

GARMENTIQ: AUTOMATED GARMENT MEASUREMENT FOR FASHION RETAIL

A PREPRINT

 **Li Yuan** *

Department of Computer Science
Swiss Federal Institute of Technology Zurich
leeyuan@ethz.ch

 **Xinrui Zhai** *

Department of Computer Science
Swiss Federal Institute of Technology Zurich
xizhai@ethz.ch

 **Fangzhou Ma** *

Department of Computer Science
Swiss Federal Institute of Technology Zurich
fangma@ethz.ch

June 13, 2025

ABSTRACT

Online fashion retail has revolutionized shopping but faces persistent sizing issues, driving return rates above 25%, costing fashion retailers billions of dollar annually, and increasing textile waste and carbon emissions. We present GarmentIQ, an end-to-end computer vision system combining garment classification, high-resolution segmentation, and landmark detection for precise, template-free measurements across nine categories. An interactive web interface lets users define custom measurement points and export structured JSON and PDF instructions. We used a 23,266-image dataset from Nordstrom and Myntra, and our tinyViT classifier achieves 95.76% accuracy, demonstrating superior generalization after fine-tuning on Zara data. BiRefNet produces high-quality segmentation, and HRNet-based landmark extraction attains high precision, with customized landmark derivation. GarmentIQ’s modular, user-friendly design streamlines workflows, reduces returns, and promotes sustainability, laying the groundwork for future automated fashion analysis.

Keywords Automated Garment Measurement · Fashion Computer Vision · Vision Transformer · Garment Keypoint Detection · Image Segmentation for Fashion

1 Introduction

The fashion industry is undergoing a profound transformation, largely propelled by the surge in online retail. This shift has introduced unparalleled convenience for consumers but has also amplified persistent challenges—most notably, the issue of inaccurate garment sizing. High return rates in online apparel sales, frequently exceeding 25%, are primarily driven by inconsistencies in sizing standards and the limitations of traditional, manual measurement methods [Miell et al., 2018, Zbavitel, 2024]. Manual garment measurement is not only labor-intensive, averaging around five minutes per item, but also susceptible to human error and inconsistency. These inefficiencies contribute to customer dissatisfaction and significant financial losses for retailers, with UK retailers alone facing costs estimated at £60 billion annually due to returns [Miell et al., 2018]. Furthermore, sizing inaccuracies exacerbate sustainability concerns by increasing carbon emissions and textile waste, thereby intensifying the environmental impact of the fashion sector [Miell et al., 2018].

Recent technological advancements are beginning to address these longstanding challenges. Digital garment measuring technologies and AI-driven solutions now enable faster, more precise, and highly consistent measurements, significantly

*All authors contributed equally to this research.

reducing both time and labor while improving fit accuracy [Kowaleczko et al., 2022, Paulauskaite-Taraseviciene et al., 2022]. For example, wearable technologies and computer vision-based systems can capture detailed body or garment measurements within minutes, streamlining processes such as custom tailoring and reducing the necessity for in-person fittings [Zbavitel, 2024, Chan et al., 2022].

Automated computer vision pipelines have demonstrated promising results in extracting garment measurements from real-world images. These systems leverage advanced segmentation and landmark detection algorithms to improve scalability and measurement precision [Zbavitel, 2024, Paulauskaite-Taraseviciene et al., 2022]. For instance, pipelines employing models like U-Net for garment segmentation and Keypoint R-CNN for landmark detection have achieved measurement errors as low as 0.75 - 1.27 cm across various garment types, highlighting their practical applicability and accuracy [Paulauskaite-Taraseviciene et al., 2022]. Neural network-based error correction further enhances measurement reliability, eliminating the need for rigid garment templates and accommodating a wider variety of apparel [Kowaleczko et al., 2022].

Despite these advancements, many current solutions remain constrained by their reliance on fixed templates or garment-specific models, limiting their flexibility and scalability in handling the diversity of real-world apparel [Kowaleczko et al., 2022, Paulauskaite-Taraseviciene et al., 2022].

In response to these multifaceted challenges, we introduce GarmentIQ - a comprehensive, automated system designed to revolutionize garment measurement in modern fashion retail. As shown in Figure 1, GarmentIQ integrates state-of-the-art computer vision techniques, including garment classification, high-resolution segmentation, and landmark detection, to enable precise measurement across a broad spectrum of garment categories. The system’s robust data pipeline incorporates large-scale image scraping and model fine-tuning, supporting nine distinct garment types.

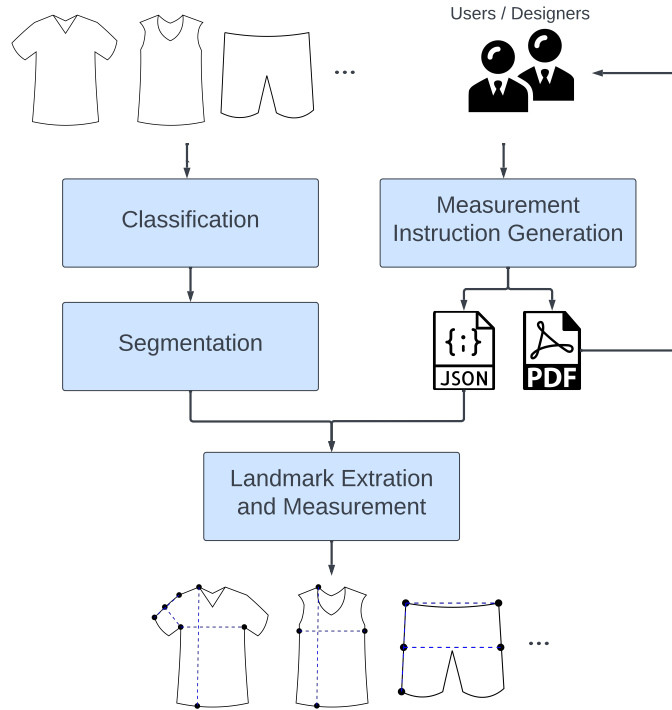


Figure 1: An overview of the GarmentIQ pipeline

A key innovation of GarmentIQ is its flexible measurement instruction generation interface. This interactive, web-based tool allows users to define custom measurement points and dimensions, generating standardized outputs in both JSON and PDF formats for consistent downstream integration. Unlike prior approaches that depend on fixed templates or manual calibration, GarmentIQ’s framework is adaptable to the variability of real-world garment images and diverse user requirements, ensuring both versatility and accuracy. This modular design philosophy aligns with recent academic recommendations advocating for adaptable frameworks in smart garment systems, which have demonstrated benefits in lowering technical barriers and enhancing creativity in design and production [Miell et al., 2018].

By combining advanced deep learning models with a user-centric, modular framework, GarmentIQ represents a significant leap forward in automated garment measurement. Experimental results validate the system’s high accuracy in garment classification and keypoint localization, underscoring its robustness and practical utility. In addressing the pressing needs of today’s fashion industry - improving fit, reducing returns, and promoting sustainability - GarmentIQ not only offers immediate operational benefits but also lays the groundwork for future innovations in flexible, production-ready fashion technology.

2 Related Work

Traditional machine learning approaches to garment classification, such as those based on handcrafted features and shallow classifiers, have achieved moderate performance. For instance, the combination of Histogram of Oriented Gradients (HOG) with a Support Vector Machine (SVM) achieved an accuracy of 86.53% on the Fashion-MNIST benchmark [K V and K, 2019] and up to 91.32% on pure clothing images [Xu et al., 2022], highlighting the limitations of these methods when faced with more complex or occluded scenarios. The advent of deep learning ushered in a new era of classification performance. Researchers have also explored optimizing Convolutional Neural Networks (CNNs) for fashion data classification, achieving a 92.68% test accuracy [Saranya and Geetha, 2022]. Moreover, early convolutional networks like AlexNet, when combined with extreme learning machines (ELM), reached 93.14% accuracy on the ACWS dataset, demonstrating the value of transfer learning and hybrid classifiers [Zhou et al., 2022].

Parallel to classification, semantic segmentation of clothing items has matured from simple threshold-based or edge-based methods to deep encoder-decoder architectures. U2Net, trained on the iMaterialist Fashion 2019 dataset, segments garments into multiple parts (upper body, lower body, full body) in complex scenes [Guo et al., 2019]. Advanced ResNet50-based models like FashionSegNet [Xiang et al., 2023] have achieved high-precision masks, addressing the variability and boundary complexity inherent in clothing images [He et al., 2015]. Extended U-Net architectures, enriched with auxiliary branches for bounding box and multi-label classification, have demonstrated IoU improvements from 73.38% to 87.37% by injecting category information, illustrating the value of multi-task learning in segmentation [Vozáriková et al., 2021]. Feature Fusion Networks (FFNet) that merge multi-stage encoder-decoder features further enhance boundary recovery and semantic consistency in parsing tasks [Li et al., 2025].

Keypoint-based landmark detection has been pioneered by the DeepFashion benchmark and its follow-up efforts. The ECCV 2016 “Fashion Landmark Detection in the Wild” introduced a cascade of CNNs to predict eight functional garment landmarks (e.g., neckline corners, hems), building the first large-scale dataset of 120K images and establishing baseline accuracies for landmark localization [Liu et al., 2016]. DeepFashion2 expands upon this work by providing dense landmarks (averaging 23 per category), per-pixel masks, and a comprehensive end-to-end Match R-CNN baseline, enabling robust landmark and pose estimation even under heavy occlusion and diverse viewpoints [Ge et al., 2019].

While many existing methods excel in one of these tasks - classification, segmentation, or landmark detection - few offer a unified, flexible framework capable of seamlessly integrating all three. Our GarmentIQ system not only achieves a competitive classification accuracy, which we will talk about in later sections, but also provides high-resolution segmentation and precise keypoint localization within a modular, user-driven interface. This positions GarmentIQ to surpass previous research by offering a holistic solution that emphasizes adaptability to varied garment types, efficient downstream integration, and extensibility for future innovations in automated garment measurement.

3 Methodology

In this section, we detail the methods used to develop GarmentIQ. For clarity, the methodology is divided into four main components: measurement instruction generation, classification, segmentation, and landmark extraction.

3.1 Measurement Instruction Generation

Measurement Instruction Generation is a preparatory step used before any measurements are taken. Once a designer finalizes a garment design, they can use our web interface to generate a complete set of measurement instructions in both JSON (for technical pipelines) and PDF (for human review). This workflow empowers non-technical users to produce precise, machine-readable instructions for downstream automated measurement.

3.1.1 Mechanism

To allow flexible, user-defined measurements on any of our nine garment types, we built a lightweight web interface (HTML / CSS / JavaScript) that guides the user through three simple steps:

1. **Garment Selection.** The user picks one of the nine garments (e.g. “short sleeve dress”) from a drop-down.
2. **Point Definition.** The user clicks on the predefined landmarks on the garment, or choose their own coordinates in the interface.
3. **Dimension Specification.** Our script automatically enumerates all unordered pairs of the selected points as candidate “dimensions.” The user then picks the pairs they care about (e.g. point 3 to point 7), assigns each a human-readable name and a description.

All interactions execute entirely client-side: JavaScript event listeners capture click locations, CSS renders draggable landmark markers, and the current points-and-dimensions state is maintained in a JSON object. Upon submission, the interface exports this instruction set as a downloadable JSON file and simultaneously generates a PDF in which each measurement line is overlaid on the garment image for easy visual verification.

3.1.2 Output Schema

The measurement interface produces a JSON file conforming to a Draft-7 schema (see Appendix A for the full listing). At a high level, the schema enforces:

- **Top-level objects:** Each key is a garment type mapping to a garment record.
- **Garment record:** Contains exactly two required fields:
 - `landmarks` A map from numeric IDs to landmark objects. Each landmark must specify:
 - `predefined` (boolean): `true` if predefined in the GarmentIQ system, `false` if user-defined.
 - `description` (string): Semantic label for the point.
 - `x, y` (number): Pixel coordinates in image space.
 - `neighbors` (object): Exactly two nearby landmarks keyed by their IDs, providing contextual connectivity for customized landmarks.
 - `derivation` (object): Specifies how a custom landmark was computed. Includes:
 - * `function` (string): The name of the derivation function used during landmark extraction.
 - * One or more additional named parameters representing arguments passed into the function, such as landmark IDs or geometric constraints (e.g., `p1_id`, `direction`).
 - `measurements` A map from measurement names to measurement definitions. Each definition must include:
 - `landmarks.start` and `landmarks.end`: The IDs of the two landmarks between which the measurement is taken.
 - `description`: A human-readable note explaining the measurement.
- No additional properties are allowed at any level, ensuring strict conformance.

This schema guarantees that downstream modules receive a consistent, self-documenting instruction set, supporting both predefined and dynamically derived custom landmarks. The complete Draft-7 JSON Schema is provided in Appendix A.

3.2 Classification

Classification serves as the first and arguably most critical step in the entire pipeline, acting as a bottleneck for subsequent processes [Petersen et al., 2014]. If a garment is misclassified at this stage, the downstream tasks will inevitably fail. To ensure high reliability, we trained three different models and compared their performances to select the best one.

Given that our target includes nine garment categories (long sleeve dress, long sleeve top, short sleeve dress, short sleeve top, shorts, skirt, trousers, vest, and vest dress) existing public datasets were insufficient to cover all types. Therefore, we scraped a substantial number of images from Nordstrom, a leading American fashion retailer, to build a more comprehensive dataset.

3.2.1 Image Scraping

To supplement existing fashion datasets with all nine target garment categories, a semi-automated scraping pipeline was developed for Nordstrom’s website. First, we manually collect raw network logs in our browser’s Developer Tools by filtering image requests and saving the results as a JSON metadata file. We then run a PowerShell script that:

1. Parses the raw JSON to extract and index each image URL.
2. Creates a local "images" directory (if absent).
3. Downloads every image.
4. Generates a cleaned metadata file and removes the raw log only if all downloads succeed.

This workflow ensures robust handling of Nordstrom’s access restrictions (via custom headers and pacing), produces well-organized image and metadata sets.

3.2.2 Model Structure

To achieve robust garment classification performance, we experimented with three different model architectures: two custom convolutional neural networks (CNN-3 and CNN-4) and one transformer-based model (tinyViT).

Both CNN-3 and CNN-4 are manually designed convolutional neural networks featuring modular structures inspired by VGG-like architectures [Simonyan and Zisserman, 2015]. Each model consists of multiple convolutional blocks, each comprising two convolutional layers followed by batch normalization, ReLU activation, max pooling, and dropout for regularization.

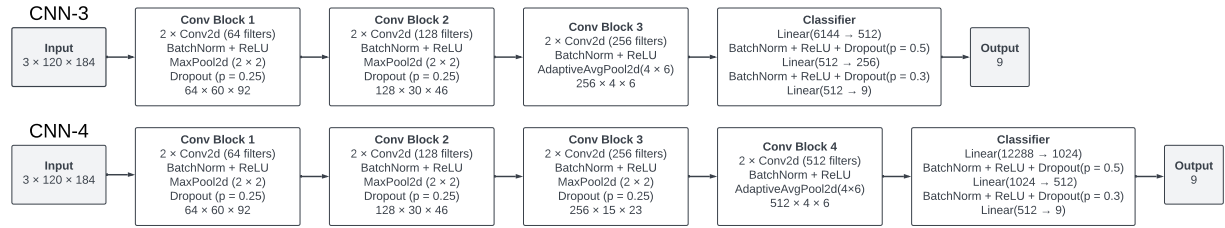


Figure 2: An overview of CNN-3 and CNN-4 model structures

As shown in Figure 2, CNN-3 consists of three such blocks, while CNN-4 introduces an additional fourth block with deeper feature extraction capabilities. To address overfitting, both models employ dropout layers at different stages. Before classification, the feature maps are adaptively average-pooled to a fixed spatial size and then flattened. The resulting features pass through two fully connected layers with batch normalization and dropout, before the final classification output.

The third model we evaluated is based on the Data-efficient Image Transformer (DeiT) architecture, specifically the tiny-sized version (deit_tiny_patch16_224), developed by Meta AI. Unlike CNN-based models, tinyViT treats the input image as a sequence of patches and processes them through a Transformer encoder, similar to models used in natural language processing. Each patch embedding is combined with positional encodings, and the token output is used for image classification. We initialized tinyViT with pretrained weights on ImageNet-1k to leverage transfer learning and fine-tuned it for our garment classification task. The final classification head was replaced with a new linear layer corresponding to the number of garment classes [Touvron et al., 2021, Wu et al., 2020, Deng et al., 2009].

3.2.3 Model Fine-tuning

To adapt all three trained base classifiers - CNN-3, CNN-4, and tinyViT - to the domain-specific Zara dataset, we fine-tuned each model using a lightweight and consistent protocol tailored for small datasets. Fine-tuning was conducted using stratified 5-fold cross-validation to ensure class balance and robust model selection. For each model, training was initialized from previously trained weights, with optional freezing of earlier layers to retain generalizable features. Early stopping based on validation loss was employed to avoid overfitting, and the best-performing model across all folds was selected according to the lowest cross-entropy loss. The selected model was then re-evaluated on the full Zara dataset to assess domain alignment. This approach ensured each classifier could be effectively adapted to the specific visual characteristics of Zara images, which was particularly important for supporting downstream landmark detection.

Given the limited size of the Zara dataset - where reserving a separate test split would have unduly reduced the already scarce training examples - using cross-validation for model selection allowed us to maximize data utilization without compromising methodological rigor. Moreover, because the classifier serves solely as a feature extractor for a landmark detection pipeline that is evaluated independently, this evaluation strategy strikes an appropriate balance between domain adaptation and robust performance estimation.

3.3 Segmentation

For the segmentation task, we employed the BiRefNet (Bilateral Reference Network) model [Zheng et al., 2024], a state-of-the-art method for high-resolution binary segmentation. We used the publicly available pretrained model without additional fine-tuning for efficient and high-quality segmentation.

3.4 Landmark Extraction

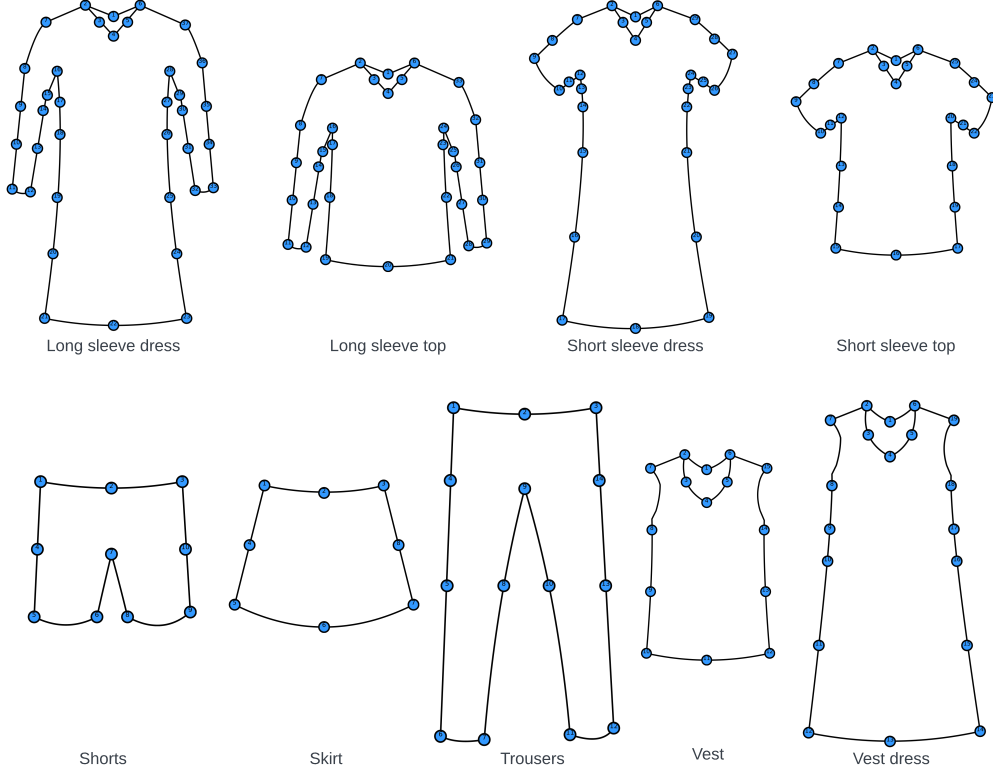


Figure 3: DeepFashion2 predefined landmarks

Accurate landmark localization is critical for downstream measurement tasks. We employ a high-resolution network (HRNet) pretrained on the DeepFashion2 dataset to extract a fixed set of semantic keypoints for each garment as shown in Figure 3. By maintaining high-resolution feature maps throughout the network, HRNet produces spatially precise heatmaps that correspond to the predefined fashion landmarks in DeepFashion2 [Ge et al., 2019].

3.4.1 Extraction on Predefined Landmarks

Each input image is first cropped to isolate a single garment, using either the ground-truth bounding box or the output of a clothing detector. The resulting crop is processed by the pretrained HRNet-for-Fashion-Landmark-Estimation model, which is trained on the DeepFashion2 dataset. This model produces a heatmap for each of the predefined DeepFashion2 landmarks. The pixel with the highest activation in each heatmap is selected as the (x, y) coordinate of the corresponding keypoint. These extracted landmarks are then retained for use in the measurement workflow [Sun et al., 2019, Xiao et al., 2018, Wang et al., 2019].

3.4.2 Extraction on Customized Landmarks

While the predefined landmarks provided by the DeepFashion2 dataset are suitable for many tasks, they do not cover all keypoints required for precise garment measurements. To address this limitation, we introduce a method for deriving customized landmarks based on the available detected points and fine-grained segmentation masks derived from the BiRefNet model.

Take the top as an example shown in Appendix C, to measure the front length of a top, one must determine the vertical distance from the shoulder seam to the lowest point of the garment’s front hem. While the pretrained model can detect the shoulder landmark, it does not provide a corresponding hem point. To resolve such problem, we design a general-purpose function that derives such customized landmarks by intersecting directional rays with the garment boundary.

The process operates as follows: Given a known keypoint (e.g., the shoulder), we define a direction vector (e.g., vertically downward). A ray is cast from the keypoint in the specified direction until it intersects with the boundary of the garment, which is extracted from the segmentation mask. The intersection point is then recorded as the customized landmark. This approach can be applied flexibly to estimate various garment-specific landmarks not directly detected by the pretrained model, enabling more comprehensive and accurate measurements.

This derivation strategy bridges the gap between keypoint detection and fine-grained garment analysis, leveraging both the spatial information from predicted landmarks and the structural cues embedded in the segmentation masks.

4 Experimental Setup

This section details the implementation and configuration of each component in our system pipeline, from user-facing tools to backend processing modules. We first describe the interactive interface used to generate measurement instructions, followed by the classification, segmentation, and landmark extraction modules. Finally, we outline how all components are integrated into a unified pipeline, enabling automated garment analysis from raw images to structured measurement data.

4.1 Measurement Instruction Generation

The measurement-instruction tool is a purely client-side application implemented in vanilla JavaScript, HTML5, and CSS3. Internally, it maintains two in-memory data structures: one tracks the set of selected landmarks for each garment, and the other holds the final instruction payload. Navigation through the three stages-garment selection, landmark picking, and measurement definition-is handled by showing or hiding the corresponding containers.

When a garment is chosen, its SVG thumbnail is loaded dynamically into the page. During landmark selection, click events on the SVG markers toggle their visual state and update the in-memory selection set. In a special “custom mode,” clicks on empty areas of the SVG insert new landmark markers with user-editable labels. Once landmarks are finalized, the application enumerates every pair of selected points and populates a table of candidate measurements, each with inputs for a user-defined name and description. Selecting a measurement draws a dashed line between the two points on the SVG; deselecting removes it.

For export, the tool serializes the landmark and measurement data into a JSON file for download. To generate a PDF, it renders the annotated SVG into an HTML canvas (with an SVG-to-canvas fallback) and then embeds the resulting raster image and a formatted measurement table into a letter-size PDF document. This event-driven, DOM-manipulation approach requires no backend and ensures a smooth, interactive workflow from user input to final instruction files.

4.2 Classification

4.2.1 Base Training Dataset

The garment classification models are trained on the “Nordstrom & Myntra Clothes Image Data - GarmentIQ” dataset [Yuan, 2025a], which combines high-quality fashion photographs from two e-commerce platforms. Specifically, Myntra clothes images are randomly chosen from the “Fashion Products Image Dataset [Aggarwal, 2019].” Figure 4 shows some sample images from the dataset.

In total, the dataset contains 23,266 images across nine garment categories, Table 1 shows the garment category breakdown from those two platforms.

The dataset is accompanied by a `metadata.csv` file that serves as a mapping directory. Each row in this CSV contains the image filename or ID, its full URL, the assigned garment category, and the source platform (Myntra or Nordstrom). By querying the CSV, downstream code can efficiently filter, download, and organize images for training, validation, and testing without hard-coding paths or URLs.



Figure 4: Sample images from each of the nine garment categories in the base training dataset

Table 1: Image counts per garment category from Myntra and Nordstrom

| Category | Myntra | Nordstrom | Total |
|--------------------|--------------|---------------|---------------|
| long sleeve dress | 0 | 2,334 | 2,334 |
| long sleeve top | 3,215 | 0 | 3,215 |
| short sleeve dress | 0 | 2,586 | 2,586 |
| short sleeve top | 3,500 | 0 | 3,500 |
| shorts | 545 | 2,566 | 3,111 |
| skirt | 128 | 1,558 | 1,686 |
| trousers | 1,653 | 617 | 2,270 |
| vest | 15 | 1,511 | 1,526 |
| vest dress | 0 | 3,038 | 3,038 |
| Total | 9,056 | 14,210 | 23,266 |

4.2.2 Fine-tuning Dataset

To specialize the garment classifiers for downstream landmark detection on Zara clothing images [Yuan, 2025b], we curated a small domain-specific dataset sourced exclusively from Zara’s product imagery. Unlike the more diverse and mixed-platform base training data, this dataset focuses solely on Zara garments to better capture their consistent visual style, photographic angles, and fabric textures. As shown in Table 2, the Zara dataset includes 834 images across the same nine garment categories used in the base dataset. Due to the relatively small size of this collection, it was used entirely for training and validation via 5-fold cross-validation during fine-tuning.

This focused dataset enables the models to better generalize to the stylistic characteristics of Zara garments, which is critical for reliable downstream tasks such as keypoint and landmark detection.

4.2.3 Training, Testing, and Fine-tuning Framework

The garment classification pipeline is implemented as a self-contained Python module that abstracts all routine tasks into four core operations, streamlining experimentation and deployment:

First, the data split utility ingests one or two ZIP archives containing images and their accompanying metadata CSV. It verifies directory structure, checks filenames against metadata for consistency, and either performs a stratified random split or accepts separate train/test archives, producing clean `train/` and `test/` folders ready for model consumption.

Second, the training function wraps model instantiation, optimizer setup, and the training loop within a stratified k-fold cross-validation framework. Early stopping and automatic checkpointing of the best validation model are built-in, so

Table 2: Image counts per garment category in the Zara fine-tuning dataset

| Category | Count |
|--------------------|------------|
| long sleeve dress | 80 |
| long sleeve top | 352 |
| short sleeve dress | 14 |
| short sleeve top | 84 |
| shorts | 56 |
| skirt | 62 |
| trousers | 80 |
| vest | 94 |
| vest dress | 12 |
| Total | 834 |

invoking training requires only the model class (e.g. CNN3, CNN4, or tinyViT), its constructor arguments, and a small hyperparameter dictionary.

Third, the testing routine loads a saved checkpoint, runs inference over the held-out set, and computes standard metrics, cross-entropy loss, overall accuracy, weighted F1 score, and prints a detailed per-class classification report. This function can be pointed at any checkpoint file without further configuration.

Finally, the prediction helper supports single-image inference: it loads a pretrained model, applies the same preprocessing pipeline (resize, normalize), and returns the top predicted label along with the full probability vector, which could make a smooth transition to the next task.

By accepting any `torch.nn.Module` subclass, this framework allows users to plug in custom architectures as easily as the provided out-of-the-box models, enabling an effortless, end-to-end workflow for training, evaluating, and deploying garment classifiers.

In addition to training from scratch, the framework supports fine-tuning of pretrained models via the `fine_tune_pytorch_nn` function. This utility enables transfer learning by loading existing weights, optionally freezing specified base layers, and retraining the model on a new dataset using stratified k-fold cross-validation. It retains all the benefits of the training workflow—early stopping, checkpointing, and consistent metric tracking—while accelerating convergence and improving generalization on limited data. The fine-tuning process is fully configurable, accepting user-defined models, datasets, and optimizer parameters, thus making it easy to adapt pretrained models to new classification tasks.

4.3 Segmentation

For image segmentation, we utilize BiRefNet model [Zheng et al., 2024], enabling us to achieve high-quality mask extraction without training a model from scratch. While the core segmentation model itself is externally sourced, we designed a flexible and extensible framework around it to streamline inference and facilitate downstream use cases such as background editing and batch processing.

The segmentation pipeline is modular and designed for ease of use and extensibility. Upon loading, the model is configured for either full or half precision, automatically selecting the appropriate hardware accelerator.

Given an input image, the pipeline resizes and normalizes it before passing it to the model for inference. The resulting segmentation mask is returned at the original resolution, enabling precise foreground extraction. A built-in utility allows for visual inspection of both the raw image and mask.

To support upcoming tasks and downstream applications, we provide functionality to replace the background with a specified color using the generated mask. Furthermore, a batch processing module enables efficient segmentation and optional background replacement for entire directories of images, automatically saving results in a structured format for further use or evaluation.

4.4 Landmark Extraction

To evaluate the accuracy of our landmark extraction method, we manually annotated all the keypoints required for garment measurement on 10 images per garment type using a self-built web interface. This interface facilitated efficient collection of ground-truth coordinates and allowed for streamlined comparison with automatically extracted keypoints.

Once annotated, the images are first passed through a classification module that predicts the garment type. The predicted class label is then used to guide the subsequent landmark extraction and derivation process. Specifically, our landmark detection model is based on HRNet trained on the DeepFashion2 dataset, which outputs predictions for up to 294 predefined keypoints across all garment types [Sun et al., 2019, Xiao et al., 2018, Wang et al., 2019]. However, not all landmarks are relevant for every garment. Therefore, we implement a filtering step where only the subset of landmarks corresponding to the predicted garment class is retained for further processing.

To handle additional measurement-specific keypoints that are not included in the predefined set, we developed a python module of landmark derivation. Internally, the module is organized into several functional layers which includes managing segmentation masks, geometric operations such as directional ray generation, boundary intersection via Shapely and OpenCV.

The detection and derivation modules are functionally decoupled but tightly integrated in the pipeline. After landmark detection and class-based filtering, relevant landmarks are passed to the derivation module when additional measurement-specific points are required. All outputs-predicted landmarks, derived landmarks, and segmentation masks-are stored in a unified data structure, enabling seamless comparison with manual annotations.

Once all relevant landmarks-both detected and derived-are obtained, we use them to compute garment measurements by calculating the pixel distance between specific landmark pairs. These measurement rules are defined in a structured JSON configuration file, which specifies which landmarks correspond to each measurement (e.g., chest width, front length, hip width). For each image, we compute the measurement ratios (e.g., chest/front length) based on the extracted keypoints and compare them to the ratios obtained from the manually annotated ground truth. This ratio-based evaluation allows us to assess the accuracy of the pipeline in a scale-invariant way and provides insight into how well the system replicates real-world garment measurements across different garment categories.

5 Results

In this section, we present the results of our system components. Since the segmentation module is based on a well-pretrained model, we did not conduct further evaluation for it. Instead, we focus on the classification component, which was trained and tested using our curated dataset. In addition, we put a lot effort on landmark extraction evaluation, especially for customized landmarks.

5.1 Measurement Instruction Generation

As previously mentioned, the measurement instruction interface produces two complementary outputs: a structured JSON file for machine processing and a PDF visualization for human verification. Taking the short sleeve top category as an example, both outputs represent the same set of garment-specific measurement logic, typically including waist, full length, and hips, defined by annotated landmarks.

The JSON output (Appendix B) encodes the defined landmarks and measurements in a machine-readable format, associating each point with coordinates and descriptive labels. It’s important to note that these coordinates correspond to positions on the example garment image used during the landmark selection process-they do not reflect the actual physical dimensions of a garment. Instead, the coordinates serve primarily for calibration and traceability. Additionally, they act as placeholders: when landmark positions are extracted from real garment images, these coordinates can be updated while preserving the same output structure, ensuring consistency and interoperability across different system components. In cases where landmarks are not predefined, the JSON also includes a derivation block specifying how a custom landmark should be generated. This block contains a function name (e.g., `derive_keypoint_coord`) along with its associated arguments, such as reference point IDs and geometric parameters (e.g., direction), which together define the logic used to compute the position of the custom landmark during landmark extraction. The PDF counterpart (Appendix C) provides a visual summary, displaying landmarks and measurement lines directly on the garment image, which allows for intuitive human verification and interpretation.

This dual-format output is essential for downstream tasks, while the JSON enables automated measurement extraction, the PDF ensures that human reviewers can validate and refine the instructions, ensuring accuracy and reliability in later stages such as dataset generation or model training.

5.2 Classification

5.2.1 Base Model Performance

We evaluated three models - CNN-3, CNN-4, and tinyViT - on an 85/15 stratified split of the dataset. This yielded 19,777 training images and 3,489 test images, with the class distributions as Table 3 below.

Table 3: Number of images per garment category in the train and test sets

| Category | Train | Test |
|--------------------|---------------|--------------|
| short sleeve top | 2,977 | 523 |
| long sleeve top | 2,773 | 442 |
| shorts | 2,626 | 485 |
| vest dress | 2,596 | 442 |
| short sleeve dress | 2,204 | 382 |
| long sleeve dress | 1,950 | 384 |
| trousers | 1,950 | 320 |
| skirt | 1,405 | 281 |
| vest | 1,296 | 230 |
| Total | 19,777 | 3,489 |

All images were resized to 120×184 and normalized with mean $[0.8047, 0.7808, 0.7769]$, standard deviation $[0.2957, 0.3077, 0.3081]$ on the RGB channels. Training used AdamW optimizer [Loshchilov and Hutter, 2019], with learning rate 2.25×10^{-4} , weight decay 10^{-4} , 15-fold cross-validation, up to 120 epochs with patience 10, and batch size 256. tinyViT additionally employed a patch size of 6.

Table 4: Summary of precision (P), recall (R), and F1-score on the test set for each base model

| Category | CNN-3 | | | CNN-4 | | | tinyViT | | |
|--------------------|--------|------|------|-------------|-------------|-------------|---------------|-------------|-------------|
| | P | R | F1 | P | R | F1 | P | R | F1 |
| long sleeve dress | 0.93 | 0.89 | 0.91 | 0.94 | 0.88 | 0.91 | 0.94 | 0.92 | 0.93 |
| long sleeve top | 0.99 | 0.97 | 0.98 | 0.99 | 0.98 | 0.99 | 0.98 | 1.00 | 0.99 |
| short sleeve dress | 0.85 | 0.93 | 0.89 | 0.87 | 0.95 | 0.91 | 0.89 | 0.91 | 0.90 |
| short sleeve top | 0.97 | 0.98 | 0.98 | 0.98 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 |
| shorts | 0.97 | 0.98 | 0.98 | 0.98 | 0.99 | 0.98 | 0.99 | 0.99 | 0.99 |
| skirt | 0.92 | 0.91 | 0.92 | 0.93 | 0.91 | 0.92 | 0.95 | 0.93 | 0.94 |
| trousers | 0.98 | 0.98 | 0.98 | 0.97 | 0.99 | 0.98 | 0.99 | 0.98 | 0.99 |
| vest | 0.94 | 0.93 | 0.93 | 0.97 | 0.94 | 0.96 | 0.92 | 0.95 | 0.94 |
| vest dress | 0.94 | 0.90 | 0.92 | 0.94 | 0.92 | 0.93 | 0.94 | 0.92 | 0.93 |
| Overall | | | | | | | | | |
| Precision | 0.9458 | | | 0.9533 | | | 0.9576 | | |
| F1 (weighted) | 0.9459 | | | 0.9533 | | | 0.9576 | | |
| Trainable Params | 4.43M | | | 17.81M | | | 5.48M | | |

Classification results on the test set are summarized in Table 4. All three models, CNN-3, CNN-4, and tinyViT, achieved strong performance, with overall accuracies exceeding 94%. tinyViT achieved the best results, reaching 95.76% accuracy and the highest macro F1 score. While all models performed well on garments like short sleeve tops, shorts, and trousers, they consistently struggled more with long sleeve dresses and vest dresses. Specifically, long sleeve dress had the lowest recall for CNN-3 (0.89) and CNN-4 (0.88), and vest dress showed slightly lower F1 scores across all models compared to other categories. These two garment types appear to be the most challenging to classify, likely due to visual similarity with other categories and high intra-class variability.

5.2.2 Fine-tuned Model Performance

Similar to the base model training, we resized, standardized the images and used AdamW optimizer [Loshchilov and Hutter, 2019], with learning rate 10^{-5} , weight decay 10^{-4} , 5-fold cross-validation, up to 50 epochs with patience 5, and batch size 128. tinyViT additionally employed a patch size of 6.

Table 5: Model performance after fine-tuning on Zara data and evaluated on both Zara (training set) and Nordstrom & Myntra (N&M) test set. Precision / F1 change is relative to baseline performance in Table 4.

| Metric | CNN-3 | CNN-4 | tinyViT |
|------------------------------|---------|---------|----------------|
| Zara Train Precision | 0.9197 | 0.9592 | 0.9916 |
| Zara Train F1 | 0.9216 | 0.9585 | 0.9917 |
| Test Precision (N&M) | 0.9074 | 0.9132 | 0.9484 |
| Test F1 (N&M) | 0.9068 | 0.9137 | 0.9483 |
| Test Change (N&M) | | | |
| Precision | -0.0384 | -0.0401 | -0.0092 |
| F1 (weighted) | -0.0391 | -0.0396 | -0.0093 |

After fine-tuning on the Zara dataset, the performance of all models was re-evaluated on both the fine-tuning (Zara) set and the original test set from Nordstrom and Myntra. As shown in Table 5, all models achieved good accuracy and F1 scores on the Zara data, confirming successful adaptation to the target distribution. Notably, the tinyViT model attained a near-perfect performance on Zara with 99.16% accuracy and an F1 score of 0.9917.

However, performance on the Nordstrom and Myntra test set diverged depending on the model architecture. All models experienced a drop in performance on the Nordstrom and Myntra test set after fine-tuning. CNN-3 and CNN-4 both saw decreases in precision (3.84% and 4.01%, respectively) and F1 score (0.0391 and 0.0396), suggesting some degree of overfitting to the Zara distribution.

Meanwhile, tinyViT demonstrated the smallest drop, with a minor decrease of only 0.92% in precision and 0.0093 in F1 score. This indicates that the transformer-based tinyViT model generalizes better to new, out-of-distribution data compared to the CNN-based models. The relatively smaller drop in performance for tinyViT suggests that, even after fine-tuning, it retains stronger generalization capabilities across domains.

5.3 Landmark Extraction

For each garment, we computed the relative error of key measurements by comparing the ratio of dimensions derived from predicted landmarks to those obtained from manual annotations. The table below reports the average absolute difference in these ratios for each measurement and garment type:

5.3.1 Without Refinement

The results in Table 6 show that for most measurements - particularly those involving the chest and front length - the method achieves low average ratio errors (below 6%), indicating strong alignment between predicted and manually annotated landmarks. In these cases, the combination of the primary landmark detection model and our customized landmark derivation strategy proves highly effective.

Table 6: Average absolute ratio error across garment types (Original)

| Measurement | Short sleeve top | Long sleeve top | Vest | Shorts | Trousers | Skirt | Short sleeve dress | Long sleeve dress | Vest dress |
|------------------------------|------------------|-----------------|-------|--------|----------|-------|--------------------|-------------------|------------|
| Back width / Arm width | 0.056 | 0.107 | — | — | — | — | — | — | — |
| Chest / Front length | 0.031 | 0.034 | 0.047 | — | — | — | 0.032 | 0.059 | 0.019 |
| Front length / Chest | 0.031 | 0.033 | 0.056 | — | — | — | 0.031 | 0.055 | 0.019 |
| Hip / Waist | — | — | — | 0.242 | 0.029 | 0.821 | 0.258 | 0.236 | 0.096 |
| Sleeve length / Front length | 0.037 | 0.015 | — | — | — | — | 0.054 | 0.038 | — |

However, larger errors appear in certain measurements, such as the Hip/Waist ratio for shorts (0.242), skirts (0.821) and dresses (ranging from 0.096 to 0.258). These discrepancies can be attributed to two main factors: (1) the landmark detection model struggles to generalize well to garments such as skirts and (2) it is inherently difficult to define a consistent hip position on such garments, which introduces ambiguity even in manual annotations.

To address the above mentioned relatively large errors, we introduced two additional strategies into the pipeline aimed at improving robustness and precision.

5.3.2 Background Replacement with Green Screen

Inspired by practices in the apparel and film industries, we replaced the original background of each image with a uniform green color, RGB (102, 255, 102). The background modification process is shown in Figure 5, where we first segment the trousers image to generate a mask, then use the mask to modify the background of the original image.

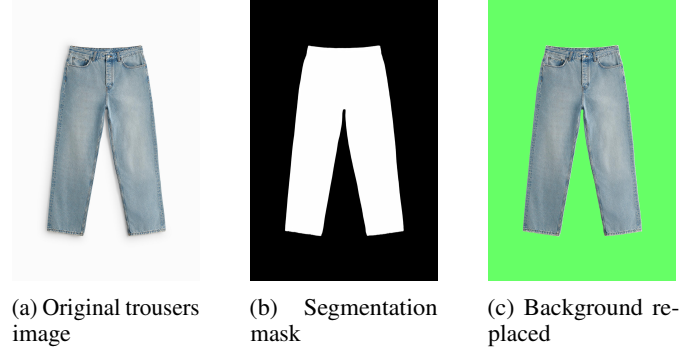


Figure 5: Illustration of the background replacement process using green screen technique

This helps reduce noise by providing strong contrast between the garment and the background. The results are shown below in Table 7.

Table 7: Average absolute ratio error across garment types (Background Replacement)

| Measurement | Short sleeve top | Long sleeve top | Vest | Shorts | Trousers | Skirt | Short sleeve dress | Long sleeve dress | Vest dress |
|------------------------------|------------------|-----------------|-------|--------|----------|-------|--------------------|-------------------|------------|
| Back width / Arm width | 0.078 | 0.134 | – | – | – | – | – | – | – |
| Chest / Front length | 0.025 | 0.041 | 0.108 | – | – | – | 0.031 | 0.064 | 0.017 |
| Front length / Chest | 0.025 | 0.039 | 0.019 | – | – | – | 0.031 | 0.060 | 0.017 |
| Hip / Waist | – | – | – | 0.025 | 0.018 | 0.218 | 0.240 | 0.211 | 0.097 |
| Sleeve length / Front length | 0.055 | 0.017 | – | – | – | – | 0.058 | 0.061 | – |

We can observe that for garments such as tops and dresses-where the original measurement errors (excluding Hip/Waist) were already relatively low (ranging from 0.015 to 0.107)-this method did not lead to consistent improvements. In some cases, it even introduced slight performance degradation, likely due to over-simplification of background cues that were not interfering in the first place. Similarly, for the Hip/Waist measurement in dresses, where the initial errors were already relatively high, background replacement had minimal impact. This suggests that the landmark detection model performs consistently before and after background modification, and that the persistent error may stem from the inherent ambiguity in defining the hip position, even in manual annotations.

However, the method proved effective for specific garment types with challenging visual conditions. In vests, for example, the error in the Front length/Chest ratio decreased significantly from 0.56 to 0.19. This improvement can be attributed to the presence of strong shadows near the hem in the original images, which the green background successfully masked, leading to clearer garment boundaries. For shorts and skirts where the original model struggled with accurate landmark detection, the background replacement also led to improved measurement accuracy, demonstrating the method’s potential in scenarios with complex or noisy image regions.

5.3.3 Gaussian Blur-Based Landmark Refinement

We also implemented a refinement step for the detected keypoints using a Gaussian blur smoothing method (window size = 5). This method operates on the predicted heatmaps to reduce spurious peaks and smooth the spatial distribution, enabling more stable and consistent landmark localization-particularly useful in regions where multiple nearby activations could lead to misidentification. The results are shown in Table 8.

After adding the refinement step on top of the original landmark detection module, we observed only minimal improvements. In most cases, the change in average absolute ratio error ranged from -0.01 to 0.031 , indicating that the refinement had limited impact on overall measurement accuracy.

Table 8: Average absolute ratio error across garment types (Blurring refinement)

| Measurement | Short sleeve top | Long sleeve top | Vest | Shorts | Trousers | Skirt | Short sleeve dress | Long sleeve dress | Vest dress |
|------------------------------|------------------|-----------------|-------|--------|----------|-------|--------------------|-------------------|------------|
| Back width / Arm width | 0.070 | 0.140 | – | – | – | – | – | – | – |
| Chest / Front length | 0.032 | 0.033 | 0.049 | – | – | – | 0.034 | 0.059 | 0.023 |
| Front length / Chest | 0.032 | 0.032 | 0.059 | – | – | – | 0.033 | 0.055 | 0.024 |
| Hip / Waist | – | – | – | 0.245 | 0.027 | 0.827 | 0.261 | 0.242 | 0.096 |
| Sleeve length / Front length | 0.038 | 0.018 | – | – | – | – | 0.059 | 0.043 | – |

6 Discussion

Our classification experiments demonstrate that all three models, CNN-3, CNN-4, and tinyViT, achieve excellent performance on the held-out test set, with overall accuracies of 94.58%, 95.33%, and 95.76%, respectively, and weighted F1 scores above 0.94. tinyViT delivers the best accuracy and F1, particularly on challenging categories such as short sleeve tops and skirts. However, this improvement comes at a substantial computational cost: on our hardware (two Nvidia T4 GPUs, each has 16GB memory; CPU memory is 32GB) CNN-3 completes training in about 4 hours, whereas CNN-4 and tinyViT require approximately 8 hours and 9 hours, respectively. Moreover, all models continue to struggle with visually similar or highly variable classes - most notably long sleeve dresses and vest dresses - indicating that further gains may depend on more advanced architectures or enhanced dataset quality (e.g. balancing, augmentation, or higher-resolution images).

Fine-tuning on Zara data further refined the model performance, with tinyViT demonstrating the best generalization ability across domains, exhibiting minimal performance drop on the Nordstrom & Myntra test sets compared to CNN-3 and CNN-4, which experienced a more significant decline. This suggests that transformer-based models like tinyViT are more robust to domain shifts after fine-tuning.

The pretrained HRNet-based landmark estimator achieves strong results on the DeepFashion2 benchmark [Sun et al., 2019, Xiao et al., 2018, Wang et al., 2019]. This confirms its high spatial precision for predefined landmarks. Our framework’s support for user-defined landmarks adds flexibility, but can fail in edge cases - such as extreme occlusions or unconventional garment shapes - where simple distance-based rules break down. Future improvements could include learnable refinement modules or interactive correction tools to ensure robust landmark prediction in all scenarios.

7 Conclusion & Future Work

In this work, we have developed a comprehensive framework for fashion image analysis encompassing four primary components: measurement instruction generation, classification, segmentation, and landmark extraction. The measurement instruction generation module automates the creation of detailed garment measurement guidelines, facilitating standardized assessments. The classification module leverages a large-scale, high-quality dataset of fashion images, which we curated and made publicly available [Yuan, 2025a]. This dataset serves as a valuable resource for training and evaluating fashion classification models. Moreover, we fine-tuned our models with the Zara dataset [Yuan, 2025b] for better downstream tasks such as landmark extraction. The landmark extraction component employs a function capable of detecting customized landmarks on garments, providing precise localization for further analysis. Additionally, we have implemented a segmentation module that delineates garment boundaries, aiding in the isolation of clothing items from complex backgrounds.

Our framework is designed with flexibility, scalability, and modularity in mind. Each component operates independently, allowing for easy updates and integration of new methodologies. This structure ensures that the system can adapt to evolving research and industry needs, accommodating various fashion analysis tasks. The modular approach also facilitates the replacement or enhancement of individual modules without disrupting the overall system, promoting long-term usability and extensibility. Moreover, we have also developed user-friendly web interfaces for these modules, enabling individuals with limited coding experience to easily utilize our framework and benefit from its research outcomes.

Despite these advancements, there are areas where our framework can be improved. In the classification module, while our base models have achieved commendable accuracy, there is still potential for further enhancement. Recent studies have reported higher performance metrics on benchmark datasets. For instance, the CNN-3-128 model achieved an impressive accuracy of 99.44% on the Fashion-MNIST dataset, outperforming previous benchmarks [Mukhamediev, 2024]. Additionally, exploring more sophisticated architectures, such as hybrid transformer models, could yield even

better results. For example, a recent study achieved 99.83% accuracy in food image classification by leveraging global and local feature fusion through transformer-based models [Jagadesh et al., 2025]. Incorporating such advanced techniques into our framework could potentially enhance classification accuracy even further. However, our fine-tuning experiments have shown that while model performance improves on the training set, it may lead to a decline in metrics on the original test set, indicating that further optimization of fine-tuning strategies and domain adaptation techniques will be necessary for robust performance across diverse datasets.

The landmark extraction module, although effective for predefined landmarks in datasets like DeepFashion2, may encounter challenges when applied to customized landmarks. The pretrained models excel in scenarios where landmarks are consistent and well-defined. However, for customized landmarks, the variability in garment designs and poses can lead to inaccuracies. Research into multimodal generative AI techniques, such as those explored in MetaCloth, offers promising avenues for enhancing the adaptability and robustness of landmark detection in diverse contexts [Ge et al., 2022].

Looking forward, several directions for future work are evident. One promising avenue is the development of an intelligent agent similar to GPT, capable of autonomously executing the entire fashion analysis pipeline. Such an agent would not only perform tasks with high accuracy but also handle edge cases effectively, learning from diverse datasets and adapting to new scenarios. Innovations in multimodal learning and few-shot learning, as demonstrated by models like MetaCloth, could be crucial in training this agent to generalize across various fashion items and contexts.

Another key direction involves enhancing the tunability of our framework. This would allow users to fine-tune the models for classification, segmentation, and landmark extraction to accommodate new garment types or variations. By incorporating a dynamic system that can automatically adjust the model parameters based on new data, the framework would be able to classify, segment, or extract landmarks from previously unseen or evolving garment categories. This flexibility would ensure the system’s continued relevance as new fashion items emerge and could significantly improve its adaptability in real-world applications.

In conclusion, our framework represents a significant step forward in the automation of fashion image analysis. By addressing current limitations and exploring innovative solutions, we aim to create a more robust and adaptable system that can meet the dynamic challenges of the fashion industry.

8 Computational Detail & Source Code

We recommend using Python version 3.11 or greater. The following Python packages and their corresponding versions are required:

- tqdm==4.67.1
- pandas==2.2.2
- numpy==2.0.0
- torch==2.7.0
- torchvision==0.22.0
- scikit-learn==1.6.1
- scipy==1.15.2
- Pillow==11.1.0
- matplotlib==3.10.0
- transformers==4.50.3
- kornia==0.8.0
- timm==1.0.15
- einops==0.8.1
- shapely==2.1.1
- opencv-python==4.11.0.86

The source code can be found at: <https://github.com/lygitdata/GarmentIQ>.

9 Acknowledgment

We sincerely thank Adrián González-Sieira and Laura Rodríguez for their invaluable suggestions and continuous support throughout the research. We are also grateful to everyone at ETH Zürich and the ETH AI Center for providing this great opportunity and coordinating between the company and the academic supervisors.

Throughout this journey, we are deeply indebted to our loved ones and dear friends, whose steadfast encouragement has been a constant source of strength. Last but not least, we leveraged generative AI to refine the linguistic quality of this paper.

References

- Sophie Miell, Simeon Gill, and Delia Vazquez. Enabling the digital fashion consumer through fit and sizing technology. *Journal of Global Fashion Marketing*, 9:9–23, 11 2018. doi:10.1080/20932685.2017.1399083.
- Cris Ian Zbavitel. Towards automated garment measurements in the wild using landmark and depth estimation, 12 2024. URL <http://hdl.handle.net/10342/13834>.
- Paweł Kowaleczko, Przemysław Rokita, and Marcin Szczuka. Neural network enhanced automatic garment measurement system. *Annals of Computer Science and Information Systems*, 32:33–38, 09 2022. doi:10.15439/2022f8.
- Agne Paulauskaite-Taraseviciene, Eimantas Noreika, Ramunas Purtokas, Ingrida Lagzdinyte-Budnike, Vytautas Daniulaitis, and Ruta Salickaite-Zukauskienė. An intelligent solution for automatic garment measurement using image recognition technologies. *Applied Sciences*, 12(9), 2022. ISSN 2076-3417. doi:10.3390/app12094470. URL <https://www.mdpi.com/2076-3417/12/9/4470>.
- Ah Pun Chan, Wai Ching Chu, Kwan Yu Lo, and Kai Yuen Cheong. Improving the apparel virtual size fitting prediction under psychographic characteristics and 3d body measurements using artificial neural network. *AHFE international*, 01 2022. doi:10.54941/ahfe1001543.
- Greeshma K V and Sreekumar K. Fashion-mnist classification based on hog feature descriptor using svm. *International Journal of Innovative Technology and Exploring Engineering*, 8, 03 2019. URL <https://www.ijitee.org/wp-content/uploads/papers/v8i5/E3075038519.pdf>.
- Jun Xu, Yumeng Wei, Aichun Wang, Heng Zhao, and Damien Lefloch. Analysis of clothing image classification models: A comparison study between traditional machine learning and deep learning models. *Fibres & Textiles in Eastern Europe*, 30:66–78, 10 2022. doi:10.2478/ftce-2022-0046.
- M. S. Saranya and P. Geetha. Fashion image classification using deep convolution neural network. *IFIP Advances in Information and Communication Technology*, pages 116–127, 2022. doi:10.1007/978-3-031-11633-9_10.
- Zhiyu Zhou, Mingxuan Liu, Wenxiong Deng, Yaming Wang, and Zefei Zhu. Clothing image classification algorithm based on convolutional neural network and optimized regularized extreme learning machine. *Textile Research Journal*, page 004051752211154, 08 2022. doi:10.1177/00405175221115472.
- Sheng Guo, Weilin Huang, Xiao Zhang, Prasanna Srikhanta, Yin Cui, Yuan Li, Matthew R. Scott, Hartwig Adam, and Serge Belongie. The imaterialist fashion attribute dataset, 2019. URL <https://arxiv.org/abs/1906.05750>.
- Zhong Xiang, Chenglin Zhu, Miao Qian, Yujia Shen, and Yizhou Shao. Fashionsegnet: a model for high-precision semantic segmentation of clothing images. *The Visual Computer*, 40:1711–1727, 05 2023. doi:10.1007/s00371-023-02881-3.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. URL <https://arxiv.org/abs/1512.03385>.
- Gabriela Vozáriková, Richard Staňa, and Gabriel Semanišin. Clothing parsing using extended u-net. *Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 15–24, 2021. doi:10.5220/0010177700150024.
- Tianping Li, Zhaotong Cui, and Hua Zhang. Semantic segmentation feature fusion network based on transformer. *Scientific Reports*, 15, 02 2025. doi:10.1038/s41598-025-90518-x.
- Ziwei Liu, Sijie Yan, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Fashion landmark detection in the wild, 2016. URL <https://arxiv.org/abs/1608.03049>.
- Yuying Ge, Ruimao Zhang, Lingyun Wu, Xiaogang Wang, Xiaoou Tang, and Ping Luo. Deepfashion2: A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images, 2019. URL <https://arxiv.org/abs/1901.07973>.
- Kai Petersen, Peter Roos, Staffan Nyström, and Per Runeson. Early identification of bottlenecks in very large scale system of systems software development. *J. Softw. Evol. Process*, 26(12):1150–1171, December 2014. ISSN 2047-7473. doi:10.1002/smr.1653. URL <https://doi.org/10.1002/smr.1653>.
- Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. URL <https://arxiv.org/abs/1409.1556>.
- Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention, 2021.
- Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. Visual transformers: Token-based image representation and processing for computer vision, 2020.

- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
- Peng Zheng, Dehong Gao, Deng-Ping Fan, Li Liu, Jorma Laaksonen, Wanli Ouyang, and Nicu Sebe. Bilateral reference for high-resolution dichotomous image segmentation. *CAAI Artificial Intelligence Research*, 3:9150038, 2024.
- Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.
- Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *European Conference on Computer Vision (ECCV)*, 2018.
- Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Minghui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. Deep high-resolution representation learning for visual recognition. *TPAMI*, 2019.
- Li Yuan. Nordstrom & myntra clothes image data - garmentiq, 2025a. URL <https://www.kaggle.com/ds/7099732>.
- Param Aggarwal. Fashion product images dataset, 2019. URL <https://www.kaggle.com/ds/139630>.
- Li Yuan. Zara clothes image data, 2025b. URL <https://www.kaggle.com/dsv/11787792>.
- Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL <https://arxiv.org/abs/1711.05101>.
- Ravil I. Mukhamediev. State-of-the-art results with the fashion-mnist dataset. *Mathematics*, 12:3174, 10 2024. doi:10.3390/math12203174.
- B N Jagadesh, Srihari Varma Mantena, Asha P Sathe, T Prabhakara Rao, Kranthi Kumar Lella, Shyam Sunder Pabboju, and Ramesh Vatambeti. Enhancing food recognition accuracy using hybrid transformer models and image preprocessing techniques. *Scientific Reports*, 15, 02 2025. doi:10.1038/s41598-025-90244-4. URL <https://pmc.ncbi.nlm.nih.gov/articles/PMC11829996/>.
- Yuying Ge, Ruimao Zhang, and Ping Luo. Metacloth: Learning unseen tasks of dense fashion landmark detection from a few samples. *IEEE Transactions on Image Processing*, 31:1120–1133, 2022. ISSN 1941-0042. doi:10.1109/tip.2021.3131033. URL <http://dx.doi.org/10.1109/TIP.2021.3131033>.

Appendices

A Measurement Instruction JSON Schema

```

1 {
2   "$schema": "http://json-schema.org/draft-07/schema#",
3   "title": "Garment Measurement Schema",
4   "description": "One or more garments, each with landmarks (all required) and
5     measurements (all required)",
6   "type": "object",
7   "patternProperties": {
8     "~.+": {
9       "$ref": "#/definitions/garment"
10    }
11  },
12  "additionalProperties": false,
13  "definitions": {
14    "garment": {
15      "type": "object",
16      "required": [
17        "landmarks",
18        "measurements"
19      ],
20      "properties": {
21        "landmarks": {
22          "$ref": "#/definitions/landmarks"
23        },
24        "measurements": {
25          "$ref": "#/definitions/measurements"
26        }
27      },
28      "additionalProperties": false
29    },
30    "landmarks": {
31      "type": "object",
32      "description": "Map of point-ID -> landmark",
33      "minProperties": 1,
34      "patternProperties": {
35        "~[0-9]+": {
36          "$ref": "#/definitions/landmark"
37        }
38      },
39      "additionalProperties": false
40    },
41    "landmark": {
42      "type": "object",
43      "description": "A single landmark on the garment",
44      "required": [
45        "predefined",
46        "description",
47        "x",
48        "y"
49      ],
50      "properties": {
51        "predefined": {
52          "type": "boolean",
53          "description": "true = model-detected; false = customized"
54        },
55        "description": {
56          "type": "string",
57          "description": "Semantic label for this point"
58        },
59        "x": {
60          "type": "number",

```

```

60     "description": "X-coordinate in image space"
61   },
62   "y": {
63     "type": "number",
64     "description": "Y-coordinate in image space"
65   },
66   "neighbors": {
67     "type": "object",
68     "description": "Exactly two nearby landmarks, keyed by their IDs",
69     "minProperties": 2,
70     "maxProperties": 2,
71     "patternProperties": {
72       "^[0-9]+$": {
73         "$ref": "#/definitions/neighborLandmark"
74       }
75     },
76     "additionalProperties": false
77   },
78   "derivation": {
79     "$ref": "#/definitions/derivation",
80     "description": "How this custom point was derived"
81   }
82 },
83 "additionalProperties": false
84 },
85 "neighborLandmark": {
86   "type": "object",
87   "required": [
88     "predefined",
89     "description",
90     "x",
91     "y"
92   ],
93   "properties": {
94     "predefined": {
95       "type": "boolean"
96     },
97     "description": {
98       "type": "string"
99     },
100    "x": {
101      "type": "number"
102    },
103    "y": {
104      "type": "number"
105    }
106  },
107  "additionalProperties": false
108 },
109 "derivation": {
110   "type": "object",
111   "required": [
112     "function"
113   ],
114   "minProperties": 2,
115   "properties": {
116     "function": {
117       "type": "string",
118       "description": "Name of the derivation function"
119     }
120   },
121   "patternProperties": {
122     "^(?!function$)[a-zA-Z_][a-zA-Z0-9_]*$": {
123       "type": "string",
124       "description": "Named argument for the derivation function"

```

```

125     }
126   },
127   "additionalProperties": false
128 },
129 "measurements": {
130   "type": "object",
131   "description": "Map of measurement-name -> measurement definition",
132   "minProperties": 1,
133   "patternProperties": {
134     "^[a-z0-9_]+$": {
135       "$ref": "#/definitions/measurement"
136     }
137   },
138   "additionalProperties": false
139 },
140 "measurement": {
141   "type": "object",
142   "required": [
143     "landmarks",
144     "description"
145   ],
146   "properties": {
147     "landmarks": {
148       "type": "object",
149       "required": [
150         "start",
151         "end"
152       ],
153       "properties": {
154         "start": {
155           "type": "string",
156           "pattern": "^[0-9]+$"
157         },
158         "end": {
159           "type": "string",
160           "pattern": "^[0-9]+$"
161         }
162       },
163       "additionalProperties": false
164     },
165     "description": {
166       "type": "string"
167     }
168   },
169   "additionalProperties": false
170 }
171 }
172 }

```

B Short Sleeve Top Measurement Instruction in JSON Format

```

1 {
2   "short sleeve top": {
3     "landmarks": {
4       "2": {
5         "predefined": true,
6         "description": "neck_left_outer",
7         "x": 83,
8         "y": 15
9       },
10      "7": {
11        "predefined": true,
12        "description": "shoulder_left_top",
13        "x": 58,
14        "y": 25
15      },
16      "12": {
17        "predefined": true,
18        "description": "armpit_left",
19        "x": 60,
20        "y": 65
21      },
22      "20": {
23        "predefined": true,
24        "description": "armpit_right",
25        "x": 140,
26        "y": 65
27      },
28      "23": {
29        "predefined": true,
30        "description": "cuff_right_outer",
31        "x": 170,
32        "y": 50
33      },
34      "25": {
35        "predefined": true,
36        "description": "shoulder_right_top",
37        "x": 143,
38        "y": 25
39      },
40      "26": {
41        "predefined": false,
42        "description": "custom_landmark",
43        "x": 82.12834930419922,
44        "y": 164.28207397460938,
45        "derivation": {
46          "function": "derive_keypoint_coord",
47          "p1_id": "2",
48          "p2_id": "4",
49          "p3_id": "16",
50          "p4_id": "15",
51          "p5_id": "17",
52          "direction": "parallel"
53        },
54        "neighbors": {
55          "15": {
56            "predefined": true,
57            "description": "hem_left",
58            "x": 56,
59            "y": 160
60          },
61          "16": {
62            "predefined": true,

```

```

63         "description": "hem_center",
64         "x": 100,
65         "y": 165
66     }
67 }
68 },
69 "27": {
70     "predefined": false,
71     "description": "custom_landmark",
72     "x": 46.79122543334961,
73     "y": 33.494102478027344,
74     "derivation": {
75         "function": "derive_keypoint_coord",
76         "p1_id": "12",
77         "p2_id": "7",
78         "p3_id": "9",
79         "p4_id": "7",
80         "p5_id": "9",
81         "direction": "perpendicular"
82     },
83     "neighbors": {
84         "7": {
85             "predefined": true,
86             "description": "shoulder_left_top",
87             "x": 58,
88             "y": 25
89         },
90         "8": {
91             "predefined": true,
92             "description": "sleeve_left_outer_mid",
93             "x": 40,
94             "y": 40
95         }
96     }
97 }
98 },
99 "measurements": {
100     "front length": {
101         "landmarks": {
102             "start": "2",
103             "end": "26"
104         },
105         "description": "26: intersection of vertical line from 2 to bottom contour"
106     },
107     "back width": {
108         "landmarks": {
109             "start": "7",
110             "end": "25"
111         },
112         "description": "/"
113     },
114     "chest": {
115         "landmarks": {
116             "start": "12",
117             "end": "20"
118         },
119         "description": "/"
120     },
121     "sleeve width": {
122         "landmarks": {
123             "start": "12",
124             "end": "27"
125         },

```

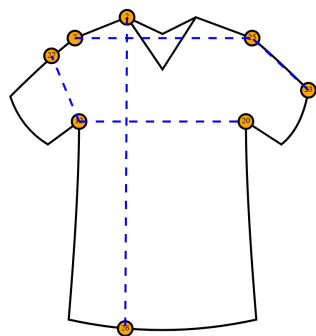
```
126         "description": "27: intersection of perpendicular line from 12 to sleeve  
127             contour"  
128     },  
129     "sleeve length": {  
130         "landmarks": {  
131             "start": "23",  
132             "end": "25"  
133         },  
134         "description": "/"  
135     }  
136 }  
137 }
```


C Short Sleeve Top Measurement Instruction in PDF Format

Measurement instruction: short sleeve top

Generated on: 05/13/2025, 10:04:25 PM

Generated via GarmentIQ.ly.gd.edu.kg - No liability assumed.



| Start | End | Name | Description |
|-------|-----|---------------|--|
| 2 | 26 | front length | 26: intersection of vertical line from 2 to bottom contour |
| 7 | 25 | back width | / |
| 12 | 20 | chest | / |
| 12 | 27 | sleeve width | 27: intersection of perpendicular line from 12 to sleeve contour |
| 23 | 25 | sleeve length | / |