Tartu University

Faculty of Science and Technology

Institute of Technology

Aleksandr Kalinin

Generative AI-based Style Recommendation Using Fashion Item Detection and Classification

Bachelor’s thesis (12 EAP)
Science and technology

Supervisors: Mr. Akbar A. Jafari
Prof. Gholamreza Anbarjafari (Shahab)

Tartu 2024

Resümee/Abstract

Generatiivne AI-põhine stiilisoovitus, mis kasutab moesemete tuvastamist ja klassifitseerimist.

Käesolev lõputöö kirjeldab tipptasemel stiilisoovitussüsteemi loomist, mis kasutab moefotode analüüsimiseks genereerivat tehisintellekti ja süvaõpet. Süsteemi eesmärk on töödelda sisendpildid, näiteks selfie'd või stuudiokvaliteediga fotod, ning väljastada tekstifail koos ulatusliku tagasisidega inimese stiili kohta ja soovitustega selle parandamiseks. Süsteem koosneb kahest põhikomponendist: YOLOv8 konvolutsiooniline närvivõrk, mis tuvastab ja lõikab rideesemeid, ning GPT-4.0 suurkeelemudel, mis genereerib informatiivseid stiilikommentaare ja soovitusi. YOLOv8 on lühidalt treenitud konkreetse andmekogumiga, et parandada selle tulemuslikkust 10 eri tüüpi riietuse äratundmisel, samas kui GPT-4.0, millele on juurdepääs OpenAI API kaudu, ülesandeks on anda sidusaid ja lühikesi stiilisoovitusi. Soovitatud lahenduse edukuse hindamiseks viidi läbi reaalseid katsekatsetusi mitmetel üritustel Madridis ja Tallinnas. Võrdluseks kasutati kolme tuntud tehisintellekti mudelit: OpenAI GPT-4.0 Vision, Google'i Gemini 1.5 Pro ja Anthropic'i Claude 3 - Opus. Osalejad hindasid iga mudeli moesoovituste kvaliteeti. Tulemused näitasid, et GPT-4.0 Vision ja Gemini 1.5 Pro said võrreldavad keskmised hinnangud, mis viitavad kõrgemale tajutud kvaliteedile kui Claude 3 - Opus. Käesolev lõputöö näitab, kuidas tipptasemel arvutinägemise ja loomuliku keele töötlemise tehnoloogia võib muuta isikustatud moenõustamise teenuseid, parandades stiilisoovituste täpsust ja asjakohasust.

CERCS: T120 Süsteemitehnoloogia, arvutitehnoloogia)

Märksõnad: Generative AI, deep learning, object detection, image processing, fashion.

Generative AI-based Style Recommendation Using Fashion Item Detection and Classification

This thesis describes the creation of a cutting-edge style recommendation system that uses generative AI and deep learning approaches to analyse fashion photos. The system is intended to process input images, such as selfies or studio-quality photos, and output a text file with extensive feedback on the individual's style and suggestions for improvement. The system consists of two main components: the YOLOv8 convolutional neural network, which detects and crops clothing items, and the GPT-4.0 large language model, which generates informative style commentary and recommendations. YOLOv8 is briefly trained on a specific dataset to improve its performance in recognising 10 different types of clothes, while GPT-4.0, which is accessible via the OpenAI API, is charged with giving cohesive and short style suggestions.

To evaluate the success of the suggested solution, real experimental trials were conducted at many events in Madrid and Tallinn. Three well-known AI models were used for comparison: OpenAI's GPT-4.0 Vision, Google's Gemini 1.5 Pro, and Anthropic's Claude 3 - Opus. Participants judged the quality of each model's fashion recommendations. The results showed that GPT-4.0 Vision and Gemini 1.5 Pro had comparable average ratings, indicating higher perceived quality than Claude 3 - Opus. This thesis demonstrates how cutting-edge computer vision and natural language processing technology may transform personalised fashion advising services, improving accuracy and relevance of style recommendations.

CERCS: T120 Systems engineering, computer technology

Keywords: Generative AI, deep learning, object detection, image processing, fashion

Contents

Resümee/Abstract 2

List of Figures 5

List of Tables 6

Abbreviations. Constants. Generic Terms 7

1 Introduction 8
1.1 Problem Statement . . . 9
1.2 Aims . . . 9

2 Related Work 10

3 Methodology 14
3.1 Convolutional neural networks . . . 14
3.2 YOLOv8 . . . 16
3.3 Large Language Models . . . 20
3.3.1 OpenAI's GPT-4.0 . . . 20
3.3.2 Anthropic's Claude 3 - Opus . . . 20
3.3.3 Google's Gemini 1.5 Pro . . . 21

4 The Results 22

5 Conclusion 31

Conclusion. Future Perspectives 31
5.1 Conclusion . . . 31
5.2 Future Work . . . 31

Bibliography 32

Lisad 33

Non-exclusive license 34

List of Figures

2.1 DeepFashion2 examples. On the left - landmark definitions. Dataset includes commercial - customer pairs of clothes. Images are shown on different levels of difficulty with respect to scale, occlusion, zoom-in, viewpoint parameters . . 11
2.2 Statistics on DeepFashion2. a) Statistics on variations in the dataset with respect to difficulty levels and scale, occlusion, zoom-in, viewpoint parameters b) Number of clothes belonging to different categories. 12

3.1 YOLOv8 architecture [1]. 16
3.2 Summary of convolution, padding, stride and Max Pooling. 17
3.3 Sigmoid linear unit (SiLU). 18

4.1 A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor. 23
4.2 A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor. 24
4.3 A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor. 25
4.4 A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor. 26
4.5 A random sample input, intermediate Yolo V8, and Gemini 1.5 Pro based fashion advisor. 27
4.6 A random sample input, intermediate Yolo V8, and Claude 3 - Opus based fashion advisor. 28

List of Tables

4.1 The average voting of participants for each LLM models. 1: very bad and 4: very good - Number of participants 18. 29
4.2 Performance metrics for YOLOv8 30

Lühendid, konstandid, mõisted

CAGR - Compound Annual Growth Rate

CNN - Convolutional Neural Network

DL - Deep learning

GPT - Generative Pre-training Transformer

HOG - Histogram Of Gradients

LBP - Local Binary Pattern

LLM - Large Language Model

RoI - Region of Interest

RPN - Region Proposal Network

SIFT - Scale Invariant Feature Transform

SVM - Support Vector Machine

UAV - Unmanned aerial vehicle

YOLO - You Only Look Once

DFL - Distribution Focal Loss

CIoU - Complete Intersection over Union

GIoU - Generalized Intersection over Union

BCE - Binary Cross Entropy

SGD - Stochastic Gradient Descent

AP - Average Precision

mAP - mean Average Precision

NLP - Natural Language Processing

MoE - Mixture-of-Experts

SiLU - Sigmoid Linear Unit

ReLU - Rectified Linear Unit

1 Introduction

Fashion is a multibillion-dollar business and as such plays a great role in people's lives. Zion Market Research reports that the global fashion retail market was valued at roughly $91.25 billion in 2023. Experts anticipate it will grow to approximately $157.88 billion by 2032, with an expected compound annual growth rate (CAGR) of around 7.09% from 2024 to 2032. [2]. Companies on the market are seeking better ways to sell their merchandise and maximize their gains, while customers are looking for more convenient ways to search and buy clothes. This has become even more important recently with the rise of the e-commerce. Computer vision and machine learning techniques can be of great help in achieving these goals. Computer vision alongside advanced machine learning algorithms help to deal with a number of different tasks including cloth detection, segmentation, classification, attribute detection, retrieval and others [3].

There are several key concepts and terms that one needs to understand when applying machine learning and computer vision in the fashion industry.

- Cloth detection refers to rough localization of the object. It is usually done by drawing rectangular boxes around the object.

- Cloth segmentation is an exact localization of objects in a pixel-by-pixel manner providing an object mask.

- Cloth classification is assigning a cloth to one of many classes, e.g. type of clothing (dress, shirt, hat). It is important to note that number of classes might vary as different companies might use different classes in their catalogues as well as different datasets might use different number of classes.

- Attribute assigning is assigning many of many attributes to a cloth e.g. color, texture, sleeve length, etc.

- Retrieval is a search for similar objects. It is particularly relevant as customers expect the companies to provide them with easy ways for searching clothing and one of the ways to do so if a customer can give a desired clothing as an input to a search engine and find similar results.

To address the aforementioned key tasks, as well as many other important tasks, various approaches have been developed and utilized [3]. Here are three main streams:

- **Traditional approaches include:** Formula-based approaches where some mathematical formula has been applied to get an output.

- **Traditional features learning** — methods based on manually created simple features (like HOG LBP, SIFT, which are then processed by simple machine learning models: SVM, Bayesian, Random Forest, etc.)

- **Deep learning** - is a relatively recent technique which proved to be particularly useful in solving object detection tasks. It is based on deep neural networks trained on large amounts of varied data. Using large datasets and deep enough models, it is possible to train the model to achieve high accuracy for many of the above mentioned tasks.

That being said, application of computer vision and deep learning is not limited to fashion analysis, but extends to many other areas including unmanned aerial vehicles (UAVs), surveillance, plant phenotypic analysis, etc. [4].

1.1 Problem Statement

In the pursuit of revolutionizing personal fashion advisory, we have developed a cutting-edge solution that leverages the synergistic power of generative AI and deep learning models. Our approach captures an image of the individual, meticulously segments the fashion items, and conducts an in-depth analysis of each piece. This comprehensive analysis enables our system to deliver two insightful sets of statements: the first provides a detailed commentary on the individual's current style, while the second offers personalized fashion recommendations aimed at enhancing their overall aesthetic.

Our extensive investigation into the efficacy of various large language models (LLMs) underpins this solution. We have critically evaluated models from leading AI research entities, including OpenAI's GPT series, Anthropic's Claude 3 Opus, Google's Gemini series, and Meta AI's LLaMA series. This rigorous analysis ensures that our fashion advisory system is built on the most advanced and suitable AI technologies available, optimizing the accuracy and relevance of the style assessments and recommendations.

By integrating these state-of-the-art LLMs, our solution not only understands and interprets the nuances of fashion but also aligns its recommendations with the latest trends and individual preferences. This endeavor represents a significant advancement in personalized fashion advisory, combining sophisticated image processing and AI-driven insights to transform the way individuals approach their style and fashion choices.

1.2 Aims

The major goal of this thesis is to create an advanced style recommendation system capable of analysing an input image (for example, a selfie or a studio-quality photo) and producing a text file with short remarks about the individual's style and ideas for improvement.

The suggested style suggestion system has two main components:

1. Clothing Item Detection: This component uses the cutting-edge convolutional neural network (CNN) YOLOv8 [5] to recognise and crop clothing items. YOLOv8 will be briefly trained on a specific dataset to improve its performance on this task [6].

2. **Style Commentary and Recommendation:** The recognised clothing items will be processed using the GPT-4.0 model, a LLM that includes vision capabilities. GPT-4.0 will create a text file with style comments and recommendations based on the recognised clothing items.

The complete system will be written in Python, with YOLOv8 and GPT-4.0 integrated using Python APIs provided by Ultralytics and OpenAI, respectively. This approach seeks to combine cutting-edge computer vision and natural language processing techniques to provide a comprehensive and informative style recommendation system.

2 Related Work

Major challenges in the field of cloth detection come from the fact that clothes come with great variety of patterns, colors, textures. Different occlusions including self-occlusions often happen negatively impact the quality of object detection and segmentation. Recently, many researchers have made a significant contribution to solve many problems in the field of computer vision. The major advances in the field are represented by extensive exploration and application of deep learning models.

The essential factor for success of the deep learning model is a choice of a suitable dataset. Many datasets have been proposed recently: DeepFashion [7], FashionAI [8], Fashionpedia [9], ModaNet [10] and many others. Researchers in [11] have summarized the advantages and disadvantages of some of the most popular datasets and proposed a new dataset - DeepFashion2.

The importance of choosing a suitable dataset is due to the fact that the model requires a dataset to be trained on. What the model will learn and how effective it will be will depend greatly on the chosen dataset. Consequently, different dataset have different applicability for different scenarios. Datasets differ by number of images, kinds of images (studio-quality, selfies, etc), classes of clothes, the tasks which the model will be trained to solve. Datasets also serve as benchmarks to estimate the efficiency of the model in solving tasks.

DeepFashion2 has several unique characteristics: 1) Large sample size. DeepFashion2 contains 491K images of 43.8K clothing identities of interest (unique garment displayed by shopping stores). On average, each identity has 12.7 items with different styles such as color and printing. DeepFashion2 contains 491K images of 801K items in total. It is the largest fashion database to date.

2) Versatility. DeepFashion2 is developed to support all main fashion analysis tasks which include object detection, segmentation, classification, attribute classification, dense landmark and pose estimation, and retrieval.

3) Expressivity. There are to aspect to it. First, multiple items are present in the image, unlike DeepFashion. Second, there are 13 landmark definitions for each of the 13 types of clothes. Each definition contains 23 landmarks.

4) Diversity. Images collected in this dataset vary greatly. To control their variation each image has been given a level of difficulty for each of the 4 parameters - scale, occlusions, zoom-in and viewpoint.

Scale is the proportion of the item compared to the image size. There are 3 levels of difficulty. For scale, levels of difficulty are: ‘small’ (<10%), ‘moderate’ (10%-40%), ‘large’ (>40%).

Occlusion refers to a cloth being blocked by some other object in the image (hair, human body, accesories, other items). Occlusion is estimated based on the number of landmarks that are occluded. Occlusion can be ‘slight’ (<20% of keypoints occluded), ‘medium’, ‘heavy’.

Zoom-in is characterized by the number of landmarks outside image. There are ‘no’, ‘large’ (>30%) and ”medium” zoom-in options.

Viewpoint can be ‘not on human’, ‘side’, ‘back’.

Intuitively, heavy occlusion, zooming-in are going to make it more difficult for the model to correctly detect the cloth. The same is true for images where picture of the garment is taken from the side or from the back. This intuition is confirmed by the benchmarks with the use of Mask R-CNN.

DeepFashion2 contains bounding boxes with category labels. Human annotators were asked to draw bounding boxes and assign category labels. There are 13 categories in total. In comparison, DeepFashion has 50 categories, however half of them contain only 5% of the images. Additionally, there is an ambiguity between some clothing types that are similar (e.g.'cardigan' and 'coat'). Defining 13 categories in DeepFashion2 solves this problem.

For defining the category DeepFashion2 utilizes landmark definitions. Landmarks are points which when joined together form a contour of the object. These landmarks were drawn by human annotators. Additionally, each landmark is assigned one of two categories: 'visible' or 'occluded'.

DeepFashion2 also contains per-pixel masks of the objects. The mask is generated is semi-automatic manner - initially the mask is generated from the contour and after that human annotators are asked to correct the mask in case it was not generated accurately. Inaccuracies in automatic generation of the masks might result from complex human postures.

Lastly, each image in the dataset is annotated with different style (e.g. color, printing, logo).

The examples of images from DeepFashion2 are shown in Fig. **2.1**. The statistics on DeepFashion2 are presented in Fig. **2.2**.

Figure 2.1: DeepFashion2 examples. On the left - landmark definitions. Dataset includes commercial - customer pairs of clothes. Images are shown on different levels of difficulty with respect to scale, occlusion, zoom-in, viewpoint parameters

There have been a significant progress in developing object detection models. The significant contribution came from advancements in deep learning. One of the state-of-the-art solutions is YOLO models. First iteration of the YOLO model was developed in 2016 [12]. The purpose of YOLO is developing a model that will be fast and precise at the same time.

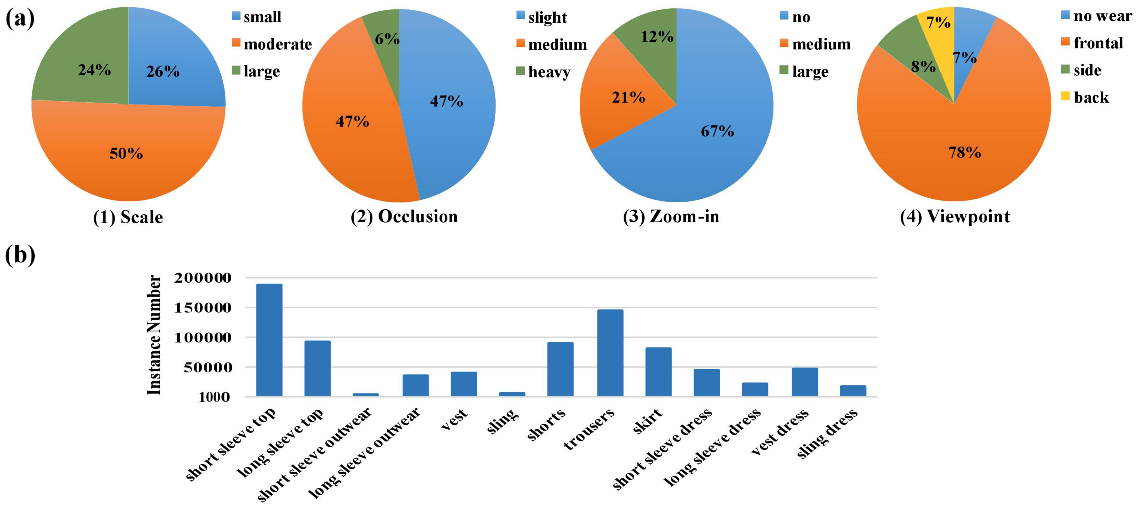

Figure 2.2: Statistics on DeepFashion2. a) Statistics on variations in the dataset with respect to difficulty levels and scale, occlusion, zoom-in, viewpoint parameters b) Number of clothes belonging to different categories.

This is a one-stage algorithm which means that the algorithm at the image once and then outputs the result. This feature makes it faster compared to other two-stage algorithms. It is also a real-time technique capable of detecting objects in video. Combined with remarkable accuracy YOLO is the powerful tool in computer vision analysis. YOLO model starts off with dividing the input image into N by N grid. Then, each cell is encoded by a vector describing it. This vector contains following information: Probability of the object class, coordinates of the center of the bounding box, height and width of the bounding box, number corresponding to the class of the object (e.g. 0 - dog, 1 - cat). If the probability P is 0 then the rest components can have random numbers and are not taken into account. Grid cells not containing the apparel are discarded.

Then there is a potential situation where several bounding boxes are detecting the same object. But since only one bounding box is required non-maximum suppression is applied to discard the remaining boxes. Subsequent version of YOLO are building on top of the original YOLO model introducing new improvements

Another state-of-the-art technique is Mask R-CNN which stands for region-based convolutional neural network. There were several previous versions - R-CNN, fast R-CNN and Faster R-CNN - all building on top of each other and bringing new improvements.

First state of the algorithm is region proposal which is generation of regions where the object might be present. Original R-CNN used a separate module to perform that with newer versions introducing an in-built RPN (region proposal network) module. Second stage is feature extraction where RoI (region of interest) is cropped from the image and resized. Then the features are extracted. After that bounding box regression and non-maximum suppression are applied.

Mask R-CNN is the newest version of R-CNN algorithms. It is an extension of Faster R-CNN. This technique enables object detection and instance segmentation. For instance segmentation Mask R-CNN provides a separate branch which is called Mask Head. It uses extracted features to create a pixel-wise mask of the object.

Researchers in [3] among another thing have highlighted the importance of having a large and representative amount of data. They had mostly studio-quality images with white background. Intuitively, model trained on such dataset might not handle well real-world images. To improve the generalization of the model, they have employed a background augmentation technique. This method substitutes the white background to a random one from a selected pool of backgrounds. Additionally, simple transforms like random change of contrast, hue, brightness, additive RGB noise are applied thereby creating a more ”disturbed” dataset. They hypothesized that the net training result for configuration trained on disturbed dataset should be lower than configuration trained on initial dataset as recognition task is more difficult. However, configuration trained on disturbed images should have higher generalization. Their results were similar to this intuition. They have used SSD300 for object detection and classification task. Model was trained on the subset of DeepFashion dataset.

Contemporary approach to clothing retrieval was explored in [13]. Clothing retrieval is a computer vision task that involves finding clothing items that are visually similar and/or share specific attributes with a given query image. Researchers in this work used Mask R-CNN model. Mask R-CNN is capable of performing object detection and instance segmentation. Mask R-CNN was used to obtain the body mask, collar mask, sleeve category and pocket location information of garment. After that they have utilised a convolutional neural network called VGG16 for feature extraction. They have extracted features of garment body and the collar. Lastly, they have used cosine similarity to measure the similarity between input feature vectors (body garment and collar) and feature vectors of clothes in the database, and showed top results.

An interesting application of deep learning models in agriculture was demonstrated in [4]. Rapeseed is a significant oil crop and plays a vital role in economy of many countries. Therefore, there is a significant need for developing method to maximize rapeseed yield by selection of plants with suitable parameters. It was discovered that yield of the rapeseed is dependent on number of parameters like plant type, plant height, number of branches, etc. Among factors which determine rapeseed yield size and shape of the rapeseed pod play an essential role. It was established that selection of plants with suitable pod size and shape would increase the yield. As it is very tedious to check each plant manually, researchers have applied deep learning methods to solve this problem. Two recent, cutting-edge methods were tested: Mask R-CNN and YOLOv8. A number of images of rapeseed pods were taken. The aim of the research is to detect the pod in the image. Researchers have also demonstrated how plant size can be estimated from the image. It was found that despite superior training model assessment metrics of YOLOv8, MASK R-CNN performed better due to the fact that YOLOv8 mistakenly categorized rapeseed stem as pod body while Mark R-CNN did not encounter the same problem. Additionally, they have compared automatic estimation of pod length, width and area with manual measuring. It was found that method was precise for pods without significant curvature.

3 Methodology

3.1 Convolutional neural networks

YOLOv8 is a deep learning convolutional neural network (CNN). Convolutional neural networks can be described as supervised deep learning (DL) models. Deep learning models are models trained on large amounts of data where computation is done through multi-layer neural networks and processing. Supervised models are ones that are trained on labeled data to perform tasks like classification and object detection. In most cases DL models consist of many hidden layers as well as input and output layers. Input layer simply takes the input which depends on the application (e.g. image, video, audio, text, etc.). Output layer provides the output of the model (e.g. bounding box, class probability, etc.). Hidden layers are essentially computational engines of the network. It is difficult to assign a specific meaning to each hidden layer, which makes DL models unintuitive in many cases.

CNNs are different from other types of supervised DL models in that their architecture in most cases includes a lot of convolution and sampling layers. Choice of convolutional layers as primary layers in the network is important since convolution provides the model with a property known as local connectivity. Local connectivity essentially means that every neuron in a given layer is connected to only a subset of neurons in the previous layer. How large is the subset of those neurons depends on the size of the convolution kernel (also called filter). Such design has several advantages. Firstly, this reduces the computational load, since there are generally less connections in the network and less computations to be performed. Secondly, it preserves spatial information because the value of the particular neuron is dependent on a specific subset of neurons in the previous layer. This property is remarkably relevant in image processing tasks, where this helps to preserve spatial information of the input image.

CNNs learn through backpropagation algorithm. The learnable parameters in the network are called weights and biases. Weights are values of the convolution kernels, and biases are constants, that are added to the result of a single convolution operation.

Backpropagation has been described elsewhere [14] [15]. To put it shortly, learning through backpropagation consists of four main steps:

1. Forward pass. The model is initialized with random weights and biases, then takes an input, passes through the network and outputs the result. The result is likely to be different from the ground truth. The error can be quantified through loss function.

2. The next step is calculation of the gradient of the loss function in the output layer with respect to weights and biases of the network. Since model output ultimately depends on the weights and biases in the network, we calculate the gradient with respect to each weight and bias in the network. The result of that operation is a vector that points in the direction of the steepest increase of the function (our loss function). Conversely, going in the opposite direction

3. Calculation of the gradient of the loss function in other (hidden) layers. Since the loss function does not directly depend on weights and biases in deeper layers, the calculation is more difficult. But since weights and biases in deeper layers still affect higher layers, the gradient with respect to these weights and biases is calculated.

4. Update weights and biases. The last step is to update the weights and biases according to how much those weights and biases contributed to the overall loss, which is given by the gradient calculated in previous steps. Values of the gradient vector determine in which direction should the weights and biases be updated, and how much they should be updated relative to other weights and biases in the network. The important parameter at this step is learning rate, which is constant that determines how much the values will be updated.

Our hope in training a CNN model is that it will eventually converge to a global minimum, although it is not always the case. Choice of learning rate greatly influences convergence of the model. Too large values can lead to overshooting the minimum, potentially leading to the model oscillating around the minimum or even diverging. On the other hand, too small values might lead to slow convergence and possibly to the model getting stuck in the local minimum (e.g. saddle point).

Several issues should be addressed when designing a CNN model. Some of the most important challenges are overfitting, vanishing (and exploding gradients):

1. Overfitting. During training, the model should ideally learn the underlying patterns in the data. However, overfitting occurs when the model learns not only these patterns but also the noise of the training data. Consequently, the model essentially memorizes the dataset, achieving high accuracy on training examples but failing to generalize to new, unseen examples. This over-specialization means that the model performs poorly on real-world data, limiting its practical utility.

   To mitigate overfitting in CNNs, several techniques can be employed, including dropout and early stopping. The dropout technique is based on the observation that models with too many parameters are more likely to overfit because they start memorizing the data instead of learning the underlying patterns. Dropout works by randomly turning off neurons during training, reducing the number of learnable parameters and forcing the network to learn more robust features. During validation and testing, all neurons are active, allowing the full network to be used.

   An opposite problem to overfitting is underfitting. Underfitting occurs when the model is too simple to capture the patterns of the data. Usually, it happens when the model was not trained enough or does not have enough parameters, and fails to capture the underlying patterns in the data, leading to poor performance on both training and new data.

2. Another important problem is that of vanishing gradients. Vanishing gradient is a phenomenon that occurs during training of neural networks, where the gradients become extremely small as they are propagated from the output layer to the earlier layers. In this case, the earlier layers of the network stop making significant contributions to the learning process rendering those layers ineffective and slowing down convergence. Conversely, sometimes the gradients might become extremely large as they are propagated from the output layer to the earlier layers. In this case, it is called exploding gradient. Exploding gradient makes the network unstable, preventing reaching convergence. One of the measures to alleviate vanishing (and exploding) gradients is normalization. Role of normalization is to ensure more stable gradient propagation. To that end, CNNs can

incorporate Batch Normalization layers. Another good practice to counter vanishing gradient is to avoid certain activation functions that map the values to undesirable ranges. An example of that is sigmoid activation function. Sigmoid activation function squishes the input to range from 0 to 1. Therefore, this means that, if the inputs are large, large changes in input result in small changes in output, making derivatives small, thus making deep enough network susceptible to vanishing gradient.

3.2 YOLOv8

The architecture of the YOLOv8 is presented in the Fig. **3.1**.

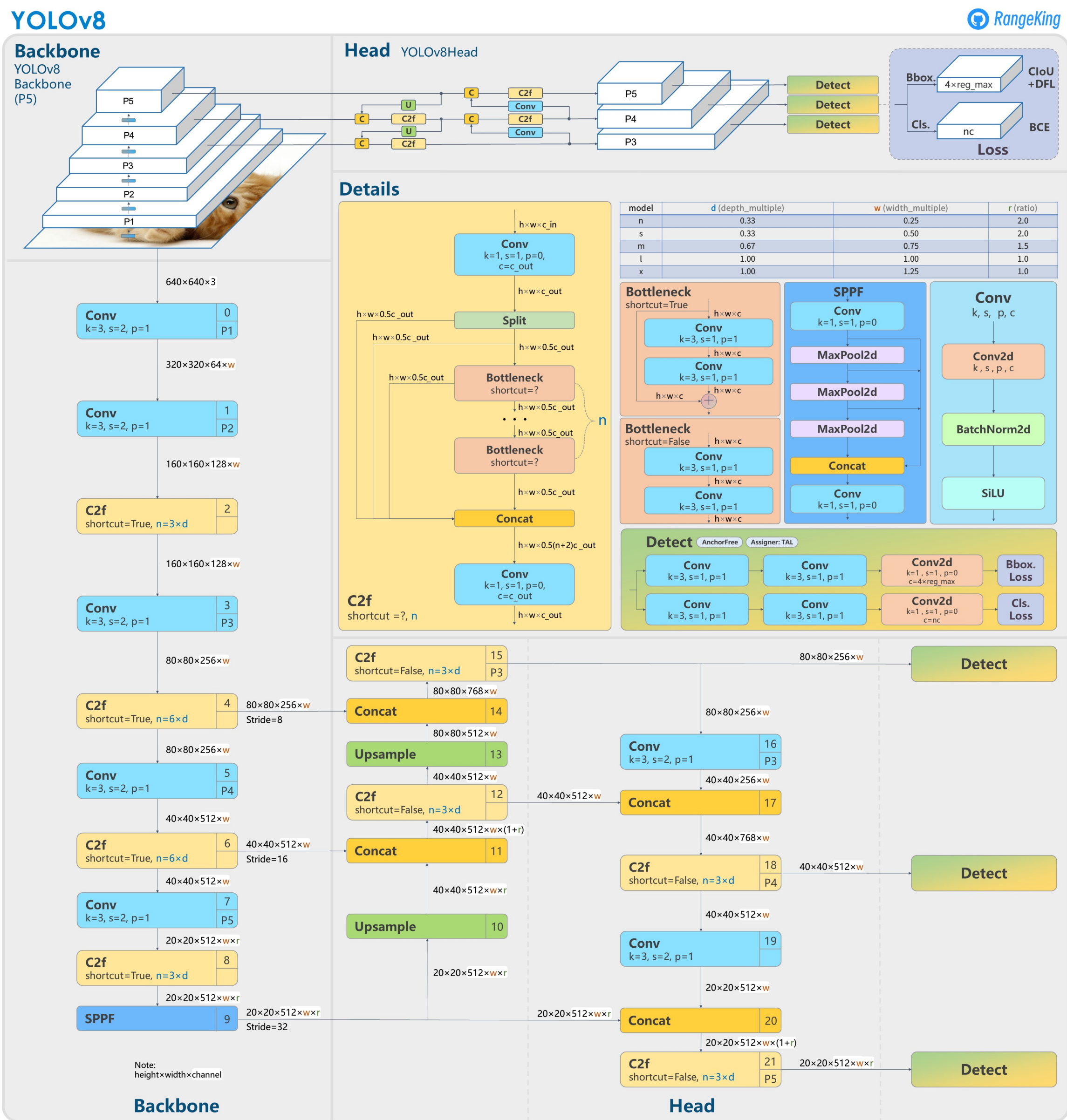

Figure 3.1: YOLOv8 architecture [1].

The model is build on PyTorch library and developed by Ultralytics. Roughly, it can be divided into the backbone and the head. The purpose of the backbone is to extract the features of the input image. These features might include edges, curves, etc. The extracted features are encoded as a tensor called feature map. The head is responsible for processing these features and providing the output of the model (most commonly bounding boxes and class probabilities).

The model itself is composed of a sequence of layers of different kinds. The basic layer in the network is Conv layer. It has 3 components:

1. The main component of that layer is Conv2d from PyTorch library. This layer performs 2D convolution on the input. The parameters k, s, p, c correspond to kernel size, stride, padding and number of channels or filters.

   Kernel is the small matrix that performs the convolution operation sliding over the image. Typically, it is a square matrix with odd dimensions (1×1, 3×3, 5×5, etc.).

   Stride is the step that the kernel takes after performing convolution calculation once. In practice, stride is often set to 1 or 2. A stride of 1 is common when the model needs to maintain a high resolution of features, which is particularly important in the initial layers of the network. A stride of 2 or more may be used in deeper layers or when the input images are large, and the model needs to reduce dimensionality to control the number of parameters and computational cost. .

   Padding is an addition of zeros along the borders of the image which is needed to ensure that convolution operation will not reduce the dimensions of the input, if that's undesirable. The reduction happens because convolution cannot be performed near corners. Padding effectively increases the dimensions of the image (e.g. 5×5 input with padding of 1 gives 7×7 image). For kernel size 3×3, padding of 1 and stride of 1 will preserve the dimensions of the input (e.g. 5x5 input with padding of 1 gives 7x7 image. After convolution with stride 1 the output's size is 5×5).

   YOLOv8's Conv layers use 3×3 kernels with padding of 1 and stride of 2 to reduce the dimensions of the input twice.

   Number of channels in Conv layer effectively refers to the number of kernels (also called filters). Each filter performs its own convolution and produces a separate output. Essentially, the "depth" dimension of the output tensor is equal to the number of channels (64 filters will produce 64 nxm images that will constitute a nxmx64 tensor). The convolution part is summarized in Fig. **3.2**.

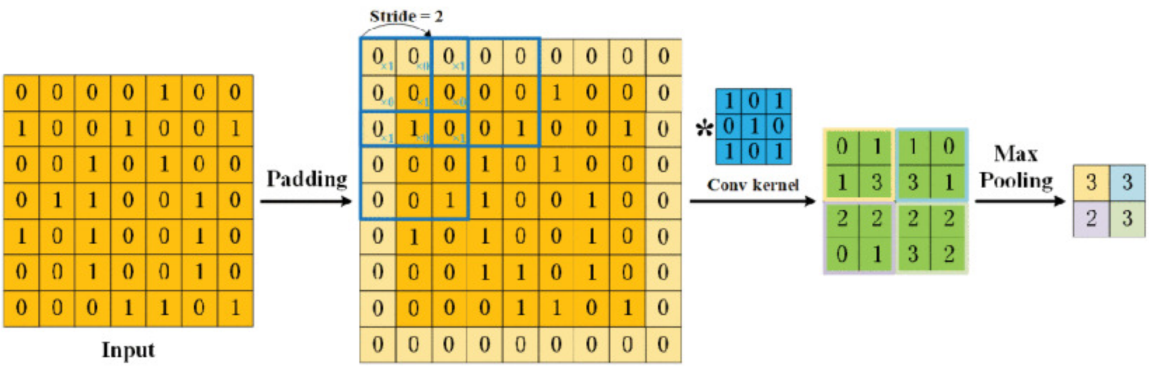

Figure 3.2: Summary of convolution, padding, stride and Max Pooling.

2. Second part of Conv layer is BatchNorm2d. This is a batch normalization layer. Its purpose is to normalize the values to certain boundaries. One of the reasons this layer is often used is because it is believed that batch normalization helps with vanishing gradients [16].

3. Last part of Conv layer is an activation function. The purpose of activation functions in neural networks is to introduce non-linearity into the model. The real world is highly complex and non-linear. Therefore, there is a need for the model that is capable of capturing these non-linear patterns in the data for better performance. Different functions can be used for activation, the popular ones include sigmoid, hyperbolic tangent, rectified linear unit (ReLU). YOLOv8 utilizes sigmoid linear unit (SiLU) function. SiLU was originally coined in [17] and later experimented with in [18]. SiLU is defined as: $silu(x) = x * \sigma$, where $\sigma(x)$ is logistic sigmoid given by: $\sigma(x) = \frac{1}{1+e^{-x}}$. The graph is presented in Fig. **3.3**.

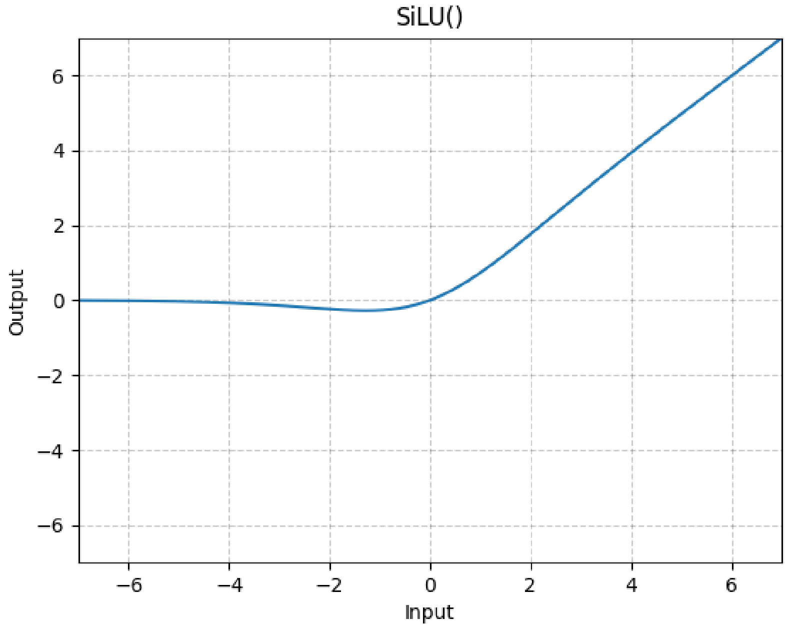

Figure 3.3: Sigmoid linear unit (SiLU).

C2f layer includes split, concat and botteleneck layers. Botteleneck layers consts of 2 Conv layers and if the shortcut is set to true, there is a skipping connection. Concat layer concatenates feature maps along the third dimension of the input tensors.

SPPF layer is a short for Spatial Pyramid Pooling - Fast. This layer includes 2 Conv layers and 3 MaxPool2d layers and there are splitting connections as well. MaxPool2d uses a kernel to slide over the image and pool the largest value thereby reducing the size of the input. The idea of the SPPF layer is to perform several max poolings reducing the size of the data

Upsample layer increases the resolution of the input tensor.

There are 3 detect modules. Detect modules are responsible for performing bounding box regression and class probabilities calculation. The idea of there being 3 modules is to detect objects at different resolutions. Large resolution is more suitable for detecting smaller objects, and smaller resolution is supposed to handle larger objects in an image.

The detect module has 2 separated paths, the design known as the decoupled head. One head is performing bounding boxes prediction, and another one is handling class probabilities prediction. It was shown, that decoupled head generally performs better than the head performing both tasks at the same time. For calculation of the bounding boxes loss YOLOv8 utilizes sum of complete intersection over union (CIoU) [19] and distribution focal loss (DFL) [20]. For class probabilities predictions YOLOv8 uses binary cross entropy (BCE) loss .

Additionally, YOLOv8 has a lot of parameters that can be tuned by the user for better performance. Some of the most important once will be described here.

1. Epochs. Epoch is one complete forward and backward pass of all the training examples. More epochs give the model more time to train and improve performance, although as described earlier, excessive training might result in overfitting.

2. Learning rate determines how much the weights and biases are updated. In YOLOv8 has several parameters to regulate learning rate. Most important ones are initial learning rate (lr0) and the final learning rate (lrf). Lr0 by default is set to 0.01. Lrf is a fraction of initial learning rate and by default its set to 0.01 such that the actual final learning rate is equal to $lr0 * lrf$. In this case, by default, final learning rate is 0.0001. The change from initial learning rate to the final learning rate follows a linear pattern. Additionally, there is a possibility to turn on cosine scheduler (cos_lr), so that the learning rate will follow a cosine pattern over the epochs.

3. Batch size is a parameter that determines how many images are processed before the weights and biases are updated. By default it is set 16. The model calculates the loss from these 16 images, then updates the weights and biases, and then proceeds to a the next batch. This goes on until the model goes through the whole dataset, i.e. one epoch.

4. Optimizer. This option chooses an optimizer for gradient descent. The options include stochastic gradient descent (SGD), Adam, AdamW, NAdam, RAdam, RMSProp. The default option is ”auto”. Optimizer affects convergence speed and stability.

5. Patience parameter is one of the measures to prevent overfitting. Patience helps by stopping the training early, if for certain amount of epochs, there was no improvement in metrics on validation dataset. By default, it is set to 100 epochs.

6. Dropout parameter, as described above, is another measure against overfitting. Dropout parameter sets the dropout rate, by default it is set to 0.

To estimate the accuracy of object detection and classification models, several metrics are typically used. Arguably, the most popular option is mean average precision (mAP), or sometimes just average precision (AP). This metric is related to metrics called precision and recall. Precision is a proportion of true positives to all predicted positives: $Precision = \frac{TP}{TP+FP}$. This metric shows how precise the model is when it predicts a positive result.

Intersection over union (IoU). Intersection over union quantifies the overlap between the predicted bounding box and the ground truth bounding box. $IoU = \frac{Intersection}{union}$. This number is from 0 to 1. By choosing different thresholds, it is possible to change what is considered a correct prediction. Recall is a ratio of true positives to all predictions. It is defined as $Recall = \frac{TP}{TP+FN}$.

Naturally, in most cases there is a trade-off between precision and recall, since for precision to be high, number of false positives needs to be decreased, but doing some may increase number of false negatives, and thus increase the recall.

Average precision is calculated using the so called precision-recall curve (PR curve). PR curve is obtained when precision is plotted against the recall. Then the area is calculated, giving a single value that encapsulates the model’s precision and recall performance.

mAP is obtained when average precision is averaged across multiple object classes. Sometimes, no distinction is made between mAP and AP. In these cases, it is usually assumed that AP is calculated across multiple object classes by default. Typically, mAPs are also calculated at different IoU thresholds.

3.3 Large Language Models

Large language models (LLMs) have achieved remarkable progress in natural language processing (NLP) in recent years. A major breakthrough in the field of LLMs is the implementation of the transformer architecture and its underlying attention mechanism, which have enhanced the models' capability to manage long-range dependencies in natural-language texts. Transformers utilize a self-attention mechanism to evaluate the importance of various parts of the input when making predictions.

Another significant advancement is the use of pre-training, where a model is first trained on an extensive dataset and subsequently fine-tuned for a specific task.

However, a major challenge is the lack of interpretability, making it difficult to understand the reasoning behind the model's predictions.

3.3.1 OpenAI's GPT-4.0

The next important part of the project is the use of a model that will generate the text. GPT-4.0 with vision was used to that end. GPT is a family of a famous text generation models. Generally, such models are trained on large amounts of text data in supervised fashion. Reinforcement learning with human feedback is then applied to improve model's performance. GPT-4.0 supports multimodal inputs in form of text or images. GPT models are created by OpenAI and are closed-source. Training data, the architecture and number of parameters are undisclosed.

OpenAI API allows users to tune several parameters. The most important ones are:

1) Temperature. Temperature parameter is a number from 0.0 to 2.0, by default it is set to 0.8. This parameter controls how creatively the model behaves. Raising the temperature leads to less predictable, less reproducible results propagating creativity of the model. Lower temperatures correspond to more consistent, deterministic behaviour. This parameter should be changed based on the desired reproducibility. 2) Max tokens. OpenAI's LLMs process text using tokens. The models learn to understand the statistical relationships between these tokens, and excel at producing the next token in a sequence of tokens. Max tokens parameter sets how many tokens can the response include.

Additionally, there is a possibility to set the behaviour of the model by passing a system prompt and asking the model to behave in a certain way.

Estimation of the performance of a text generation model is a challenging task. Often, the models are evaluated on standardized tests initially designed for humans (e.g. SAT, LSAT, AP exams, etc.). The drawback is that there are no standardized tests for many tasks. Another way is a simple human evaluation which is still arguably the most reliable way to evaluate model's performance, albeit requiring certain number of volunteers.

3.3.2 Anthropic's Claude 3 - Opus

Claude 3 Opus is the most capable Anthropic model achieving high results on various benchmarks such as undergraduate level expert knowledge (MMLU), graduate level expert reasoning (GPQA), basic mathematics (GSM8K). Additionally, the Claude 3 models have sophisticated vision capabilities on par with other leading models. They can process a wide range of visual formats, including photos, charts, graphs and technical diagrams

3.3.3 Google's Gemini 1.5 Pro

Gemini 1.5Pro is a recently developed LLM by Google. It delivers significantly enhanced performance. Gemini 1.5 Pro. was the first model that was released for early testing. It's a mid-size multimodal model, optimized for scaling across a wide-range of tasks, and performs on par with to 1.0 Ultra, the largest model by Google. Gemini 1.5 Pro comes with a standard 128,000 token context window. Gemini 1.5 is built utilizes Mixture-of-Experts layer (MoE) architecture. While traditional Transformer functions as one large neural network, MoE models are divided into smaller ”expert” neural networks.

4 The Results

The style suggestion system consists of two components combined into a single Python script: Ultralytics' YOLOv8 package and the GPT-4.0 model (gpt-4-vision-preview), which is accessed via the OpenAI API.

YOLOv8 was taught to distinguish ten types of clothing: sunglasses, hats, jackets, shirts, pants, shorts, skirts, dresses, bags and shoes. The training duration was set at 5 epochs. The performance metrics after 5 epochs of training are presented in Table 4.2. Metrics include precision, recall, mAP50 (i.e. mAP at IoU threshold of 0.5), mAP50-95 (i.e. averaged mAP at different IoU thresholds from 0.5 to 0.95). The model was validated on 537 images from dataset [6]. In total 2035 instances of objects were present. Metrics indicate that the model struggles the most with sunglasses, having the lowest recall and achieving the worst mAP50 and mAP50-95 for objects of this class. On the other hand, pants show the best metrics in terms of precision, recall, mAP50 and ,AP50-95.

The detection procedure begins when YOLOv8 recognises and crops the clothing pieces, which are subsequently saved in distinct folders. Following that, a request is made for the GPT-4.0 model. This request includes the following elements: the input image, samples of detected and cropped clothing pieces, text prompts describing the classes of these clothing items, a prompt asking for an opinion on the style, and a prompt requesting a style recommendation.

We have conducted live experimental trials at multiple events in Madrid and Tallinn. To demonstrate the outcomes, the results of the fashion advisor solution are depicted in here.

Fig. 4.1 illustrates a female in which three fashion items have been detected. The fashion advise by GPT model is as follows:

Your Style:
- You have a chic and winter-ready look.
- The jacket provides texture with its fur collar detail, adding luxury and warmth.
- A cohesive color palette of dark tones gives an elegant and streamlined appearance.
- The black opaque tights are a practical choice for colder weather and complement the black skirt well.
- The ankle boots are stylish and practical, perfect for a city environment.
- The skirt's pleats add movement and a feminine touch to your outfit.

Suggestions for Improvement:
- Consider accessorizing with a statement piece, such as a bold necklace or a vibrant scarf, to add a pop of color.
- Swap the black scarf for a lighter shade to create contrast and lighten the overall look.
- Experiment with patterned tights to add visual interest to your outfit.
- If functionality allows, a skirt with a slightly higher waistline could elongate your legs.
- Opt for a handbag with a bit more structure to juxtapose the soft lines of the jacket and skirt.

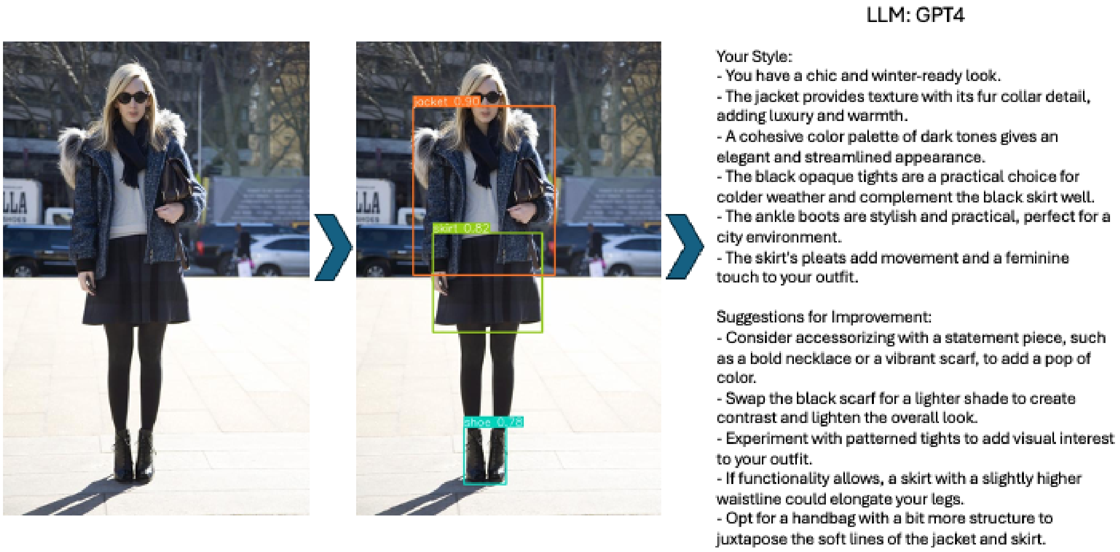

Figure 4.1: A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor.

Fig. 4.2 illustrates a female in which four fashion items have been detected. The fashion advise by GPT model is as follows:

- Relaxed and casual: Your style exudes a comfortable and effortless vibe, perfect for a sunny day out.
- Coordinated accessories: The straw hat, sunglasses, and brown belt add a cohesive and stylish touch to the ensemble.
- Earthy tones: Your choice of warm, neutral colors creates a harmonious and inviting palette.
- Proportions: The tucked-in top balances the volume of the high-waisted shorts, creating a flattering silhouette.

Improvement Suggestions:
- Top Selection: Consider a top with more structure or an interesting detail like a subtle ruffle or embroidery to add an extra element of interest to your outfit.
- Belt Upgrade: Switch the belt for one with a more distinctive buckle to serve as a statement piece and draw the eye.
- Footwear Variation: Try a pair of espadrilles or wedges for added height and a more streamlined leg line.
- Bag Contrast: Opt for a bag in a contrasting color to your outfit to introduce an unexpected pop of color and vibrancy.

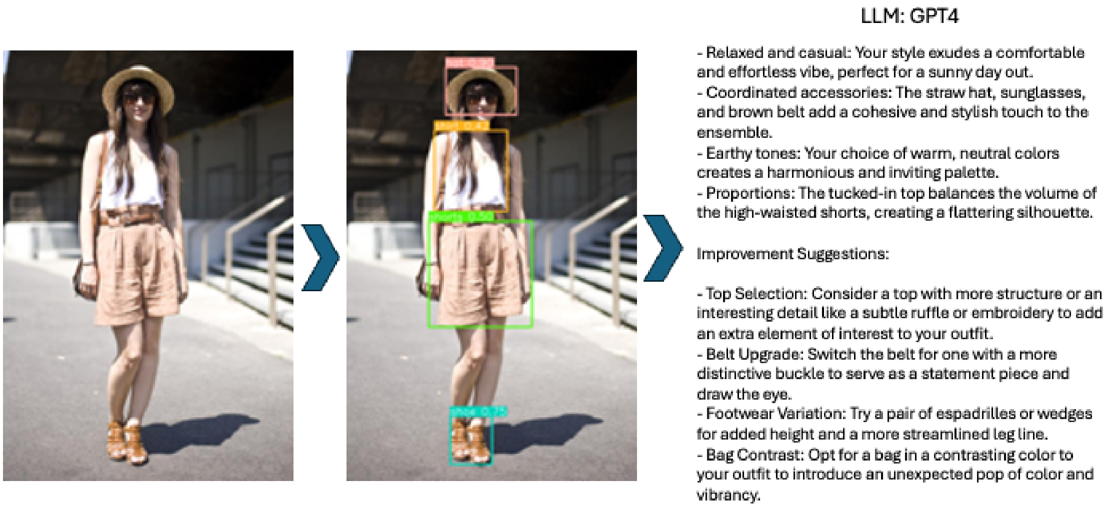

Figure 4.2: A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor.

Fig. 4.3 illustrates a male in which 3 fashion items have been detected. The fashion advise by GPT model is as follows:

- Tailored fit: Your blazer fits you well, emphasizing a structured silhouette.
- Color scheme: The combination of the blue check blazer with the brown pants creates a pleasing contrast that's stylish and modern.
- Layering: The addition of the patterned shirt under the blazer adds visual interest and depth to the outfit.
- Accessories: The lapel pin and pocket square are subtle details that show attention to style.

Style improvements:
- Hem pants: Consider hemming the pants for a cleaner break at the shoe, avoiding excess bunching.
- Shoe color: A darker shoe color could provide a stronger foundation for the outfit and create a more balanced look.
- Belt choice: Opt for a belt that complements the shoes to tie the outfit together cohesively.
- Shirt fit: Ensure the shirt is well-fitted around the waist to avoid any billowing fabric under the blazer.

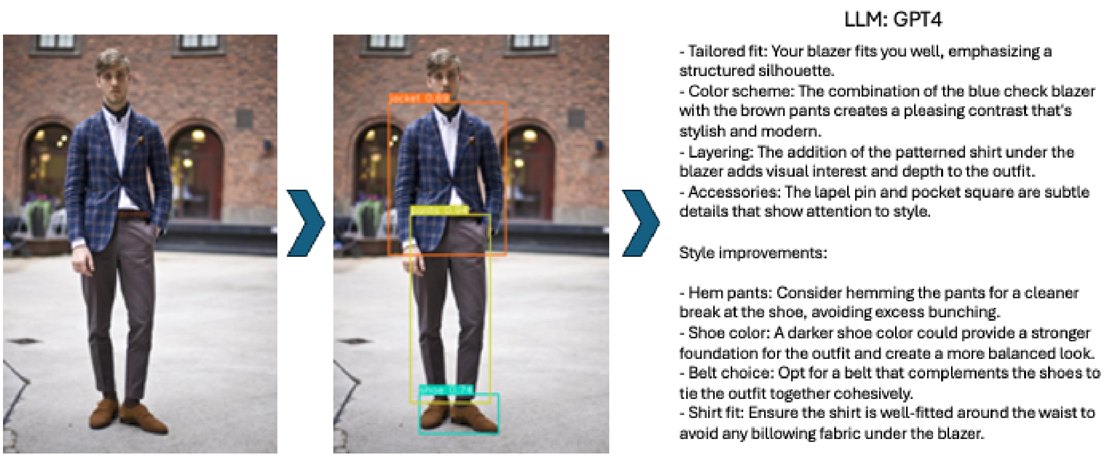

Figure 4.3: A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor.

To illustrate the differences between OpenAI's GPT-4.0 Vision, Google's Gemini 1.5 Pro, and Anthropic's Claude 3 - Opus, we conducted an experiment where each model generated fashion advice based on the same fashion image. An example of this experiment where a male model with detected 4 item is presented below.

Fig. 4.4 illustrates the result of GPT-4.0 Vision:

- Tailored fit: Your blazer fits you well, emphasizing a structured silhouette.
- Color scheme: The combination of the blue check blazer with the brown pants creates a pleasing contrast that's stylish and modern.
- Layering: The addition of the patterned shirt under the blazer adds visual interest and depth to the outfit.
- Accessories: The lapel pin and pocket square are subtle details that show attention to style.

Style improvements:
- Hem pants: Consider hemming the pants for a cleaner break at the shoe, avoiding excess bunching.
- Shoe color: A darker shoe color could provide a stronger foundation for the outfit and create a more balanced look.
- Belt choice: Opt for a belt that complements the shoes to tie the outfit together cohesively.
- Shirt fit: Ensure the shirt is well-fitted around the waist to avoid any billowing fabric under the blazer.

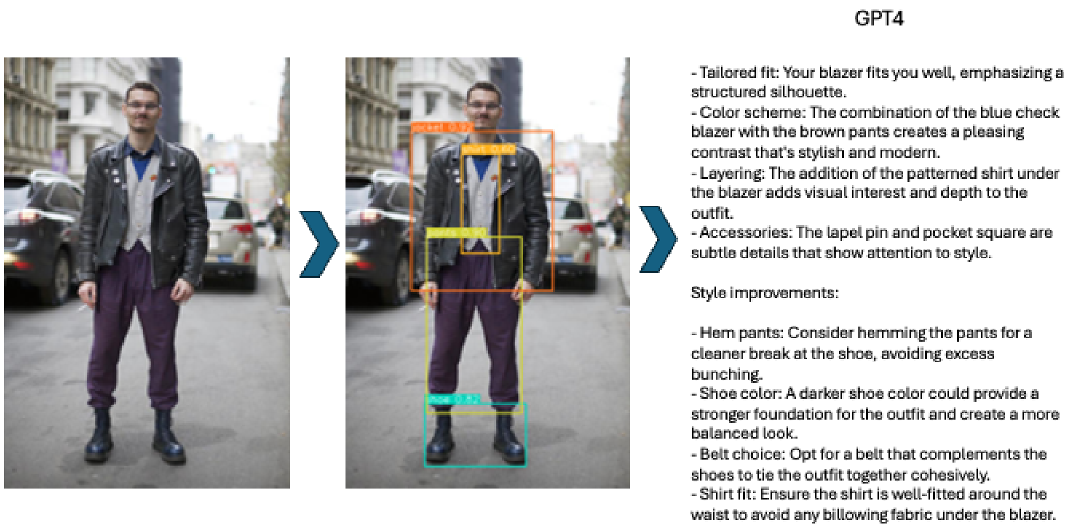

Figure 4.4: A random sample input, intermediate Yolo V8, and GPT-4.0 Vision based fashion advisor.

Fig. 4.5 illustrates result of Gemini 1.5 Pro model:

Style Comments:
Bold and Unique: Your style definitely stands out and has a strong personality.
Retro Vibes: The pants and jacket combination gives off a cool vintage feel.
Edgy Touch: The leather jacket and boots add an element of rebellion to the overall look.

Style Improvement Suggestions:
Pants Fit: The pants, while unique, could benefit from a more tailored fit. Consider a slimmer silhouette that tapers at the ankle. Example: Straight-leg chinos or tailored trousers in a similar pattern.
Shirt Choice: A crisp white shirt or a patterned shirt with a smaller print could create a more balanced look with the patterned pants. Example: A white Oxford shirt or a navy blue shirt with a subtle geometric print.
Vest Refinement: While the vest adds a vintage charm, a different color or pattern could elevate the outfit. Example: A charcoal grey vest or a subtly patterned vest in navy blue or dark green.

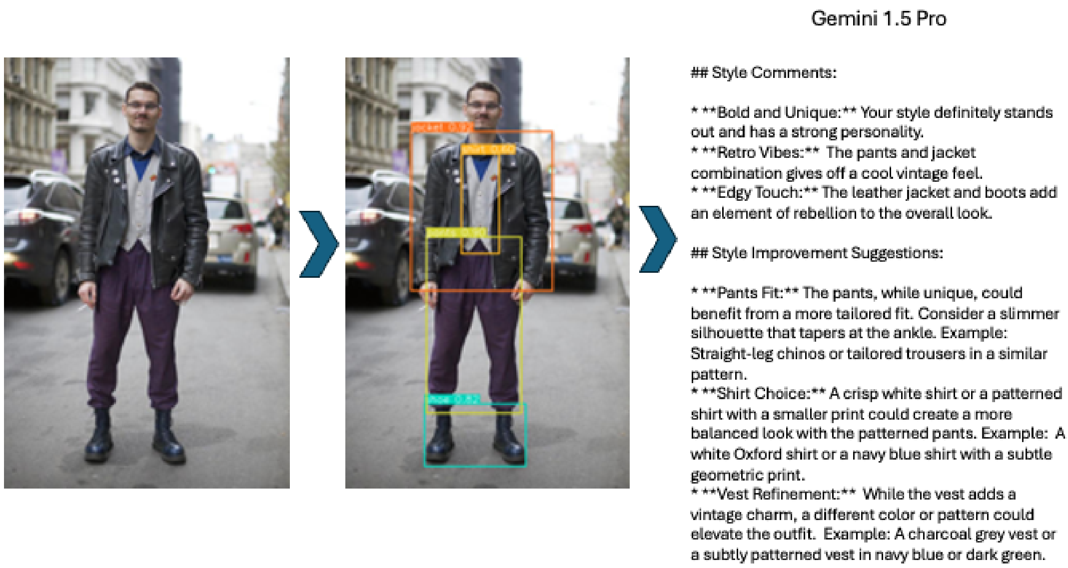

Figure 4.5: A random sample input, intermediate Yolo V8, and Gemini 1.5 Pro based fashion advisor.

Fig. 4.6 illustrates result of Claude 3 - Opus model:

Here are my brief comments on your style:
- The leather jacket, vest, and boots create an edgy, rocker-inspired look
- The purple plaid pants add a bold pop of color and pattern
- The light gray button-up shirt provides contrast against the darker jacket
- Overall it's a confident outfit that makes a strong style statement

Some suggestions for what could potentially be improved, with concrete examples:
- Consider swapping the plaid pants for a solid color like black jeans or trousers for a more streamlined look that lets the jacket be the focal point
- A crisp white shirt could further enhance the contrast with the black jacket
- Chelsea boots or lace-up boots in black leather would continue the sleek monochromatic theme
- If you want to include color, a jewel-toned shirt like emerald green or burgundy could complement the edgy vibe

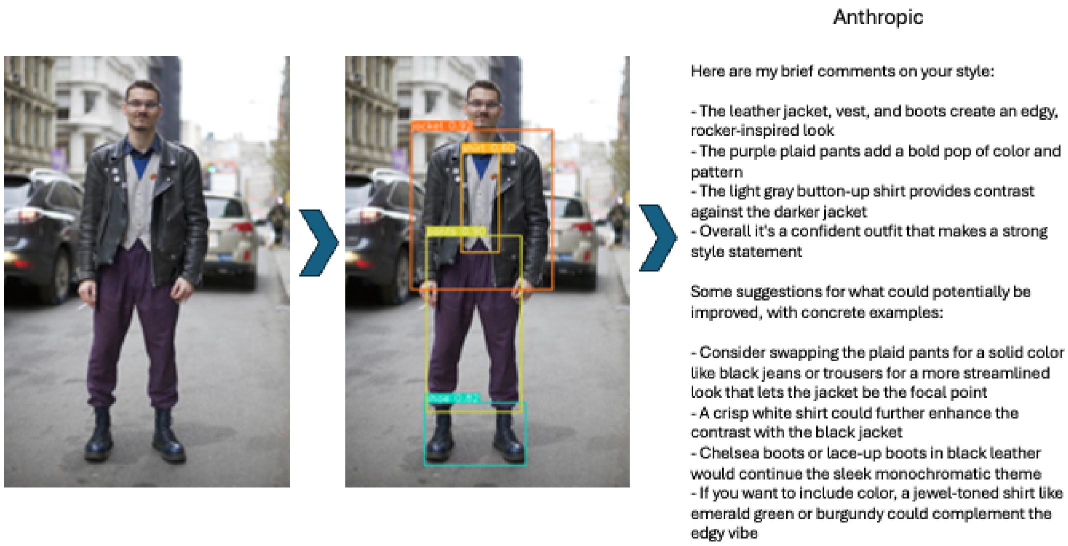

Figure 4.6: A random sample input, intermediate Yolo V8, and Claude 3 - Opus based fashion advisor.

The table summarizes the results of a comparative evaluation of three AI models: GPT-4.0 Turbo, Gemini 1.5 Pro, and Claude 3 - Opus, based on their fashion recommendation capabilities. Participants were asked to rate the quality of the recommendations on a scale from 1 (very bad) to 4 (very good) across three images, leading to a total of 12 recommendations. Eighteen participants provided feedback. The average scores across the models were as follows: GPT-4.0 Turbo and Gemini 1.5 Pro both achieved an average score of 2.8, while Claude 3 - Opus received an average score of 2.3. This indicates a slightly higher perceived quality of recommendations from GPT-4.0 Turbo and Gemini 1.5 Pro compared to Claude 3 - Opus.

Table 4.1: The average voting of participants for each LLM models. 1: very bad and 4: very good - Number of participants 18.

Images used	GPT-4.0 Turbo	Gemini 1.5 Pro	Claude 3 - OPUS
	2.8	2.9	2.4
	2.8	2.7	2.4
	2.7	2.7	2.2
Average	2.8	2.8	2.3

Table 4.2: Performance metrics for YOLOv8

Class	Images	Instances	Precision	Recall	mAP50	mAP50-95
all	537	2035	0.7	0.745	0.76	0.504
Sunglasses	537	82	0.6	0.0533	0.27	0.0945
Hat	537	77	0.685	0.805	0.795	0.471
Jacket	537	181	0.603	0.895	0.862	0.647
Shirt	537	366	0.798	0.762	0.823	0.555
Pants	537	114	0.886	0.904	0.955	0.719
Shorts	537	107	0.802	0.831	0.816	0.517
Skirt	537	186	0.659	0.855	0.813	0.623
Dress	537	128	0.603	0.812	0.761	0.584
Bag	537	274	0.664	0.708	0.712	0.391
Shoe	537	520	0.699	0.822	0.798	0.437

5 Conclusion

5.1 Conclusion

In this thesis, we created an advanced style recommendation system by combining cutting-edge computer vision and natural language processing technology. Using the YOLOv8 CNN, we successfully recognised and cropped clothing items from input photographs, which were then analysed using OpenAI's GPT-4.0 Vision model with vision capabilities. This system generated complete fashion advice, including analysis on current trends and personalised recommendations for improvement. The experimental study demonstrated the system's effectiveness, highlighting its potential to transform fashion advising using AI-driven insights. Furthermore, a comparison of GPT-4.0 Vision, Gemini 1.5 Pro, and Claude 3 - Opus revealed the respective merits of each model in terms of fashion recommendations.

5.2 Future Work

Future work will focus on extending the style suggestion system's capabilities to include real-time video analysis. By expanding the system to process live video feeds, we hope to provide quick fashion recommendations, making the service more dynamic and responsive. In addition, we will work on increasing fashion item segmentation in order to achieve more exact and detailed garment detection. This will entail improving the model's capacity to handle complicated patterns, textures, and occlusions, increasing the accuracy and quality of style recommendations. This improvement will not only enhance the user experience, but will also open the way for more widespread applications in fashion retail and personal style.

Bibliography

[1] M. Contributors, "MMYOLO: OpenMMLab YOLO series toolbox and benchmark." `https://github.com/open-mmlab/mmyolo`, 2022.

[2] "Global fashion retail market analysis." `https://tinyurl.com/4na63vma`. Accessed: 2024-05-27.

[3] J. Cychnerski, A. Brzeski, A. Boguszewski, M. Marmolowski, and M. Trojanowicz, "Clothes detection and classification using convolutional neural networks," in *2017 22nd IEEE international conference on emerging technologies and factory automation (ETFA)*, pp. 1–8, IEEE, 2017.

[4] N. Wang, H. Liu, Y. Li, W. Zhou, and M. Ding, "Segmentation and phenotype calculation of rapeseed pods based on yolo v8 and mask r-convolution neural networks," *Plants*, vol. 12, no. 18, p. 3328, 2023.

[5] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," 2023.

[6] S. Liu, J. Feng, C. Domokos, H. Xu, J. Huang, Z. Hu, and S. Yan, "Fashion parsing with weak color-category labels," *IEEE Transactions on Multimedia*, vol. 16, no. 1, pp. 253–265, 2014.

[7] Z. Liu, P. Luo, S. Qiu, X. Wang, and X. Tang, "Deepfashion: Powering robust clothes recognition and retrieval with rich annotations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1096–1104, 2016.

[8] X. Zou, X. Kong, W. Wong, C. Wang, Y. Liu, and Y. Cao, "Fashionai: A hierarchical dataset for fashion understanding," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pp. 0–0, 2019.

[9] M. Jia, M. Shi, M. Sirotenko, Y. Cui, C. Cardie, B. Hariharan, H. Adam, and S. Belongie, "Fashionpedia: Ontology, segmentation, and an attribute localization dataset," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pp. 316–332, Springer, 2020.

[10] S. Zheng, F. Yang, M. H. Kiapour, and R. Piramuthu, "Modanet: A large-scale street fashion dataset with polygon annotations," in *Proceedings of the 26th ACM international conference on Multimedia*, pp. 1670–1678, 2018.

[11] Y. Ge, R. Zhang, X. Wang, X. Tang, and P. Luo, "Deepfashion2: A versatile benchmark for detection, pose estimation, segmentation and re-identification of clothing images," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5337–5345, 2019.

[12] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

[13] Q. Huang, X. Han, T. Lu, and G. Liu, "Clothing image retrieval based on parts detection and segmentation," in *Proceedings of the 2021 3rd International Conference on Image Processing and Machine Vision*, pp. 53–59, 2021.

[14] J. Han, M. Kamber, and J. Pei, "Data mining concepts and techniques, third edition," 2012.

[15] M. Cilimkovic, "Neural networks and back propagation algorithm," 2010.

[16] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *CoRR*, vol. abs/1502.03167, 2015.

[17] D. Hendrycks and K. Gimpel, "Bridging nonlinearities and stochastic regularizers with gaussian error linear units," *CoRR*, vol. abs/1606.08415, 2016.

[18] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *CoRR*, vol. abs/1702.03118, 2017.

[19] S. H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. D. Reid, and S. Savarese, "Generalized intersection over union: A metric and A loss for bounding box regression," *CoRR*, vol. abs/1902.09630, 2019.

[20] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, and J. Yang, "Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection," *CoRR*, vol. abs/2006.04388, 2020.

Non-exclusive licence to reproduce thesis and make thesis public

I, Aleksandr Kalinin,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   "Generative AI-based Style Recommendation Using Fashion Item Detection and Classification"

   supervised by Mr. Akbar A. Jafari and Prof. Gholamreza Anbarjafari

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Aleksandr Kalinin
28.05.2024