Food Analyzer: DCNN Transfer Learning for Food Recognition and Ripeness Prediction

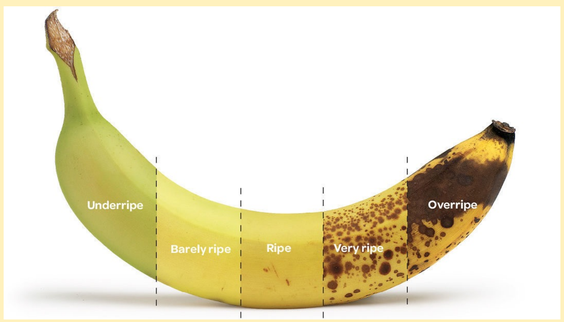
Henry Bobeck, Sam Stazinski, Zane Snider, Devarshi Bhadouria

Abstract

From a user-side process, the Food Analyzer first takes in a provided input image of a piece of food. It then interprets the image and provides back information of 1) the name of the food, 2) its current ripeness/predicted expiration, and 3) basic and nutritional info about the food. In order to achieve this final goal, we provide solutions for the following subproblems: first, to design and train a neural network to correctly identify food in an image. Second, to design and train individual neural networks to correctly predict food ripeness for a specific food. And third, to write a script to retrieve basic & nutritional information for a given food name. In this way, we approach the design structure of the project using a cascading style, in which the results for subproblems (2) and (3) above rely on the result from subproblem (1), the food recognition model. Through the efficacy of our resulting final product, we show that this method of combining cascading deep convolutional models and scripts into one fluid, intuitive user-side process is an effective approach for AI-based application development.

Introduction

We expand on existing food classification infrastructure by associating classifications with expected expiration, ripeness level, and nutritional information.



VS.

Because of the unique nature of aging, food ripeness classification is still performed manually in most real-world applications.

Methodology

Transfer Learning. by replacing the last few layers of a proven, pre-trained DCNN model, freezing the deep layers, and then training on our specific data, we make use of the powerful base model while also directing the predictions to our custom set of classes. This approach is called transfer learning, and its benefits include much less required data, much shorter training time, and much better results compared to building from scratch.

Training Data. The majority of our training data comes from the openly-available dataset Food101. We supplement this data with 256 images of a new class 'banana', which we pulled from google image results. We also sort these banana images into three distinct ripeness classes based on the appearance of the bananas. This sorted banana data is then used as training data for the banana-specific ripeness detection model, which we use as a proof of concept for ripeness detection.

Image-Data Pipeline. We make use of keras' image preprocessing utilities to read the image data from the DataFrame into an ImageDataGenerator Dataset. As part of this process, all input images are sorted into batches of 32, converted to rgb, and resized to 224x224. The labels are one-hot encoded. Furthermore, in order to reduce potential for overfitting, we apply randomized transformations to the training images (not the testing images) regarding brightness, width, rotation, and flips. The images are then preprocessed to the desired input format of the base MobileNetV3 model, and a validation split of 0.15 is applied for the training images.

Food Information Scraper. Once we have a resulting prediction from an input user image, we utilize the Python library Scrapy to collect information displayed by Google after a Google search of "what is an \%FOODNAME\%" in the form of a string. The resulting text for a specific food is then displayed to the user when that food is detected in an image, as part of our final user-side interface.

Notable Obstacles & Struggles

Dataset Format Issues

ActiveLoop Hub was the host of the most promising dataset of food images we found to serve our model training. However, lack of documentation and features for this novel dataset format led to major roadblocks and time spent before we eventually turned to other sources for food recognition data.

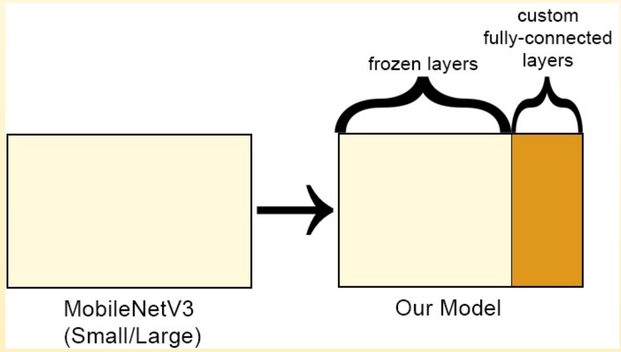
Lack of Tensorflow Experience

With a lack of previous experience using Tensorflow, considerable time was spent writing and rewriting a process for the image-data pipeline to properly handle incoming image data.

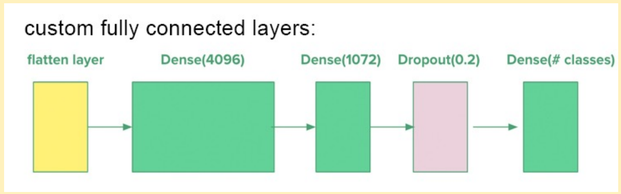
Data and Training Limitations

Excessive amounts of required training time is a common issue for training deep neural networks for image recognition. In our case, high training time and lack of data forced us to forego early intentions to test more models and configurations, as well as to collect enough data for more food classes and specificity in ripeness prediction.

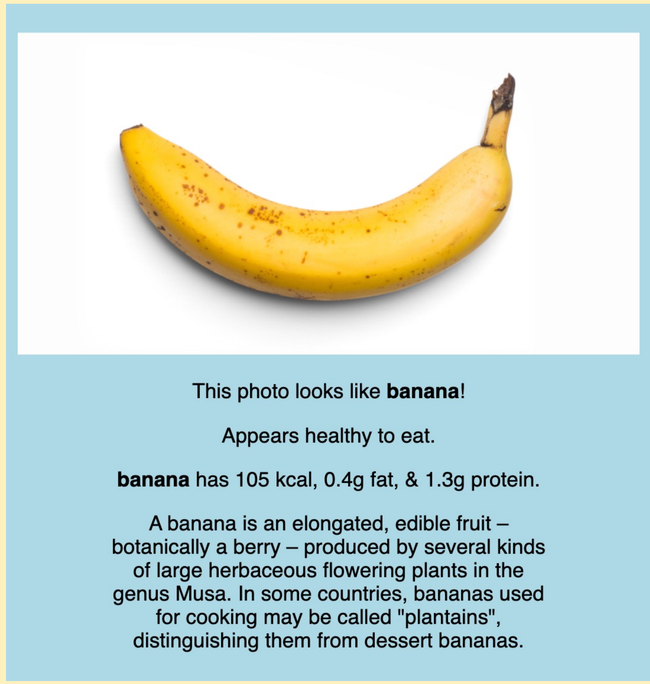
Flask User Interface. We use Flask to create a full-stack web application combining all elements of our project into one easy-to-use interface. Through creation of an appealing and intuitive UI, it is our intention to show that our method of combining cascading deep convolutional models and scripts into one fluid, intuitive user-side process is an effective approach for AI-based application development.



A graphic representing our transfer-learning approach with MobileNetV3.



A visual representation of our custom final layers in both models.



An example results page from our final web-based user interface.

Results

Our custom models achieved a top-1 accuracy of 82.07% for our general food recognition model with 11 classes and 96.15% for our banana ripeness model. Furthermore, the application user interface involves a simple and intuitive introduction page as well as a results page, wherein the required models are loaded lazily when needed. The user interface assumes no prior knowledge of computer science computer vision, and functions incredibly smoothly.

Example prediction results for our food recognition model.

Conclusion

Given the time constraints faced during our development process, we are highly satisfied with both the efficacy of the deep convolutional models constructed and the appeal and fluidity of our final user-side application. Considering our results above, we feel that our approach to the image-data pipeline and transfer learning has been a success, especially considering the limited data we had access to for training purposes. The final web application works incredibly smoothly, and usage is highly intuitive. While there are undoubtedly aspects to improve for future editions, such as the inclusion of far more training data, increased specificity in classes, and the inclusion of more specialized models for increased informational output, we believe our final product provides ample evidence for the efficacy and future potential of state-of-the-art image recognition techniques for use in consumer-facing apps.

Acknowledgements

Banana Ripeness Classification Based on Deep Learning using Convolutional Neural Network

Computer Vision and Image Analysis based Techniques for Automatic Characterization of Fruits – a Review

Object identification from few examples by improving the invariance of a Deep Convolutional Neural Network

[2103.16107] Large Scale Visual Food Recognition

[1409.1556] Very Deep Convolutional Networks for Large-Scale Image Recognition