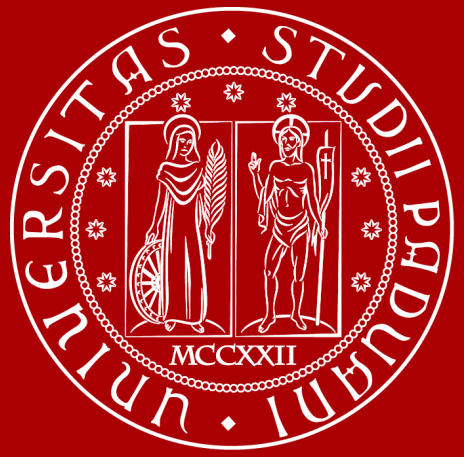
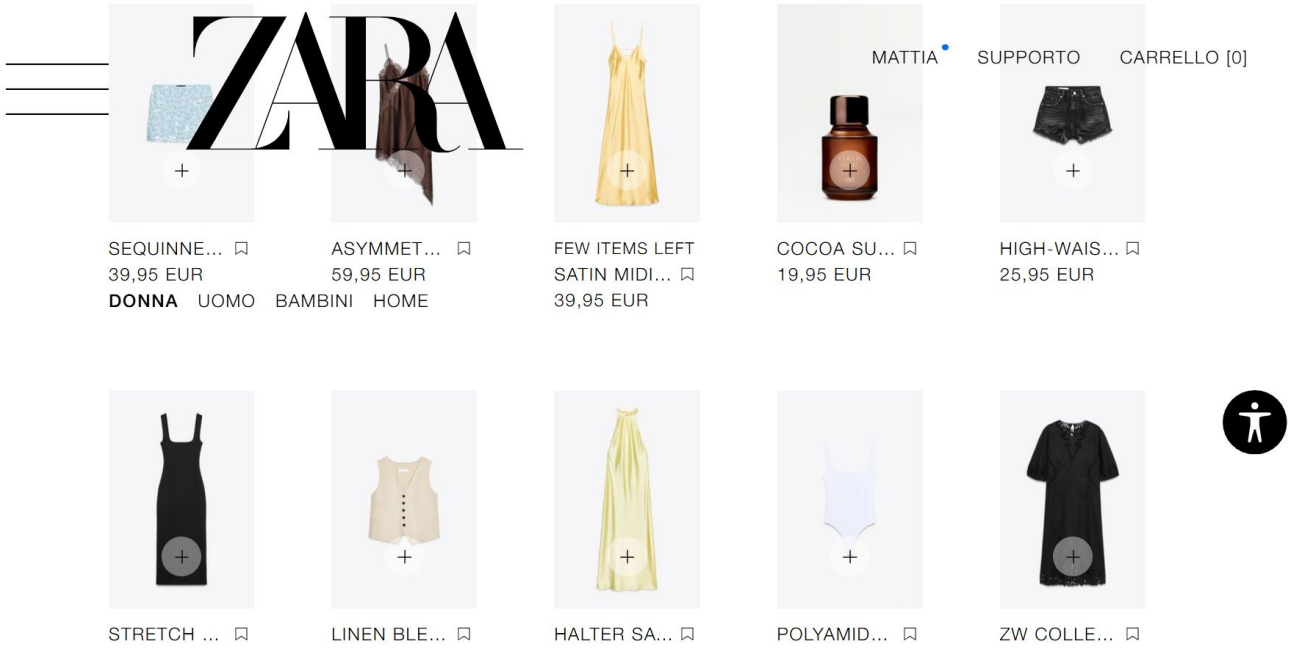


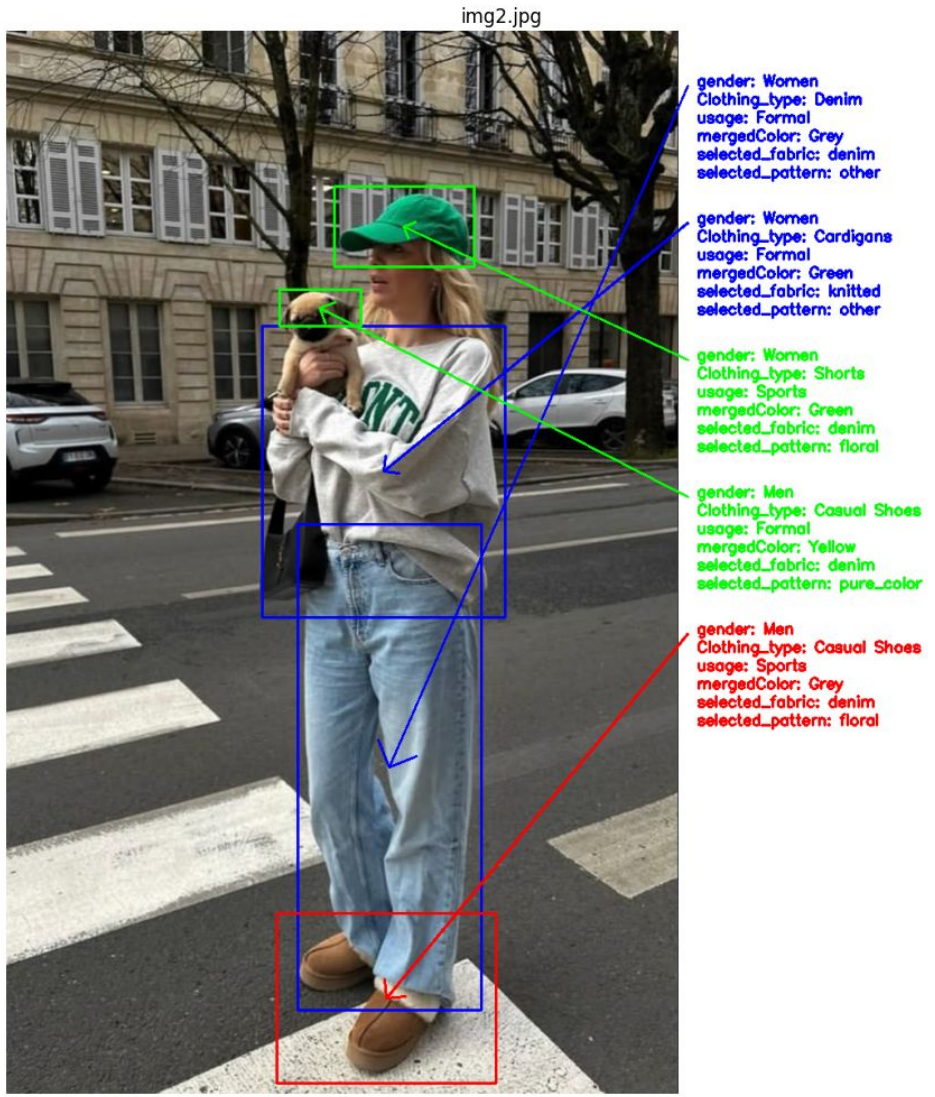
UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Modular Fashion Item Detection and Semantic Matching for Smart Outfit Retrieval

Mattia Roccatello, Ulaşcan Akbulut, Ayşenur Oya Özen

The project's purpose





Why Fashion Items? Why This Work?

Fashion understanding requires

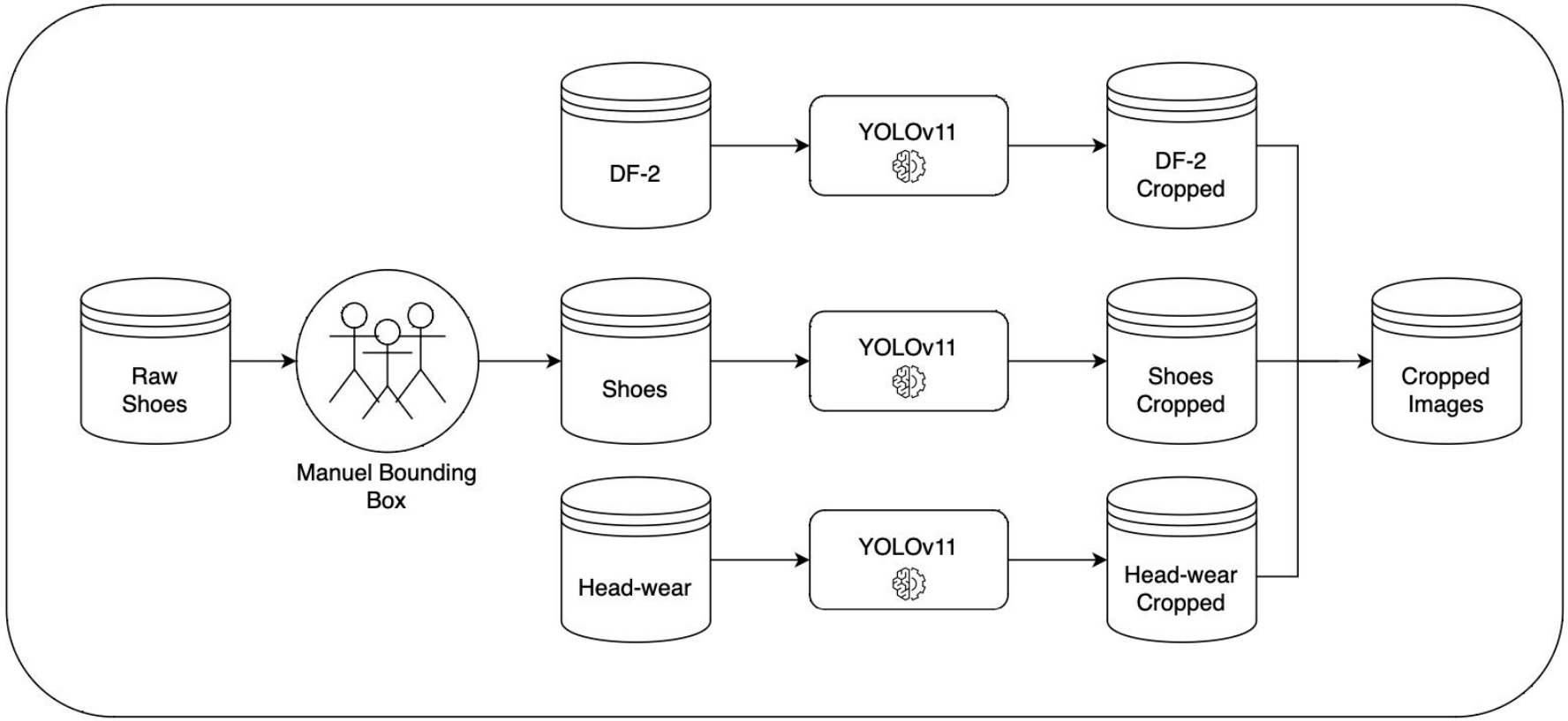
fine-grained detection + semantic understanding + retrieval
→ ALL TOGETHER

Traditional end-to-end models fail to generalize across heterogeneous fashion domains

We need a **modular**, **scalable**, and **interpretable** solution that can:

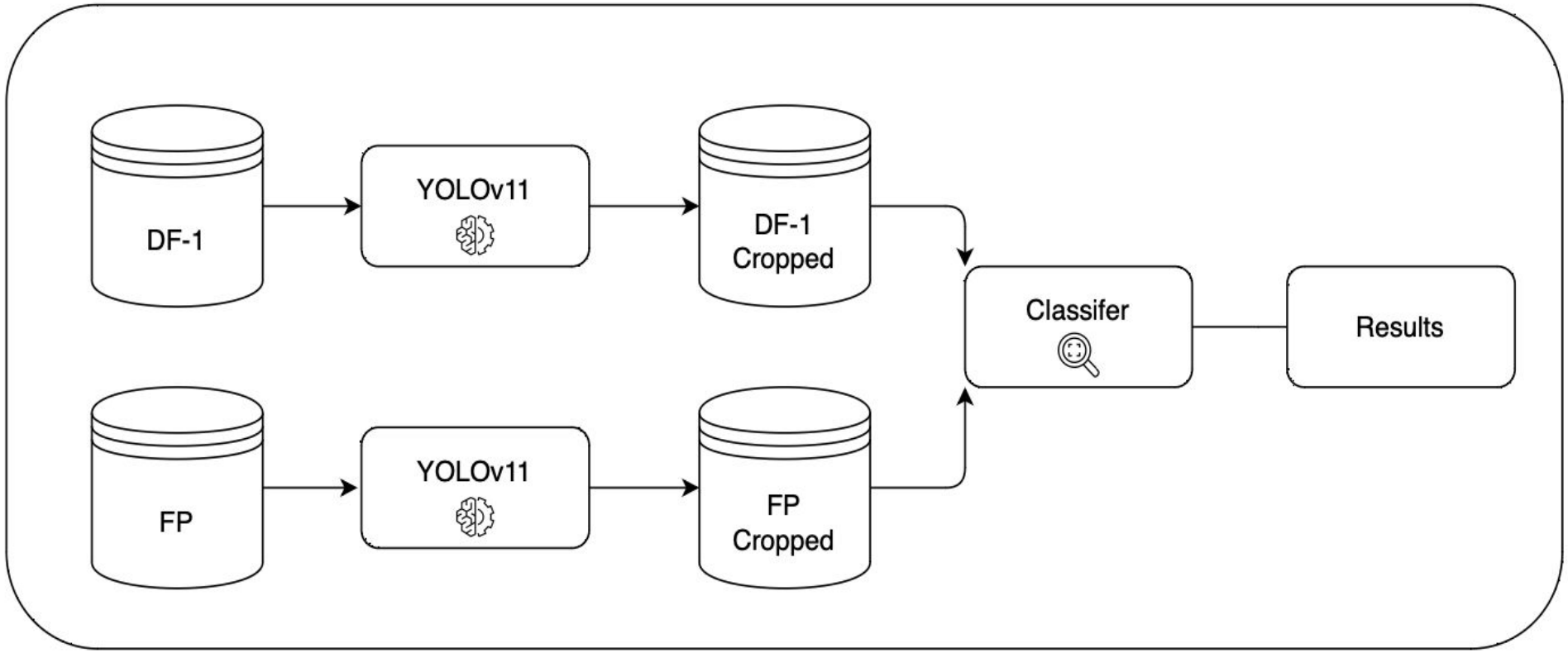
- Detect clothing items
- Classify attributes
- Perform semantic retrieval

Modular Multi-Stage Pipeline



Detection → Classification → Matching

Modular Multi-Stage Pipeline



Detection → Classification → Matching

Our Contributions

Modular pipeline

A two-stage system with separate models for detection and classification to overcome incomplete datasets

Dataset-specific detectors

Trained separate YOLOv11x models on DeepFashion2, Shoes, and Headwear datasets to ensure full clothing category coverage

Multi-head classifier

Predicts attributes like gender, color, fabric, pattern, and usage using interleaved learning from DeepFashion1 and Fashion Product datasets

Custom matching logic

Bridges instance-level crops with image-level labels using filename structure and predicted types

Contrastive pretraining

Self-supervised SimCLR-based encoder improves representation robustness for subtle features like fabric and pattern

Semantic retrieval

Uses embedding similarity to match query outfit items with similar pieces in a virtual wardrobe

Tackling Dataset - Detection Data

DeepFashion2

Headwear

Shoes

DeepFashion1

Fashion Products

Scale: 2, Viewpoint: 2, Occlusion: 1, Zoom_in: 1, Set: train, Category: trousers

800K images, 13 categories → Merged to 5 superclasses

Tackling Dataset - Detection Data

DeepFashion2

Headwear

Shoes

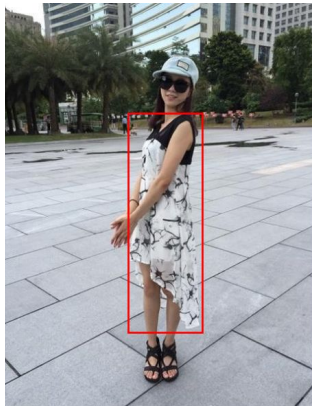
DeepFashion1

Fashion Products

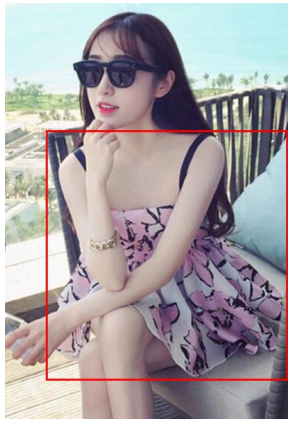
800K images, 13 categories → Merged to 5 superclasses

long sleeve dress

vest dress



sling



long sleeve outwear, short sleeve outwear	outwear
short sleeve dress, long sleeve dress, vest dress, sling, sling dress	dress
trousers, shorts, skirt	bottom
long sleeve top, short sleeve top	top
vest	vest

Tackling Dataset - Detection Data

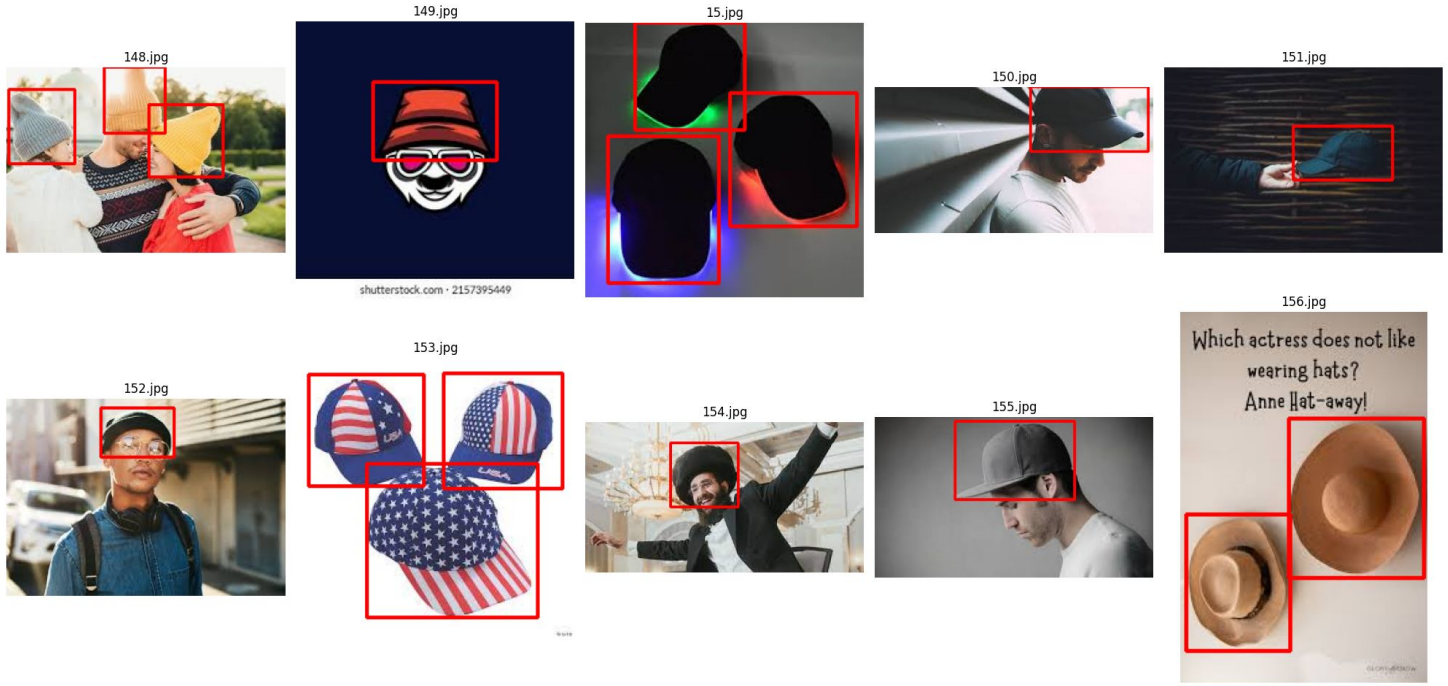
DeepFashion2

Headwear

Shoes

DeepFashion1

Fashion Products



Pascal VOC to YOLO format, 650 images

Tackling Dataset - Detection Data

DeepFashion2
Headwear
Shoes
DeepFashion1
Fashion Products

Manually labeled, ~12K images

Tackling Dataset - Classification Data

DeepFashion2

Headwear

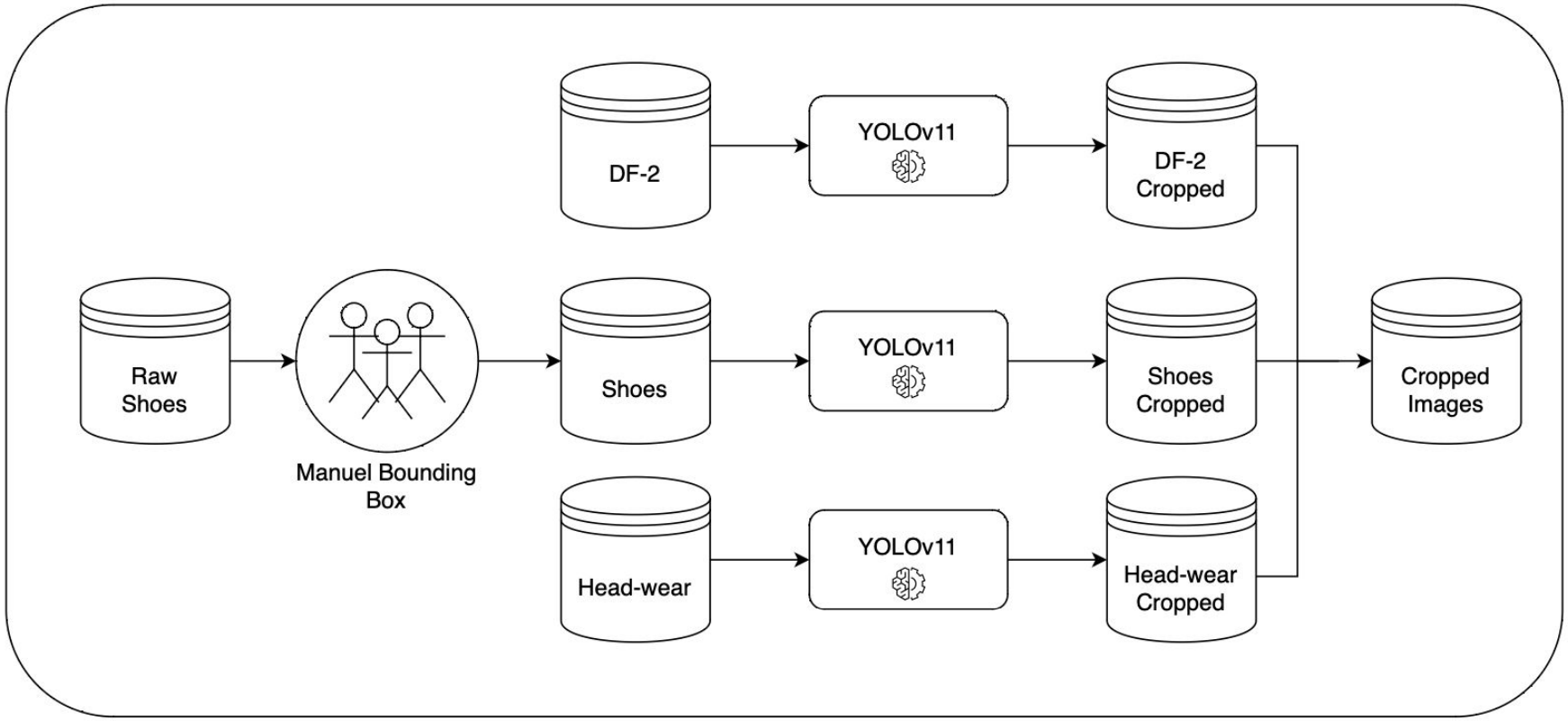
Shoes

DeepFashion1- Rich attributes but weak crop-to-label mapping

Fashion Products - Strong structured metadata

Attributes: gender, color, usage, article type

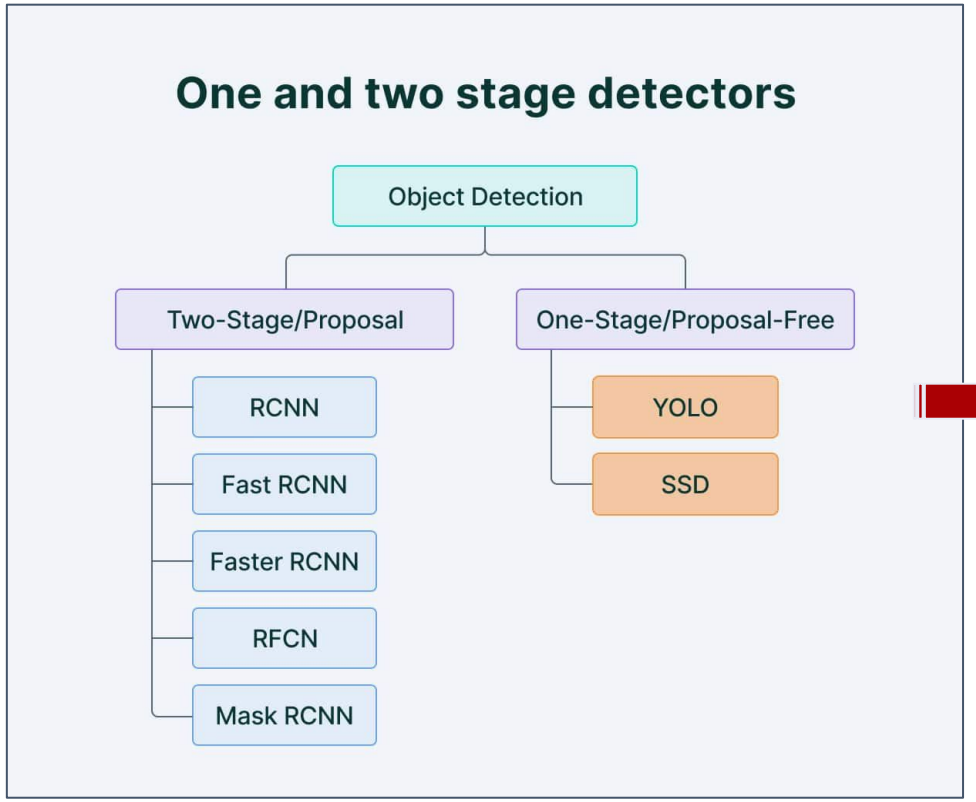
Modular Multi-Stage Pipeline - Detection Module



YOLOv11x Detection Models - Why YOLO?

Detects **what** objects are in an image and **where** they are, in a single forward pass

You Only Look Once !



It divides the input image into an S×S grid

Each grid cell directly predicts bounding boxes + class probabilities

YOLOv11x Detection Models - YOLOv11x Version

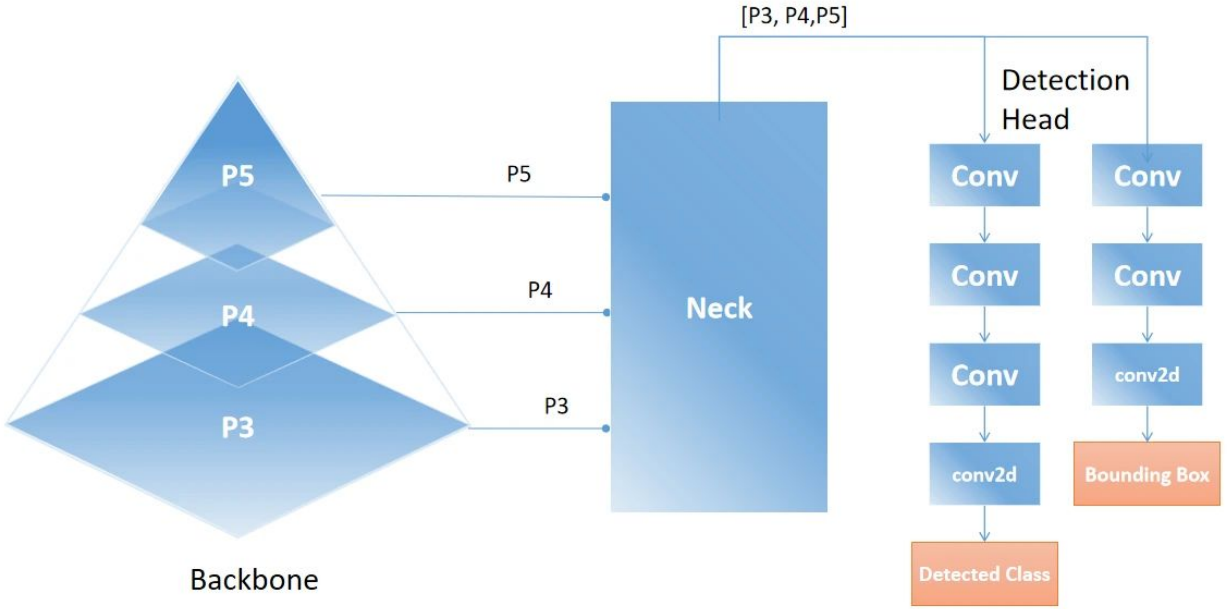
Version	Key Improvements
YOLOv1–v3	Basic object detection, faster than R-CNN
YOLOv4–v5	Added better backbones, anchor boxes, better mAP
YOLOv7 / v8	Ultralytics versions, modular, optimized
YOLOv11x	Our choice: high-capacity version optimized by Ultralytics, handles multiple datasets and small objects well

The "x" indicates it's the largest version

- More layers → better feature extraction
- Suited for many small or overlapping objects (like clothing)

Pretrained on COCO, fine-tuned on our datasets

YOLOv11x Detection Models - Architecture



Backbone → Neck → Detection Heads

Feature Extraction → Multi-Scale Feature Fusion → Detection Output

YOLOv11x Detection Models - Our Training Setup

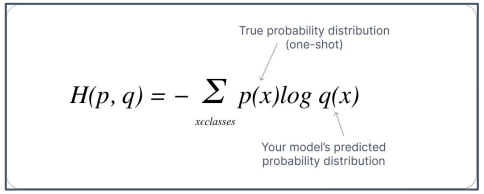
Component	Configuration
Model	YOLOv11x (pretrained on COCO)
Image Size	256 × 256
Epochs	57 (DeepFashion2), 20 (Cap - dataset), 150 (Shoes - dataset)
Batch Size	32 (auto-adjusted for memory)
Optimizer	SGD (momentum = 0.937)
LR Scheduler	Cosine Annealing
Losses	CIoU (bbox), BCE (objectness), CE (classes)
Validation Metrics	Accuracy, precision, recall, mAP@0.5, mAP@0.5-0.9
Data Augmentation	Mosaic, HSV, Flip, Affine
Logging	Weights&Biases
Early Stopping	Enabled

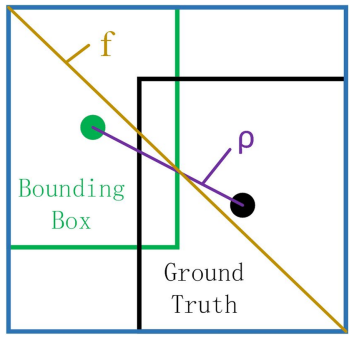
We trained **separate YOLOv11x models per dataset**

What we achieve with this modular design?

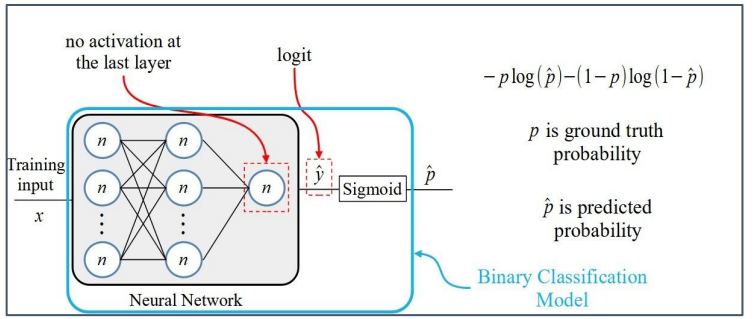
– prevent label conflicts
– allow specialized detection for each clothing category group

Cross-Entropy Loss



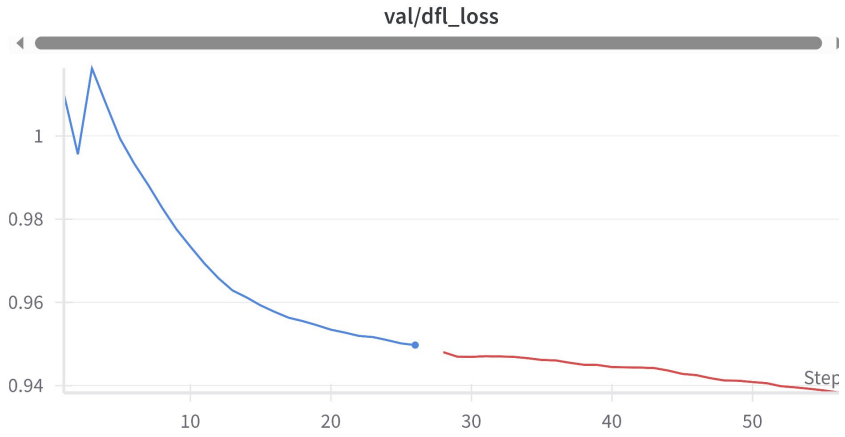


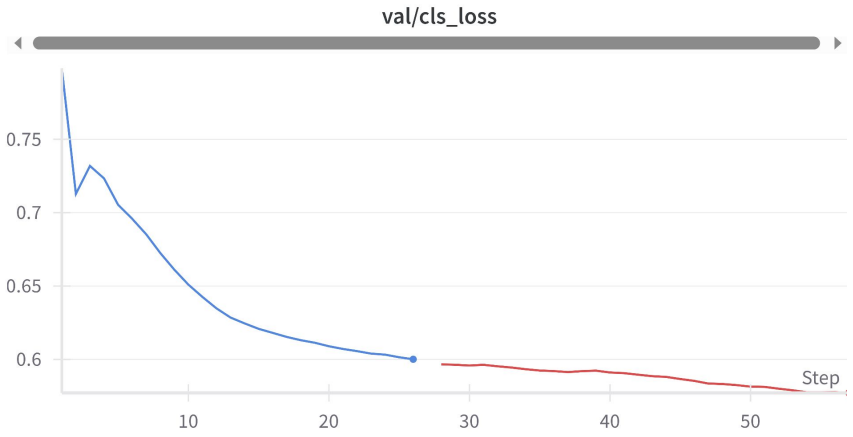
Complete Intersection over Union

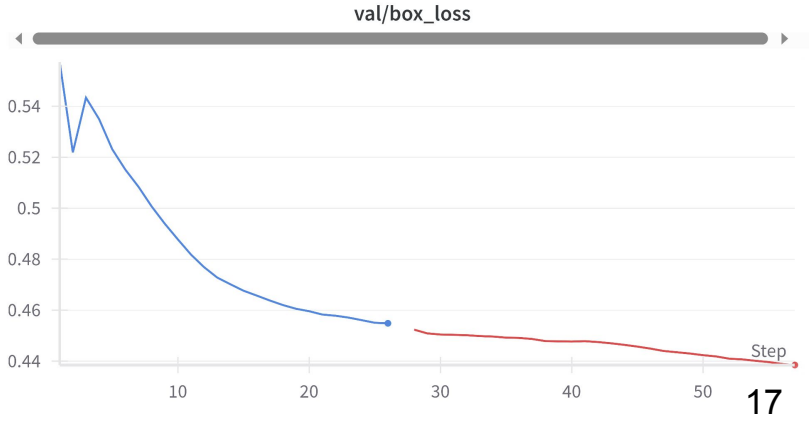


Binary Cross-Entropy Loss

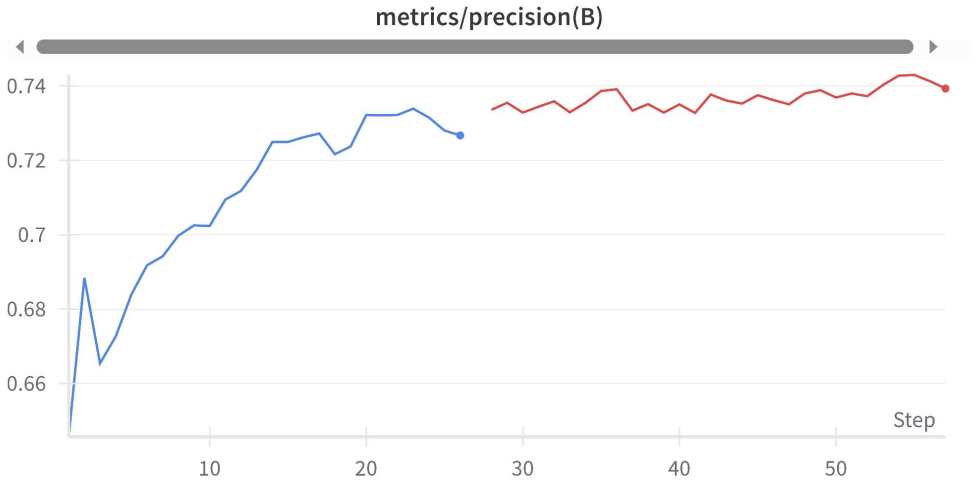
YOLOv11x Detection Models - DF2 Dataset Loss Curves

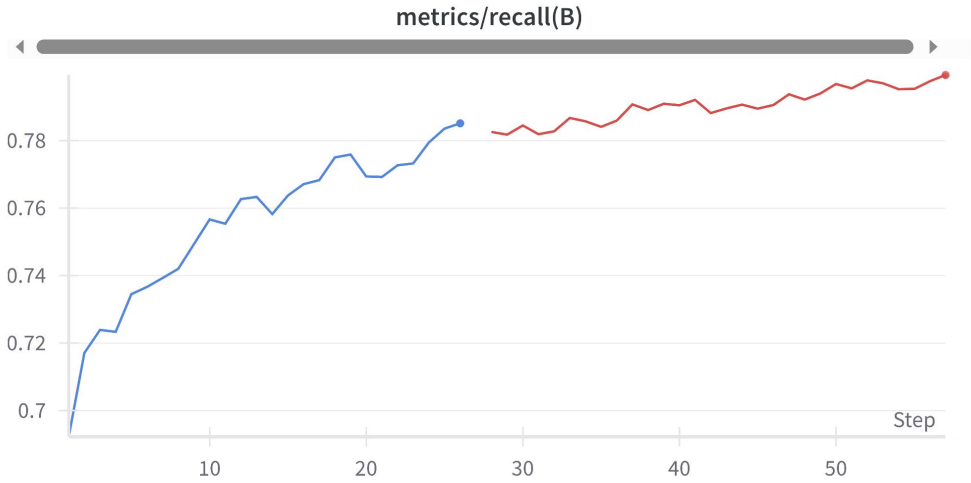


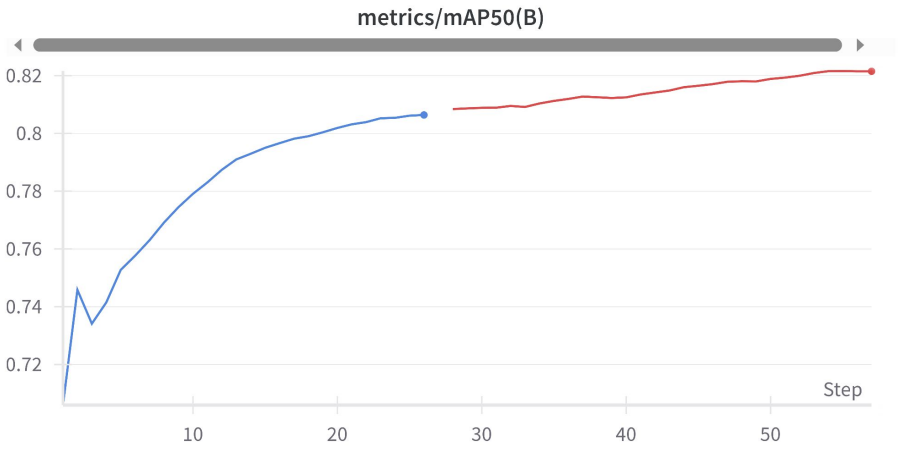


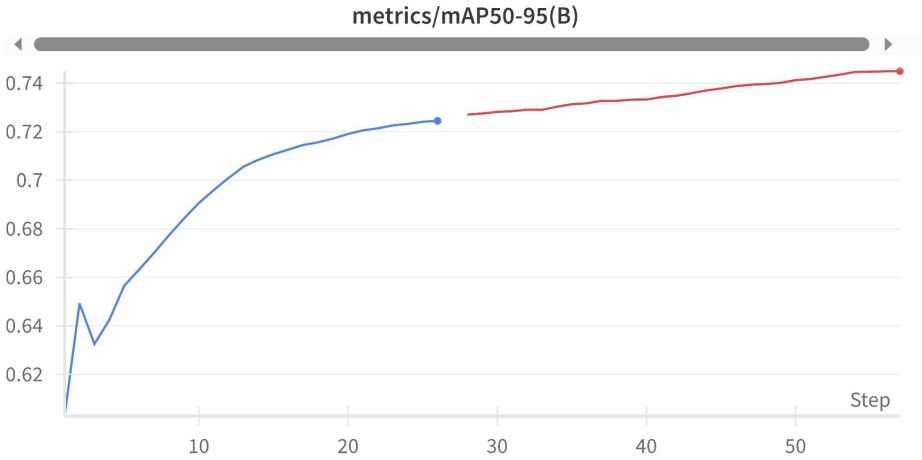


YOLOv11x Detection Models - DF2 Dataset Evaluation Metrics

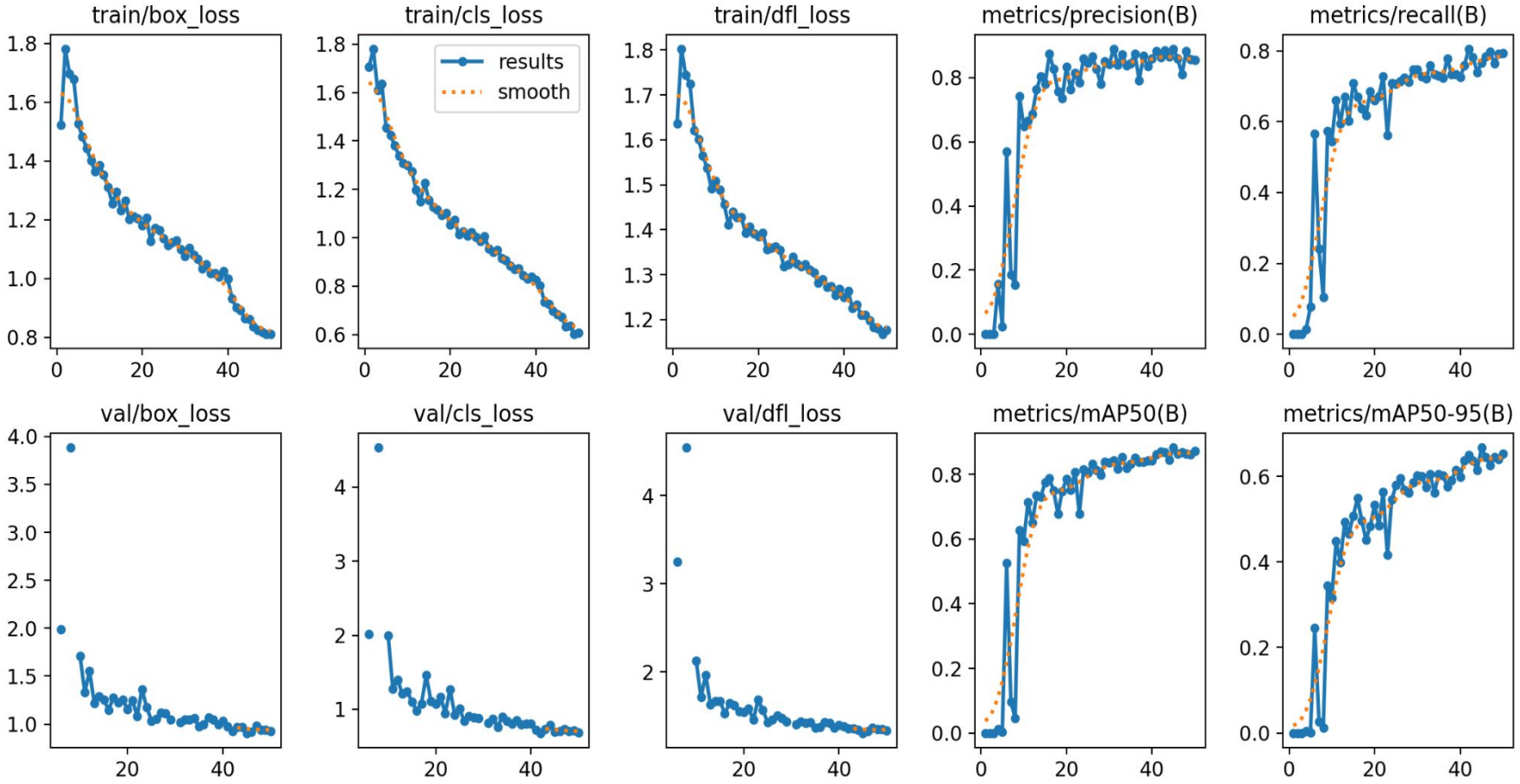




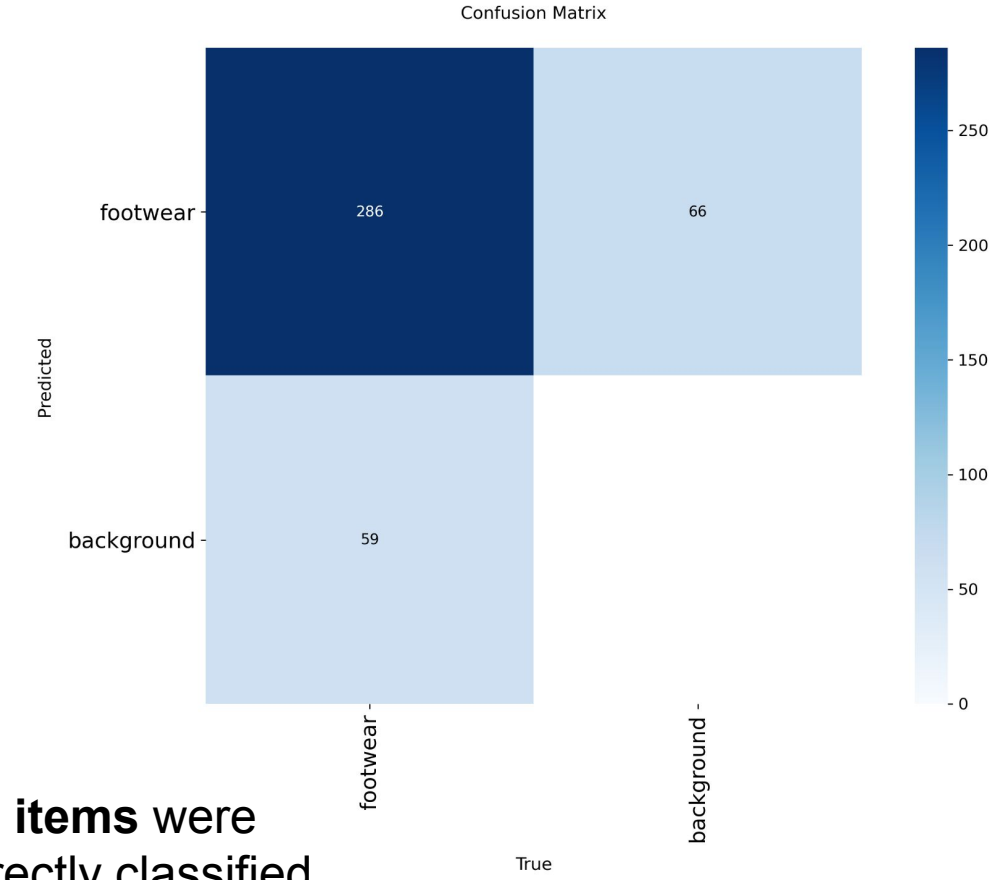


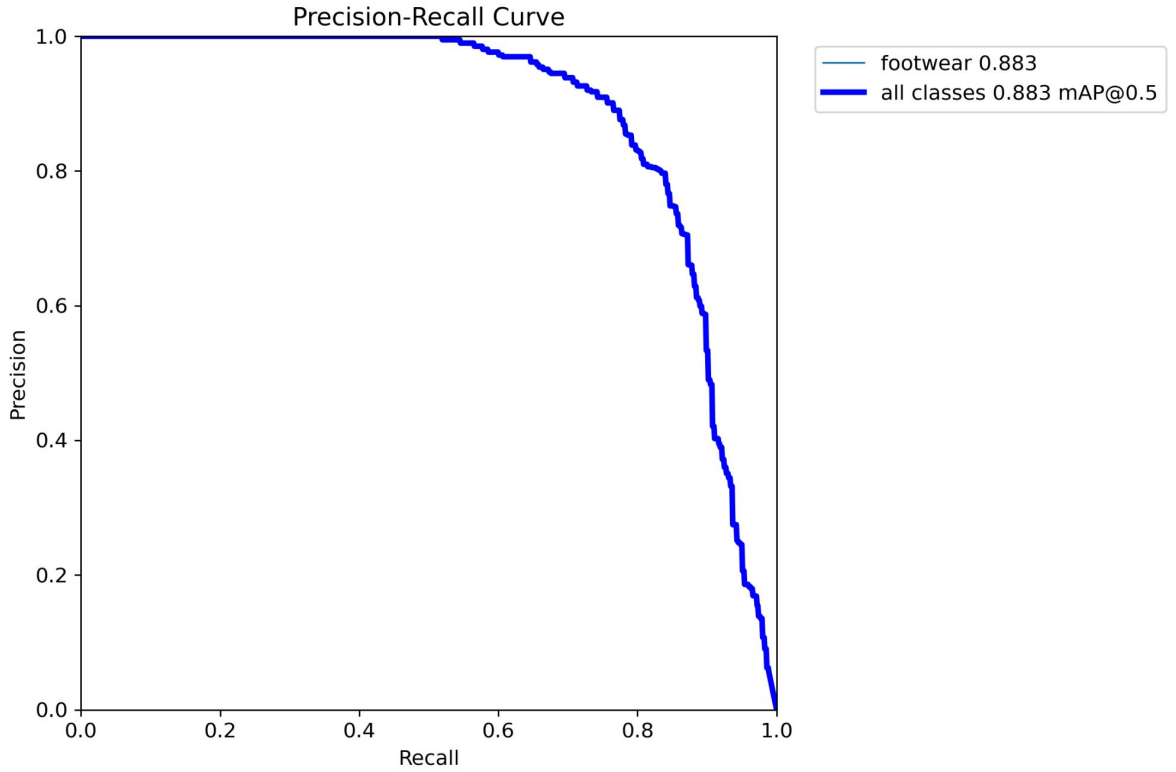


YOLOv11x Detection Models - Shoes Dataset Loss Curves



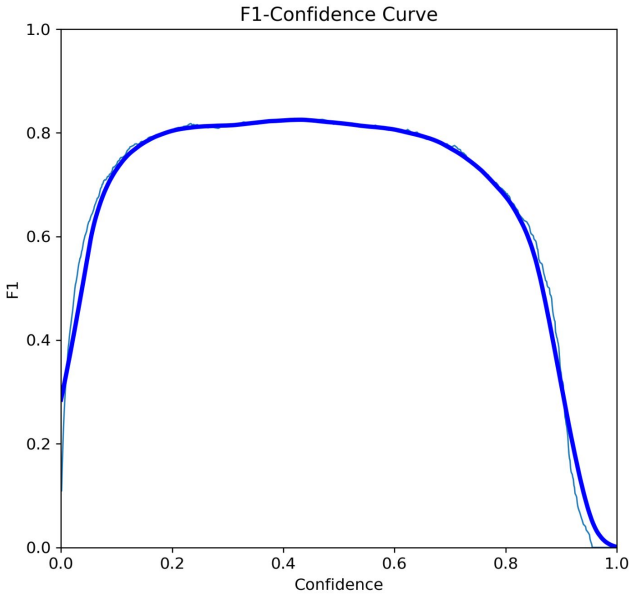
YOLOv11x Detection Models - Shoes Dataset Confusion Matrix & PR Curve

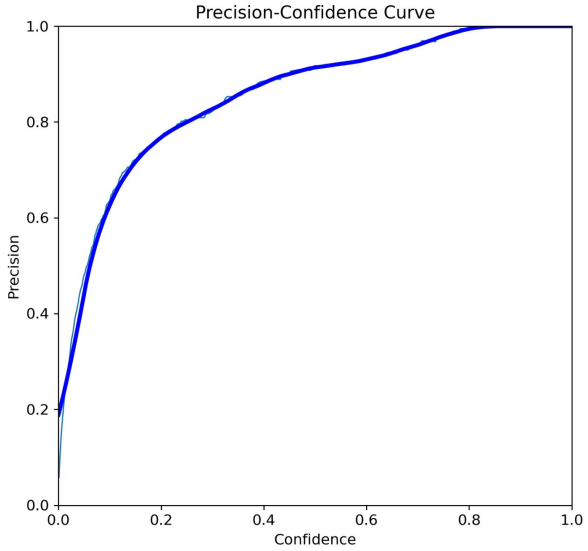


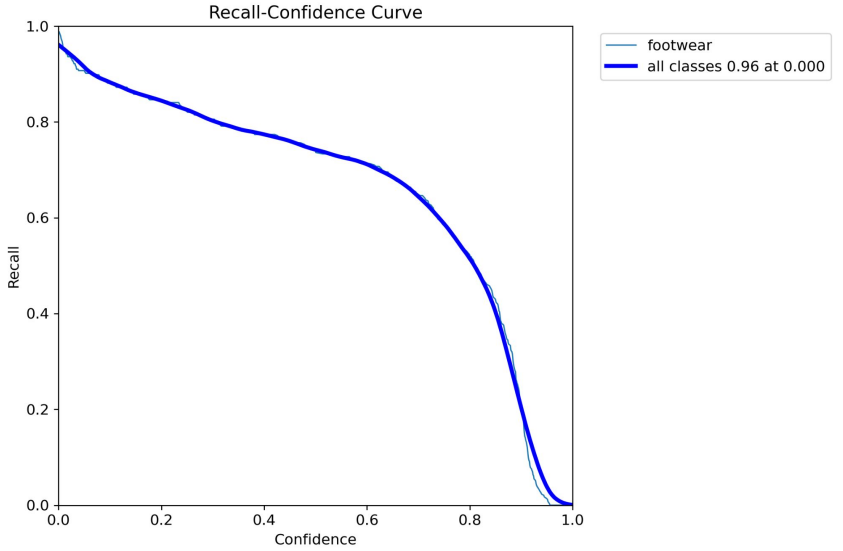


286 items were correctly classified as shoes!

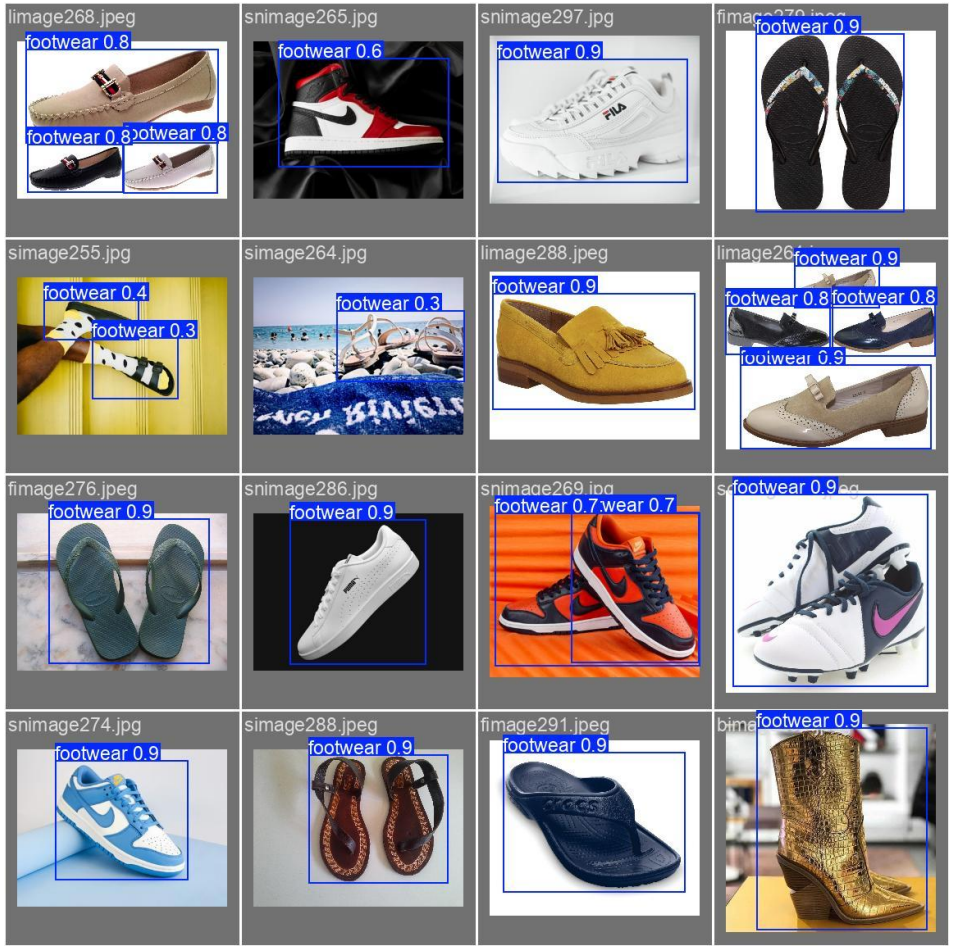
YOLOv11x Detection Models - Shoes Dataset Confidence-Based Metric Curves





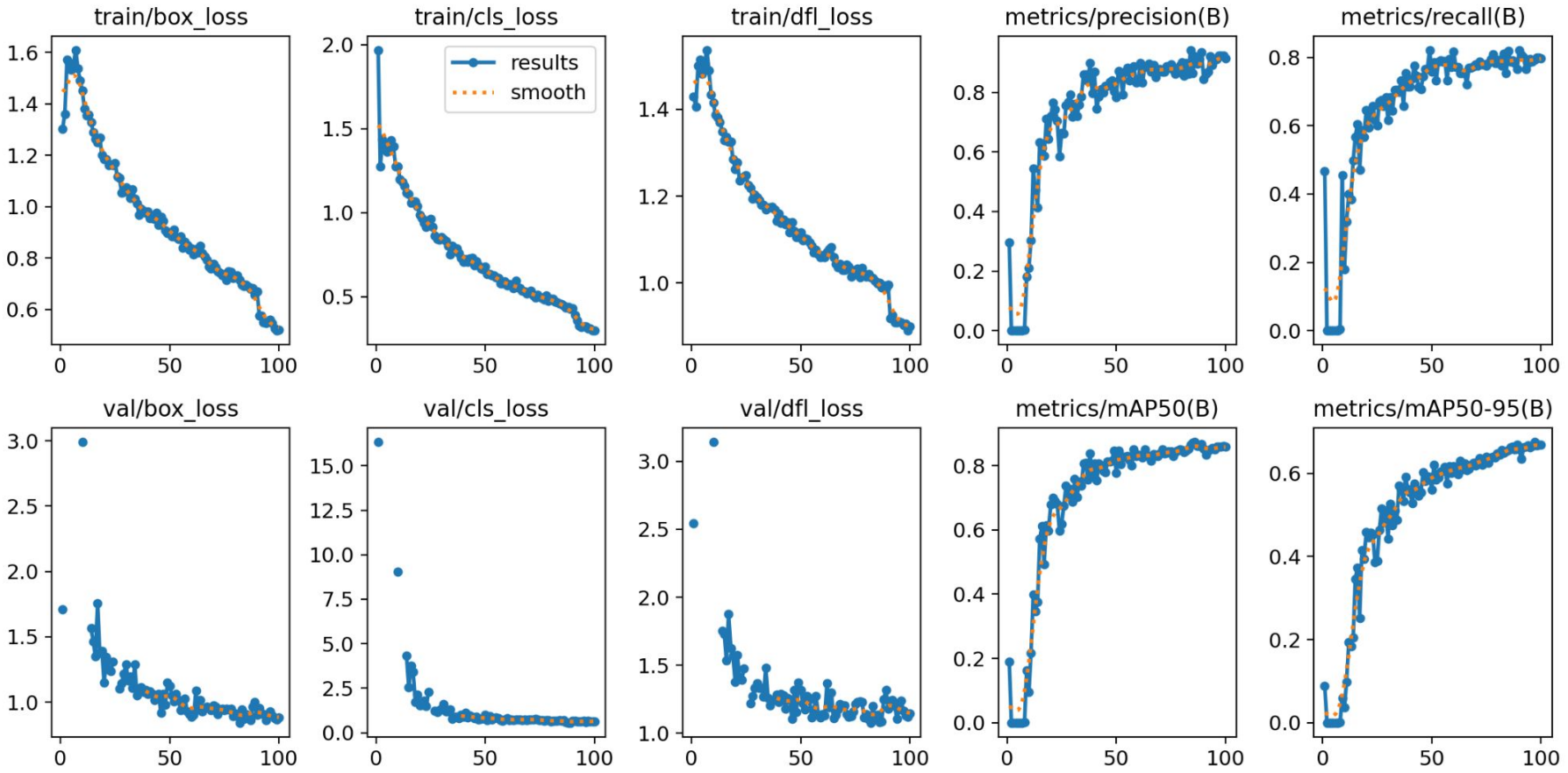


YOLOv11x Detection Models - Shoes Dataset Sample Detection

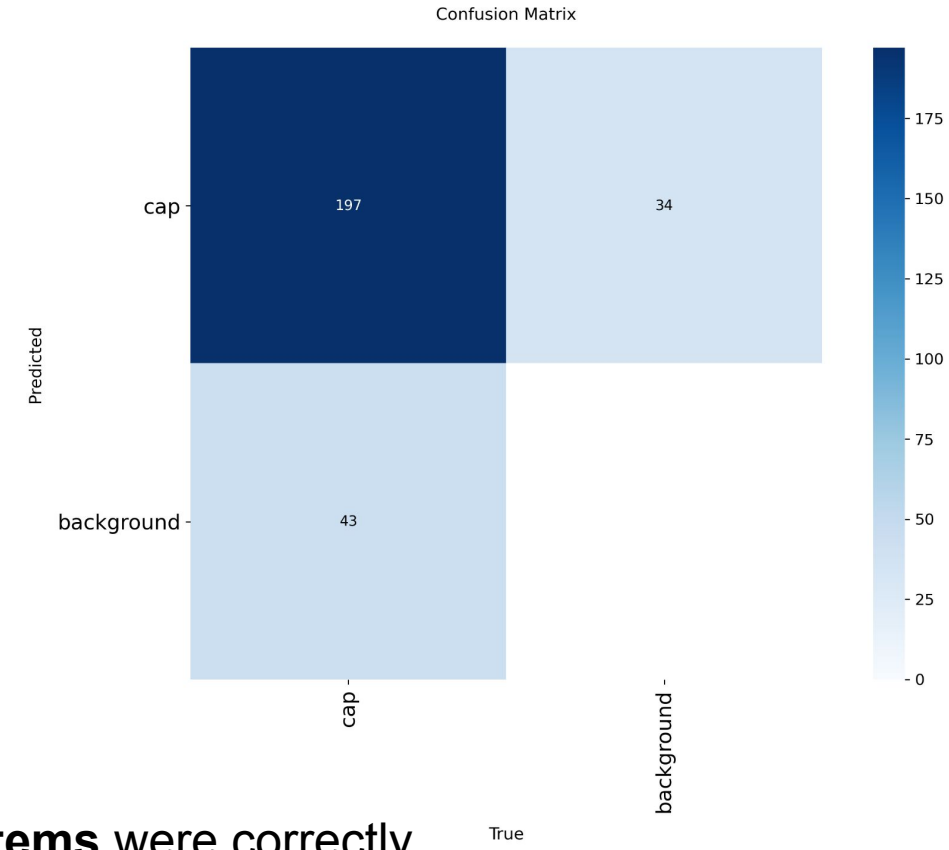


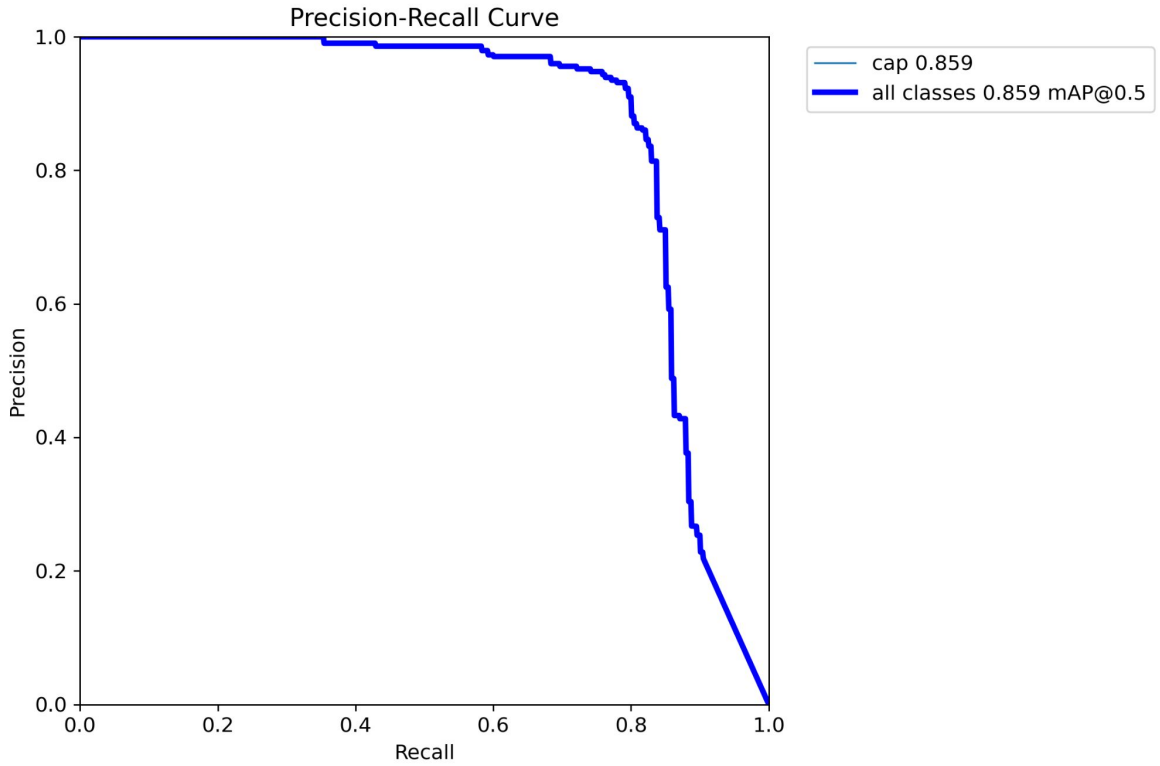
The model successfully detects various shoe types with **high confidence scores**!

YOLOv11x Detection Models - Headwear Dataset Training Curves



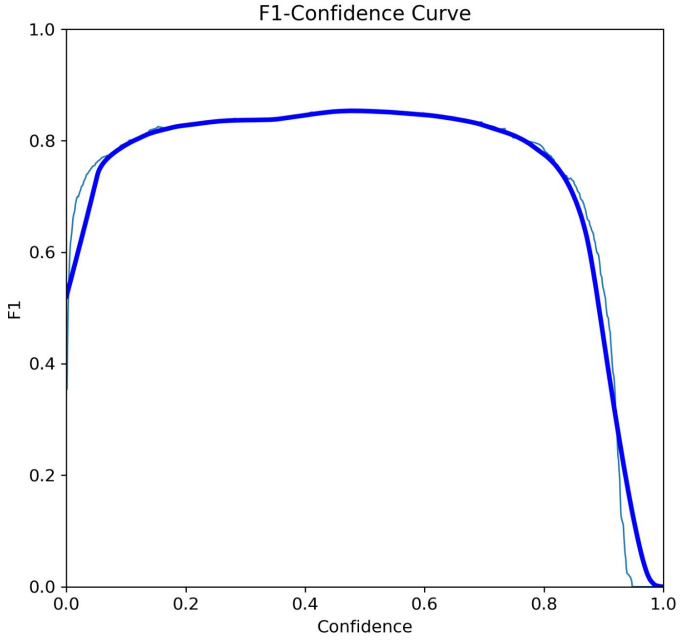
YOLOv11x Detection Models - Headwear Dataset Confusion Matrix & PR Curve

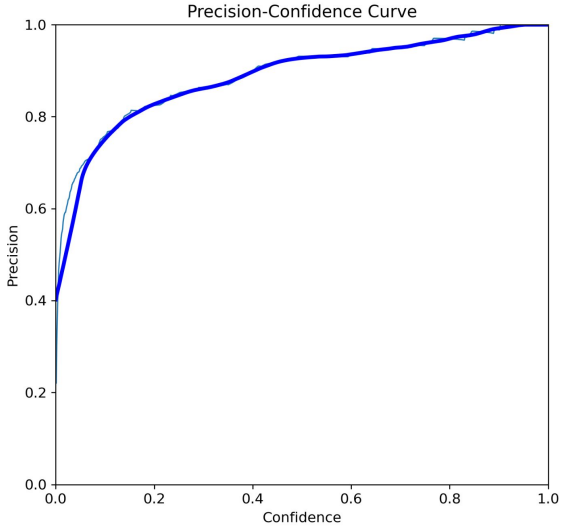


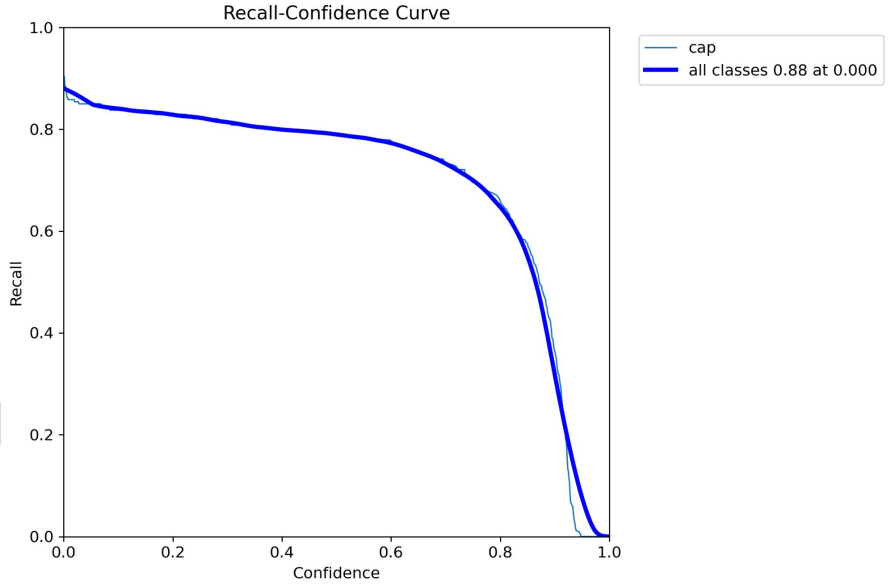


197 items were correctly classified as caps!

YOLOv11x Detection Models - Headwear Dataset Confidence-Based Metric Curves







YOLOv11x Detection Models - Headwear Dataset Sample Detection

The model successfully detects various headwear types with **high confidence scores!**

What We Have At The End with YOLOv11x?

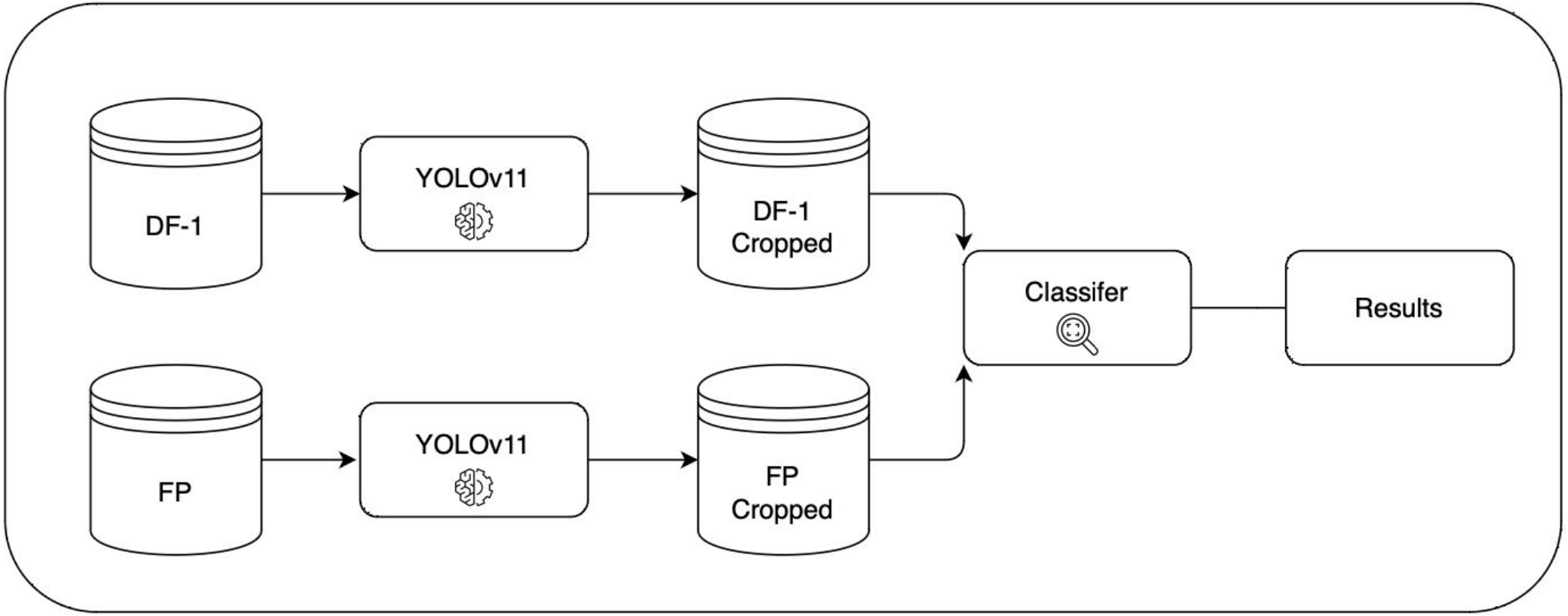
Thanks to these reliable detections:

- We can **crop fashion items** from full-outfit images with high precision.
- These cropped items are then used as inputs for our **semantic attribute classifier**, which attempts to identify features like gender, usage, fabric, and color.
- Furthermore, it sets the stage for **matching detected items** with visually or semantically similar products.

with YOLOv11x, we're not just detecting fashion items!!!

we're **preparing clean, object-level inputs** for the next stages of our pipeline! —> —>

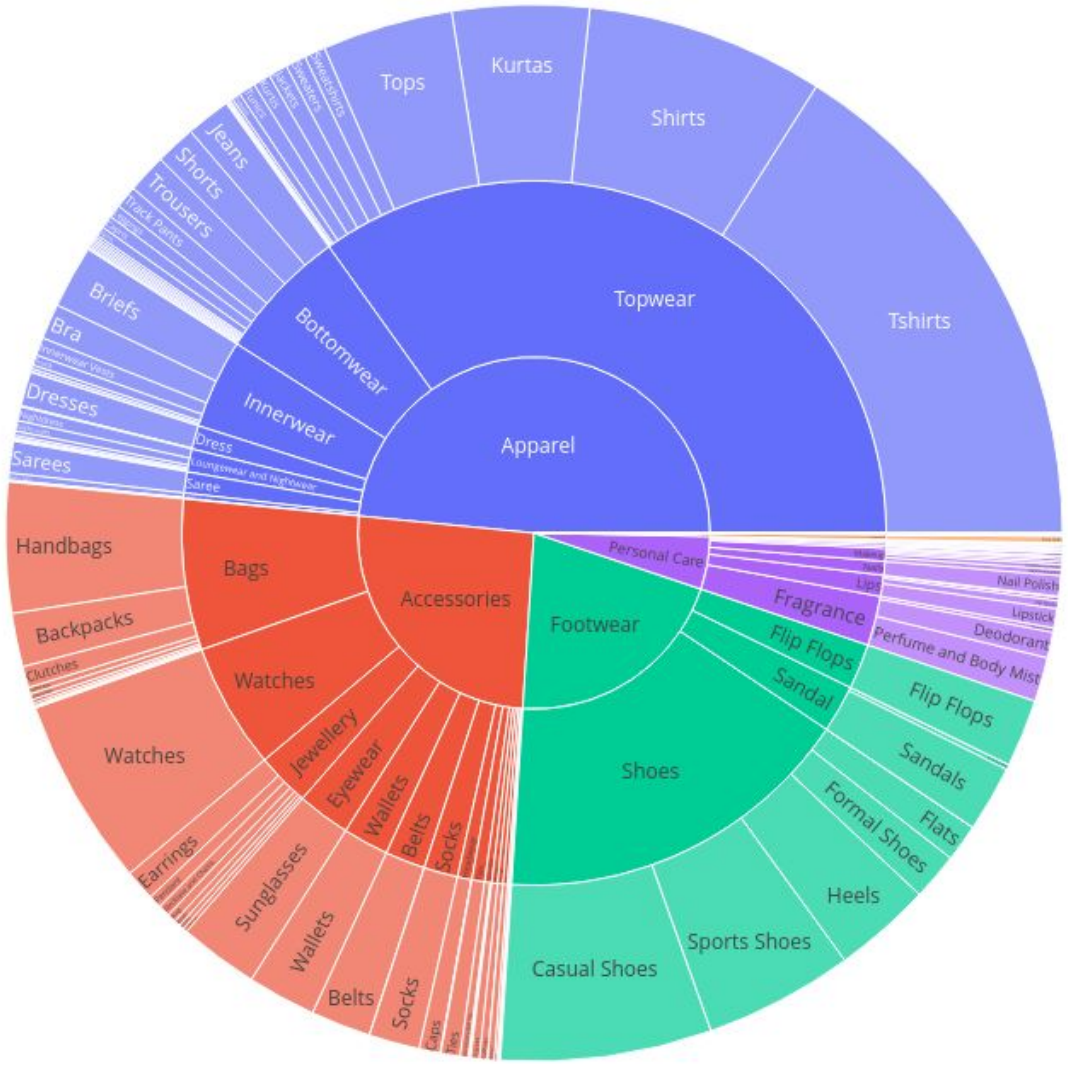
Dataset Preparation for Classification



Dataset Preparation for Classification

Fashion Products Images (44.4k images)

DeepFashion (300k items)



Fashion Product Images: data cleaning

Dropped classes: Year and Season

Fall

Summer

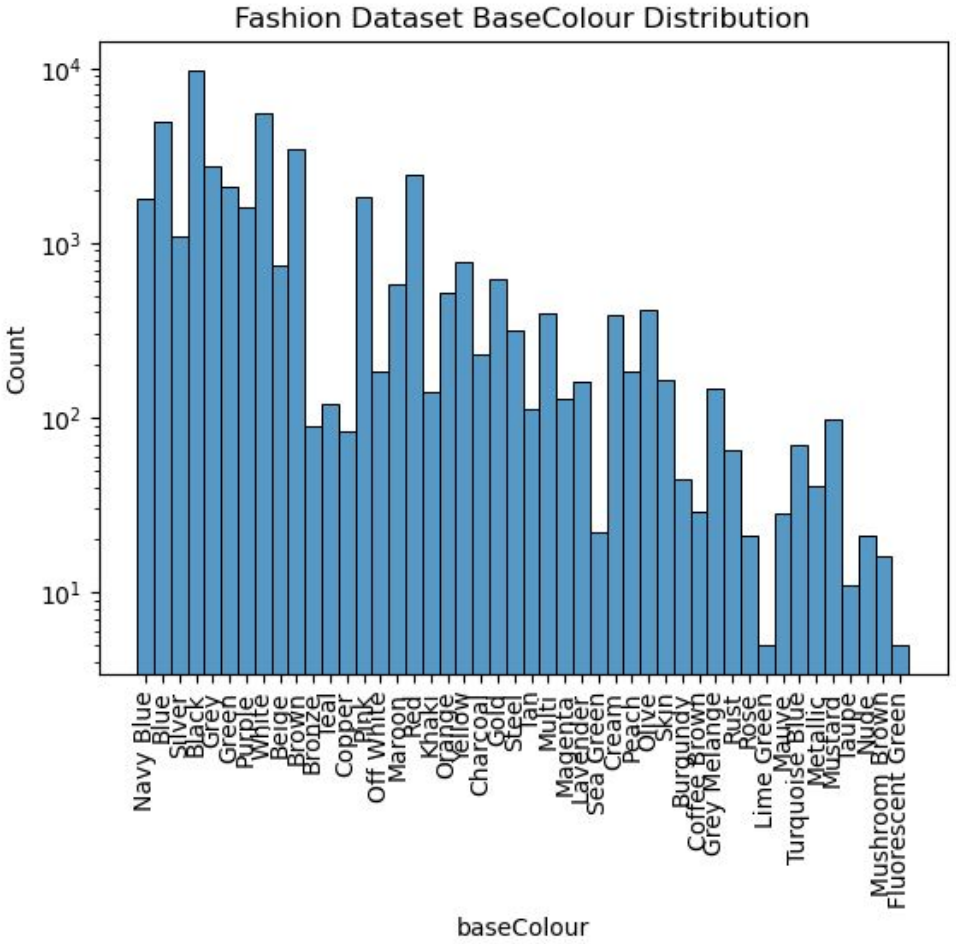
Dropped labels: Unisex, Kids, Girls dropped from Gender

ID: 2710 | subCategory: Dress

ID: 2699 | subCategory: Dress

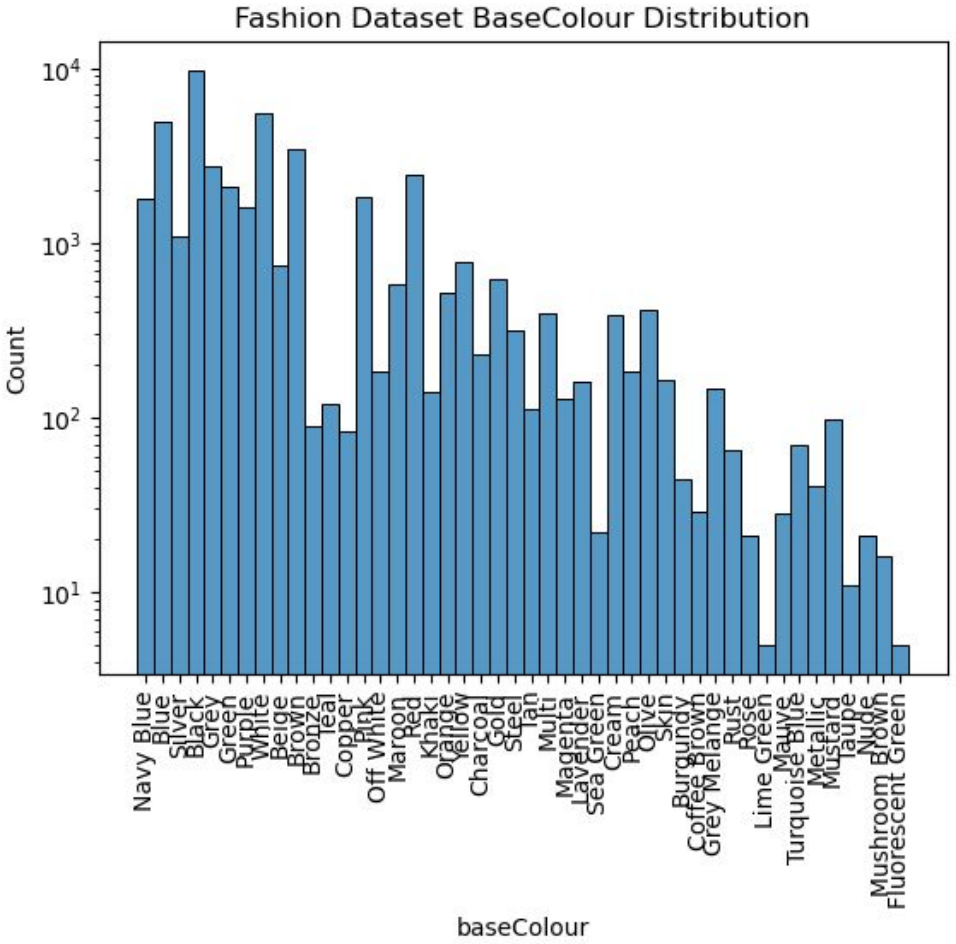
ID: 2697 | subCategory: Dress

Merged: Color

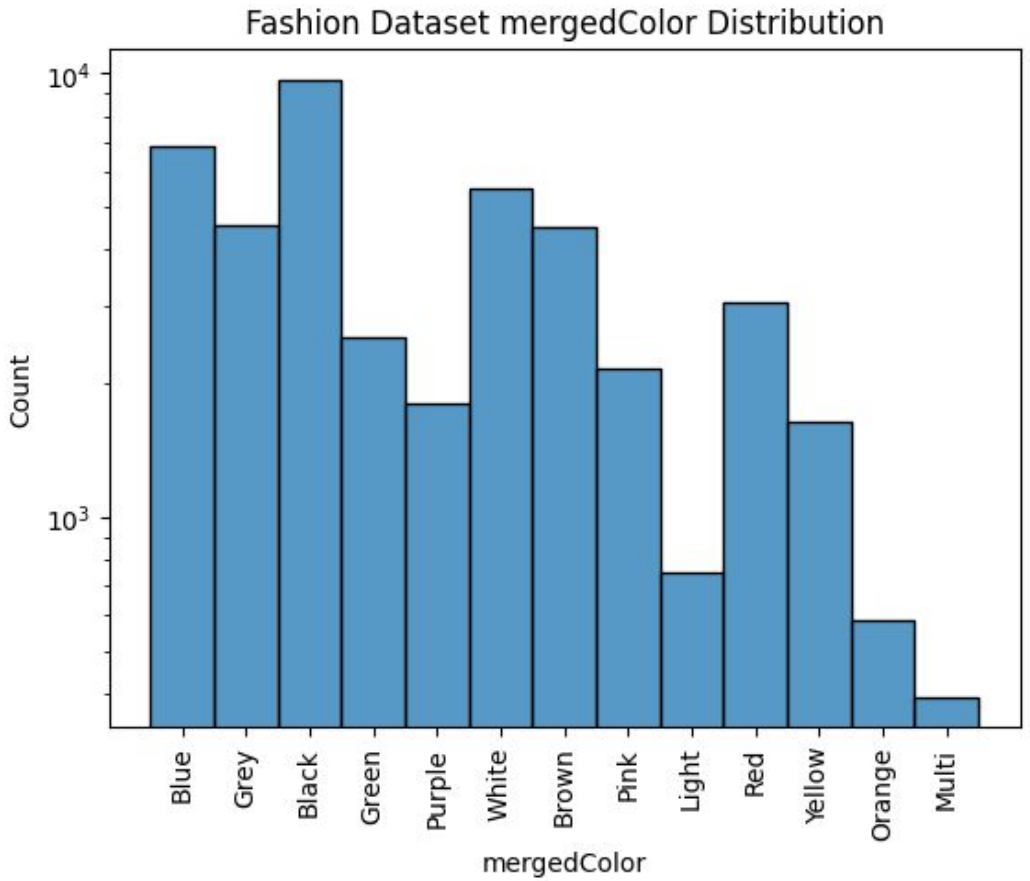


Blue	Navy Blue	Turquoise Blue	Teal	Blue				
Green	Lime Green	Fluorescent Green	Sea Green	Olive	Green			
Grey	Charcoal	Grey Melange	Steel	Metallic	Silver	Grey		
Light	Off White	Cream	Nude	Skin				
Brown	Beige	Tan	Taupe	Mushroom Brown	Coffee Brown	Bronze	Copper	Brown
Red	Maroon	Burgundy	Red					
Pink	Peach	Rose	Magenta	Pink				
Purple	Lavender	Mauve	Purple					
Yellow	Mustard	Gold	Khaki	Yellow				
Orange	Rust	Orange						
Black	Black							
White	White							
Multi	Multi							

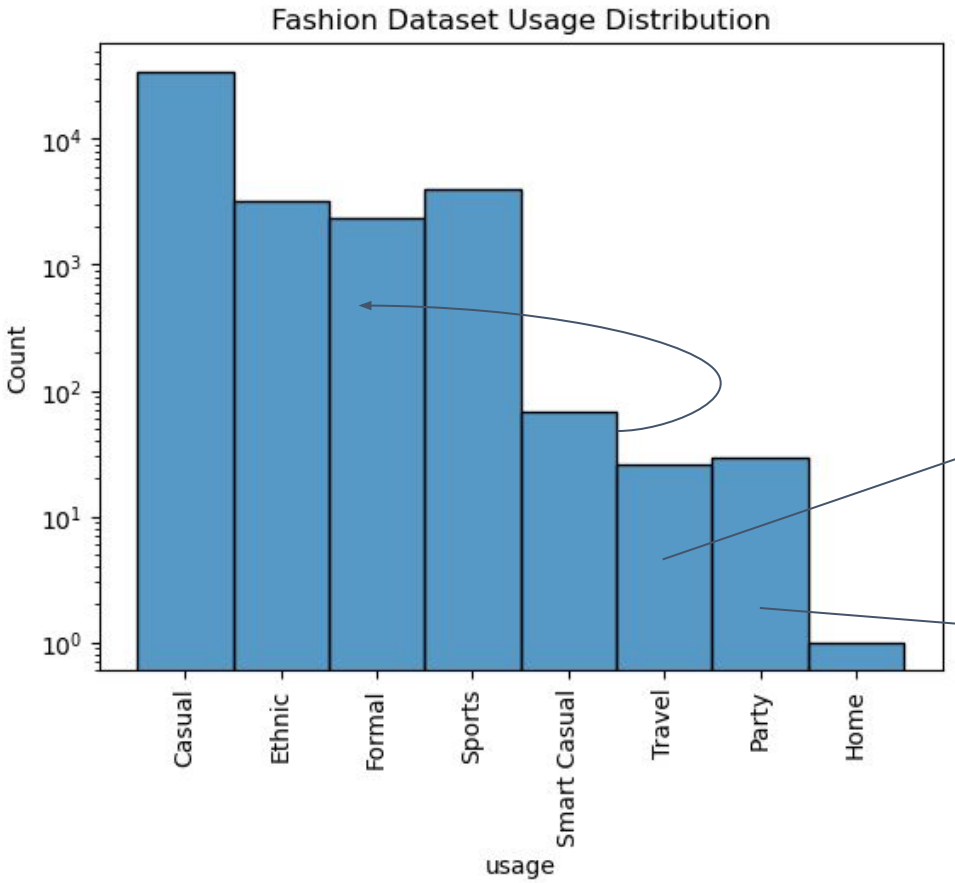
Merged: Color



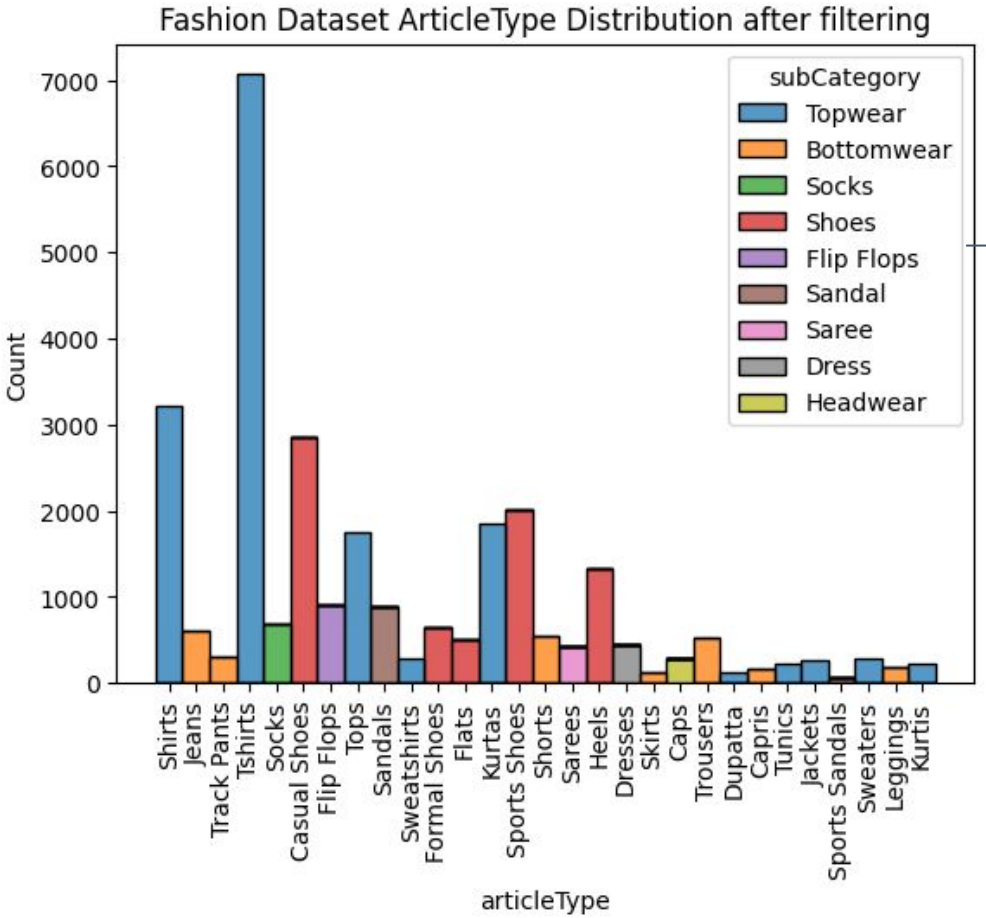
after merging equivalent colors



Cleaned: Usage



Matching articleType to the Image



Merged into:

Topwear
Bottomwear
Footwear
Dress
Headwear

Matching articleType to the Image

Merged subCategory	Detector applied
‘Headwear’	Trained on cap-dataset
‘Footwear’	Trained on shoes-dataset
‘Topwear’, ‘Bottomwear’, ‘Dress’	Trained on df2-dataset

Matching articleType to the Image

Cleaned out mismatched labels

Matching articleType to the Image

The curse of multiple detections

Image ID: 26114

Image ID: 29182

Image ID: 9952

Image ID: 57174

Matching articleType to the Image: making predicted label and subCategory match

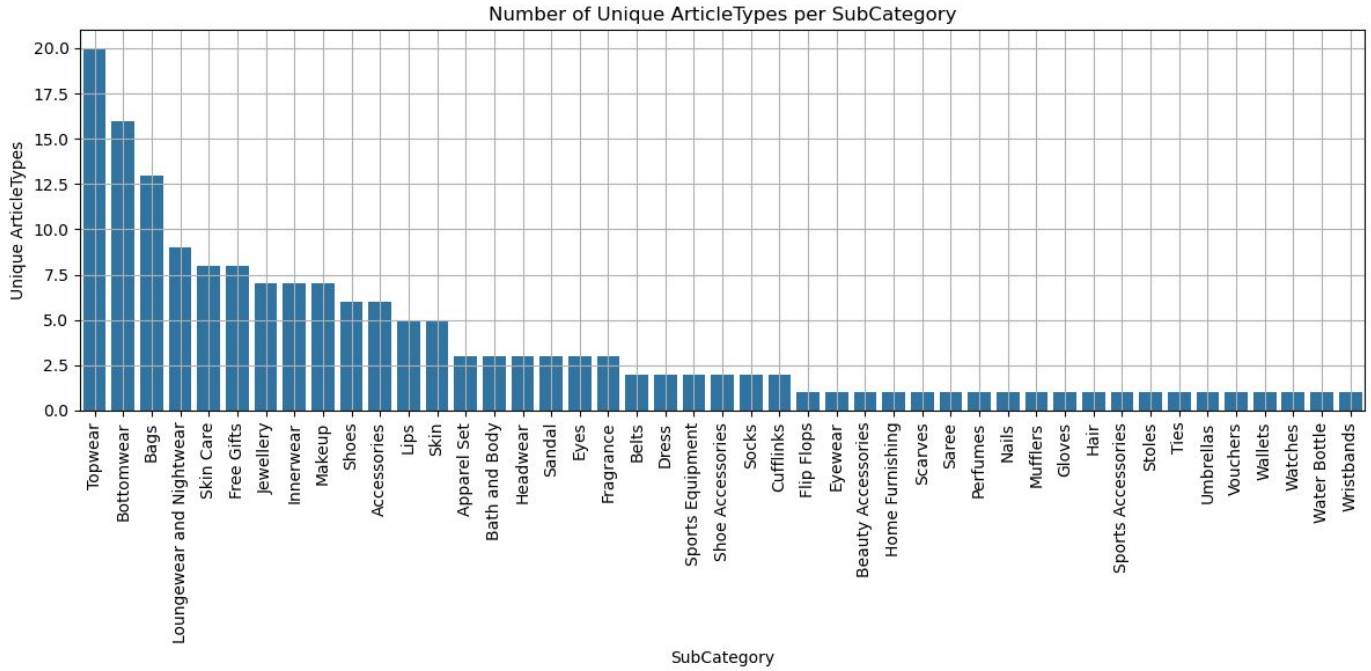
ID: 6863 | subCategory: Bottom

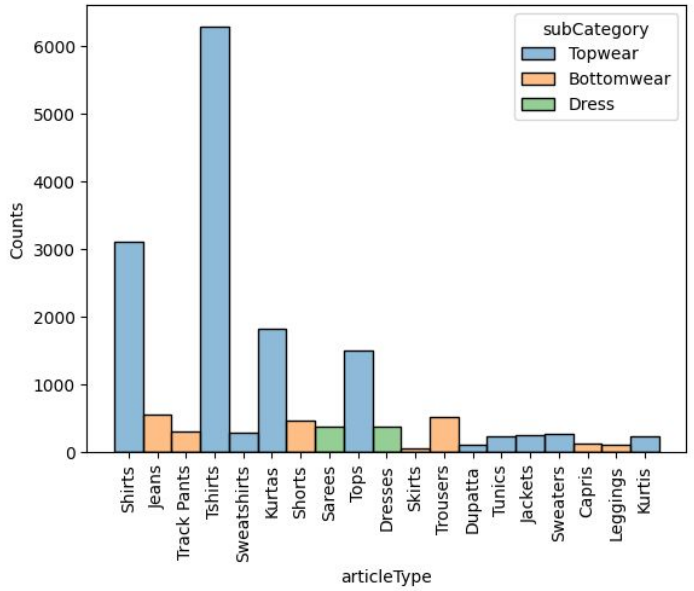
ID: 6863 | subCategory: Bottom

ID: 6982 | subCategory: Bottom

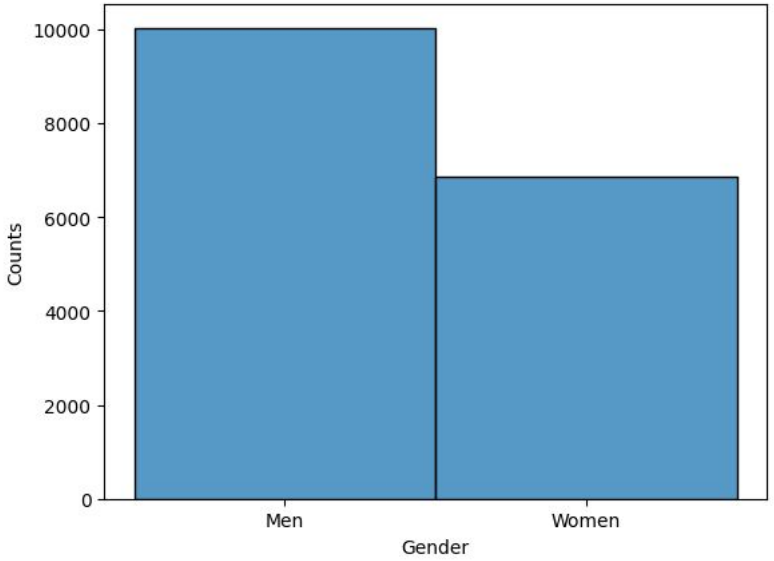
ID: 6982 | subCategory: Bottom

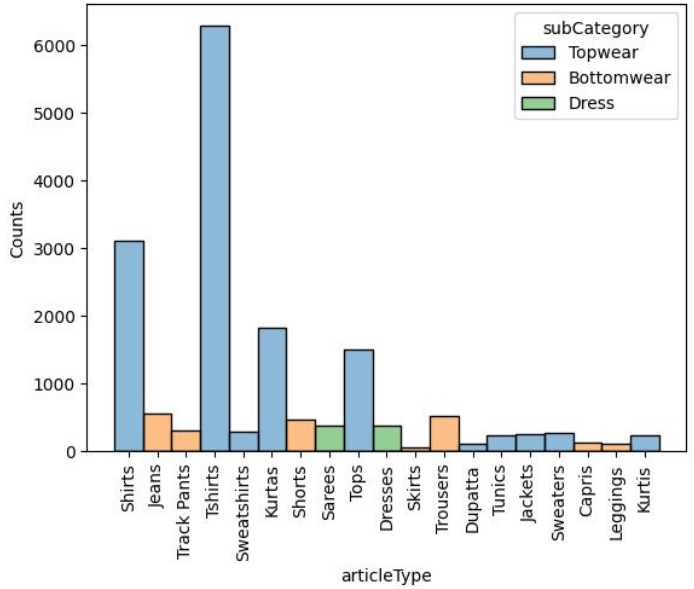
articleType: merged

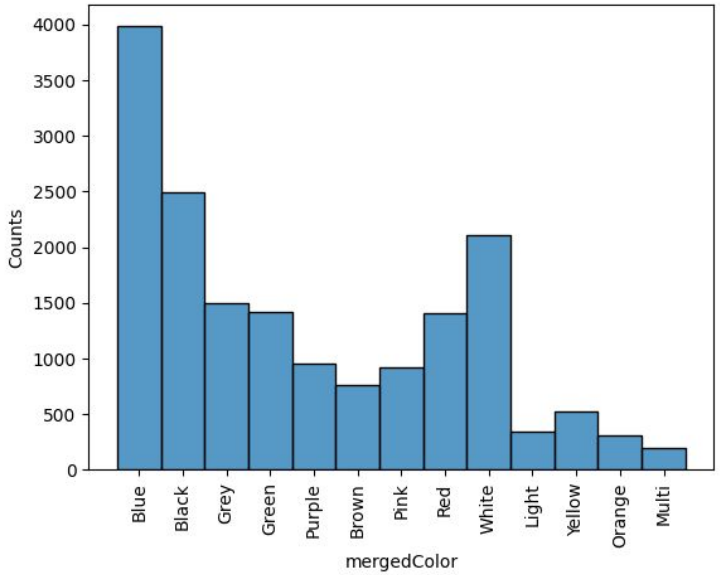


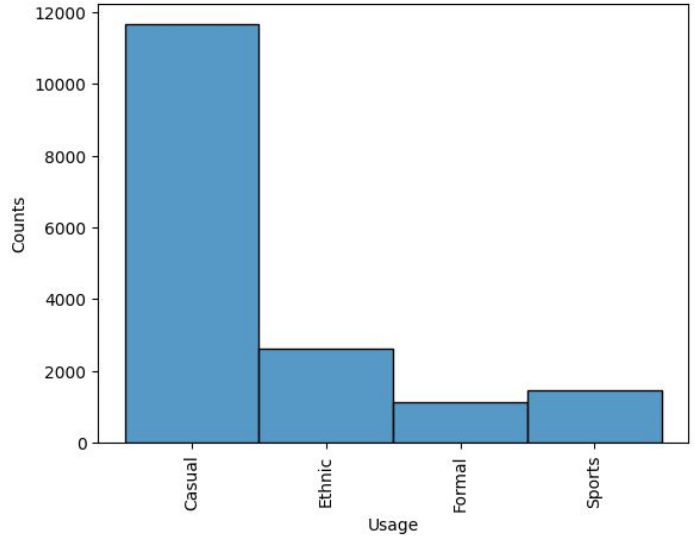


Statistics on the cleaned dataset (~18000) images







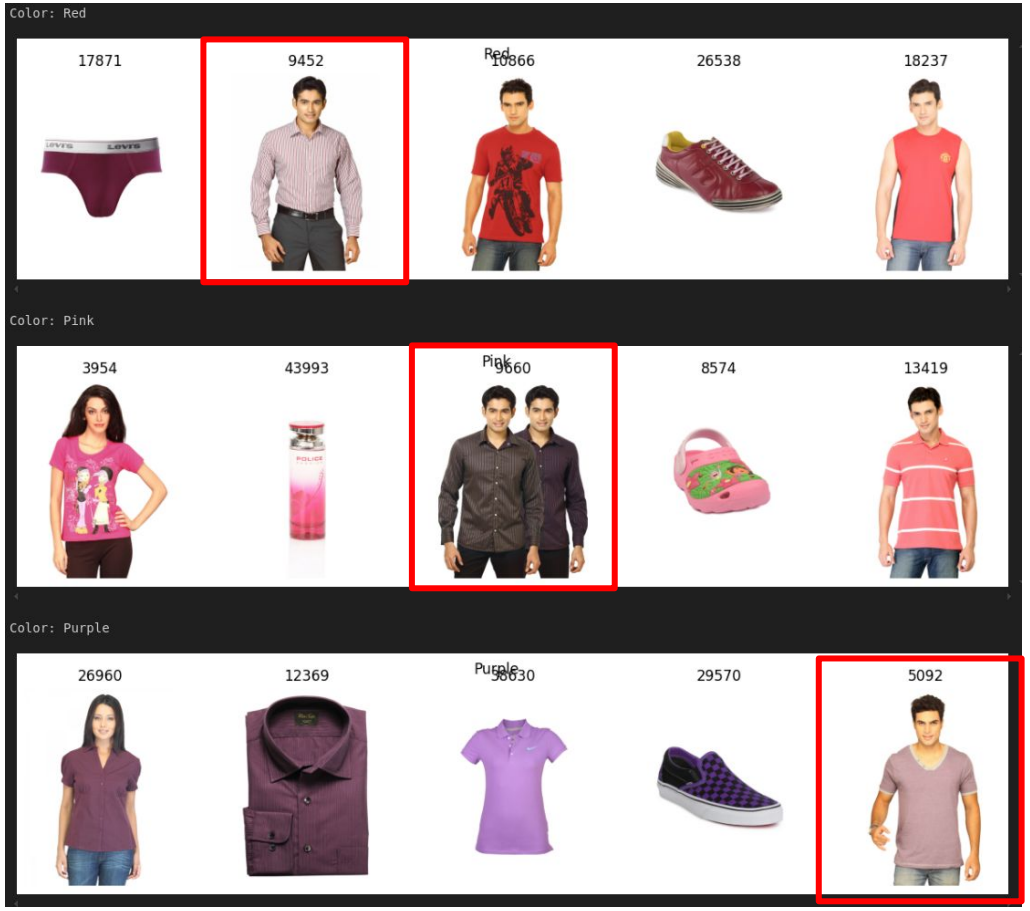


Weaknesses of FP Dataset (an example)

Red

Pink

Purple



Labels inconsistency and mismatches

DeepFashion dataset structure

naming convention:
{gender}-{clothing_item}-id_{id}-{n1}_{n2}_{pos}.jpg

DeepFashion dataset structure

Images with id=00000065, n1=02, n2=1, different types

type: Blouses_Shirts
pos: front

type: Pants
pos: front

- id: the model
- id + n1: outfit (of the model)
- id+n1+n2: model, outfit, position
- id+n1+n2+clothing_item: single item in the outfit

YOLO detector applied: same strategy to associate type to the correct bounding box

ID: 65, n1: 2, Type: Blouses_Shirts, Labels: 2

ID: 181, n1: 3, Type: Blouses_Shirts, Labels: 0

ID: 181, n1: 3, Type: Jackets_Coats, Labels: 0

ID: 181, n1: 3, Type: Jackets_Coats, Labels: 3

YOLO detector applied: same strategy to associate type to the correct bounding box

ID: 291, n1: 5, Type: Cardigans, Labels: 1

ID: 533, n1: 4, Type: Sweaters, Labels: 1

Mismatches:cleaned out

Image-specific features:dropped

sleeve lenght	long
lower_clothing_length	long
socks	NA
hat	NA
glasses	NA
neckwear	yes
wrist_weraing	no
ring	no
waist_accessories	yes
neckline	round
outer_clothing_a_cardigan	yes
upper_clothing_covering_navel	yes

The challenge of fabric and patterns

upper_fabric	lower_fabric	outer_fabric
cotton	denim	cotton

upper_pattern	lower_pattern	outer_pattern
pure_color	other	other

Image-related features: how to associate the label to the correct detected item?
a.k.a.: what is upper, lower and outer?

Priority types retrieval

Bottom-types: easy

shorts
skirts
pants
leggings
denim

Outer-types: guess

jackets_coats
cardigans
jackets_vests

Upper-types: guess

Dresses
Tees_Tanks
Sweaters
Suiting
Blouses_Shirts

⋮

The policy applied

- If, for an image, the match is clear (one outer_type and one upper_type), retrieving the fabric/pattern label is clear.
- Otherwise:
  - Check for missing labels
  - Same label (no confusion)
  - Otherwise:
    - Image (and detected items): DROPPED

Some examples

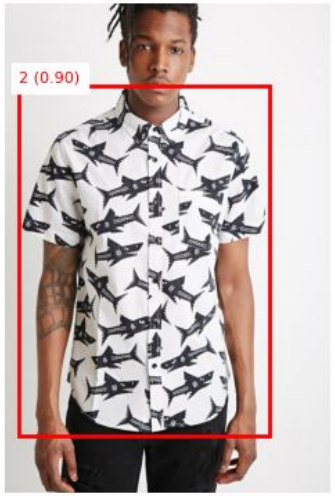
Cardigans
Fabric: knitted | Pattern: other

Tees_Tanks
Fabric: cotton | Pattern: pure_color

Tees_Tanks
Fabric: cotton | Pattern: pure_color

Shorts
Fabric: denim | Pattern: pure_color

Sweaters
Fabric: cotton | Pattern: graphic

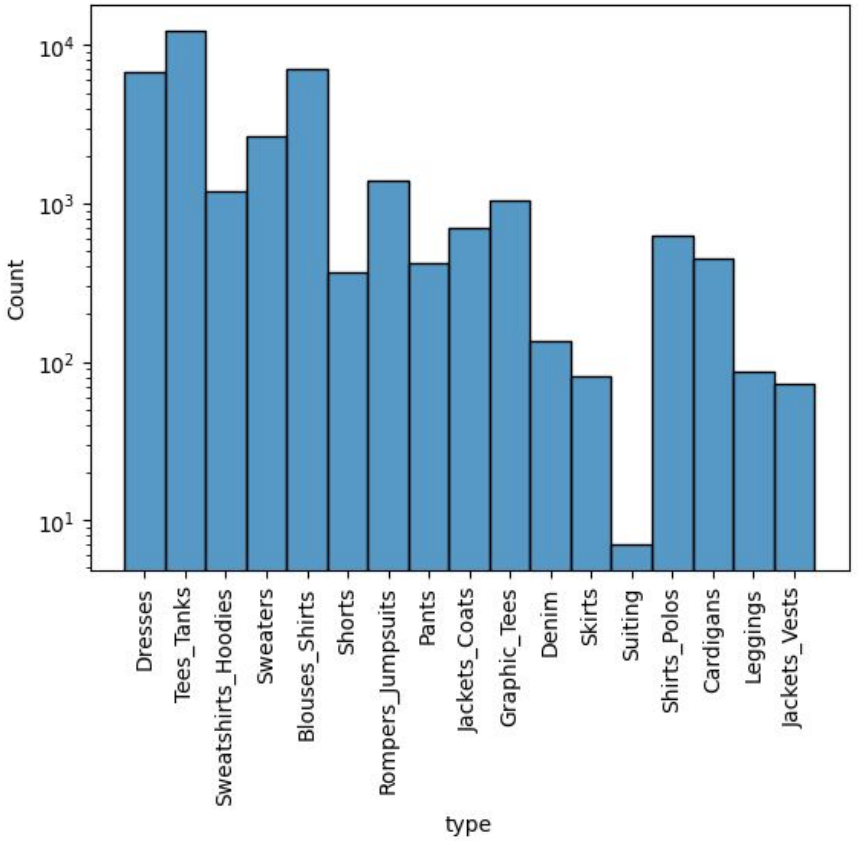


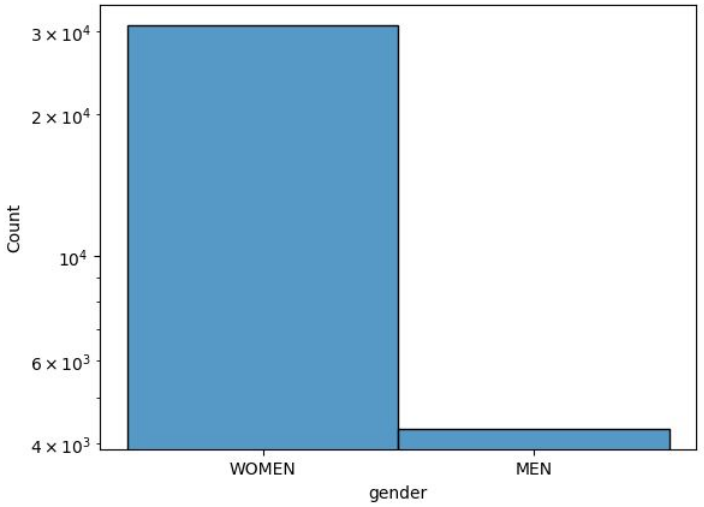
Tees_Tanks
Fabric: cotton | Pattern: pure_color

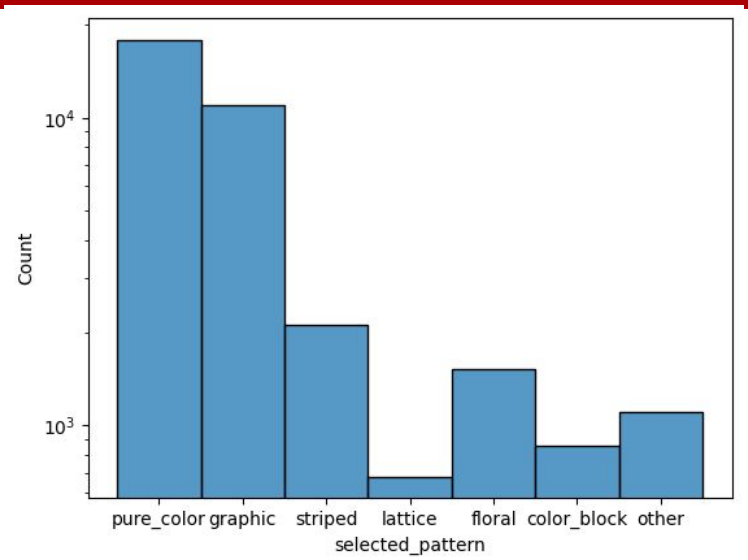
Rompers_Jumpsuits
Fabric: cotton | Pattern: pure_color

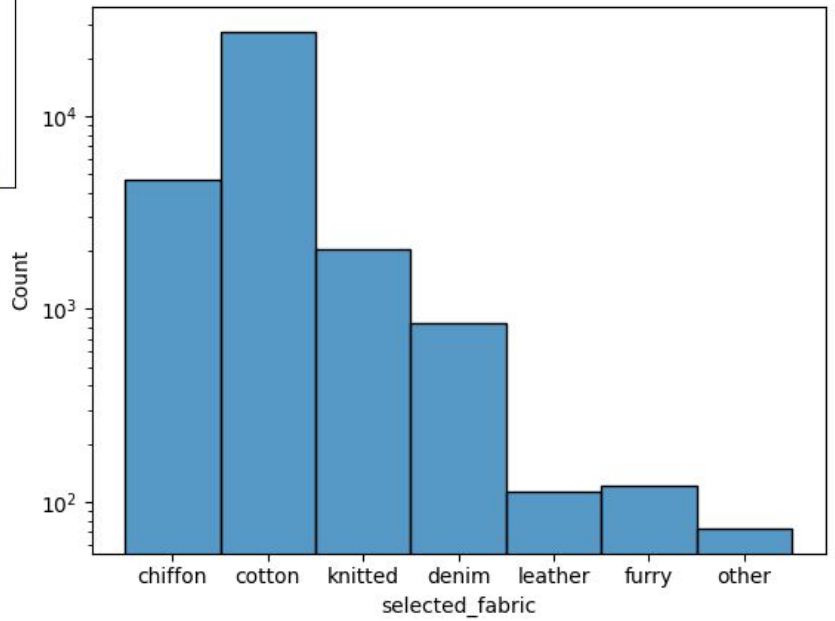
Tees_Tanks
Fabric: cotton | Pattern: striped

Final statistics on DeepFashion dataset







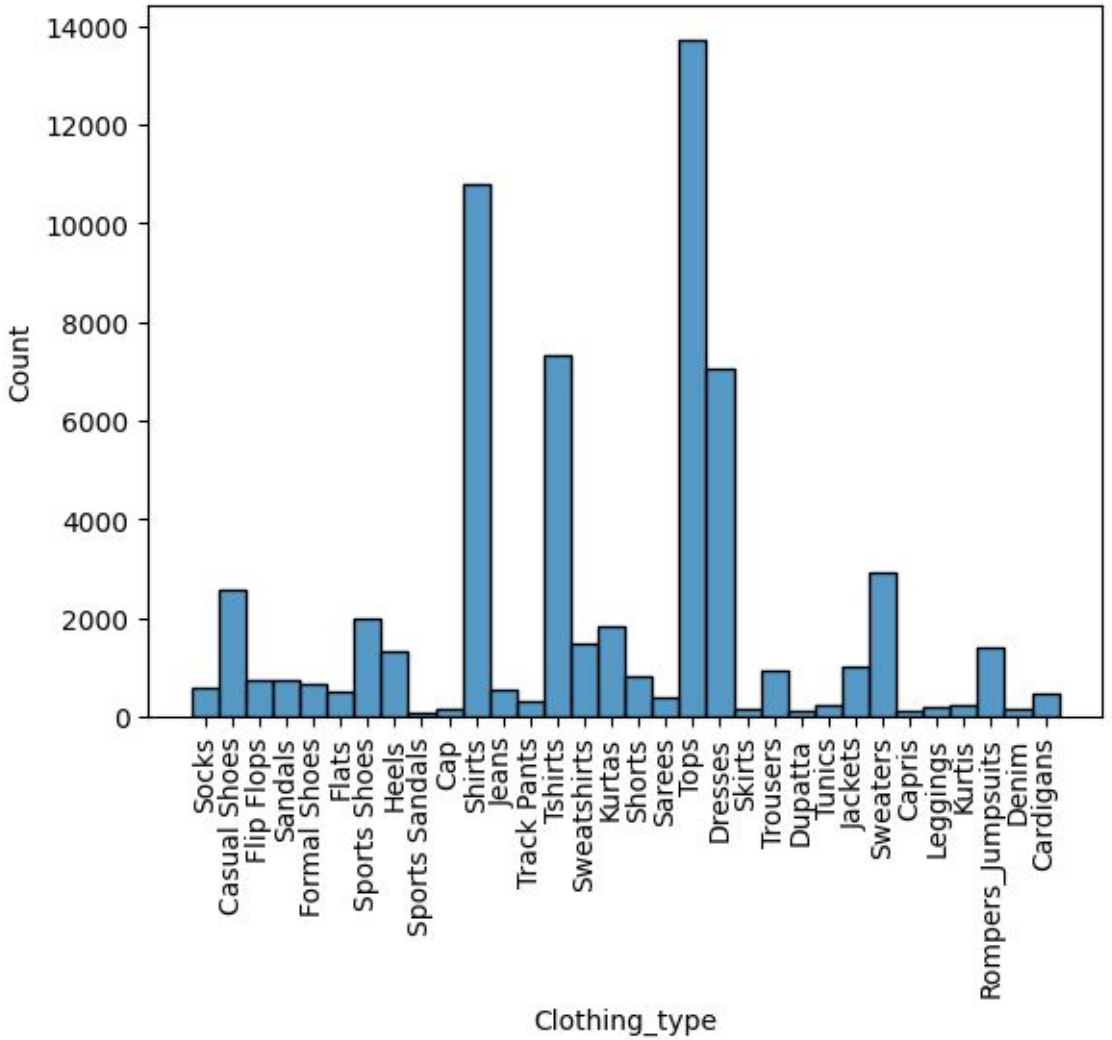


Finally

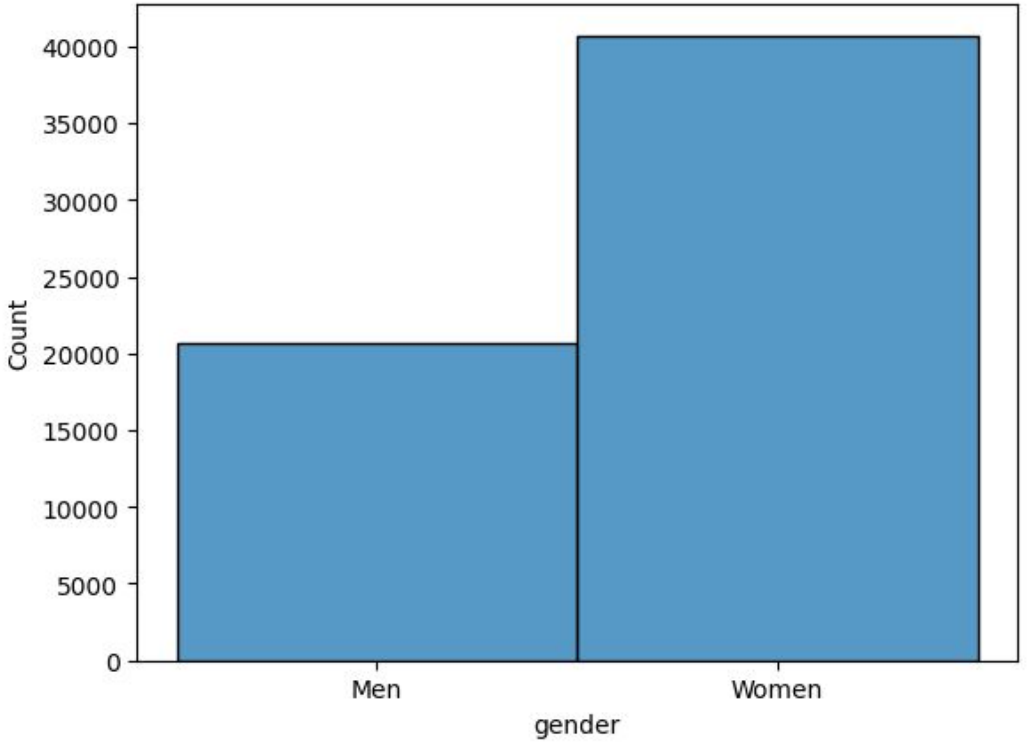
Feature	Source Dataset
Gender	FP & DF
Clothing Type	FP & DF
Usage	FP
Color	FP
Fabric	DF
Pattern	DF

Each cropped image were assigned with its corresponding labels.

Final statistics the merged Dataset

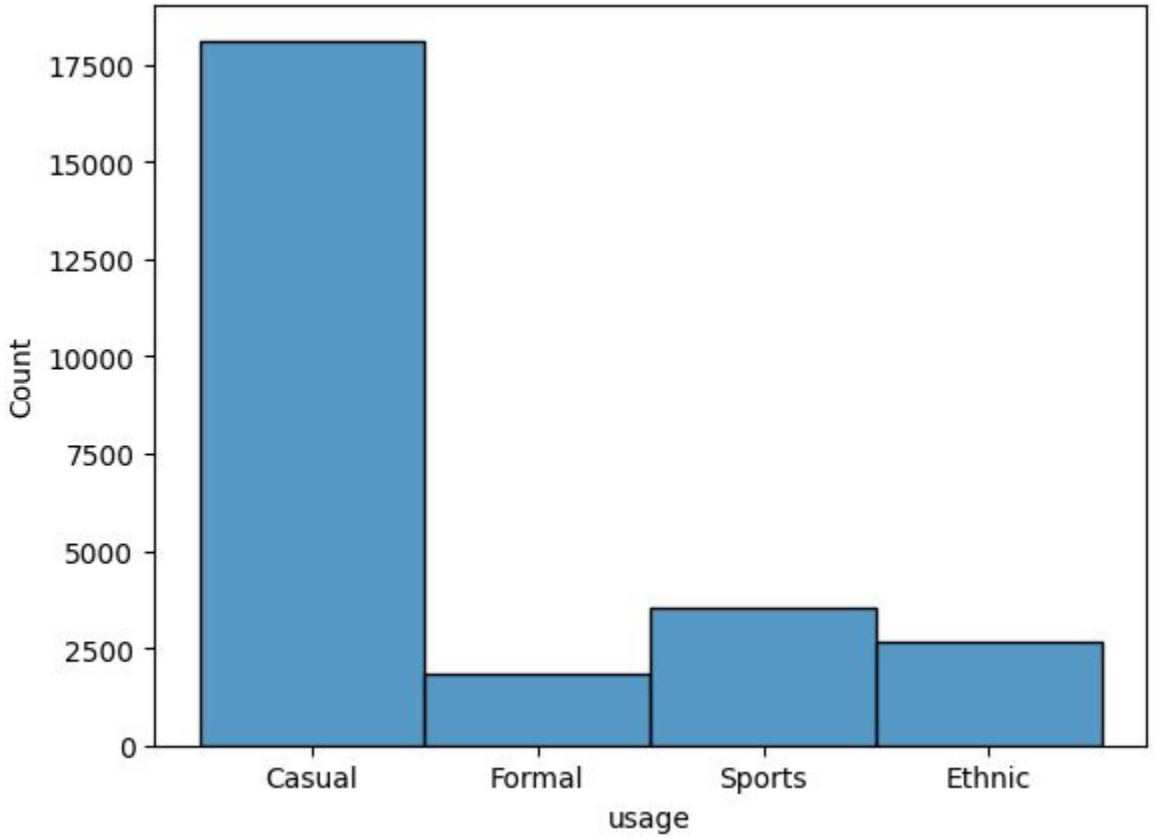


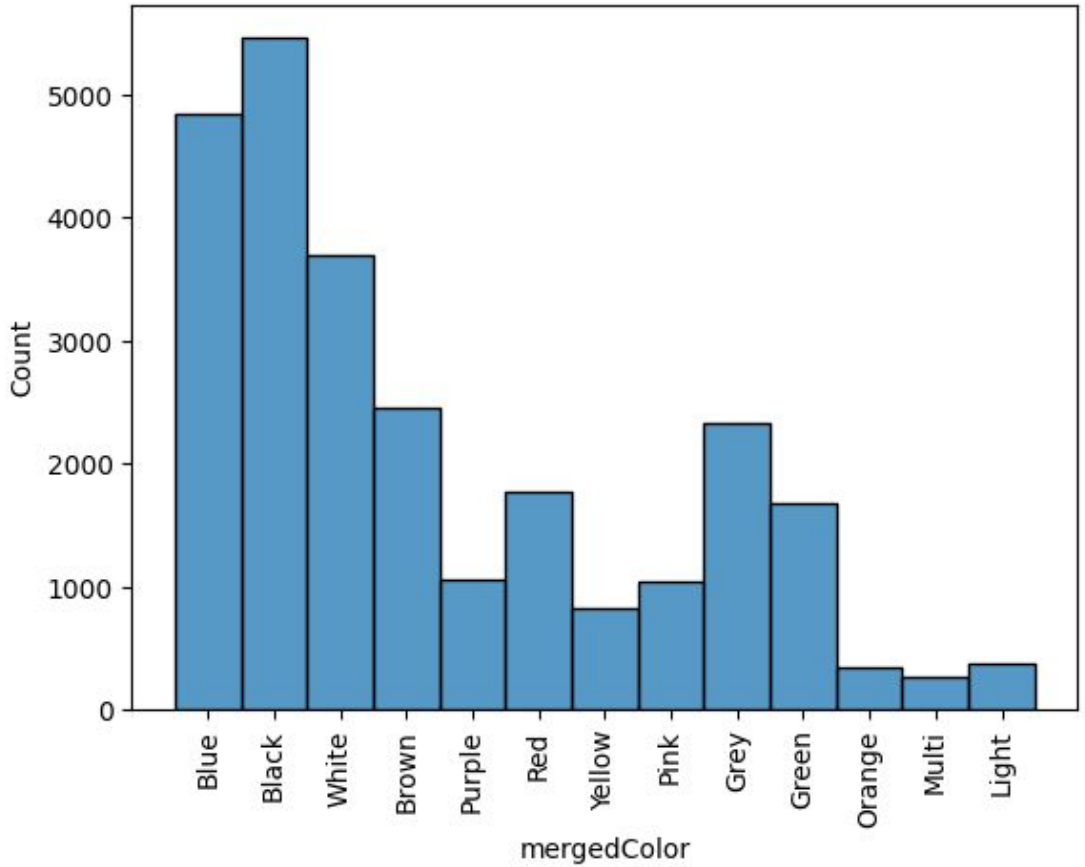
DF and FP (Clothing_type adapted)



Final statistics the merged Dataset

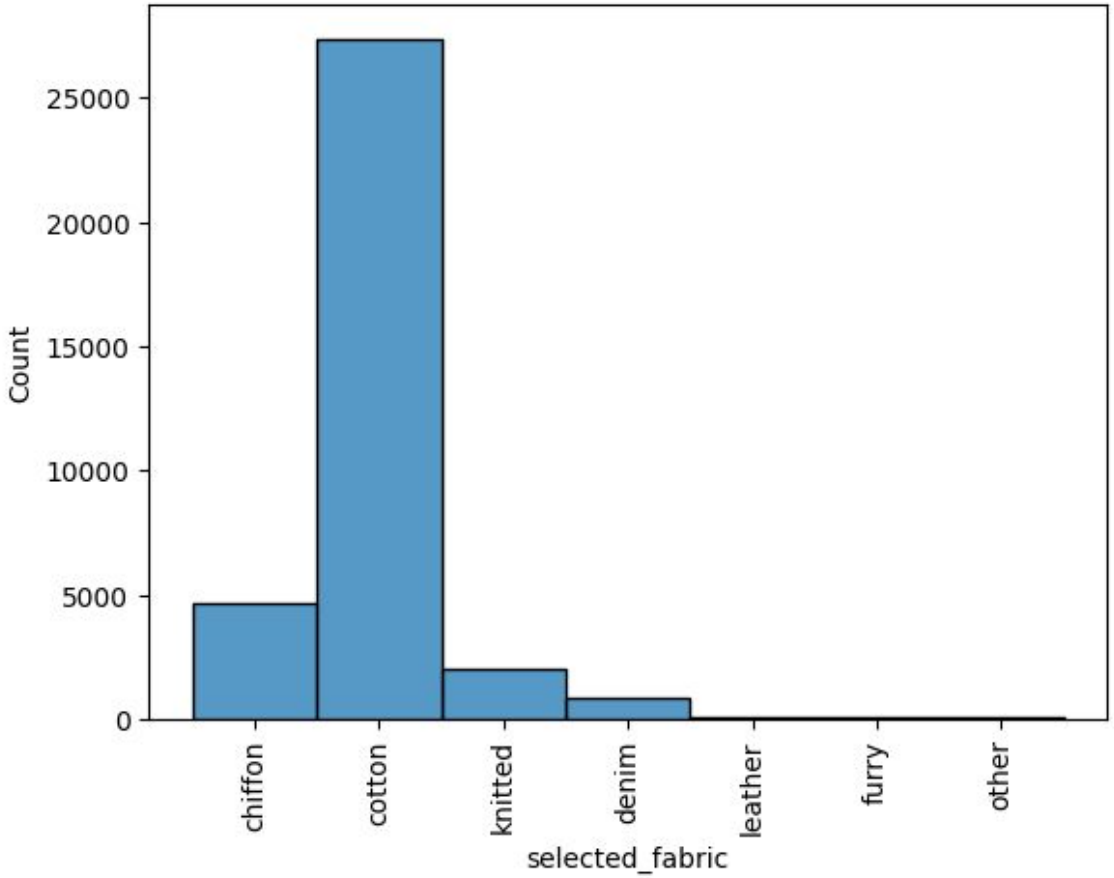
Only from Fashion Product

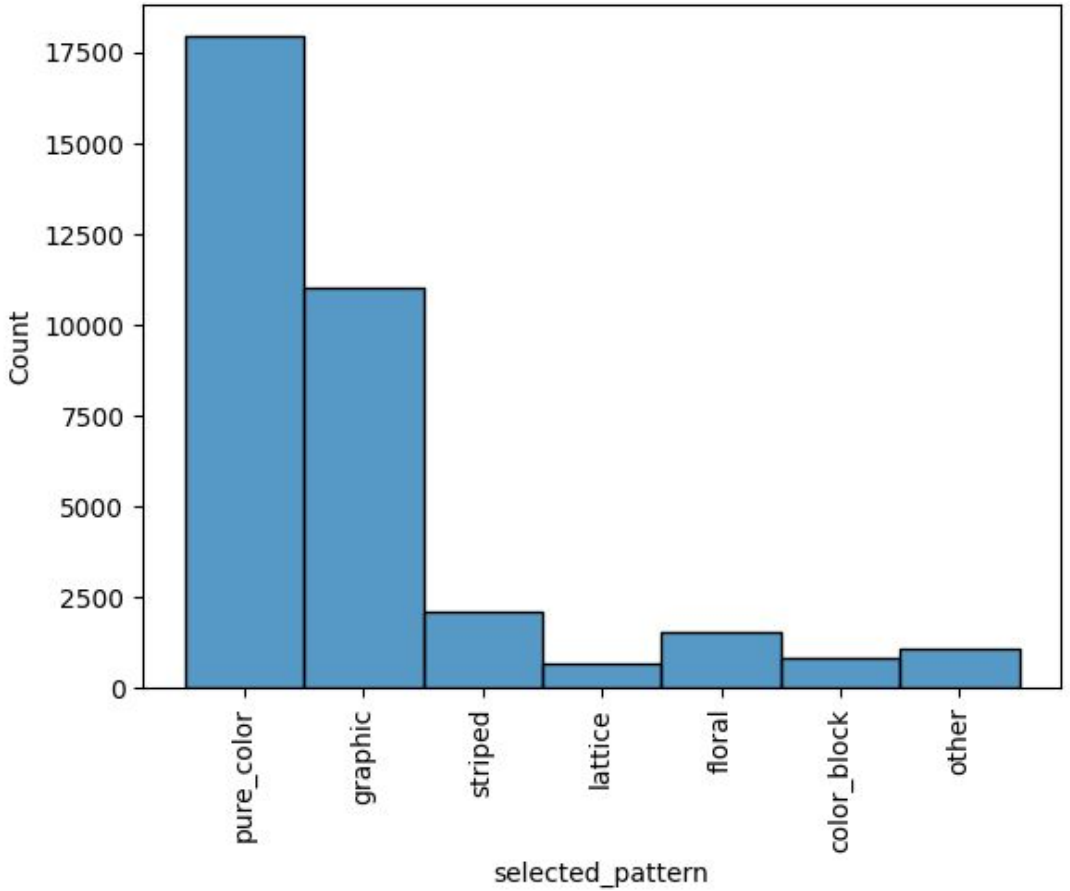




Final statistics the merged Dataset

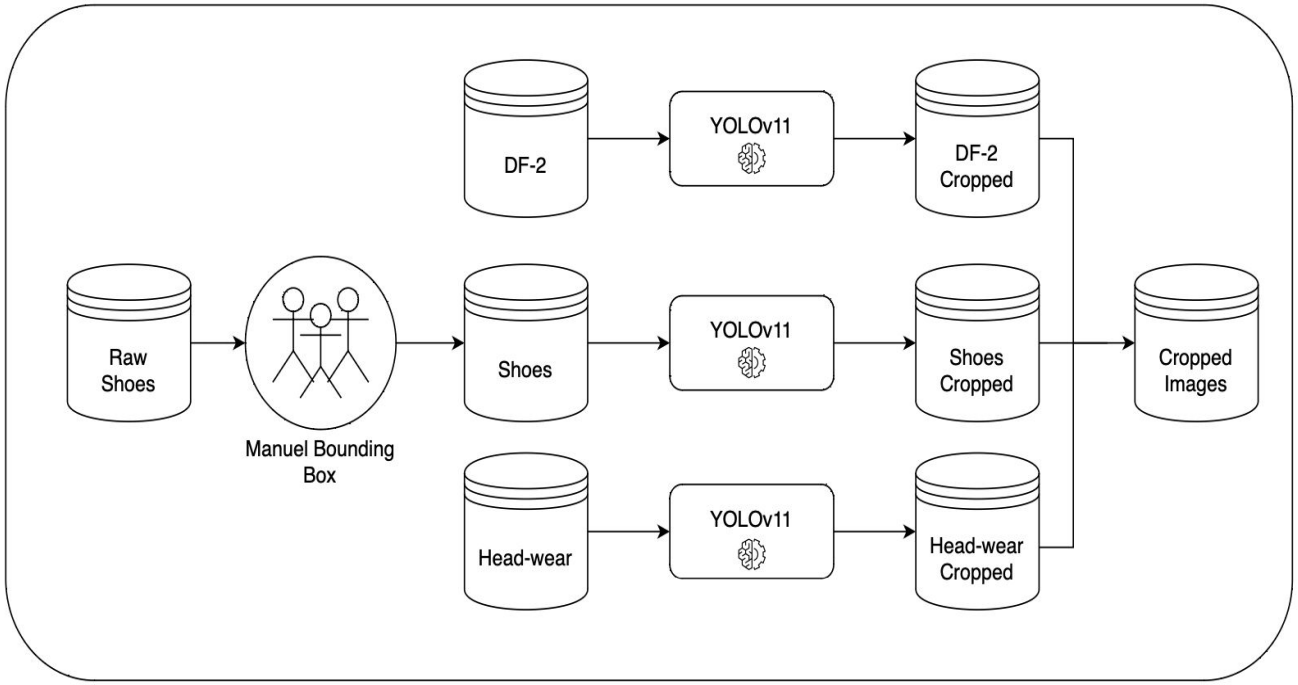
Only from DeepFashion

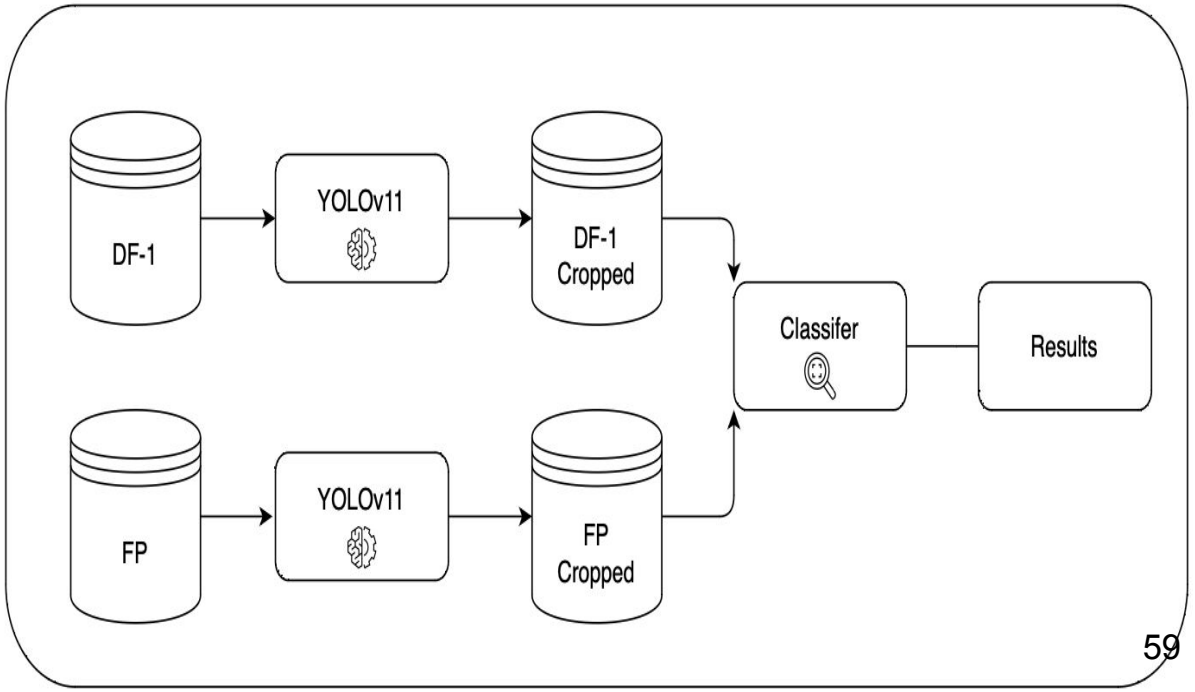




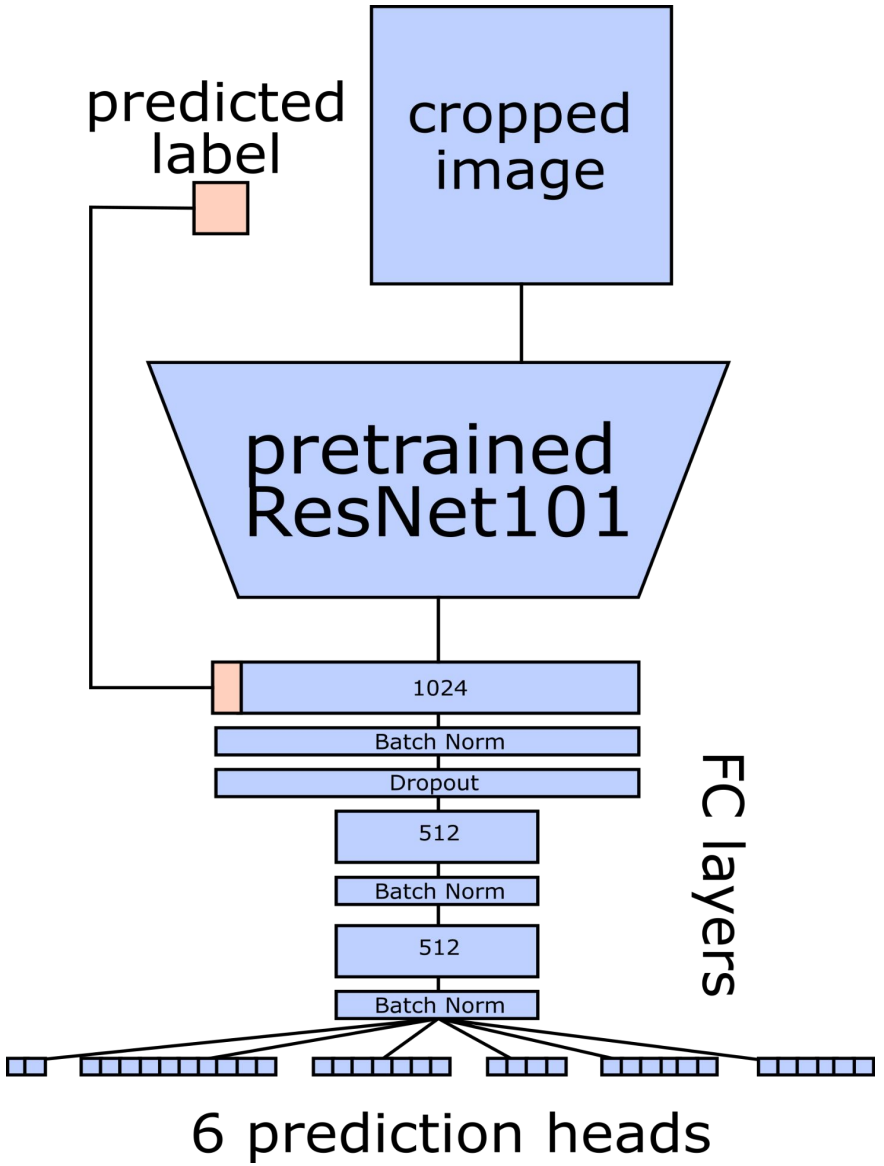
Important remark: Footwear and Headwear classes belong only to FP Dataset. Therefore, we don't have any information on the fabric and the pattern associated with these kind of items. It's all up to the generalizing power of the Classifier.

Multi-Head Classifier Model-A





Component	Description
Backbone	ResNet101 pretrained with self-supervised learning
Input	Cropped images + predicted label from detectors
FC Layers	[1024 + 1 → 512 → 512] with BatchNorm, LeakyReLU, Dropout, where +1 is the predicted label value from the applied YOLO detectors
Task Heads	6 linear heads for: gender (2), type (32), usage (4), color (13), fabric (7), pattern (7)
Loss	Focal Loss + Masked Cross Entropy with class weights
Augmentation	Blur, brightness/contrast shift, JPEG compression
Training Strategy	Interleaved learning based on dataset origin (FP or DF)
Optimizer	Adam: LR = 1e−4 (heads/FC), 5e−5 (backbone)
Epochs	20
Logging	W&B + per-task confusion matrices



Data Augmentation

Technique	Probability	Parameter Range	Description
Gaussian Blur	30%	Radius: 0.5 – 1.5	Applies a mild blur to simulate focus variation or motion blur.
Brightness Adjustment	10%	Factor: 0.8 – 1.2	Randomly increases or decreases image brightness.
Contrast Adjustment	10%	Factor: 0.8 – 1.2	Randomly adjusts the contrast of the image.
JPEG Compression	30%	Quality: 40 – 70	Simulates lossy compression artifacts by saving and reloading the image.
Resize	100%	Output Size: (256, 256)	Resizes the final output to the specified dimensions for model compatibility.

Multi-Head Classifier (Model-A)

The Final model

1) ResNet101 backbone, pre-trained with self-supervised learning

2) A unique feature of our approach is that we concatenate the predicted clothing type label (obtained from the detector) as an additional input alongside image features. This adds semantic info and enforces dependence on detection model.

3) The model includes a fully connected multi-layer block followed by independent heads, one for each semantic label (e.g., gender, fabric, usage)

Multi-Head Classifier (Model-A)

The Final model

4) For each item, only the relevant heads are updated based on the source dataset (DF1 or FP).

5) Interleaved learning applied on Data Loaders to prevent forgetting of information learned through the weights.

6) To handle strong class imbalance, we combine masked cross-entropy and Focal Loss, and apply class-specific weighting.

What is focal loss?

Focal Loss is a loss function designed to address class imbalance during training especially in classification problems where one class significantly outnumbers others.

Challenges in Multi-Head Classifier (Model-A)

- Validation accuracy for the Clothing Type class

  (We enriched the input by combining the cropped item image with its predicted category label from the YOLO detector.)
- This injects prior knowledge derived from outfit detection phase
- Classes (Color, Fabric, and Pattern)
- Pre-trained backbones such as ResNet18 and ConvNeXt, struggled to extract sufficiently rich representations for these appearance-centric attributes.
- To overcome this limitation, we trained a ResNet101 encoder from scratch in a self-supervised manner.

Encoder

- Uses ResNet101
- SimCLR (Simple Contrastive Learning of Representations)

Component	Value
Backbone	ResNet101 (ImageNet, headless)
Proj. Head	MLP: 2048 → 2048 → 512
Loss	NT-Xent (contrastive, temp. scaled)
Batch Size	32
Image Size	256×256
Epochs	30
Augmentations	Flip, rotation, masking
Color Aug.	**Disabled** (to retain color cues)
Optimizer	SGD (lr=2e-2, mom=0.9, wd=5e-4)
Scheduler	Cosine annealing
Validation	KNN on frozen embeddings
Framework	PyTorch Lightning + W&B

SimCLR Pretraining

1) SimCLR (Simple Contrastive Learning of Representations)
2) It's a self-supervised learning method
3) It can learn meaningful image features without needing labels.

SimCLR Pretraining How is applied?

1) For each image, two different augmentations are created (random horizontal flip and rotation).

2) These are treated as a positive pair different views of the same image.

3) All other images in the batch are negative pairs.

4) The model learns to: Pull positive pairs closer in feature space. Push negative pairs apart.

SimCLR Pretraining Method

5) This is done using a contrastive loss called NT-Xent (Normalized Temperature-scaled Cross Entropy).

6) To ensure the final representations preserved color fidelity, augmentations involving color jitter and grayscale conversion were explicitly disabled, unlike standard Sim-CLR pipelines

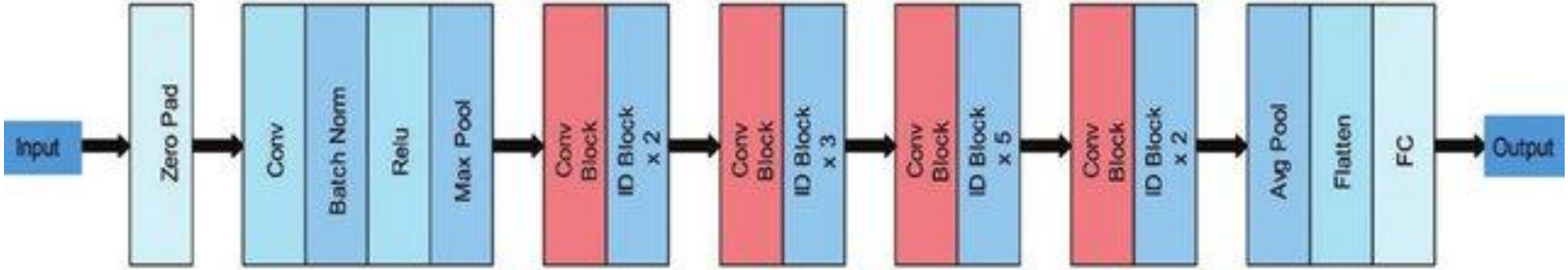
7) Once trained, the projection head is discarded, and the ResNet backbone is used to generate embeddings.

SimCLR Pretraining Method

8) These were evaluated using K-Nearest Neighbors in the latent space, confirming that visually similar garments clustered closely, even without labels. Weights are saved for classification.

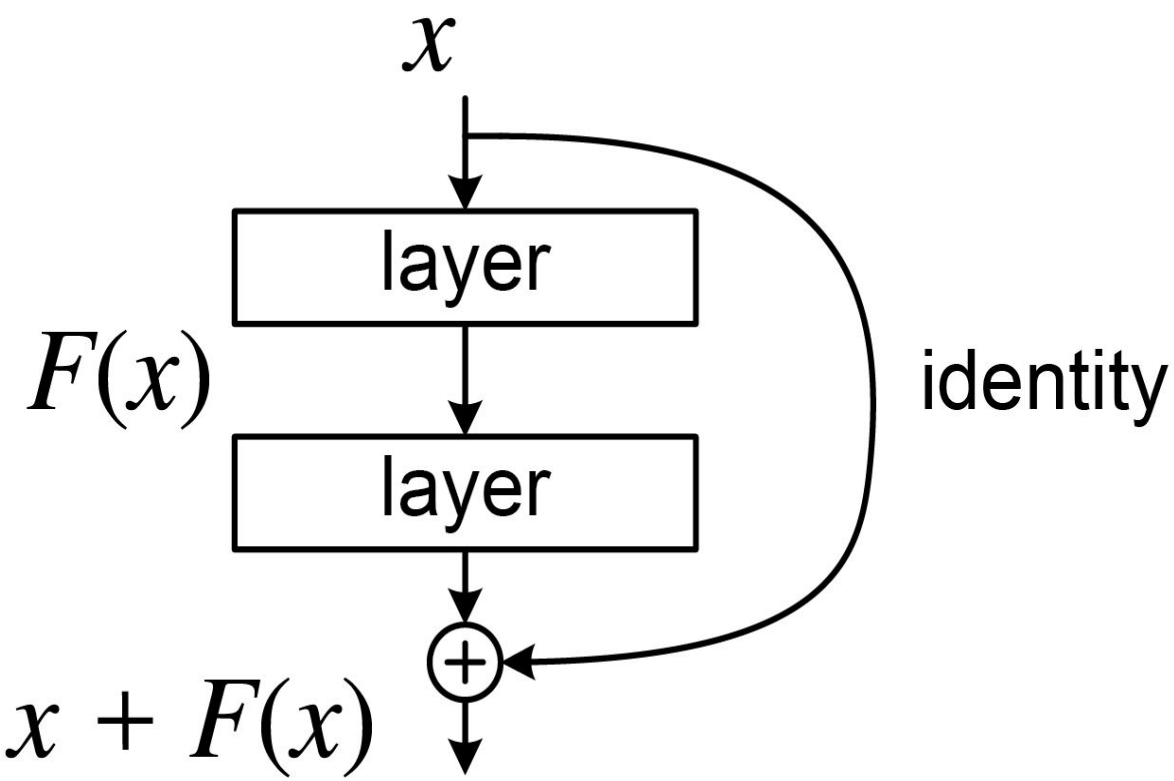
ResNet 101

- Deep CNN
- It has residual connections or (skip connections)



Sonavane, R., Ghonge, P., & Patil, S. U. (2024). Exploring resnet101, inceptionv3, and xception for modi script character classification. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(17s), 117-124.

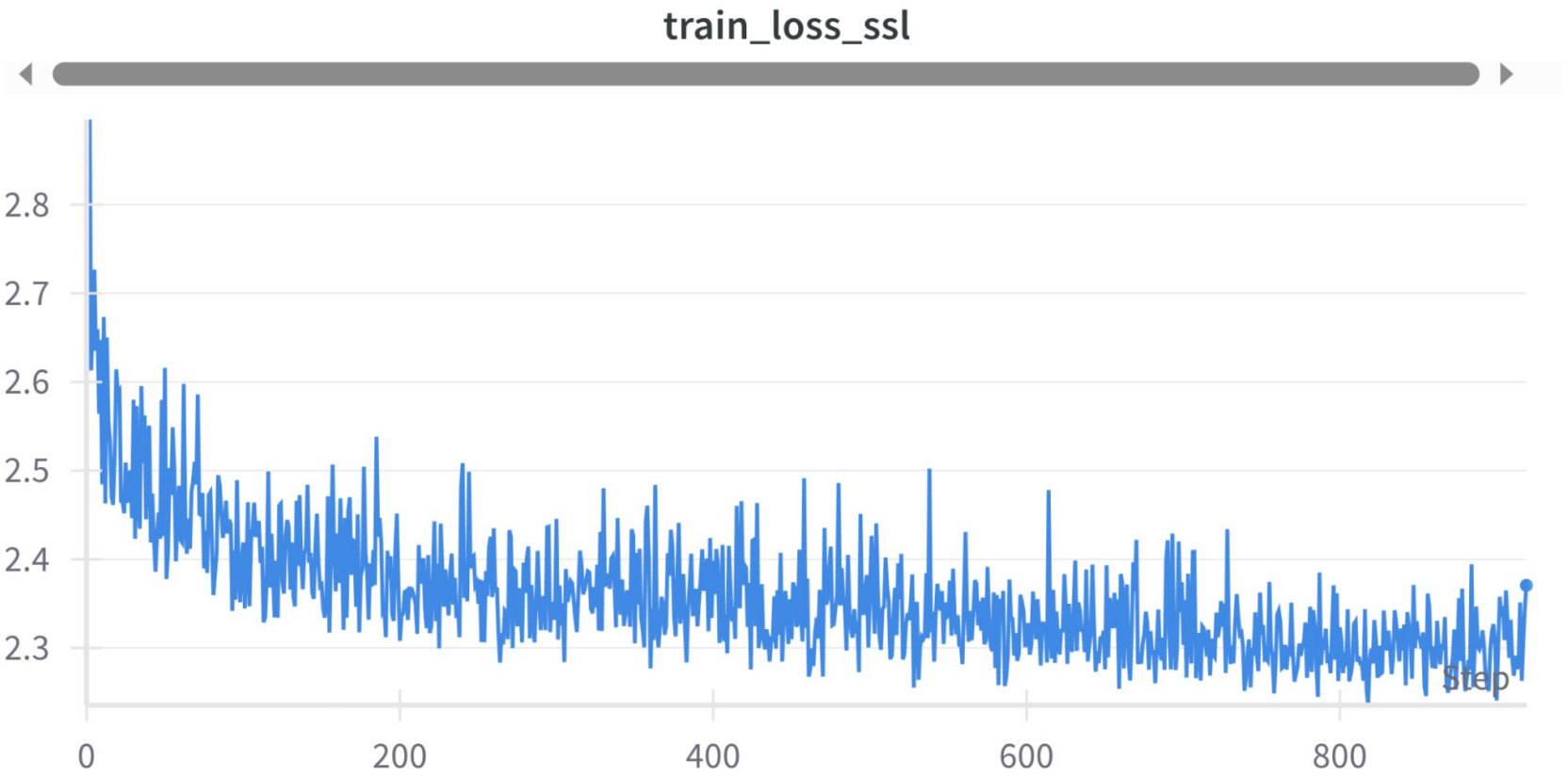
Residual Connections



- Effective for learning color, Fabric, and texture.
- They allow information (especially gradients) to flow more easily during training.
- This avoids the vanishing gradient problem
- Preserve high-frequency details. (color shifts, texture details.)

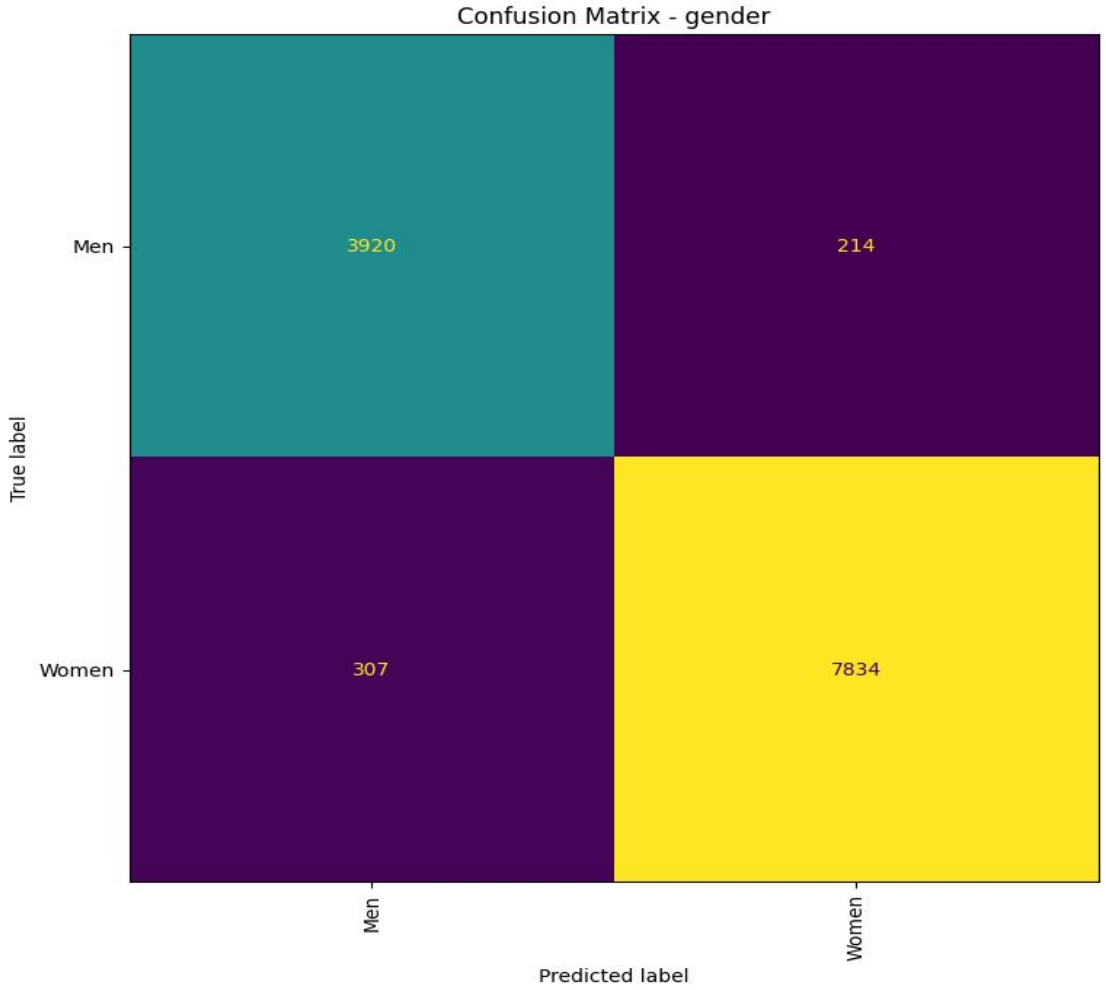
He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

ResNet Training Loss



Loss function of the ResNet 101 architecture training in a self-supervised way using the SimCLR implementation

Confusion Matrix for Classification Model



Positive Class = Woman Negative Class = Man

True Positives (TP) = 7834

False Negatives (FN) = 307

True Negatives (TN) = 3920

False Positives (FP) = 214

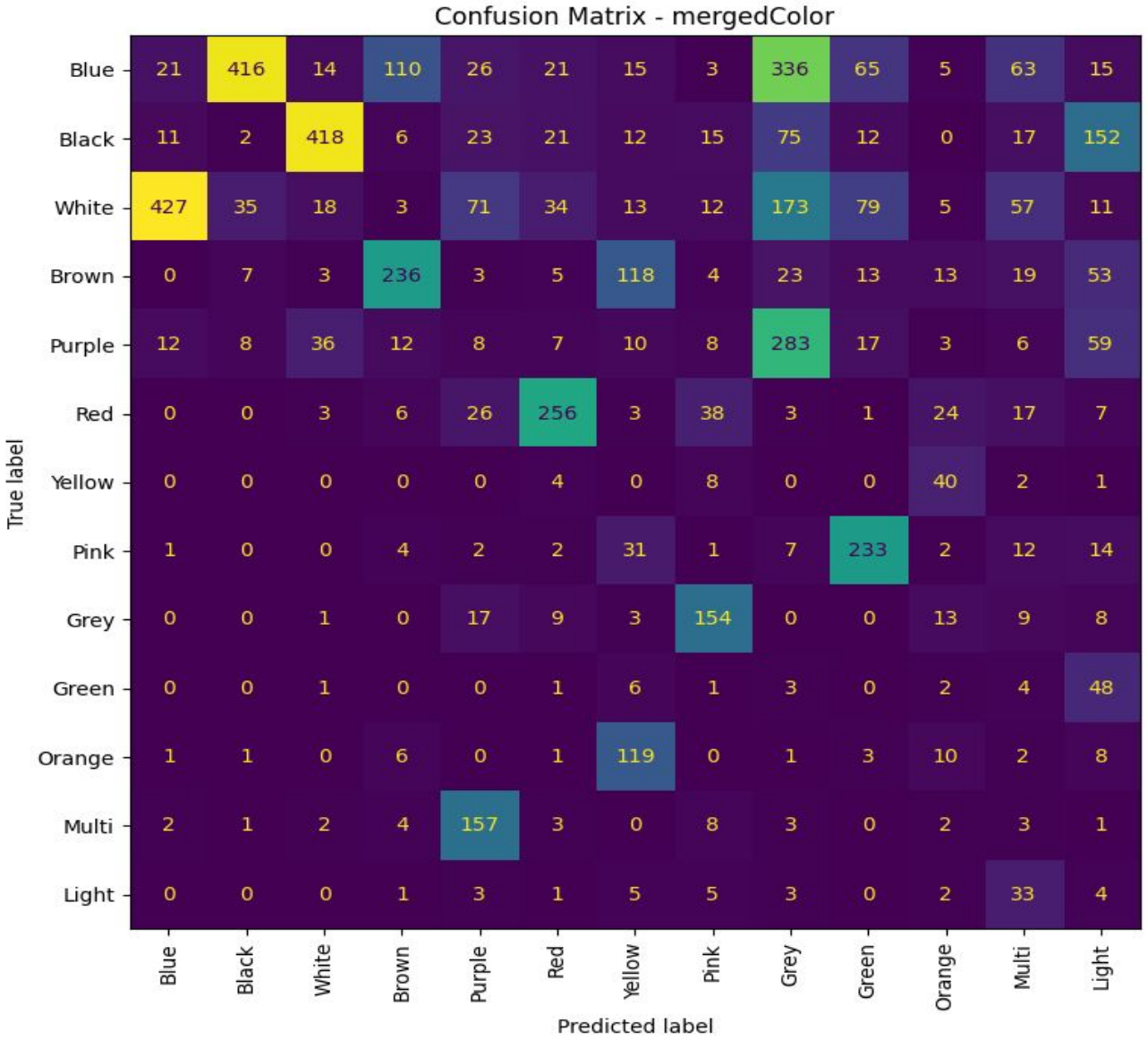
Classification Metrics for Gender

Accuracy: 95.76%

Precision: 97.34%

Recall: 96.27%

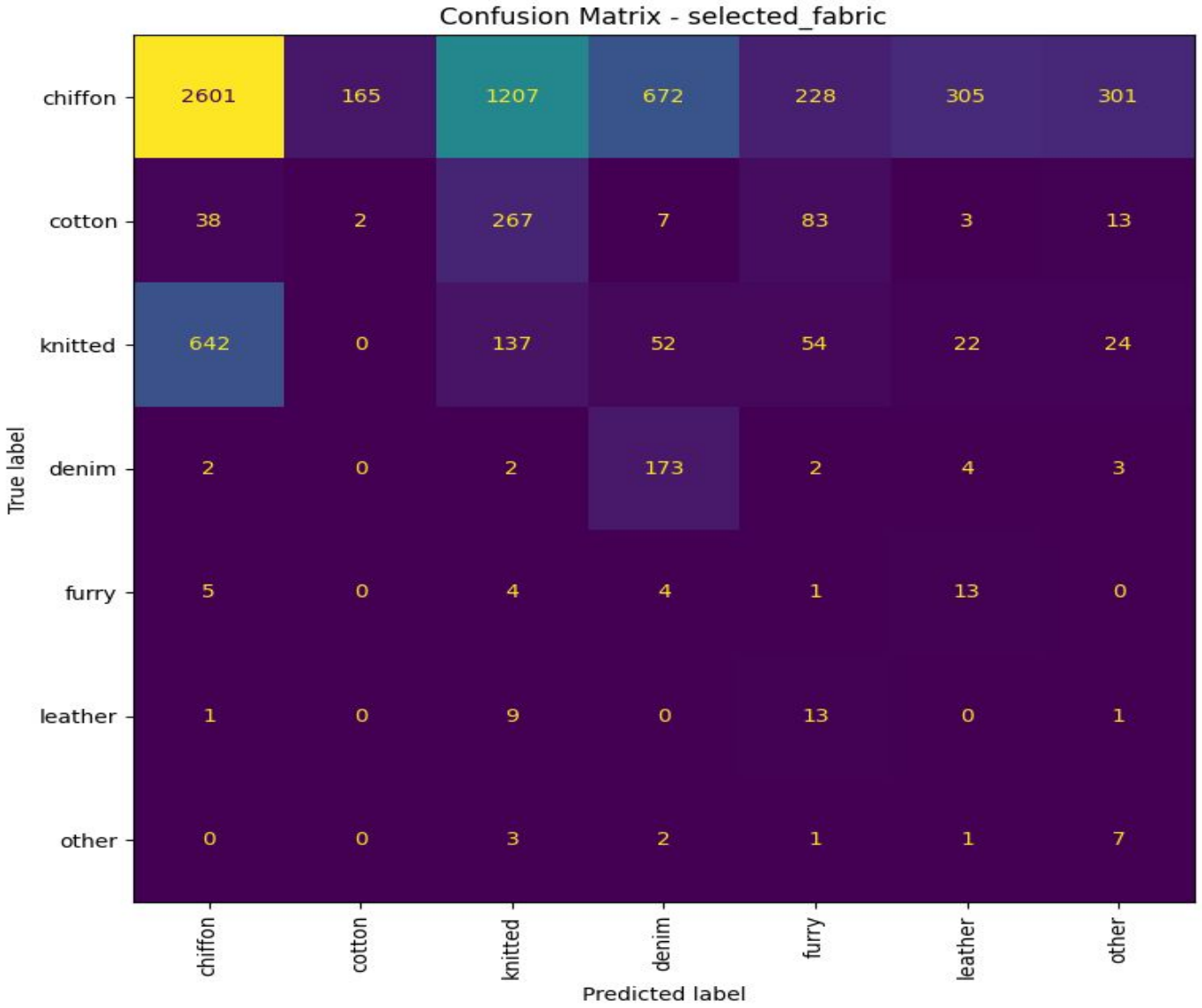
F1 Score: 96.80%



True (Actual Class)	Commonly-Misclassified
Blue	Black
White	Black
Black	White
Purple	Grey
Yellow	Orange
Pink	Green
Grey	Pink
Green	Light
Orange	Yellow
Multi	Purple
Light	Multi

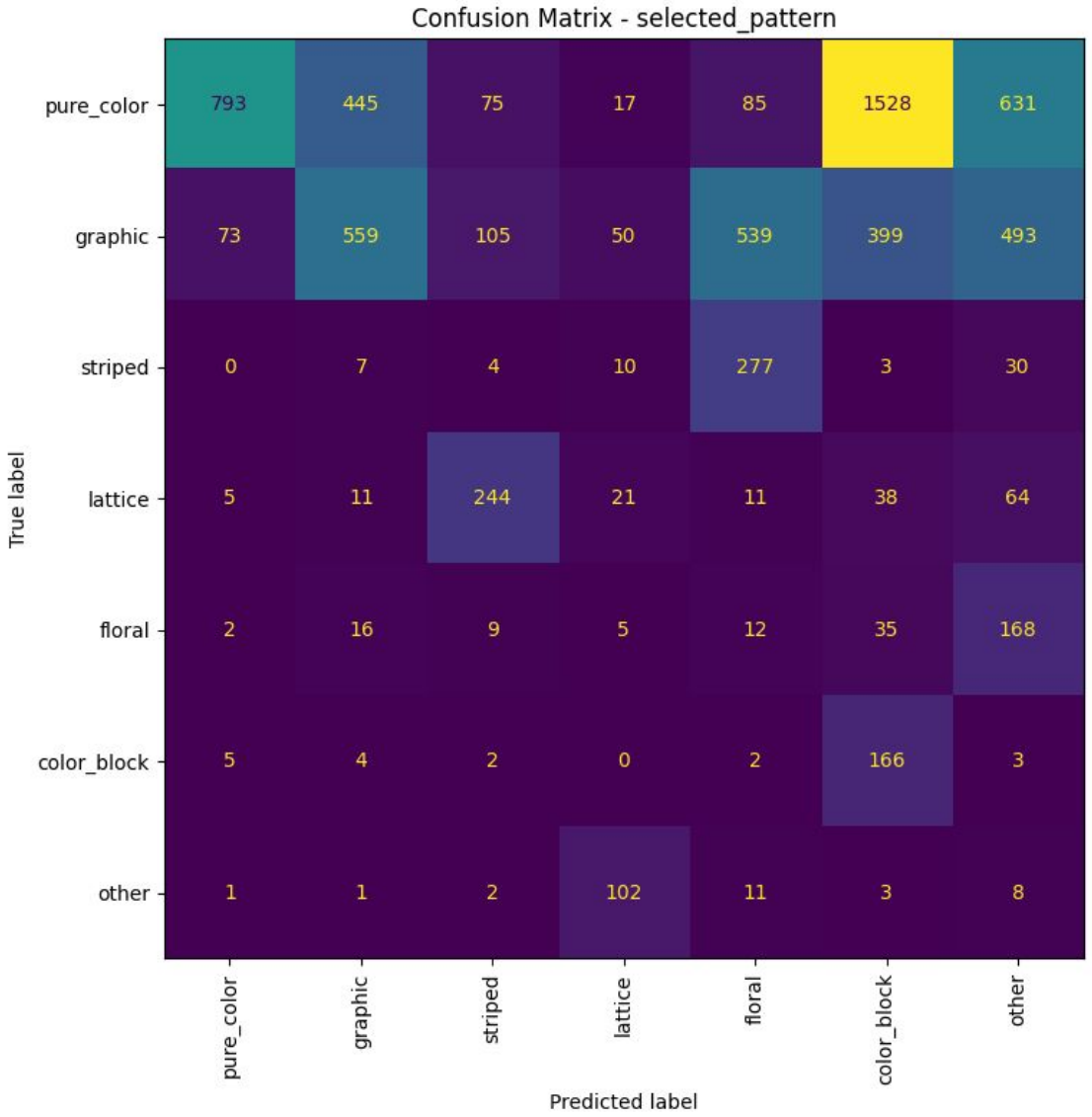
Mislabelled samples

True (Actual Class)	Commonly-Misclassified
Blue	Black
White	Black
Black	White
Purple	Grey
Yellow	Orange
Pink	Green
Grey	Pink
Green	Light
Orange	Yellow
Multi	Purple
Light	Multi



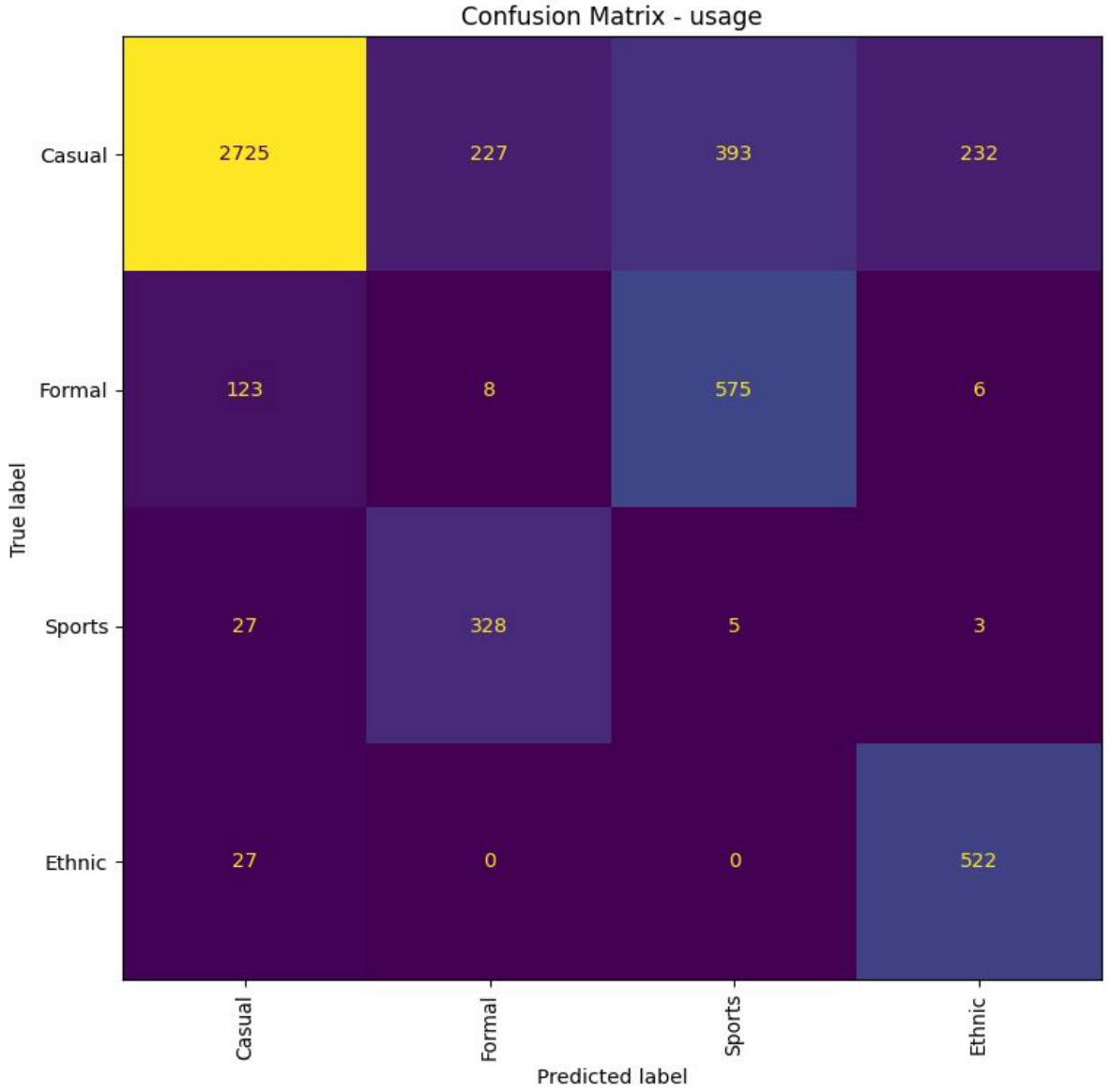
Problematic Classes

True (Actual Class)	Commonly-Misclassified
Cotton	Knitted
Knitted	Chiffon
Furry	Leather
Leather	Furry



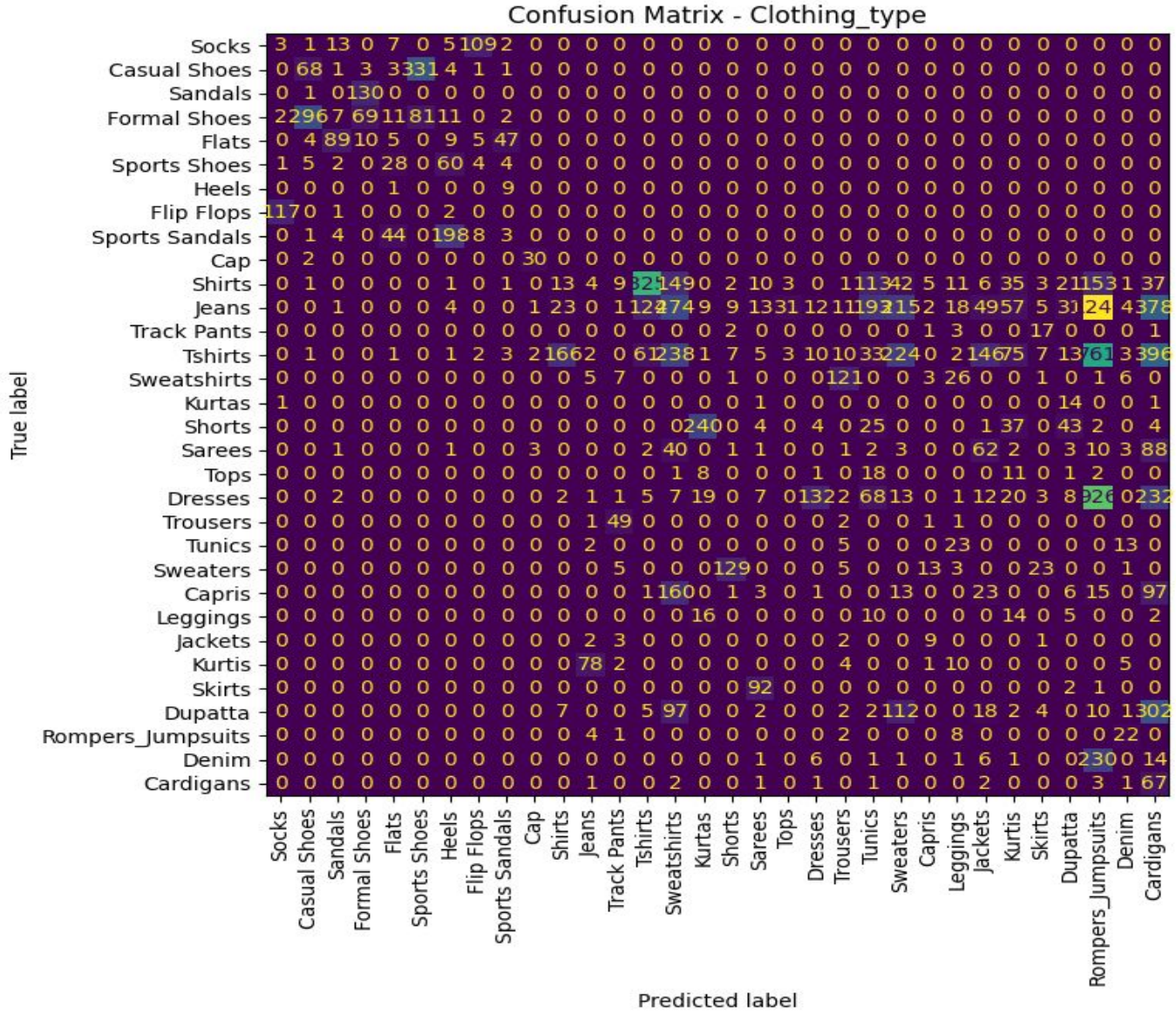
Problematic Classes

True (Actual Class)	Commonly-Misclassified
pure-color	color-block
stripped	floral
lattice	stripped
floral	other
other	lattice



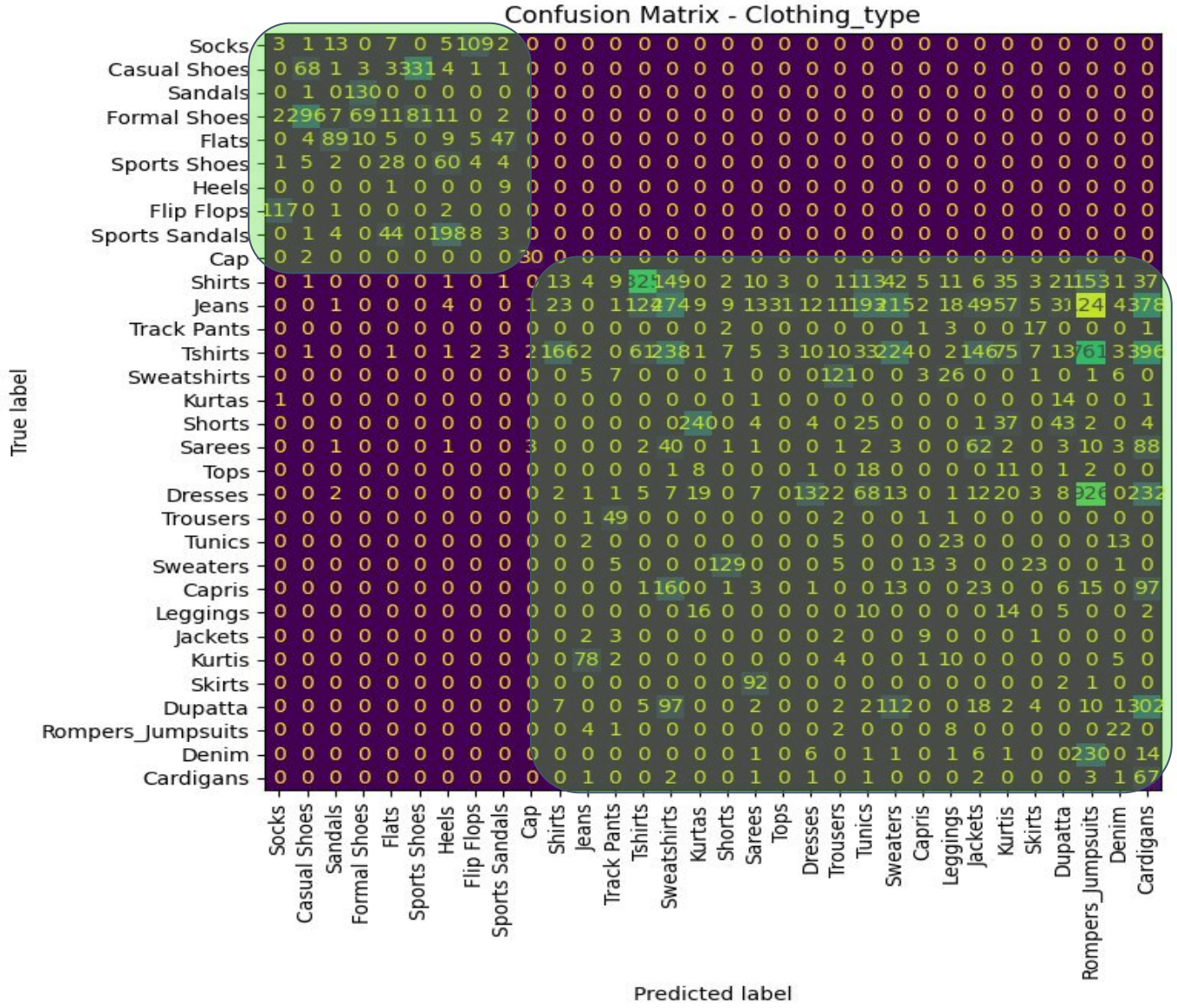
Problematic Classes

True (Actual Class)	Commonly-Misclassified
Formal	Sports
Sports	Formal



Problematic Classes

1) Jeans confused with other bottom wear
2) Socks and Flip Flops

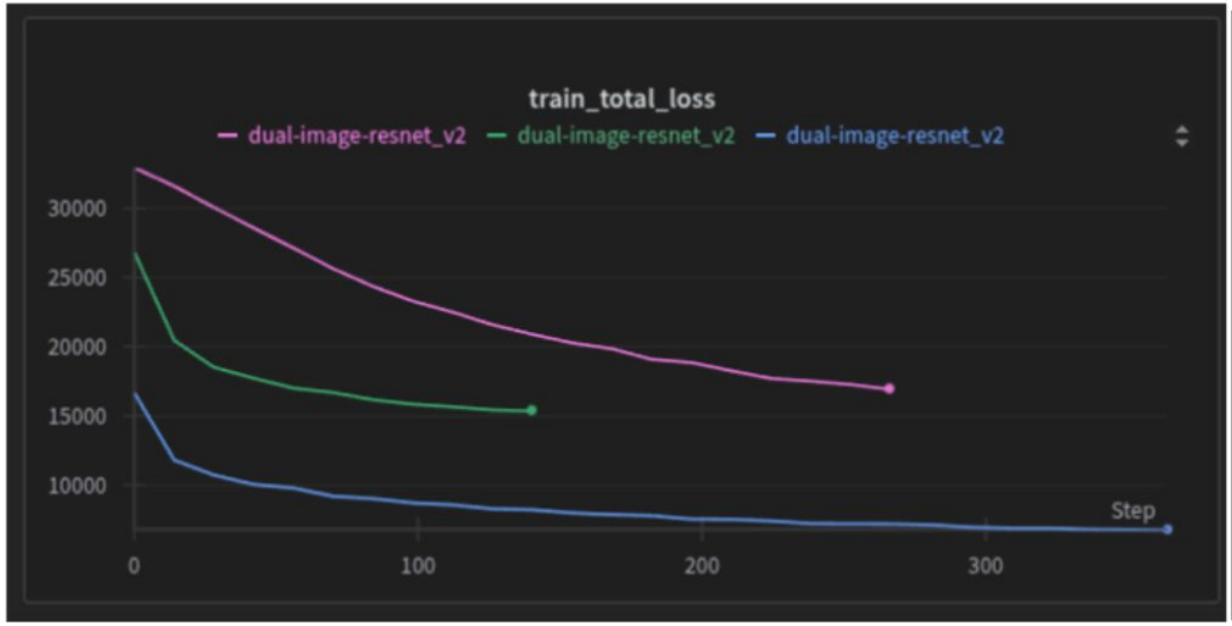


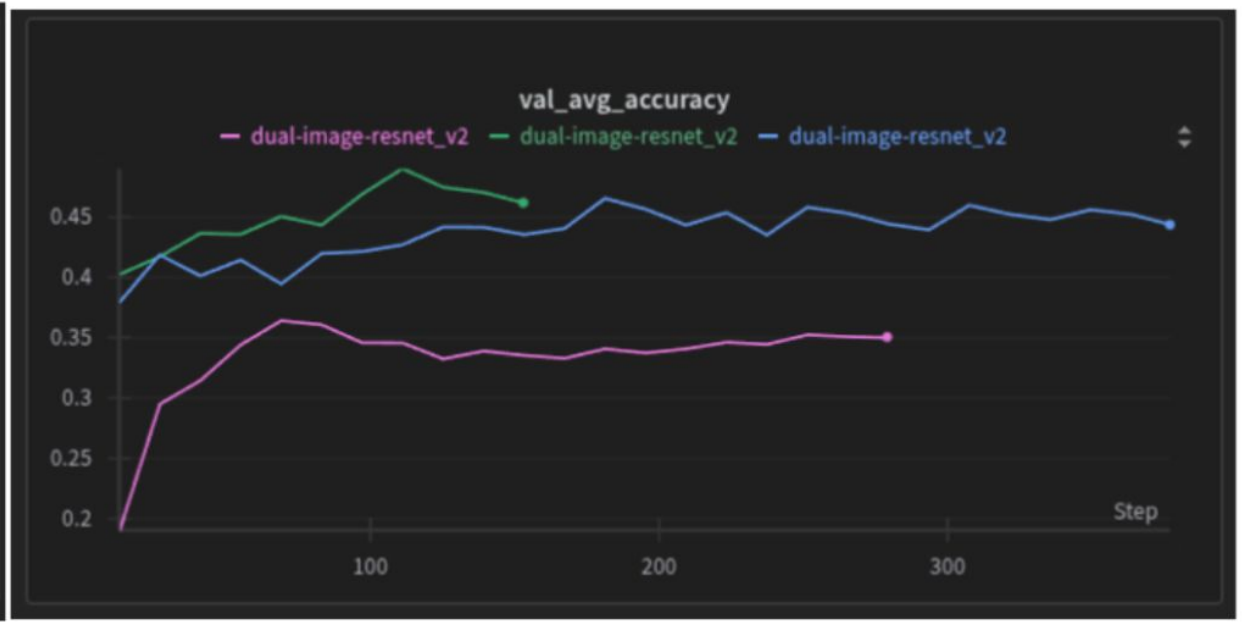
We have some structure in the matrix: Footwear and Clothes are unlikely to be confused.

The final model performance on the validation set

	Gender	Usage	Fabric	Pattern	Color	Clothing type
Recall	0.95	0.42	0.26	0.21	0.12	0.07
Precision	0.95	0.24	0.16	0.24	0.12	0.07
Accuracy	0.96	0.60	0.41	0.30	0.13	0.04

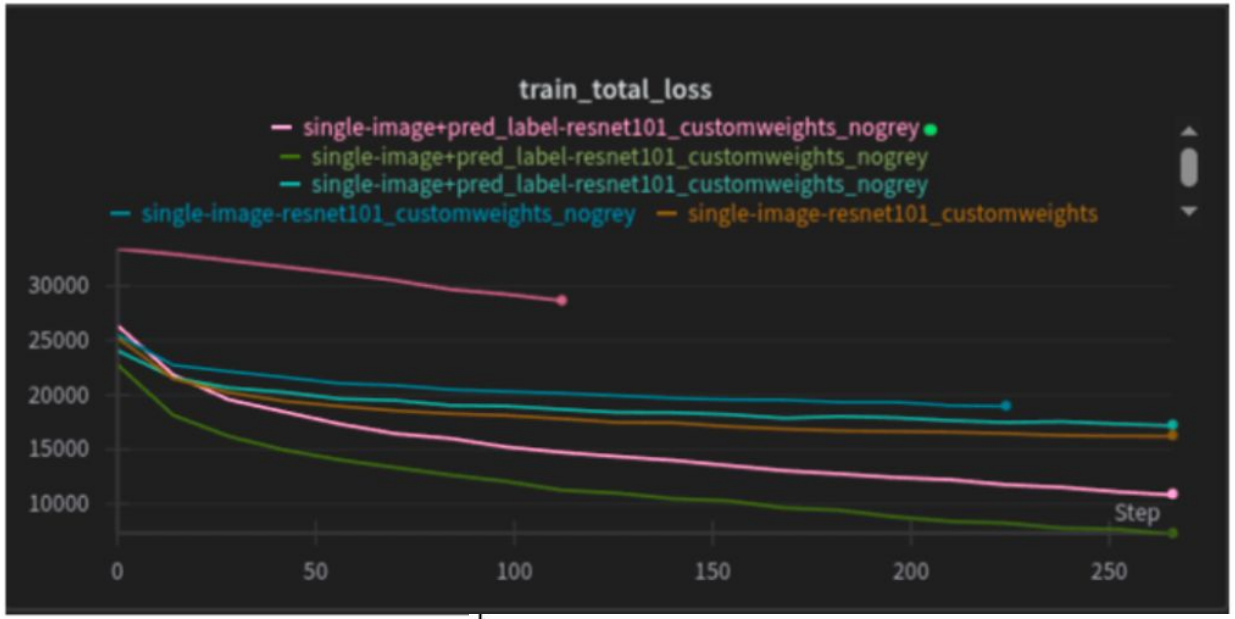
Experiments

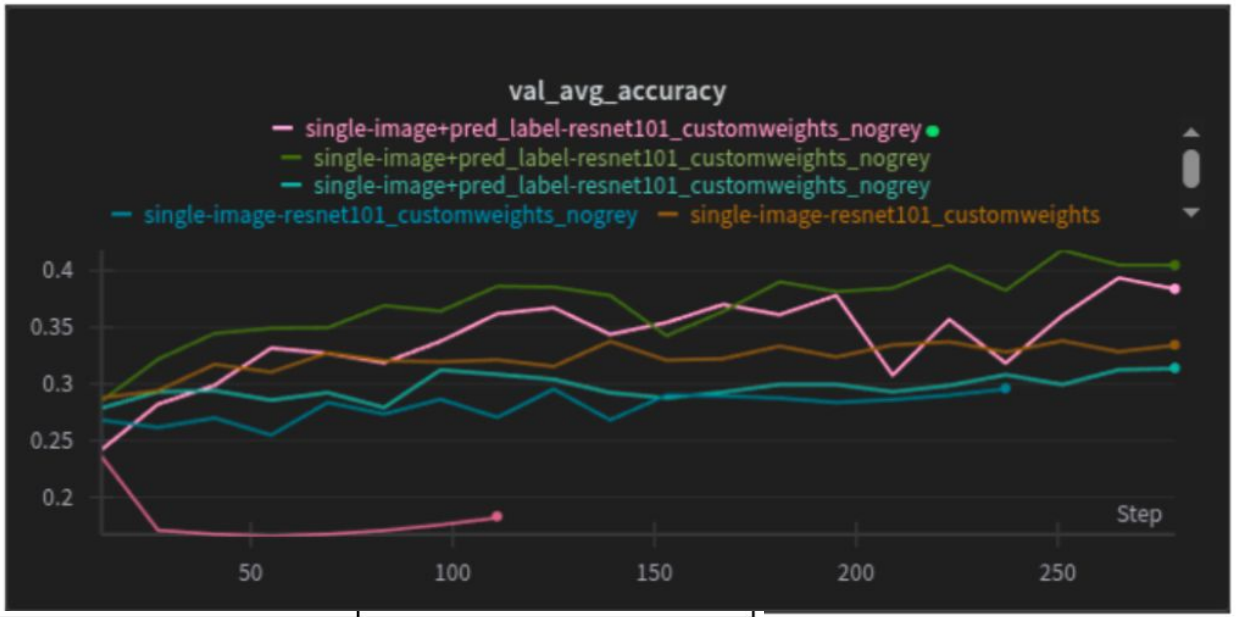




Architecture	Val. avg. acc.
Starting structure	0.35
ResNet50, 3 FC layers, Dropout	0.44*
BatchNorm, masked loss	0.46*

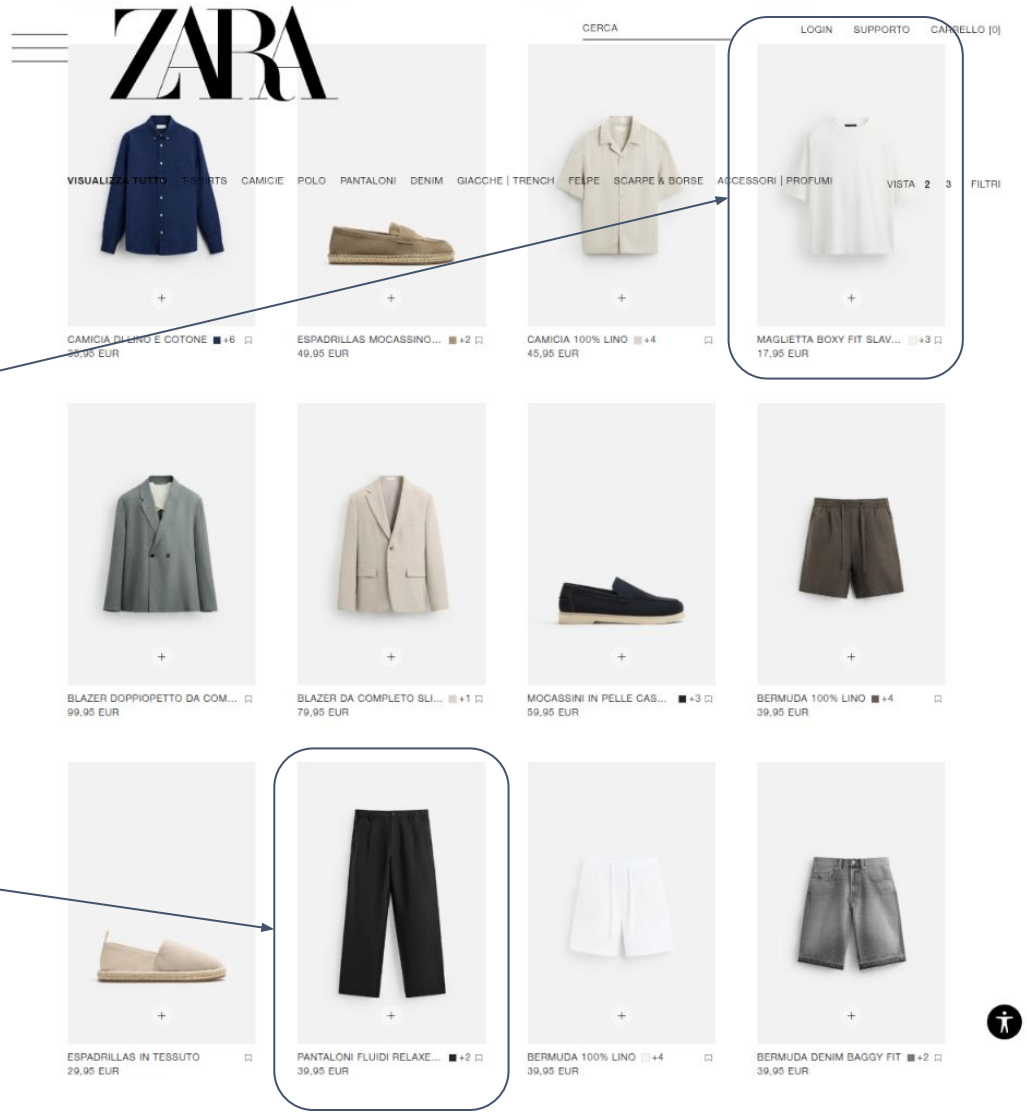
Experiments





SimCLR ResNet101 for custom pre-training, 2 FC layers, no Dropout, Input: cropped images	0.29
3 FC layers, Dropout, weight decay	0.31
Input: cropped images + YOLO predicted label	0.33
Backbone unfreezed: slow training	0.38
Reduced regularization (Dropout and L2)	0.41

The Matcher


ZARA
CERCA
LOGIN
SUPPORTO
VISUALIZZA TUTTO
T-SHIRTS
CAMICIE
POLO
PANTALONI
DENIM
GIACCHE | TRENCH
FELPE
SCARPE & BORSE
ACCESSORI | PROFUMI
VISTA 2 3
FILTRI
CAMICIA DI LINO E COTONE
ESPADRILLAS MOCASSINO...
CAMICIA 100% LINO
MAGLIETTA BOXY FIT SLAV...
BLAZER DOPPIOPETTO DA COM...
BLAZER DA COMPLETO SLI...
MOCASSINI IN PELLE CAS...
BERMUDA 100% LINO
ESPADRILLAS IN TESSUTO
PANTALONI FLUIDI RELAXE...
BERMUDA 100% LINO
BERMUDA DENIM BAGGY FIT

The Matching strategy

‘query’

The Matching strategy

‘query’

detectors

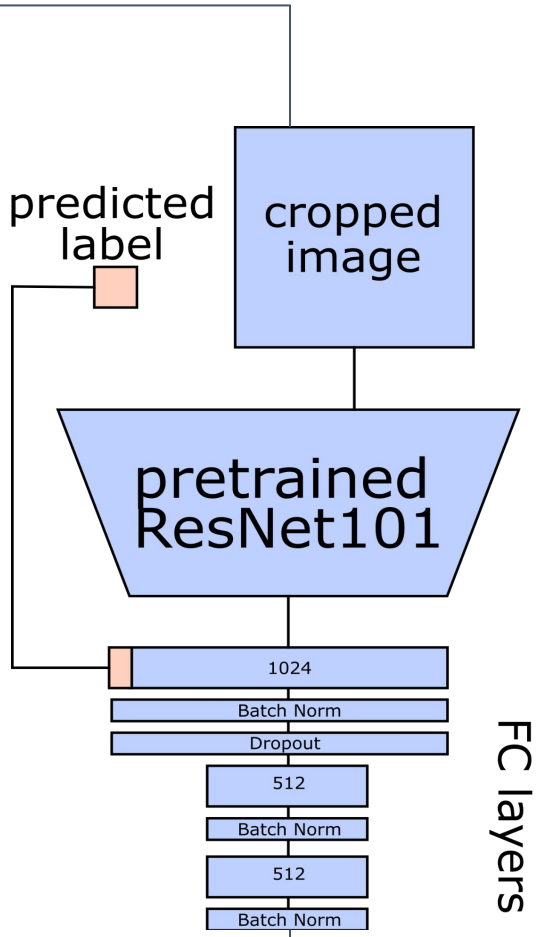
predicted label	1 (‘bottomwear’)
confidence	0.34

The Matching strategy

‘query’

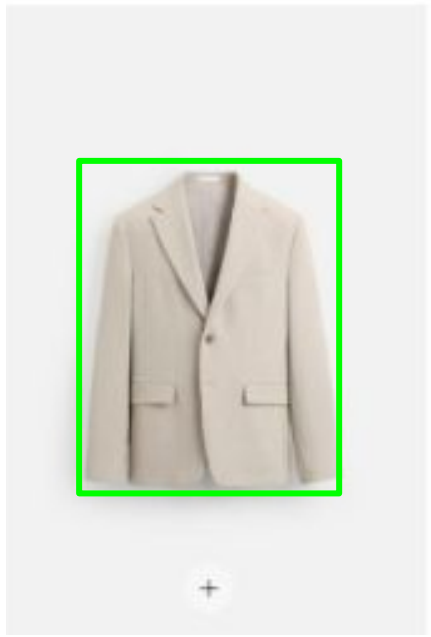
detectors

predicted label	1 (‘bottomwear’)
confidence	0.34



vectorial embedding of items

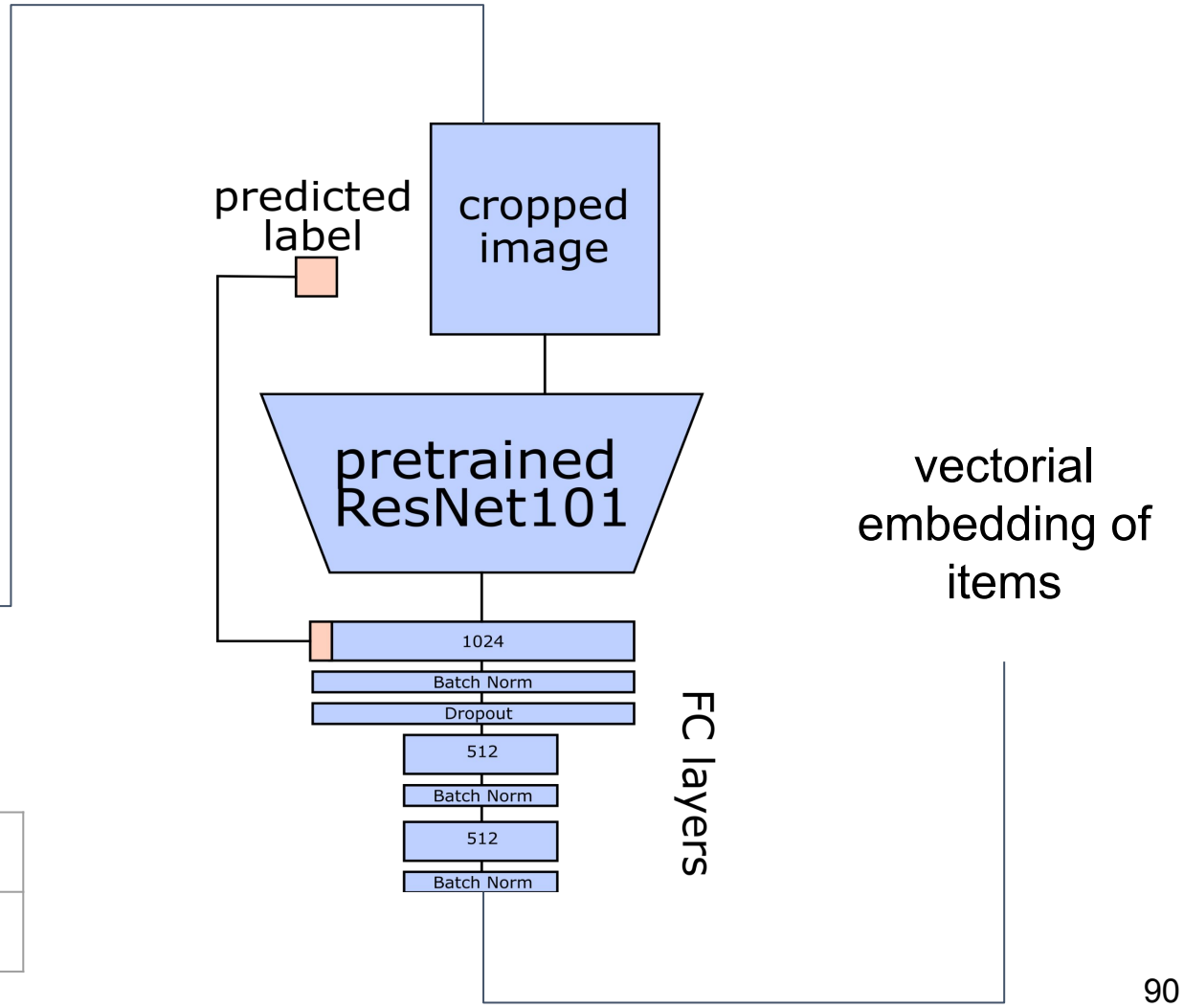
The Matching strategy



detectors

predicted label	0 (outwear)
confidence	0.71

‘wardrobe’



The Matching strategy

Each item is mapped into a **512**-dimensional, batch-normalized **vector**.
The matching assumption: **similar items lay close** in the embedding space

How to evaluate the matcher **performance**?

Again on DeepFashion Dataset

‘pos’ information: images on outfit are wore-on, while images in ‘wardrobe’ are flat

Again on DeepFashion Dataset

Again we used the naming convention to collect clothes with a ‘flat’ and a not-’flat’ version

Set of Queries

Wardrobe

Matcher Evaluation Strategy

In this scenario we have the **ground truth**

Set of Queries

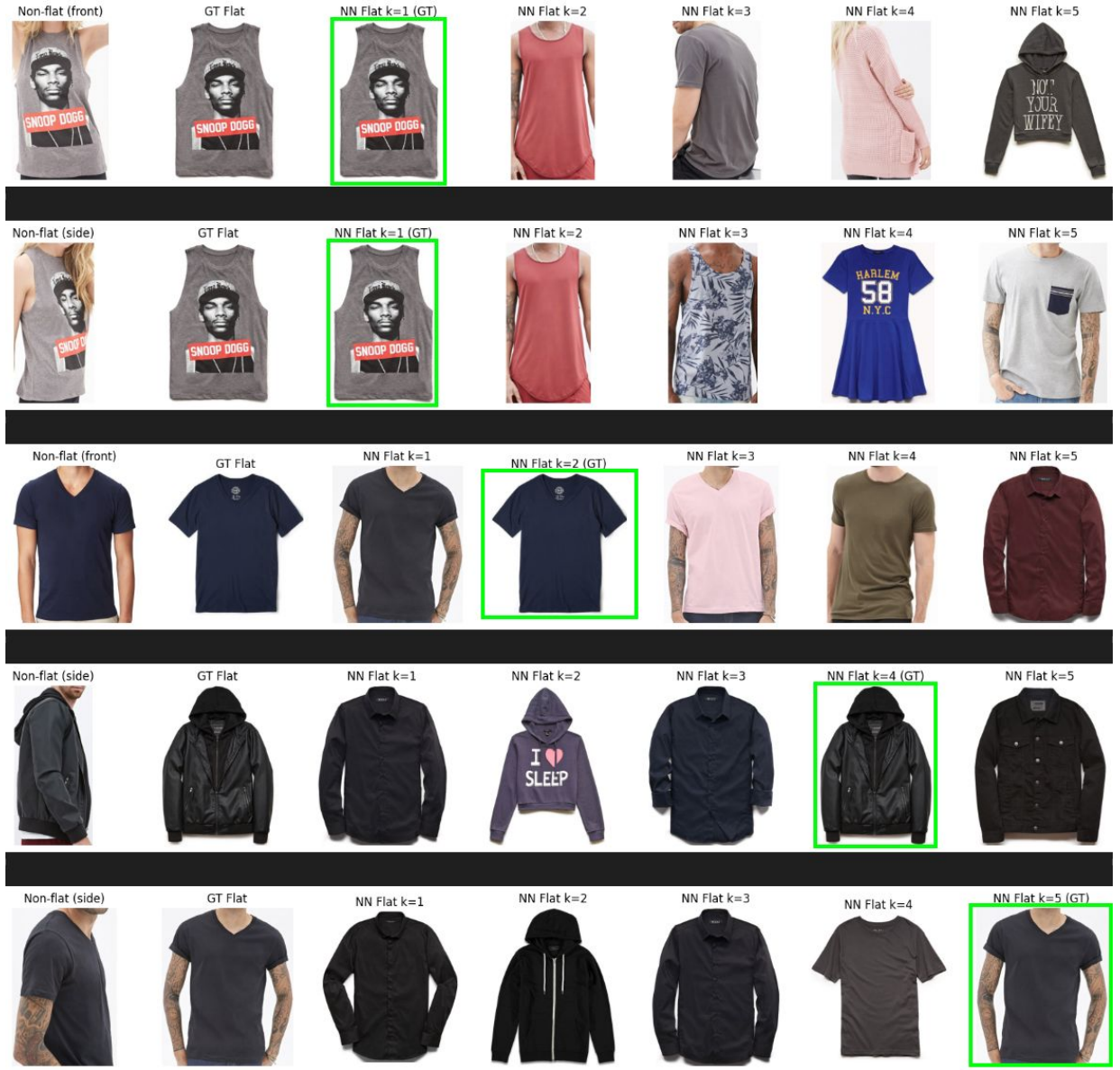
Wardrobe

Matcher Evaluation Strategy

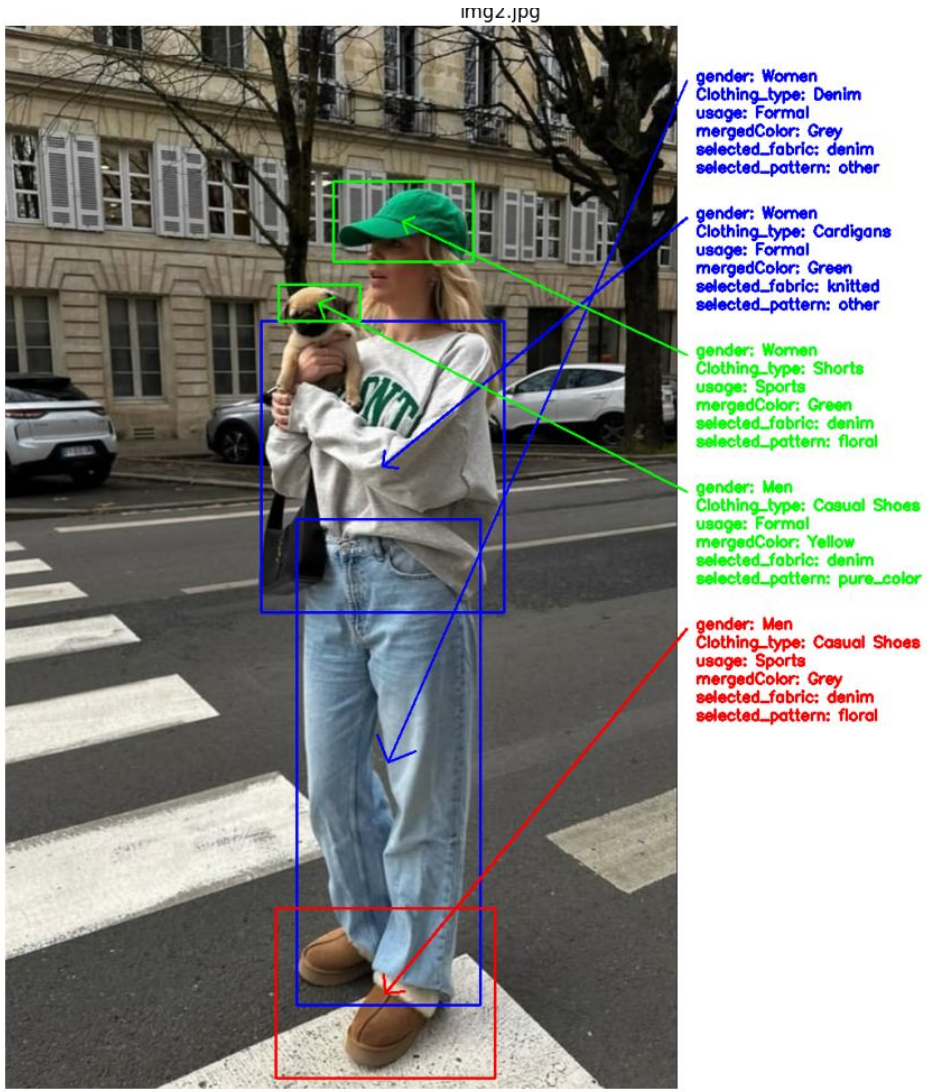
Is the ground-truth within the k - closest neighbors of the query encoding, in the wardrobe space?

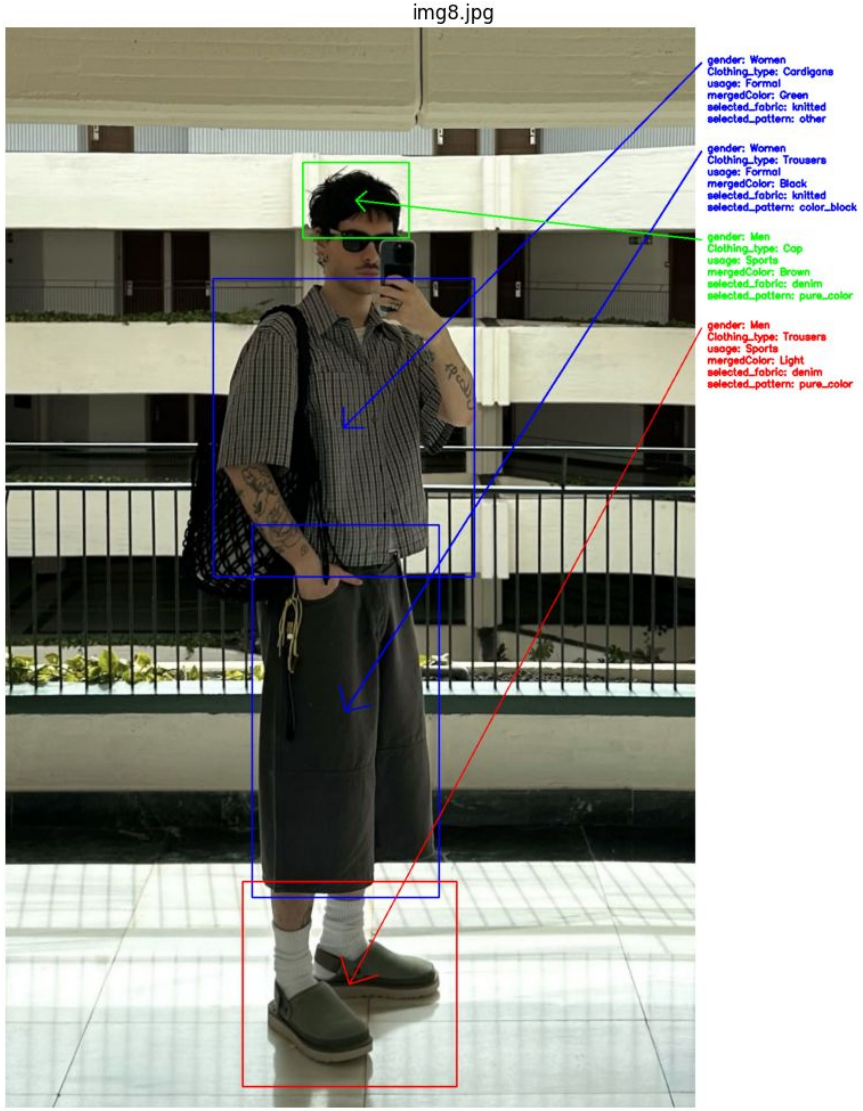
Matcher Evaluation: Results

Top-k	Accuracy
1	37.36%
2	48.28%
3	56.90%
4	63.79%
5	67.24%



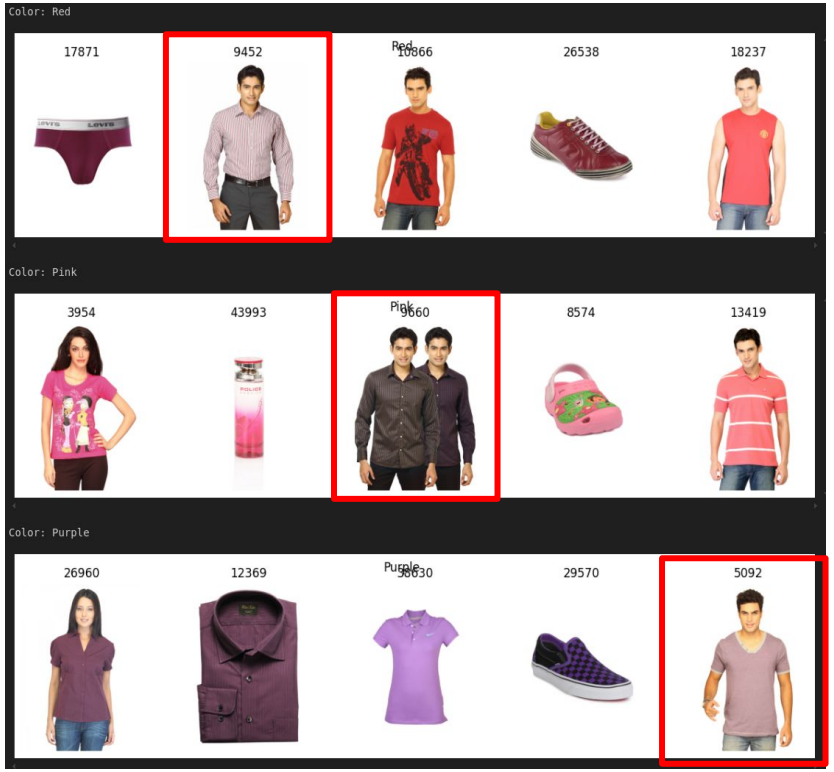
Conclusion


img2.jpg
gender: Women
Clothing_type: Denim
usage: Formal
mergedColor: Grey
selected_fabric: denim
selected_pattern: other
gender: Women
Clothing_type: Cardigans
usage: Formal
mergedColor: Green
selected_fabric: knitted
selected_pattern: other
gender: Women
Clothing_type: Shorts
usage: Sports
mergedColor: Green
selected_fabric: denim
selected_pattern: floral
gender: Men
Clothing_type: Casual Shoes
usage: Formal
mergedColor: Yellow
selected_fabric: denim
selected_pattern: pure_color
gender: Men
Clothing_type: Casual Shoes
usage: Sports
mergedColor: Grey
selected_fabric: denim
selected_pattern: floral


img8.jpg
gender: Women
Clothing_type: Cardigans
usage: Formal
mergedColor: Green
selected_fabric: knitted
selected_pattern: other
gender: Women
Clothing_type: Trousers
usage: Formal
mergedColor: Black
selected_fabric: knitted
selected_pattern: color_block
gender: Men
Clothing_type: Cap
usage: Sports
mergedColor: Brown
selected_fabric: denim
selected_pattern: pure_color
gender: Men
Clothing_type: Trousers
usage: Sports
mergedColor: Light
selected_fabric: denim
selected_pattern: pure_color

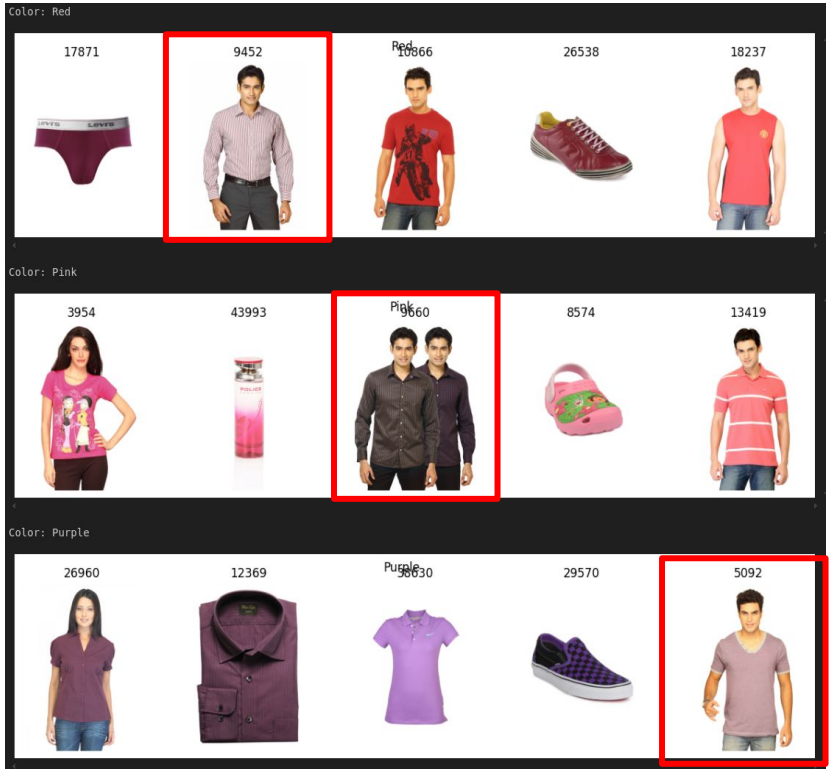
Further Improvements

rubbish in - rubbish out: Fashion Product Images Dataset

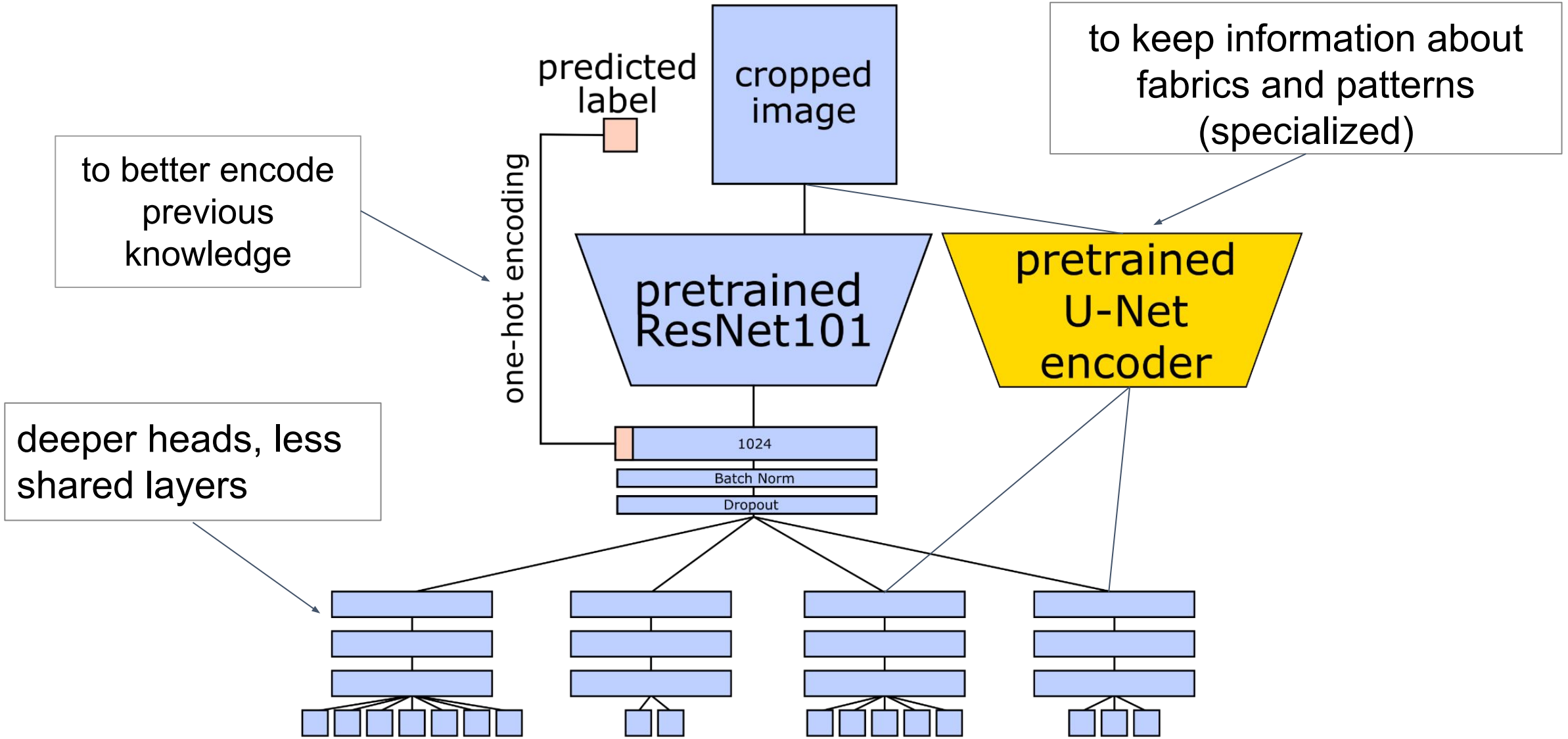


Further Improvements

Different dataset (complex to find) or weak-classifier(s) to re-label the dataset through unsupervised learning (by clustering features)



Further Improvements



Thank you for your attention!

Mattia Roccatello
Ulaşcan Akbulut
Ayşenur Oya Özen

UNIVERSITÀ
DEGLI STUDI
DI PADOVA