An Effective Framework of Multi-Class Product Counting and Recognition for Automated Retail Checkout

Junfeng Wan[1*], Shuhao Qian[1*], Zihan Tian[1*], Yanyun Zhao[1,2†]
[1]Beijing University of Posts and Telecommunications
[2]Beijing Key Laboratory of Network System and Network Culture, China
{wanjunfeng,qiansh,Tianzh,zyy}@bupt.edu.cn

Abstract

*As the field of computer vision grows, Automated Retail Checkout has become a highly anticipated development goal. The key of this task is to improve the accuracy rate. If there is an error, it will bring serious losses to the business and awful experience for customers which is not our expected. This competition gives us an opportunity to simulate check-out in a real world scenario, so that we can identify problems and solve them, not only for the competition, but also for the practical application. As one of the participating teams in this task, we pursue the goal of avoiding misdetection and misclassification, and build a complete set of framework to achieve high-precise, high-recall performance. In addition, there is an excessive difference between the training data and test data. How to use limited data to make up for the differences in this part is also one of the highlights of our framework. In general, our framework consists of three main parts. Firstly, the **Pre-Processing module** to make up for the differences between training and test data. The **DTC module** completes the overall process of automatic recognition. Finally the **MTCR module** is proposed to post-process the output of the DTC module. On the TestA data of AICITY2022 Task 4, we have achieved significant result compared to the other teams. Finally, our model is ranked **1st** in AICITY2022 Task 4 [17]. The code is available at: `https://github.com/w-sugar/DTC_AICITY2022`.*

1. Introduction

The challenges of AI CITY have promoted the research of vision problem in recent years, especially in intelligent transportation, such as traffic statistics, vehicle re-recognition, cross-camera tracking, abnormal event analysis and other application scenarios related to intelligent transportation. It is known as the "ImageNet Competition for Intelligent transportation video Analysis".

A growing application of AI and computer vision is in the retail industry. Of the various problems that can be addressed, track 4 of 2022 AI City Challenge, Multi-Class Product Counting and Recognition for Automated Retail Checkout, focuses on accurate and automatic check-out in a retail store. Participating teams will count and identify products as they move along a Retail checkout conveyor belt. The challenge stems from the real-world scenario of occlusion, movement, similarity in products being scanned, novel SKUs that are created seasonally, and the cost of misdetection and misclassification [17].

In track 4 task, we need to detect the objects, tracking for counting and classify them correctly in test videos. Image classification is one of basic vision tasks, and its goal is to classify images into corresponding categories. Automated retail checkout is more than an image classification task. Before classification, it is necessary to locate when and where products appear in test videos, so we firstly need to train object detection model with training dataset of track 4 to detect products, and then classify the detecting objects. As shown in Figure 1, the training data is shown on the left and the testing data is shown on the right. We can see that the training data and the test data [18, 28] are very different in appearance. The images of training set are low resolution and have aliasing textures, and also some areas of the object are very dark. In contrast, there are no such quality problems in testing data. The distribution difference of ap-

Figure 1. Training data shows that images with low quality which has noise and imbalanced color.While the images in the test are high quality which are really different from the training data.

*Equal contribution
†Corresponding authors

pearance between training and test data greatly leads to the phenomenon for excellent performance on the training set but poor performance on the test set. For example, it works well on our homemade detection training set, but there are a lot of false detections and missed detections on the test videos. Therefore, how to make up for the difference between training and test data is also an urgent problem to be solved in this task. So we exploit an image enhancement algorithm [11] to preprocess the images of training set, which decreases the difference between the two data sets, and get satisfactory performance of detection model and classification model.

In addition, each image of training set is only including one object of products, and they can be used directly for object classification network training, but not for detection network training. While the test data is untrimmed videos and each frame contains various objects in the scene, so it is necessary to locate when and where the product appears in advance. Then we construct an object detection dataset to train an effective detector with the preprocessing training data.

Tracking of objects is necessary as each product needs to be counted only once while it passes through the region of checkout. Therefore, we track the product detections in videos based on DeepSORT [27], and one tracking trajectory obtained represents a product.

In light of these challenges and the characteristic of automated retail checkout, we propose a precise and efficient framework and get an excellent evaluation result. Our main contributions are as follows:

1) Based on Track 4 automated retail checkout in AI CITY 2022 Challenge, we propose a reasonable and effective product identification and counting framework. It consists of **Pre-processing**, **Detection-Tracking-Classification(DTC)** module and **Post-processing**. Finally, we get the first place in test A.

2) Aiming at the issues of aliasing and too dark brightness in the images of the training set, we enhance the training images with **Multi-Scale Retinex with Color Restoration(MSRCR)** algorithm [11], makes up the quality difference between training and test images, which greatly improves the performance of object detection and classification.

3) We also present a specific post processing for the output of DTC module, and name it as **MTCR** algorithm which improves the accuracy of classification.

2. Related Work

In the section, an overview of related work is presented in response to the proposed research work, which mainly includes the following three aspects: object detection, tracking and classification.

2.1. Object Detection

Object detection task is to find out all the objects of interest in images, and locate their positions. It is one of the core problems in computer vision and an essential part of our entire pipeline. There are two kinds of categories of object methods: one-stage methods [13,15,20,22] and two-stage methods [4, 9, 21, 25]. One-stage means that extract features directly in the network to predict classification and location without region proposal. Many modern object detectors demonstrate outstanding performances by using the mechanism of looking and thinking twice, the first step is to give a region proposal may contain objects and then classify and locate through a CNN module, which is also the overall idea of the two-stage method. Existing detectors are trained using large-scale datasets. How to complete the detection task without trianing data is a difficult problem.

2.2. Tracking

The state-of-the-art methods of Multiple Object Tracking(MOT) [8,16,27] usually divide into Two-Step MOT and One-Shot MOT. Two-step MOT [1, 6, 7, 27] methods often treat object detection and association as two separate tasks which first use detectors to localize all objects of interest in the frame of videos, and then in a separate step they crop object regions according to the boxes and feed them to the identity embedding network to extract Re-ID features to associate objects and tracks. However, One-Shot MOT [26] simultaneously accomplish object detection and identity embedding in a single network that can reduce the inference time through sharing most of the computation. In this competition, we pay more attention to the tracking accuracy rather than the speed, so we chose DeepSORT [27] which allows users to better optimize the detector and tracker for the tracking effect.

2.3. Classification

Image classification is a fundamental task in computer vision research. Recent work such as object detection, semantic segmentation has significantly boosted image classification which serves as the backbone. In recent years, with the development of deep learning networks, a number of classification networks with good performance have emerged. Like transformer [24] leverages attention mechanism to improve model training speed which has higher accuracy and performance than the previously popular RNN [29]. Efficientnet [23] uniformly scales depth, width and resolution with a fixed set of scaling factors for improved accuracy. ResNeSt [30] is a deformation model of ResNet [10] by stacking Split-Attention block that enables across feature-map groups. For improving the accuracy of classification, data augmentation [2, 3] is an effective way. The most common data augmentation methods are random erasing [31], resize, random flip, random gray scale and so on. By data augmentation, the quality of images has obviously improved, allowing it to be processed better.

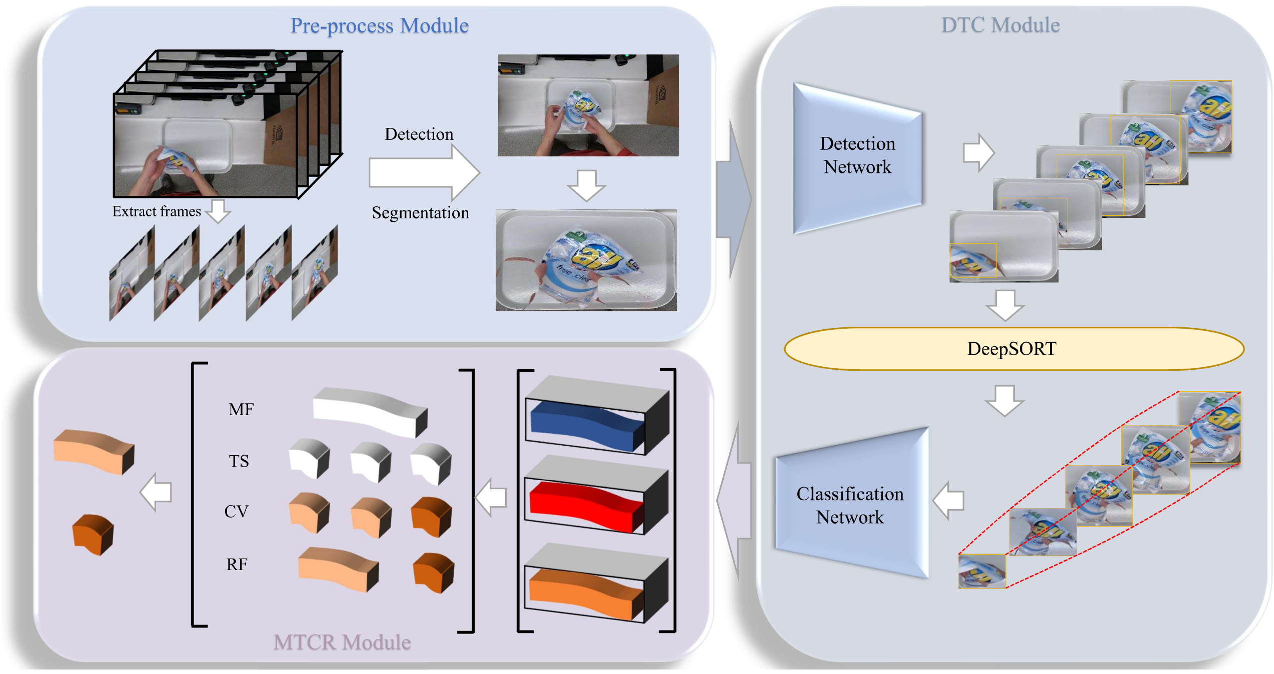


Figure 2. Overview of the proposed system for counting and recognition in test set A videos. Frames are extracted from video and pre-processed by pretrained model. pre-processed frames are fed into detection network, producing location bounding boxes. DeepSort produces tubelets with bounding boxes, which are then classified and passed to our MTCR system, from which products recognition and timestamp are obtained.

3. Method

In order to count and recognize products in Track 4 accurately, we propose a system that contains pre-processing, detecting-tracking-classifying(DTC) and Post-processing modules as shown in Figure 2, which takes frames extracted from test set A as input. Firstly, the frames are fed into pre-processing module, producing cropped and masked frames. Secondly, the processed frames are fed into detection network, outputing location bounding boxes. Then, frames with location message are fed into DeepSort and classification network, generating trajectories with category scores. Finally, counting and recognition results could be gotten by MTCR system from the trajectories.

3.1. Pre-processing Module

In order to solve the problem of image aliasing and too dark brightness in the training set, we have made many attempts to make the distribution of RBG in the training dataset and the distribution of the test dataset as close as possible to improve the detection and classification effect of this scheme.

The most important processing methods at this stage are data augmentation with MSRCR [11]. As shown in the top of Figure 3, the images in the original dataset are of poor quality, most of them are too bright or too dark, we exploit MSRCR. The key formula of MSRCR's algorithm is:

$$Diff_{i,c} = log(I_c) - log(I_c * G_{\sigma_i}), \tag{1}$$

$$MSR_c = \sum_{i=1} Diff_{i,c}/3, \tag{2}$$

$$MSRCR_c = MSR_c * (log(125 I_c) - log(I_R + I_G + I_B)), \tag{3}$$

where I is the input color image, $c \in \{R, G, B\}$, σ_i is the scales, $i = 1, 2, 3$ means three color channels.

MSRCR has perfect performance in image enhancement by applying it on each color channel independently, so that a color balance can be effectuated. Experiments illustrate that the images enhanced by MSRCR can improve the performance of both detector and classifier. For detector, the high quality of images can be used to detected and localized well. And for classifier, the enhanced images can be easily distinguished by classification network model with high accuracy, for the reason that the enhanced training set image RGB distribution is closer to the test set RGB distribution.

Different from the training dataset with only images, the test datasets are videos, and someone is holding the product for detecting. For the purpose of eliminating this difference, people need to be detected and segmented, then use the first frame that usually does not contain people to complement. Using open source model for segmentation may leave some areas that cannot be masked.

3.2. DTC Module

For accurate identification and counting of products, our core module in this framework is DTC Module. The DTC module mainly includes three parts: detector, tracker and classifier.

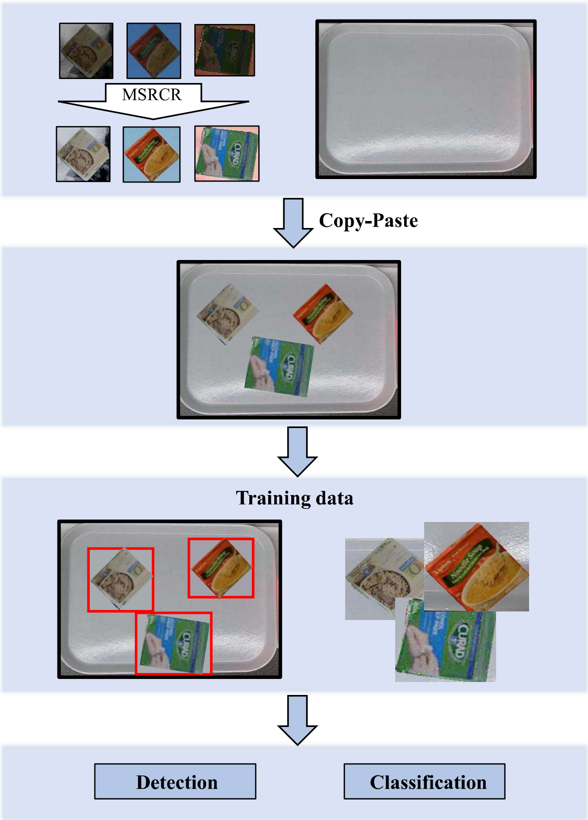


Figure 3. The overall flow of the training phase. The training data is generated using the combination of the white pallet and the MSRCR-processed images.

3.2.1 Detector

We need to count and recognize products in the delivery process video. It is important that commodities are detected in the video to eliminate background interference for object classification.

In order to train the detector, as shown in Figure 3, we use the white tray as the background, the products processed by MSRCR as the foreground, and paste these foregrounds into the background randomly, and its position can be used as the target of the detector training. Without violating the rules of using other datasets, in order to make the background styles diverse, we randomly selected data from other tasks as the background and resized them to the same size as the white tray. And the detector we use is DetectoRS [19], which achieves the best performance in detection.

To accurately detect product and record the start time when the product appears, we pay more attention to detect the tray in videos. We use DetectoRS with pre-trained parameters to detect the tray, and objects.

By constructing the training set and selecting the detection network, we improve the detection performance.

3.2.2 Tracker

Object tracking is necessary since each object is only counted once while it passes through the checkout region. DeepSort [27] is one of the most typical methods of tracking-by-detection category. In our system, we apply DeepSort to associate the product detections of different frames in test videos, and get the trajectories of products. In order to extract more discriminative features for object association, we exploit the last layer features of our classifier to represent appearance of products, and then fill them to the cost matrix, and use the Hungarian algorithm [12] to match products in the DeepSort algorithm. Benefitting from image enhancement in section 3.1, we extract more expressive appearance features for object association from our classifier. Therefore, the feature matching process is more stable and robust.

3.2.3 Classifier

Accurate classifying products is a critical component of our system, so we pay more attention to designing our classifier network which is used to classify each object from one product trajectory we gotten. We choose multiple classifiers with better performance on the ImageNet [5], such as ResNeSt [30], EfficientNet [23], etc. Different models have complementary effects on different categories.

An excellent classifier could classify hard negative samples as other categories, so that they can be eliminated before subsequent processing. To depress misclassification, such as human hands, empty white trays, etc., we use false detections on our homemade detection training set and the randomly cropped region on it as the background category of our classifier training, motivating from the practice of two-stage detectors, such as Faster R-CNN [21]. Because of adding background category, we can classify the hard negative samples correctly and improve the recognition function of DTC module.

In general classification tasks, we use the hard label method to perform one-hot encoding on the label, and adding a little noise to the one-hot encoding obtained by the hard label can make the network convergence effect better.

In the classification task, the cross-entropy loss is as follows:

$$H(y, y^{'}) = -\sum_{i=1}^{K} y_i * log(y_i^{'}). \tag{4}$$

The $y^{'}$ after label smooth operation is as follows:

$$y_i^{'} = \begin{cases} 1-\varepsilon, & i = target \\ \frac{\varepsilon}{K-1}, & otherwise \end{cases}, \tag{5}$$

where ε is an artificially specified number of 0-1, K is the number of categories.

In the part of classifier, we use multiple networks to train the classifiers and add background category to the training set which make our classifier works well.

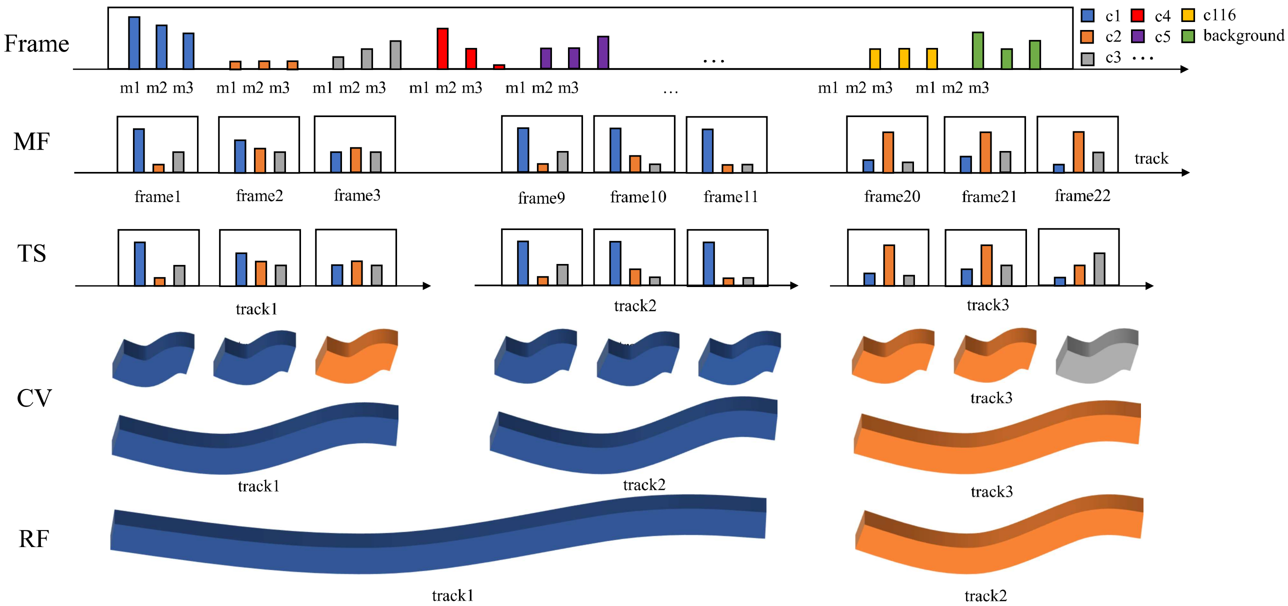


Figure 4. The overall flow of the MTCR algorithm. We firstly fuse results from different models. Secondly, we split tracks that have mistakes. And then, we vote to confirm the category of each track while taking the medium frame as the timestamp. Finally, we fuse the results that belong to the same category and have neighbor timestamps.

3.3. MTCR Module

In order to further improve the accuracy of our results, we input the results from DTC module to MTCR module, which further improves the product counting and recognition. The entire MTCR moduleis as shown in Figure 4, and illuminated in Algorithm 1. Firstly results of DTC module are fused by Model fusion algorithm; secondly, problematic trajectories are splitted to depress mistakes by Track splitting algorithm. And then, Category voting algorithm determines the category of trajectories. Finally, the Results fusion algorithm merges the system output results.

Model fusion. To take advantage of different models, we fuse the results from 3 different models by averaging category scores. We use s_{ijkl} to represent the lth category score in the jth trajectory and the kth frame from the ith model after DTC module. And we use $s^{'}_{jkl}$ to represent the lth category score in the jth trajectory and the kth frame after Model Fusion. Formally, the category score after Model Fusion $s^{'}_{jkl}$ is defined as:

$$s^{'}_{jkl} = \sum_{i=1}^{3} s_{ijkl}/3. \tag{6}$$

Comparing to weighting all model results, our approach is more universal and not just achieving recognition and counting task on the video of TestA.

Track splitting. There is a case of misconnecting two trajectory with a long time interval in DTC module. To solve the problem, we split the trajectory with a long time interval. We use $f_{jk}(f_{jk} \in T_j)$ to represent the frame index of the kth frame in the jth trajectory, and f_thr to represent the time threshold of long time interval. We split the trajectory T_j when $f_{j(k+1)} - f_{jk} > f\_thr$ happends.

Category voting. The synthetic images can not include all perspective of products, which could lead to awful scores for some frames in a predicted trajectory and matter classification. Thus, we regard the category having top score as the prediction of frame and vote for the category of a trajectory with them to avoid this extreme situation. We defined scores in the jth trajectory and the kth frame S_{jk} as $S_{jk} = \{s^{'}_{jkl} \mid l \in \{1, 2, \ldots 117\}\}$. The index l of max score $s^{'}_{jkl}$ in S_{jk} represents the category of the kth frame in the jth trajectory and we use a counter $counter_l(l \in \{1, 2, \ldots 117\})$ to record the times of index l appearing. And the most frequently index represents the category of a trajectory, which is assumed as temporary c_j. And then, we use $s^{''}_{j}$ to represent the score of the jth trajectory, which is defined as:

$$s^{''}_{j} = \sum_{k=1}^{m_j} s^{'}_{jkc_j}/m_j, \tag{7}$$

where m_j denotes the number of frames in the jth trajectory.

Comparing to firstly averaging category scores of frames in a trajectory and then selecting the category having the top score which we do in most cases, selecting firstly and voting then averaging the category elected scores avoids the effects of extreme scores, which is more robust.

Results fusion. Because of the lack of data for training detectors, we can not detect a product continuously at some time, which might lead to Cut-off error in tracking task. Thus, we fuse the results which have the same category and the neighbor timestamps. We use $p_i = (c_i, t_i)$ to represent

Algorithm 1 The MTCR module which is post-processing of our system.

Input: $A = \{s_{ijkl} \mid i \in \{1, 2, 3\}, j \in \{1, \dots n\}, k \in \{1, \dots m_j\}, l \in \{1, \dots 117\}$, s_{ijkl} to represent the lth category score in the jth trajectory and the kth frame from the ith model after DTC module. A is a set of s_{ijkl}, m_j is the number of frames in the jth trajectory; $S_{jk} = \{s^{'}_{jk1}, s^{'}_{jk2}, \dots s^{'}_{jk117}\}$, S_{jk} is a set of category scores, $'$ represents after Model Fusion; $F_{jk} = \{f_{jk}, S_{jk}\}$, F_{jk} include S_{jk} and frame index f_{jk}; $T_j = \{F_{j1}, F_{j2}, \dots F_{jm_j}\}$, T_j is a set of F_{jk}; $M = \{T_1, T_2, \dots T_n\}$, M is a set of T_j, representing trajectories in a model.

Output: a set of predictions, P

procedure MTCR(A)
  $P \leftarrow \{\}$
  $M = Model\ Fusion(A)$
  for all $T_j \in M$ **do**
    $ST_j \leftarrow Track\ Splitting(T_j)$
    for all $T^{'}_j \in ST_j$ **do**
      $(c_j, s^{''}_j) \leftarrow Category\ Voting(T^{'}_j)$
      if $|T^{'}_j| > t_1$ & $c_j < 117$ & $s^{''}_j > t_2$ **then**
        t_j = average $T^{'}_j$ by f_{jk}
        append (c_j, t_j) to P
      end if
    end for
  end for
  $P \leftarrow Result\ Fusion(P)$
  return P
end procedure

the prediction before Result fusion. Firstly, we use P_c to represent the prediction group that have the same category c. Secondly, we search for p_is which have neighbor t_i and average t_is to form a new prediction. We use thr to represent the interval threshold and fuse the p_i and the p_j when $|t_i - t_j| < thr(p_i, p_j \in P_c)$ happends.

Not only could Result fusion module make up intermittent detection, it also could compensate the mistakes caused by Track splitting, which makes our approach more robust.

Our post-processing follows: model fusion-track splitting-category voting-result fusion. The post-processing method alleviated the problems such as continuous missed detection, Mix-up error in tracking task, lack of perspectives of products data, which makes our approach more robust. The detail of four parts of MTCR Algorithm is shown in Algorithm 2.

4. Experiments

We demonstrate the effectiveness of our proposed frame work on the test set A, in which the customer hold the merchandise item. And train our models on training set with synthetic images. All results follows are performed on the test set A.

Algorithm 2 algorithm functions of the MTCR module.

function MODEL FUSION(A)
  for $s_{ijkl} \in A$ **do**
    Formula 6
    update $s^{'}_{jkl}$ in M
  end for
  return M
end function

function TRACK SPLITTING(T)
  Sort T_j by f_{jk}
  $ST \leftarrow \{\}, fid = f_{j1}, new\_T = \{F_{j1}\}$
  for $i = 2 \rightarrow m_j - 1$ **do**
    if $|f_{ji} - fid| < f\_thr$ **then**
      append F_{ji} to new_T, $fid = f_{ji}$
    else
      append new_T to ST
      $fid = f_{ji}, new\_T = \{F_{ji}\}$
    end if
  end for
  append new_T to ST
  return ST
end function

function CATEGORY VOTING(T_j)
  $counter = \{0\} * 117$
  for $(f_{jk}, S_{jk}) \in T_j$ **do**
    c = MaxIndex(S_{jk}), $counter_c$++
  end for
  c_j = MaxIndex($counter$)
  Formula 7
  return $c_j, s^{''}_j$
end function

function RESULT FUSION(P)
  P_new = {}
  Group P by Category, get P_c
  Sort P_c by t_{ci}
  for $c^{'} \in c$ **do**
    $P_t \leftarrow \{(c^{'}, t_{c^{'}1}), fid = t_{c^{'}1}$
    for $i = 2 \rightarrow |P_{c^{'}}|$ **do**
      if $|t_{c^{'}i} - fid| < thr$ **then**
        append $(c^{'}, t_{c^{'}i})$ to P_t, $fid = t_{c^{'}i}$
      else
        $(c^{'}, t^{'})$ = average p_t by t
        append $(c^{'}, t^{'})$ to P_new, $fid = t_{c^{'}i}$
      end if
    end for
    $(c^{'}, t^{'})$ = average p_t by t
    append $(c^{'}, t^{'})$ to P_new, $fid = t_{c^{'}i}$
  end for
  return P_new
end function

Rank	Team ID	Score
1	**16**	**1.0000**
2	94	0.4783
3	104	0.4545
4	165	0.4400
5	66	0.4314
6	76	0.4231
7	117	0.4167
8	4	0.4082
9	9	0.4000
10	55	0.4000

Table 1. Comparison of the scores with different teams, our team ID is 16, and we get 1.0000 in the test A, means that all the merchandise items are classified correctly.

4.1. Datasets

The dataset contains a total of 116,500 synthetic images from 116 different merchandise items for training and 5 video clips appearing these items for testing in test set A. The synthetic images are created from 3D scanned object models. The perspective of the test video clips is above the checkout counter and facing straight down while a customer is pretending to perform a checkout action by "scanning" objects in front of the counter in a natural manner. And there is a shopping tray placed under the camera to indicate where the model should focus.

4.2. Implementation details

We choose the DetectoRS [19] as the detection network, Resnest50, Resnest101 [30] and efficientnet-b2 [23] as the classification network, DeepSORT [27] as the tracking network.

Training phase. We train detection models with input size 500x400 for 5 epochs with the learning rate multiplied by 0.1 after 4 epochs. We train classification models with size 224x224 after crop for 20 epochs with the learning rate multiplied by 0.1 after 5, 10, 15 epochs. To expand the dataset, we load the pre-training model parameters on MS-COCO [14] and ImageNet [5].

Test phase. Firstly, we extract frames from video clips in test set A with FFmpeg library. Secondly, using model pre-trained on MS-COCO, we detect the white tray and mask person with frame in which its IOU with the white tray is 0. We crop the region of white tray in masked images where we inference our model.

4.3. Experimental results

Through a series of iterative optimizations, we gradually improve the performance of our framework. On the official test set A, we try to evaluate the performance of our framework. Incredibly, our method achieves a score of **1.0000**, outperforming the performance of all other team methods. The results of all teams are shown in Table 1.

Method	Ori	MSRCR	Precision	Recall	F1-Score
Effi-b0	✓	-	0.8571	0.6207	0.7200
	-	✓	0.9259	0.8621	0.8929
	✓	✓	**0.9310**	**0.9310**	**0.9310**

Table 2. Comparison with different methods in the pre-processing part. "✓" means this method is used. "-" means this method is not used.

4.4. Ablation study

In order to prove the effectiveness of each module of our system, we analyze the preprocessing part, the DTC module and the MTCR module separately.

4.4.1 Pre-processing

This part we use EfficientNet-b0 as the classifier and use the same results for detection and tracking. As shown in the Table 2, the first row uses the original images to train classifier, which has poor quality, obviously the result is not satisfactory. Therefore we use MSRCR to enhance the original image to adjust those pixels that are too bright or too dark. We can see image quality has been significantly improved, and it is more suitable for processing. After the fusion training of the two types of data, the results have been improved to a certain extent.

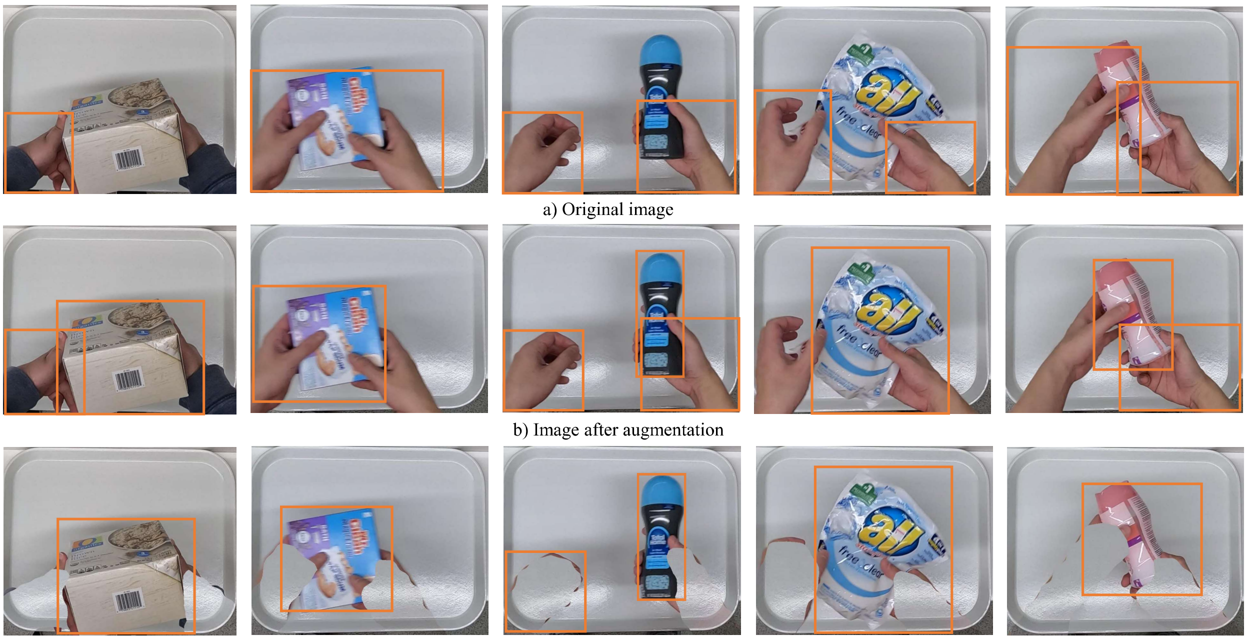
4.4.2 DTC Module

In this stage, our main purpose is to compare the effects of DTC part. And in detection, we visualize the detection results using three different data as shown in Figure 5. The first set of data uses the original images to train detector, which have poor quality. Obviously the detection effect is not satisfactory. Therefore we use MSRCR to enhance the original image. We can see that many products are detected, but in many cases, human hands are also detected by mistake. Taking into account the actual situation during the test, using the masked image is a good idea and a nice boost.

Table 3 shows the results of the comparison. We also use EfficientNet-b0 as the classifier. The first row uses the original images to train detector. After we use MSRCR to enhance the original image, the detection effect is satisfactory obviously. Then after we use classifier's feature to track, the precision and recall have been improved. We can see in the test video, someone is holding the products for detection, in order to eliminate the interference of the background, focus as much as possible on the products itself, we also mask the hand. The results are best when both classifier's feature and mask are used.

4.4.3 MTCR Module

As shown in Table 4, we train multiple single models and fuse the models based on their scores on this task. As can



a) Original image

b) Image after augmentation

c) Image after segmentation

Figure 5. Strong contrast when the same type of product is used for detection. Image after segmentation can be best detected, the only error-checked class is the background class while original image are difficult to be detected and the image after augmentation has many false positive boxes.

Method	Ori	MSRCR	Track	Mask	Precision	Recall	F1-Score
Effi-b0	✓	-	-	-	0.8620	0.8620	0.8620
	-	✓	-	-	0.8436	0.9310	0.8852
	-	✓	✓	-	0.8710	0.9310	0.9000
	-	✓	✓	✓	**0.9310**	**0.9310**	**0.9310**

Table 3. Comparison with different methods in the DTC part. Track is we use classifier's feature to track. Mask is that uses background color to erase hands. "✓" means this method is used. "-" means this method is not used.

Method	MF	TCR	Precision	Recall	F1-Score
M1	-	✓	0.9032	0.9655	0.9333
M2	-	✓	0.9355	1.0000	0.9667
M3	-	✓	0.9667	1.0000	0.9831
Fusion	✓	✓	**1.0000**	**1.0000**	**1.0000**

Table 4. Comparison with different methods in the MTCR algorithm. M1 is EfficientNet-b2, M2 is ResNeSt50, M3 is ResNeSt101 and Fusion means all the three models are used together to get a robust result. MF means Model Fusion, TCR means the mix of Track splitting, Category voting and Results fusion. "✓" means this method is used. "-" means this method is not used.

be seen from the results, each model has its own advantages. After the MTCR algorithm fuses the results of multiple models, the results have been greatly improved.

5. Conclusion

In this paper, we analyze the key issues that need to be addressed urgently in the AICITY2022 Task 4 and propose an effective and excellent solution framework, which include Pre-Processing, DTC and MTCR modules. Our Pre-Processing strategy has well enhanced the quality of training images and makes up the data difference between training set and test set, which improves the accuracy of classification highly. DTC module gives a clear process flow that inputs are all frames of the test video and gets the trajectory of each product and the multi-class score of the trajectory on each frame. And the MTCR module as post-processing makes robust recognition and counting of products, and it's also suitable for other vision tasks. Extensive experiments demonstrate the effectiveness of our system. The whole framework have achieved sig- nificant improvement compared to the other teams. In the future, we would mainly focus on how to improve the speed of our method.

6. Acknowledgements

This work is supported by Chinese National Natural Science Foundation under Grants (U1931202, 62076033).

References

[1] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*, pages 3464–3468. IEEE, 2016. 2

[2] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019. 2

[3] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 2

[4] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. *Advances in neural information processing systems*, 29, 2016. 2

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 4, 7

[6] Yunhao Du, Yang Song, Bo Yang, and Yanyun Zhao. Strongsort: Make deepsort great again. *arXiv preprint arXiv:2202.13514*, 2022. 2

[7] Yunhao Du, Junfeng Wan, Yanyun Zhao, Binyu Zhang, Zhihang Tong, and Junhao Dong. Giaotracker: A comprehensive framework for mcmot with global information and optimizing strategies in visdrone 2021. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2809–2819, 2021. 2

[8] Kuan Fang, Yu Xiang, Xiaocheng Li, and Silvio Savarese. Recurrent autoregressive networks for online multi-object tracking. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 466–475. IEEE, 2018. 2

[9] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[11] Daniel J Jobson, Zia-ur Rahman, and Glenn A Woodell. A multiscale retinex for bridging the gap between color images and the human observation of scenes. *IEEE Transactions on Image processing*, 6(7):965–976, 1997. 2, 3

[12] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 4

[13] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 2

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 7

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2

[16] Nima Mahmoudi, Seyed Mohammad Ahadi, and Mohammad Rahmati. Multi-target tracking using cnn-based features: Cnnmtt. *Multimedia Tools and Applications*, 78(6):7077–7096, 2019. 2

[17] Rame Chellappa Milind Naphade et al. aicity. `https://www.aicitychallenge.org/`. 1

[18] Milind Naphade, Shuo Wang, David C Anastasiu, Zheng Tang, Ming-Ching Chang, Yue Yao, Liang Zheng, Sharifur Rahman, et al. The 6th ai city challenge. In *CVPR Workshop*, 2022. 1

[19] Siyuan Qiao, Liang-Chieh Chen, and Alan Yuille. Detectors: Detecting objects with recursive feature pyramid and switchable atrous convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10213–10224, 2021. 4, 7

[20] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 2

[21] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 2, 4

[22] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 2

[23] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 2, 4, 7

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[25] Junfeng Wan, Binyu Zhang, Yanyun Zhao, Yunhao Du, and Zhihang Tong. Vistrongerdet: Stronger visual information for object detection in visdrone images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2820–2829, 2021. 2

[26] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. Towards real-time multi-object tracking. In *European Conference on Computer Vision*, pages 107–122. Springer, 2020. 2

[27] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. 2, 4, 7

[28] Yue Yao, Liang Zheng, Xiaodong Yang, Milind Napthade, and Tom Gedeon. Attribute descent: Simulating object-centric datasets on the content level and beyond. *arXiv preprint arXiv:2202.14034*, 2022. 1

[29] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014. 2

[30] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Haibin Lin, Zhi Zhang, Yue Sun, Tong He, Jonas Mueller, R Manmatha, et al. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020. 2, 4, 7

[31] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020. 2