CSCI 1430 Final Project Report: Your Food Friends

Team name: Winston Li, John Ryan Byers, Nathan DePiero.
TA name: Anh Duong. Brown University

Abstract

Millions around the world are visually impaired and have trouble identifying food placed in front of them. Tourists in new countries might not know a dish and are too embarrassed to ask what it is.

With a simple picture, YourFoodFriends offers an aid to accurately identify the foods on a plate or tray. Trained on 101 foods ranging from all cultures and cuisines, our segmentation-classification ensemble seeks to help the visually impaired and tourists know what they are eating.

1. Introduction

We seek to offer a solution for identifying multiple foods on a plate or tray. Our ensemble workflow is especially useful for people that are blind and can not identify what they are eating without assistance. In addition, our model can be used for tourists trying to identify a new dish.

Major difficulties of this project involve accurately segmenting plates of food. Food is inherently complicated, and it is difficult to generalize individual components such as lettuce leaves from a salad due to diverse appearances. Another difficulty is gathering enough food categories to train on due to the wide range of foods across different cultures around the world.

We present a two-stage approach by first segmenting an input image into different food candidate bounding boxes and then returning the food label of each of those boxes through classification. As will be discussed, we used MetaAI's Segment Anything-Model (SAM) model [8] for segmentation. Our classifier was trained on data from the Food 101 Dataset [2] with Inception Resnet V2 [17] serving as our base architecture.

Our full workflow was tested on images we gathered from The Sharpe Refectory (Ratty), which is a Brown University Dining Hall.

2. Related Work

Our ensemble uses SAM as our segmenter to generate food candidates, which are classified using a model based upon the Inception ResNet V2 model.

We chose to work with a segmentation model as a backbone as opposed to an object detection alternative due to the former's greater versatility and depth. While object detection models like YOLO [11] have already been combined with prompt-based selectors [12] in food detection, such models generally output only bounding-boxes. With segmentation models, entire masks are generated instead. These provide further fine-grained detail such as object size, from which more quantiative features can be modeled. Especially with regards to food, relative mask size can be used as a predictor for nutrient and caloric information.

Lastly, the expansiveness of SAM means masks can still be generated for food items not currently included in our classifier's possible labels list. This means new food items only require the classifier to be updated, which can be done via transfer-learning of the top layers.

As for our classifier, we selected the Inception ResNet V2 model primarily due to its accuracy. Refer to section 3.2.1 for further details.

3. Method

3.1. Segmentation

Our ensemble workflow begins with segmenting individual food candidates on a food tray. SAM was employed for this task. First, all possible masks were generated using the predefined *SamAutomaticMaskGenerator()* method. However, as SAM generates *any* mask, beyond possible food candidates, these generated masks then had to be filtered. From these filtered masks, crops of the original images defined by the masks were extracted and inputted into the classifier.

3.1.1 Mask Generation

SAM's built-in automatic mask generator produces *all possible* masks for an image, as it is not specialized for just

food segmentation. This produced inputs inappropriate for our classifier, and so they had to be filtered. Naive heuristics for filtering based on mask areas, stability scores, and predicted-IOUs were all considered.

First, mask areas were chosen on the reasoning that individual food items on a food tray cannot be too big (e.g. the overall food-tray should be the largest item in a picture) nor too small (e.g. the classifier's possible label outputs do not include small food components like the lettuce leaves of a salad). Although our implementations feature a hard-coded max mask area threshold, this can be parameterized with respect to the image for more fine-tuned filtering.

Next, stability scores represented the measure of a mask's quality produced by SAM. We noticed that for extremely small items such as relatively amorphous lettuce pieces in a salad, stability scores tended to be low. Thus, it was thought that stability scores can serve as a proxy metric for mask largeness, and that we can filter out too-small items for our classifier.

Lastly, predicted intersection-over-union (iou) represented the model's own prediction of a mask's quality. When viewing all possible masks generated, it was apparent that masks (outputted in a list by SAM) were ordered in descending predicted-iou's. Larger masks such as the entire picture, large food items, and other big non-food masks appeared first, while finer-grain masks such as food components appeared later. Thus, we used predicted-ious as another analogue of a area-based filter heuristic.

3.1.2 Candidate Extraction

Most importantly, masks produced by SAM contained a COCO-RLE [9] representation and bounding-box information. From the bounding box coordinates, we were able to extract specific cutouts from the original image pertaining to the mask, which theoretically represented a valid food candidate.

However, simply extracting an entire bounding-box introduces unwanted noise as the actual mask does not necessarily take up the entire bounding-box. For example, a diagonal mask has a rectangular bounding-box that may include other items off the diagonal. In this vein, we used the actual COCO-RLE-encoded mask representation to selectively crop the image to just have the mask and zeroed-out pixels elsewhere within the bounding box. These extracts were then inputted into our classifier for labeling.

3.2. Classification

After receiving potential food candidates from the segmenter, our classifier assigned labels to them. All classification models were trained on the Food-101 dataset, which included 101 different food-labels with 1000 images each. Three-fourth of images were reserved as training-images and contained noise (including deliberate mis-labelings), with the remainder being for validation use. The classifier models were trained using the TensorFlow framework via Keras [1] [3].

3.2.1 Model Selection

Several classification base models were considered on the ground of their accuracy, speed, and overall size. In general, all models were a product of first transfer-learning where we attached our own head layers to a model pre-trained on the ImageNet dataset [5]. These were trained for a few epochs, and the best models were then chosen for further fine-tuning. Results of the models can be found in Table 2.

Model	Size (MB)	Parameters	TPIS[a] (ms)
VGG16	528	138.4M	4.2
IRNv2	215	55.9M	10.0
Xception	88	22.9M	8.2
ENB3	48	12.3M	8.8
MNv2	14	3.5M	3.9

Table 1. Model Comparisons (from Keras Applications). **a.** Time per inference step on GPU on a Tesla A100.

Out of familiarity, we began with the VGG16 [15] model. This model takes advantage of several layers small convolutional filters (3×3) in series. We removed its top layers and replaced it with our own (Flatten $\rightarrow$ Dense 4096 $\rightarrow$ Dropout 0.5 $\rightarrow$ Dense 4096 $\rightarrow$ Dropout 0.5 $\rightarrow$ Output). Ultimately, this model proved too inaccurate, too slow, and too big for our uses.

Next, we moved to an Inception ResNet v2 (IRNv2) model as our base. This model features several "grid-reduction" modules combined with residual connections, a type of skip-connection. To this model we attached our head layers (Global Average Pooling $\rightarrow$ Dense 2048 $\rightarrow$ Dropout 0.5 $\rightarrow$ Output). While comparatively large and slow to train, just a few epochs of training only the head led to great results, and so it was chosen for further fine-tuning.

We then worked with $Xception$ [4] , which is similar to $Inception$ models though with a reversed pointwise/depthwise convolution order. This model was much smaller than $IRNv2$ with comparable accuracies and TPIS, thus motivating its selection. The best head layer we came up with was a simple Global Average Pooling (GAP) followed by output. Nevertheless, training just the head layer produced lack-luster results, so we did not continue to fine-tune this model.

Afterwards, we experimented extensively with the EfficientNet B3 (ENB3) [18] model due to its small size and high accuracy (comparable to IRNv2). Using a head containing a GAP layer followed by output, we obtained extremely

high accuracies upon training just this head for a few epochs. Thus, it was chosen to be fine-tuned; however, Keras had difficulties saving the model, so it was not ultimately used.

Lastly, out of interests in minimizing model size and TPIS, we considered MobileNet V2 (MNv2) [14], which features an inverted residual structure and lightweight depthwise convolution filters. To the head we attached a GAP layer prior to the output. Although it was extremely quick to train, its performance was too low for further development.

3.2.2 Model Training

Models were trained using high-performance computing clusters available to Brown members. The majority of models were trained on Oscar CCV, though a few were also trained on Brown CS Dept.'s GridEngine. These resources featured powerful GPUs and reliable service over personal computers and Google Colab.

Additionally, we employed mixed-precision training to take advantage of Nvidia GPU architecture. This was combined with batching to help accelerate model development.

Promising models were further fine-tuned by unfreezing *all* layers, including the base-model prior to fitting. To prevent futile training and over-fitting, we employed learning rate schedulers activated upon plateaus and early stoppers. These were dictated by validation accuracies. Generally, fine-tuning stopped prior to 20 epochs and featured 10-15% greater validation accuracy compared to just training the head layers.

3.3. Labeling

Once the classifier returns a prediction for each food candidate, this output can be further processed into a food label to be displayed back on the original image.

3.3.1 Extracting Labels

Initially, the softmax output layer of the classifier produced a probability distribution over the 101 possible labels. However, we recognized that SAM's input candidates to the classifier were not necessarily food. We reasoned that if a valid food candidate was passed into the classifier, and that the true label of the candidate was part of classifier's trained-upon 101 classes, then output distribution will have a high probability at just one point and low values elsewhere. Conversely, if a invalid candidate (either non-food or not in our 101 labels) was inputted into the classifier, then the classifier would produce a distribution without any comparatively high probabilities.

Consequently, candidates with "confident" predictions (i.e. the most-likely probability is very high) were given its corresponding label, while "diffident" predictions (i.e. the most-likely probability was not high compared to its peers) were given a reserved marker as a label.

Figure 1. Masks produced by SAM

To note, we experimented with different ways in determining confidence in predictions. Naive thresholding at a constant value tended to produce "confidently incorrect" predictions. Furthermore, multi-modal distributions where labels are split between very similar labels likewise split max probabilities, leading to valid candidates falling under the threshold.

3.3.2 Label Reporting

After all labels are extracted, they are then combined with the original masks to generate labeled bounding boxes on the original image. This was done iteratively. For each label in the extracted label list, those with actual labels were reported. However, those given the reserved marker were not reported (i.e. no bounding box drawn). This served as a secondary filtering method in addition to the initial SAM mask-quality-based filtering methods to reduce non-food reporting.

4. Results

4.1. Segmentation Mask Filtering

The SAM model we used generated masks, but many of the identified bounding boxes were not nearly as accurate as we had hoped. An example SAM output can be seen in Figure 1. Many of these masks were unusable such as the cut up versions of the hotdog bun and the vertical slices of the fries. We were not able to determine a specific accuracy rate of SAM as it is not published online, and there are so many arbitrary segments. We attempted to filter through the returned masks to find the most precise ones, however, this was challenging as the number of incorrect masks was high.

4.2. Model Comparisons

The fine-tuned Inception ResNet V2 model had the best test accuracy. However, all fine-tuned models had extremely

Base	Head	Train Acc	Test Acc
VGG16	Custom1[a]	0.4645	0.5023
Xception	GAP2D[b]	0.7007	0.6727
FT MNv2	GAP2D	0.9999	0.7895
FT ENB3	GAP2D	0.9998	0.8265
FT IRNv2	Custom2[c]	0.9997	0.8713

Table 2. Accuracies of various models.
a. Flatten, Dense(4096), Dropout(0.5), Dense(4096), Dropout(0.5).
b. Global Average Pooling 2D
c. GAP2D, Dense(2048), Dropout(0.5)

high training accuracies. This is likely due to the quality of the Food101 dataset that the models were trained upon: deliberate mis-labelings were introduced into the training set to represent noise; due to the strong resolving power of these models, they quickly "latched on" to these mistakes, which translated poorly during validation. Nevertheless, an accuracy of 87% lands FT IRNv2 among the top 10 models ordered by Top 1% accuracy that are on PapersWithCode [10].

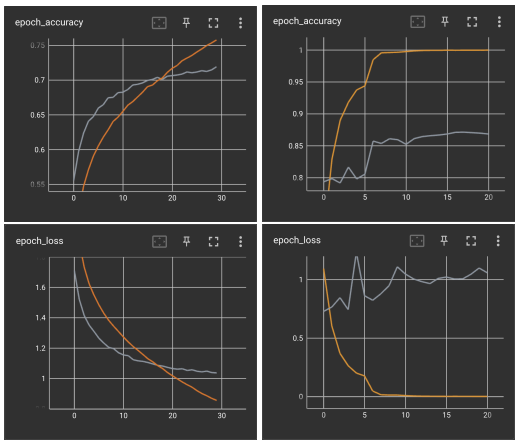

Figure 2. Accuracy (top) and Loss (bottom) over time. Training in orange, validation in gray. *Left:* Inception ResNet V2. *Right:* Fine-Tuned Inception ResNet V2.

4.3. Label Reporting

Our workflow was impressively able to identify many foods in the test data we passed to it. As seen in Figure 3, our workflow correctly identifies, pizza, Caesar salad, soup (lobster bisque), and frozen yogurt. The model struggles with hummus which it misclassifies as an omelette. We suspect that this is due to the hummus' particularly semi-oval shape.

As our classifier accuracy improved, so did the labelling accuracy of our model. However, we were limited by the number of diferent foods in our training data. Carrots for example were not a class in Food101 and therefore it is not identified at all in our model as seen in Figure 3.

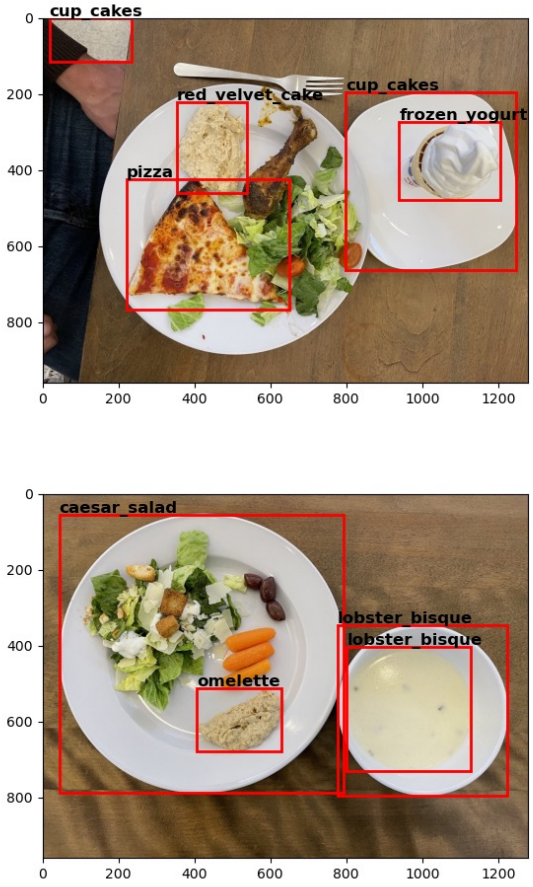

Figure 3. *Top:* Our classifier correctly identifies pizza and frozen yogurt. *Bottom:* lol

4.4. Technical Discussion

The greatest difficulty in this project was incorporating SAM into a food-focused workflow. Since SAM is an ultra-versatile segmenting model, it produces masks beyond just those containing foods; as such, we had to filter out non-food masks.

Purely mask-based filter heuristics were insufficient since we cannot specify food-defining metrics; rather, such heuristics are fundamentally arbitrary. Thus as a supplement, we introduced classification-based filtering as discussed in section 3.3.1. This tactic relies on Softmax probability distribution outputs to be valid representations of both confidence and more importantly, uncertainty. The former requires additional model calibration to be valid [6]. The latter however, appears to be more complicated with the literature split on whether Softmax outputs can be conflated (at least as an analogue) to uncertainty [7].

Currently, we are developing a new filtering strategy that will employ a binary food-not food classifier to filter out-

puts from SAM going into the IRNv2 label classifier. This binary classifier was trained on the Food 5K dataset [16] and employs a simple MobileNet v2 architecture for quick training and inference. After 5 epochs of training, the model had a training accuracy of 0.9043 and a validation accuracy of 0.914, which we found sufficient for our uses.

4.5. Socially-responsible Computing Discussion via Proposal Swap

The first critique from our proposal swap indicated concern about how to deal with foods that are less common. It identified a problem where our model and dataset can not identify food for diverse cultural backgrounds. We accept this criticism and acknowledge this problem. We selected our dataset for its variety in 101 different foods, but understand that this is not enough to cover food from all cultures. To mitigate potential problems we could append our dataset with other types of food and their associated images. We would then retrain our model, and then be able to classify these new foods.

Another critique that was brought to our attention was how to ensure the credibility of our data. The other team was concerned that misclassiying food could impact people with food allergies or dietary restrictions. Ensuring the individual accuracy of all 101,000 images in our dataset would be unrealistic, but we can assign some confidence to the data because it is published as part of a research paper. As regards to the impact of people with food allergies or dietary restrictions our workflow should be used with caution. We have achieved an 87 percent accuracy on our classifier, but this is no where near high enough for someone with a deathly food allergy to trust. Our workflow should be used to get the general concept of foods on a plate, but not as a check to prevent allergies. If the user has an allergy or are not supposed to eat a food for a particular reason another source should be consulted.

The third critique identified by the group was concern over how we would decompose ingredients from food categories. They explained that the same foods can have varying ingredients and it would be extremely hard to identify these components from an image. We recognized this concern ourselves in one of our early developmental discussions and agreed to remove this component. We agree that it is nearly impossible to determine the ingredients of a food just from a picture because foods have such a variety of ingredients. Therefore, in our project we only produced images with our predicted labels of the food.

5. Conclusion

Ultimately, our workflow demonstrates a way for food to be segmented and classified on a plate. While it does not contain the ability to identify any dish from any culture, the project acts as a pathfinder for future teams looking to improve food recognition in complex images. Our workflow could potentially be used to assist the visually impaired or tourists seeking to understand what they are eating. Moving forward we recommend future teams experiment with segmentation models other than SAM. While SAM could provide segmentation without training, many of the bounding boxes generated were incorrect.

References

[1] Martín Abadi Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 2

[2] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *European Conference on Computer Vision*, 2014. 1

[3] François Chollet et al. Keras. `https://keras.io`, 2015. 2

[4] François Chollet. Xception: Deep learning with depthwise separable convolutions. *CoRR*, abs/1610.02357, 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. 2

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017. 4

[7] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. *CoRR*, abs/1706.04599, 2017. 4

[8] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023. 1

[9] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 2

[10] MetaAI. Image classification on food-101. https://paperswithcode.com/sota/image-classification-on-food-101-1, 2018. 4

[11] Eslam Mohamed, Abdelrahman Shaker, Hazem Rashed, Ahmad El Sallab, and Mayada Hadhoud. INSTA-YOLO: real-time instance segmentation. *CoRR*, abs/2102.06777, 2021. 1

[12] Hoang-Lan Nguyen. Food-recognition. https://github.com/lannguyen0910/food-recognition, 2021. 1

[13] Tim Pearce, Alexandra Brintrup, and Jun Zhu. Understanding softmax confidence and uncertainty, 2021.

[14] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks, 2019. 3

[15] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 2

[16] Ashutosh Singla, Lin Yuan, and Touradj Ebrahimi. Food/non-food image classification and food categorization using pre-trained googlenet model. In *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*, MADiMa '16, New York, NY, USA, 2016. Association for Computing Machinery. 5

[17] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016. 1

[18] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*. PMLR, 09–15 Jun 2019. 2

Appendix

Team contributions

W. Li Assisted in dataset selection. Assisted in incorporation of Segment-Anything-Model. Oversaw training, fine-tuning, and incorporation of classifier models. Managed team's high-performance computing requirements and debugging. Assisted in paper composition.

J.R. Byers Worked on finding the dataset and compiling it into usable format. Developed SAM implementation and filtration techniques to get appropriate masks and bounding boxes. Ran training on GridEngine and optimized classifier models. Managed team branches on Github.

N. DePiero Assisted in dataset collection/loading. Created initial preprocess and training loop. Led integration of SAM with classifier to output final image. Experimented with SAM filtration techniques to get masks/bounding boxes for just the food items.