Cafeteria Food Automatic Recognition

Kuo Yuan-Ting
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
g2502828308@gmail.com

Lu Kuan-Yu
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
a796106676@icloud.com

Chen Han-Sheng
Department of Materials Science and Engineering
National Tsing Hua University
Hsinchu, Taiwan
hanson.chen9005@gmail.com

Huang Hong-Siang
Interdisciplinary Program of Engineering
National Tsing Hua University
Hsinchu, Taiwan
blog40813@gmail.com

Hsiao Yu-Chun
Interdisciplinary Program of Nuclear Science
National Tsing Hua University
Hsinchu, Taiwan
mamiehsiao@gapp.nthu.edu.tw

Melinda Khosasih
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
melindakhosasih@gmail.com

***Abstract*—This project addresses the issue between a standard cafeteria pricing and actual charges made by cashiers, highlighting the need for a more precise and transparent pricing system. Our objective is to enhance cafeteria pricing fairness through the implementation of an Artificial Intelligence (AI) model, specifically YOLOv7. By utilizing real-time object detection, our proposed model aims to reduce inconsistencies in pricing, ensuring customers are charged appropriate prices for their meal orders. Furthermore, our model implementation not only reduces the inconsistencies in pricing, but also speeds up the checkout process, addressing the issue of long queues during peak hours. This project outlines a significant step towards simplifying cafeteria pricing and checkout, fostering fairness and efficiency.**

***Index Terms*—YOLOv7, Object Detection**

I. Introduction

In Taiwan, some of the cafeteria evaluate the price of the meal using different ways, which leads to the price evaluation of the cafeteria meals may differ between individual's point of view. Therefore, it might cause some unfair and awkward situation. To solve this embarrassing problem we decide to develop a model that can give the most precised and acceptable pricing of the meal using computer vision and deep learning techniques.

There was already a project detecting the price of cafeteria food using YOLOv3 [5]. Although YOLOv3 has been a powerful model to in computer vision task, we aim to reach much higher accuracy by finding other object detection model to address the unfair pricing problem to reach a more convincing results. Therefore, we choose YOLOv7 for our model development, due to the higher accuracy and shorter execution time.

Our goal is to develop a model capable of delivering accurate identification of cafeteria food item, and determine the price based on the assigned pricing rule instantly. The model aims to ensure fairness and consistency in pricing across different instances.

II. Methods

A. Data

Our goal is to estimate the value of meals through photos of cafeteria food according to the main course and the total number of side dishes. Therefore, we processed our data with the following four steps.

1) We combined images from [5] with photos collected from friends and customers dining at the cafeteria. With these two sources, we could ensure the amount and variety of the data.

Fig. 1. Example of original image.

2) Before training, we performed data augmentation by taking photos from different angles, directly rotating the image, or rearranging the dishes (Fig. 1 to 3). Additionally, we also eliminated unlabeled dishes that had infrequent occurrences to address the issues of data imbalance and insufficient data volume. In this way, it prevented overfitting effectively in our model.
3) Subsequently, we labeled each image by using LabelImg [1], which enabled us to gather the name of each dish into classes.
4) Last but not least, to fit the input format for YOLOv7 [3], we slightly modified the code [2], used for converting files, found on GitHub to successfully convert our data into txt files(Fig. 4 and 5). Afterward, we proceeded with training.

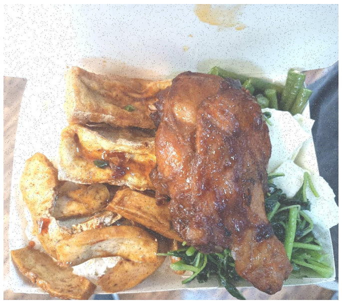


Fig. 2. Example of augment image1.

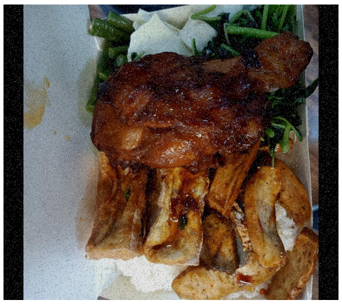


Fig. 3. Example of augment image2.

B. Code

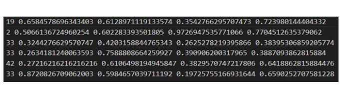
YOLOv7 [3] introduces extended ELAN (E-ELAN) [4], which achieves the continuous enhancement of network learning capabilities without disrupting gradient path states through techniques such as expand cardinality, shuffle cardinality, merge cardinality, and more. Fig. 6 shown structure of E-ELAN.

Before entering the model training phase, we prefer to divide the data into three sets: training, validation, and testing sets (Fig. 7). The training set is used directly to train the model; the validation set is employed to evaluate the model's performance after each training iteration on the training set. Once the model performs reasonably well on the validation set, the final testing set is utilized for blind testing of the model.

1.xml

```xml
<annotation>
    <folder>anns</folder>
    <filename>1.jpg</filename>
    <path>data/train/imgs/</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>1772</width>
        <height>2362</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>carrot_eggs</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>474</xmin>
            <ymin>539</ymin>
            <xmax>1522</xmax>
            <ymax>1966</ymax>
        </bndbox>
    </object>
</annotation>
```

Fig. 4. Example of xml.



```
19 0.6584578696343403 0.6128971119133574 0.3542766295707473 0.723980144404332
2 0.5066136724960254 0.602283393501805 0.9726947535771066 0.7704512635379062
33 0.3244276629570747 0.4203158844765343 0.2625278219395866 0.38395306859205774
33 0.2634181240063593 0.7588808664259927 0.390906200317965 0.3887093862815884
42 0.2721621621621621 0.6106498194945847 0.3829570747217806 0.6418862815884476
33 0.8720826709062003 0.5984657039711192 0.19725755166931644 0.6590252707581228
```

Fig. 5. Example of txt.

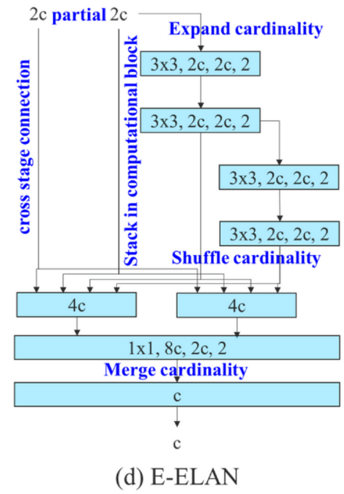


Fig. 6. Structure of E-ELAN.

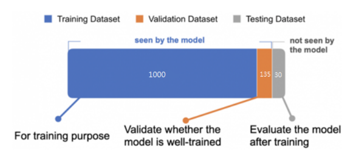


Fig. 7. Dataset overview.

Next, we proceed to train the model with the 1000 data points in the training set. Rather than specifying a fixed number of training iterations, we implement the early stopping mechanism. This means that training stops when the model's loss does not decrease for five consecutive epochs. In other words, the model's learning has saturated, and additional training iterations won't further reduce the loss. At this point, we obtain the model that represents a relatively optimal solution. This training session took almost 18 hours to complete on Google Colab, utilizing the T4 GPU.

III. Results

This is the performance outcomes of our model, we have three version.

A. First version

In the first version (Fig. 8), we can observe that there are a lot of different classes, but we don't have enough data to recognize each type of meal. As a result, the outcome of the confusion matrix is not ideal.

B. Second version

In the second version, we made some changes. We reduced the number of classes by combining all side dishes into one

Fig. 8. First version of confusion matrix.

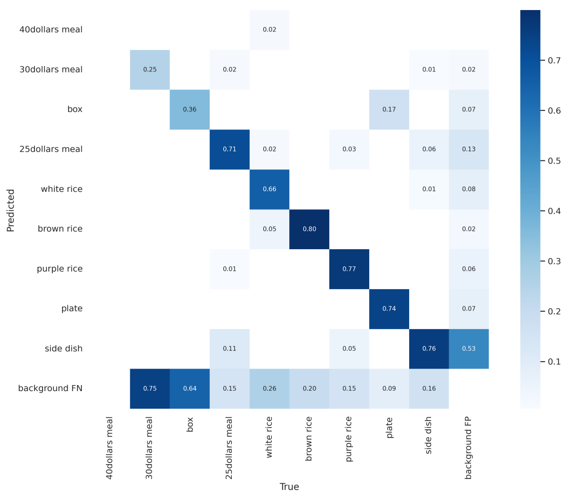


Fig. 9. Second version of confusion matrix.

class called "side dish" and grouped main meals based on their prices, categorizing into "25 dollar meal", "30 dollar meal", and "40 dollar meal". After applying these changes, we observed a notable improvement in the confusion matrix shown in Fig. 9.

C. Final version

In the final version, as shown in Fig. 10, we adjust the iteration, stopping training when the model's loss no longer decreased for 5 consecutive epochs. As you can see, it performed better compared to previous models.

D. Results

In Fig. 11 and 12, the left side shows the original input image, the upper right side shows the output image and lower right side shows the corresponding price. The output (image and price) can be obtained in real-time, and we use Hugging Face Spaces as the demo platform.

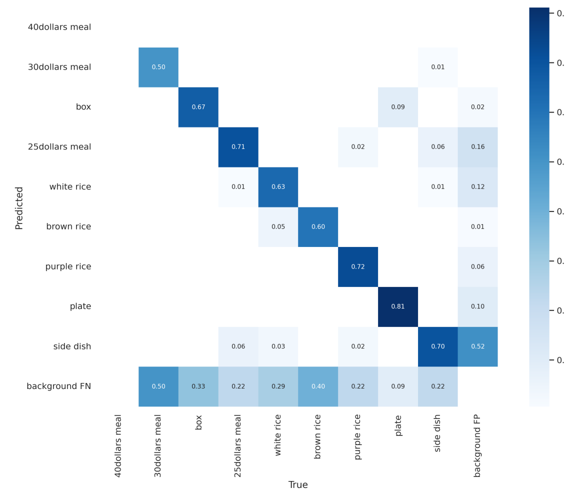


Fig. 10. Final version of confusion matrix.

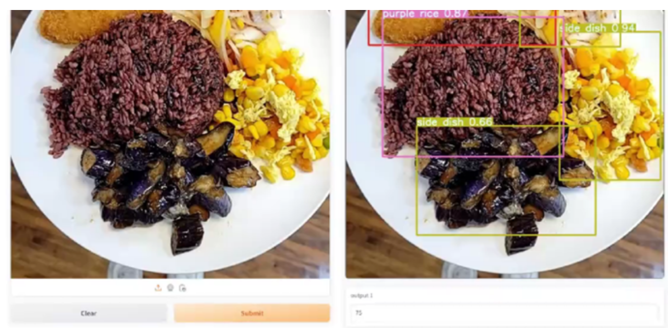


Fig. 11. Example of meal1 and its price.

Fig. 12. Example of meal2 and its price.

IV. Discussion/Conclusion

The reason we wanted to implement a tool for automatically calculating cafeteria prices is that we often heard about inconsistencies in prices within the campus cafeteria. The subjective consciousness of the checkout personnel often affects pricing, potentially resulting in unfair treatment. Therefore, our initial idea was to address this issue through a tool that ensures fair treatment for everyone.

When initially choosing a model to use, we opted for YOLOv7 [3]. Its code is open-source, and there are numerous examples and tutorials available online. Additionally, YOLOv7 [3] demonstrates high accuracy in training data, making it a convenient and efficient choice for us.

During the training phase, we deployed YOLOv7 [3] on Google Colab. However, due to the limitations of the free GPU usage, we frequently exceeded the free quota, which meant we had to wait for a few hours before continuing. This process consumed a significant amount of time, but fortunately, we ultimately succeeded in training the desired results.

Although we achieved successful results, we believe that there is still room for improvement. Insufficient data resulted in certain main dishes being inaccurately identified, causing confusion with side dishes. However, accurate identification and correct pricing were achieved for more popular main dishes.

Through completing this project during the semester , we learned how to utilize machine learning to improve daily life convenience by completing this project. The automated cafeteria price calculation tool not only speeds up the checkout process but also prevents fairness issues that may arise from inconsistent pricing. In addition, we learned about teamwork, task delegation, setting agendas for meetings, and ensuring equitable distribution of workload. These are the valuable lessons that this project has imparted to us.

V. Data and Code Availability

Our github : please check the master in branches not main https://github.com/110062306/ML-Final-project/tree/master

Data available :

- Original data : https://drive.google.com/drive/u/1/folders/1hBlf0RE-n4SQvG3TB5R9vAsyc8LkWLCa
- Revised data : https://github.com/110062306/ML-Final-project/tree/master/data

VI. Author Contribution Statements

Project Manager & Programmer: Kuo Yuan-Ting

- study design
- programming
- model construction
- **Total:** 16.66%

Programmer: Lu Kuan-Yu

- study design
- programming
- model construction
- **Total:** 16.66%

Data Collector: Chen Han-Sheng

- data collection
- data interpretation
- image processing
- **Total:** 16.66%

Data Collector: Huang Hong-Siang

- data collection
- data interpretation
- image processing
- **Total:** 16.66%

Researcher & Tester & API: Hsiao Yu-Chun

- study design
- API design
- model testing
- **Total:** 16.66%

Researcher & Tester & API: Melinda Khosasih

- study design
- API design
- model testing
- **Total:** 16.66%

References

[1] Karndeep Singh. Train yolov5 on custom dataset fast easy - object detection — deep learning — data science. https://www.youtube.com/watch?v=2nR2e4J4ZaI, 2021.

[2] uncle ll. Yolo image detection dataset format conversion xml and txt formats are mutually converted. https://blog.csdn.net/uncle_ll/article/details/116197413, 2022.

[3] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[4] Chien-Yao Wang, Hong-Yuan Mark Liao, and I-Hau Yeh. Designing network design strategies through gradient path analysis. *Journal of Information Science and Engineering*, 2023.

[5] Andy Wu. Chinese cafeteria food recognition. https://github.com/andywu0913/Chinese-Cafeteria-Food-Recognition/tree/experiencor, 2020.