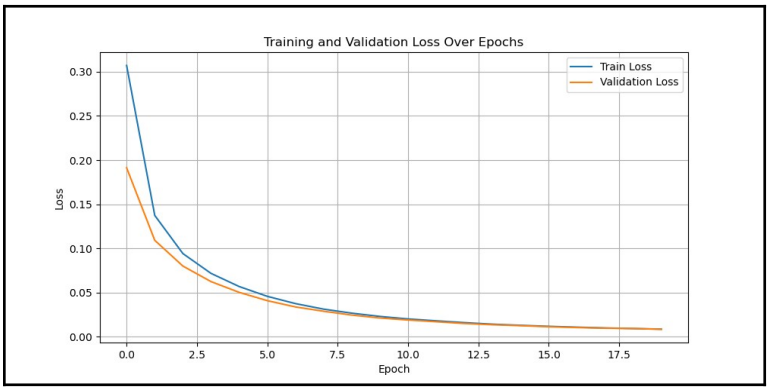


Figure 6-Plot showing training and validation loss over 20 epochs

Hyperparameter Tuning

We developed a separate script dedicated to tuning hyper parameters such as learning rate and batch size to optimize model performance. This was achieved using a grid search algorithm. The algorithm uses experimental iteration over a defined range of values, to find the best performing model which is saved for further evaluation. The experiment shows that a learning rate of 0.001 and a batch size of 2 are the optimal parameters, that yield the lowest validation loss.

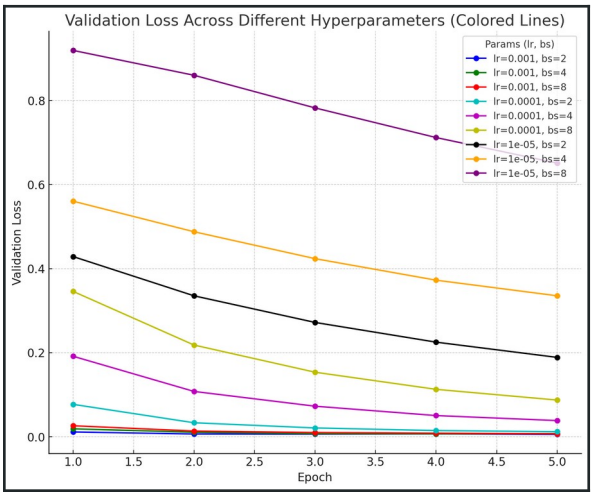


Figure 7- Plotting validation loss across different hyperparameters.

Evaluation and Retraining

The model's generalization capabilities were assessed using a test dataset not seen by the model during training. This helped in understanding the model’s performance in real-world scenarios. Based on the insights gained from initial training and testing, the model underwent retraining with adjusted hyper-parameters to enhance its accuracy and efficiency.

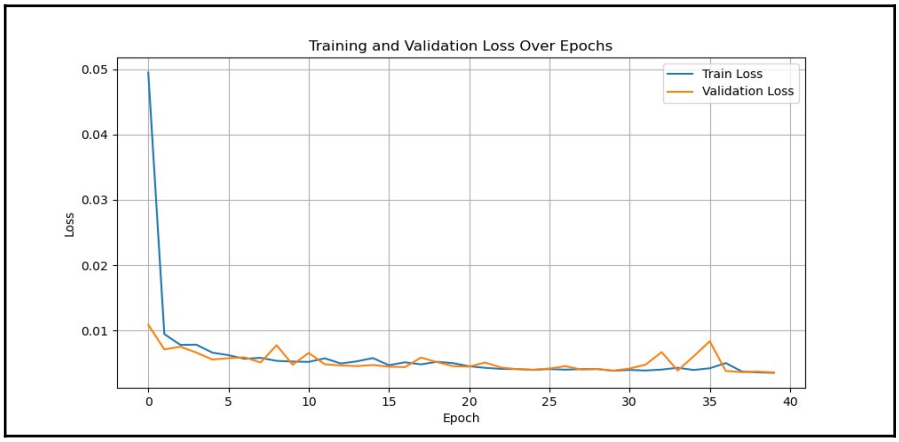


Figure 8-Plotting training and validation loss of finetuned model over 40 epochs.

The methodology implemented in this project ensured a comprehensive approach to developing a robust semantic segmentation model. From data preparation and handling through training regimes and hyper-parameter tuning, to thorough validation and testing — each step was aimed to optimize the model's performance.

Inference Results and Discussions

To validate the real-world applicability of this model we utilize the trained semantic segmentation model to predict and visualize the segmentation masks on new, unseen images. This allows us to assess the model's practical performance and effectiveness directly.

The pre-trained semantic segmentation model is loaded from its saved state and set to evaluation mode. The input images are read and transformed using the same methods (e.g., resizing and normalization) that were used during the training phase. This ensures consistency in how the data is handled, avoiding discrepancies that could affect model performance. The preprocessed data passes through the model to generate predictions. These predictions are segmentation masks that identify where specific classes of objects are located within the images. Predicted masks are processed to convert probabilities into binary masks using a thresholding technique. Finally, the predicted masks are overlaid on the original images to visually assess how accurately the model has segmented the objects in the image.

We initially tested the model on synthetically generated images that were not in the training or validation sets. The results are shown below, as visually evident the model performs well and segments the target object in each trial.

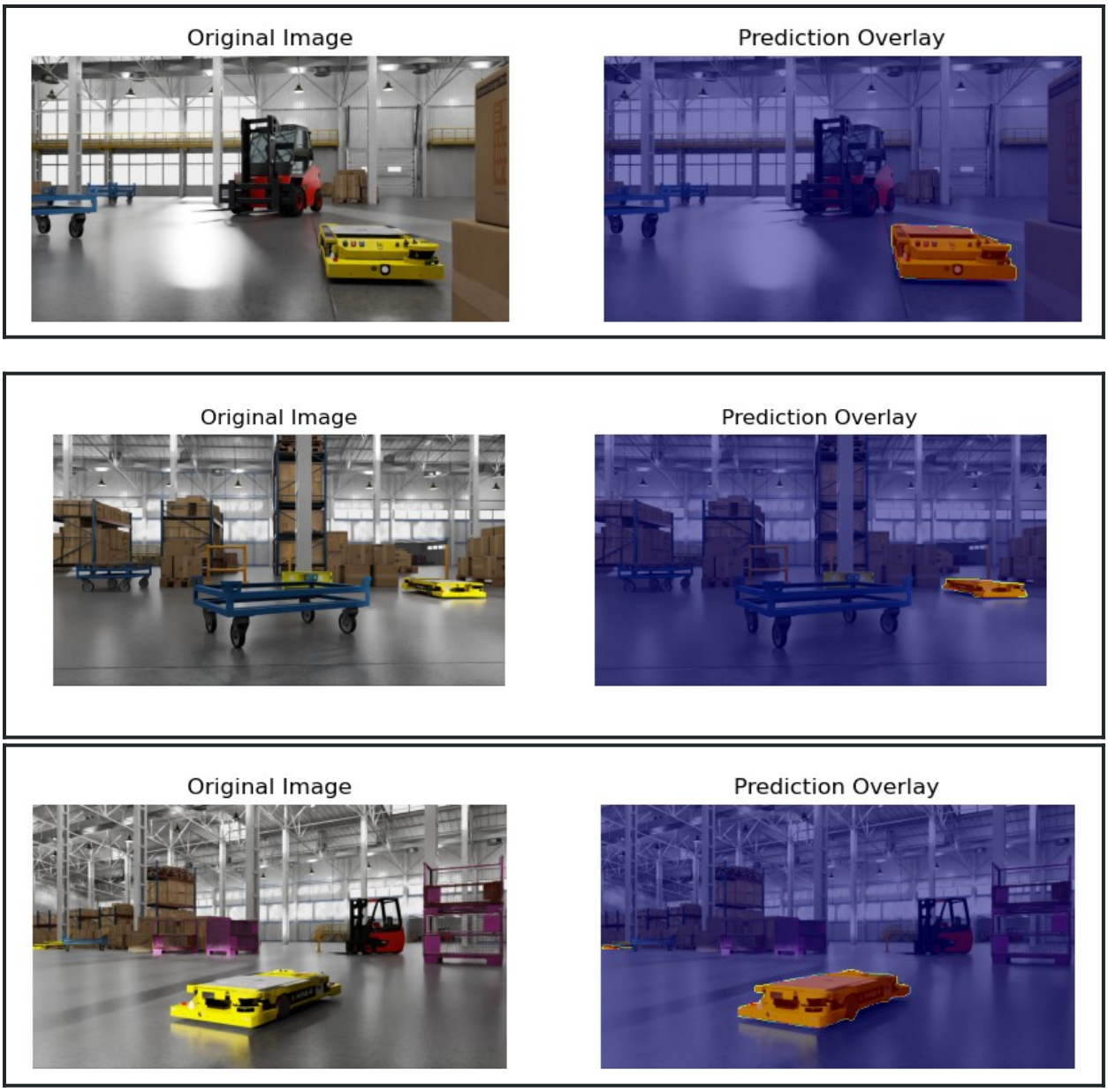


Figure 9- Results of inferencing on unseen synthetic data

To assess the generalizability of the model, we test the inference of the semantic model on real images of the target object scoured from the internet. The results are not optimal as shown in the sample results below.

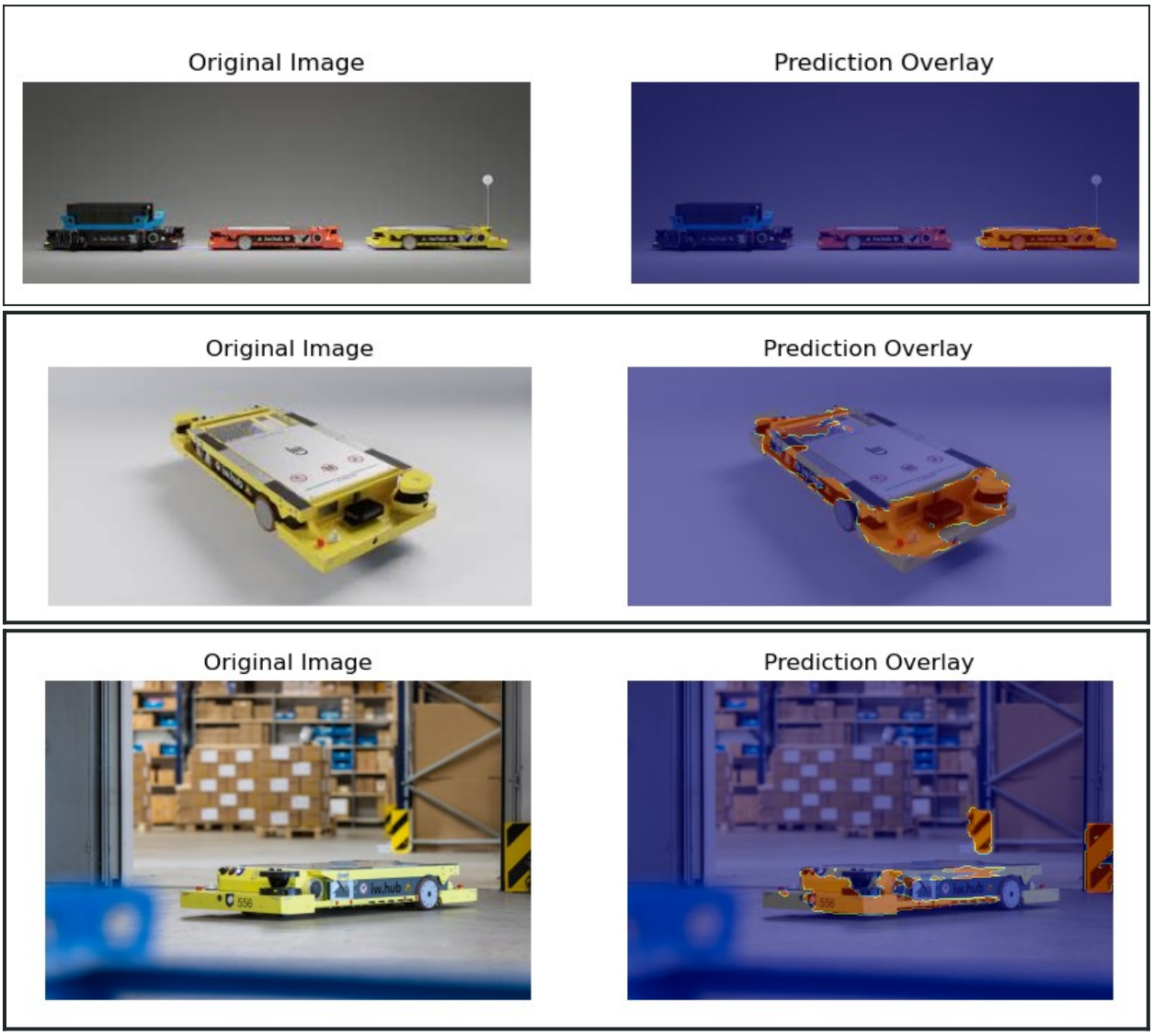


Figure 10-Results of inference on real images.

This finding aligns with state-of-the-art research concerned with the use of synthetic data in image classification tasks. Recent approaches promote the use of multi-source that is, both real and synthetic data to enhance model robustness and generalizability. Synthetic data allows for training on diverse scenarios, but often fails to capture real-world complexities, leading to a domain gap known as the “reality” gap. Integrating real-world data helps ground the model’s learning. This multi-source strategy is proven to improve the overall performance and generalizability of deep learning models. [7]

Conclusion and Future Work

In this project report we detailed the development and evaluation of a semantic segmentation model tailored for applications in autonomous robotics. The journey through the stages of implementing and refining a semantic scene understanding model provided profound insights into the challenges and capabilities of current deep learning technologies. We have illustrated the evolution from traditional feature engineering methods to advanced deep learning. We trained, validated, and fine-tuned a U-Net based architecture, with an EfficientNet as a backbone for semantic segmentation tasks.

The results obtained demonstrated the efficacy of the model in interpreting synthetic data. However, the challenges encountered when applying the model to real-world images highlighted the limitations of training exclusively on synthetic datasets. The discrepancy in performance underscores the importance of incorporating diverse data sources to bridge the “reality gap” and enhance model robustness.

Future enhancements to the project will focus on integrating multi-source data, including both real-world and synthetic datasets, to improve the generalizability and robustness of our semantic segmentation model. This approach will help the model close the “reality gap". Additionally, we plan to explore advanced data augmentation techniques and semi-supervised learning methods to optimize the use of available data. The ultimate goal is to integrate this enhanced model into a Visual SLAM system, aiming to boost the autonomous navigation capabilities of robotics by providing more accurate and real-time scene understanding in diverse settings.

References:

[1] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., & Terzopoulos, D. (2021). Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, *44*(7), 3523-3542.

[2] Ni, J., Chen, Y., Tang, G., Shi, J., Cao, W., & Shi, P. (2023). Deep Learning-based scene understanding for autonomous robots: a survey.

[3] Guo, Z., Huang, Y., Hu, X., Wei, H., & Zhao, B. (2021). A survey on deep learning based approaches for scene understanding in autonomous driving. *Electronics*, *10*(4), 471.

[4] Mirsadeghi, S. E., Royat, A., & Rezatofighi, H. (2021). Unsupervised image segmentation by mutual information maximization and adversarial regularization. *IEEE Robotics and Automation Letters*, *6*(4), 6931-6938.

[5] https://idealworks.com/en/news/en/largest-open-source-synthetic-dataset-for-ai-applications/

[6] https://sordi.ai/generative

[7] Eversberg, Leon, and Jens Lambrecht. "Combining Synthetic Images and Deep Active Learning: Data-Efficient Training of an Industrial Object Detection Model." *Journal of Imaging* 10.1 (2024): 16.