Politecnico di Torino

Politecnico di Torino

Master's Degree in Computer Engineering

Academic Year 2024/2025

Graduation Session October 2025

3D PERCEPTION OF DEFORMABLE OBJECTS IN ROBOTIC APPLICATION

Supervisor:

Prof. LIA MORRA

Candidate:

NASRIN RAHIMI ZADEH

Abstract

The manipulation of deformable objects, particularly garments, remains a significant challenge in robotics due to their infinite-dimensional state space and unpredictable dynamics. This thesis addresses the critical need for robust perception systems by designing, implementing, and evaluating an end-to-end vision pipeline for autonomous robotic handling of clothing. The research is contextualized within the VolPix project from EUROBIN, which targets the automation of laundry tasks involving ten distinct garment categories in both wet and dry states.

The proposed system employs a multi-stage approach to transform a single RGB image into actionable data for a robotic manipulator. The pipeline begins with instance segmentation to isolate individual garments from cluttered scenes, followed by object recognition to determine each item's category. Subsequently, a specialized keypoint detection module localizes semantic landmarks crucial for grasping and folding, and a final stage reconstructs the garment's 3D mesh using a monocular depth estimation technique. To train and validate these components, a custom dataset was collected and annotated, supplementing pretraining on the large-scale DeepFashion2 dataset.

This work establishes a comprehensive perception framework that integrates segmentation, recognition, keypoint detection, and 3D reconstruction, providing a strong foundation for advancing autonomous robotic manipulation of deformable objects.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Lia Morra, for her invaluable guidance, constant support, and encouragement throughout the course of this thesis. Her expertise and insightful feedback have been instrumental in shaping both the research and my academic growth.

I am sincerely thankful to Davide Tricarico and Daniele Scaffidi Gennarino, whose constructive discussions and technical assistance greatly enriched this work.

My appreciation also goes to my colleagues and friends, for their collaboration, motivation, and for making this journey more enjoyable.

A special thanks to my family - My mother, father, and brothers - for their unconditional love, patience, and encouragement. Without their constant support, this achievement would not have been possible.

Finally, I acknowledge AITEM company for providing the resources and environment that made this research feasible.

"Nasrin Rahimi Zadeh"

October 2025, Torino

Table of contents

List of Tables

List of Figures

Acronyms

2D / 3D
Two-Dimensional / Three-Dimensional

4D-Human
Four-Dimensional Human dataset

AP
Average Precision

BCE
Binary Cross-Entropy

BUFF
Bodies Under Flowing Fashion (dataset)

CLASP
CLoth Action Semantic Points

CNN
Convolutional Neural Network

COCO
Common Objects in Context (dataset)

DETR
DEtection TRansformer

FCN
Fully Convolutional Network

FLOP / FLOPs
Floating Point Operation / Floating Point Operations per second

FPN
Feature Pyramid Network

GAN
Generative Adversarial Network

GPU
Graphics Processing Unit

HRNet
High-Resolution Network

IoU
Intersection over Union

MSE
Mean Squared Error

MLP
Multi-Layer Perceptron

NMS
Non-Maximum Suppression

OKS
Object Keypoint Similarity

PCK
Percentage of Correct Keypoints

PIFu
Pixel-aligned Implicit Function

R-CNN
Region-based Convolutional Neural Network

ResNet
Residual Network

RPN
Region Proposal Network

ROI / RoIAlign
Region of Interest / Region of Interest Align

RGB / **RGB-D**
Red, Green, Blue / Red, Green, Blue + Depth

SAM
Segment Anything Model

SMPL
Skinned Multi-Person Linear Model

ViT
Vision Transformer

YOLO
You Only Look Once

Chapter 1

1 Introduction

1.1 Motivation

The field of modern robotics is rapidly expanding from structured industrial settings into the unstructured environments of everyday human life, where robots must handle a far greater diversity of objects. In particular, the manipulation of **deformable objects** (such as cloth, garments, and other soft materials) presents unique challenges not encountered with rigid bodies. Unlike rigid objects, which can be described by a small number of pose parameters, textiles and garments have an effectively infinite-dimensional state space with complex, non-linear dynamics[1]. Even small forces can cause large and unpredictable shape changes, and garments readily fold, bend, or wrinkle in ways that occlude parts of the object. These characteristics (high sensitivity to external forces, frequent self-occlusion, and dramatic topology changes) make deformable objects fundamentally difficult to perceive, model, and handle reliably[2]. In computer vision terms, for example, segmenting or tracking a garment in an image is far harder than segmenting a rigid object, because clothing continuously deforms and lacks a stable geometry.

Despite these challenges, the ability to manipulate soft materials is crucial for many high-impact applications. In industrial settings, advanced cloth handling could transform garment manufacturing, automated packing, and logistics (for example, automating sorting, folding, or seam-sealing processes). In the service and domestic domains, robots that can sort laundry, fold garments, or assist with household chores would dramatically reduce human effort and expand the utility of personal assistant robots. In the healthcare and assistive sectors, robots capable of handling soft materials are essential for tasks like assistive dressing, moving bedding or linens, and providing aid to the elderly or disabled. In fact, enabling robots to manage soft fabrics could improve quality of life and generate significant economic benefits across industry and daily living[1].

Recent years have seen remarkable breakthroughs in deep learning for computer vision, with neural networks achieving near-human accuracy on many recognition tasks. However, the effective handling of garments by robots remains a largely unsolved research problem. Much of the existing computer vision progress in clothing comes from the fashion industry or e-commerce (e.g. clothing classification, virtual try-on, and retrieval), where images typically show a well-posed garment on a model under controlled conditions. Large fashion datasets (e.g., DeepFashion and its variants) contain hundreds of thousands of images of apparel, but these images are structured and clean: garments are worn by people or displayed in ideal orientations[3]. By

contrast, in real-world robotic scenarios clothing items are often found crumpled in a pile, upside-down, partially occluded, or even wet or soiled. Such "**non-ideal**" configurations are rarely present in fashion datasets[2]. Moreover, the fashion datasets focus on tasks like category labeling or landmark detection on flat clothing, without providing the detailed spatial or topological information needed for physical interaction. In practice, robotic manipulation requires fine-grained understanding of garment geometry and state, for example, the precise location of sleeves, cuffs, collars, and corners, far beyond what is encoded in typical fashion images.

This gap between fashion-centric vision tasks and the physical needs of robotics motivates the development of specialized perception pipelines and tailored datasets. In particular, a vision system for garment manipulation must be able to handle the difficult, realistic states in which clothing is often encountered by robots (e.g. garments in a cluttered bin or lying on a folding table). Addressing this gap is essential to enable truly intelligent robotic clothing manipulation. The present thesis therefore aims to bridge the gap between computer vision and robotic manipulation of deformable objects, by designing a multi-stage vision pipeline and creating suitable data resources that together enable reliable perception of garments in challenging real-world conditions[2].

1.2 Problem Statement

The central research problem of this thesis is **to design, implement, and evaluate a robust vision-based perception system** for autonomous robotic manipulation of garments in realistic settings. The system must accurately interpret garments under diverse, real-world configurations (e.g. crumpled in a pile, partially folded, or lying on a surface) and do so quickly enough for use in a manipulation pipeline. This problem directly confronts the core challenges of deformable object perception: high variability and unpredictability in shape, infinite-dimensional state of cloth, and frequent self-occlusion and deformation under any robot-induced force[2]. In particular, garments can take on countless shapes and poses, and visual appearance can change dramatically with lighting or pose; any perception system must be robust to this diversity.

This thesis is carried out within the **VolPix** project, which is one of the research activities conducted under the **euROBIN** network. The euROBIN network aims to advance AI tools, software, architectures, and hardware components through a reproducible approach. Within this framework, **VolPix** focuses on automating the handling of laundry by robots. It targets 10 distinct clothing categories (such as T-shirts, trousers, socks, etc.) and requires operation in both wet and dry states. To support the autonomous manipulation tasks proposed by VolPix, the perception system must carry out a complete sequence of vision tasks, each of which presents its own challenges. These tasks include:

- **Instance Segmentation:** The first step is to **isolate each garment instance** from a cluttered scene (e.g. a pile of wet or dry clothes). This requires distinguishing one garment

from another and from the background at the pixel level. The difficulty is that garments often overlap and lack clear boundaries, and may share similar colors or textures. Effective segmentation in such cases is an open problem. Reliable methods must work without prior knowledge of the scene or a predefined background[1]. In practice, we employ state-of-the-art instance segmentation networks (e.g. Mask R-CNN or YOLO-based models) trained on garment images, but their accuracy can suffer under occlusion and background clutter. As noted in the literature, a major obstacle for data-driven garment segmentation is the scarcity of annotated images showing garments in realistic, occluded configurations[3]. To address this, our approach will investigate data augmentation to improve segmentation robustness on the intended domain.

- **Object Recognition (Category Classification and Detection):** Once each garment is segmented, the system must identify its category (e.g. "t-shirt" vs. "trousers" vs. "sock"). This task is inherently challenging due to **intra-class variation** (many different styles of t-shirts, trousers, or skirts) and **inter-class similarity** (e.g. briefs and boxers, or certain shirts and towels that share similar textures and shapes). Traditional deep convolutional neural networks such as ResNet or EfficientNet can be employed for multi-class image classification, but their success depends heavily on the availability of representative training data. In robotic manipulation contexts, labeled datasets are often small and task-specific, forcing careful adaptation through fine-tuning and transfer learning. For example, rotation-invariant or attention-based architectures can help account for arbitrary garment orientations, while augmentation strategies such as random rotations, scaling, and brightness variation improve robustness to appearance changes. In this work, transfer learning from large fashion datasets like DeepFashion2 is used as a foundation, with fine-tuning on our custom dataset to adapt to the specific challenges of EUROBIN.

  In addition to classification, this thesis also investigates **object detection** as an alternative recognition strategy. Rather than treating each garment image as a whole, object detection methods predict both the **bounding box** and the **class label** of each item. This approach leverages the spatial localization capabilities of modern detectors and makes better use of large-scale datasets that provide bounding-box annotations. In this project, a lightweight **YOLOv11-N** model was pretrained on DeepFashion2 and then fine-tuned on a smaller, custom-collected dataset. The detection-based approach proved particularly effective: by jointly learning localization and classification, the YOLO-based model demonstrated improved robustness to cluttered scenes and positional variations, outperforming purely classification-based methods in several scenarios. As a result, object detection with YOLOv11 was adopted as the primary recognition method within the pipeline, while classification served as a complementary baseline for comparative analysis.

- **Keypoint Detection:** To facilitate grasping and manipulation, the system must detect **semantic keypoints** or landmarks on the garment, such as sleeve ends on a shirt, collars,

cuffs, or waist ends. These keypoints serve as grasp points or as references for planning actions (e.g. folding along a detected edge). Unlike rigid objects with fixed features, garments have no fixed skeleton or topology, so keypoints must be localized from appearance alone. The problem is further complicated when garments are crumpled: important landmarks may be occluded, folded, or obscured by other cloth. Contemporary keypoint detection techniques typically treat this as a heatmap regression problem with convolutional networks. Prior work on robotic cloth has successfully used such models [2]to find corner points and edges. As Lips *et al.* demonstrate, detecting *non-occluded* keypoints on flattened clothes enables downstream tasks like folding via scripted motions[2]. In our system we adopt Detectron2's Keypoint R-CNN, which predicts 2D heatmaps for each semantic keypoint within a region of interest, providing accurate localization of garment landmarks. We must carefully design the set of keypoints for each garment type (e.g., sleeve endpoints and collar for a shirt) and train the model on labeled examples. The core difficulty is robustness: the detector must handle arbitrary deformations and partial occlusions, and yet still reliably identify points. We will explore grouping strategies treating similar clothes as a group.

- **3D Shape Inference (Mesh Prediction):** Finally, the system must reconstruct the 3D shape (mesh) of a garment from a single RGB image. A full 3D model is crucial for advanced reasoning about manipulation tasks such as estimating drape, tension, or planning folding actions. Since specialized RGB-D cameras were not available in this project, we adopted a monocular approach that relies on learning-based priors. Our method follows a multi-stage pipeline: a state-of-the-art monocular depth estimator (Depth Anything V2) provides dense depth information from the RGB input; garment segmentation masks from YOLOv11 (optionally refined with SAM) are applied to isolate the target garment; and the segmented depth map is then used to reconstruct a textured 3D mesh. This approach allows mesh reconstruction without additional hardware, though it inherits the limitations of monocular inference, particularly in handling thin structures such as edges and open garment boundaries. Due to the lack of ground-truth 3D garment datasets for evaluation, we performed a qualitative assessment, focusing on the plausibility of reconstructed topology, folds, and textures. While this method demonstrates the feasibility of RGB-only mesh prediction, it also highlights open challenges in achieving high-fidelity garment reconstruction for robotic manipulation.

In summary, the research problem is to build a **vision perception pipeline** that integrates these four core tasks (instance segmentation, category classification, keypoint detection, and 3D mesh reconstruction) into a unified system for robotic garment manipulation. Each subtask brings its own open challenges, compounded by the highly deformable and variable nature of cloth [1], [2].The success criterion is a system that can take an input image of a real garment (or pile of garments) and output all needed information reliably enough to be used by a downstream motion planner or control algorithm.

1.3 Thesis Structure

This thesis is organized into six chapters, progressing from background concepts to final conclusions:

- **Chapter 2 (Literature Review):** This chapter surveys related work in vision-based deformable object manipulation. It begins with an overview of robotic manipulation of deformable objects, highlighting classical modeling and modern data-driven approaches. It then reviews the specific vision tasks in our pipeline: instance segmentation (e.g. Mask R-CNN, U-Net methods for cloth segmentation), garment classification (deep networks trained on fashion datasets), semantic keypoint or landmark detection (especially in cloth manipulation contexts), and 3D shape estimation techniques (including mesh prediction from images). For each task, we compare the existing algorithms, emphasizing those focused on clothing or similar soft objects.

- **Chapter 3 (Dataset):** This chapter describes the data used for training and evaluation. We make use of the public *DeepFashion2* dataset, which contains hundreds of thousands of annotated images of garments (with segmentation masks and landmarks). However, DeepFashion2 primarily consists of well-dressed models and studio images, so we detail how we adapt it to our robotic context. Crucially, we introduce our **custom garment dataset** collected for the EUROBIN project. This dataset includes images of the ten target garment categories in both *dry* and *wet* conditions, captured on a flat surface by a single camera. We describe the data collection protocol (variations in pose, lighting, wetness) and annotation process, which includes instance masks, category labels, and keypoint locations. We also compare statistics of the datasets (image count, keypoints per item, etc.) to show their coverage and relevance to the task.

- **Chapter 4 (Methodology):** Here we detail the proposed multi-stage perception pipeline. We first describe the instance segmentation model (e.g. a fine-tuned Mask R-CNN or a YOLO-based segmentation network), including network architecture, loss functions, and training setup. Next, we present the **object recognition stage**, which is investigated using **two complementary approaches**: (1) traditional **multi-class classification models** (e.g. ResNet, EfficientNet, VGG16), trained with transfer learning and extensive augmentation to cope with limited data, and (2) a **YOLO-based object detection framework**, which simultaneously localizes and classifies garments, leveraging bounding-box annotations and demonstrating improved robustness to clutter and positional variation. Then we detail the keypoint detection approach: the network architecture, the choice of semantic keypoints for each garment type, and the training procedure. Finally, we discuss the 3D mesh prediction model. We describe the model that takes an RGB image and outputs vertex positions of a garment mesh. Throughout, we discuss our design choices, network inputs/outputs, and implementation details. Any novel architectural contributions or multi-task learning strategies are also explained in this chapter.

- **Chapter 5 (Experiments & Results):** This chapter presents the evaluation of each component and the overall system. We report quantitative results using standard metrics: mean Average Precision (mAP) for segmentation and object detection, Object Keypoint Similarity (OKS) for keypoint detection, and classification accuracy. For segmentation, we compare models trained with different architectures and training epochs. For object recognition, we analyze both **classification** and **object detection** methods: evaluating the effect of transfer learning for classification and comparing its performance against YOLOv11-based detection models fine-tuned on the custom dataset. For keypoints, we evaluate the performance of the group-based models and show qualitative examples of predicted keypoints on real images. For 3D shape, we visualize predicted meshes overlaid on images and provide qualitative assessments of mesh coherence. We discuss the results thoroughly, highlighting which approaches worked best and analyzing failure cases. An end-to-end system evaluation, showing the pipeline in operation on robot-like tasks, is also included.
- **Chapter 6 (Conclusion and Future Work):** In the last chapter we will discuss about the key findings and results of this work and possible future work. We restate the importance of robust cloth perception for robotic automation and note how our multi-stage pipeline addresses the challenges. We summarize the performance gains and novel findings shown in the experiments. We also discuss the limitations of the current system and how these might be overcome. Finally, we propose directions for future research: this may include collecting more diverse training data or extending the pipeline to handle dynamic manipulation. We highlight how the learnings from this work can serve as a foundation for advancing deformable object manipulation in robotics.

Each chapter builds upon the previous, moving from foundational concepts through implementation details to evaluation and broader implications. Together, they constitute a comprehensive study of **3D perception of garments in robotic applications**, grounded in the goals of the EUROBIN project.

Chapter 2

2 Literature Review

2.1 Vision-Based Robotic Manipulation

Manipulating deformable objects presents a significant set of unresolved challenges in robotics, encompassing modelling, perception, and control [4]. The complexity of this problem arises from two primary factors that characterize deformable objects like cloth: their state is high-dimensional and difficult to represent canonically, and their interaction dynamics are non-linear and influenced by physical properties that are typically not known in advance. While properties such as elasticity, stiffness, and friction are evidently significant in cloth manipulation, accurately categorizing them remains a difficult task [5].

To generalize manipulation skills, robots must be able to adapt to variations in an object's pose, shape, and physical properties. Feedback-loop manipulation is a powerful class of methods for adapting to these variations; however, its application to deformable objects is under-explored due to the core challenges of state estimation and dynamics modelling [6]. A robust computer vision pipeline is therefore a foundational component for enabling these advanced manipulation strategies.

A recent comprehensive survey of the field [7], organizes the current state-of-the-art around several key research thrusts that highlight the primary issues and dominant approaches. This review follows a similar structure, examining the literature through the lenses of state representation, the use of simulation and the resulting reality gap, and the trend towards end-to-end learning policies.

One of the most fundamental issues tackled in the literature is state representation. Accurately describing the configuration of a piece of cloth is non-trivial and is a prerequisite for any successful manipulation. Approaches range from using explicit geometric descriptors, such as a sparse set of semantic keypoints or a dense 3D mesh[8], to more recent methods that learn implicit, latent representations directly from sensor data. These learned representations aim to capture the essential features of the cloth's state without being constrained to a predefined structure, which is a key focus of current data-driven methodologies[9]. The choice of representation directly impacts the feasibility and success of downstream manipulation tasks.

A dominant approach to overcoming data scarcity in robotics is the use of simulation for policy learning. Researchers leverage physics simulators to generate millions of interaction samples, which would be infeasible to collect in the real world[1]. However, this approach introduces the

significant issue of the "simulation-to-reality" gap. Policies trained exclusively in simulation often fail on physical systems due to subtle differences in dynamics, friction, and visual appearance. Consequently, a major cluster of recent research focuses explicitly on bridging this gap, with studies dedicated to benchmarking the performance drop from sim-to-real and developing techniques like domain randomization to create more robust policies [10].

Paralleling these efforts is the trend towards learning **end-to-end manipulation policies** that map perception directly to motor commands. Instead of relying on a modular pipeline of state estimation followed by planning, these methods use large neural network models, often based on Transformer or diffusion architectures, to learn the entire control sequence from raw visual input [11]. While powerful, these methods are data-hungry and often opaque, making their success heavily dependent on the quality of training data and the effectiveness of the sim-to-real transfer. This highlights a foundational requirement across all modern approaches: the need for a robust and comprehensive perception system, as developed in this thesis, to provide the high-quality state information that these advanced policies depend on.

2.2 Instance Segmentation for Garment Isolation

2.2.1 Instance Segmentation

In the field of computer vision, there has been a clear progression from coarse to fine-grained image inference. This evolution begins with **image classification**, the task of assigning a single categorical label to an entire image. An incremental step forward is **object detection**, which not only classifies objects but also localizes them within the image, typically by drawing a bounding box around each one. A further refinement is **semantic segmentation**, which aims to classify every pixel in an image according to the object class it belongs to. However, semantic segmentation does not differentiate between separate instances of the same class; for example, it would label all pixels belonging to multiple t-shirts as one single "t-shirt" region [12].

Instance segmentation represents a more advanced and challenging stage in this evolution, as it combines the goals of the previous tasks. The objective is to correctly detect all objects in an image while also precisely segmenting each individual instance [13]. This means it provides a different label or mask for separate objects, even if they belong to the same class. In essence, instance segmentation can be understood as a task that simultaneously solves the problem of object detection and semantic segmentation [12], [13]. The rapid progress in this area has been significantly driven by the introduction of large-scale benchmark datasets, such as Microsoft COCO, which provide the rich, per-instance mask annotations necessary for training and evaluating models [14].

A useful way to frame these different tasks is by dividing a scene's components into **"stuff"** and **"things"** [15]. "Stuff" refers to amorphous, uncountable regions like the sky or a road, while

"things" are countable, distinct objects like cars or people. With this framework, semantic segmentation can be seen as a method that excels at understanding "stuff" but merges all "things" of the same class. In contrast, instance segmentation focuses specifically on "things," aiming to delineate each one precisely. A third paradigm, panoptic segmentation, unifies the two by providing a comprehensive map of both "stuff" and "things" [15]. The conceptual differences between these tasks are clearly illustrated in Figure 1.

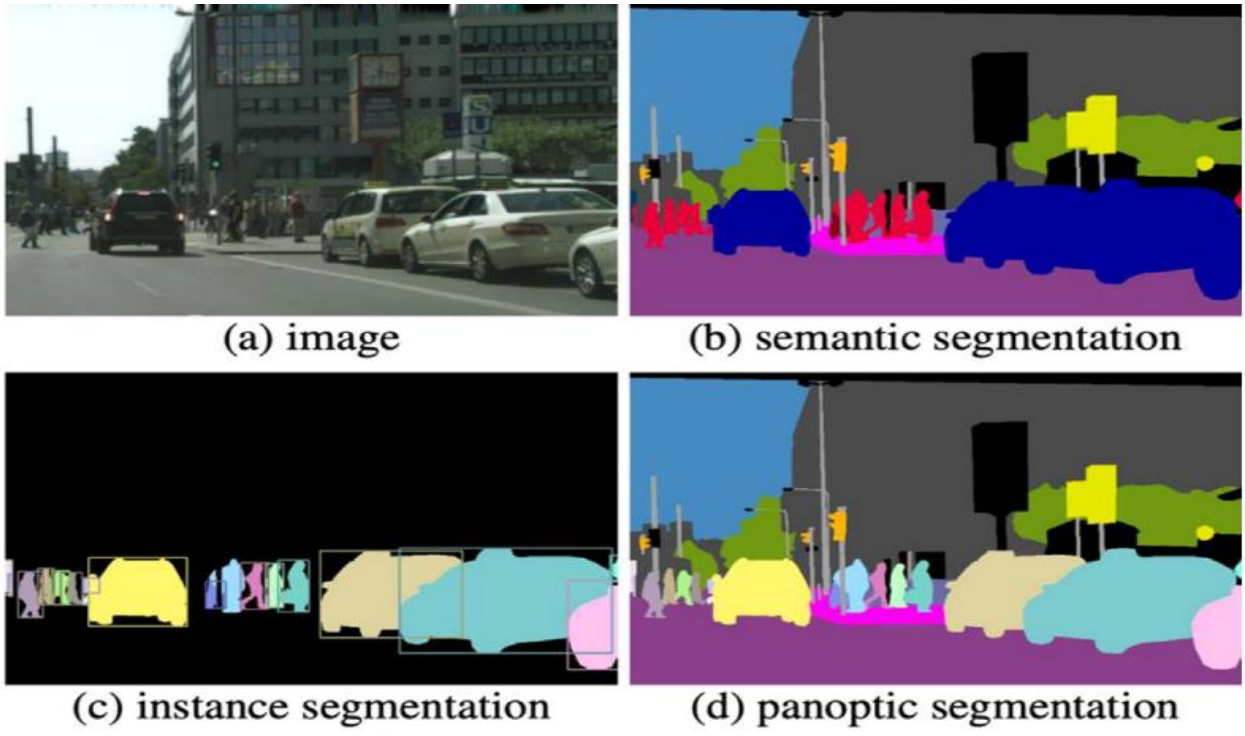

Figure 1: A visual comparison of segmentation tasks. (a) The original image. (b) Semantic segmentation classifies all pixels. (c) Instance segmentation isolates each distinct object ("thing"). (d) Panoptic segmentation provides a complete scene map of both "stuff" and "things".[16]

For this project, **instance segmentation** was selected as the most appropriate method. While semantic segmentation would fail at the primary goal of separating one t-shirt from another in a pile, panoptic segmentation introduces unnecessary complexity and computational overhead for the task at hand. The robotic manipulation of a garment does not require a complete semantic map of the entire scene; rather, it requires a direct answer to the question: "Where is the specific, individual garment that I need to pick up?"[15]. Instance segmentation provides the perfect balance by focusing exclusively on delineating the countable "things", the garments, that are the target of manipulation.

Therefore, applying instance segmentation in this context moves the challenge beyond standard benchmarks into a complex, real-world scenario where the "things" are deformable, heavily occluded, and lack a fixed shape. A highly accurate and robust instance segmentation model is the

critical foundation for the entire perception pipeline, making it an essential area of investigation for this thesis.

2.2.2 State-of-the-Art Approaches

Modern approaches to instance segmentation are predominantly categorized into two main families, representing a fundamental trade-off between performance and computational efficiency. The first are **two-stage methods**, which prioritize accuracy by first proposing regions of interest and then generating masks for each region in a sequential process, a paradigm famously established by Mask R-CNN [13]. In contrast, **single-stage methods** are designed for speed and real-time applications, performing object detection and mask prediction simultaneously in a single pass, an approach popularized by the YOLO (You Only Look Once) family of models [17]. The selection between these two paradigms is a critical design choice, often dictated by the specific requirements of the application, such as the need for high-precision masks versus the demand for low-latency inference in robotic systems.

2.2.2.1 Two-Stage Methods: Mask R-CNN

Two-stage methods are renowned for their high accuracy. The archetypal model for this category is **Mask R-CNN**, which extends a powerful object detector (Faster R-CNN) with the capability to produce high-quality segmentation masks for each detected instance. Its architecture can be understood as a multi-step process that refines information progressively[13].

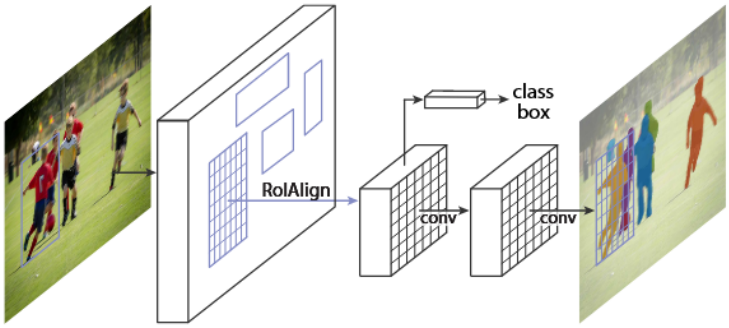

Figure 2. The Mask R-CNN framework for instance segmentation.[13]

The main components of the Mask R-CNN architecture are[13]:

1. **Backbone Network:** The process begins with a standard **Convolutional Neural Network (CNN)**, which acts as a feature extractor. This "backbone," such as a ResNet or the ResNeXt-101 model used in this thesis, processes the input image and generates a rich set of feature maps that capture details at various scales.
2. **Region Proposal Network (RPN):** The feature maps are then fed into an RPN. This network efficiently scans the features and proposes a set of rectangular **Regions of Interest**

(RoIs) areas that are likely to contain an object. This completes the first stage of the process.

3. **RoIAlign:** This is a key innovation of Mask R-CNN. For each proposed RoI, RoIAlign extracts a small, fixed-size feature map. It does so with precise alignment, preserving the exact spatial locations of features, which is crucial for generating accurate pixel-level masks.
4. **Prediction Heads:** In the second stage, the fixed-size feature map for each RoI is passed to three parallel "heads" to perform the final tasks:
    - A **classification head** predicts the object's class (e.g., "t-shirt," "trousers").
    - A **bounding box regression head** refines the coordinates of the box to tightly enclose the object.
    - A **mask head**, which is a small Fully Convolutional Network (FCN), generates a pixel-level binary mask that outlines the object's exact shape within the bounding box.

By decoupling the tasks of finding objects (stage one) and classifying/masking them (stage two), Mask R-CNN often achieves superior precision, making it a benchmark for tasks where mask quality is paramount.

2.2.2.2 Single-Stage Methods: YOLO

In contrast to the multi-step process of two-stage models, **single-stage methods** are engineered for speed and efficiency, making them highly suitable for real-time applications like robotics. The pioneering and most prominent family of models in this category is **YOLO (You Only Look Once)**, which reframes instance segmentation as a problem that can be solved in a single pass through a neural network.

The foundational architecture of YOLO is built on a unified detection pipeline that treats object detection as a single regression problem [17]. The model divides an input image into an S×S grid, and a single Convolutional Neural Network (CNN) simultaneously predicts bounding boxes, confidence scores, and class probabilities for each grid cell. A key concept is the confidence score, which combines the probability that a box contains an object with the Intersection over Union (IoU) of the predicted and ground-truth boxes. After this single network pass, a post-processing step called Non-Max Suppression (NMS) is used to prune duplicate detections and yield the final set of bounding boxes. This unified design is the source of YOLO's renowned speed.

To extend this high-speed detector to instance segmentation, modern YOLO-based models add a parallel **mask prediction branch** that also operates in a single stage. This is a key difference from

Mask R-CNN, which first finds the object and then generates a mask in a second step. A common approach, popularized by models like YOLACT, involves two parallel tasks [18]:

1. A "protonet" branch generates a set of general-purpose "prototype" masks over the entire image.
2. The main detection head, in addition to predicting classes and boxes, also predicts a set of "mask coefficients" for each detected instance.

The final mask for an instance is then generated by linearly combining the prototype masks using the predicted coefficients for that instance. Because this entire process, detection and mask creation, is done in a single forward pass without any feature re-pooling, the model maintains its real-time performance, making it a powerful choice for robotic applications where both speed and segmentation are required. The **YOLOv11** model used in this thesis is an evolution of this single-stage philosophy.

2.3 Garment Recognition: Classification and Detection

In garment-manipulation applications, vision systems must both recognize what clothing is present (classification) and where it is (detection). Clothing manipulation is inherently challenging due to the deformable, highly variable nature of fabrics. For example, Nocentini *et al.* note that “clothing manipulation is a daily activity and represents a challenging area for a robot,” and emphasize that **detection and classification are key points for the manipulation of clothes**[19]. Recent robotic systems therefore often combine deep networks for garment categorization with detectors for salient features. Gustavsson *et al.* (2022) propose a pipeline that first classifies the garment category from an image and detects landmarks on the cloth, then uses this information to plan a stretching strategy[20]. In practice, classification typically determines the garment class (e.g. “shirt” vs “pants”) to select an appropriate manipulation, while object detection and gives a bounding box with the class label[21]. Below we review the key architectures used in this context, focusing on **VGG16, ResNet, EfficientNet (classification)** and **YOLOv11 Nano (detection)**, including their building blocks, depth, parameter/FLOP counts, and trade-offs between accuracy and efficiency.

2.3.1 Image Classification Models

Deep convolutional neural networks (CNNs) are the standard for visual classification. After the breakthrough of AlexNet (2012), deeper architectures like VGG16 and ResNet were introduced[22].

VGG16: A deep CNN with 16 weight layers (13 convolutional + 3 fully-connected). VGG16 uses only small 3×3 convolutions (stride 1) and 2×2 max-pooling, stacked uniformly through the network. This simple, repetitive structure yields rich feature representations. The final layers are

three FC layers (two with 4096 units) and a 1000-way softmax. It has roughly **138 million parameters** and requires about **15.3 billion** FLOPs per 224×224 input [23].

- **Advantages:** strong classification accuracy (92.7% top-5 on ImageNet) and wide transfer-learning support.
- **Drawbacks:** very large model size and slow inference, making it cumbersome on embedded robotic hardware.

ResNet: A family of deep convolutional neural networks that introduced the concept of "residual learning" to solve the degradation problem that plagued very deep models. This issue manifested as a paradoxical decrease in accuracy as network depth increased, even on the training data, indicating a fundamental optimization challenge rather than just overfitting. The core innovation is the reformulation of what layers learn; instead of learning a direct mapping, the network learns a residual mapping relative to the input[24].

This is implemented architecturally through the use of **skip or shortcut connections**, which bypass one or more layers and add the input to the output of the stacked layers[24]. This simple addition allows gradients to flow more directly to earlier layers during backpropagation, mitigating the vanishing gradient problem and making it possible to effectively train networks with hundreds or even thousands of layers[24], [25], [26]. The architecture is built from repeating blocks, such as the **"basic block"** (two 3x3 convolutional layers) used in shallower models like ResNet-34, or the more computationally efficient **"bottleneck block"**. The bottleneck block, used in deeper models like ResNet-50, employs a sequence of 1x1, 3x3, and 1x1 convolutions to reduce and then restore the number of channels, making the 3x3 layer a computational bottleneck and significantly improving efficiency[25], [26].

- **Advantages:** Its revolutionary skip connections enabled the training of extremely deep networks, a fundamental breakthrough in deep learning. The architecture's simplicity and strong performance have made it a robust and versatile baseline for a wide range of computer vision tasks. Its reliance on standard convolutions makes it highly optimized for hardware like GPUs and TPUs.[27]

  It has much lower parameter count and computational cost than VGG16 for comparable accuracy; easier training of very deep models due to identity shortcuts. For garment tasks, a ResNet-50 backbone is often used as a feature extractor: e.g., GarmNet employs a pretrained 50-layer ResNet to produce a 7×7 feature map for garment classification and landmark detection[28].

- **Disadvantages:** It is less parameter-efficient than more modern architectures, requiring a higher computational cost (FLOPs) and more parameters to achieve the same accuracy as

models like EfficientNet. The conventional method of scaling ResNets by simply adding more layers is less optimal than more principled scaling approaches. [29]

EfficientNet: A family of convolutional neural networks designed to optimize both accuracy and computational efficiency through a principled approach to model scaling. It introduced a novel compound scaling method that uniformly scales network depth, width, and input resolution with a single coefficient, ensuring a balanced allocation of resources. The architecture's baseline (EfficientNet-B0) was discovered through a neural architecture search and is built upon mobile inverted bottleneck blocks (MBConv) that incorporate efficient depthwise separable convolutions and Squeeze-and-Excitation modules for channel-wise feature recalibration.[29], [30], [31]

- **Advantages:** Achieves state-of-the-art accuracy with significantly fewer parameters and FLOPs compared to previous models like ResNet, making it highly efficient. Its lightweight and efficient design makes it ideal for deployment in resource-constrained environments, such as mobile and edge devices. [29]
- **Disadvantages:** slightly lower raw accuracy than very large models, and the complex block structure can be more intricate to implement. The use of depthwise separable convolutions can be less efficient on certain hardware accelerators where memory access is a bottleneck, potentially leading to higher inference latency than the low FLOP count might suggest.

Tan & Le (2019) show that EfficientNet models achieve state-of-the-art accuracy with far fewer parameters; for instance, EfficientNet-B7 attains 84.4% top-1 accuracy on ImageNet while being ~8.4× smaller and 6.1× faster than the previous best models[29].

In garment tasks, these pretrained CNNs (VGG, ResNet, EfficientNet, etc.) are often fine-tuned on clothing datasets (e.g. DeepFashion, Fashion-MNIST) to classify apparel. The resulting model can robustly recognize garment categories under varying poses or lighting, providing the robot with the item's identity and thereby informing downstream actions.

2.3.2 Object Detection Frameworks

Object detectors extend classification by also localizing items in the image. Early methods (R-CNN family) use region proposals, but these are slow for real-time use. In contrast, the **YOLO** (You Only Look Once) family of one-stage detectors predicts bounding boxes and class scores in a single forward pass[22]. In YOLO, the image is divided into a grid and each cell directly outputs the coordinates of any object it contains along with confidence scores. This design dramatically speeds up detection: the model needs only one evaluation per image, making YOLO ideal for real-time applications. For example, YOLO-based systems have been successfully deployed on high-speed textile production lines for automated defect inspection[32]. The single-stage approach trades a small drop in accuracy for large gains in efficiency and throughput. Subsequent YOLO

versions (v2, v3, v4, etc.) incorporate multi-scale feature maps, anchor boxes, and attention mechanisms to boost accuracy while preserving speed[32]. Compact variants like **Tiny YOLO** or **YOLO Nano** are designed for embedded devices. Wong *et al.* introduce **YOLO Nano**, a highly compact model (~4 MB) optimized via human-machine design; it requires only ~4.6 billion operations and achieves ~69.1% mAP on Pascal VOC, outperforming Tiny YOLOv2/v3 in accuracy despite its smaller size[33].

In the fashion and textile context, YOLO-based detectors have been adapted to find clothing items and features. Lee & Lin (2021) propose a two-phase YOLOv4 detector for *fashion apparel*: their model detects garments (jackets, tops, pants, skirts, bags) in images and benefits from transfer learning on fashion datasets[34]. Li *et al.* (2024) develop a real-time **fabric wrinkle and corner detector** using YOLOv5: they train on a custom dataset of cloth deformations and achieve over 90% detection accuracy[21]. The detected wrinkle lines and corner points are then used by the robot to perform a quadrilateral flattening maneuver, successfully smoothing the fabric. Such examples illustrate how object detection integrates into robotic cloth workflows: the vision system not only identifies the garment, but also pinpoints key regions for grasping or spreading, thus closing the loop between perception and action.

Building upon the advances in YOLO-based garment detection, our work employs the latest YOLOv11 Nano, which combines the efficiency of prior Nano variants with modern architectural enhancements tailored for robotic applications.

YOLOv11 Nano: A lightweight one-stage detector in the YOLO family, tailored for edge devices. YOLO models split an image into a grid and simultaneously regress bounding box coordinates and class probabilities[28]. The YOLOv11 architecture (2024) employs an optimized backbone and neck for enhanced feature extraction, and its *Nano* variant is pruned for speed. YOLO11-Nano contains only **2.6 million parameters**, and about **6.5 billion** FLOPs at 640×640 resolution. Despite its small size, YOLO11-Nano achieves competitive accuracy by leveraging modern improvements (e.g. efficient CSP-like modules, feature pyramid networks).

- **Advantages:** extremely fast real-time detection (designed for sub-40ms inference on a modern GPU) with a tiny model size, suitable for onboard processing.
- **Disadvantages:** lower accuracy than larger YOLO models, and still higher FLOPs than lightweight classifiers because detection requires multi-scale heads. In garment tasks, YOLO-style detectors can directly locate garments or landmarks: their single-shot output (object bounding boxes) speeds up recognition of deformed cloth pieces under clutter[28].

In summary, modern garment-manipulation pipelines leverage powerful CNN classifiers (e.g. VGG16, ResNet, EfficientNet) to recognize clothing types, and efficient detectors (e.g. YOLO variants) to localize garments and cloth features. Together, these methods provide the semantic and spatial understanding needed for robotic arms to autonomously handle and manipulate garments[19], [21].

2.4 Keypoint Detection for Robotic Grasping

2.4.1 Introduction

Keypoint detection aims to localize semantically meaningful points on objects or scenes (e.g. human joints, object corners) from images. For deformable objects like garments, reliable keypoints (e.g. sleeve ends, collar corners) provide a compact representation of object configuration that is useful for recognition and manipulation. In robotics, knowing the positions of a few salient keypoints on a piece of cloth or clothing can reduce the high-dimensional perception problem to a tractable state (e.g. four corners of a towel)[35]. Indeed, early cloth manipulation systems exploit cloth corners or landmarks to plan folding[35], [36].

Modern methods for keypoint detection range from traditional feature-based approaches to deep learning techniques. This review focuses on approaches applied to garments and robotic cloth manipulation. We will consider more recent CNN- and transformer-based models, such as Detectron2's Keypoint R-CNN, highlighting their advantages and limitations when applied to deformable objects. While classical approaches were functional in controlled environments, they were fundamentally brittle, struggling with complex textures, requiring precise prior segmentation, and failing to generalize to varied garment shapes or cluttered scenes. These limitations highlighted the lack of robustness of feature-based methods and motivated the shift toward modern, data-driven techniques[37]. Finally, we discuss the evaluation metrics most commonly reported in the literature, including the COCO benchmark's OKS-based average precision.

2.4.2 Deep Learning Approaches

2.4.2.1 Heatmap-Based CNN Methods

Modern keypoint detectors are dominated by deep neural networks, which learn to output confidence "heatmaps" for each keypoint. A common strategy is to use a convolutional backbone (e.g. ResNet, Hourglass) and append deconvolutional layers that produce a spatial heatmap for each of K keypoint types. Each heatmap pixel represents the probability of a keypoint at that location. For training, a 2D Gaussian (or peaked label) is placed at each ground-truth keypoint, and the network is trained (e.g. with pixel-wise cross-entropy or MSE) to match this target. This was used in seminal works like Convolutional Pose Machines[35] and the Stacked Hourglass model, and continues in state-of-the-art pipelines.

For example, Lips *et al.* use a fully convolutional "U-Net" style network to detect cloth keypoints as heatmaps[35]. Their network has encoder-decoder skip connections, ReLU activations, and a final sigmoid output for probability (trained with pixel-wise BCE loss)[35]. Similarly, the Mask R-CNN architecture adds a small "keypoint head" on top of ROI features: it applies four 3×3 conv layers followed by a 2× up-sampling (deconvolution) to output K heatmaps (one per keypoint) at e.g. 56×56 resolution[13].

The Detectron2 Keypoint R-CNN (built on Mask R-CNN) follows this design: it uses a ResNet+FPN backbone to extract features and ROI Align to crop proposals, then a keypoint head

that outputs per-keypoint heatmaps. Each heatmap uses a sigmoid and is trained to match a Gaussian-labeled ground truth at the true keypoint location[35]. Because heatmaps preserve spatial detail, these architectures achieve high precision in keypoint localization. In practice, modern pose estimation models (e.g. OpenPose, HRNet, SimpleBaseline) all adopt heatmap regression. The output keypoint predictions are then extracted by taking the argmax (or using a small neighborhood max-filter) of each heatmap[35].

Heatmap methods are very effective when plenty of annotated data is available. They elegantly handle a varying number of instances (in bottom-up approaches) or per-instance output (in top-down pipelines). However, they require careful calibration: output resolution vs. input down-sampling trade-offs. They also produce dense outputs even when many pixels contain no keypoints, and may struggle with highly deformable or symmetric patterns (leading to multiple high responses). In clothing scenarios, heatmap methods have been applied to landmark detection on garments and to human-cloth interactions.

This heatmap regression approach has proven far more effective than earlier methods that attempted to directly regress keypoint coordinates.

2.4.2.2 End-to-End Detection Frameworks (Keypoint R-CNN)

A powerful modern approach is to integrate keypoint detection into an object detection pipeline. For example, Mask R-CNN extends Faster R-CNN to output segmentation masks[13]; a similar extension is *Keypoint R-CNN*, which outputs keypoint heatmaps per detected instance. Detectron2's implementation is a state-of-the-art example. In this top-down pipeline, the network first generates object proposals and classifies them (e.g. to find each person or garment). Then, for each proposal, ROI features are pooled (via ROI Align) and fed to multiple prediction heads: one head for bounding box regression, one for class, one for mask (if used), and one for keypoints. The keypoint head consists of several convolutional layers and upsampling to produce K heatmaps as described above. During training, it only computes loss on visible keypoints. Inference yields for each detected object both its bounding box and a set of keypoint coordinates (the argmax of each heatmap). In practice, training Keypoint R-CNN requires labeled bounding boxes and keypoint annotations, as in COCO. Its advantage is *instance awareness*: it explicitly ties keypoints to detected objects. This is very useful when multiple cloth items overlap or multiple humans appear. However, it is a multi-stage and relatively heavy approach (RPN + ROI heads) and may not leverage image-wide context for keypoint grouping.

Recently, Transformer-based architectures like Vision Transformer (ViT) and DETR have emerged as a powerful alternative to CNNs for keypoint detection. By leveraging self-attention mechanisms to capture global context, these models have shown results comparable or superior to established CNN methods on standard benchmarks. However, they are often computationally demanding, and their specific application to the challenges of deformable garment keypoint detection remains an active area of research[38], [39].

Figure 3. A visualization of the heatmap regression technique for keypoint detection. The model predicts a distinct heatmap for each keypoint (right), where the brightest area indicates the most likely location. These heatmaps are then used to determine the final keypoint coordinates on the input image (left).[40]

2.4.2.3 Detectron2 Keypoint R-CNN Architecture

Detectron2 (a PyTorch framework from Meta) provides a modular implementation of Mask R-CNN, including the keypoint head. Its architecture illustrates a typical state-of-art pipeline:

- **Backbone and FPN**: ResNet (or ResNeXt) network extracts multi-scale convolutional features. A Feature Pyramid Network (FPN) combines these into a pyramid of feature maps (with strides e.g. 4, 8, 16, 32)[35].
- **Region Proposal Network (RPN)**: On top of the backbone, an RPN proposes candidate object bounding boxes.
- **ROI Align**: Proposed boxes are cropped from the backbone features using ROIAlign to yield a fixed-size feature (e.g. 7×7×C) per proposal.
- **Bounding Box and Class Heads**: Standard Fast R-CNN heads (fc layers) classify each ROI and refine its box coordinates.
- **Keypoint Head**: For K keypoints per instance, the keypoint head takes the ROI features (e.g. 14×14 if up-sampled) and applies four 3×3 convolutions (with ReLU), followed by a deconvolution (transpose conv) to up-sample to e.g. 56×56 spatial resolution[35]. This

yields K heatmap channels. Each channel is a sigmoid map indicating the probability of that keypoint at each pixel. In training, these heatmaps are supervised with binary cross-entropy to target heatmaps (Gaussians at ground-truth locations)[35]. The total loss includes the sum of all keypoint map losses (often normalized by number of visible keypoints).

Because it is integrated into Mask R-CNN, Detectron2's Keypoint R-CNN is end-to-end trainable (given boxes and keypoints). It benefits from strong backbones and FPN context, and shares computation with the detection tasks. Its limitations include requiring box annotations (to train the RPN) and being relatively heavy for real-time. Nonetheless, it remains a popular choice for both human pose and object landmark tasks. As evidence of performance, on fashion images DeepFashion2, a Mask R-CNN baseline yields only ~0.56 AP on the landmark task[41], indicating keypoint R-CNN is struggling with highly variable cloth. In contrast, in domains with more data (like human pose), Keypoint R-CNN and its variants set strong baselines.

2.4.2.4 Keypoint Detection for Garments and Robotic Manipulation

Garments pose unique challenges: they are nonrigid, highly deformable, and often self-occluding. Clothing landmarks (e.g. garment corners, collar points, garment-specific joints) must be defined in a way that is both semantically meaningful and physically reachable. In fashion vision, *fashion landmark detection* has been studied to improve clothing recognition and retrieval. Liu *et al.* and Yan *et al.* introduced landmark sets for garments (e.g. neckline corners, sleeve ends, hem corners)[37], [42]. Yan *et al.*'s DLAN network jointly detected clothes bounding boxes and landmarks in unconstrained images, achieving robust results without manual cropping[37]. These approaches treat landmarks similar to human body joints but on garments.

In robotic cloth manipulation, papers often focus on a few keypoints relevant to tasks. For example, in towel folding one needs the four corners; in shirt folding, elbows and shoulders may define fold lines. Classical robot pipelines (e.g. Doumanoglou *et al.*) used edge detection and corner templates for towels, then computed folds[43]. More recent work learns to detect cloth corners with CNNs. Lips *et al.* train a CNN on synthetic towels to detect all four corner points as heatmaps[35]. Even when transferring to real towels, their detector achieved a grasp success rate of 77% and full fold success of 53%. The key was generating a diverse synthetic dataset (random cloth textures, distractors) and using a U-Net style heatmap predictor[35]. Similarly, Lips *et al.* extend this idea to multiple garment types (T-shirts, shorts, towels), reporting ~64% AP on real images from synthetic-only training (improving to 74% after limited real fine-tuning)[8]. These works underline that cloth keypoint detectors can generalize if enough variability is synthesized.

Strengths and Weaknesses: CNN-based keypoint detectors excel when adequate training data (or realistic simulations) are available[8]. Their localization precision can be very high for visible keypoints. However, garment detection suffers from occlusion and ambiguity: when cloth is crumpled or overlapping, even humans may disagree on "where" a sleeve end is. Keypoint R-CNN

mitigates some issues by reasoning per detected object, but background clutter and clothing prints still confound models. Moreover, many garment classes have relatively few annotated examples, so transfer learning or synthetic data (as in Lips et al. or ClothesNet[44]) is often used. In contrast, classical methods need strong assumptions (flat cloth, known color), and unsupervised methods may not discover *semantically stable* points like sleeves.

Semantic Keypoints: A novel trend is to define *semantic* garment keypoints (e.g. "left sleeve cuff", "right hem") that match human language and commonsense. *Deng and Hsu* propose semantic keypoints for clothing items, learned via vision-language models. Each keypoint has a text label (e.g. "collar") and a 2D location, offering interpretability[45]. Their system (CLASP) automatically discovers such points on prototypes and transfers them to new clothes, which helps a robot plan folds by following language-like instructions. While very promising for **generalizing across many garment types**, semantic keypoint methods are in their infancy and rely on large foundation models. They highlight that beyond purely visual features, language-grounded knowledge can improve garment representations.

2.4.3 Summary and Outlook

Keypoint detection methods have evolved from handcrafted features and geometric fit to deep learning models that produce dense heatmaps or end-to-end detections. For rigid or semi-deformable objects (like humans or articulated bodies), deep CNNs (stacked hourglass, HRNet, Mask R-CNN) achieve high accuracy given large annotated datasets. For highly deformable objects like garments, however, challenges remain. Classical approaches require strong assumptions (flatness, plain background) that limit real-world use[35]. CNN-based methods can learn robustness, but must grapple with occlusion, variability, and limited data. Synthetic data and augmentation help (as shown by Lips *et al.* achieving 77% grasp success on towels[35]), but a reality gap persists[35].

Detectron2's Keypoint R-CNN embodies the current standard pipeline: a ResNet-FPN backbone with ROI heads for classification, bounding box regression, mask prediction and keypoint heatmaps. It leverages multi-task training and provides strong performance when data is abundant. Yet even Mask R-CNN yields modest performance on clothing landmarks (AP ≈56% on DeepFashion2[41]). Transformer-based models (DETRPose, ViTPose) offer alternative pipelines that remove components like ROI cropping or introduce global attention. Early results suggest transformers can match or exceed CNNs on pose tasks[38], [39], but they demand more compute and data.

In the context of clothing manipulation, effective keypoint detection requires both visual precision and task relevance. For example, detecting arbitrary corner points is insufficient if they don't correspond to graspable features. Future work is likely to combine vision with language and physics: the CLASP semantic keypoints approach[45] is one example where keypoints are chosen

for their actionability. On the technical side, gaps remain in few-shot learning of landmarks, unsupervised adaptation to new garment types, and robustness to heavy occlusion.

2.5 3D Mesh Prediction from a Single Image

Single-view 3D reconstruction has seen rapid advances in recent years, driven by the demand for virtual modeling and robotics applications (e.g. real-to-sim). Early approaches used voxel grids or coarse point clouds, but more recent work focuses on directly predicting *mesh* geometries from RGB images. Meshes are preferred in graphics and robotics (for simulation) due to their compact explicit surface representation[46], [47]. The challenge of reconstructing 3D meshes from a single image has been approached through several distinct methodologies. This review will cover the main paradigms, from early template-based and volumetric methods to more recent implicit and depth-assisted pipelines, highlighting the advantages and disadvantages of each.

2.5.1 Template Deformation and Regression-based Methods

Template-based methods start from a fixed mesh (often a simple ellipsoid or human body model) and learn to deform it to match the image. **Pixel2Mesh** is a seminal example: it uses a graph-CNN to iteratively deform an ellipsoid so that its rendered image matches the input[48]. A coarse-to-fine strategy ensures stability, and various mesh-specific losses (edge length, normal consistency) help produce plausible geometry. Wang *et al.* report that Pixel2Mesh yields more detailed meshes and higher shape accuracy than prior methods[48]. Follow-up work (e.g. Pixel2Mesh++ for multi-view) and **Mesh R-CNN** extend this idea. Mesh R-CNN augments Mask R-CNN detection with a mesh branch: after detecting an object, it predicts a coarse voxel shape which is converted to a mesh and then refined by graph convolutions[46]. These methods excel on object benchmarks (ShapeNet, Pix3D)[46].

For garments and humans, body models provide a natural template: e.g. **CAPE** learns a generative clothing model as an extension of the SMPL body mesh. CAPE trains a conditional mesh-VAEs (with mesh-GAN discriminators) to deform the SMPL surface according to clothing type and pose[49]. As a result, CAPE can "dress" SMPL bodies in a variety of clothing styles, preserving global shape and local wrinkles[49]. These template methods are fast at inference (single forward pass) and work well when the training categories match the test (e.g. known garment types), but they can overfit to limited topologies.

DeepFashion3D highlights this: existing cloth models were limited to fixed topologies, so the authors propose an *"adaptable template"* that can represent multiple clothing topologies in one mesh[42]. In practice, this combines a base mesh (like SMPL) with learned offsets for different garment types, yielding strong reconstruction on a new garment dataset. In summary, template-

deformation models (graph CNNs on meshes) provide a direct mesh output and leverage image features effectively[49], [50], but they may struggle with open boundaries or unseen topologies unless specifically designed (as in DeepFashion3D).

2.5.2 Volumetric and Implicit Function-Based Methods

A second class of methods predicts a coarse volumetric shape (e.g. a voxel grid or occupancy grid) from the image, which is then converted to a mesh (e.g. by marching cubes). The earliest neural examples (e.g. 3D-R2N2) used 3D convolutional decoders to generate a low-resolution occupancy grid from one or more images[51]. While volumetric methods can represent arbitrary topology and are easy to train with 3D CNNs, their resolution is typically limited by memory. Mescheder *et al.* cast the problem in *function space*: instead of a fixed grid, a neural network predicts an occupancy value for any 3D point given the image. This defines a continuous surface as the learned decision boundary. Occupancy Networks can produce very high-resolution shapes "at infinite resolution" without huge memory (just by querying the network many times)[52].

In experiments, Occupancy Networks achieved competitive single-view reconstruction results, handling complex topologies and noisy input[52]. Park *et al.* similarly train a neural signed-distance function per shape class, enabling high-quality interpolation and completion[53]. These implicit or occupancy approaches can naturally represent thin structures (like garments) and unseen topology. They also easily fuse multiple views or depth as inputs. However, they are often slower at inference, since evaluating the implicit network many times is needed to extract a mesh (by marching cubes).

Related to occupancy nets are methods that directly learn *implicit fields* from images. A breakthrough in 2019 was **PIFu** (Pixel-aligned Implicit Function)[54]. PIFu represents the human (with clothing) by an implicit function that maps 3D points to occupancy (or distance), where the function is conditioned on aligned image features. In practice, a CNN encodes the input image to a feature map, and a small MLP takes a 3D point's image-plane projection as input to predict if it's inside the surface. PIFu allows fine detail (hair, wrinkles, clothing layers) and arbitrary topology, and its authors demonstrate that it produces extremely high-resolution meshes that capture unseen parts (like the back of a person)[54].

Importantly, PIFu's implicit surface is memory-efficient and continuous, unlike a voxel grid[54]. The multi-level extension **PIFuHD** (CVPR 2020) further improves fidelity by operating at multiple scales. Other works use similar ideas: Occupancy networks applied to images, or conditional NeRFs (e.g. PixelNeRF) for multi-view. In general, implicit methods (PIFu) excel in detail and generality, but often need large networks and sampling loops, making them slower than direct mesh regression[47].

2.5.3 Depth- and Multi-View Assisted Pipelines

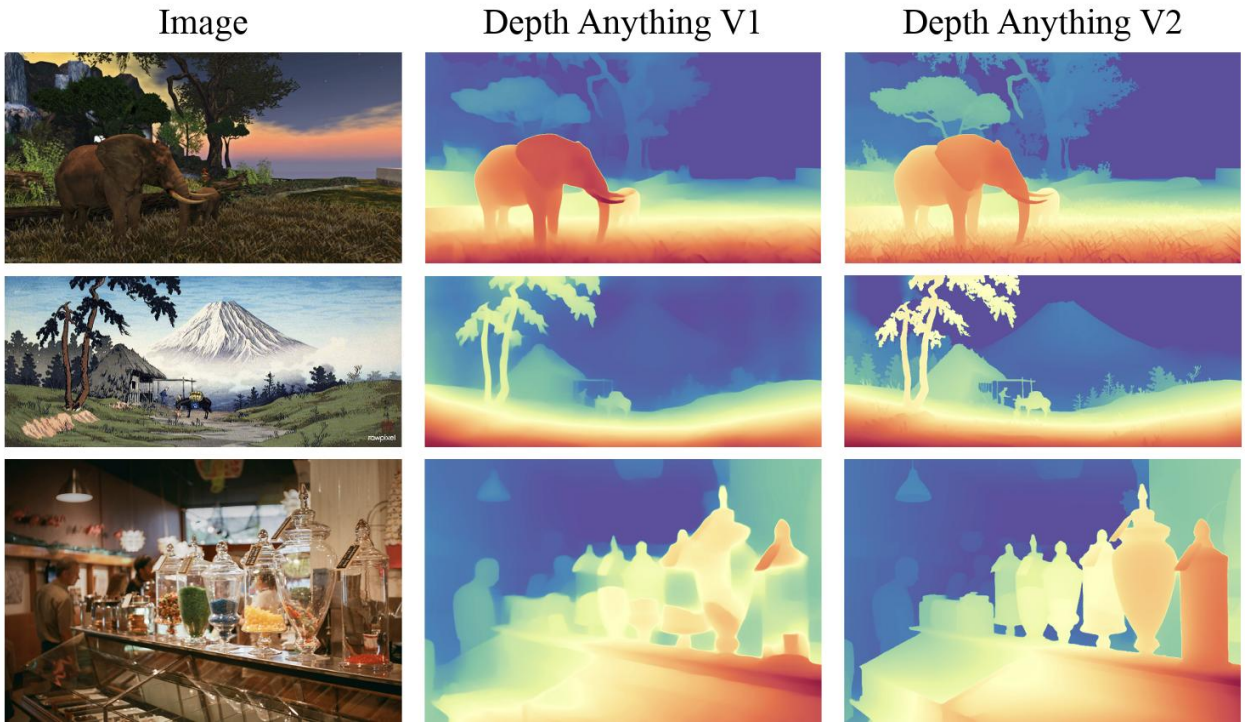

Figure 4. A qualitative comparison of monocular depth estimation models. Depth Anything V2 produces finer details and higher accuracy compared to prior state-of-the-art method Depth Anything V1. (Image from Depth anything V2 paper)

Some recent systems leverage intermediate depth or multi-view reasoning to improve reconstruction from a single image. For example, GarmentCrafter (2025) first predicts a depth map from the input garment image, warps it to generate novel views, and then uses a multi-view diffusion model to "inpaint" occluded areas before a final RGB-D to mesh pipeline yields the 3D garment[55]. This type of hybrid approach helps enforce consistency across different viewpoints and can recover geometry from limited image data.

Similarly, when multiple images or a video sequence are available, traditional methods like multi-view stereo or structure-from-motion can be used. However, for single-image cases, these heuristics introduce significant uncertainty, especially in occluded regions. To address this challenge in RGB-only pipelines, a crucial approach is to rely on monocular depth estimation (MDE) networks.

Building upon this, the **Depth Anything** model significantly advanced the field by designing a data engine to leverage massive-scale unlabeled data, over 62 million images, to greatly enhance the model's generalization and robustness[56]. The most recent iteration, **Depth Anything V2**, further refines this data-centric philosophy[57] (Figure 4). It trains a highly capable teacher model exclusively on precise synthetic data to learn fine-grained details, and then uses this teacher to generate high-quality pseudo-labels for large-scale real images. This strategy effectively bridges

the synthetic-to-real domain gap while retaining detail, resulting in depth predictions that are both robust and precise. The ability of Depth Anything V2 to generate high-quality depth maps for any image without task-specific fine-tuning is the primary reason it was selected for the methodology of this thesis.

2.5.4 Garment- and Deformable-specific Methods

Garments pose special challenges (thin surfaces, self-occlusion, large deformations). Dedicated datasets and methods have emerged. **DeepFashion3D** provides ~2000 real 3D garment models with varied categories (shirts, skirts, etc.) and corresponding images. Zhu *et al.* use it to benchmark single-view cloth reconstructions, proposing a hybrid mesh+implicit network with an adaptable mesh template[50]. They show that combining mesh templates (for global shape) with implicit detail (for wrinkles) yields state-of-art garment reconstructions. Likewise, **Layered-Garment Net** tackles multi-layer clothing on a human: it represents each layer by a signed-distance field (SDF) and enforces intersection-free layering via a "garment indicator field"[58]. This implicit approach can model, e.g., an inner shirt and outer jacket simultaneously. In summary, garment-specific methods leverage strong priors about cloth (e.g. templates, learned style parameters) and often combine multiple cues (segmentation, skeletal pose, normal maps) to disambiguate shape. They highlight that open boundaries and occlusions are especially severe: simple voxel or implicit grids struggle to model the thin open surfaces of a shirt or dress, so template meshes (with learned offsets) or layered SDFs are favored[58].

2.5.5 Challenges and Trade-offs

Key issues in single-image mesh prediction include **occlusion**, **generalization**, and **fidelity**. Occlusion of unseen surfaces is inherent: methods must "hallucinate" back-of-object geometry from context. Nolte *et al.* find that occluded regions incur **40–95% higher reconstruction error** compared to visible parts[47]. Some works add a dedicated completion module (e.g. image inpainting or learned depth fusion) to mitigate this, but at computational cost. Clothing exacerbates this: a folded sleeve's underside is rarely visible, requiring strong shape priors or multiple layers of implicit fields.

Generalization is another concern. Many networks are trained on limited object classes or synthetic datasets. They often overfit to training shapes and fail on novel categories. Implicit methods (DeepSDF, PIFu) are somewhat more class-agnostic but still rely on training priors; generative diffusion models aim to be "open-vocabulary" but currently exhibit artifacts on everyday objects. In garments, generalizing to new styles or textiles remains hard: networks must learn shape *and* fine-scale wrinkle priors, which is why large real-cloth datasets (DeepFashion3D, CAPE) are crucial.

Fidelity (surface detail) involves a trade-off with speed and data. Pixel2Mesh yields reasonably detailed shapes quickly[48], while PIFu produces extremely detailed cloth geometry at the expense of a slow implicit inference. Nolte *et al.* report that most state-of-the-art reconstructions take **multiple seconds to tens of seconds per object** (even on a powerful GPU)[47]. Only a few methods (e.g. recent feed-forward mesh decoders like SF3D or optimized parametric fits) can run in under a second[47]. For real-time robotics, such latency is usually prohibitive. Memory usage also varies: volumetric methods are heavy in 3D tensors, while implicit fields use smaller networks but require many evaluations to extract a mesh.

Overall, recent benchmarks (e.g. DeepFashion3D, CAPE, Pix3D) show steady progress. Pixel2Mesh and similar achieve reasonable IoUs on chairs/cars; PIFu achieves state-of-art on clothed humans by capturing detail[54]. However, surveys like Nolte *et al.* highlight that **current single-view methods often fall short of robotics needs**: meshes may have holes, collisions, or be unstable under physics[47]. This gap suggests that future work must better address occlusions (through scene context or learned priors), ensure collision-free outputs, and optimize speed.

In summary, the literature presents a clear trade-off. Template-based and regression models are fast but often lack the flexibility for the diverse topologies of garments. In contrast, implicit and volumetric methods offer this flexibility but at a higher computational cost for inference. Recent surveys highlight that many of these methods still fall short of robotic needs, often producing meshes with holes or physical instabilities. This suggests that a hybrid approach, which leverages the strengths of powerful pretrained models for an initial geometric estimate, offers a practical path toward generating robust 3D meshes for real-world manipulation.

Given the need for a practical, hardware-independent solution for robotic manipulation, a hybrid pipeline leveraging a state-of-the-art monocular depth estimator presents the most promising balance of performance and flexibility, which is the approach adopted in this work.

Chapter 3

3 Dataset

Deep learning models rely heavily on the diversity and reliability of the training data for both performance and generalization. This is especially true for complex computer vision tasks such as the robotic manipulation of deformable objects, where models must learn to handle wide variations in texture, shape, and lighting conditions. Therefore, this thesis adopts a dual-dataset strategy. This approach leverages a large, publicly available dataset to build a foundational understanding of garment features, which is then refined using a smaller, custom-collected dataset tailored to the specific requirements of the manipulation pipeline.

The first component of this strategy utilizes the large-scale DeepFashion2 dataset [41] for pretraining. Its extensive and richly annotated collection of clothing items provides a robust starting point for learning generalizable features. The second component involves a custom dataset created specifically for this project, featuring 10 categories of clothing in both wet and dry states to address the unique challenges of the EUROBIN project's objectives. The mentioned dataset is essential for fine-tuning the models on scenarios directly relevant to robotic interaction, which are not present in existing public datasets.

In this chapter, the two datasets used in this study are described in detail. We describe the data collection and annotation procedures, the strategies for splitting the data into training, validation, and test sets, and the extensive preprocessing and augmentation techniques used to enhance model robustness and performance across all vision tasks.

3.1 DeepFashion2 Dataset Overview

To establish a strong baseline for feature extraction, this work utilizes DeepFashion2[41], a large-scale and diverse benchmark dataset. While other garment datasets exist, DeepFashion2 was selected for its unique combination of scale and rich, multi-modal annotations[41]. It comprises 491,895 images containing 801,732 distinct clothing items across 13 popular categories such as *short sleeve top*, *long sleeve top*, *trousers*, and *skirt*. A key strength of DeepFashion2 is its sourcing from both commercial stores and consumer photographs, which ensures that models are exposed to a comprehensive range of real-world variations[41].

Figure 5: Deepfashion2 examples. Image reproduced from [41]

Each clothing item in the dataset is accompanied by a rich set of annotations that are particularly well-suited for pre-training manipulation-oriented models. These include:

- **Bounding Boxes and Per-Pixel Segmentation Masks:** For precise localization and identification of garments.
- **Dense Landmarks:** A total of 294 landmarks are defined across the 13 categories, identifying key points such as collars, hemlines, sleeve cuffs, and waistbands. These are critical for learning a structured understanding of garment topology.
- **Viewpoint and Occlusion Labels:** Each item is labeled with a viewpoint such as *no wear*, *frontal*, or *side/back*. The 'no wear' images, which depict garments on flat surfaces, are especially valuable for providing a canonical view of clothing shape, which is a useful prior for robotic unfolding tasks.

The dataset is formally divided into training (391,000 images), validation (34,000 images), and test (67,000 images) sets, providing a standardized structure for model development and evaluation.

Within the scope of this thesis, DeepFashion2 serves a critical role as the pre-training source for the segmentation, classification, and keypoint detection models. By pre-training on such a vast and

varied dataset, the initial models learn a robust feature representation capable of handling challenges such as occlusions, different viewpoints, and significant variations in clothing appearance.

However, it is crucial to acknowledge the limitations of DeepFashion2 in the context of robotic manipulation. The dataset predominantly features clothing that is either worn by models or laid out flat. It lacks images of garments in complex, non-ideal states such as crumpled, folded, or in a pile, that are characteristic of real-world robotic interaction scenarios. This gap makes the use of a specialized, custom-collected dataset for fine-tuning necessary. The foundational training on DeepFashion2 provides a powerful starting point, which is then adapted using our custom data to specialize the models for the specific demands of the manipulation pipeline.

3.2 Custom Dataset for Robotic Manipulation

While large-scale datasets like DeepFashion2 provide an excellent foundation for pretraining, they often lack the specific requirements of applications like robotic manipulation. In particular, they lack images of garments in challenging, real-world states such as crumpled piles, partial occlusion, or wet conditions, which are common in robotic cloth handling. To bridge this gap, a custom dataset was developed to fine-tune the models on tasks and conditions directly relevant to the EUROBIN project's goals.

Figure 6 - sample of pile of clothes

Figure 7 - samples of 10 categories

3.2.1 Data Collection

A total of **51 physical clothing items** spanning **10 categories** were purchased to ensure coverage of a representative range of garment types commonly encountered in laundry-handling scenarios. These garments served as the **source materials** for image acquisition, from which the dataset samples were generated.

The selected categories include:

- T-shirt
- Sweater
- Tank top
- Crop top
- Trousers
- Long Socks
- Shorts
- A-line skirts
- Briefs
- Boxers

The distribution of the physical garments across categories is shown in Figure 8. Multiple images were then captured for each item or items under different configurations and viewpoints to build the final dataset used for training and evaluation.

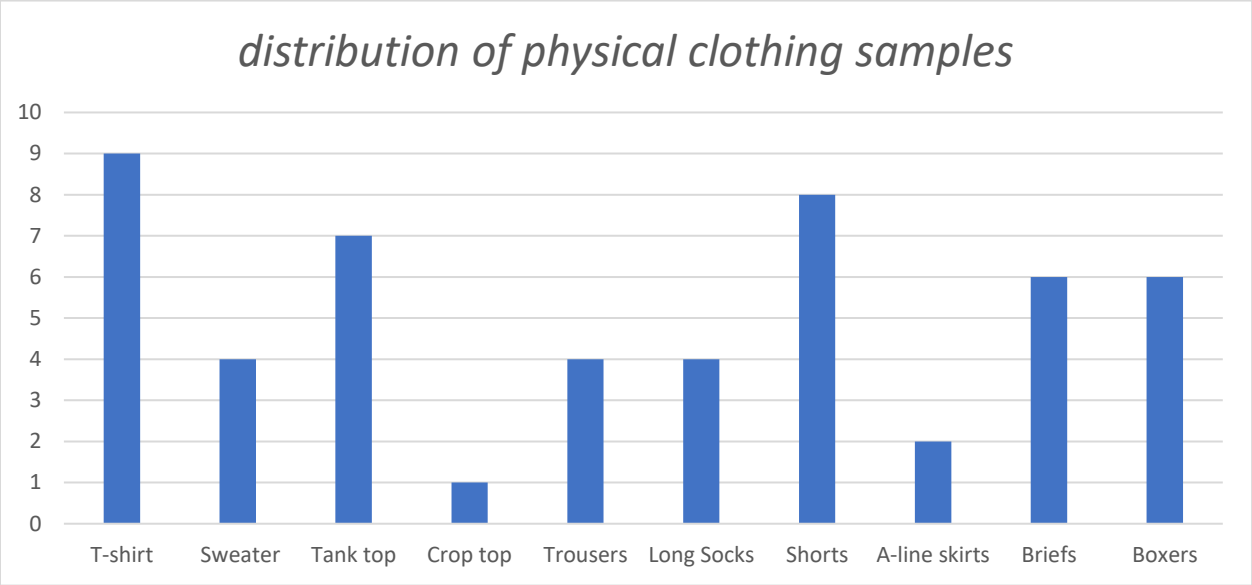

Figure 8 - chart of distribution of physical clothing samples

For each garment, images were captured in **both wet and dry conditions**, enabling the models to learn material appearance changes due to water absorption, reflections, and altered drape. This approach is crucial for improving robustness in real-world robotic operations, where the same object may present significantly different visual features depending on its state.

Although the dataset was custom-collected, the distribution of garment categories could not be fully balanced because the clothing items were provided as part of the project resources rather than being independently selected. As a result, some categories, such as crop tops and A-line skirts, were underrepresented. Despite this limitation, the dataset still ensured sufficient variability for training and evaluating the perception models, as discussed in Section 5.

3.2.2 Data Annotation Process

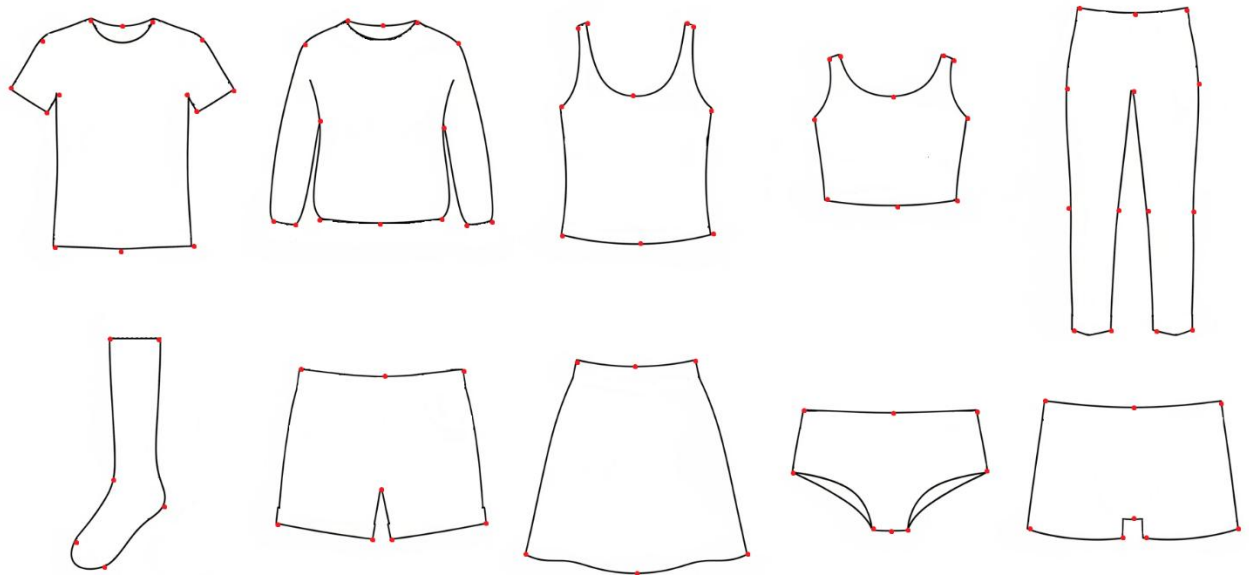

Figure 9 - key point labeling samples

Annotations were performed using **Label Studio**, with a single annotator to ensure **consistency and labeling quality** across all samples. These annotations included:

- **Segmentation masks** for accurately identifying the precise boundary of each clothing item.
- **Bounding boxes** to locate each garment within the image.
- **Keypoints** marking essential features, such as the shoulders or corners of a garment, which are critical for robotic grasping and manipulation (Figure 9).

3.2.3 Dataset Structure and splits

Separate subsets were prepared for each vision task (segmentation, classification, object detection, and keypoint detection) to enable task-specific optimization. An example split for the segmentation task includes:

- 610 total images
  - 310 dry
  - 300 wet
- Split: 90 % training, 5 % validation, 5 % testing

For the object recognition task, a smaller dedicated dataset of 96 images was created due to the manual overhead of bounding box verification. This was split into 80 % training, 10 % validation, and 10 % testing.

For keypoint detection, the dataset was organized into garment-specific groups according to their number of labeled keypoints, ensuring more effective model specialization.

3.2.4 Preprocessing and Augmentation

Subsequently, a comprehensive suite of data augmentation techniques was applied to enhance model robustness and generalization. Geometric transformations, including horizontal flips, rotations, and perspective distortion, were used to simulate diverse cloth orientations and layouts. To address lighting differences, particularly between wet and dry garments, color adjustments such as variations in hue, saturation, and brightness were applied. Finally, advanced augmentation strategies such as Mosaic, MixUp, and Copy-Paste were employed to introduce greater structural diversity, making the models more resilient to real-world challenges like occlusion and background clutter.

- **Instance segmentation & Object detection**: All images were resized to **640 × 640 px** and normalized, following the input specifications of YOLOv11-based architectures.
- **Classification**: Images were prepared at two resolutions (**256 × 256 px** and **500 × 500 px**) to compare the effect of input resolution on accuracy across architectures like VGG16, ResNet, and EfficientNet.
- **Keypoint detection**: Images were resized according to Detectron2's keypoint detection pipeline defaults, preserving aspect ratio while fitting the model's expected input scale.

Given the modest dataset size, a diverse augmentation strategy was applied to simulate real-world variations and improve generalization. Augmentations were grouped into three main categories and applied **inline (during training)** for all tasks, **except for the keypoint detection dataset**, where augmentations were generated **offline** to preserve annotation consistency.

Common augmentation:

1. **Geometric Transformations**

    - Horizontal flips (50 %)
    - Vertical flips (30 %)
    - Rotation (±15°)
    - Translation (20 %)
    - Scaling (±20 %)
    - Shear (10 %)
    - Perspective distortion (15 %)

2. **Color Adjustments**
    - Hue shift (±0.015)
    - Saturation variation (±70 %)
    - Brightness variation (±40 %)
3. **Advanced Augmentations**
    - MixUp (15 %)
    - Copy-Paste (30 %)
    - Random Erasing (20 %)
    - Mosaic (applied for object detection and segmentation tasks)

Representative examples of augmented images for the keypoint detection task are shown in Figure 10, illustrating the applied geometric and color transformations.

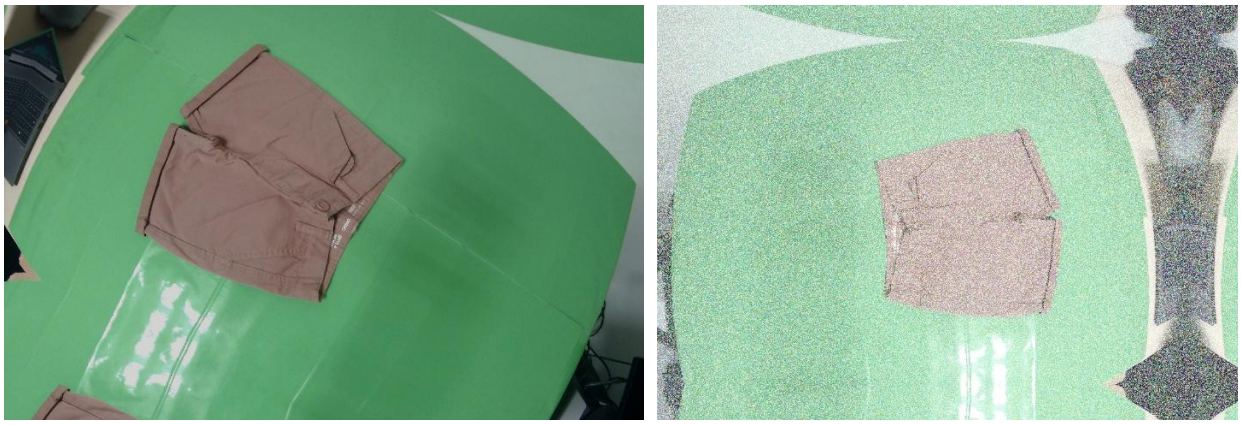

Figure 10 - Augmented samples for key point detection task

Task-specific application:

While the above list summarizes the full augmentation toolbox, in practice the exact combination and intensity of augmentations were **tailored to each vision task**:

- **Instance segmentation**: Prioritized spatial diversity (flips, rotation, scaling, perspective distortion) and moderate color changes to handle lighting variation between wet and dry garments.
- **Object detection**: Used the full augmentation set, with emphasis on composition techniques like Mosaic, MixUp, and Copy-Paste to handle occlusion, clutter, and scale variation.
- **Keypoint detection**: Applied transformations carefully to preserve garment geometry, focusing on controlled rotations, flips, and mild color jitter. For garment groups with fewer samples, augmentation intensity and variety were increased to mitigate class imbalance.

Chapter 4

4 Methodology

This chapter provides an overview of the vision-based pipeline developed for deformable garment manipulation in the EUROBIN project. The system processes RGB images of clothing to extract information that enables robotic manipulation through a multi-stage perception approach. The chapter details each pipeline component, including segmentation, object recognition, keypoint localization, and 3D mesh reconstruction, with emphasis on model selection, training procedures, and evaluation strategies.

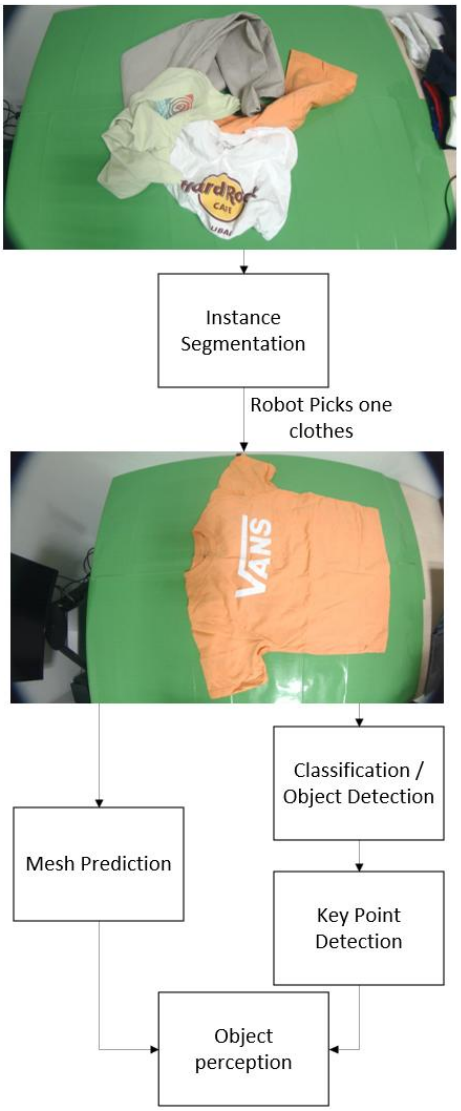

Figure 11 - overall system pipeline stages

The methodology builds upon established approaches in fashion-based computer vision but adapts and extends them to meet the challenges of robotic interaction with real-world garments, including both wet and dry variations. Instead of using a single-task system, this work emphasizes a complete multi-task pipeline capable of analyzing complex visual scenes such as piles of overlapping clothes. Particular attention was given to balancing accuracy with real-time feasibility, which is important for robotic applications.

The proposed system for robotic cloth manipulation implements a structured vision pipeline (see Figure 11) that begins with instance segmentation to separate a pile of clothes into individual garments, followed by object recognition (classification and object detection) to determine each item's category and location. Instance segmentation was approached using lightweight real-time models (YOLOv11-N and YOLOv11-S) and high-accuracy frameworks (Detectron2 with Mask R-CNN), while classification and detection were evaluated with pretrained backbones such as EfficientNet and ResNet and YOLOv11-N. Once the clothing type is identified, a task-specific grouping strategy for keypoint detection locates critical points like corners and edges for grasping. Finally, 3D mesh reconstruction is achieved by combining monocular depth estimation with segmentation and post-processing to produce detailed meshes from RGB input, enabling precise tracking of garment deformation and movement during manipulation.

4.1 Instance segmentation

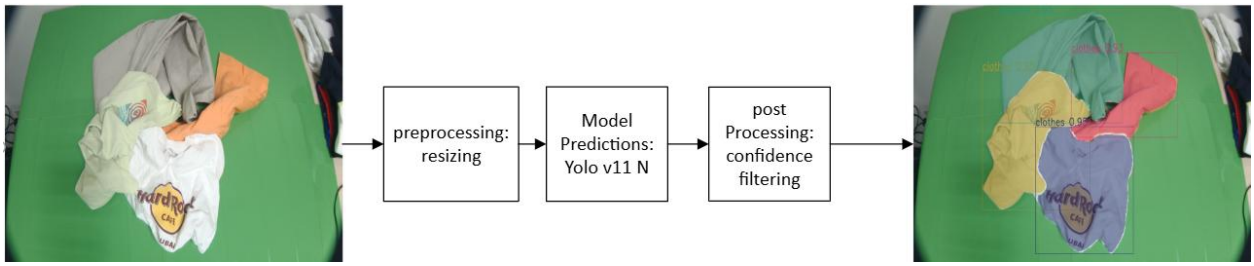

Figure 12 - Instance segmentation task of clothes pile process

The initial and fundamental step within the proposed vision-based pipeline for robotic manipulation of deformable objects is **instance segmentation**. This technique goes beyond simply identifying objects; it aims to detect and create a pixel-level mask for each individual garment in an image. This is particularly critical in scenarios involving cluttered environments, such as a pile of clothes, where garments often overlap and obscure one another. The objective is to provide a comprehensive perception of the clothing items, distinguishing their precise boundaries and locations.

The overall process begins with an input image containing a pile of clothes (Figure 12). This image undergoes a preprocessing step and augmentation, typically involving resizing to a standardized

resolution. The preprocessed image is then passed to a segmentation model, which predicts a mask and a confidence filter is applied to retain only the most reliable detections.

To determine the grasping point for robotic manipulation, a **post-processing** step computes the **maximum inscribed point (MIP)** within the segmentation mask. Unlike the geometric centroid, which may lie near edges or outside the garment in irregular shapes, the MIP represents the pixel **farthest from any boundary point**, ensuring that it is well within the garment's interior. This point is obtained by applying a **Euclidean distance transform** on the binary mask and selecting the location of the maximum distance value. The resulting coordinate provides a stable and reliable target for robotic grasping, reducing the risk of slippage or failed picks during manipulation.

To determine the grasping point, we computed the **Maximum Inscribed Point (MIP)** within each segmentation mask. Given a binary mask $M(x, y)$, the **Euclidean distance transform** $D(x, y)$assigns each pixel its distance to the nearest background pixel:

$$D(x, y) = \min_{(x', y') \in \text{boundary}(M)} \sqrt{(x - x')^2 + (y - y')^2} \qquad (1)$$

The **maximum inscribed point** is then defined as:

$$(x^*, y^*) = \arg \max_{(x,y) \in M} D(x, y) \qquad (2)$$

This point lies deepest inside the mask, ensuring a robust and central grasping location for the robotic arm.

This ability to isolate and target a single, distinct garment through segmentation and MIP extraction serves as the foundation for all subsequent tasks in the pipeline, including classification, keypoint detection, and 3D mesh prediction.

4.1.1 Models and architecture

For the instance segmentation task, a comparative study was conducted using two distinct model families, each selected for its specific strengths in computer vision. The goal was to identify a model that could achieve a robust balance between segmentation accuracy and computational efficiency, a critical requirement for a real-time robotic application.

YOLOv11:

YOLOv11, a prominent single-stage, real-time object detection and segmentation model, was selected primarily for its exceptional speed and efficiency. Its architecture is optimized for fast

inference, making it an ideal candidate for integration into robotic systems where low latency is paramount. To assess the performance, two different versions of the model were evaluated:

- **YOLOv11-N (Nano):** This is the most lightweight variant, designed for minimal computational overhead. It is well-suited for deployment on edge devices and robotics platforms with limited processing resources, offering high throughput at the expense of a slight reduction in accuracy compared to larger models.
- **YOLOv11-S (Small):** This version represents a larger, more powerful variant. It provides a more balanced compromise between speed and accuracy, maintaining real-time capabilities while delivering improved detection and segmentation performance, making it a strong contender for the final pipeline.

Detectron2:

In parallel, the Detectron2 framework, developed by Facebook AI Research, was employed to implement a more complex, two-stage segmentation approach. This framework is widely recognized for delivering high accuracy across diverse computer vision tasks. The specific architecture used was **Mask R-CNN** with a **ResNeXt-101 (X-101-32x8d-FPN-3x)** backbone. This configuration was chosen for its robustness and proven performance on complex instance segmentation tasks. The multi-stage refinement process of Mask R-CNN, which first proposes regions of interest and then refines them, is particularly effective in scenarios with overlapping or deformable objects. Given that the pipeline also includes keypoint detection for garments, a task for which Detectron2's capabilities are well-suited, its inclusion provided a valuable benchmark for performance and a potential unified solution.

4.1.2 Training Details

To achieve robust and generalizable performance, a two-stage training strategy was implemented. The chosen models were first pretrained on the large-scale DeepFashion2 dataset to learn a rich feature representation of garments. Subsequently, the models were fine-tuned on a smaller, custom dataset specifically designed for the instance segmentation task. This dataset consisted of **610 images**, including **310 images of dry clothes** and **300 images of wet clothes**, to ensure the model was robust to real-world conditions. The dataset was partitioned into a 90% training set, a 5% validation set, and a 5% testing set.

All images were preprocessed and augmented as described in Section 3.2.4, including resizing, normalization, and a range of geometric, color-based, and compositional augmentations to enhance model robustness during fine-tuning.

Hyperparameter Configuration:

For YOLOv11-based models, the training was conducted using a batch size of 16, image size of 640 × 640 px, and an initial learning rate of 0.01 with cosine learning rate scheduling and SGD optimizer (momentum = 0.937, weight decay = 0.0005). The training incorporated built-in regularization through flip (0.5), rotation (±10°), HSV-saturation (0.7), HSV-value (0.4), and perspective distortion (0.001) augmentations as defined in the training script.

For the Detectron2 Mask R-CNN model, training was performed with a batch size of 2 images per iteration, a base learning rate of 0.00025, and the Adam optimizer. The maximum iterations were varied (500, 1000, and 2000) to evaluate convergence behavior.

These hyperparameters were empirically selected based on common defaults for each framework and fine-tuned through preliminary experiments to balance stability, speed, and segmentation accuracy.

4.1.3 Evaluation Metrics

For evaluating the performance of the segmentation models, a standard set of metrics was used to ensure a comprehensive assessment:

- **Mean IoU (Intersection over Union):** This metric quantifies how well the predicted segmentation masks match the ground truth masks. It computes the average overlap ratio between the predicted and true masks for all classes.[16]
- **mAP@50 (mean Average Precision at IoU threshold 0.50):** This metric evaluates the performance of object detection and instance segmentation models by calculating the average precision when the Intersection over Union (IoU) threshold is set to 0.50.[16]
- **Precision:** Precision measures the correctness of positive predictions. It represents the proportion of true positives among all instances that the model predicted as positive. High precision indicates few false positives.[59]
    - Precision = $\frac{TP}{TP+FP}$ (3)
- **Recall:** Recall measures the ability of the model to detect all relevant instances. It represents the proportion of actual positives correctly identified by the model. High recall indicates few false negatives.[59]

$$\text{Recall} = \frac{TP}{TP+FN} \quad (4)$$

- **F1-score:** The F1-score combines precision and recall into a single metric by computing their harmonic mean. It is particularly useful when dealing with imbalanced classes or when both false positives and false negatives are important.[59]

$$\text{F1-score} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (5)$$

The models were trained under various configurations to identify the optimal setup. The YOLOv11 N model was trained for both 200 and 299 epochs. The YOLOv11 S model was trained for 300 and 600 epochs. Detectron2 was trained for 500, 1000, and 2000 epochs to observe performance changes over extended training periods.

4.2 Object Recognition

4.2.1 Rationale for Dual-Methodology Investigation

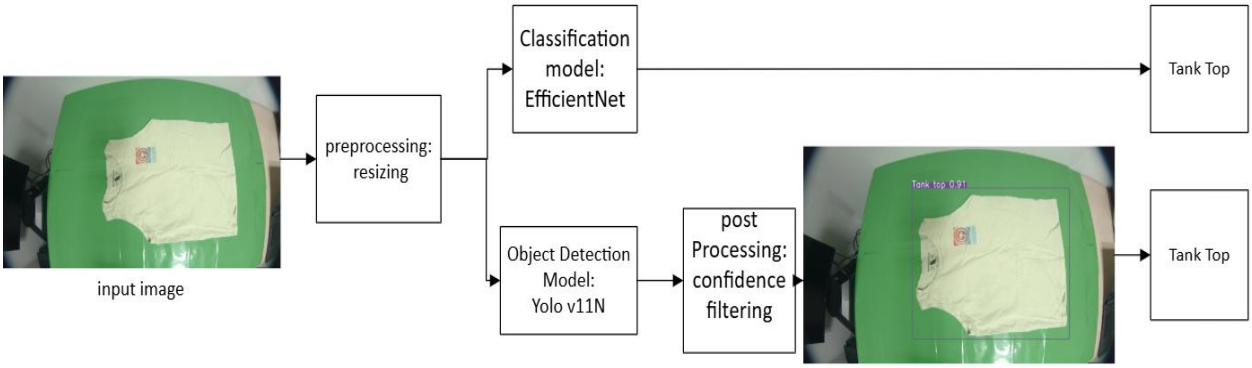

Figure 13 - Classification and Object detection tasks process

Once a single garment is isolated by the instance segmentation module, the next critical step is to identify its category. For the object recognition task, a study using two complementary methodologies was conducted to identify the most effective approach for this particular application. The core question was whether a straightforward image classification model would be sufficient, or if a more complex object detection framework would yield superior performance.

As shown in Figure 13, The first approach, **multi-class image classification**, represents the more traditional path. It treats the entire image of the single garment that has already been physically isolated from the pile as a single entity and assigns it to a category label. This method is computationally simpler but relies on the model learning distinguishing features from the pixel data alone.

The second approach reframes the task as **object detection for classification**. Here, the model's goal is not only to classify the garment but also to localize it with a bounding box. The primary motivation for exploring this method was its potential to better utilize large-scale datasets like DeepFashion2, which contain extensive bounding box annotations. By pretraining on both localization and classification data, the model could potentially learn a more robust and spatially aware feature representation, making it less sensitive to variations in the garment's position or scale within the frame.

Through empirical evaluation of both methodologies, this study seeks to determine the best trade-off between model complexity and classification accuracy, ultimately choosing the most dependable approach for the subsequent stages of the robotic manipulation pipeline.

4.2.2 Multi-Class Image Classification

The first methodology investigated for object recognition was a traditional multi-class image classification approach. In this approach, an image of a single garment is processed as a whole and passed through a deep convolutional neural network (CNN) to classify it into one of the ten predefined categories. The primary advantage of this approach is its relative simplicity and the wide availability of well-studied, high-performing architectures.

4.2.2.1 Model Architectures

To ensure a thorough evaluation, three distinct and influential CNN architectures were selected, each representing a different design philosophy:

- **VGG16:** A classic architecture known for its straightforward design, which uses a deep stack of small (3x3) convolution filters. Its uniform structure has proven effective for many image classification tasks, serving as a solid baseline.
- **ResNet (Residual Network):** This model introduced the concept of "residual connections" or "skip connections," which allow the network to learn residual functions. This innovation effectively mitigates the vanishing gradient problem in very deep networks, enabling the training of much deeper and more powerful models.
- **EfficientNet:** A more modern architecture designed through a principled approach of compound scaling. It systematically balances network depth, width, and resolution to achieve state-of-the-art accuracy with significantly fewer parameters and lower computational cost compared to older models, making it highly efficient.

4.2.2.2 Experimental Setup and Training

A set of experiments were designed to systematically evaluate the performance of these models under different conditions:

- **Image Resolution:** To study the impact of input detail on classification accuracy, all models were trained and evaluated on two different image resolutions: **256x256** and **500x500** pixels.
- **Pretraining Strategy:** The effect of transfer learning was a key point of investigation. For each model and resolution, two training schemes were executed: (1) training **from scratch**

using only the custom-collected dataset, and (2) **pretraining** the model on the large-scale DeepFashion2 dataset before fine-tuning it on the custom dataset of **96 images**.

- **Training Parameters:** All models were trained for a maximum of **100 epochs**. An **early stopping** mechanism was employed to avoid overfitting, halting the training process if no improvement in validation loss was observed over a set number of epochs (patient for 15 epochs). Performance was measured using Test Loss (Categorical Cross-entropy), Accuracy, Precision, and Recall, averaged across all 10 clothing classes.

4.2.3 Object Detection for Classification

The second methodology reframed the recognition task as an object detection problem. Instead of treating the input as a monolithic image for classification, this approach leverages a model that simultaneously localizes the garment with a bounding box and assigns to it a class label. This was hypothesized to be a more robust strategy, as it could capitalize on the extensive localization data available in large-scale pretraining datasets.

4.2.3.1 Model and Rationale

The **YOLOv11-N** model was selected for this approach. The choice was motivated by its demonstrated efficiency and strong performance in the instance segmentation phase, making it a natural candidate for reuse. By employing the same lightweight architecture, the potential for a streamlined, computationally efficient end-to-end pipeline was preserved. The core rationale was that a model pretrained on both object localization and classification would develop a more spatially aware feature representation, leading to better generalization, especially when fine-tuning on a small custom dataset.

4.2.3.2 Training and Fine-Tuning Strategy

A two-stage transfer learning strategy was implemented to maximize performance:

1. **Pretraining:** The YOLOv11-N model was first pretrained on the **DeepFashion2 dataset** for 20 epochs. This initial phase allowed the model to learn a general, robust feature representation for a wide variety of clothing items and their bounding boxes.
2. **Fine-Tuning:** The pretrained model was then fine-tuned on the small, custom dataset of **96 images**. To investigate the impact of training duration, fine-tuning was conducted for two distinct lengths: **100 epochs** and **300 epochs**.

The same comprehensive suite of augmentations from the segmentation task was applied, including geometric transformations (flips, rotation, scaling), color The same comprehensive suite of augmentations from the segmentation task was applied, including geometric transformations

(flips, rotation, scaling), color adjustments (hue, saturation, brightness), and advanced techniques like Mosaic, MixUp, Copy-Paste. This ensured the model was exposed to a wide variety of structural and contextual variations during training.

4.2.4 Evaluation and Comparative Framework

To facilitate a rigorous and impartial comparison between the two distinct methodologies, a unified evaluation framework was essential. Although an object detection model's primary output includes localization data (a bounding box), its efficacy within this specific context is determined by its ability to correctly classify the garment. Consequently, the performance of the object detection approach was benchmarked against the same standard classification metrics used for the dedicated classification models. This systematic application of common evaluation criteria allows for an empirical conclusion on the more effective methodology for the specific challenge of garment recognition in a robotic context.

The performance of all models across both approaches was assessed using the following metrics, calculated on the held-out test set:

- **Test Loss (Categorical Cross-entropy):** This metric provides a measure of model generalization by quantifying the divergence between the predicted probability distribution and the ground-truth distribution of the classes. A lower loss value signifies a model that is better calibrated and generalizes more effectively to unseen data.
- **Accuracy:** A primary metric representing the overall correctness of the model, defined as the ratio of correctly classified instances to the total number of instances in the test set.[59]
- **Precision:** This metric assesses the reliability of positive predictions for each class. It is the ratio of true positives to the sum of true positives and false positives. The reported value is the macro-average precision across all ten classes, providing a measure of performance that is not biased by class imbalance.[59]
- **Recall (Sensitivity):** This metric evaluates the model's ability to identify all relevant instances of each class. It is the ratio of true positives to the sum of true positives and false negatives. As with precision, the macro-average recall is reported to give equal weight to each class's performance.[59]

By systematically applying this common set of evaluation criteria, we can have a clear, data-driven conclusion regarding the superior methodology for the specific challenge of garment recognition in a robotic manipulation context.

4.3 Key point detection

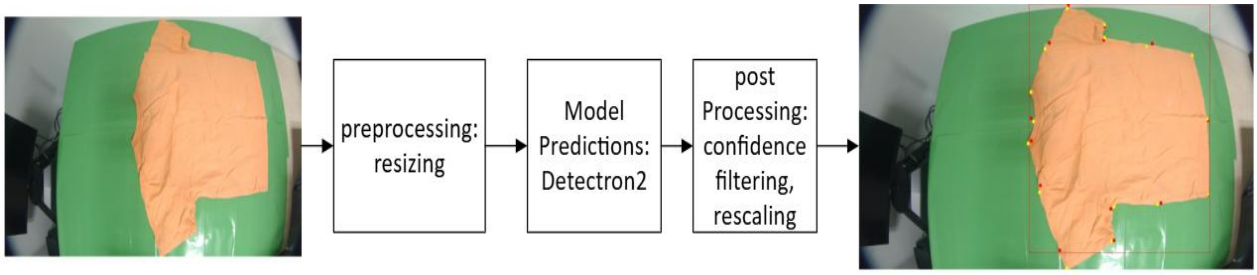

Figure 14 - Key point detection process

Following garment recognition, the next crucial stage in the perception pipeline is keypoint detection. The objective of this task is to localize a predefined set of semantic landmarks on a garment, such as the corners of a collar, the ends of sleeves, or points along a waistline (Figure 9). These keypoints provide a sparse, structured representation of the garment's geometry and pose. For a robotic manipulator, this information is important. It transforms a complex, continuous deformable object into a set of discrete, functionally important coordinates, which are essential for planning robust and precise manipulation strategies, such as defining stable grasp points for picking up a shirt or identifying corners for initiating a folding sequence.

4.3.1 Objective and Rationale for a Grouping-Based Approach

A significant challenge in garment keypoint detection arises from the vast topological diversity across different clothing categories. A model trained to find the 14 keypoints of a t-shirt, for instance, cannot be directly applied to a pair of socks, which may only require 6 keypoints and possesses a fundamentally different structure. Initial experiments confirmed that a single, "one-size-fits-all" model struggled to perform accurately across all 10 categories, exhibiting poor performance especially for classes with unique structures or fewer training samples.

To overcome this limitation, a key methodological innovation of this work was the development of a **task-specific grouping strategy**. Instead of training a unified model, the garment categories were clustered into four distinct groups based on their structural similarity and, most importantly, their total number of keypoints. This approach allowed for the training of separate, specialized models for each group. The underlying rationale is that a model trained on a more homogenous set of objects with a consistent keypoint schema will learn a more focused and accurate feature representation, leading to superior localization performance compared to a generalist model.

4.3.2 Model Architecture and Implementation

To perform keypoint detection, the **Detectron2** framework was selected due to its robust and integrated support for complex computer vision tasks. Specifically, the **keypoint_rcnn_R_50_FPN_3x** architecture was employed. This model is a variant of Mask R-CNN that extends the architecture to simultaneously predict object bounding boxes and a set of keypoints. Its Feature Pyramid Network (FPN) backbone is particularly effective at detecting objects and features at multiple scales, a critical capability when dealing with garments of varying sizes and camera distances. The model was first initialized with weights pretrained on the DeepFashion2 dataset, which offered a solid basis for recognizing garment features prior to fine-tuning on the custom dataset.

4.3.3 Group-Based Training Details

The practical implementation of the keypoint detection methodology was centered on the proposed grouping strategy. The ten garment categories were partitioned into four distinct clusters, each defined by a unique keypoint schema:

- **Group 1 (14 keypoints):** T-shirt, Sweater, Trouser
- **Group 2 (10 keypoints):** Tank Top, Crop Top
- **Group 3 (8 keypoints):** Boxers, Shorts, Briefs
- **Group 4 (6 keypoints):** Long Socks, Skirt

For each cluster, a dedicated Detectron2 model was fine-tuned, enabling the training process to be specialized for the unique challenges of each group. A baseline training protocol was first established for all models, involving fine-tuning for 5,000 epochs while applying a standard suite of 10 data augmentations, such as random rotations and color jitter, to promote generalization.

However, an adaptive training strategy was employed to address the performance variance observed between the groups. For clusters that demonstrated lower initial efficacy, specifically Groups 3 and 4, the training protocol was intensified. The training duration for these models was extended to 10,000 epochs, and a more aggressive set of 20 augmentations was utilized. This enhanced protocol was designed to further diversify the training data and improve model robustness, thereby compensating for the limited number of unique samples in these categories and ensuring that each specialized model was trained to its optimal performance.

4.3.4 Evaluation Framework

To quantitatively assess the performance of the specialized keypoint detection models, a framework based on two standard, complementary metrics was adopted. These metrics provide a comprehensive view of localization accuracy from different perspectives.

- **Object Keypoint Similarity (OKS):** This is the primary metric used for this task and is analogous to the Intersection over Union (IoU) metric in object detection. OKS calculates a score based on the normalized distance between a predicted keypoint and its ground-truth counterpart, scaled by the object's size. This scaling ensures the metric is robust to variations in object size and camera perspective. The score, which varies between 0 and 1, offers a detailed measure of localization accuracy, with higher values reflecting closer alignment.
- **Percentage of Correct Keypoints (PCK):** This metric offers a more intuitive, threshold-based measure of accuracy. A predicted keypoint is assumed "correct" if the Euclidean distance to its corresponding ground-truth annotation falls within a predefined threshold. To ensure scale invariance, this threshold is defined as a fraction of a reference object dimension. In this work, a threshold of 0.2 times a reference distance was used. The final PCK value is the percentage of all keypoints across the test set that satisfy this condition, offering a clear measure of overall model reliability.

Together, OKS and PCK form a robust evaluation framework, allowing for both a detailed analysis of per-keypoint similarity and a high-level understanding of the model's practical accuracy.

4.4 Mesh prediction

The final piece of our system is about building a 3D model of a garment using just a single 2D photo. This ability is vital for the robot to truly understand the clothing's shape, position, and how it might bend or fold, which is necessary for tricky tasks beyond simply picking it up.

While special 3D cameras can measure depth directly, they are often expensive, not always available, and can fail with reflective materials. For these reasons, our goal was to develop a method that can figure out the 3D shape using only a standard camera. This approach makes the system more flexible, affordable, and accessible, as it cleverly uses recent advances in software to create 3D understanding from a simple 2D image, without needing any specialized hardware.

4.4.1 Multi-Stage Reconstruction Pipeline

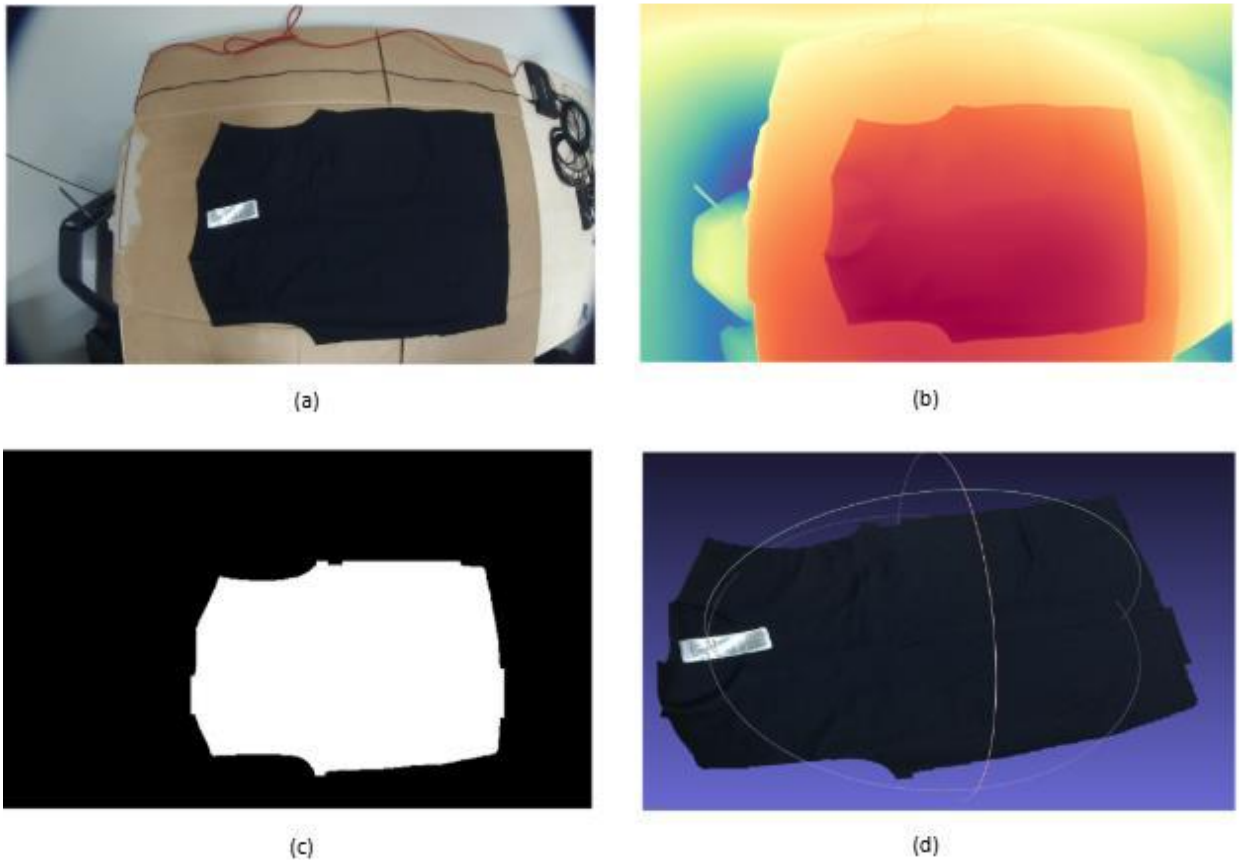

Figure 15 - 3D Mesh Prediction Pipeline from a Single RGB Image.

To achieve 3D reconstruction from a single 2D image, a multi-stage pipeline was designed, as illustrated in Figure 15. This pipeline sequentially processes the image to extract the necessary components for mesh generation:

1. **Monocular Depth Estimation:** The input RGB image is first passed to a state-of-the-art monocular depth estimation model to generate a dense depth map. This map provides a per-pixel estimation of the distance from the camera, forming the initial 3D geometric information.
2. **Garment Segmentation:** To isolate the garment from the background, the binary segmentation mask produced by the previously trained YOLOv11-N model is utilized. This ensures that only the pixels corresponding to the garment of interest are considered for reconstruction.
3. **Mask Refinement and Depth Segmentation:** The binary mask is applied to the depth map, effectively cropping out the depth information for the background. To clean up noisy edges and improve the boundary definition, an erosion operation is applied to the mask before this step.

4. **3D Mesh Reconstruction:** Finally, with the original RGB image (for texture), the segmented depth map (for geometry), and the refined mask (for boundaries), a textured mesh representation of the garment.

4.4.2 Models and Implementation Details

The implementation of the reconstruction pipeline relies on two key pretrained models:

- **Depth Anything V2:** This model was used for the monocular depth estimation step. It is a state-of-the-art foundation model for depth perception that has demonstrated remarkable performance and generalization capabilities across a wide variety of scenes without requiring fine-tuning.
- **Segment Anything Model (SAM):** While the primary segmentation was performed by the YOLOv11-N model, for particularly challenging cases with ambiguous boundaries or complex folds, the SAM was employed to generate more precise binary masks, improving the quality of the final reconstruction.

4.4.3 Evaluation Approach

Due to the absence of ground-truth 3D data for the custom-collected garments, a quantitative evaluation of the mesh prediction accuracy was not feasible. Therefore, no numerical quality scores were computed. Instead, the performance of this component was assessed qualitatively through visual inspection of the generated 3D meshes. The evaluation focused on the overall coherence of the 3D shape, the correctness of the reconstructed topology (e.g., folds and wrinkles), and the realism of the applied texture. While this assessment is inherently subjective, it provides a practical indication of the model's ability to produce visually consistent and plausible 3D representations, which aligns with the exploratory objectives of this work.

Chapter 5

5 Experiments & Results

This chapter reports the experimental findings obtained from the methodologies described in Chapter 4. Each section is dedicated to a core component of the vision pipeline, providing a quantitative and qualitative analysis of the models' performance. The findings are supported by visualizations and performance metrics to offer a comprehensive evaluation of the system's effectiveness for robotic garment manipulation.

5.1 Instance Segmentation Results

The primary objective of the instance segmentation task was to accurately detect and isolate each individual garment from a cluttered pile, a critical first step for any subsequent manipulation task. This section details the performance of the evaluated models both YOLOv11 (Nano and Small variants) and Detectron2 (Mask R-CNN) on the custom-collected dataset.

5.1.1 Quantitative Performance Analysis

To determine the most effective model, a series of experiments were conducted by training each architecture for a varying number of epochs. The performance was measured using Mean Intersection over Union (Mean IoU), mean Average Precision (mAP) at an IoU threshold of 0.50, precision, recall, and the F1-score.

Model/epochs	Mean IOU	MAP (IOU 50)	precision	recall	f1
YOLOv11 N, 200 Epochs	**0.8616**	**1.0000**	**1.000**	**0.9091**	**0.9524**
YOLOv11 N, 300 Epochs	0.8563	0.9855	0.9855	0.8831	0.9315
YOLOv11 S, 300 Epochs	0.8736	0.9851	0.9851	0.8571	0.9167
YOLOv11 S, 600 Epochs	0.8754	0.9853	0.9853	0.8701	0.9241
Detectron2, 500 epochs	0.8927	0.7628	0.9385	0.7444	0.8303
Detectron2, 1000 epochs	0.9001	0.8486	0.9306	0.8701	0.8993
Detectron2, 2000 epochs	0.9107	0.8231	0.8148	0.8571	0.8354

Table 1 Performance Comparison of Instance Segmentation Models - This table presents the performance metrics for the YOLOv11 and Detectron2 models on the held-out test set. The best-performing model configuration is highlighted in bold.

The results, outlined in Table 1, demonstrate that the YOLOv11 N model trained over 200 epochs achieved the best performance for our application. It achieved a perfect **mAP@50 of 1.0000** and the highest **F1-score of 0.9524** among all tested configurations. This suggests an exceptional ability to both correctly localize and segment nearly every garment instance in the test set.

Figure 16 shows the loss curves for both training and validation of the chosen model. The curves indicate that the model trained smoothly, with losses for both sets steadily decreasing and converging, reflecting good generalization. For reference, the loss curves of all other model configurations tested can be found in Appendix A.

The loss curves, depicted in Figure 16, demonstrate a characteristic and intentional drop during the final epochs of the 200-epoch training cycle. This behavior results from the standard training strategy adopted in YOLO-based models, where the **mosaic data augmentation** is disabled in the final phase of training. This practice, implemented through the close_mosaic parameter in the Ultralytics YOLO framework, deactivates mosaic augmentation during the last few epochs (typically the final 10) to allow the model to fine-tune on unaltered images that better resemble the validation and deployment data[60]. During the initial ~190 epochs, the model was trained with mosaic augmentation, which combines four training images into one to increase diversity and robustness. While this technique enhances generalization by exposing the model to occluded and multi-scale objects, it naturally leads to higher loss values. The transition to simpler, non-augmented images in the final phase thus allows the model to refine its weights, leading to the observed sharp decrease across all loss metrics.

A deeper analysis reveals several key insights:

- **YOLOv11 N vs. YOLOv11 S**: While the larger YOLOv11 S model achieved a slightly higher Mean IoU (0.8754 at 600 epochs), it did not surpass the Nano version in the crucial mAP and F1 metrics. Given that the Nano version is significantly more lightweight and computationally efficient, its superior performance on these key metrics makes it the more practical choice.
- **Impact of Training Epochs**: For the YOLOv11 N model, extending the training from 200 to 300 epochs led to a slight decrease in all metrics, suggesting that early stopping at 200 epochs captured the optimal model state before overfitting could occur.

  It should be noted that the second training session of YOLOv11 Nano was set for a maximum of 500 epochs, but it stopped at epoch 300 due to the activation of early stopping. This regularization strategy helps prevent overfitting by tracking the model's performance on the validation set. Training is automatically terminated when a key validation metric, in this case the mask mean Average Precision (mAP50-95), does not improve for a specified number of epochs (patience parameter), indicating that the model has reached an optimal state.

- **Detectron2 Performance**: The Detectron2 models, leveraging the Mask R-CNN architecture, achieved the highest Mean IoU scores overall, with the 2000-epoch model reaching **0.9107**. This indicates a superior capability in fitting the segmentation masks precisely to the ground truth. However, its mAP@50 scores were notably lower than YOLOv11's, peaking at 0.8486. This mismatch implies that while Detectron2 creates high-quality masks when it finds an object, it is less effective at correctly detecting all object instances compared to the YOLOv11 N model in this specific task.

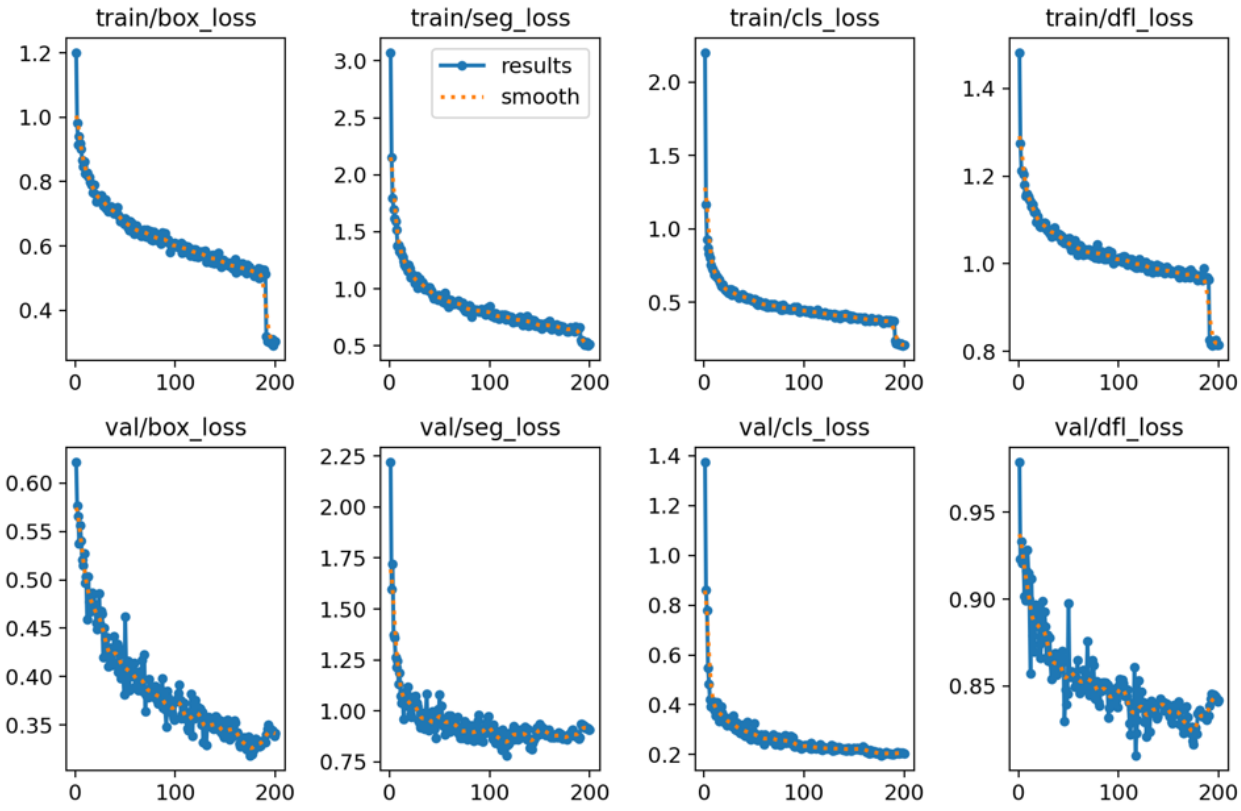

Figure 16- Training and validation loss curves for the YOLOv11-N model trained for 200 epochs. The sharp final decrease is because of mosaic augmentation.

5.1.2 Qualitative Results

Visual inspection of the model outputs on test images supports the quantitative findings. The figures below provide a qualitative comparison of the different models' segmentation capabilities on a representative image of a clothing pile.

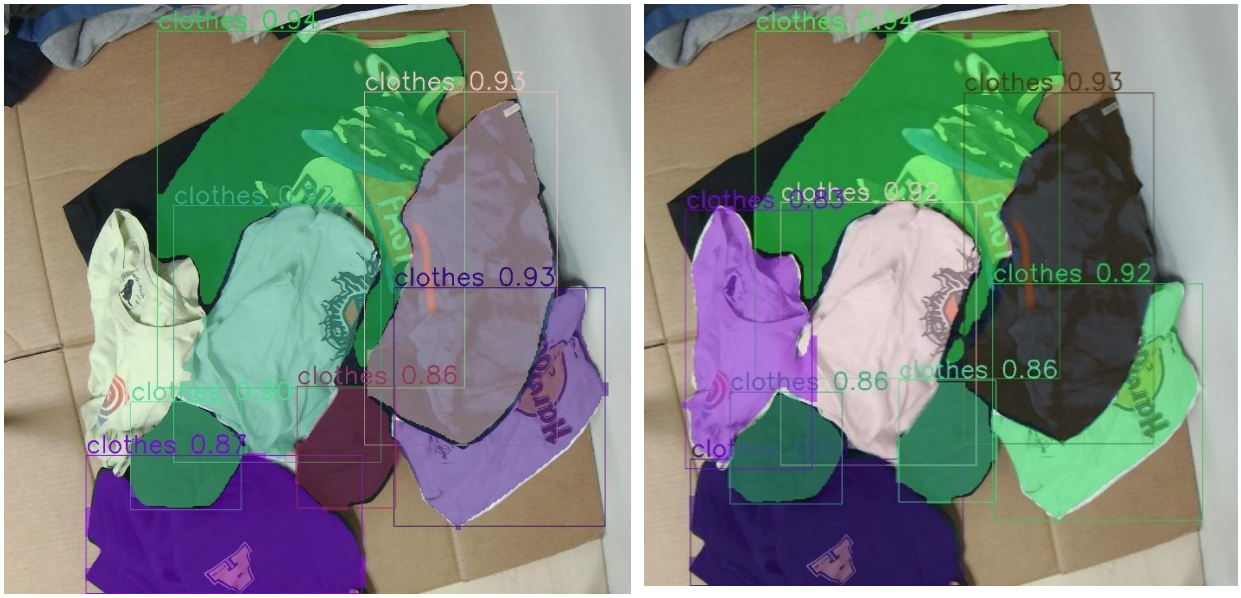

*Figure 17 - sample inference of **YOLOv11n** Model with 200(left) and 299(right) epochs*

*Figure 18 - sample inference of **YOLOv11 S** Model with 300(left) and 600(right) epochs*

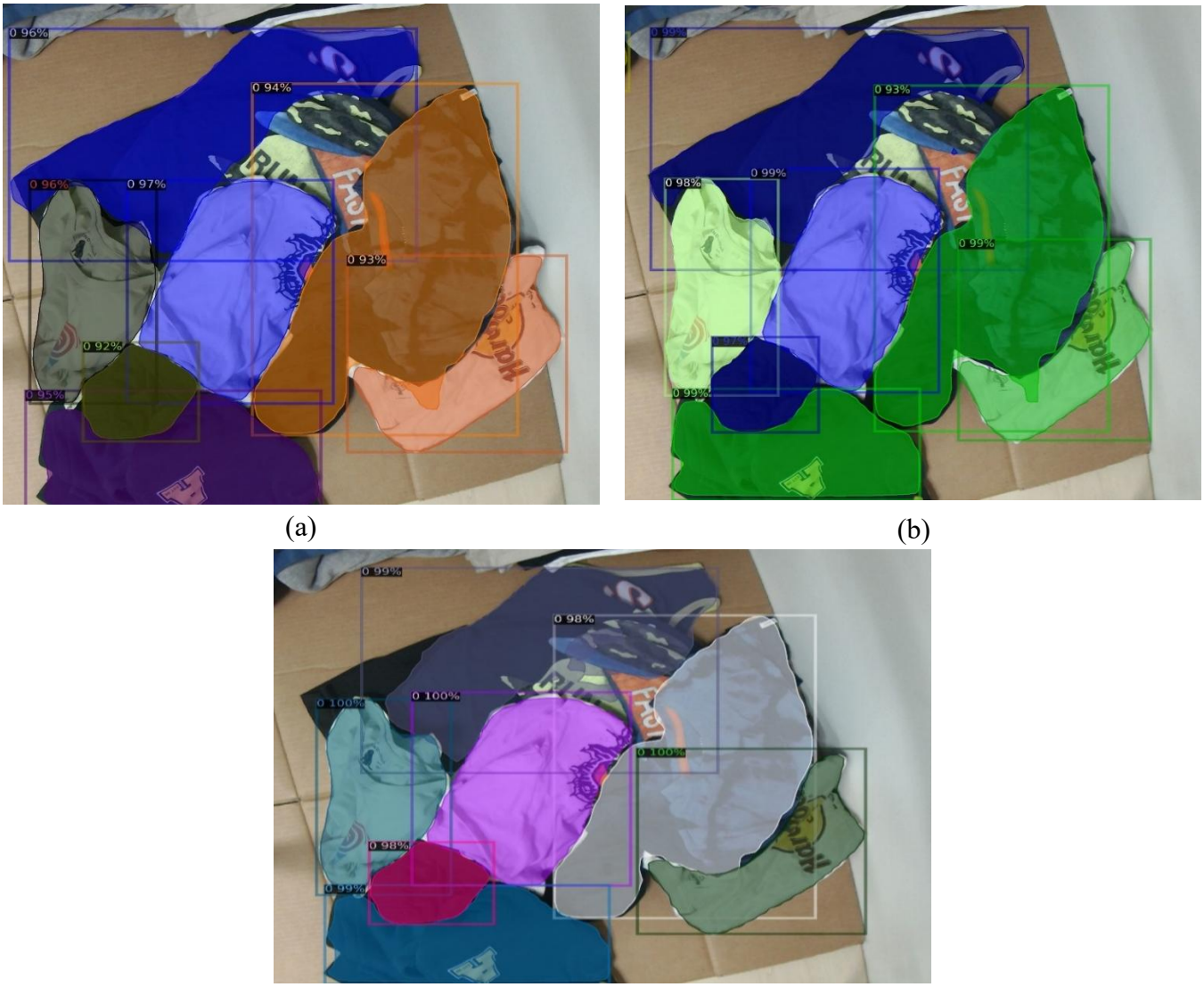

*Figure 19 - **Detectron2** segmentation results after different training durations: (a) 500 epochs, (b) 1000 epochs, and (c) 2000 epochs. Longer training improves the model's ability to fit precise boundaries, but it occasionally misses overlapping items, explaining its lower recall.*

The visual results in Figure 17 show that the YOLOv11 N model effectively segments the overlapping garments with high confidence. Figure 19 demonstrates that while Detectron2 produces very tightly fitted masks (aligning with its high Mean IoU), it occasionally struggles with distinguishing overlapping items or may miss certain garments entirely, which explains its lower recall and mAP scores.

5.1.3 Real-world Demonstration

In addition to quantitative evaluation, we validated the segmentation module in a real robotic setup. A dual-arm robotic platform was tasked with picking garments from a table based on the segmentation outputs. The real-world demonstration reported here used the **maximum inscribed point (MIP)** as the grasp target (see section 4.1). MIP was chosen to maximize clearance from mask boundaries and reduce boundary-only contacts during grasping. Figure 20 shows snapshots from the demonstration, where the robot successfully isolated and grasped individual garments from a cluttered scene. This experiment highlights the practical feasibility of the proposed perception pipeline, demonstrating that the segmentation model can generalize from offline dataset training to real-world robotic manipulation scenarios. While these trials were limited in scope, they provide evidence that the system can serve as a reliable perception front-end for downstream manipulation tasks.

Figure 20 - instance segmentation testing with robotic arms with the Yolo v11 Nano model trained for 200 epochs. The robustness of the model is shown in real world applications.

5.1.4 Discussion and Model Selection

Based on the combined quantitative and qualitative evidence, the **YOLOv11 N model trained for 200 epochs was selected** as the optimal choice for the instance segmentation task.

Its selection is justified by three main factors:

1. **Superior Performance**: It achieved the highest F1-score (0.9524) and a perfect mAP@50, indicating the best overall balance of precision and recall.

2. **Computational Efficiency**: As the "Nano" variant, it offers extremely fast inference speeds with a low computational footprint, which is a critical requirement for integration into a real-time robotic system.
3. **Reliability**: The model showed strong confidence in its predictions and seldom failed to detect garment instances, providing a dependable basis for the following stages of the perception pipeline.

During experimentation, the primary challenges observed were handling severe occlusion and distinguishing between garments of very similar color and texture. While the chosen model performed admirably, these challenges account for the minor imperfections in recall and justify the future work direction of incorporating more diverse and complex pile configurations into the training dataset.

5.2 Object Recognition Results

Following the successful isolation of individual garments via instance segmentation, the next critical stage in the pipeline is object recognition. The goal of this task is to accurately determine the category of each detected garment from the 10 predefined classes. To identify the most effective and robust method for this purpose, a comparative study was conducted between two distinct methodologies: a traditional multi-class image classification approach and an object detection framework repurposed for classification.

5.2.1 Multi-Class Image Classification Performance

This approach treats the entire image of an isolated garment as a single input, leveraging various Convolutional Neural Network (CNN) architectures to assign a category label.

5.2.1.1 Quantitative Performance Analysis

A set of experiments was carried out to assess three widely used CNN architectures: **ResNet, VGG16, and EfficientNet**. The study examined the effects of important factors, such as input image resolution (256×256 versus 500×500) and the application of pretraining on the DeepFashion2 dataset. Model performance was evaluated based on test loss, accuracy, precision, and recall.

Model	Image Size	Pretraining	Test Loss	Accuracy	Precision	Recall
ResNet	256×256	No	1.3161	0.625	0.7647	0.5417
VGG16	256×256	No	1.5049	0.5833	0.6500	0.5417
EfficientNet	256×256	No	1.1156	0.6665	0.7619	0.6666
ResNet	500×500	No	0.9691	0.7083	0.8000	0.6667
VGG16	500×500	No	1.6645	0.4583	0.4615	0.2500
EfficientNet	500×500	No	1.1523	0.5833	0.6471	0.4583
ResNet	500×500	Yes	0.8741	0.7083	0.7778	0.5833
VGG16	500×500	Yes	1.0233	0.5833	0.7222	0.5417
EfficientNet	**500×500**	**Yes**	**0.7398**	**0.7500**	**0.7143**	**0.6250**

*Table 2: **Performance Comparison of Image Classification Models** - This table presents the key performance metrics for the classification models on the held-out test set. The best-performing configuration is highlighted in bold.*

The curves showing training and validation losses of the classification models provide additional insights into their performance. Figure 21 shows the curves for the best-performing configuration, EfficientNet pretrained on DeepFashion2 with 500×500 input resolution. The model exhibits a stable decline in both training and validation loss, with close alignment between the two, indicating good generalization and minimal overfitting. This complements the quantitative metrics reported in Table 2, confirming that EfficientNet benefited substantially from higher-resolution inputs and pretraining. The loss curves for the other classification models (ResNet and VGG16) are included in Appendix B for reference. Comparing the curves of training with and without transfer learning highlights how pretrained models started from a significantly better initial point, achieving lower training loss from the beginning.

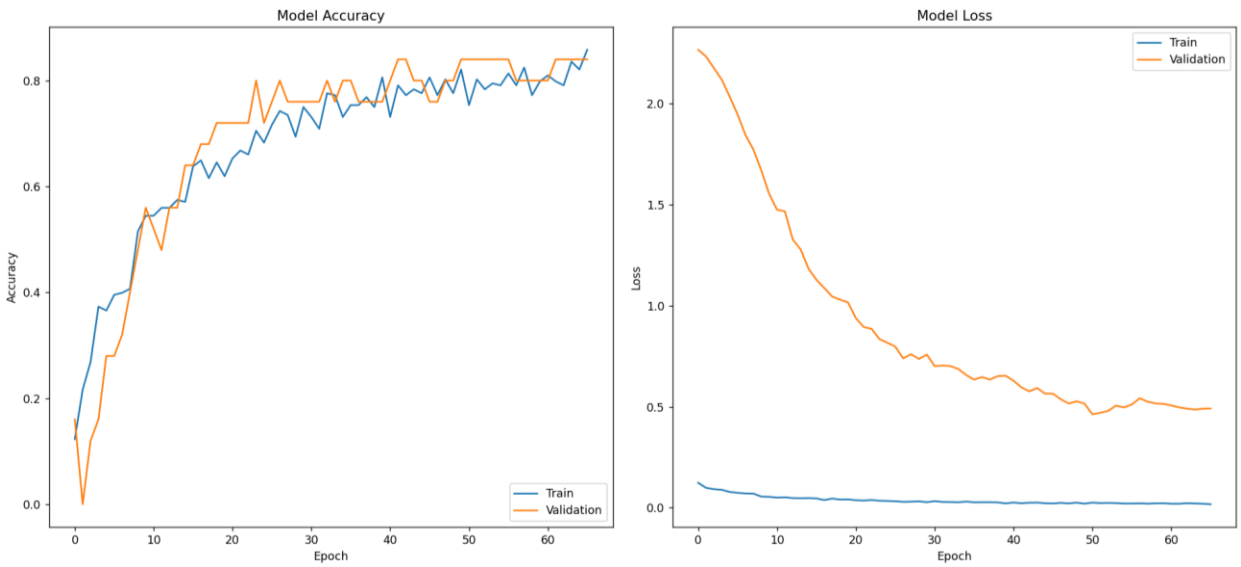

Figure 21 - training and validation loss for EfficientNet with pretrained weights on deep fashion 2 dataset

The results, summarized in Table 2, yield several important insights:

- **Impact of Image Resolution**: Increasing the input image resolution from 256x256 to 500x500 provided a notable performance boost for both ResNet and EfficientNet, underscoring the value of higher-detail input for distinguishing between garment types. VGG16, however, struggled to capitalize on the increased resolution.
- **Effect of Pretraining**: Pretraining on the large-scale DeepFashion2 dataset consistently improved model performance. The most significant gain was observed in the **EfficientNet** model, which achieved the **highest accuracy of 75%** and the lowest test loss of 0.7398 when pretrained on 500x500 images.
- **Model Architecture Comparison**: Across all experiments, EfficientNet and ResNet were strong performers, while VGG16 consistently lagged behind. The superior parameter efficiency and architecture of EfficientNet ultimately gave it the edge, establishing it as the best model within this methodology.

5.2.2 Object Detection for Classification Performance

This second methodology reframes the recognition task as an object detection problem, where the model's goal is to both localize the garment with a bounding box plus a class label. The YOLOv11-N model was selected for this task, leveraging its efficiency and the potential for a more robust, spatially aware feature representation.

5.2.2.1 Quantitative Performance Analysis

The YOLOv11-N model was first pretrained on DeepFashion2 and then fine-tuned on the custom dataset for 100 and 300 epochs.

Model	Epochs	Test Loss	Accuracy	Precision	Recall
YOLO (Fine-tuned)	100	5.5243	0.3636	0.2727	0.3636
YOLO (Fine-tuned)	**300**	**2.3283**	**0.8182**	**0.7727**	**0.8182**

Table 3: Performance of YOLOv11-N for Classification - The results show a dramatic improvement in all metrics when extending the fine-tuning duration.

The results in Table 3 are definitive:

- **Impact of Training Duration**: Extending the fine-tuning process from 100 to 300 epochs resulted in a dramatic improvement in performance. The model's accuracy surged from a modest 36.4% to **81.8%**, while the test loss was more than halved. This indicates that the additional training was crucial for the model to adapt its pretrained features to the particulars of the custom dataset.

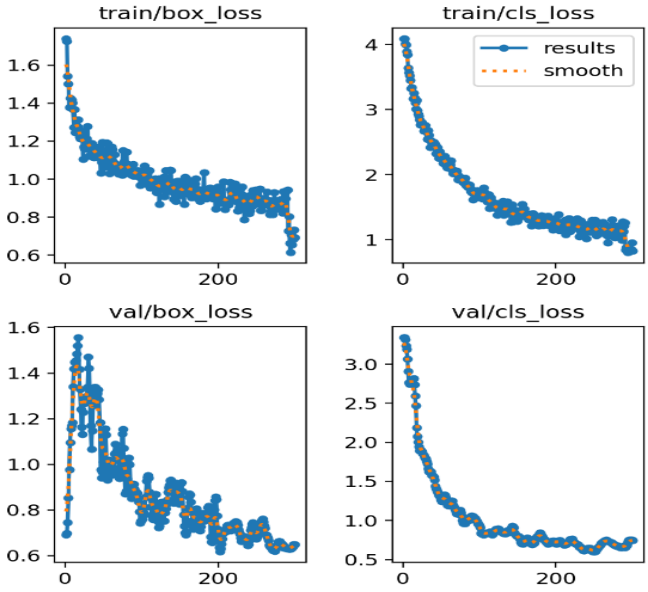

Figure 22 - training and validation loss for the selected model yolo v11 Nano, 300 epochs

The training dynamics of the YOLOv11-N model are illustrated in Figure 22, which presents the learning curves for the model's two fundamental tasks: bounding box localization (box_loss) and garment classification (cls_loss). These graphs provide compelling evidence of a stable and highly successful training process. Both training and validation losses decrease in parallel, indicating

strong generalization and no major signs of overfitting. Like the segmentation, the drop of loss at the final epochs are due to the Mosaic augmentation technique.

The classification loss, which is the most critical metric for this stage of the pipeline, shows an ideal convergence pattern. The validation cls_loss curve closely tracks the decline of the training loss, indicating that the model effectively learned to distinguish between garment categories and successfully generalized this knowledge to unseen data. This is strongly supported by the bounding box loss, which also decreased consistently for both training and validation. The combined success in both minimizing loss for localization and classification explains the model's superior performance and justifies its selection as the final, most robust method.

5.2.3 Discussion and Final Method Selection

Comparing the best-performing models from both methodologies, the YOLOv11-N object detection model fine-tuned for 300 epochs (**81.8% accuracy**) significantly outperformed the best classification model, EfficientNet (**75% accuracy**).

Based on this clear evidence, the YOLOv11-N object detection approach was selected as the final method for the object recognition task. This decision is justified by three key factors:

1. **Higher Accuracy:** The object detection model achieved a notably higher final accuracy score. Training the model to perform both localization and classification at the same time allowed it to learn more effective features for telling the garments apart.
2. **Better Generalization and Reliability:** Both EfficientNet (with pretraining) and YOLOv11-N exhibited stable convergence and minimal overfitting. However, YOLO achieved this while also delivering substantially higher accuracy and more robust validation performance. This makes YOLO not only better at generalization but also the more reliable choice for deployment.
3. **Pipeline Consistency:** Using a YOLO model matches the architecture chosen for the instance segmentation task. This creates a more streamlined and unified system, which could make it easier to deploy and manage in the future.

The primary challenge in this task was the limited size of the custom dataset used for fine-tuning. This lack of data caused the overfitting seen in the pure classification model. The success of the YOLOv11-N model highlights how a good pretraining strategy, combined with the benefit of having the model learn two tasks at once, helped it overcome this problem.

5.3 Keypoint Detection Results

Following the object recognition stage, the keypoint detection models were evaluated to evaluate their ability to localize semantic landmarks on each garment. This task is crucial for enabling downstream robotic manipulation, as these keypoints provide a structured representation of the garment's geometry and the location that the robot can pick and fold the clothes. The evaluation was based on the specialized, group-based approach detailed in the methodology, where separate models were trained for clusters of garments with the same number of keypoints.

5.3.1 Performance of Group-Based Models

The quantitative performance of the specialized models for each garment group was assessed using Object Keypoint Similarity (OKS) and Percentage of Correct Keypoints (PCK). The results, summarized in Table 4, show the effectiveness of the grouping strategy, though performance varied significantly across the different groups.

Test	Group	Training Setup	Overall PCK	Mean OKS
1	**1 (T-shirt, Sweater, Trouser)**	**5000 epochs**	**0.97**	**0.9001**
2	**2 (Tank Top, Crop Top)**	**5000 epochs**	**0.55**	**0.4581**
3	3 (Boxers, Shorts, Briefs)	5000 epochs	1.00	0.8660
4	**3 (Boxers, Shorts, Briefs)**	**5000 epochs more Aug**	**1.00**	**0.8813**
5	3 (Boxers, Shorts, Briefs)	1000 epochs more Aug	0.96	0.8444
6	4 (Long Socks, Skirt)	5000 epochs	0.50	0.2476
7	**4 (Long Socks, Skirt)**	**1000 epochs With more Aug**	**0.50**	**0.3563**

Table 4 - the performance of models for different groups in key point detection tasks. The best performing model in each group is bolded.

5.3.2 Analysis and Observations

A key consideration for this analysis is that the validation set for all groups was very small. Consequently, the validation loss curves appear noisy and are not a fully reliable indicator of the model's final generalization ability. Instead, the combination of final test metrics (Table 4) and visual inspection of inference results provides a more accurate assessment of performance.

The qualitative results, shown in the figures below, align with the quantitative metrics. For high-performing groups like T-shirts and Trousers, the predicted keypoints (red) show a very close alignment with the ground truth annotations (yellow). For underperforming categories like the Skirt, the predictions are less precise, visually confirming the lower OKS scores.

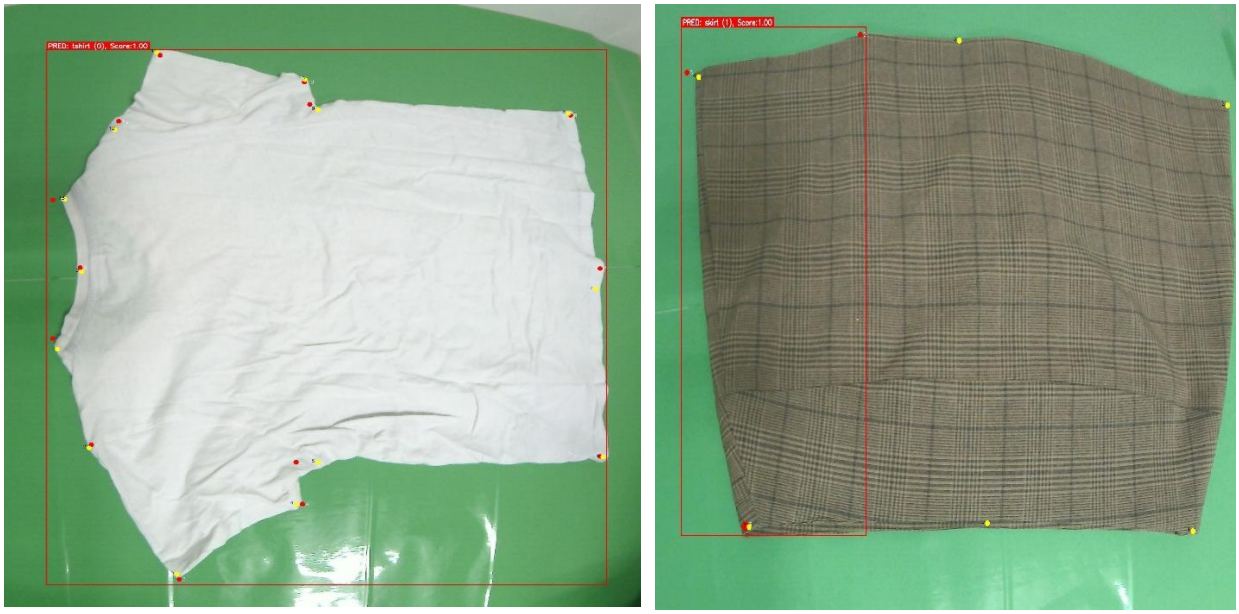

Figure 23 – (left) inference of key point detection for test 1 , group 1 on a t-shirt , (right) inference of key point detection for test 7, group 4 on a skirt, the red points are the predicted ones, and the yellow points are the ground truth

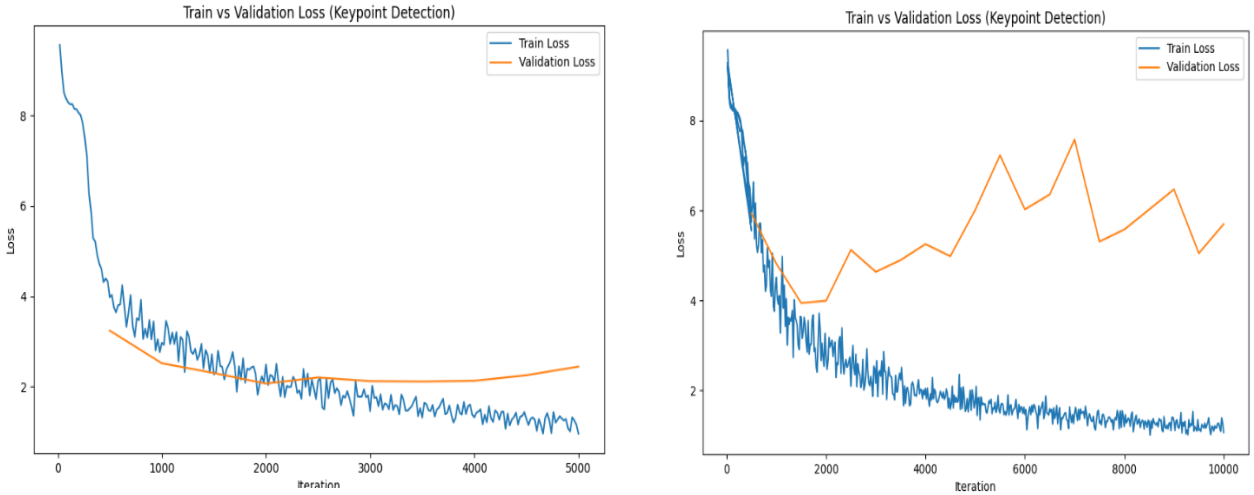

Figure 24 – training loss curves of test 1, group 1 with 5000 epochs (left) and test 7, group 4 with 1000 epochs (right)

The results clearly indicate that model specialization led to high performance for categories with sufficient samples. **Group 1** and **Group 3** achieved excellent results, with the augmented Group 3 model reaching a perfect **PCK of 1.00** and a high Mean OKS of **0.8813**.

This high accuracy is confirmed visually in Figure 23, where predicted keypoints on garments like T-shirts show a near-perfect match with the ground truth. The learning process for these successful groups was also stable, as exemplified by the Group 1 loss curve in Figure 24 (left). In the graph, the train loss (blue line) shows a strong, consistent decrease, which is a clear sign that the model is effectively learning from the training data. The corresponding validation loss (orange line), while noisy, remains stable for most of the training process and only begins to trend slightly upward at the very end. This illustrates a healthy learning dynamic that achieved strong performance before any significant overfitting could occur, a conclusion supported by the excellent final test metrics.

On the other hand, the performance for **Group 2** (Tank Top, Crop Top) and **Group 4** (Long Socks, Skirt) was significantly lower, a result directly attributable to the limited number of training samples for these categories. The less precise predictions for these groups are evident in the example for the skirt in Figure 23(right).

The training dynamics for these struggling models were also visibly unstable, as shown by the Group 4 loss curve in Figure 24(right). The validation loss (orange line) is highly noisy and exhibits an upward trend, a pattern that typically suggests overfitting.

Nevertheless, this conclusion should be approached carefully. Due to the extremely small validation set for these groups, the validation loss curve is not a reliable indicator of the model's true generalization ability. In fact, the performance on the separate, held-out test set tells a different story. As shown in Table 4, extending the training for Group 4 actually improved the Mean OKS from 0.2476 to 0.3563. Actually, the model has low performance for the skirts in group 4 and tank top in group 2. These two categories of clothes have the least samples among all datasets as shown in Figure 8 - chart of distribution of physical clothing samples.

This suggests a complex scenario: while the model was likely beginning to overfit to the few specific examples in the tiny validation set (causing the validation loss to rise), it was simultaneously continuing to learn broader, more useful features that improved its performance on the unseen test data. This highlights a key limitation of relying on small validation sets and confirms that the final test metrics provide the most accurate assessment of the model's practical performance.

This analysis underscores both the success of the group-based training strategy and the critical dependence on sufficient, category-specific training data. While the approach is sound, future work

must focus on increasing dataset for the underrepresented categories to achieve consistent, high accuracy keypoint detection across all garment types.

5.4 3D Mesh Prediction Results

The final component of the perception pipeline focused on reconstructing a three-dimensional mesh of a garment from a single 2D RGB image. This capability is crucial for enabling a robot to understand an object's full spatial properties, which is a prerequisite for executing advanced manipulation tasks such as folding. This section presents the qualitative results of the mesh prediction methodology.

5.4.1 Mesh Reconstruction from a Single RGB Image

The reconstruction process was designed as a multi-stage pipeline that leverages state-of-the-art deep learning models to infer 3D geometry from 2D inputs. As illustrated in Figure 15, the process begins with an RGB image of a garment. A dense depth map is generated using the **Depth Anything V2** model, providing an initial estimation of the object's geometry. Simultaneously, the selected model in segmentation task (Yolo v11 Nano with 200 epochs) produces a segmentation mask to isolate the garment from the background. For instances with particularly complex folds or ambiguous boundaries, the **Segment Anything Model (SAM)** was employed to achieve a more precise segmentation.

The refined mask is then applied to the depth map to isolate the garment's depth information. This segmented data, combined with the original RGB image for texture, is used to reconstruct the final 3D mesh.

5.4.2 Visualizing Mesh Prediction in a Manipulation Scenario

To demonstrate the pipeline's effectiveness in a practical context, the mesh reconstruction was applied to a sequence of images depicting a T-shirt at different stages of a manipulation task: flattened, partially folded, and fully folded. The mesh prediction results, shown in Figures 25, 26, and 27, illustrate the system's ability to capture the changing shape and deformation of the garment throughout the process. This visual evidence confirms that the pipeline can successfully track the garment's topology in various states.

Figure 25 - 3D mesh reconstruction of a T-shirt in a flat state. The pipeline successfully captures garment shape and surface topology from a single RGB image using monocular depth estimation and segmentation.

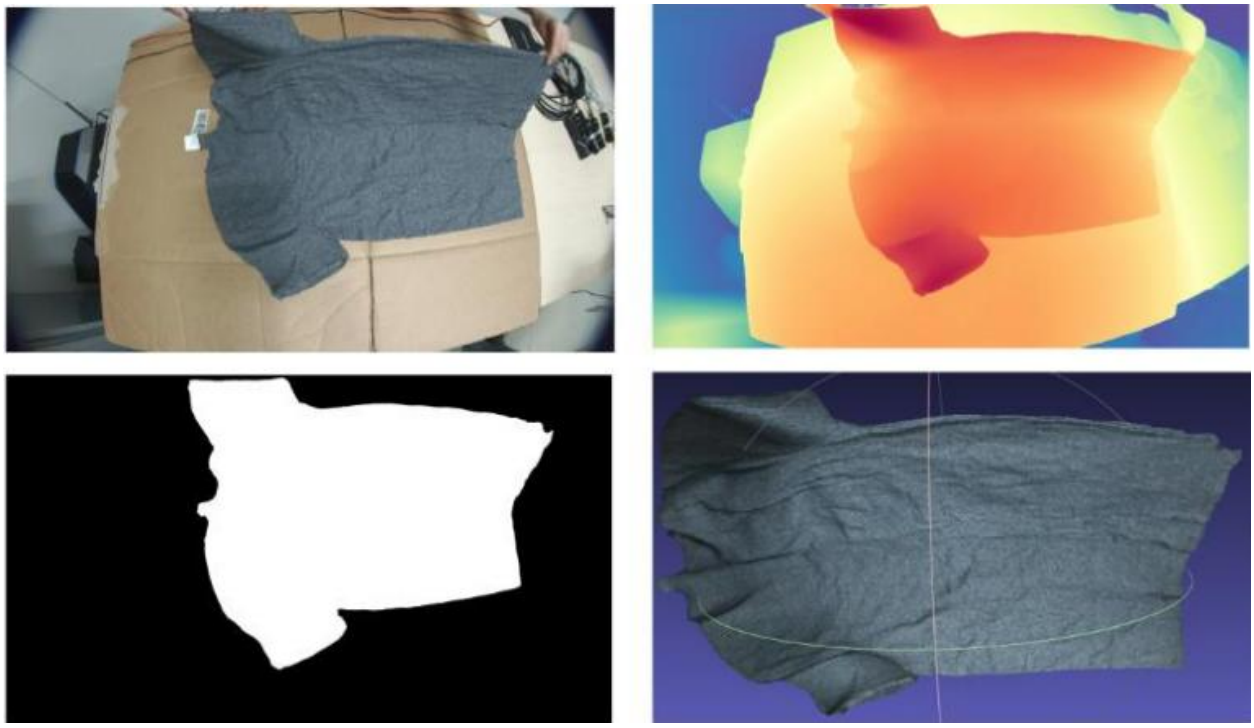

Figure 26 - 3D mesh reconstruction of a T-shirt in a partially folded state. The reconstructed mesh adapts to the garment's changing geometry, demonstrating the pipeline's capability to track deformation.

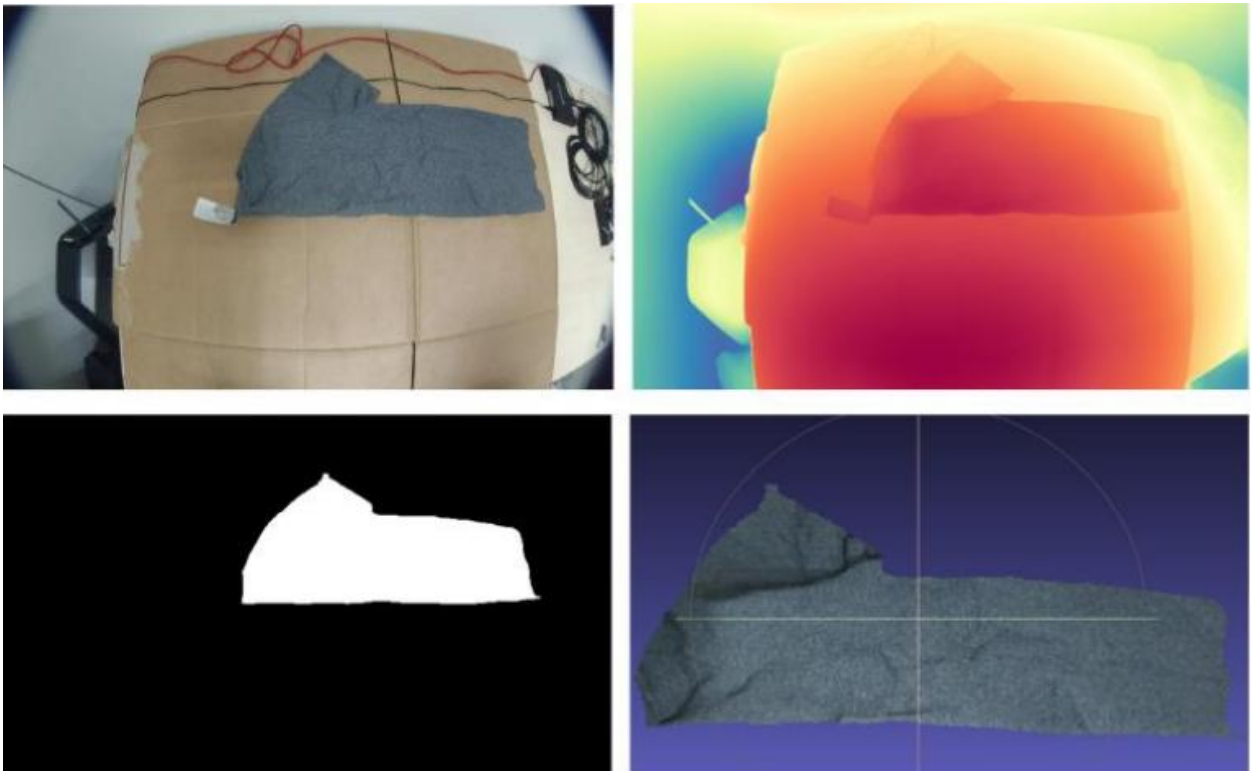

Figure 27 - 3D mesh reconstruction of a T-shirt in a fully folded state. The results show coherent surface representation despite occlusions, confirming the pipeline's effectiveness in handling real-world garment manipulation scenarios.

5.4.3 Qualitative Assessment

Because of the lack of ground-truth 3D data for the custom-collected garments, a quantitative evaluation of the mesh prediction accuracy was not feasible. Therefore, the performance of this component was assessed qualitatively through visual inspection. The generated 3D meshes were evaluated based on the coherence of their overall shape, the accuracy of the reconstructed topology (e.g., correctly representing folds and wrinkles), the absence of noises and the visual quality of the applied texture.

The qualitative assessment confirms that the proposed RGB-only pipeline is capable of generating plausible and coherent 3D representations of garments in various states. The findings highlight the efficiency of this approach in 3D perception for robotic manipulation, providing a strong basis for future research that may incorporate RGB-D sensors to improve accuracy.

5.5 Evaluating the Integrated Perception Pipeline

Although the previous sections examined each element of the perception pipeline separately, its overall value for the EUROBIN project relies on its performance as a fully integrated system. Because the pipeline operates sequentially, the success of each step depends heavily on the output of the preceding step, which can lead to a cascade where errors accumulate and increase.

- **The "Successful Path" and Key Strengths:**

The pipeline's primary strength lies in its "front-end" performance. The selected **YOLOv11-N instance segmentation model** (200 epochs) proved to be highly robust, achieving a perfect mAP@50 and an F1-score of 0.9524. This ensures that the initial and most critical step, isolating a single garment from a cluttered pile, is highly reliable. As demonstrated in preliminary robotic integration tests, this module's accuracy and efficiency are sufficient to guide a robotic arm to successfully pick a target garment from a pile, validating its real-world applicability. Once a garment is isolated, the **YOLOv11-N object recognition model** (300 epochs) provides a strong classification accuracy of 81.8%. When these first two stages succeed, the system correctly passes a correctly identified garment to the appropriate specialized keypoint detection model.

- **Bottlenecks and Error Propagation Analysis:**

The pipeline's overall reliability is constrained by the performance of its subsequent stages, creating two primary failure points:

1. **Object Recognition Errors:** The 81.8% accuracy of the recognition model, while strong, implies that for approximately one in five cases, a garment will be misclassified. This type of error is critical. For example, if a "T-shirt" is misclassified as a "Tank Top," the system will invoke the Group 2 keypoint model (10 keypoints) instead of the correct Group 1 model (14 keypoints). The resulting keypoint predictions would be meaningless and unusable for any downstream manipulation task, causing a complete failure of the pipeline for that item.
2. **Keypoint Detection Inconsistency:** Even with correct classification, the performance of the keypoint detection varies significantly across garment groups. For common categories like T-shirts and Trousers (Group 1), the model is highly reliable, with a PCK of 0.97 and a Mean OKS of 0.90. However, for underrepresented categories like Skirts (Group 4), the performance is poor (Mean OKS of 0.3563). Therefore, even if a skirt is correctly segmented and identified, the pipeline would likely fail to provide the accurate grasp points necessary for manipulation.

In summary, the integrated pipeline is highly effective for a significant portion of common garment types where data is plentiful. However, its overall reliability is currently bottlenecked by the object recognition stage and the inconsistent performance of keypoint detection on less-frequent clothing

categories. The system's architecture is sound, but its end-to-end success rate is dictated by its weakest links.

Chapter 6

6 Conclusion and Future Work

6.1 Conclusion

The research presented in this thesis has addressed the problem of vision-based robotic cloth manipulation, with the objective of building and validating a perception pipeline able of identifying, localizing, and reconstructing garments from a small, custom dataset. The work focused on four interconnected tasks: instance segmentation, object recognition, keypoint detection, and 3D mesh prediction. Together, these tasks form the essential building blocks of a pipeline that was successfully demonstrated within the EUROBIN robotic system for autonomous textile handling.

This research achieved several significant results. Firstly, a custom dataset was collected and annotated which has ten garment categories under both wet and dry conditions. Although relatively small in size, this dataset is as a valuable benchmark for training and evaluating models on diverse perception tasks.

In the area of instance segmentation, extensive experiments with YOLOv11 and Detectron2 showed that YOLOv11-N trained for 200 epochs provided the most balanced solution, combining near-perfect mAP@50 with excellent recall and efficiency. This initial stage proved highly robust in physical experiments with robotic in real world application, consistently enabling a robotic manipulator to accurately distinguish and pick a target garment from a cluttered pile, validating its real-world applicability.

For object recognition, we compared traditional classification networks with an object detection method. While EfficientNet, pretrained on DeepFashion2 and trained on our dataset, showed strong generalization and reasonable performance, YOLOv11-N once again demonstrated superior accuracy and reliability, ultimately making it the selected choice for this phase of the pipeline.

To address the challenge of reliably identifying manipulation landmarks, we used a group-based keypoint detection approach to have different models according to the specific structures of different garments. This strategy proved particularly effective for categories with sufficient training examples, such as T-shirts and trousers, where PCK and OKS scores approached high and acceptable accuracy. While performance was lower for less-represented categories, such as skirts and crop tops, the experiments still demonstrated that the specialized model framework is feasible for robotic manipulation and highlighted how important dataset balance is for achieving strong performance.

Finally, this thesis introduced a pipeline for predicting 3D garment meshes using only RGB images, combining monocular depth estimation, segmentation masks, and reconstruction techniques. Although the evaluation was necessarily qualitative due to the absence of ground-truth meshes, the results showed that it is possible to generate plausible cloth reconstructions without specialized hardware. Overall, these contributions provide a complete proof of concept for a robotic perception system capable of manipulating deformable objects, indicating that the YOLO-based architectures are effective, also showing the benefits of transfer learning, and the importance of a modular design in creating a robust pipeline.

6.2 Limitations

While the thesis establishes a promising foundation, several limitations restrict the scope and generalizability of its findings. The most significant constraint lies in the size and diversity of the dataset. With only 51 garments represented and considerable imbalance between categories, some models, particularly for keypoint detection, struggled to generalize to less frequent classes. Wet garments were included to increase variability, but challenges such as reflections and lighting effects remained unresolved.

Another limitation concerns the evaluation of the 3D mesh prediction pipeline. Without access to ground-truth 3D data, performance could only be assessed visually, making it difficult to quantify accuracy in capturing fine details such as folds or fabric thickness. In addition, we could not train any model for mesh prediction for clothes specifically and we had to use pretrained models like Depth Any Thing. The reliance on monocular RGB input also imposed constraints, as depth estimation from single images remains prone to artifacts and distortions.

6.3 Future Work

The outcomes of this study point to multiple promising directions for future research. The first step would be to expand the dataset and especially increase the samples of underrepresented classes, since broader and more balanced data would directly address current limits in model generalization. Collecting a larger set of garments that spans different poses, fabrics, and environmental conditions would strengthen the pipeline considerably. At the same time, synthetic data generated through garment simulations or rendering could be used to complement real samples, especially for underrepresented categories, while also allowing for experiments under more controlled conditions.

Building on the initial successful integration with the EUROBIN platform, a clear next step is to enhance the closed-loop control system. A real-world test with two robotic arms illustrates how perception translates to successful grasps; future work should focus on using the pipeline's visual feedback in real-time to create adaptive strategies during manipulation of the clothes. For instance,

the robot could dynamically adjust its folding trajectory based on continuous feedback from the keypoint detection and mesh prediction modules. That kind of integration would reveal the robustness of the pipeline under real-time constraints and unstructured conditions.

Future work could also explore the use of RGB-D fusion. By introducing depth cameras, the pipeline could achieve more accurate mesh prediction while reducing the reliance on monocular depth estimation. Fusion of color and depth information is likely to improve robustness under challenging lighting and occlusion scenarios.

To improve the performance of specific pipeline modules, several targeted enhancements could be pursued. For object recognition, techniques like automated background removal could reduce noise and improve classification accuracy. For keypoint detection, developing more targeted data augmentation strategies or even re-evaluating the keypoint definitions for structurally ambiguous garments like skirts could yield significant gains.

Advances in keypoint detection offer another direction. Transformer-based or graph-based models could better capture the structural relationships between garment parts, while semi-supervised learning might reduce annotation costs. For 3D mesh prediction, future efforts should aim at quantitative benchmarking by creating a small test set with ground-truth 3D scans and training using real dataset with depth information. Real-time mesh refinement during manipulation could eventually allow the robot to adapt its perception dynamically as the garment deforms.

Finally, the pipeline’s modular design itself points toward a promising future in end-to-end architectures. We could develop a multi-task network to perform segmentation, recognition, and keypoint detection in unison, thus reducing latency through shared features. This integration would be a major advantage for deployment on robotic systems, which critically depend on speed and robustness.

6.4 Concluding Remarks

This thesis has demonstrated that vision-based methods can serve as a practical basis for robotic cloth manipulation. Through a systematic evaluation of different architectures and strategies, it has shown the effectiveness of YOLOv11 models in both segmentation and recognition tasks, highlighted the advantages of transfer learning for classification, and confirmed the feasibility of extracting manipulation landmarks and generating 3D reconstructions from limited data. The project still faces key obstacle, namely the dataset's limited diversity, the need for more rigorous 3D mesh evaluation, and the challenge of practical pipeline integration. Nonetheless, the outcomes lay down a firm groundwork for future investigations. With larger datasets, the use of depth sensors, and integration into robotic systems, the pipeline proposed here could develop into a

reliable perception framework for garment handling, moving the field closer to the broader goal of autonomous textile manipulation.

Bibliography

[1] F. Gu, Y. Zhou, Z. Wang, S. Jiang, and B. He, “A Survey on Robotic Manipulation of Deformable Objects: Recent Advances, Open Challenges and New Frontiers,” Dec. 16, 2023, *arXiv*: arXiv:2312.10419. doi: 10.48550/arXiv.2312.10419.

[2] T. Lips, V.-L. D. Gusseme, and F. wyffels, “Learning Keypoints for Robotic Cloth Manipulation using Synthetic Data,” May 21, 2024, *arXiv*: arXiv:2401.01734. doi: 10.48550/arXiv.2401.01734.

[3] T. Ziegler, J. Butepage, M. C. Welle, A. Varava, T. Novkovic, and D. Kragic, “Fashion Landmark Detection and Category Classification for Robotics,” Mar. 26, 2020, *arXiv*: arXiv:2003.11827. doi: 10.48550/arXiv.2003.11827.

[4] I. Garcia-Camacho, A. Longhini, M. Welle, G. Alenyà, D. Kragic, and J. Borràs, “Standardization of Cloth Objects and its Relevance in Robotic Manipulation,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, May 2024, pp. 8298–8304. doi: 10.1109/ICRA57147.2024.10610630.

[5] A. Longhini *et al.*, “EDO-Net: Learning Elastic Properties of Deformable Objects from Graph Dynamics,” Dec. 20, 2024, *arXiv*: arXiv:2209.08996. doi: 10.48550/arXiv.2209.08996.

[6] A. Longhini, M. C. Welle, Z. Erickson, and D. Kragic, “AdaFold: Adapting Folding Trajectories of Cloths via Feedback-loop Manipulation,” Dec. 20, 2024, *arXiv*: arXiv:2403.06210. doi: 10.48550/arXiv.2403.06210.

[7] A. Longhini *et al.*, “Unfolding the Literature: A Review of Robotic Cloth Manipulation,” *Annu. Rev. Control Robot. Auton. Syst.*, vol. 8, no. 1, pp. 295–322, May 2025, doi: 10.1146/annurev-control-022723-033252.

[8] T. Lips, V.-L. De Gusseme, and F. Wyffels, “Learning Keypoints for Robotic Cloth Manipulation Using Synthetic Data,” *IEEE Robot. Autom. Lett.*, vol. 9, no. 7, pp. 6528–6535, July 2024, doi: 10.1109/LRA.2024.3405335.

[9] H. A. Kadi and K. Terzić, “Data-Driven Robotic Manipulation of Cloth-like Deformable Objects: The Present, Challenges and Future Prospects,” *Sensors*, vol. 23, no. 5, p. 2389, Feb. 2023, doi: 10.3390/s23052389.

[10] D. Blanco-Mulero, O. Barbany, G. Alcan, A. Colomé, C. Torras, and V. Kyrki, “Benchmarking the Sim-to-Real Gap in Cloth Manipulation,” *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2981–2988, Mar. 2024, doi: 10.1109/LRA.2024.3360814.

[11] X. Zhang *et al.*, “DiffCloth: Diffusion Based Garment Synthesis and Manipulation via Structural Cross-modal Semantic Alignment,” in *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, Paris, France: IEEE, Oct. 2023, pp. 23097–23106. doi: 10.1109/ICCV51070.2023.02116.

[12] A. M. Hafiz and G. M. Bhat, “A survey on instance segmentation: state of the art,” *Int. J. Multimed. Inf. Retr.*, vol. 9, no. 3, pp. 171–189, Sept. 2020, doi: 10.1007/s13735-020-00195-x.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” Jan. 24, 2018, *arXiv*: arXiv:1703.06870. doi: 10.48550/arXiv.1703.06870.

[14] T.-Y. Lin *et al.*, “Microsoft COCO: Common Objects in Context,” Feb. 21, 2015, *arXiv*: arXiv:1405.0312. doi: 10.48550/arXiv.1405.0312.

[15] Y. Chuang, S. Zhang, and X. Zhao, “Deep learning-based panoptic segmentation: Recent advances and perspectives,” *IET Image Process.*, vol. 17, no. 10, pp. 2807–2828, Aug. 2023, doi: 10.1049/ipr2.12853.

[16] W. Gu, S. Bai, and L. Kong, “A review on 2D instance segmentation based on deep neural networks,” *Image Vis. Comput.*, vol. 120, p. 104401, Apr. 2022, doi: 10.1016/j.imavis.2022.104401.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, June 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[18] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, “YOLACT: Real-Time Instance Segmentation,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South): IEEE, Oct. 2019, pp. 9156–9165. doi: 10.1109/ICCV.2019.00925.

[19] O. Nocentini, J. Kim, M. Z. Bashir, and F. Cavallo, “Image Classification Using Multiple Convolutional Neural Networks on the Fashion-MNIST Dataset,” *Sensors*, vol. 22, no. 23, p. 9544, Dec. 2022, doi: 10.3390/s22239544.

[20] O. Gustavsson, T. Ziegler, M. C. Welle, J. Bütepage, A. Varava, and D. Kragic, “Cloth manipulation based on category classification and landmark detection,” *Int. J. Adv. Robot. Syst.*, vol. 19, no. 4, p. 17298806221110445, July 2022, doi: 10.1177/17298806221110445.

[21] C. Li, T. Fu, F. Li, and R. Song, “Design and Implementation of Fabric Wrinkle Detection System Based on YOLOv5 Algorithm,” *Cobot*, vol. 3, p. 5, July 2024, doi: 10.12688/cobot.17687.1.

[22] M. H. M. Noor and A. O. Ige, “A Survey on State-of-the-art Deep Learning Applications and Challenges,” *Eng. Appl. Artif. Intell.*, vol. 159, p. 111225, Nov. 2025, doi: 10.1016/j.engappai.2025.111225.

[23] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” Apr. 10, 2015, *arXiv*: arXiv:1409.1556. doi: 10.48550/arXiv.1409.1556.

[24] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” Dec. 10, 2015, *arXiv*: arXiv:1512.03385. doi: 10.48550/arXiv.1512.03385.

[25] A. G. Howard *et al.*, “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” Apr. 17, 2017, *arXiv*: arXiv:1704.04861. doi: 10.48550/arXiv.1704.04861.

[26] C. Szegedy *et al.*, “Going deeper with convolutions,” in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA: IEEE, June 2015, pp. 1–9. doi: 10.1109/CVPR.2015.7298594.

[27] X. Liu, “Comparison of Four Convolutional Neural Network-Based Algorithms for Sports Image Classification,” in *Proceedings of the 2023 International Conference on Data Science, Advanced Algorithm and Intelligent Computing (DAI 2023)*, vol. 180, B. H. Ahmad, Ed., in Advances in Intelligent Systems Research, vol. 180. , Dordrecht: Atlantis Press International BV, 2024, pp. 178–186. doi: 10.2991/978-94-6463-370-2_20.

[28] D. F. Gomes, S. Luo, and L. F. Teixeira, “GarmNet: Improving Global with Local Perception for Robotic Laundry Folding,” June 30, 2019, *arXiv*: arXiv:1907.00408. doi: 10.48550/arXiv.1907.00408.

[29] M. Tan and Q. V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” Sept. 11, 2020, *arXiv*: arXiv:1905.11946. doi: 10.48550/arXiv.1905.11946.

[30] W. Qiu, C. Xie, and J. Huang, “An improved EfficientNetV2 for garbage classification,” Mar. 27, 2025, *arXiv*: arXiv:2503.21208. doi: 10.48550/arXiv.2503.21208.

[31] M. Tan and Q. V. Le, “EfficientNetV2: Smaller Models and Faster Training”.

[32] M. Mao and M. Hong, “YOLO Object Detection for Real-Time Fabric Defect Inspection in the Textile Industry: A Review of YOLOv1 to YOLOv11,” *Sensors*, vol. 25, no. 7, p. 2270, Apr. 2025, doi: 10.3390/s25072270.

[33] A. Wong, M. Famuori, M. J. Shafiee, F. Li, B. Chwyl, and J. Chung, “YOLO Nano: a Highly Compact You Only Look Once Convolutional Neural Network for Object Detection,” Oct. 03, 2019, *arXiv*: arXiv:1910.01271. doi: 10.48550/arXiv.1910.01271.

[34] C.-H. Lee and C.-W. Lin, “A Two-Phase Fashion Apparel Detection Method Based on YOLOv4,” *Appl. Sci.*, vol. 11, no. 9, p. 3782, Apr. 2021, doi: 10.3390/app11093782.

[35] T. Lips, V.-L. D. Gusseme, and F. wyffels, “Learning Keypoints from Synthetic Data for Robotic Cloth Folding,” May 13, 2022, *arXiv*: arXiv:2205.06714. doi: 10.48550/arXiv.2205.06714.

[36] S. Miller, J. Van Den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, “A geometric approach to robotic laundry folding,” *Int. J. Robot. Res.*, vol. 31, no. 2, pp. 249–267, Feb. 2012, doi: 10.1177/0278364911430417.

[37] S. Yan, Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, “Unconstrained Fashion Landmark Detection via Hierarchical Recurrent Transformer Networks,” in *Proceedings of the 25th ACM international conference on Multimedia*, Mountain View California USA: ACM, Oct. 2017, pp. 172–180. doi: 10.1145/3123266.3123276.

[38] S. Janampa and M. Pattichis, “DETRPose: Real-time end-to-end transformer model for multi-person pose estimation,” June 16, 2025, *arXiv*: arXiv:2506.13027. doi: 10.48550/arXiv.2506.13027.

[39] Y. Xu, J. Zhang, Q. Zhang, and D. Tao, “ViTPose++: Vision Transformer for Generic Body Pose Estimation,” Dec. 14, 2023, *arXiv*: arXiv:2212.04246. doi: 10.48550/arXiv.2212.04246.

[40] bitsauce, “keypoint_RCNN: An implementation of Mask R-CNN on PyTorch.” [Online]. Available: https://github.com/bitsauce/Keypoint_RCNN

[41] Y. Ge, R. Zhang, X. Wang, X. Tang, and P. Luo, “DeepFashion2: A Versatile Benchmark for Detection, Pose Estimation, Segmentation and Re-Identification of Clothing Images,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA: IEEE, June 2019, pp. 5332–5340. doi: 10.1109/CVPR.2019.00548.

[42] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, “DeepFashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA: IEEE, June 2016, pp. 1096–1104. doi: 10.1109/CVPR.2016.124.

[43] A. Doumanoglou *et al.*, “Folding Clothes Autonomously: A Complete Pipeline,” *IEEE Trans. Robot.*, vol. 32, no. 6, pp. 1461–1478, Dec. 2016, doi: 10.1109/TRO.2016.2602376.

[44] B. Zhou *et al.*, “ClothesNet: An Information-Rich 3D Garment Model Repository with Simulated Clothes Environment,” Aug. 19, 2023, *arXiv*: arXiv:2308.09987. doi: 10.48550/arXiv.2308.09987.

[45] Y. Deng and D. Hsu, “General-purpose Clothes Manipulation with Semantic Keypoints,” Mar. 26, 2025, *arXiv*: arXiv:2408.08160. doi: 10.48550/arXiv.2408.08160.

[46] G. Gkioxari, J. Malik, and J. Johnson, “Mesh R-CNN,” Jan. 25, 2020, *arXiv*: arXiv:1906.02739. doi: 10.48550/arXiv.1906.02739.

[47] F. Nolte, A. Geiger, B. Schölkopf, and I. Posner, “Is Single-View Mesh Reconstruction Ready for Robotics?,” Aug. 11, 2025, *arXiv*: arXiv:2505.17966. doi: 10.48550/arXiv.2505.17966.

[48] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, “Pixel2Mesh: Generating 3D Mesh Models from Single RGB Images,” Aug. 03, 2018, *arXiv*: arXiv:1804.01654. doi: 10.48550/arXiv.1804.01654.

[49] Q. Ma *et al.*, “Learning to Dress 3D People in Generative Clothing,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Seattle, WA, USA: IEEE, June 2020, pp. 6468–6477. doi: 10.1109/CVPR42600.2020.00650.

[50] H. Zhu *et al.*, “Deep Fashion3D: A Dataset and Benchmark for 3D Garment Reconstruction from Single Images,” July 04, 2020, *arXiv*: arXiv:2003.12753. doi: 10.48550/arXiv.2003.12753.

[51] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3D-R2N2: A Unified Approach for Single and Multi-view 3D Object Reconstruction,” Apr. 02, 2016, *arXiv*: arXiv:1604.00449. doi: 10.48550/arXiv.1604.00449.

[52] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, “Occupancy Networks: Learning 3D Reconstruction in Function Space,” Apr. 30, 2019, *arXiv*: arXiv:1812.03828. doi: 10.48550/arXiv.1812.03828.

[53] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, “DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation,” in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA: IEEE, June 2019, pp. 165–174. doi: 10.1109/CVPR.2019.00025.

[54] S. Saito, Z. Huang, R. Natsume, S. Morishima, A. Kanazawa, and H. Li, “PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization,” Dec. 03, 2019, *arXiv*: arXiv:1905.05172. doi: 10.48550/arXiv.1905.05172.

[55] Y. Wang *et al.*, “GarmentCrafter: Progressive Novel View Synthesis for Single-View 3D Garment Reconstruction and Editing,” Mar. 11, 2025, *arXiv*: arXiv:2503.08678. doi: 10.48550/arXiv.2503.08678.

[56] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, “Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data,” Apr. 07, 2024, *arXiv*: arXiv:2401.10891. doi: 10.48550/arXiv.2401.10891.

[57] L. Yang *et al.*, “Depth Anything V2,” Oct. 20, 2024, *arXiv*: arXiv:2406.09414. doi: 10.48550/arXiv.2406.09414.

[58] A. Aggarwal, J. Wang, S. Hogue, S. Ni, M. Budagavi, and X. Guo, “Layered-Garment Net: Generating Multiple Implicit Garment Layers from a Single Image,” Nov. 22, 2022, *arXiv*: arXiv:2211.11931. doi: 10.48550/arXiv.2211.11931.

[59] J. E. Teixeira *et al.*, “Player Tracking Data and Psychophysiological Features Associated with Mental Fatigue in U15, U17, and U19 Male Football Players: A Machine Learning Approach,” *Appl. Sci.*, vol. 15, no. 7, p. 3718, Mar. 2025, doi: 10.3390/app15073718.

[60] Ultralytics, “YOLOv8 Documentation – Train Mode Parameters.” Accessed: Nov. 10, 2025. [Online]. Available: https://docs.ultralytics.com/modes/train

Appendix:

Appendix A: Additional Instance Segmentation Training Curves

This appendix provides the complete set of training and validation loss curves for all instance segmentation models evaluated in Chapter 5.1. These plots include the performance of YOLOv11-N (300 Epochs), YOLOv11-S (300 and 600 Epochs), and all Detectron2 configurations. These figures complement the summary table and primary loss curves in the main text, offering deeper insight into the training dynamics and convergence behavior of each model.

A.1 YOLOv11 N (300 Epochs)

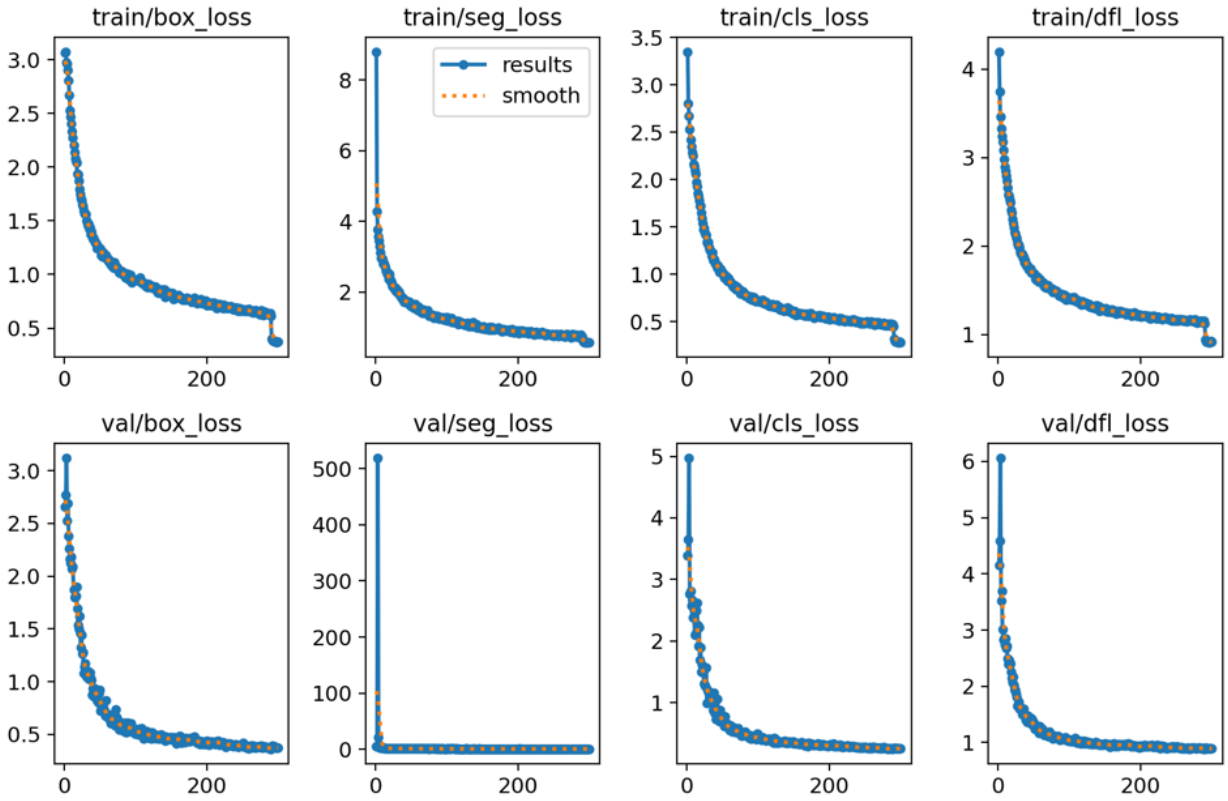

Figure 28 - Training and validation loss curves for the YOLOv11-N model trained for 300 epochs. The curves show early convergence, with performance plateauing, suggesting that extended training offered limited additional benefit compared to 200 epochs.

A.2 YOLOv11 S (300 Epochs)

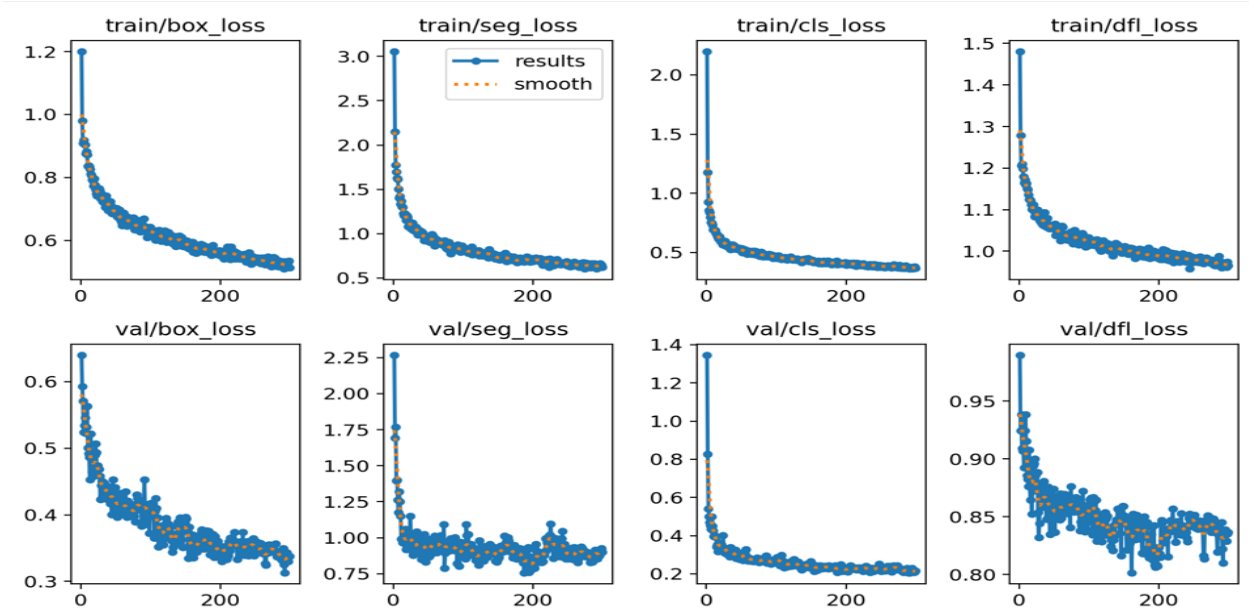

Figure 29 - Training and validation loss curves for the YOLOv11-N model trained for 300 epochs. The curves show early convergence, with performance plateauing, suggesting that extended training offered limited additional benefit compared to 200 epochs.

A.3 YOLOv11 S (600 Epochs)

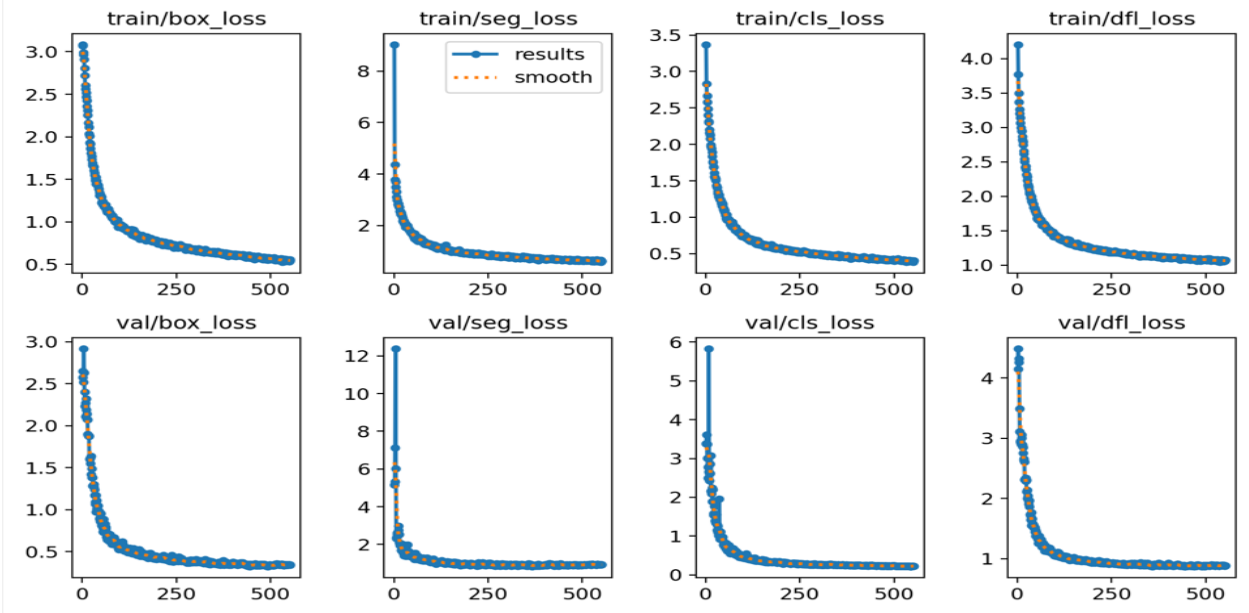

Figure 30- Training and validation loss curves for the YOLOv11-S model trained for 600 epochs. The extended training further reduces loss but offers only marginal improvements in segmentation quality, consistent with observed results. A.4 Detectron2 (Mask R-CNN)

A.4 Detectron2:

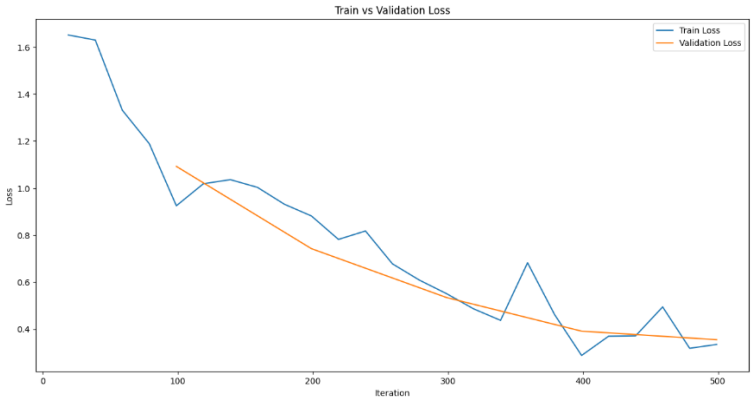

Figure 31 - Training and validation loss curves for the Detectron2 model configurations (500 epochs).

Figure 32 - Training and validation loss curves for the Detectron2 model configurations (1000 epochs).

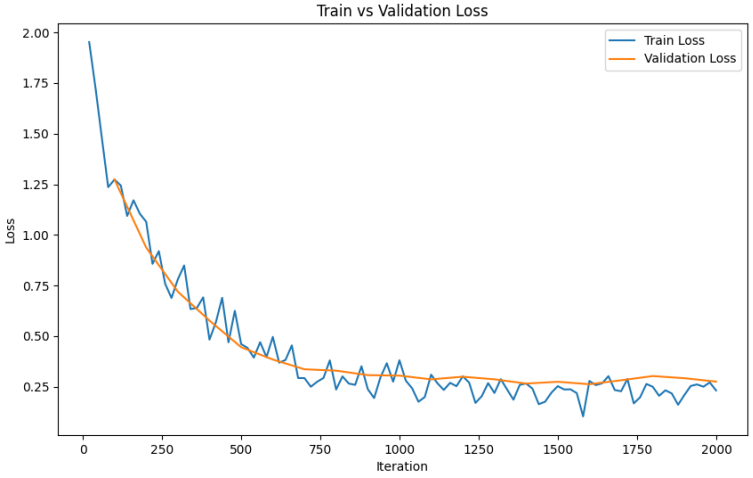

Figure 33 - Training and validation loss curves for the Detectron2 model configurations (2000 epochs).

Appendix B: Additional Object Recognition Training Curves

This appendix contains the training and validation accuracy and loss curves for the multi-class image classification models evaluated in Chapter 5.2.1. The figures illustrate the learning dynamics of ResNet, VGG16, and EfficientNet, both when trained from scratch and when pretrained on the DeepFashion2 dataset. These plots provide a visual comparison of model generalization and overfitting tendencies, supporting the quantitative analysis presented in the main results section.

B.1 ResNet (500x500, Pretrained)

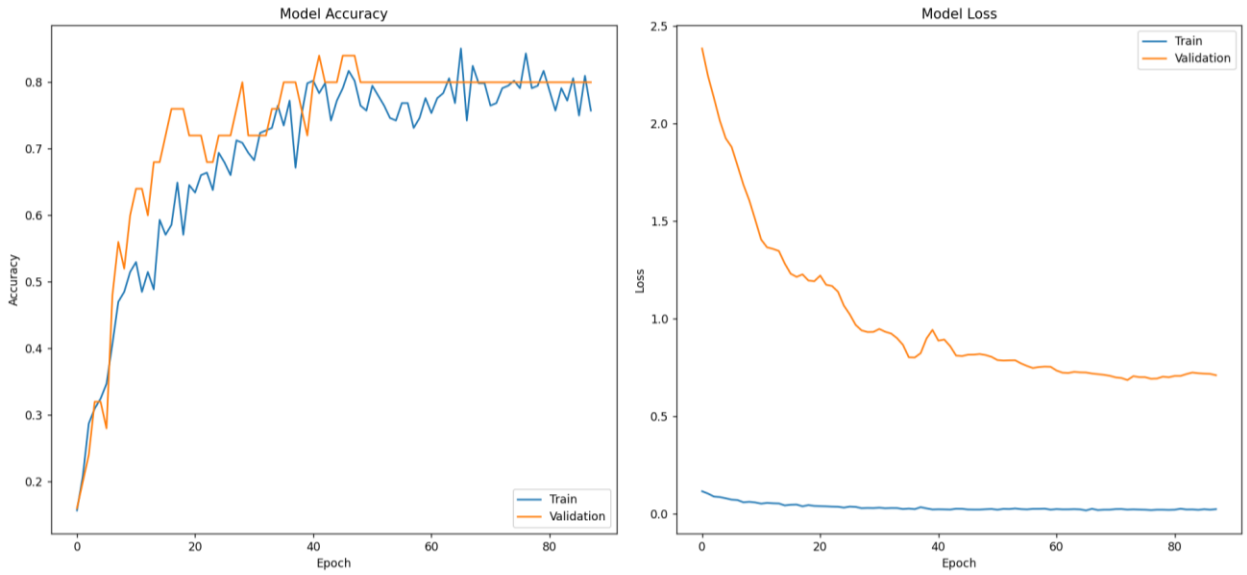

Figure 34 - Training and validation curves for the ResNet model pretrained on DeepFashion2 with 500x500 resolution.

B.2 VGG16 (500x500, Pretrained)

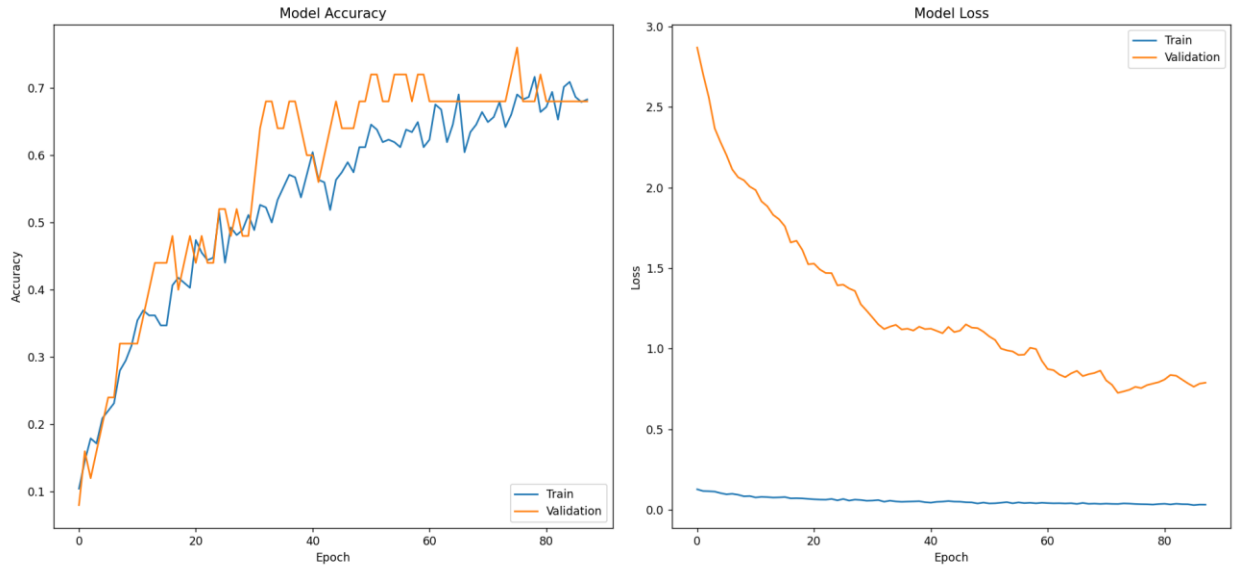

Figure 35 - Training and validation curves for the VGG16 model pretrained on DeepFashion2 with 500x500 resolution.

B.3 Models without Pretraining

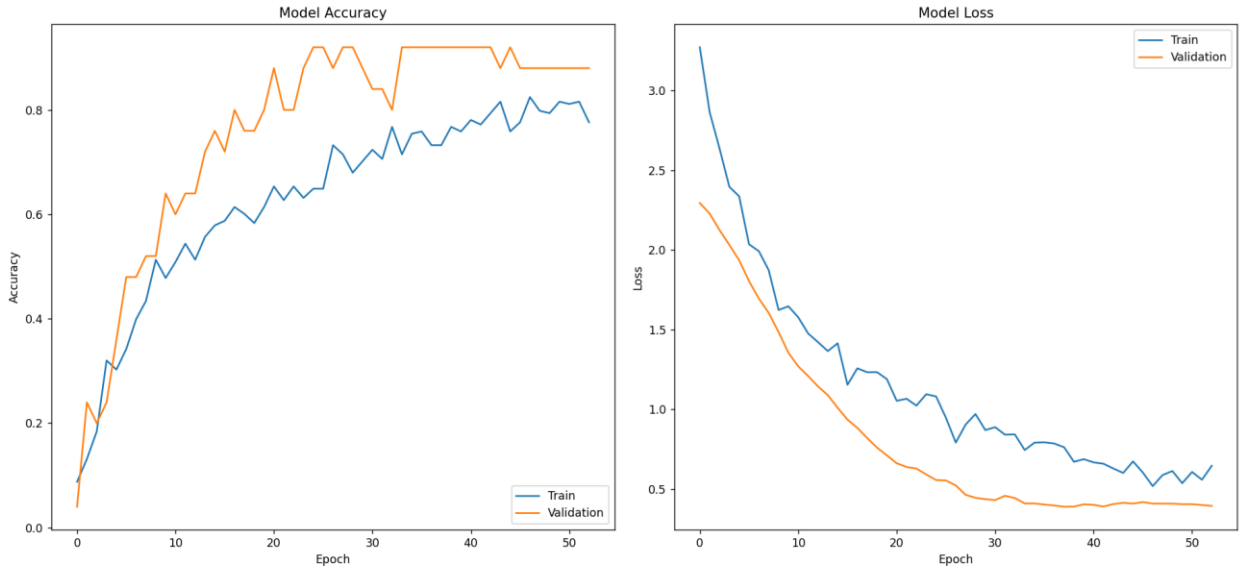

*Figure 36 - Training and validation curves for EfficientNet models trained from scratch on the custom dataset image size 500*500.*

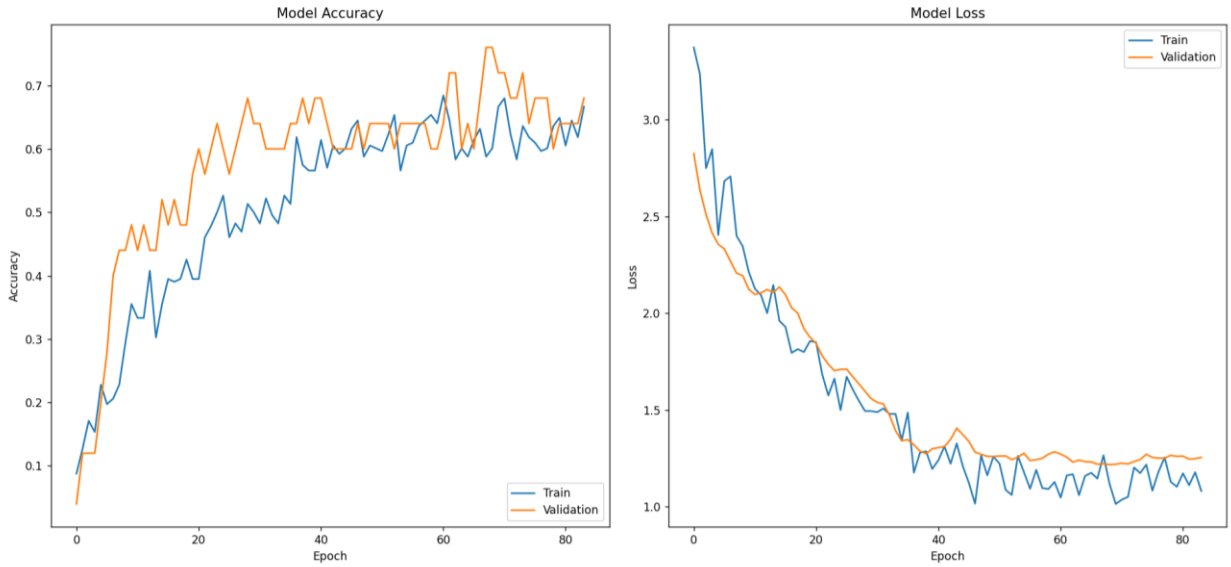

*Figure 37 - Training and validation curves for VGG16 models trained from scratch on the custom dataset image size 500*500.*

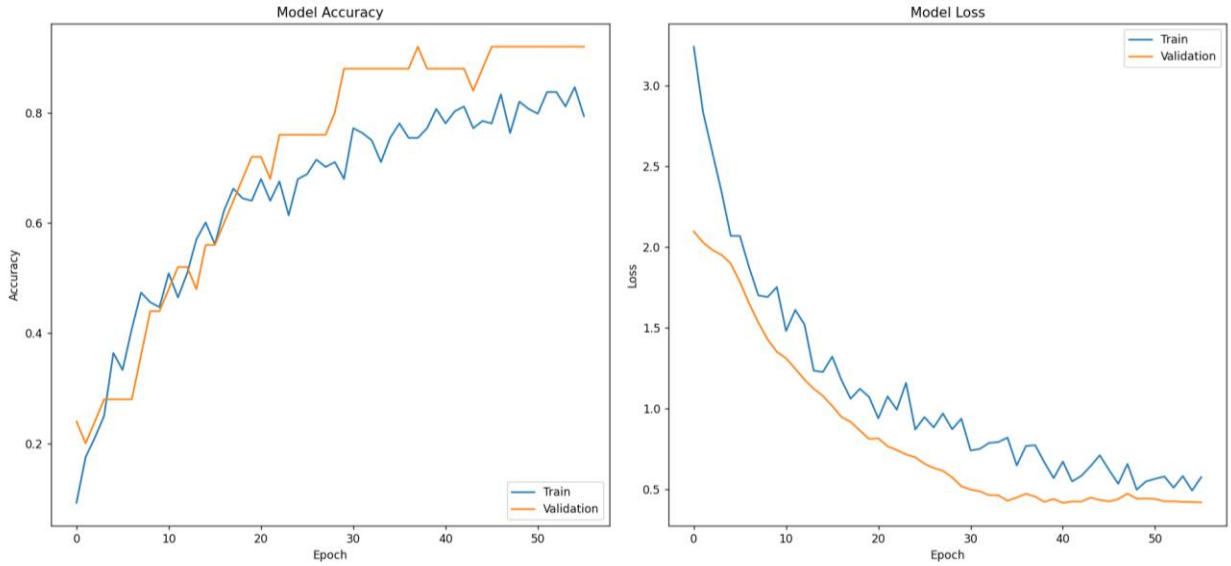

*Figure 38 - raining and validation curves for ResNet models trained from scratch on the custom dataset image size 500*500.*

Appendix C: Keypoint Detection Training Curves

This appendix presents the training and validation loss curves for the specialized, group-based keypoint detection models discussed in Chapter 5.3. The plots correspond to the different garment groups and training configurations. It is important to note that due to the small validation set sizes for some groups, these curves can appear noisy and may not be a perfect indicator of final test performance, but they provide valuable context for the training process.

C.1 Group 2 (Tank Top, Crop Top)

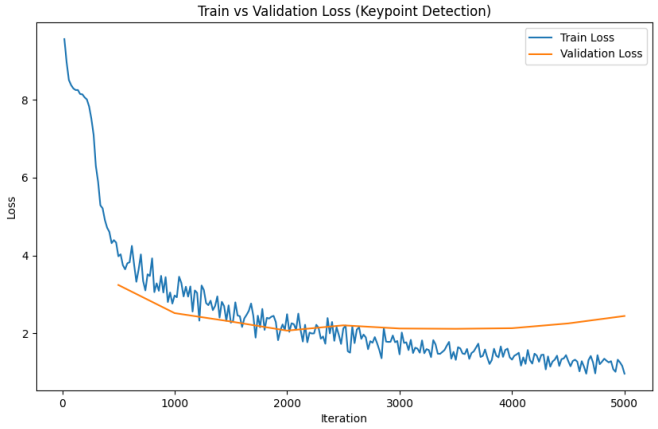

Figure 39 - Training and validation loss curve for the Group 2 model, trained for 5000 epochs.

C.2 Group 3 (Boxers, Shorts, Briefs)

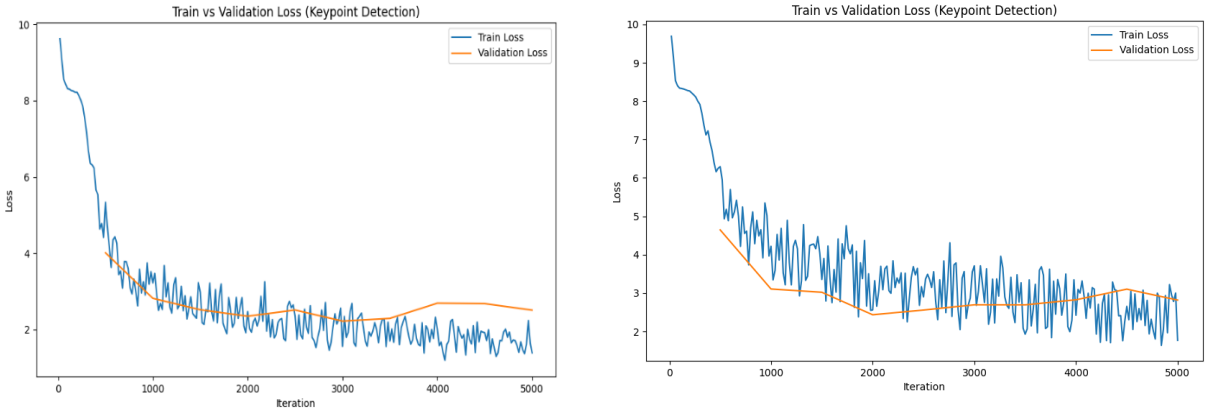

Figure 40 - Training and validation loss curves for the Group 3 models, including the baseline training and the runs with extended epochs and increased augmentation.

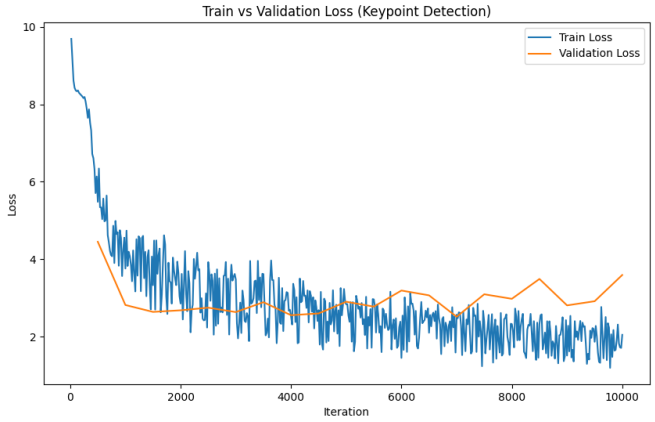

Figure 41 - Training and validation loss curves for the Group 3 models, including the baseline training and the runs with extended epochs and increased augmentation.

C.3 Group 4 (Long Socks, Skirt)

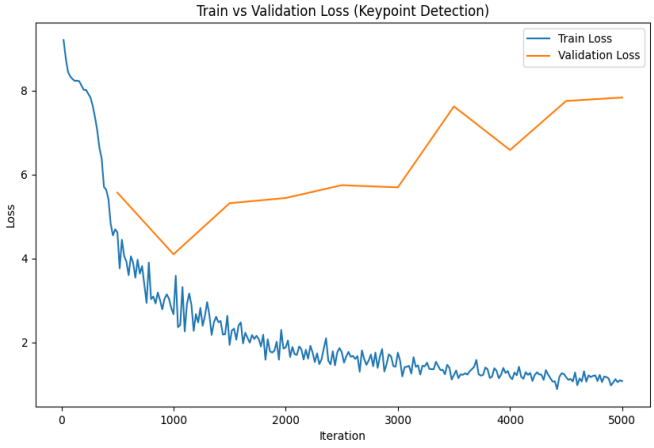

Figure 42 - training and validation curve for group 4 with 5000 epochs

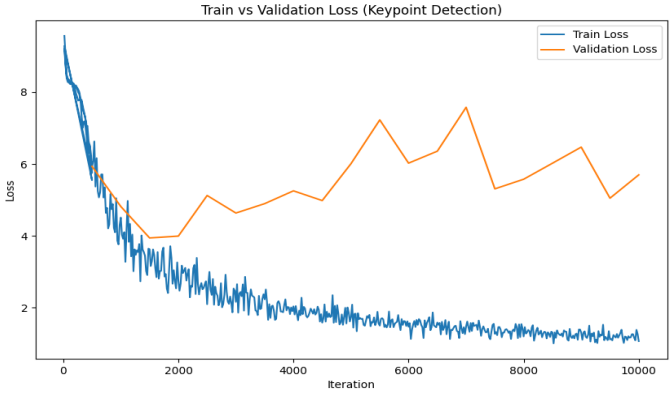

Figure 43 - training and validation curves of group 4 for 10000 epochs