CS230

Food Classification with CNNs

Katherine Gjertsen[*]
Department of Symbolic Systems
Stanford University
kgjert@stanford.edu

Abstract

Food remains at the core of our daily functioning. In addition to meeting one of our basic needs, food also operates as a mechanism for engagement with others. Some folks share pictures of food on social media to cultivate a stronger community, while others might take pictures of their food to keep track of their intake. Given the wide variety of food image applications, accurate food classification methods can have a high utility. In this paper, we experiment with various convolutional neural network models on small subsets of data.

1 Introduction

Food is not only integral to our biological functioning; it can also be an avenue for fostering community, learning, and creativity. This idea on display in the mass of food pictures that are consistently shared across social media platforms between friends and followers. Classifying and organizing the data can allow for a more streamlined and efficient sharing process. Beyond the social element of food, image classification can also be a powerful tool for standardizing self-reported dietary tracking apps. These relevant applications inspired the work that will be explored in this project: food image classification using Convolutional Neural Networks (CNNs). Our goal is to experiment with various aspects of the CNN architecture to observe the outcomes of the balance of the model's efficiency and overall computational cost.

2 Related work

In 2014, Bossard et al.[1] approached the problem of food classification by experimenting with a dataset that we will also be examining in this project, Food-101. The combination of the highly detailed dataset and the novelty of using Random Forests to min discriminative parts allowed Bossard et al. to yield an average accuracy of 50.76%.

In recent years, many developers have improved accuracy and efficiency through the application of refined architectures and algorithms. This increased performance can be seen through Liu et al.'s DeepFood [2], which was able to achieve a top-1 accuracy of 77.4% and a top-5 accuracy of 93.7% at its peak performance.

In 2019, Sahoo et al. [3] developed FoodAI based in Singapore that also acts as an API service. FoodAI consists of a highly accurate model trained on a corpus of 400,000 food images made up of 756 classes. The model was pre-trained on ImageNet and recent models such as ResNeXt and SENet were utilized to yield a top-1 accuracy of 80.86% and a top-5 accuracy of 95.61%. Using a comprehensive dataset, pre-training the model on ImageNet, and applying state of the art CNN architectures allowed the researchers to develop a highly accurate model.

[*]

Figure 1: Three example images from the Food-101 Dataset

3 Dataset and Features

In this project, we will be utilizing the Food-101 dataset originally used in Bossard et al.'s 2014 paper [1]. The dataset contains 101,000 images that are composed of 101 popular food classes. Each class contains 1000 images. The dataset was already split into training and testing sets with 75,750 training images and 25,250 testing images. The Food-101 dataset contains some mislabeled, noisy images that were purposefully not cleaned to account for potential mislabeling errors in the future. The maximum side length of any image is 512 pixels, however we experimented with normalizing the dimensions to smaller resolutions such as 150 x 150 and 256 x 256. The images were also augmented using methods such as flipping, rotating, and cropping in order to prevent the model from overfitting. Three sampled images from the dataset are shown in Figure 1. The dataset was originally used in Bossard et al.'s paper "Food-101 – Mining Discriminative Components with Random Forests" [1] and obtained from kaggle.com [4].

4 Methods

In this project, our model was run using Google Colab, a Google research product that is well suited to Machine Learning [5]. All our models are written in Tensorflow and a high-level neural network library, Keras [6]. We decided to use a randomly selected subset of classes within our Food-101 dataset for a faster paced iteration process. Although the small subset allows for a faster experimentation process, the smaller subset is at a greater risk for overfitting [6]. Overfitting occurs when the model only learns from a small amount of examples and therefore cannot generalize the predictions to new data [6].

One way to prevent overfitting is by focusing on the amount of information that the model stores, namely the model's entropic capacity [6]. We utilized a CNN with a limited number of layers to lower the entropic capacity so the model would only store the most relevant features. The model also uses dropout to limit the occurrence of random correlations. We utilized dropout and limited layers to reduce overfitting instead of data augmentation in order to speed up the experimentation process.

We randomly selected three classes and normalized their respective images to a size of 150 x 150 x 3. We trained the model using three convolutional layers and two fully connected layers [6][7]. The first four layers use a ReLU activation function and the final output layer uses a softmax activation function. We first experimented with a small number of epochs (10), a batch size of 16, and an RMSprop activation function. Although the model yielded a validation accuracy of 74.05%, there was extremely high variance [6][7].

We continued experimenting by tuning different hyperparameters. We increased the number of epochs to 20 and changed the optimization function from RMSProp to an Adam optimizer since it does well handling noisy data and often outperforms RMSProp. The model achieved an accuracy of 76.4% after training on 20 epochs. We hypothesized that the increase in accuracy was a result of the increase in epochs. However, despite the various methods used to reduce overfitting, the small model and dropout was not enough to prevent a high variance of the validation accuracy[6][7].

Although training with more epochs would most likely result in higher accuracy, it is difficult for a highly variant model to generalize predictions to new data. In order to continue training with the small dataset, we began to incorporate pre-trained architectures. After exploring a few different models, we decided to attempt transfer learning through the utilization of the InceptionV3 model [7][8]. The InceptionV3 architecture was first developed by Szegedy, et al. and is an efficient image recognition trained on the ImageNet dataset. We trained the subset of 150 x 150 dimensional images. The data was also pre-processed in order to reduce overfitting. We were hopeful that utilizing transfer learning would lead to an increase in validation accuracy.

5 Experiments/Results/Discussion

We utilized different models during our experimentation process. First, we started with the simple, standard CNN that was not pre-trained. The first model that utilized RMSProp and 10 epochs ran into the problem of extremely high variance.

Although we started with a small number of epochs in order to experiment more rapidly, we decided to increase epochs and tune various other features such as the optimizer and image size. The accuracy of the results only slightly increased and the problem of overfitting remained constant throughout various changes in features as seen in Figure 2:

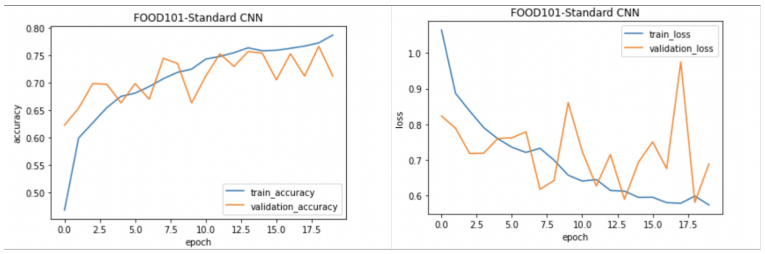

Figure 2: CNN with Adam optimizer and 20 epochs (still overfitting)

We recognize that these results are most likely due to our small dataset. We then transitioned to utilizing transfer learning through an InceptionV3 model[7][8]. Image pre-processing, dropout rate, and image size were all features involved in the experimentation process. By pre-training our three classes on the InceptionV3 model, we were able to obtain a validation accuracy of 89.67%[7]. Despite the high accuracy, the model was unfortunately now underfitting the data as displayed in Figure 3:

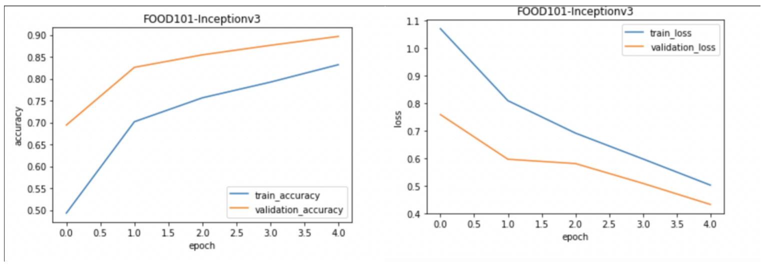

Figure 3: InceptionV3 architecture results (underfitting)

The model was likely due to the small number of epochs, noisy data and short training duration[6].

6 Conclusion/Future Work

We experimented with the Food-101 dataset to compare the results of various model architectures. The baseline model was able to increase accuracy through increasing training time, however the small sample size was subject to overfitting that wasn't mitigated by various augmentation and dropout methods. We then experimented with transfer learning by using the InceptionV3 model[8]. Although we were able to yield greater accuracy of about 89.67%, the potential noise and shortened runtime led to underfitting. Although transfer learning is beneficial for accuracy, it is important to train a larger set of data over many epochs in order to fully utilize the architecture's potential.

Future work would involve increasing the size of the data and the overall training time. We could also experiment with tuning different parameters. Further, after training a model that fits the data in a more effective way, it would be interesting to take a closer look at the images that are most frequently classified incorrectly. From there, we could create a subset with the more difficult images and engage in a more specialized experimentation process.

7 Contributions

This was an independent project.

References

[1] Bossard, L., Guillaumin, M., & Van Gool, L. (2014) Food-101–mining discriminative components with random forests Tesauro, D.S. Touretzky and T.K. Leen (eds.), *European conference on computer vision.* Springer, Cham.

[2] Liu, Chang, et al. (2016) Deepfood: Deep learning-based food image recognition for computer-aided dietary assessment. *International Conference on Smart Homes and Health Telematics.* Springer, Cham.

[3] Sahoo, Doyen et al. (2019) FoodAI: Food image recognition via deep learning for smart food logging. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining.*

[4] DanB Food 101. https://www.kaggle.com/dansbecker/food-101 (2017)

[5] Colaboratory. https://research.google.com/colaboratory/faq.html (2020)

[6] Chollet, Francois. Building powerful image classification models using very little data. *Keras Blog.* (2016).

[7] Kappa, Avinash. Multiclass Classification using Keras and TensorFlow on Food-101 Dataset. https://colab.research.google.com/github/theimgclist/examples/blob/MultiClassTF2.0/community/en/multi_class_classification/food_classifier.i

[8] Advanced Guide to Inception v3 on Cloud TPU. *cloud.google.com* (2020).

[9] Stratospark. Food Classification with Deep Learning in Keras/Tensorflow *https://github.com/stratospark/food-101-keras/blob/5db3df9755e3697317d570d525a3b38125a1d81c/Food%20Classification%20with%20Deep%20Learning%20in%20Keras.md# and-Preprocessing-Dataset*

[10] Kagaya, Hokuto, & Kiyoharu Aizawa (2015) Highly accurate food/non-food image classification based on a deep convolutional neural network. *International conference on image analysis and processing.* Springer, Cham.

[11] Convolutional Neural Networks. *tensorflow.org*

[12] Image Classification. *tensorflow.org*

[13] Effective TensorFlow 2. *tensorflow.org*

[14] food101. *tensorflow.org*

[15] Load Images. *tensorflow.org*

[16] Datasets. *tensorflow.org*

[17] Chollet, Francois. deep-learning-models: VGG16, VGG19, and ResNet50. *github.com*