CheckSORT: Refined Synthetic Data Combination and Optimized SORT for Automatic Retail Checkout

Ziqiang Shi, Zhongling Liu, Liu Liu, Rujie Liu
Fujitsu R&D Center
Beijing, China
shiziqiang@fujitsu.com

Takuma Yamamoto, Xiaoyu Mi, Daisuke Uchida
Fujitsu Research Japan
Kawasaki, Japan

Abstract

In this paper, we propose a method called CheckSORT for automatic retail checkout. We demonstrate CheckSORT on the multi-class product counting and recognition task in Track 4 of AI CITY CHALLENGE 2023. This task aims to count and identify products as they move along a retail checkout white tray, which is challenging due to occlusion, similar appearance, or blur. Based on the constraints and training data provided by the sponsor, we propose two new ideas to solve this task. The first idea is to design a controllable synthetic training data generation paradigm to bridge the gap between training data and real test videos as much as possible. The second innovation is to improve the efficiency of existing SORT tracking algorithms by proposing decomposed Kalman filter and dynamic tracklet feature sequence. Our experiments resulted in state-of-the-art (when compared with DeepSORT and StrongSORT) F1-scores of 70.3% and 62.1% on the TestA data of AI CITY CHALLENGE 2022 and 2023 respectively in the estimation of the time (in seconds) for the product to appear on the tray. Training and testing code will be available soon on github.

1. Introduction

Recently, automatic checkout has become more and more popular in the field of retail industry, since it can facilitate customers to check out quickly, and at the same time help merchants to save manpower. The task of automatic retail checkout is to achieve accurate multi-class product counting and recognition based on computer vision during the customer checkout process. By leveraging a large number of real product images from different angles with location information, we can train the state-of-the-art deep neural network (DNN) to detect the location and type of products in the camera field of view to overcome this task. However, in practical applications, it is very difficult to obtain a large number of images of various products, different angles, lighting, and combinations. [27] proposes to build a 3D model for each product, and then generate infinite synthetic product images from different angles and different lighting. This happens to be the idea adopted by the organizers of AI CITY CHALLENGE 2023. What they provide to the participants is these synthesized images of a large number of products.

However, the test data provided to us is real checkout videos. A camera is placed above a white tray, and the user sequentially takes one or more products from the left bag to the tray, stays for a very short time (several seconds or less), and then takes them to the right bag. This completes the automatic checkout process. The organizer hopes that we will develop a computer vision-based solution, so that we can automatically identify how many and what kind of products the user has bought, neither more nor less. This leads to our first research topic, how to overcome the huge gap in the data discrepancy between training and real usage scenarios.

Our solution is to make the training data and the actual scene data as close as possible through pre-processing. Based on the training data provided by the organizer, we designed a controllable higher-level paradigm to generate combined product images. When generating data for training product detection and classification models, we systematically investigated the impact of different hyperparameters on performance. These hyperparameters include the number of products on the image, the occlusion between two products, and the scaling of product sizes. Based on this, the parameters most suitable for real checkouts are obtained. Since we only need to detect and recognize the products on the tray, for the test video in the actual application scenario, we will detect the tray, then remove the human hands and arms on the tray, and fill them up. In this way, we get an unoccluded test image that only contains products and the background is a tray. After this preprocessing, the training and test images are very close, that is, we can train good product object detectors and classifiers.

Besides the detector and classifier, another indispensable module is the tracker. The most important thing about a practical automatic checkout system based only on vision is accuracy. It should have 100% accurate, because supermarkets don't want to see missing checkouts during the checkout process, and customers don't want to have extra products that are wrongly checked out. Based on the detection and classification results of each frame, we need a robust association algorithm to overcome the influence of occlusion and blur, accurately determine the trajectory of a product on the tray, and infer when it enters and when it exits. Although existing tracking methods have validated effectiveness on pedestrian datasets, they are not optimal for product tracking in retail automatic checkout. We improve the previously efficient association algorithms DeepSORT [26] and StrongSORT [7] in two aspects. In the first aspect, since the center motion and aspect ratio change of the product present completely different patterns, we propose to use decomposed Kalman filters for prediction. In the second aspect, a dynamic adjustment method is adopted to maintain the feature sequence for each tracklet, so that the features added to the tracklet each time are far from the features in the current trajectory. In this way, it can be guaranteed that there are typical features from different angles in the tracklet.

The contribution of this paper is summarized as follows:

- We propose two new improvements to traditional methods, one is **synthetic data optimization** to bridge the gap between training data and actual application scenarios. Another improvement is to solve the problem that different observations cannot be predicted uniformly and the tracking efficiency is low. A product detection box association method called **CheckSORT** customized for automatic checkout scenarios is proposed.
- On the automated checkout test datasets TestA2022 and 2023, CheckSORT outperforms the previous state-of-the-art algorithms, such as DeepSORT [26] and StrongSORT [7].
- Extensive experiments are performed to analyze why our method works, and providing insights for the design of vision-based automated checkout systems.

2. Related works

A complete vision-based "multi-class product counting & recognition" system consists of five parts, namely data preprocessing, detector, tracker, classifier, and post-processing as shown in Figure 1. This section presents the work of others in each module that is most relevant to our contribution.

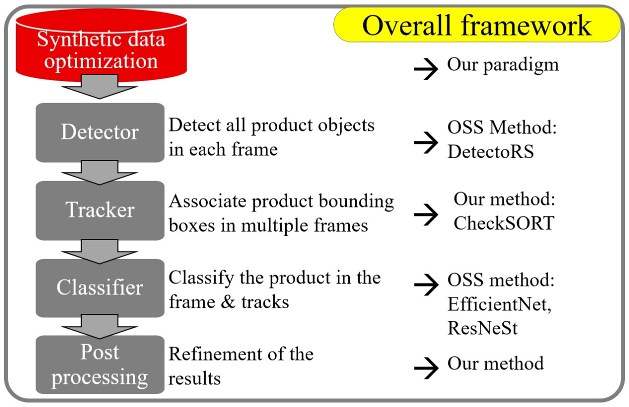
Synthetic data optimization
Overall framework
Our paradigm
Detector
Detect all product objects in each frame
OSS Method: DetectoRS
Tracker
Associate product bounding boxes in multiple frames
Our method: CheckSORT
Classifier
Classify the product in the frame & tracks
OSS method: EfficientNet, ResNeSt
Post processing
Refinement of the results
Our method

Figure 1. Block diagram of our vision-based automatic self-checkout system. Here OSS is short for 'open source software'.

2.1. Synthetic data generation

The 4th track of AI CITY CHALLENGE 2023 provides enough synthetic product images and masks, but does not provide background images. And the organizer strictly restricts the use of external data, so generating a variety of background images and synthesizing them with product images has become a task that needs to be solved first. Image generation is a field that has been studied for many years, but it is still very active with new results emerging every day. At present, researchers have proposed several approaches to generate realistic, such as generative adversarial network (GAN) [8], flow-based methods [6], variational auto-encoder (VAE) [11, 18], and diffusion-based approaches [21, 22].

2.2. Multi-object tracking

Vision-based retail automatic checkout should belong to the field of multi-object tracking, which is a basic and important task of computer vision. The multi-object tracking algorithm is divided into two paradigms, one is tracking-by-detection (first detection and then tracking), and the other is joint training of detection and tracking. Although due to the recent introduction of Transformers [25], a number of excellent joint training-based works have emerged, such as TransTrack [23], TransMOT [4], MOTR [28], etc., their performance is still slightly inferior to two-stage methods, such as ByteTrack [30], StrongSORT [7], and BoT-SORT [1]. Therefore we use a two-stage approach.

2.3. Object detection

The purpose of object detection is to detect all objects of interest in the image and determine their locations. This is a widely used and very mature field. Object detection algorithms are generally divided into two types, namely, single-stage methods and two-stage methods. Typical single-stage methods include SSD [15], RetinaNet [13], and YOLO [19], and two-stage methods include Faster-RCNN [20] and Mask-RCNN [9].

We hope to remove the influence of the hand before identifying the product, so it is required to give the mask of the hand while detecting the product. Thus we chose a multi-stage algorithm DetectoRS [17], which can do detection and segmentation at the same time, and both have achieved good performance on the Microsoft COCO dataset [14].

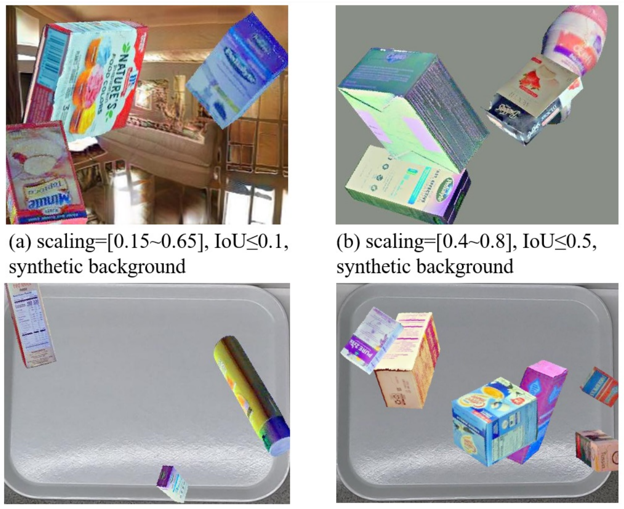
(a) scaling=[0.15~0.65], IoU≤0.1, synthetic background
(b) scaling=[0.4~0.8], IoU≤0.5, synthetic background
(c) scaling=[0.15~0.5], IoU≤0.1, real background
(d) scaling=[0.4~0.8], IoU≤0.5, real background

Figure 2. Synthetic images for training with different scaling, IoU, and background.

2.4. Image classification

After the product is detected, it needs to be identified or classified. For image classification algorithms, convolutional neural networks have dominated since AlexNet [12], such as ResNet [10], EfficientNet [24], ResNeSt [29], etc., and now Transformers [25] have become popular, such as ViT, Swin-Transformer [16], etc., and the performance of classification has gradually improved. Many of these algorithms have pre-trained models on ImageNet [5]. We fine-tune the pre-trained models on the product dataset to make them more suitable for object recognition in checkout scenarios.

2.5. Association and post-processing

After obtaining the products detected on each frame and the preliminary classification results, in order to obtain the exact category and quantity of the products at checkout, a robust association algorithm is needed to determine the precise trajectory of the products at checkout. Most of the current association methods are based on the Hungarian algorithm, such as SORT [2], DeepSORT [26], StrongSORT [7], ByteTrack [30], BoT-SORT [1], TransMOT [4], and OC-SORT [3], etc. The frameworks of these algorithms are similar, the difference lies in the different modules, such as different Kalman filters, appearance embedding, ReID features, assignment matrix calculation methods, camera compensation methods, etc. Since the movement of products and pedestrians is completely different, this paper attempts to improve the Kalman filter, the appearance feature in tracklet, and the calculation method of assignment matrix to improve the performance of the Hungarian algorithm in product trajectory prediction.

3. Controllable synthetic data optimization

Publicly available video data for automated retail checkouts is very scarce and difficult to annotate. [27] proposes to build a 3D model for each product, so that unlimited product images from various angles and lighting can be generated. The product image is used as the foreground and pasted on the background, we can get unlimited data for training product detection and classification models. Therefore, how to embed product images in the background becomes the key to the success of the whole system.

We designed a controllable synthetic data optimization scheme, in which three hyperparameters need to be adjusted, namely the number of products on the background, the occlusion degree between two products, and the scaling of product sizes. Here the occlusion degree is represented by Intersection over Union (IoU), and the scaling size is also within $[0, 1]$. These three parameters affect each other. If we put too many products on the background at one time and keep the original size of the products, the occlusion between the products will inevitably increase. The user can specify the range of these parameters, such as 1-6 products, the occlusion degree is less than 0.3, and the zoom size is within $[0.15, 05]$ to generate different training data. We adjusted different hyperparameters, conducted ablation experiments, and found that with regard to occlusion, it is best to either have an upper bound of 0.5, or not have occlusion; and the scaling size should be set around 0.5 as much as possible. At this time, the system will achieve better performance. For detailed experiments, please refer to Section 5.1.

Comparison with real image as background We also conducted experiments to investigate the performance difference caused by using synthetic images and real images as backgrounds. Experimental results in Section 5.2 show that using real images improves performance. This also tells us that for automated retail checkouts, synthetic data is only a compromise, and it is better to use real data when conditions permit. Figure 2 shows some synthetic images for training with different scalings, IoUs, and backgrounds.

Usage of these data DetectoRS [17] is used as the product detection model, which is pre-trained on the Microsoft COCO dataset [14]. The data prepared above are used to fine-tune this pre-trained model. And in order to obtain

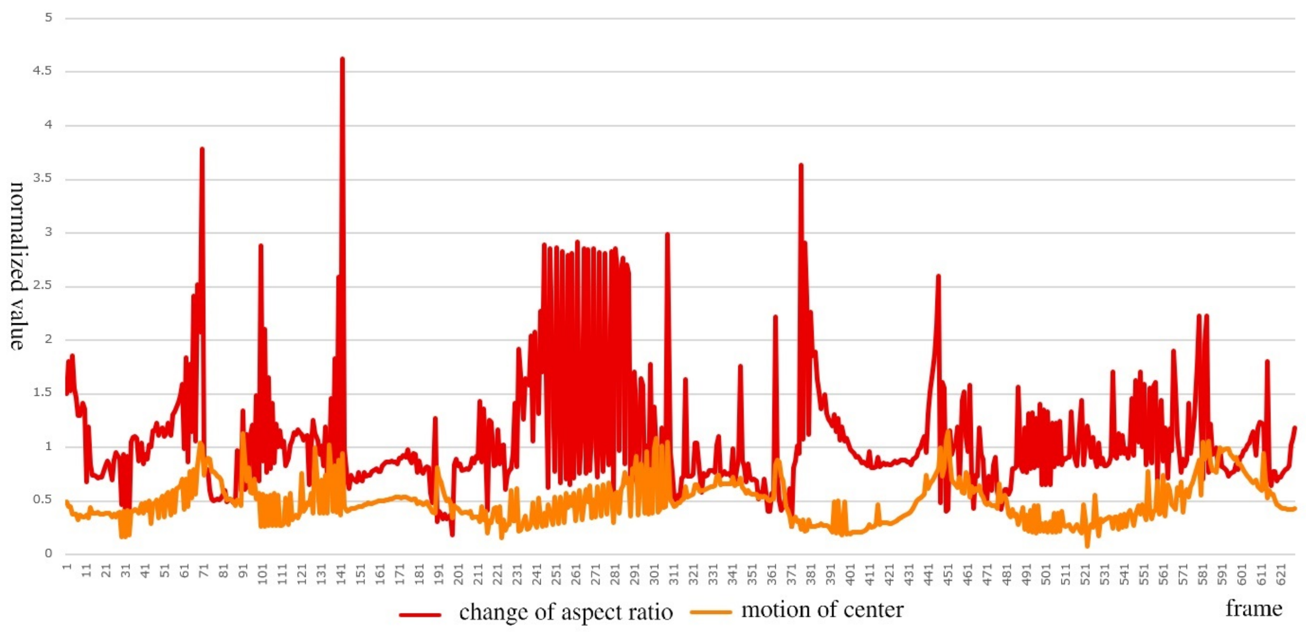

Figure 3. The orange one is the distance curve from the product center to the origin. The red one is the curve of aspect ratio over time. There are multiple product checkouts in this figure.

a robust classifier, in addition to the data provided by the organizer, product images with different backgrounds are extracted from the optimized synthetic data and used for fine-tuning. Three types of classification models are used, namely EfficientNet, ResNeSt-50, and ResNeSt-100 pre-trained on ImageNet [5].

4. CheckSORT

The association algorithm we use is based on StrongSORT [7] with several improvements, including decomposed Kalman filtering and dynamic tracklet feature sequences.

4.1. Fundamentals of SORT-like algorithms

The input to SORT-like algorithms is continuous frames with product detection results, and the output is several clean product checkout trajectories. All trajectories are initialized to be empty. When faced with the detection result of the first frame, each bounding box is initialized as a trajectory, including the mean and variance measured in the initialization Kalman filter (generally highly correlated with the bounding box). All trajectories are in one of three states, tentative (all newly created trajectories are tentative), confirmed (if several consecutive frames that match the detection box, it is confirmed, otherwise it is deleted), deleted (if a track has not been matched by a detection box for a long time, then delete).

After the trajectories are initialized, it starts to process one frame at a time. For each frame, first do *prediction*. The Kalman filter predicts the position (its mean and variance) of the bounding boxes that may appear in the current frame according to the historical state of the trajectory and the motion equation. Next comes the *updating*. For the confirmed trajectory, the matching between current bounding box and confirmed trajectory is realized through the cosine distance-based cost matrix and the Hungarian algorithm. For unconfirmed and unmatched trajectories, a second match is performed by the Mahalanobis distance between the predicted state and the measured signal (current detection box) through Kalman filtering. Then update the state of all trajectories based on the two matching results.

4.2. Decomposed Kalman filter

In most previous association algorithms, a single Kalman filter is used to model and predict $\mathbf{x} = [x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h}]$ by

$$\mathbf{x}_k = \boldsymbol{A}\mathbf{x}_{k-1} + \mathbf{w}_{k-1}, \text{(transition equation)} \quad (1)$$

$$\mathbf{z}_k = \boldsymbol{H}\mathbf{x}_k + \mathbf{v}_k, \text{(measurement equation)} \quad (2)$$

where $\boldsymbol{A}$ is the motion matrix, $\boldsymbol{H}$ is the updating matrix, $\mathbf{w}_k$ and $\mathbf{v}_k$ are the process noise and measurement noise with covariance matrix of $\boldsymbol{Q}$ and $\boldsymbol{R}$ respectively

$$p(\mathbf{w}) \sim \mathcal{N}(\cdot; \mathbf{0}, \boldsymbol{Q}) \quad (3)$$

$$p(\mathbf{v}) \sim \mathcal{N}(\cdot; \mathbf{0}, \boldsymbol{R}). \quad (4)$$

Here $\mathcal{N}$ is the normal distribution, x and y are the center point of the bounding box, a is the aspect ratio, h is the

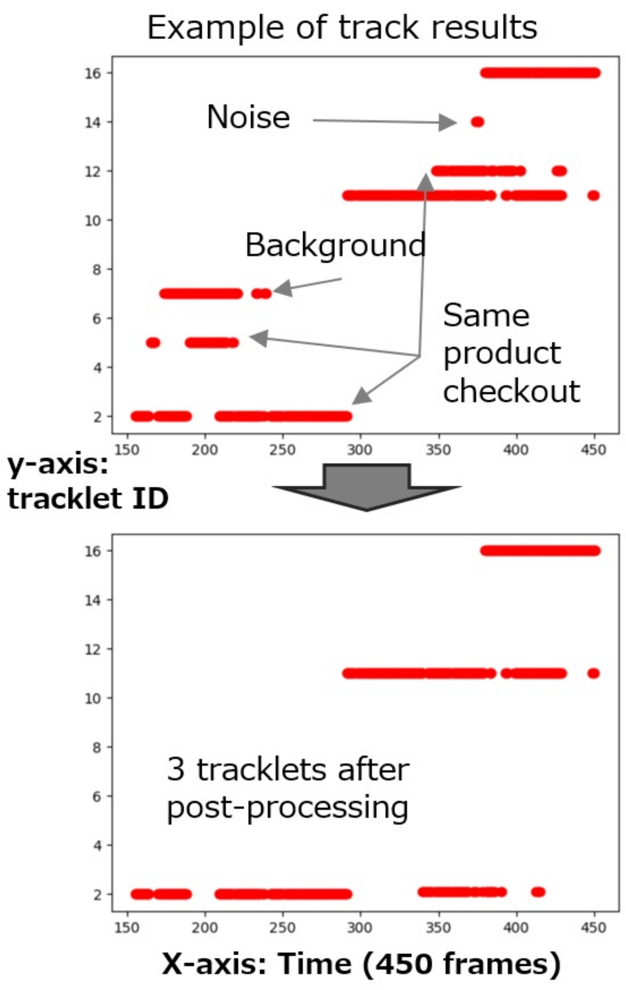

Figure 4. The post-processing of CheckSORT can effectively reduce false detection.

height, and the " $\dot{}$ " represent the first derivative, which is the rate of change. But in fact, $[x, y]$ and $[a, h]$ show completely different motion patterns. The movement of products at checkout can be decomposed, one is the smooth movement of the center $[x, y]$, and the other is a nearly independent rigid body motion, such as the rotation of a product. The translational movement is relatively simple, almost linear, and the rotation corresponds to the nonlinear sharp change of $[a, h]$ of the bounding box. Figure 3 shows the comparison between the center movement and the aspect ratio change curve of a product in a video from TestA 2022. It can be seen that the change in aspect ratio is much larger than the movement of the center. We therefore propose the decomposed Kalman filter (DKF), which can model $\mathbf{p} = [x, y, \dot{x}, \dot{y}]$ and $\mathbf{b} = [a, h, \dot{a}, \dot{h}]$ of products separately. The Kalman filters that characterize the position $\mathbf{p}$ and the aspect ratio $\mathbf{b}$ share the same motion and update matrices, they are

$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ and } \boldsymbol{H} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad (5)$$

respectively. However, the variances of the initial measurements of the two Kalman filters are different. The variance of position $\mathbf{p}$ is highly correlated with the product, while the variance of the aspect ratio is completely random, and we set its initial value to $p(a) \sim \mathcal{N}(\cdot; 0, 0.01)$.

Distance matrix When matching the detection boxes of the current frame and the historical trajectory, it is necessary to obtain the pairwise distance between them. In our DKF, we need to calculate the distance matrix of position and aspect ratio separately, they are

$$d^{(p)}(i,j) = (\mathbf{p}_j - \mathbf{t}_i^p)\boldsymbol{P}_i^{-1}(\mathbf{p}_j - \mathbf{t}_i^p), \quad (6)$$

and

$$d^{(b)}(i,j) = (\mathbf{b}_j - \mathbf{t}_i^b)\boldsymbol{B}_i^{-1}(\mathbf{b}_j - \mathbf{t}_i^b), \quad (7)$$

where $(\mathbf{t}_i^p, \boldsymbol{P}_i)$ and $(\mathbf{t}_i^b, \boldsymbol{B}_i)$ are the projection parameters of the i-th tracklet of the position and aspect ratio in the corresponding measurement space, and $\mathbf{p}_j$ and $\mathbf{b}_j$ are the position and aspect ratio related measurement of the j-th bounding box.

Since DeepSORT [26], different appearance features start to be applied in the distance calculation and matching between the tracklet and the detected object. We extract the appearance features of products and trajectories through an EfficientNet [24] pre-trained on ImageNet [5] and fine-tuned on the product dataset.

Dynamic exponential moving average (EMA) tracklet feature sequences Different from DeepSORT [26] maintaining a gallery of 100 continuous descriptors for each track and StrongSORT [7] using EMA to filter the noise of the detection results, we propose a dynamic gallery of variable-length non-continuous EMA features. Only when the normalized EMA feature $\mathbf{e}_i^T$ of the currently detected object is sufficiently different from the EMA feature in the current gallery $\mathcal{E}_i = \{\mathbf{e}_i^t | t = 1, ..., L_i\}$ (L_i is the size of the gallery), it will be added to the gallery. Then the appearance distance matrix can be computed by

$$d^{(a)}(i,j) = \min\{1 - \mathbf{e}_i^T \mathbf{e}_j^t | \mathbf{e}_j^t \in \mathcal{E}_j\}. \quad (8)$$

Gating and cost matrix In this way, we get three distance matrices. Similar to DeepSORT, we can get a gating matrix for each matrix and the final cost matrix. The difference is that our cost matrix is a weighted sum of three different matrices

$$c(i,j) = \lambda_p d^{(p)}(i,j) + \lambda_b d^{(b)}(i,j) + \lambda_a d^{(a)}(i,j). \quad (9)$$

4.3. Post refinement

After getting all the trajectories through the above algorithm, we need to do some refinement to improve the accuracy. Under some empirical tuning, we specify the following rules to process these raw trajectories:

- If the track is very short, or the track is classified as a background class and does not belong to any product class, delete it.
- If the track has a gap greater than half a second in the middle, it is broken into two traces.
- If the classification results of several trajectories are the same, and the distance between two trajectories is less than 3 seconds, these trajectories are merged.

Figure 4 shows an example, it can be seen that post-processing is very important and can significantly improve the results.

5. Experiments

We noticed that the 4th track of AI CITY CHALLENGE 2023 is slightly different from previous year. This year we need to predict which frame in the video the products start to appear, while last year we only need to predict the second from which they appear. Of course, we hope to verify the algorithm on the test data of AI CITY CHALLENGE 2023, but these data have no ground truth. So we settled for the next best thing, and manually label the appearance time (seconds) of products for the two TestA datasets in 2022 and 2023, and verified the algorithm on these two datasets. Same to the evaluation method in 2022, we use F1-score as the criterion, where true positive (TP) identifications are considered when objects are correctly identified in the region of interest. False negative (FN) identifications are ground-truth objects that are not correctly recognized.

The organizer provided 116,500 synthetic images from 116 3D objects based on the pipeline from [27].

5.1. Effect of synthetic data optimization

Table 1. Performance comparison on TestA 2022 under different IoUs (in the generation of training set). When generating the training set, the scaling size is fixed in the range of 0.15∼0.5. The tracker used is DeepSORT.

IoU≤	0.25	0.2	0.15	0.1	0.05
Recall(%)	58.6	44.8	55.1	58.6	58.6
Precision(%)	58.6	39.3	50	54.8	44.7
F1-score(%)	**58.6**	41.9	52.4	56.7	50.7

In Section 3, there are only two hyperparameters that can be adjusted, one is the upper bound of IoU, and the other is the range of scaling. Tables 1 and 2 show the recognition performance on TestA 2022 and 2023 under different IoUs with fixed scaling in the range of 0.15∼0.5. Both experiments show that larger upper bounds on IoUs lead to better performance. That is to say, the more occlusions between two products and the more complex data generated are more conducive to multi-object detection, classification and tracking. Tables 3 and 4 show the different performances under different scaling ranges when the upper bound of IoU is fixed at 0.1. It can be seen that scaling around 0.55 will produce better performance. Larger scaling cannot generate enough training data, and too small scaling will not match the actual situation.

Table 2. Performance comparison on TestA 2023 under different IoUs (in the generation of training set). When generating the training set, the scaling size is fixed in the range of 0.15∼0.5. The tracker used is DeepSORT.

IoU≤	0.25	0.2	0.15	0.1	0.05
Recall(%)	53.6	53.6	47.3	52.6	56.8
Precision(%)	48.5	46.3	38.1	47.6	45
F1-score(%)	**51**	49.7	42.2	50	50.2

Table 3. Performance comparison on TestA 2022 under different scalings (in the generation of training set). When generating the training set, the IoU is fixed to be upper bounded by 0.1. The tracker used is DeepSORT.

Scaling =	0.15∼0.5	0.15∼0.55	0.15∼0.60
Recall(%)	58.6	55.1	58.6
Precision(%)	54.8	61.5	50
F1-score(%)	56.7	**58.2**	53.9

Table 4. Performance comparison on TestA 2023 under different scalings (in the generation of training set). When generating the training set, the IoU is fixed to be upper bounded by 0.1. The tracker used is DeepSORT.

Scaling=	0.15∼0.5	0.15∼0.55	0.15∼0.60
Recall(%)	52.6	53.6	51.5
Precision(%)	47.6	47.2	44.1
F1-score(%)	50	**50.2**	47.5

5.2. Synthetic vs. real background

According to the known how obtained in the previous section, we tested the performance of the model using real or synthetic images as the background on TestA 2022 and 2023 under two configurations of IoU upper bound and scaling. It can be seen from Tables 5 and 6 that no matter what configuration or test data set, using real images as the background will result in better performance. Especially under the configuration of IoU$\leq$0.5, scaling = 0.4$\sim$0.8 on TestA 2022, using real images as the background can improve the performance by about 8%.

Table 5. Performance comparison under IoU$\leq$0.5, scaling = 0.4$\sim$0.8. The tracker used is DeepSORT.

	%	TestA 2022	TestA 2023
Synthetic	Recall	58.6	49.4
	Precision	62.9	58.7
	F1-score	60.7	53.7
Real	Recall	65.5	52.6
	Precision	73.0	57.4
	F1-score	**69.0**	**54.9**

Table 6. Performance comparison under IoU$\leq$0.5, scaling = 0.3$\sim$0.75. The tracker used is DeepSORT.

	%	TestA 2022	TestA 2023
Synthetic	Recall	55.2	49.4
	Precision	64	58.7
	F1-score	59.2	53.7
Real	Recall	65.5	57.8
	Precision	73.0	58.5
	F1-score	**69.0**	**58.2**

5.3. CheckSORT

In Tables 7 and 8 we compare the performance of DeepSORT, StrongSORT, and CheckSORT on two datasets and two configurations. Whether compared to DeepSORT or StrongSORT, CheckSORT has different degrees of improvement.

Table 7. Performance comparison under different trackers (D. for DeepSORT, S. for StrongSORT, C. for CheckSORT) with IoU$\leq$0.5, scaling = 0.3$\sim$0.75.

	TestA 2022			TestA 2023		
%	D.	S.	C.	D.	S.	C.
Recall	65.5	65.5	65.5	57.8	60	62.1
Precision	73	70.3	76.0	58.5	60	62.1
F1-score	69.0	67.8	**70.3**	58.2	60	**62.1**

On the leaderboard Table 9 is the official ranking of the current results (the final ranking still needs to submit the code for testing on TestB), and we are ranked fifth. It looks like we still have a lot of potential for improvement.

Table 8. Performance comparison under different trackers (D. for DeepSORT, S. for StrongSORT, C. for CheckSORT) with IoU$\leq$0.5, scaling = 0.4$\sim$0.8.

	TestA 2022			TestA 2023		
%	D.	S.	C.	D.	S.	C.
Recall	65.5	65.5	65.5	52.6	51.5	53.6
Precision	73.0	73.0	76.0	57.4	53.8	58.6
F1-score	69.0	69.0	**70.3**	54.9	52.6	**56.0**

Table 9. Current leaderblead of AI CITY CHALLENGE 2023.

Rank	Team ID	Team Name	Score
1	33	SKKU Automation Lab	0.9792
2	21	BUPT_MCPRL	0.9787
3	13	Zebras	0.8254
4	1	SCU Anastasiu Lab	0.8177
5	23	Fujitsu R&D Center	0.7684
6	200	Centific	0.6571
7	65	dtb2023	0.4757
8	64	Fu	0.4215
9	9	HCMIU-CVIP	0.3837
10	68	UTE_AI	0.3441

6. Conclusion

The goal of this paper is to propose a solution to the competition. To bridge the large gap between training and testing data, we propose a synthetic training data optimization paradigm. And we also propose a customized CheckSORT tracking algorithm based on the particularity of product checkout scenarios. Experimental results show that both methods can improve the recognition performance of automatic self-checkout.

References

[1] Nir Aharon, Roy Orfaig, and Ben-Zion Bobrovsky. Bot-sort: Robust associations multi-pedestrian tracking. *arXiv preprint arXiv:2206.14651*, 2022. 2, 3

[2] 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. 3

[3] Jinkun Cao, Xinshuo Weng, Rawal Khirodkar, Jiangmiao Pang, and Kris Kitani. Observation-centric sort: Rethinking sort for robust multi-object tracking. In *2023 IEEE conference on computer vision and pattern recognition*. IEEE, 2023. 3

[4] Peng Chu, Jiang Wang, Quanzeng You, Haibin Ling, and Zicheng Liu. Transmot: Spatial-temporal graph transformer for multiple object tracking. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 4870–4880, 2023. 2, 3

[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. 3, 4, 5

[6] Laurent Dinh, David Krueger, and Yoshua Bengio. Nice: Non-linear independent components estimation. *arXiv preprint arXiv:1410.8516*, 2014. 2

[7] Yunhao Du, Zhicheng Zhao, Yang Song, Yanyun Zhao, Fei Su, Tao Gong, and Hongying Meng. Strongsort: Make deepsort great again. *IEEE Transactions on Multimedia*, 2023. 2, 3, 4, 5

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, volume 27, pages 2672–2680, 2014. 2

[9] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 3

[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. 3

[11] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR 2014 : International Conference on Learning Representations (ICLR) 2014*, 2014. 2

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, pages 84–90, 2017. 3

[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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 3

[15] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016. 2

[16] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 3

[17] 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. 3

[18] Ali Razavi, Aaron Van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with vq-vae-2. *Advances in neural information processing systems*, 32, 2019. 2

[19] 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. 3

[20] 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. 3

[21] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 2

[22] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2020. 2

[23] Peize Sun, Jinkun Cao, Yi Jiang, Rufeng Zhang, Enze Xie, Zehuan Yuan, Changhu Wang, and Ping Luo. Transtrack: Multiple object tracking with transformer. *arXiv preprint arXiv:2012.15460*, 2020. 2

[24] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 3, 5

[25] 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, 3

[26] 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, 3, 5

[27] 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, 3, 6

[28] Fangao Zeng, Bin Dong, Yuang Zhang, Tiancai Wang, Xiangyu Zhang, and Yichen Wei. Motr: End-to-end multiple-object tracking with transformer. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXVII*, pages 659–675. Springer, 2022. 2

[29] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2736–2746, 2022. 3

[30] Yifu Zhang, Peize Sun, Yi Jiang, Dongdong Yu, Fucheng Weng, Zehuan Yuan, Ping Luo, Wenyu Liu, and Xinggang Wang. Bytetrack: Multi-object tracking by associating every detection box. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*, pages 1–21. Springer, 2022. 2, 3