

Real Time Deep Learning Model for Food Item Identification and Recipe Data Generation

Sanshruth Ralhan¹, Ronit Kumar Das^{1,*}, and Sweta Srivastava²

¹ Research Scholar ASET, Amity University Uttar Pradesh, Noida. 201303, India

² Assistant Professor, ASET, Amity University Uttar Pradesh, Noida. 201303, India

*Corresponding Author Email: ronit.das@s.amity.edu

Abstract. Accurate and fast identification of various food items can be very useful, in terms of preventing harm caused by allergies and other problems. In this work, an attempt is made to classify food items from real-world images with great accuracy and in real time. The proposed model is designed in a dual-phase classification method incorporating unsupervised clustering followed by a classification model to improve the initial prediction results. The dataset, selected primarily for its various food classes, consists of real-world images of food, captured outside controlled conditions. During the first stage, clustering is used to group class predictions into clusters. In the next stage, the model for the cluster with the highest prediction value is loaded to make the final prediction. Each cluster model is trained on a small subset of classes, reducing time and cost thereby improving performance. The accuracy metrics of the general model and some of the sub-models are compared to see if using smaller label subsets provides improved performance without a large increase in training time. Finally, for the generation of detailed information about food items and suggested recipes, an LLM will be integrated into the proposed model. Custom prompts will be used to generate contextually relevant data more effectively.

KEY WORDS: LLM, Clustering, Classification, Image Recognition, YOLO

1 INTRODUCTION

In today's world, people are increasingly aware of what they eat and its impact on their health. This awareness of health, well-being and taste has revolutionized the food industry. The variety of methods for cooking, preparing ingredients and serving food, along with various styles and achievements, has created enormous options for each meal. However, sometimes the uncertainty about what ingredients have been used in a food item has troubled people with some possible allergies. Multiple life-threatening cases have been reported around the world. In addition to this, food preferences also vary among meat eaters, vegetarians, and vegans. People traveling to foreign regions often find it difficult to get familiar with specific meals based on diet due to language barriers. Thus, a fast and reliable method to identify a food item from an image and obtain useful details like nutritional value, calorific value, possible ingredients used, and alternatives is the need of the time.

Traditional food classification models face challenges like high computational costs due to complex architectures and struggle with poor generalization in diverse environments due to variations in lighting, occlusion, and cultural differences in food presentation. The proposed model is designed on a two-stage classification problem that initially clusters the food images using the FAISS [9]

based K-means algorithm and then applies cluster-specific YOLO [10] submodels for refined classification that can help by providing an easy-to-use and reliable tool for food item identification and quick retrieval of the list of ingredients, nutritional value, and other useful details. The YOLO version used is YOLOv8n [11].

2 LITERATURE REVIEW

Existing research on food recognition looks at various approaches, including traditional machine learning and deep learning models, focusing on CNNs, transfer learning, and data augmentation. Most researchers discussed key challenges such as image variations due to lights and surroundings, reduction of noise in the model, and optimizing and making it more efficient and scalable food recognition systems. Some of the significant work is discussed in this paper [1-3]. McAllister, Patrick, et al. [4] explored ResNet-152 and GoogleNet models for food image classification where, the Deep features were extracted and used to train machine learning classifiers like ANN, SVM, Random Forest, and Naive Bayes. The model worked well for some of the datasets used by them, however, it faced challenges such as noise and randomness in the images available. Mezgec [5] introduced NutriNet, which was designed on a deep CNN framework for food and drink im-



Figure 1. Sample Images from the Dataset



Figure 2. Sample Images from the Dataset

age recognition on a 520-class dataset achieving a remarkable performance. The model was integrated into a mobile dietary assessment app for Parkinson’s patients, featuring online training for continuous model updates.

However, challenges like lower accuracy on real-world noisy images overfitting concerns, and difficulty in recognizing multiple food items in a single image remained a concern. Ka-reem et al. [6] proposed an intelligent system to recognize food from images and retrieve recipes using a customized MResNet-50 model, Word2Vec and Transformers, along with an ontology-based approach. The model was found to be quite a reliable tool for the food industry.

3 DATASETS

This paper uses the Food101 [7] dataset for training the base model for classification. It consists of 101,000 images divided equally into 101 food types. The images were taken in real-world conditions and were not cleaned to reduce noise to better represent the type of input images for which the model is expected to be used. All the images in the dataset have a size of 512 x 512 pixels.

75,750 images (750 randomly selected images per class) were used to train the model and 2,250 images were used to test the model’s performance. This distribution shows that the dataset is balanced for all class labels, so bias in the model embeddings from a class imbalance can be avoided. Some of the images in the dataset show dishes, cups, and other utensils, like Figure 1, which might introduce unintended correlations, potentially affecting model performance. However, most of the images focus on food items, as can be seen in Figure 2.

It might be possible that the model associates items that are not food with food item labels, so it is important to ensure that as much as possible of the image is only of the food item.

4 METHODOLOGY

The proposed Deep Learning-Based Hybrid Framework for Food Classification and Recipe Generation involves six key steps as discussed.

4.1 Data Preprocessing

Prior to implementation of the proposed methodology described in this paper, the data must first be processed to extract the necessary components to be used and transformed into the appropriate set of data expected to be used as input to the model.

- The Food101 dataset [7] is loaded for training. It is a dataset manually compiled using food images from real world conditions instead of controlled conditions.
- The images are resized to 224*224 pixels. They are originally stored in the form (Height * Width * Channels) and after the transformation is applied it becomes (C * H * W). Since the images are coloured, their pixel values are going to represent their RGB values. The transformation will also scale the tensor values between 0 and 1.
- After transformation, the values are normalized. In terms of images, this refers to subtracting the mean pixel values of the dataset from all the pixel values of each image and dividing the result by the dataset’s standard deviation. Normalization is useful because it ensures that the dataset has a consistent range and distribution, which helps models reach convergence faster.
- Divide the dataset into training and testing sets.
- After splitting, separate the features from labels into different data objects.

The main idea of the model is to use clustering methods from the FAISS[9] library to group the item into classes of similar items and then use cluster-specific models to predict the item from those classes. This two-stage method will help increase the accuracy by reducing the features each sub-model needs to learn, and thus improve inference time overall. To cluster the items, the K-means

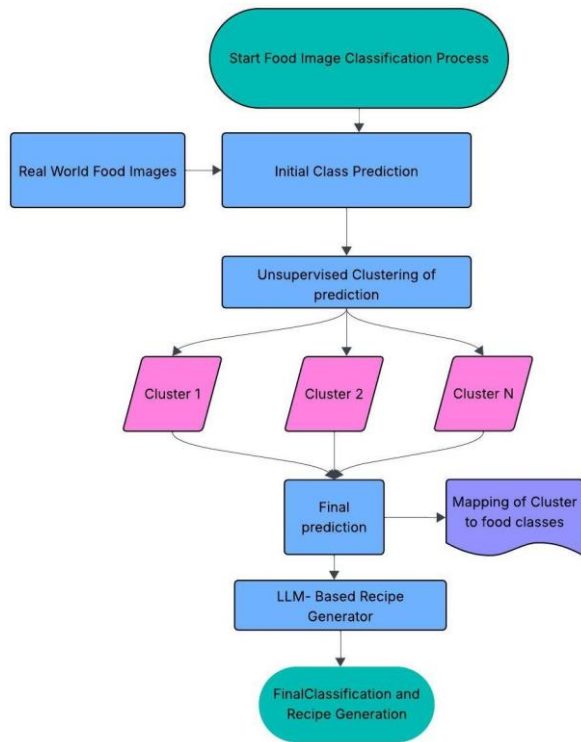


Figure 3. Clustering labels into classes

algorithm available in the library will be used. For classification, the Ultralytics YOLOv8n [11] model is used as the base model with a custom classifier head on top of it.

4.2 Clustering Using FAISS

This section describes the custom classifier head used for the classification of food items for the model. This should replace the final layers of the selected base model. There are two tasks in this section, first to group images by label into different clusters and then generate cluster-specific models for each one for more specific classification. For efficient search and selection of one of several sub-models, it is necessary to obtain a heuristic value to predict which cluster the image likely belongs to and select the sub-model accordingly. To obtain this heuristic value, the first step should be to form clusters of similar labels from the set of all unique labels in the dataset. Figure 1 shows a high-level overview of the process of forming the clusters and saving them to a dictionary for future use of food classification and LLM-based recipe generation.

- The first step is to initialize the clustering object from the FAISS library[9]. This library contains methods which are optimized for efficient similarity searching and clustering of dense vectors and are thus useful for large datasets like the one used in this paper.
- The K-means clustering algorithm can be initialized with the number of clusters set to 10 clusters as a starting point. This is only a first-guess value, and based on the results and further analysis, it can be adjusted. One common method is the Elbow method.

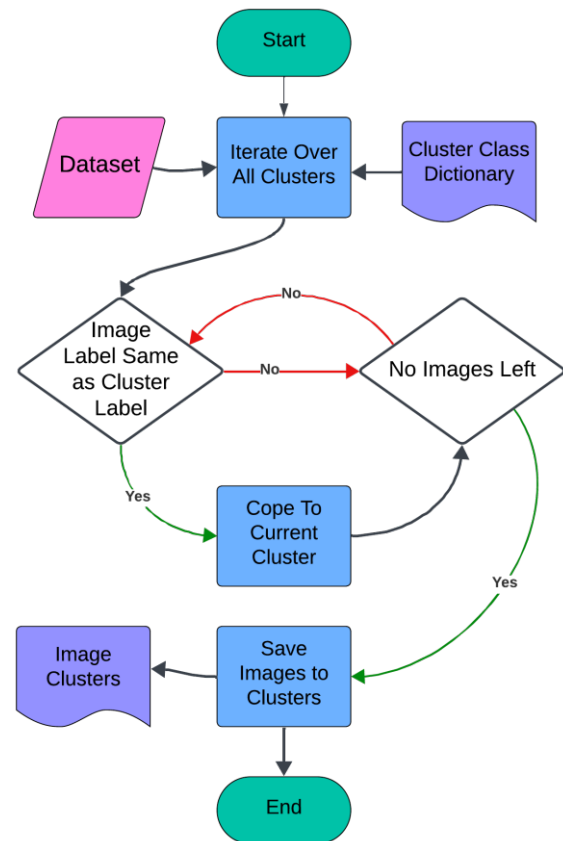


Figure 4. Separation of Images into Clusters by labels

- Train the cluster object on the label array, get the cluster labels, and map the clusters' labels to actual item labels. This ensures the naming is correct. After that, start assigning labels to clusters, based on the highest frequency cluster for that label.
- Save clusters and associated labels to a file or dictionary. This dictionary will store clusters and all assigned classes for each cluster. It can be loaded when assigning all the images to their correct clusters later on.

4.3 Assigning Images to Clusters

After the clusters have been created and labels assigned to each one, there is still the need to assign image data to all the clusters for model training. Figure 2 shows the process of assigning images to their correct clusters for later use in training the submodels for each cluster.

- Start the process by loading the dataset and the cluster class dictionary. The dictionary stores all clusters and the labels assigned to each cluster. Each entry in the image dataset contains the image itself and its associated label. The image's label will be used to match the image to its appropriate cluster.
- Iterate over all clusters in the dictionary to assign images to the clusters. This loop will run as long as the current cluster is not the last in the dictionary.

- The next step is a comparison of the image label with the list of labels in the cluster. This is the important portion of the process. Denoting the image label as I_l , the current cluster as J and the set of labels in the current cluster as the set J_{labels} , check if I_l is found in J_{labels} , then add I_l to the image set or dictionary for J . If image label is not found in cluster's label list, move to next iteration I_l+1 and return to the previous step.
- Once the image label comparison is done, check if I_l for all I in the dataset have been compared to J_{labels} . If not, move to the next image iteration. If it was the last image, end the current Cluster iteration and move on to the next one.
- After all images have been assigned to the correct clusters, copy all images listed in a cluster to its directory. The saved images will be used to train the sub-models for each cluster.

4.4 Model Training

First, a general model will be trained on all the classes in the dataset. The sub-models will be trained on all the images within a specific cluster. These are all specialized models that are only good at classifying a small subset of images. To validate that all images have been assigned, the images in each cluster will be counted and summed up. After verification is complete, the process to train the sub-models starts. The overview of the process is shown in Figure 3. To train the sub-models, the process that is used is as follows.

- The system starts the training process by iterating over all the image clusters saved earlier one by one. The total volume of images in each cluster is a small fraction of the total images in the dataset.
- The system generates a new model instance each time a new cluster is picked. It is not the general model trained earlier, but a fresh instance of the selected base model.
- The system fits the model instance to the images saved in the current cluster. This is the main point of using clustering earlier to group the labels.
- After the model instance is finished training, save its weights separately. Next, the system repeats the above steps for all clusters and trains one instance of the selected base model for each cluster, saving the best weights separately each time.

4.5 Model Prediction

The model is meant to predict labels using a two-stage process to classify food items. It will use a combination of a general YOLO[10] model trained on all labels and YOLO submodels fine-tuned for more specialised prediction. Figure 4 shows the methodology workflow for predicting the class of an input image using the groups that have been created beforehand during the preparatory steps of the two-stage process.

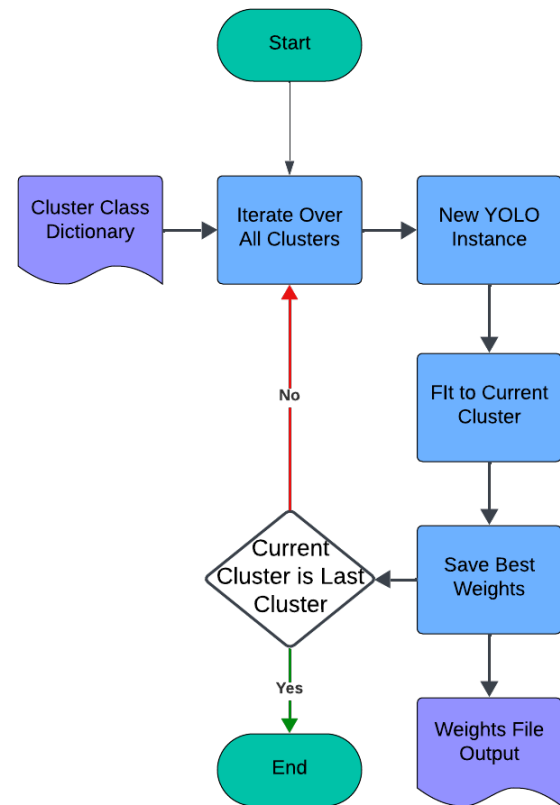


Figure 5. Train Sub Models on images in each Cluster

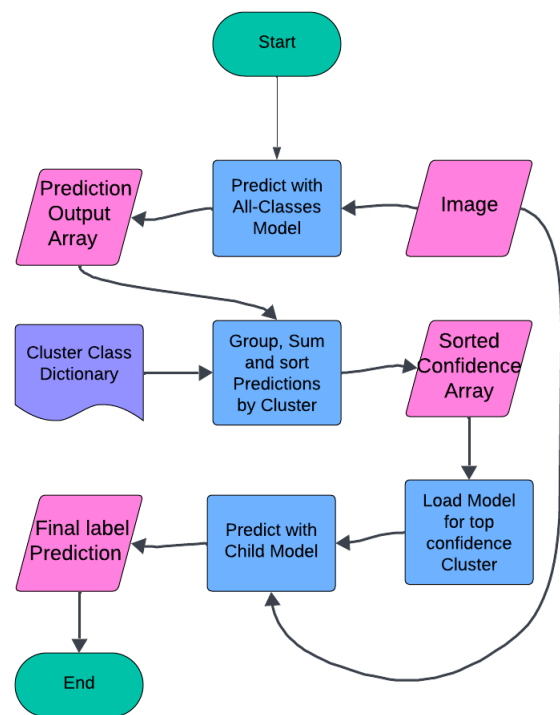


Figure 6. Two-stage Image Prediction

- The first step in this process starts when the image is given to the system. The general model performs an initial classification, outputting an array of predictions for the image. The general model will have lower accuracy compared to the sub-models since it is trained to recognize features of a much larger set of items.
- The general model will generate an array of predictions for the image and the system will organize them based on predefined label clusters using the previously saved Cluster Class Dictionary.
- Once the initial predictions are grouped into clusters, the system will calculate the confidence value for each cluster. This sum is a value that denotes how confident the model is that the image it received belongs to a particular cluster.
- After the system has obtained cluster confidence values, they are sorted in descending order. The top-ranked cluster will be the one with the highest confidence value so that one will be selected as the most probable cluster to which the input image belongs.
- Once the system has selected the most likely cluster for the image's class, the next step is to improve the classification further. Each cluster has a specific sub-model trained specifically on images in that cluster. The system loads the sub-model weights for the selected cluster.
- With the sub-model for the cluster loaded, the final label prediction is made. Therefore, the final prediction result has much better accuracy as compared to using just a general model.

4.6 Recipe Generation Using LLM

For generating recipe details and preparation instructions, the model will use an LLM. The language model used for it will be the Gemma 2B[8] model developed by Google. After the system obtains the final prediction, the user can request some details of the item, which will prompt the RAG model to generate some facts about the food item, along with nutritional data and calorific value. To achieve this, a few functions are defined to load the LLM and tokenizer that prepares the string inputs for the LLM model. Then, prompt insertion is used to pass a few queries to the function, where an encoder converts the prompt into embeddings. The LLM model then generates output based on the prompts and stores them in a variable named dish history. After that, the index values obtained from a FAISS[9] method are used to perform a nearest-neighbour search on the history, and the closest or most similar results are retrieved by the function. Further functions are also used to generate and store data about the recipe details and a summary of nutritional values.

5 RESULTS AND DISCUSSION

To evaluate the proposed two-stage method, the performance metrics of the general model are compared to those of the sub-models for the same label. The metrics are displayed in Figure 7. The percentage accuracy of the general

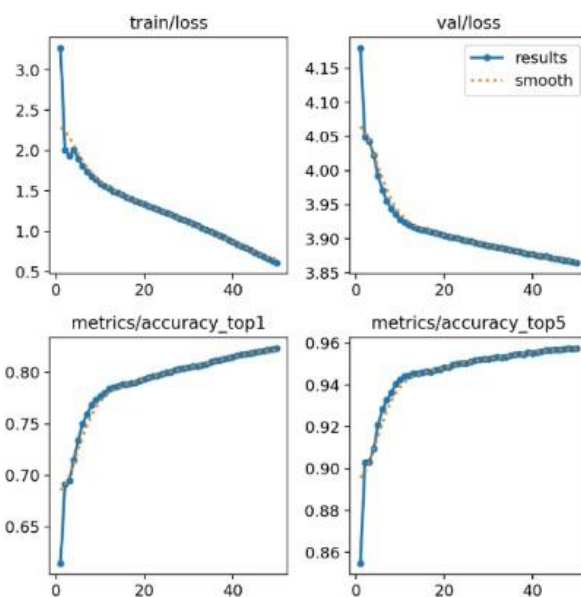


Figure 7. Accuracy and Loss Metrics for General YOLO Model

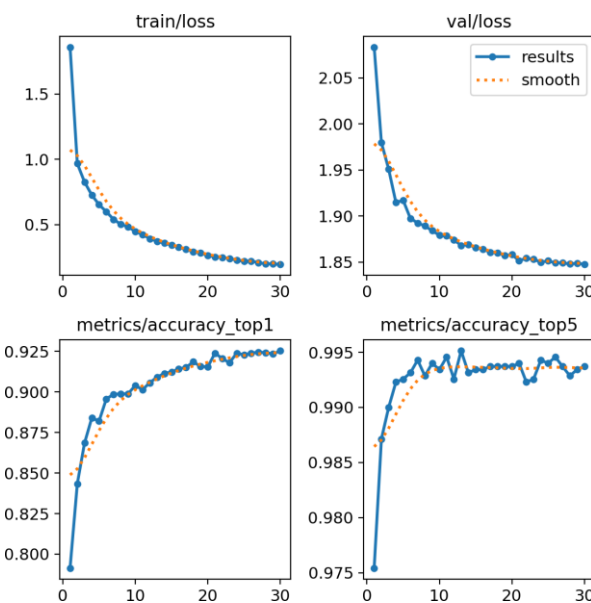


Figure 8. Performance Metrics for cluster 1

YOLO model is 82. Comparatively, the overall accuracy of the proposed methodology is 89 percent. The accuracy of the sub-models is consistently over 95 percent after training over 30 epochs. Figures 8 and 9 show the performance metrics for clusters 1 and 2 respectively. This comparison explains the benefit of using multiple sub-models alongside the general model to improve the overall performance of the proposed methodology.

6 CONCLUSION

In this paper, the previously used methods for the classification of food items have been studied and it has been

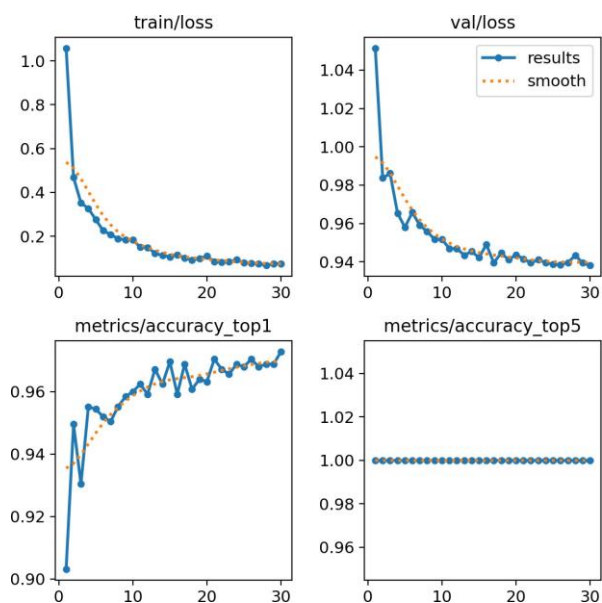


Figure 9. Performance Metrics for cluster 2

identified that the major problem with obtaining high accuracy and reliably fast models is the lack of datasets with variation. To address the issue of inference time, the proposed methodology uses the object detection algorithm YOLO[10], which is optimized for deployment on edge devices and fast inference because it is a single-shot computer vision algorithm. To reduce computational requirements, it has been proposed to apply clustering on the data using the K-means algorithm from the FAISS[9] library, meant for efficient clustering of dense vectors, hence it is useful for large datasets. After clustering, the methodology proposes the use of multiple specialized sub-models trained on small subsets of the entire corpus of data and the use of a generalized initial model to find the specific cluster sub-model to use, thus saving computation cost, simply the model for the cluster that the input image most likely belongs to. This two-stage methodology can be scaled for even larger and more complex datasets.

A limitation of the method in this paper is the inability to incorporate textual and other ancillary data in the prediction system itself because of the use of the YOLO model for classification, which cannot make use of data other than images, leading to the reliance on the Gemma[8] model for the generation of textual information about the food item.

7 FUTURE WORK

For future research, hybrid models or enhancements to the YOLO model used in this paper could incorporate additional data for more accuracy and relevance. Another method to improve the accuracy of generated outputs would be to add a RAG pipeline to the LLM to add a contextual and factual information database it can use to help with text generation. For this step, information about the food items will be taken, then converted to

vector embeddings and stored in a vector database. The LLM will be able to use similarity search on this database to augment its generated text outputs, thereby increasing the likelihood of generation factual information and reducing the risk of hallucination.

REFERENCE

1. Zhou, Lei, et al. "Application of deep learning in food: a review." *Comprehensive reviews in food science and food safety* **18.6** 1793-1811 (2019).
2. Pan, Siyuan, et al. ChefGAN: Food image generation from recipes. In *Proceedings of the 28th ACM International Conference on Multimedia* (2020).
3. Ma, Peihua, et al. Deep learning accurately predicts food categories and nutrients based on ingredient statements. *Food Chemistry* **391**, 133243 (2022).
4. McAllister, Patrick, et al. Combining deep residual neural network features with supervised machine learning algorithms to classify diverse food image datasets. *Computers in biology and medicine*, **95**, 217-233 (2018).
5. Mezgec, Simon, and Barbara Koroušić Seljak. NutriNet: a deep learning food and drink image recognition system for dietary assessment. *Nutrients* **9.7**, 657 (2017).
6. Kareem, Razia Sulthana Abdul, Timothy Tilford, and Stoyan Stoyanov. Fine-grained food image classification and recipe extraction using a customized deep neural network and NLP. *Computers in Biology and Medicine* **175**, 108528 (2024).
7. Luka Bossard, Matthieu Guillaumin, Luc Van Gool. Food-101 – Mining discriminative components with random forests. In *Computer Vision–ECCV 2014, Part VI*. Springer International Publishing, 446–461 (2014).
8. Gemma 2B model. <https://huggingface.co/google/gemma-2b>
9. FAISS Library <https://faiss.ai/>
10. Redmon, J. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. (2016).
11. Ultralytics YOLOv8. <https://docs.ultralytics.com>