Complex Food Classification

Bryce Long, I-Sheng Yang, Priyal Mehta
molohov@stanford.edu, ishengy@stanford.edu, priyalm@stanford.edu

Introduction

Food image classification has many uses in everyday tasks. Users may want to find a picture of a specific dish on the internet using image search, or want to upload a picture to social media and have it suggest the name of the food based on the image. Classifying images of food also has use in meal tracker applications by improving the speed at which the user can log their meals.

In this project, we explore the complexities associated with training a neural network to perform food classification. Our final model uses an InceptionV3 network with weights pre-trained from ImageNet, which we incrementally retrain on our dataset (Figure 1).

Dataset

- The original dataset is Food-101, which contains 101 classes with 1000 images per class.
- We added an additional 4000 images from our custom web crawler script
- We pre-processed each image before training to be a standard 150x150 size
- We split the dataset using a 90/5/5 training/dev/test split, resulting in 99802/5050/5050 images respectively
- We used on-the-fly data augmentation techniques to make the most of small dataset (Figure 2)

Outcomes

- SGD had better performance than Adam/RMSProp optimizers (Figure 3), and momentum had a noticable impact on validation accuracy (Figure 4)
- We could only achieve around 65.69% accuracy on our test set
- Visualizing the output filters of the last convolutional layer does not show any food-specific features. It is likely that the network has not learned enough features to make accurate predictions. (Figure 5)
- In error analysis (Figures 6, 7, 8), we found the top 10 most accurately predicted classes tend to have simple, consistent shapes and well defined edges, while the bottom 10 have lots of variety in their presentation
- Future work would include expanding the dataset, committing to longer training time, trying different optimizers and deeper CNN architectures.

Methods

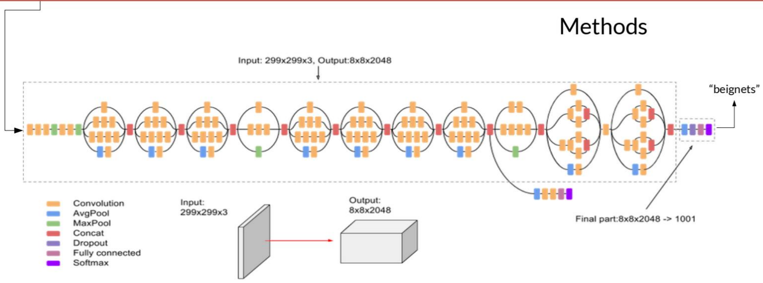

Figure 1:InceptionV3 architecture we used for our final model. 314 layers deep and 22 million trainable parameters. Customized to have 150x150x3 input features, 101 output classes

Figure 2: On-the-fly data augmentation samples. We used this technique to try to reduce our variance gap

Analysis

Figure 7: Top 10 performing classes from left to right: edamame, miso soup, pho, lobster bisque, crème brulee, spaghetti carbonara, seaweed salad, oysters, hot and sour soup, dumplings. Note the simple, consistent shapes

Figure 8: Bottom 10 performing classes from left to right: pork chop, ceviche, steak, omelette, foie gras, bread pudding, apple pie, tuna tartare, huevos rancheros, chocolate mousse. These classes have a wide range of different presentations.

Optimizer	L2	Minibatch Size	Epoch	Val Acc
SGD, LR 0.01, Momentum 0.8	0.005	64	30	0.6167
RMSprop, LR = 0.01	0.005	64	30	0.2380
Adam, LR = 0.01, $\beta_1 = 0.9$, $\beta_2 = 0.999$	0.1	64	30	0.2916
SGD, LR 0.01 , Momentum 0.8	0.2	64	30	0.5720
SGD, LR 0.01 , Momentum 0.8	0.005	32	30	0.6196
SGD, LR 0.01 , Momentum 0.8	0.005	128	30	0.5805
SGD, LR 0.01 , Momentum 0.9	0.005	32	30	0.5967
SGD, LR 0.2 , Momentum 0.8	0.005	32	30	0.6012

Figure 3: Optimizer search. SGD had larger impact than Adam or RMSProp optimizers.

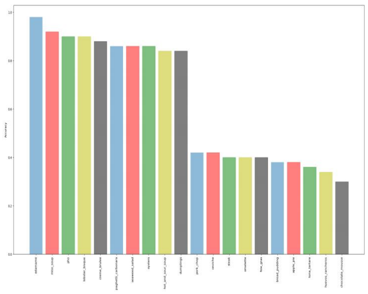

Figure 6: Accuracy of top and bottom 10 performing classes

Data Supplement	PreTraining	SGD, LR 0.01	L2	Epoch	Val Acc
None (Baseline)	ImageNet	Momentum 0.8	0.2	30	0.6090
Augmentation	ImageNet	Momentum 0.8	0.2	30	0.6130
Augmentation	ImageNet	Momentum 0.8	0.2	30	0.6190
Download + Aug.	ImageNet	Momentum 0.8	0.2	30	0.6290
Download + Aug.	from 0.6290 run	Momentum 0.7	0.2	5	0.6495
Download + Aug.	from 0.6495 run	Momentum 0.6	0.2	5	0.6531
Download + Aug.	ImageNet	Momentum 0.6	0.2	30	0.6569

Figure 4: Effect of momentum on the validation accuracy

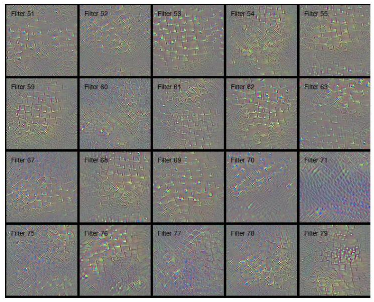

Figure 5: Visualization of the last layer of filters in the CNN. It is hard to discern meaningful food image features being learned by the network