
UNIVERSITY OF
SCIENCE
VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY

# A Seminar Report on

# PRODUCT LOOKUP AND RECOMMENDATION SYSTEM IN RETAIL INDUSTRY

In partial fulfillment of requirements for the degree of Bachelor of Data Science

SUBMITTED BY:

Nguyen Xuan Hai

Under the Guidance of

Dr. Tran Anh Tuan

# ACKNOWLEDGEMENT

This project would not have been possible without the support of many people. Many thanks to my adviser, Dr. Tran Anh Tuan, who read my numerous revisions and helped make some sense of the confusion.

# INDEX PAGE

# List of Figures

# List of Tables

# Abstract

The main aim of this work is to give a new solution to enhance customer's shopping experience followed by the boost of sales. In the advance of Deep Learning, especially Computer Vision, which can be applied in retail operations to replace the existence of barcodes while looking for product information, make it fast, automatic, and convenient. Due to the enormous number of product classes, we can perceive this problem as a fine-grained recognition, which requires a lot of constraint with normal approaches. The solution in this document is One-Shot Learning **[1]**, with this approach, we do not need many samples in every class, require less data in general and can easily handle new classes added in. The pre-processing is essential in this entire project since available data online are from other problems, in this case, product checkout problem **[2]**, which is similar but not the best for our specific problem. We have experimented with multiple methods on different stages of the system including data pre-processing, image detection, image comparison and recommendation to give the best performance in consideration of speed and accuracy.

# CHAPTER 1: INTRODUCTION

## 1.1 Motivation

Food is one of the most important factors that has a big impact on people's health. The relationship between food and the human body **[3]** can be compared with fuel and cars. The fuel is what interacts the most with a car's engine day by day, if the fuel quality is not good enough, the cars can easily be broken and have multiple potential risks. People have known this truth for a very long time ago but have only seriously considered it recently when the food industry exploded with the large number of products. We can say that nowadays the perception of people about health has improved and there is a consequence that they care about what they consume every day more than ever.

To take that opportunity, we develop a system that can help people lookup product information by simply putting the product in front of a camera, which can be equipped on every shopping cart. Besides, we integrate a recommender system to give suggestions in a natural way that can make people feel more comfortable while seeing those types of 'advertisements' and be willing to buy more products.

## 1.2 Problem Description

From across the world, there are several attempts to apply the power of Computer Vision into retail operations, for example, checking SKUs of products **[4]** which can help staff of the store manage the number of products on shelf faster and easier, automatic product checkout **[2]** replace the present of traditional checkout fashion and reduce operating cost, … In Vietnam, there are very few, if any, applications made to improve the retail businesses. So we decided to develop a system that can solve a relatively new problem in retail about checking product information.

A device for checking product information is not a strange thing in some big retail stores in Vietnam, but it has the 'barcode problem'. Normal customers barely take time to find the barcode on the product and sometimes cannot find it, this can be considered annoying and a big wall which stops them from gaining access to the information. Understanding those limitations, we make this document to present a new approach for retail stores to remove the inconvenience, increase customer's shopping experience, let the customer be more proactive to know what they buy, and store owners can take advantage of to make more revenue.

## 1.3 Organization

The rest of the report is organized as follows,

• In Chapter 2, we discuss about our problem with relevant works and challenges, then we propose our approach.

• In Chapter 3, we discuss and considerate different methods for every process to choose the best ones.

• In Chapter 4, we give an overall view on every process and estimate the performance of techniques.

# CHAPTER 2: LITERATURE REVIEW

## 2.1 Related Works

Up to this time, there have been a lot of problems having a similar approach with our specific problem, such as face detection, vehicle identification, product management, etc. All these problems have the same issue that the number of classes needed to predict, is too large or may be unknown. If we use a normal classification approach, the need for data is a big problem, not to mention the scalability of the model when new classes are added.

In **[2]**, the authors proposed a new retail product dataset, which has a large amount of product images taken from different angles. They claimed that the dataset can easily dominate every earlier existing product dataset such as GroZi-120 by the number of samples. Based on the dataset, they provided solutions to design an automatic product checkout function and integrated available data with Cycle-GAN **[5]** to generate new images with different numbers of objects and image environment cases.

The most similar work to our problem can be found in **[6]**, they proposed a new recognition system based on the PaddlePaddle platform **[7]**, which can perform effectively on different recognition scenarios, including product recognition. A noticeable thing that can be seen in this paper is that they can balance the trade-off between accuracy and speed. The system can be deployed on every device, providing accurate real-time recognition results.

We can observe the same approach from our examples about retail products above, generally the authors will localize and cut the products out then compare them with each image class in the available database to identify the products.

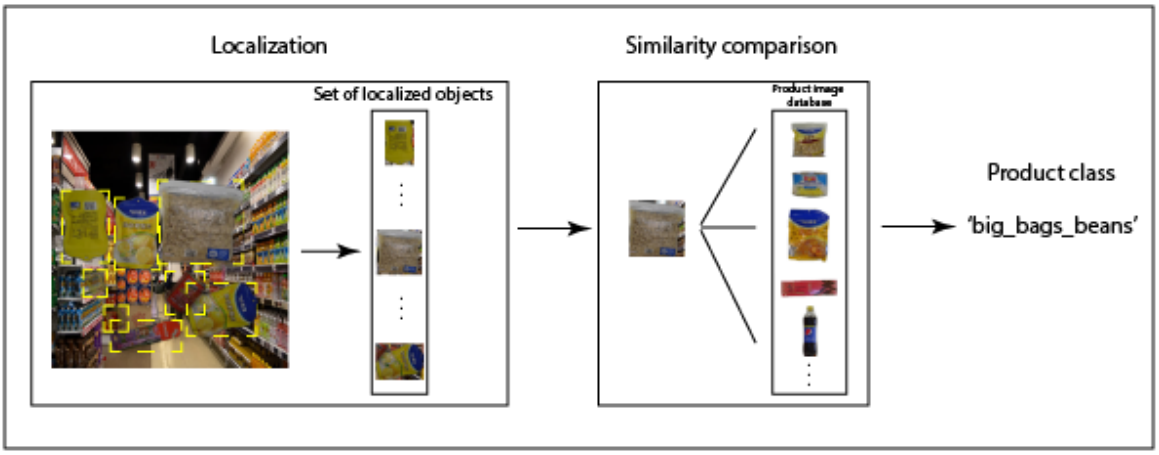


**Figure 1.** General structure of related problems

## 2.2 Challenges

The common obstacle of every product-related problem is that there is an enormous number of categories and sub-category of products. To precisely determine a specific product, we need to know almost every information such as product type, product version, flavor, manufacturer, etc. Each of the information combines together to define a distinct class and the number of possible combinations is definitely large.

Another big challenge of the problem is that there are relatively few people who do research and develop this system, leading to the missing of real data that should be obtained with the right requirements. Therefore, we focus on improving the quality of data, trying to use available data to generate more data that can replicate the common conditions in every retail store.

## 2.3 Proposed Approach

As every fine-grained recognition problem, we choose One-Shot Learning approach **[1]**, which is the most common approach for this type of task. One-Shot Learning contains 2 steps: localize the objects and compare similarity with product pictures in the database. Following by the idea of One-Shot Learning, we construct a lookup and recommendation system with 3 main parts: a detector for localizing every product that exists on the frame, a feature extractor to select crucial details that can represent cropped product images for comparison and a recommender for giving recommendation with identified product class.

We considered different augmentation to make the detector work as best as it can on several background and environment scenarios. With the data background, we use rembg Python library which utilize the structure of saliency detection model $U^2$-Net **[8]** to extract the foreground from every image and randomly combines them with a set of in-store backgrounds. To solve the problem of image quality and other aspects that might reduce the performance of the model, we have tried some traditional methods such as blur, scale, translate, etc. and SOTA augmentations including Mixup and Mosaic **[9]**. All of these augmentation can be modified easily by using YOLOv8, which was developed by the Ultralytics team **[10]**. Not only augmentation, but the development team has also integrated a lot of useful utilities. We can set up data pipelines, fine-tune and use several techniques on the model just by a few lines of code.

After several attempts to preprocess and fine-tune, we have a model that can successfully identify the presence of products in many difficult scenarios. The output then will be turned to a set of keypoints by ORB algorithm **[11]**, which is similar to the process of feature extraction. These keypoints can be used to compare with every keypoint representation of image classes in the database.

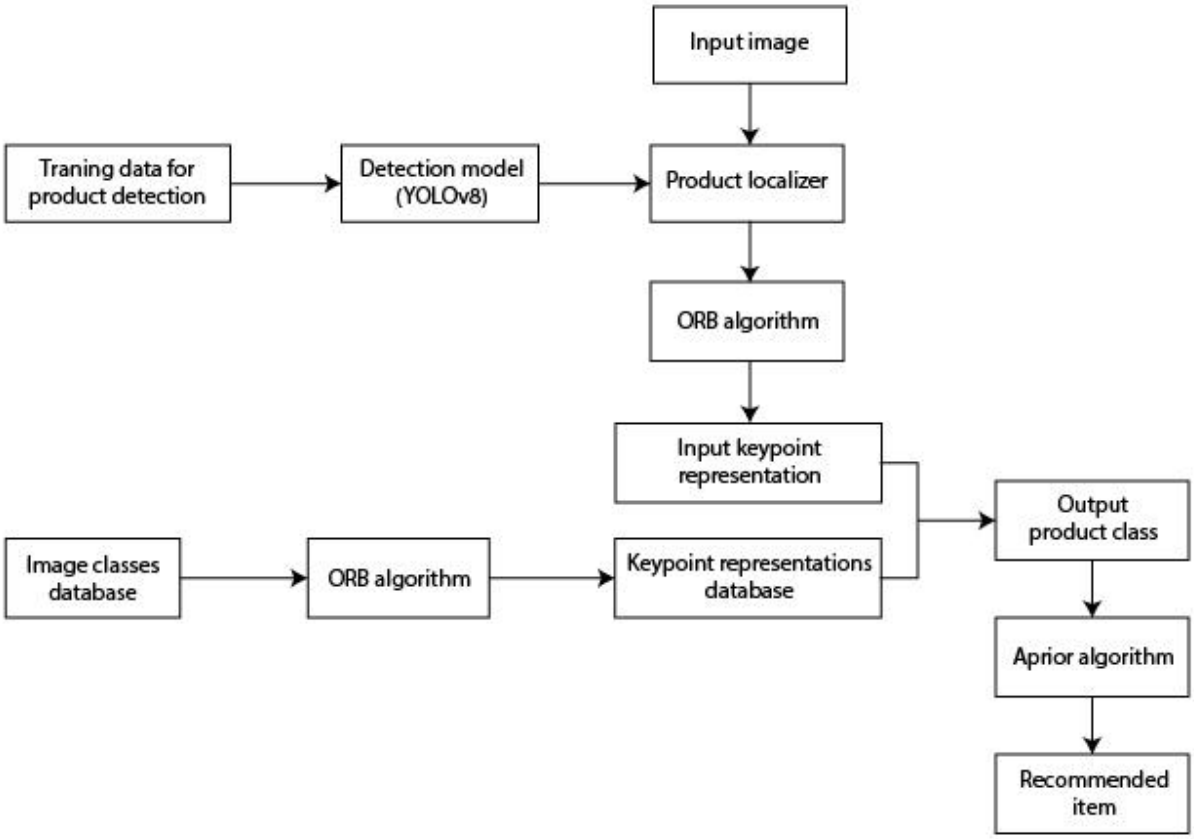


**Figure 2.** Overview of the system

## 2.4 Contributions

We proposed a new problem and solution to optimize retail operation:

- Problem of finding product information with recognition system.
- Solution for a system that can help store owners attract customers and increase in-store shopping experience.

following with the idea of handle inappropriate dataset.

# CHAPTER 3: METHODOLOGY

## 3.1 Data Pre-processing

To increase the performance of the detection model, we want to generate new data that can replicate the closest in-store captured images including image quality, image background and light condition. We have researched and tried several methods to overcome this challenge. We believe that these approaches can be possible solutions for our system.

The first problem we try to solve is changing the background of image data. GANs **[12]** are a very useful tool for synthesizing realistic data and have an important role in many data augmentation processes. The general structure of the model contains 2 main parts: Generator and Discriminator. In a nutshell, the Generator will try to generate the most realistic images and the Discriminator will try to figure out the generated images are fake. There are many GANs that can effectively solve the problem of creating new data with different backgrounds **[13, 14]**. These models can separate every foreground and background existing in the original dataset, automatically combine them together and synthesize the most realistic pictures. Despite the usefulness of GANs, there is no available image with in-store background which is require for learning the correlation of foreground and background in the dataset. Another approach that we have considered is object segmentation, which is a very powerful method for extracting objects properly and automatically. The structure model for this method varies depending on a lot of aspects such as: computational cost, infrastructure, performance, accessibility, … Normal approaches for image segmentation were often trained on a specific problem and require labels to perform transfer learning in a different task, unfortunately, our dataset that we are processing does not have segmentation labels. In the end, our proposed method for this problem is Saliency Object Detection, which is still an object segmentation approach but does not require any target label to train on and can be used for multiple purposes.

$U^2$-Net **[8]** is a novel model structure which was developed from the original segmentation model U-Net **[15]** to provide a solution for detecting saliency objects. The overall form of the network is basically the same as U-Net. The only big differentiation is from the nodes, while U-Net using the normal convolutional structure, $U^2$-Net using another U-structure, the authors called it ReSidual U-blocks (RSU). Although the structure of RSU is quite complicated, the computational cost of the network in comparison with the former architecture is not very different.


**Figure 3.** The result after applying saliency detection $U^2$-Net and background changing to the dataset. (a) The original images. (b) A simple visualization of saliency map. (c) Background removed images. (d) New dataset with different in-store backgrounds

To perform the changing background task on the dataset, we have utilized the advantages of Python library *rembg* **[16]**, which is a tool specifically born to solve the problem of image background removal. After a few attempts, we can say that the library performs well on the dataset, especially when the image contains a lot of objects. However, as the limitations of the dataset, the accuracy of the process can not be compared with normal object segmentation approaches. In **Figure 3**, we can observe that the method struggles to determine non-object space among different objects and sometimes partly or cannot identify some objects. Although this problem is worth considering, but it does not significantly affect the system performance since the only goal of the dataset is to provide materials for the detection process to localize every potential product appearing in the images.

Now there are still some objective problems which can possibly come up while deploying the system in real life. One of the aspects that can hugely impact the performance of the model is input data quality and light condition of the environment. With

augmentation techniques like Blur, AdvanceBlur or MotionBlur from the albumentations library **[17]**, we only can partly solve the problem of low-quality input since every object recognition system needs to process every detail in the image. On the other hand, light condition is a more common problem that exists in every image processing task. Several methods have been proposed, such as applying random shadow and highlights **[18]**, performing different levels of global and local illumination changes on images **[19]**. We have taken advantage of these two methods and use them respectively with the belief that the model will be invariant with light conditions.

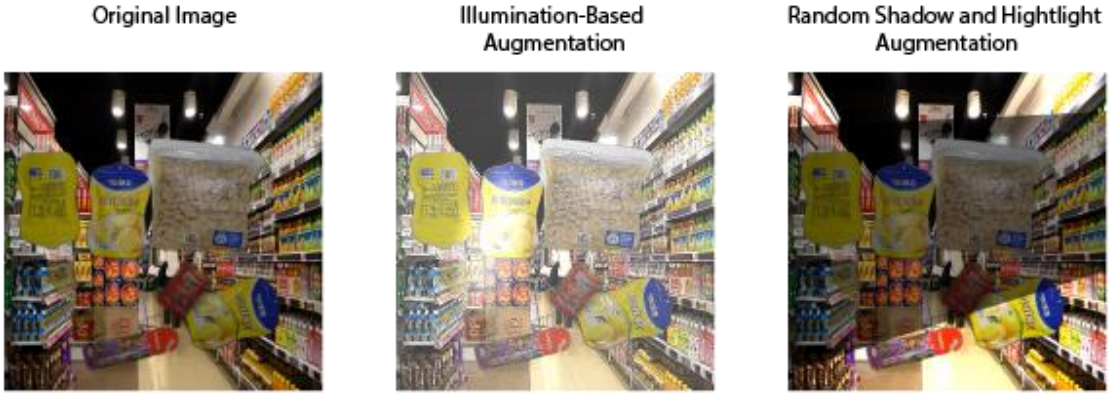


**Figure 4.** Example result of the two augmentation methods

Illumination-based augmentation includes 2 processes: local changes and global changes. Local changes will apply a random characteristic point on the image and global changes will determine the overall amount of light on the image. The method synthesizes local changes by some following simple steps. Firstly, randomly choose a coordinate on the image $p = I(x_0, y_0)$ and a diameter of the point circle, which depends on image size $d = k \times min(ImageWidth, ImageHeight)$. After that, we can specify a simple binary mask $M_1$ of the circle by the following formula:

$$M_1(x,y) = \{1, \quad if\ (x-x_0)^2 + (y-y_0)^2 \le d^2 \quad 0, \quad if\ (x-x_0)^2 + (y-y_0)^2 > d^2$$

To make the light (or shadow) point be more realistic, the mask then was transformed by Euclidean Distance Transform (EDT), which can be briefly described as a technique used to find the nearest distance from the foreground binary mask boundary of every pixel in the image. The result of the transform $M_2 = EDT(M_1)$ is that it makes the binary mask of the circle have a gradient-like effect. Finally, the synthesized image $I_s$ is created by randomly pixel-wise add (create lamp-post effect) or subtract (create shadow effect) the original image $I$ by the mask $M_2$:

$$I_s = I \pm (M_2 \times z), \quad z\ is\ a\ random\ integer$$

Similar to the final step of local changes, global changes increase or reduce the overall light of the image by the formula:

$$I_s = I \pm z, \qquad z \ is\ a\ random\ integer$$

Then the final combined formula of the Illumination-based augmentation will be:

$$I_s = I \pm z_1 \pm (M_2 \times z_2), \qquad z_1, z_2\ are\ random\ integers$$

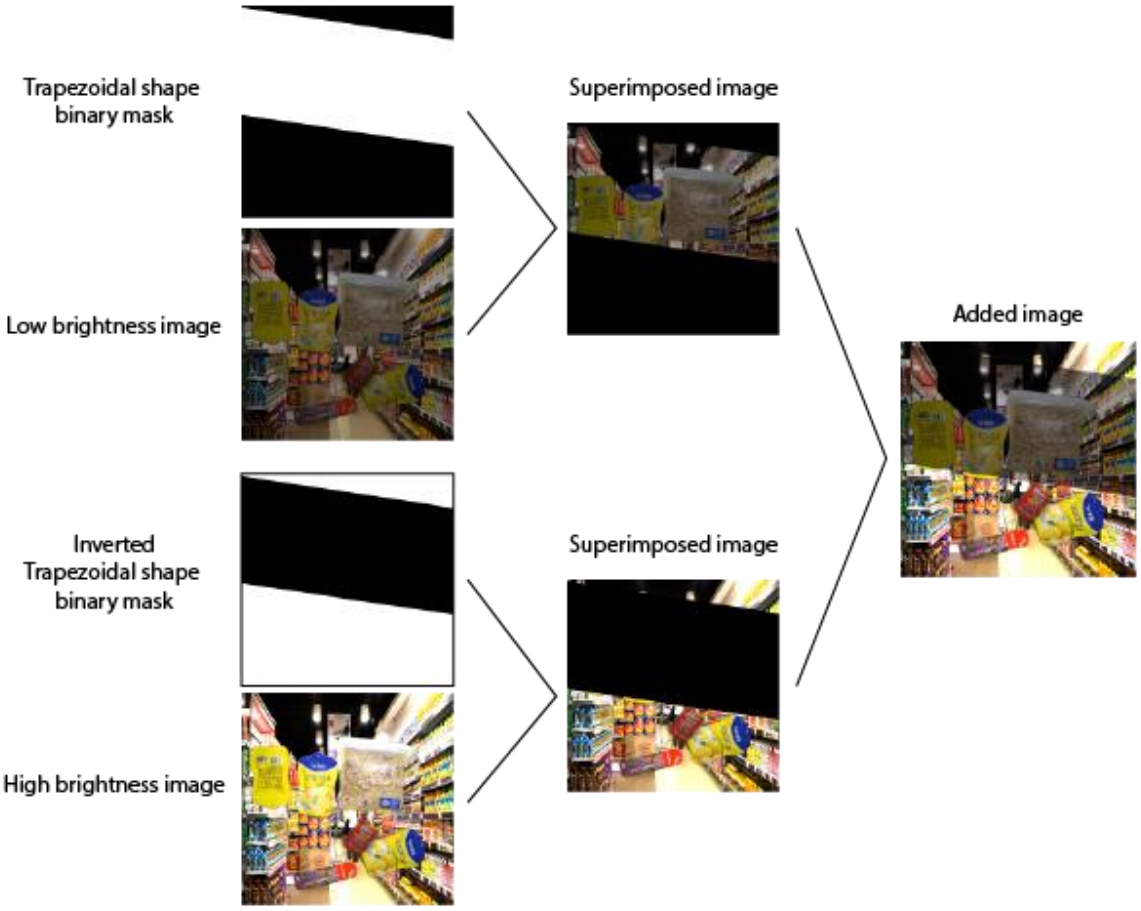


**Figure 5.** Random Shadows and Highlights method workflow

The authors of Random Shadows and Highlights augmentation method realize that the shadows in images usually appear in the shape of trapezoids. Therefore, the method randomly generates a trapezoidal shape mask and an inverted one. The trapezoidal shape mask then will be combined with a low brightness copy of the input image to simulate the shadow zone. On the other hand, the binary inverted mask will superimpose the high brightness image to represent the rest of the resulting augmented output. **Figure 5.**

## 3.2 Localization

The development of Computer Vision was rapidly grown in the 20s century and have been continuing to improve in recent years. With the higher and higher requirement, the researchers have surpassed many limitations in the need of accelerating model prediction speed to apply in many real-time systems. To take advantage of these achievements, we have experimented with several newest inventions to solve our own problem, which is localizing products that appear in the input images at real-time processing speed.

The YOLO detection model series **[20]** was first introduced in 2015 with the presence of the original YOLOv1 **[21]**. From the beginning, the framework has been standing out as the most popular approach for solving real-time object detection task, which requires the perfect balance between speed and accuracy. Over time everything changes, the YOLO still evolves and releases new versions to address earlier version's limitations and enhance the performance. In this document, we will discuss the two latest YOLO versions, which are YOLOv8 **[10]** and YOLO-NAS **[22]**.

YOLOv8 is a latest version of the traditional YOLO image processing approach, which was released in early 2023. YOLOv8 was developed by Ultralytics team, who also created the influential and industry-defining YOLOv5 **[23]**. Based on the idea of YOLOv5, the developers have experimented with different architecture changes and improvements to announce a new YOLO version which is the most efficient in real-time detection task at the time.

As shown in **[10]**, YOLOv8 has a significant enhancement in consideration of both speed and accuracy to every type of former real-time object detection model. Like every state-of-the-art YOLO version, the model has the Feature Pyramid Network (FPN) **[24]** structure with three heads for multi-scale object detection, but still there are some differences in the architecture when we compare it with the former version developed by the same author. A big difference that can easily be noticed is that the CSPLayer in YOLOv5 was replaced by the C2f module, which has a bigger impact on the output prediction since it combines high-level features with contextual information in the input image. Another noteworthy point of this YOLO version is that it performs anchor-free detection tasks, which was firstly applied in the YOLO series by YOLOX **[25]**. This solution makes the algorithm faster since there are less prediction boxes in Non-Max Suppression (NMS) post-processing and helps handle anchor box's size mismatching between training dataset and custom dataset.

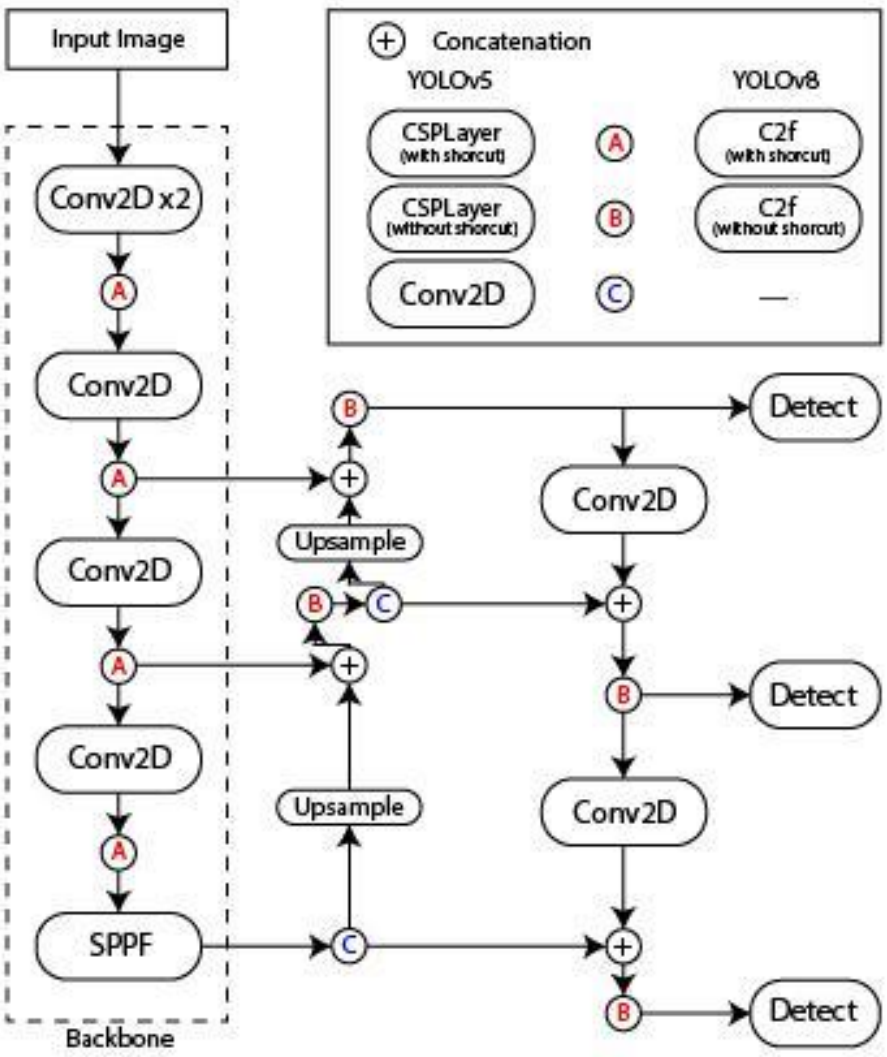


**Figure 6.** Difference between YOLOv5 and YOLOv8 architecture

Not a long time later, in May 2023, a novel model for real-time detection, YOLO-NAS **[22]** was released. The model has the common ideal of YOLO and further integrated with a Neural Architecture Search system **[26]** called AutoNAC **[27]**, which is an exclusive property of the authors Deci AI, to automatically find a suitable architecture that can balance latency and throughput. Moreover, the algorithm also applies re-parameterization and 8-bit quantization. Both methods aim to optimize processing speed by reducing inference time, but there is an enormous accuracy loss appearing when simultaneously using them. To successfully take advantage of the two methods, the architecture was implemented with quantization aware modules **[28]** called QSP and QCI. The last notable point about the algorithm is the strategy of the quantization process, which tries to only quantize some parts of the model in the expectation of finding the most efficient trade-off point between processing speed and predicting performance.

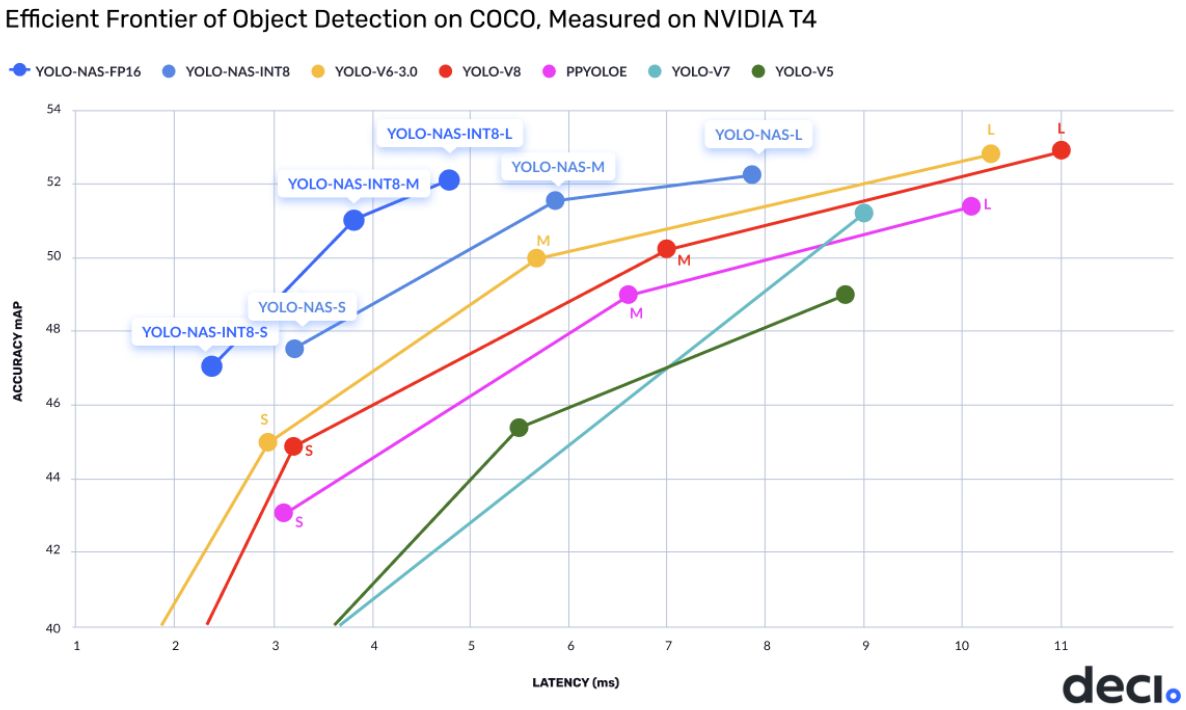


**Figure 7.** YOLO-NAS performance in comparison with other real-time detectors announced by Deci AI

In recent years, the boundaries between different model capabilities are increasingly unclear. An approach which was originally developed to solve a task can be used as a solution for another task, there is no limitation. The popular approach that we want to mention in this localization problem is Transformer **[29]**, which was developed in 2017 firstly for the Natural Language Processing task. The authors of this algorithm have introduced a very interesting approach, which is self-attention, that makes the model have the ability to simulate the way humans process and understand the meaning of a certain text string. The general idea of the authors is that they want to develop an architecture which can give weight to every word in the text string to figure out the correlation among surrounding words and contexts. Throughout the time, people realize that not only words in text strings need 'attention', but also image data. Apparently, common Computer Vision algorithms use a backbone to extract various features from the input image then propagate these through certain convolutional-based architecture to produce an optimal output result. After the existence of the self-attention mechanism, people considered it as a new potential method to map from a set of features of the backbone to the output prediction. The application of Transformer architecture in Computer Vision creates a revolutionizing approach called Vision Transformer (ViT) **[30]**.

RT-DETR **[31]** was announced by Baidu as the first real-time transformer-based object detector which is one of several variants of the original DERT **[32]**. As we have known, YOLO architecture depends heavily on post-processing and various hand-craft components, this is unfortunately a burden and negatively affects model performance. On the other hand, DERTs is an end-to-end architecture which directly gives bounding boxes prediction without the need of any additional processes and resources. To invent a new approach that can effectively outperform any recent real-time detector, the authors take this advantage and do further experiments to eliminate limitations that exist in other DETRs which make them unable to achieve real-time performance. In comparison with other state-of-the-art real-time architectures that have the same scale, the model produces an outstanding result and outperforms them in both speed and accuracy.

Firstly, the authors proposed an Efficient Hybrid Encoder, which optimizes the accuracy and removes the redundancy in computation. In previous versions of DETR, the authors can observe that the encoder block usually processes a set of features from the backbone by concatenating multi-scale features or simultaneously processing intra-scale and cross-scale features, which is computational inefficient because of duplication of information across feature scales. To overcome this challenge, they have tried several variants and finally proposed a new architecture for their transformer encoder.

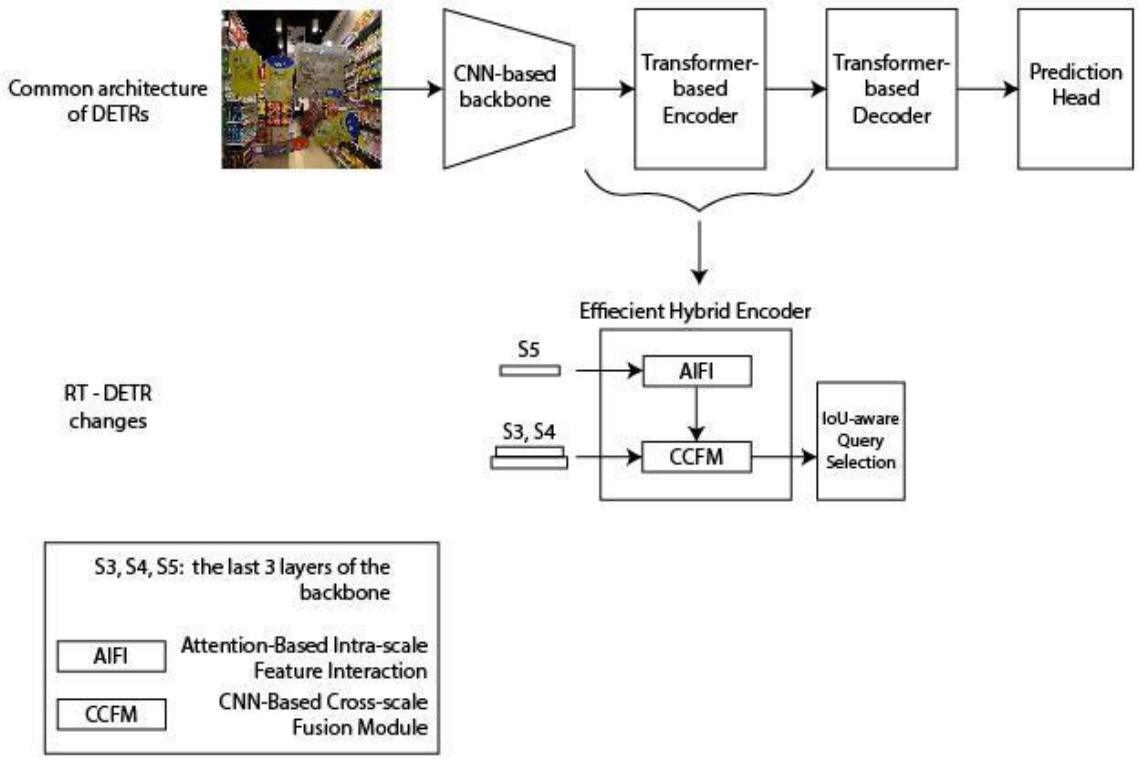


**Figure 8.** Optimization of RT-DETR on model architecture

Efficient Hybrid Encoder includes two parts: Attention-based Intra-scale Feature Interaction (AIFI) and CNN-based Cross-scale Feature-fusion Module (CCFM). This encoder will receive the last three layers {S3, S4, S5} of the backbone and use S5, which has the richest semantic concept as input of AIFI to find connection and correlation among conceptual entities in the image. Output of the module is an array which will be reshaped as a 2D matrix to be synchronized with S3, S4 and be treated as input of CCFM. CCFM includes several fusion blocks and has an important role that combines features from multi-scale inputs and produces a new valuable feature set.

Another interesting thing about the algorithm is that they apply an IoU-aware mechanism **[33]** into the query selection module called IoU-aware Query Selection. As every normal query selection that exists in DETRs, the IoU-aware Query Selection chooses top K features from the feature set exported from the encoder. The thing that differentiates it from normal query selection is the choosing strategy, which requires both accuracy in classification score and box confidence while vanilla sometimes keeps predictions with high class confidence and low IoU score.

### 3.3 Image Similarity

Auto-Encoders (AEs) **[34]** is one of the most popular architectures in the field of image processing. In the modern world, data comes with high volume and velocity. To successfully handle this enormous amount of information, we need to find a way that can compress data into some sort of representation. By using this way, multiple costs for processing the data are reduced and the compressed data can be easier to be used for multiple purposes. In Computer Vision, AEs is the one that was specifically born to undertake this task.

The structure of AEs is simply defined by an encoder followed by a decoder, the encoder will take the input of a given 2D image shape and transform them into an array, the decoder will use the array and try to recreate an image which have the most similarity with the input image. This structure aims to make the model extract important features, which are good enough to be regenerated again, from the encoder.

For image similarity tasks, we need AEs to turn the input image into a vector that can represent it in lower dimensional space to perform finding neighbors at efficient computational cost and time. Unfortunately, after several attempts to use, we observed that this approach has not achieved the expected performance and is robust to input shape variation which significantly impacts the encoding quality of the extracted feature set. The cause of scale variation in this problem comes from the fact that product pictures cropped from real case scenarios have different shape and size depending on the capture distance. To overcome this obstacle, we consider another method that can effectively solve the problem with high performance.

Feature matching using keypoint descriptors was firstly proposed in 2004 with SIFT algorithm **[35, 36]**. SIFT, Scale-Invariant Feature Transform, operates on the idea that it finds a certain number of keypoints on the image, each keypoint is described by a descriptor that records information in the surrounding pixels. When being used in certain scenarios, set of keypoints and descriptors of an image will be compared with another image's one to recognize the similarities, thanks to this mechanism, the difference of image size and orientation between the two does not significantly affect the accuracy in comparing.

To determine the location of keypoints, the algorithm performs blob detection in the image. Given the Gaussian $G$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$$

The scale-normalized Laplacian of Gaussian is defined as

$$LoG = \sigma^2 \Delta^2 G,$$

when being convolved with input image $I(x, y)$, it can generate a highlight map of edges that appeared in the image, if we choose the right $\sigma$, those edges will spread out and turn to blobs. Unfortunately, the computational cost of LoG is large, thus we need to find a less complicated solution that still performs well, and it is different-of-Guassian (DoG). DoG is a method which try to approximate the LoG by subtracting two Gaussians and is defined as follow:

$$DoG = G(x, y, k\sigma) - G(x, y, \sigma),$$

Then the highlight map (DoG image) is computed as

$$DoG * I(x, y) = G(x, y, k\sigma) * I(x, y) - G(x, y, \sigma) * I(x, y)$$

To successfully calculate DoG at several levels of Gaussian, the algorithm constructs a scale space with several octaves, each octave stacks several same scale layers of $G(x, y, k^n\sigma_0) * I(x, y)$. In these octaves, for every chosen two adjective layers, the layer with greater $\sigma$ will subtracts the smaller one to produce a DoG image.

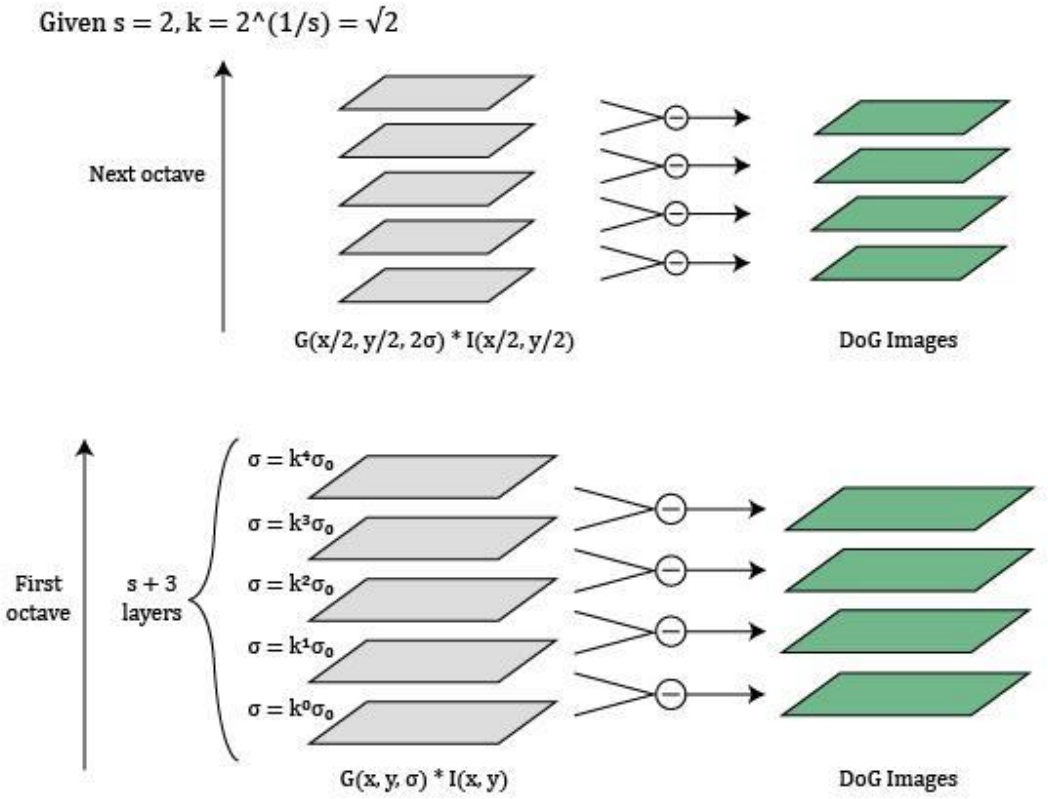


**Figure 9.** Scale Space architecture for computing DoG images

After that, we can detect extrema of DoG images by comparing a certain pixel to its 3x3x3 neighbor region, which contains 8 neighbor pixels in the same scale and other 18 neighbor pixels in adjective scales. However, these extreme points are just candidates for generating the final set of keypoints since they contain lots of noise. The noise here is a general idea of points that have a weak ability, which was judged by considering nearby data for location, scale, and ratio of principal curvatures, to represent object features in the image through various conditions and scenarios.

Now that we have sets of keypoint located in the image and make the algorithm invariant of scale, next we have to calculate orientations and descriptors of each keypoint by considering magnitudes and orientations of surrounding pixels.

For computing a keypoint orientation, a set of neighbor pixels will be chosen. With every pixel in the set, calculate the orientation and magnitude using 4 adjective pixels. Given $M$ as the magnitude, $\varphi$ as the orientation and

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y),$$

we have

$$M = \sqrt{Different_x(L)^2 + Different_y(L)^2}$$

$$= \sqrt{[L(x+1,y,\sigma) - L(x-1,y,\sigma)]^2 + [L(x,y+1,\sigma) - L(x,y-1,\sigma)]^2},$$

and

$$\varphi = \frac{Different_y(L)}{Different_x(L)}$$

$$= tan^{-1}\frac{L(x,y+1,\sigma) - L(x,y-1,\sigma)}{L(x+1,y,\sigma) - L(x-1,y,\sigma)}$$

By using these formulas, we can determine the orientation and magnitude of every neighbor pixel and then transfer them to histogram with 36 bins corresponding to 360 degrees to select the orientation which has the largest magnitude weight and use it as a representation of the keypoint orientation.

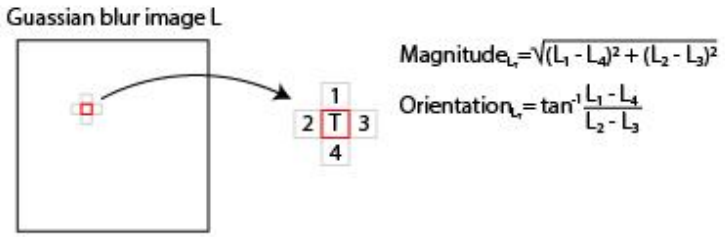


**Figure 10.** Demonstrate on calculating orientation of a certain pixel.

Last but not least, the keypoint descriptor is an important part which makes the algorithm identify any keypoint and have the ability to compare features from image to image. Descriptors in SIFT consider a 16x16 neighbor region around a keypoint which is weighted by a Gaussian circle window to reduce the impact of far pixels, then based on that, we perform calculation on pixel orientation. After finishing calculating, all the orientations and magnitudes are used for another histogram with 8-bins. Each 4x4 pixel in the 16x16 region will construct an array containing 8 values representing magnitude of 8 directions of orientation.

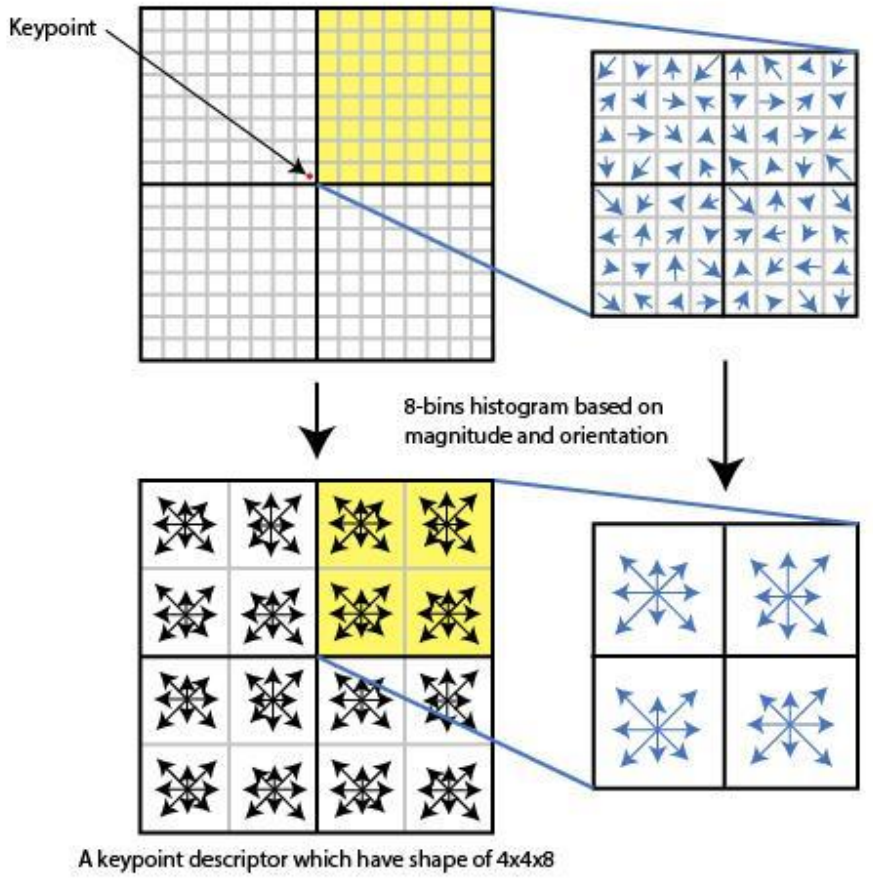


**Figure 11**. Calculating a keypoint descriptor

Unfortunately, there are some big disadvantages such as the requirement of gray-scale image and high computational cost which make it inefficient when deploying. Several algorithms based on the ideal of SIFT have been proposed to make improvements and eliminate the limitations, and major of them was made to reduce the processing speed of the algorithm. Although SIFT has gained a lot of attention on computational cost, it still requires a lot of hardware power to perform quickly. By noticing that problem, binary descriptors come in as an alternative which reduces a lot of processing time with little accuracy loss. The big factor which differentiates binary descriptors with normal descriptors in SIFT is that the value stored in binary descriptors is binary, which makes it faster in computing and easier to make comparisons while normal descriptors using floating-point value.

There are various algorithms that use binary descriptors, and Oriented FAST and Rotated BRIEF (ORB) **[11]** is one of the most notable ones. Just like its name, the algorithm takes advantage of two other algorithms, which is FAST **[37, 38]** for detecting keypoints and BRIEF **[39]** for computing descriptors. However, the authors of ORB observed that

the two algorithms have some deficiency such as FAST features do not have the usual keypoint orientation component and BRIEF is variant with rotation. Therefore, they propose an improvement on both called Oriented FAST (oFAST) and Rotated BRIEF (rBRIEF). By combining different techniques and improvements, ORB becomes one of the fastest feature matching algorithms that can be performed at real-time speed.

### 3.4 Recommender System

There are various types of recommender systems, from mathematical algorithms to machine-learning-based approaches. Those methods tried to find the hidden information, similarity, and relationship among the data. Some popular approaches can be named as collaborative filtering, which gives recommendations for the target customer based on similar customers and items, or context-based method which is based on several properties of the target customer's favorite item list to make predictions on potential favorite items. Despite the popularity, these approaches can not be used in our problem because of fast-pace and repeatable calculation. When deploying in real life, the system is required to have the ability to give predictions continuously and update customer favorite lists at high frequency. The algorithm that we want to use is Apriori **[40, 41]**, a rule-based recommendation that was widely used in the retail industry but rarely applied in Vietnam.

Apriori is an algorithm which is used to find patterns in data, more specifically, in billing, it finds items that often come together in every billing record. The big advantage of applying Apriori is that association rules of the algorithm can be precomputed from purchase history and stored in a database. When being used, the system only needs to compare lists of predicted items with a priori items in association rules to give recommendations as a posteriori.

The calculation of the algorithm is quite simple, an algorithm's loop is started by a list of k-itemset ($0 < k \leq K$), which has k items in each sample and was produced by calculating result of the previous loop, then we count the frequency of appearance of items in sample $X$ that come together in a billing database. Given $N$ is the number of billing records in the database and $Y$ is a subset of $X$:

$$Support(X) = \frac{Frequency(X)}{N}$$

$$Confident(Y \rightarrow X - Y) = \frac{Frequency(X)}{Frequency(Y)}$$

### 3.5 Overall Process

*Input:* A image of shape 640x640

*Output:* A list of recommended items based on association rules

The algorithm works in the following steps:

*Step 1*

Setting up required parameters, load detection model weight, prepare product database, class representation database and recommender association rules.

*Step 2*

Capture images from a recording device.

*Step 3*

Process several transform and normalization steps on the image.

*Step 4*

i. Transfer the image into a detection model and give bounding boxes prediction.

ii. If confident > box_conf, draw the corresponding bounding boxes on the image.

iii. If any confident > box_crop, go to Step 5.

iv. Go to Step 2.

*Step 5*

i. Crop images based on the location of bounding boxes.

ii. Transfer those images through a feature extractor to produce representations.

iii. Matching feature from the representations with class representation database.

iv. Give class predictions.

*Step 6*

Based on the output of Step 5, analyze the product database and the available association rules and to display product information and give recommendations.

# CHAPTER 4: EXPERIMENTAL RESULTS

## 4.1 Database Overview

The data we used to solve the problems in this document is RPC dataset **[2]**, which is announced in 2019 as the largest dataset about products ever exists. The dataset contains more than 50,000 one-object images in the training set, 6,000 and 24,000 multiple-object images in the validation set and test set respectively. After many evaluation and consideration processes on the performance of the system, we decided to use only the validation set and test set since we want to make it challenging for the system, which still performs well even with fewer data.

**Table 1.** Overview information of the RPC dataset

| Split | # images | # objects | #objects/image | #categories/image |
|---|---|---|---|---|
| Training set | 53,739 | 53,739 | 1 | 1 |
| Validation set | 6,000 | 73,602 | 12.27 | 6.33 |
| Test set | 24,000 | 294,333 | 12.26 | 6.31 |


**Figure 12.** Diverse object forms and categories in RPC dataset

Although the quality of the dataset is very good, there are a few limitations that we can observe while working with it. The model, which was trained with RPC dataset, does not perform well on low-quality and complex background images. This problem is big for our system since the in-store environment which is captured by camera is often very colorful and sometimes can be with different light conditions. Furthermore, to invest in a lot of high-quality cameras can be a challenge to small retail stores, so we try to make the model predict precisely even with low-quality input.

## 4.2 Each Step Evaluation

**Figure 13.** Preprocessing pipeline

**Figure 14.** Localization process

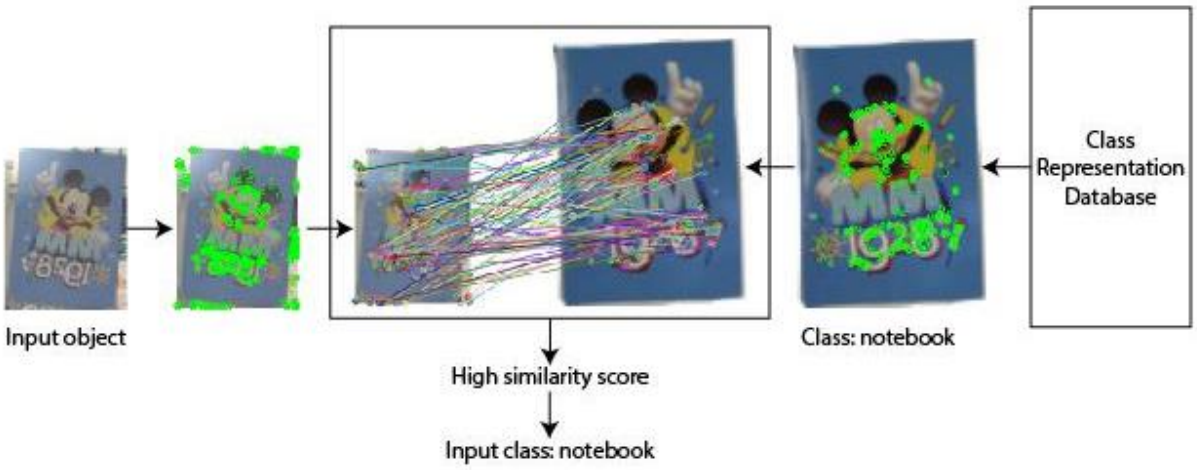


**Figure 15.** Feature matching and similarity calculation to give prediction

## 4.3 Overall Process Evaluation

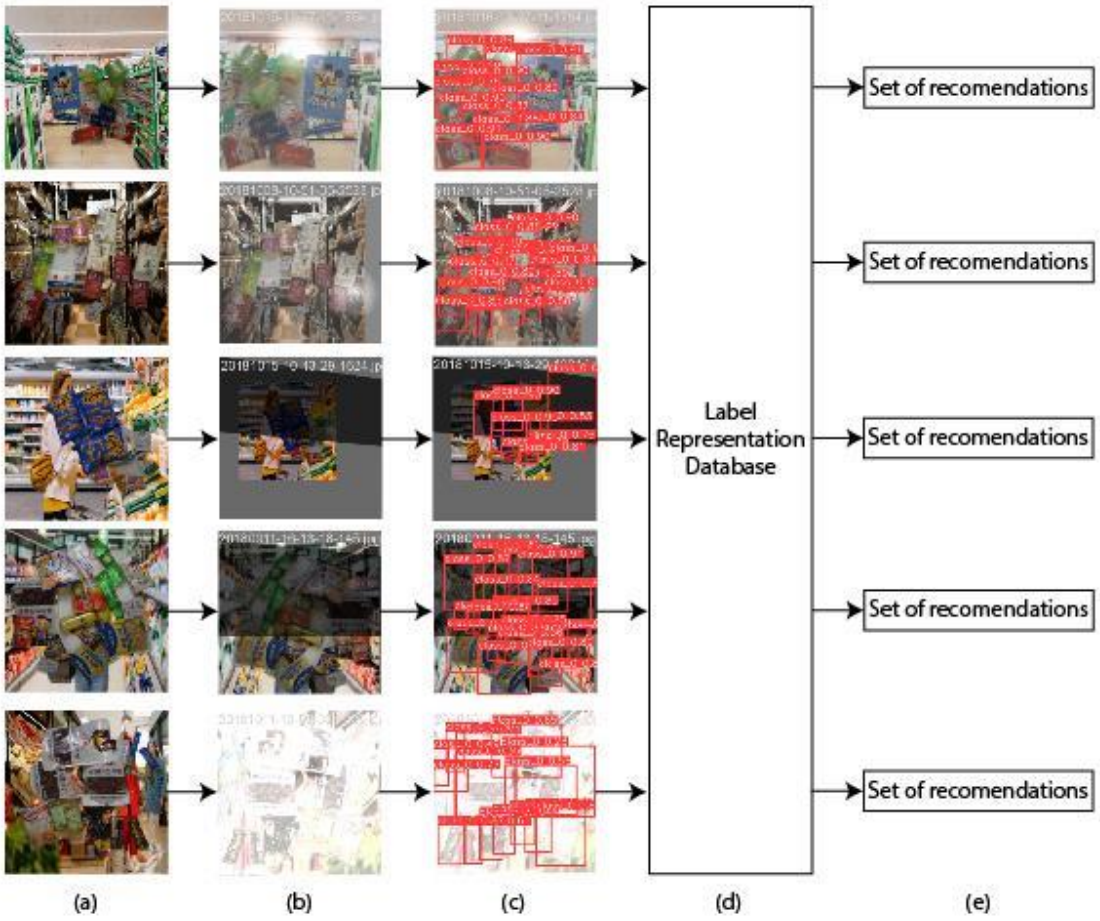


**Figure 16.** Overall process of the system. (a) Image after changing background (b) Apply different augmentation (c) Transfer through detection model (d) Cropped image representations

are compared with representation database (e) Find predicted classes in association rules to give recommendations

### 4.4 Other Approaches Comparison

**Table 2.** Performance of fine-tuned YOLOv8-S trained on original training dataset

| Validation dataset | Precision (%) | Recall (%) | $mAP_{50}$ (%) | $mAP_{50-95}$ (%) |
|---|---|---|---|---|
| Original | 99.2 | 99 | 99.5 | 88.7 |
| Augmented | 80.4 | 73.5 | 80.8 | 64.2 |

**Table 3.** Performance measurement on the dataset with the same training condition

| Models | Params (M) | Precision (%) | Recall (%) | $mAP_{50}$ (%) | $mAP_{50-95}$ (%) |
|---|---|---|---|---|---|
| YOLO-NAS-S | 22.2 | 94.9 | 89.2 | 91.8 | 75.6 |
| YOLOv8-M | 25.9 | 99.2 | 89.6 | 94.9 | 80.2 |
| RT-DETR-L | 32 | 98.1 | 90.7 | 95.2 | 81.9 |

**Table 4.** Average processing time per image on GPU Tesla T4

| Models on T4 GPU | Preprocessing (ms) | Inference (ms) | Postprocessing (ms) | Total (ms) |
|---|---|---|---|---|
| YOLOv8-M | 1.6 | 15.2 | 2.7 | 19.5 |
| YOLOv8 (TensorRT) | 1.6 | 5.3 | 1.1 | 8 |
| RT-DETR-L | 3.6 | 55.0 | 0.7 | 59.3 |
| YOLO-NAS-S | - | - | - | 15.9 |

**Table 5.** Predicting time on the class database with 92 images

| Matching methods | Processing time (s) |
|---|---|
| SIFT | 5.26 |
| ORB | 1.18 |

# CHAPTER 5: DISCUSSION

## 5.1 Benefits of Proposed Approach

Through several preprocessing methods, we can achieve a system which can perform well in various severe scenarios including background complexity, illumination and blur captured objects. By using state-of-the-art detector YOLOv8, we can do real-time detection at the speed of 125 frames per second. Feature extraction and matching using descriptors approach ORB makes the comparison fast and invariant with object scale and orientation. The system is integrated with a recommender to give customers a friendly buying suggestion which can increase business profitability.

## 5.2 Limitations of Proposed Approach

There are several big limitations of our approach. Firstly, in the preprocessing stage, the image background needs to be removed. In consideration of what we have and what we can do, we use saliency detection to segment objects and remove image background. This process contains risk of information loss and wrong labeling due to the imperfect object segmentation which cannot identify all the objects presenting in image. Another limitation stands out from the feature matching process, while ORB is a powerful approach in handling image similarity tasks, it has the same weakness as other descriptor-based methods which require gray-scale image as input, this can cause some loss in accuracy since color is quite an important feature when identifying products.

## 5.3 Future Works

Every machine learning and deep learning task requires the right data, and our system does not have the thing. Hopefully in the future there will be a dataset which can satisfy the quality requirements of our problem, or we can discover some methods that can efficiently remove image background while keeping all the information.

Feature extractor and matching is also a thing that needs to be improved. Since every product can be very different from each other by the color, if we can integrate the color properties into descriptors, it could be a very effective system.

# CONCLUSIONS

In this paper, we have discovered and discussed multiple approaches for each process in the system. From that, we can find the optimized trade-off point between accuracy and speed which satisfies hard requirements in the rise of machine learning. Although the system can meet the target performance and be usable in many real-life scenarios, there is room for improvement. Finally, we need to determine scale, technology, infrastructure, and pipeline of data when we come to deploy the system since it requires an appropriate amount of investment.

# REFERENCES

[1] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, pp. 594–611, 2006.

[2] X. Wei, Q. Cui, L. Yang, P. Wang, and L. Liu, "RPC: A Large-Scale Retail Product Checkout Dataset," 2019, https://arxiv.org/abs/1901.07249.

[3] Duru, A. (2019). Food-Body Relationship. In: Kaplan, D.M. (eds) Encyclopedia of Food and Agricultural Ethics. Springer, Dordrecht.

[4] A. Sinha, S. Banerjee, and P. Chattopadhyay, "An Improved Deep Learning Approach For Product Recognition on Racks in Retail Stores," 2022, https://arxiv.org/abs/2202.13081.

[5] J. Zhu, T. Park, P. Isola and A. A. Efros, "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks," *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017, pp. 2242-2251.

[6] S. Wei, R. Guo, C. Cui et al., "PP-ShiTu: A Practical Lightweight Image Recognition System," 2022, https://arxiv.org/abs/2111.00775.

[7] Y. Ma, D. Yu, T. Wu, and H. Wang. PaddlePaddle: An Open-Source Deep Learning Platform from Industrial Practice[J]. *Frontiers of Data and Computing*, 2019, 1(1): 105-115.

[8] X. Qin, Z. Zhang, C. Huang et al., "U2-Net: Going deeper with nested U-structure for salient object detection," *Pattern Recognition*, 106 (2020).:107404-

[9] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz and D. Terzopoulos, "Image Segmentation Using Deep Learning: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 7, pp. 3523-3542, 2022.

[10] G. Jocher, A. Chaurasia, and J. Qiu (2023). YOLO by Ultralytics (Version 8.0.0) [Computer software]. https://github.com/ultralytics/ultralytics.

[11] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," *2011 International Conference on Computer Vision*, Barcelona, Spain, 2011, pp. 2564-2571.

[12] I. Goodfellow, J. Pouget-Abadie, M. Mirza et al., "Generative adversarial nets," *Advances in Neural Information Processing Systems*, MIT Press, Cambridge, MA, USA, 2014.

[13] K. Cui, G. Zhang, F. Zhan, J. Huang, and S. Lu, "FBC-GAN: Diverse and Flexible Image Synthesis via Foreground-Background Composition," 2021, https://arxiv.org/abs/2107.03166.

[14] K. K. Singh, U. Ojha and Y. J. Lee, "FineGAN: Unsupervised Hierarchical Disentanglement for Fine-Grained Object Generation and Discovery," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, 2019, pp. 6483-6492.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015, https://arxiv.org/abs/1505.04597.

[16] D. Gatis (2022). Rembg [Computer software]. https://github.com/danielgatis/rembg.

[17] A. Buslaev, A. Parinov, E. Khvedchenya, V. I. Iglovikov, and A. A. Kalinin, "Albumentations: fast and flexible image augmentations," 2018, https://arxiv.org/abs/1809.06839.

[18] O. Mazhar, J. Kober, "Random Shadows and Highlights: A new data augmentation method for extreme lighting conditions," 2021, https://arxiv.org/abs/2101.05361.

[19] S. Dimitrios, H. P. H. Shum and E. S. L. Ho, "Illumination-Based Data Augmentation for Robust Background Subtraction," *2019 13th International Conference on Software, Knowledge, Information Management and Applications (SKIMA)* (2019): 1-8.

[20] J. Terven, D. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," 2023, https://arxiv.org/abs/2304.00501.

[21] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 779-788.

[22] R. team, "YOLO-NAS by Deci Achieves State-of-the-Art Performance on Object Detection Using Neural Architecture Search." https://deci.ai/blog/yolo-nas-object-detection-foundation-model, 2023. Accessed: Dec 13, 2023.

[23] G. Jocher (2020). YOLOv5 by Ultralytics (Version 7.0) [Computer software]. https://doi.org/10.5281/zenodo.3908559.

[24] T. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan and S. Belongie, "Feature Pyramid Networks for Object Detection," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 2017, pp. 936-944.

[25] Z. Ge, S. Liu, F. Wang, Z. Li and J. Sun, "YOLOX: Exceeding YOLO Series in 2021," 2021, https://arxiv.org/abs/2107.08430.

[26] T. Elsken, J. H. Metzen and F. Hutter, "Neural Architecture Search: A Survey," 2019, https://arxiv.org/abs/1808.05377.

[27] R. team, "Automated Neural Architecture Construction (AutoNAC)." https://deci.ai/deep-learning-glossary/automated-neural-architecture-construction-autonac, 2023. Accessed: Dec 28, 2023.

[28] X. Chu, L. Li, and B. Zhang, "Make RepVGG Greater Again: A Quantization-aware Approach," 2022, https://arxiv.org/abs/2212.01593.

[29] Vaswani, Ashish, Noam M. Shazeer, Niki Parmar et al., "Attention is All you Need," *Neural Information Processing Systems*, 2017.

[30] A. Dosovitskiy, L. Beyer, A. Kolesnikov et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," 2020, https://arxiv.org/abs/2010.11929.

[31] W. Lv, Y. Zhao, S. Xu et al., "DETRs Beat YOLOs on Real-time Object Detection," https://arxiv.org/abs/2304.08069.

[32] N. Carion, F. Massa, G. Synnaeve et al., "End-to-End Object Detection with Transformers", 2020, https://arxiv.org/abs/2005.12872.

[33] H. Zhang, Y. Wang, F. Dayoub and N. Sünderhauf, "VarifocalNet: An IoU-aware Dense Object Detector," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Nashville, TN, USA, 2021, pp. 8510-8519.

[34] J. Zhai, S. Zhang, J. Chen and Q. He, "Autoencoder and Its Various Variants," *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Miyazaki, Japan, 2018, pp. 415-419.

[35] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the 2017 7th International Conference on Computer Vision*, Kerkyra, Greece, 1999.

[36] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[37] E. Rostenand and Tom Drummond, "Machine Learning for High-Speed Corner Detection," *European Conference on Computer Vision*, 2006.

[38] E. Rosten, R. Porter and T. Drummond, "Faster and Better: A Machine Learning Approach to Corner Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 1, pp. 105-119, 2010.

[39] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "BRIEF: Binary Robust Independent Elementary Features," *European Conference on Computer Vision*, 2010.

[40] R. Srikant, “Fast algorithms for mining association rules and sequential patterns,” UNIVERSITY OF WISCONSIN, 1996.

[41] M. Al-Maolegi, B. Arkok, “An Improved Apriori Algorithm for Association Rules,” 2014, https://arxiv.org/abs/1403.3948.