Kiwibot Ai Vision & LLM Engineer – Technical Challenge

Felipe Rojas Arredondo
March 3, 2025

• Task 1: Computer Vision Task – Object detection & tracking

For the solution of this task a real-time object detection and counting system was developed. The prototype was trained to identify 3 types of food items: burgers, pizzas, and fries; using the YOLO (You Only Look Once) deep learning model and a public dataset from fast food found in the next Link. The system detects objects via a webcam feed and displays the count of detected objects dynamically through a bar graph. At last, when the process is finished, a file called "detection_results.txt" is generated. It will contain a JSON with the summary of the detection process at different time stamps.

1. Model Training Process

To train the model two different iterations were made. The training parameters for each of the iterations are presented in Table 1. The first iteration was trained only to detect two type of objects, while in the second integration a new category was introduced.

Table 1: Training parameters

Iteration	Pre-trained model	Training dataset size per category	Validation dataset size per category	Epochs	Training batch size
1	Yolov8n	30	6	50	8
2	Yolov8n	102	20	70	8

2. Training results

When finishing the training process, both models showed very good results being able to identify the stablished objects with high accuracy and precision. Since the model from iteration 2 is more complete, only its results will be shown in this report. Nevertheless, the results of iteration 1 are available on the training code.

Image 1: Normalized confusion matrix for iteration1

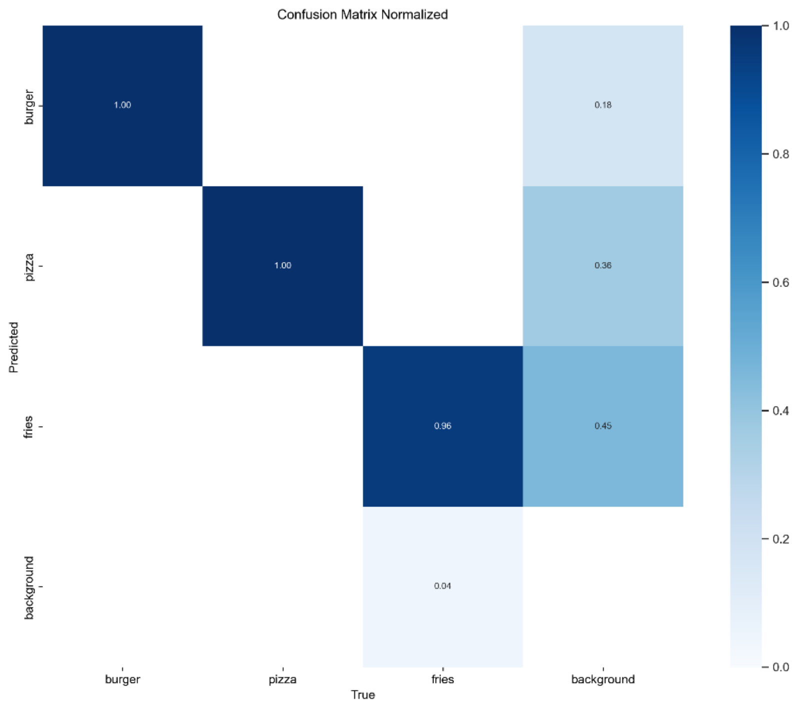
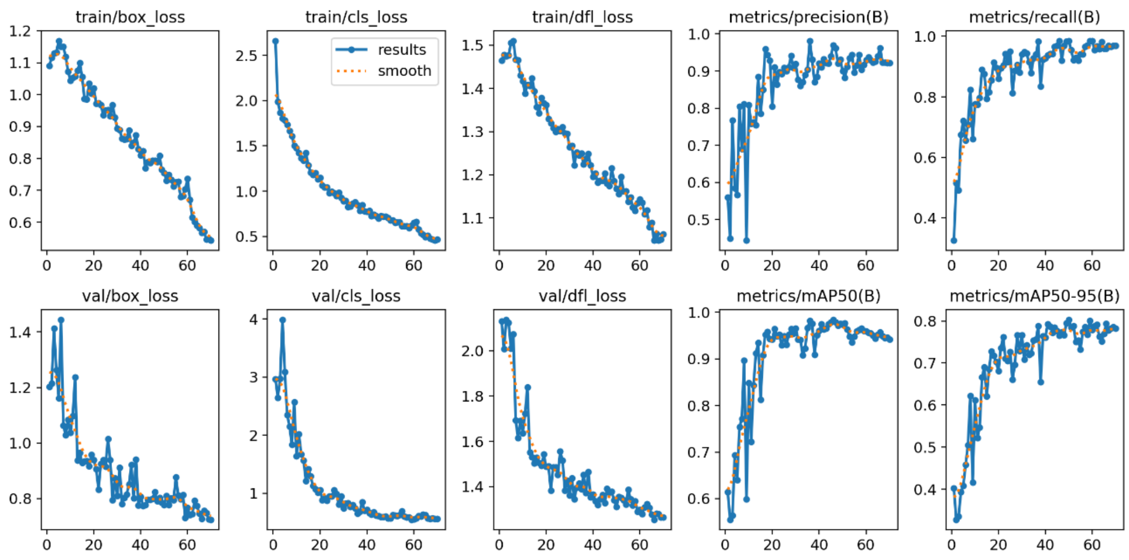


Image 2: training results for iteration1



3. Real-Time Detection Implementation

After training the model, it was used for a real time detection system using the computer camara. The system detects and counts each element that belongs to one of the categories. At the same time, a bar graph shows in real time the number of objects counted. Finally, when the process finishes, a text file called “detection_results.txt” is created summarizing the detection process in a JSON. An example of the results presented in the txt file are shown below. To view a demo of the prototype working, access the video “Solution_demo.mp4”.

[{

"Time": "2025-03-02 19:22:35",

"burger": 0,

"pizza": 0,

"fries": 0

},

{

"Time": "2025-03-02 19:22:35",

"burger": 0,

"pizza": 0,

"fries": 0

}]

- ## Task 2: LLM Task – Structured Data Extraction & Processing

For the solution of this task an Azure OpenAI's Retrieval-Augmented Generation (RAG) model was trained to extract specific information from unstructured kitchen logs. The extraction process is done following the next process:

1. **Text extraction**

The first step is to extract the text of each of the files. The code allows the extraction of PDF or Word files.

2. **Text preprocessing**

The second step is to preprocess the extracted text in order to delete unwanted characters.

3. **LLM text analysis**

Third, the extracted text is inputted to the RAG LLM model to convert the relevant data into a JSON. The parameters used for the LLM model are shown in Table 2.

Table 2: LLM parameters

Parameter	Value
Model	GPT 4o mini
Temperature	0
Frequency penalty	0
Presence penalty	0

Additionally, the used prompt was:

“Instructions:
1. Based on the given restaurant order, extract the most relevant information of the order.
2. Format the response as a Python dictionary.
3. Ensure that all response is entirely in English, even if the input is in another language.

4. Organize all keys of the dictionary as text with single quotes and avoid doing it as python lists.
5. Adhere to the following structure:

```
{   'Restaurant': 'Name of the restaurant',
    'Order ID': 'All numerical and number sequence corresponding to the ID of the order',
    'Food item names': 'ordered item 1, ordered item 2, ordered item 3, ordered item 4',
    'Actions taken': 'action 1, action 2, action 3, actions 4',
    'Time of action': 'date and time the order was made',
    'Total Price': 'total price of the order'
}”
```

4. LLM result preprocessing

When the code is runed, the result of the LLM will be a python dictionary containing the extracted information of each file. The final step is to process the results in order to delte unwanted characters and to transform it to a JSON format. When the code finishes execution, the JSON is exported to a txt file named “extractor_results.txt”. An example of the results is shown below. To view a demo of the prototype working, access the video “Solution_demo.mp4”.

```
[
  {
    "Restaurant":"Sunshine Caf ",
    "Order ID":"1003",
    "Food item names":"Chicken Sandwich, Fruit Smoothie, Brownie",
    "Actions taken":"Load, Prepare",
    "Time of action":"03\/02\/2025 12:30 PM",
    "Total Price":"$17.47"
  },
  {
    "Restaurant":"The Veggie Spot",
    "Order ID":"1010",
        "Food item names":"Quinoa Salad, Avocado Toast, Berry Smoothie",
        "Actions taken":"Load, Prepare",
        "Time of action":"03\/02\/2025 2:15 PM",
        "Total Price":"$22.47"
    }
]
```