From Jello Molds to Surf & Turf: Culinary Deep Learning

Introduction

In the modern digital world images are everywhere and play a central role in our communications. We share experiences and ideas with all from friends and family to customers and followers in image format. Within this communication, one of the things we photograph most is food. The hashtags #food, #foodporn, #instafood, #yummy and #foodie are 5 of the top 100 hashtags on Instagram and, combined, have been used over 1 billion times[1]. In spite of all of this sharing and the rise of the food blog, we also see cookbooks continue to be a medium in which we explore food[2]. Undeniably, food is a necessary part of our everyday lives, but something that also ties us culturally and emotionally.

As such, we have a long history of photographing our food with the first example attributed to scientist Joseph Nicéphore Niépce in 1832[3]. In the 70s the term “food porn” was used as “unhealthy for human consumption,” but was later reframed to comment on the aesthetically appealing qualities of the food[4]. On the topic of pornography Justice Potter Stewart said “I know it when I see it,” but whether you’re sharing your newest kitchen creation with your followers, or are a professional chef enticing bookstore browsers to pick up your newest cookbook (yes, we’re absolutely judging this book by its cover), we ask: What is it that makes our photos worthy of being #foodporn?

Image datasets of food exist, such as Food-101, however there are none, that I’m aware of, that attempt to capture the quality of the food. Fortunately, Reddit has the longstanding FoodPorn (FP) and ShittyFoodPorn (SFP) communities whose images I used in a classification task with convolutional neural networks. Professional food photographers and content creators such as bloggers or influencers can use the developed model to quickly judge the quality of their food photos.

---

[1] https://top-hashtags.com/instagram/

[2] https://www.nbcnews.com/business/consumer/recipe-success-cookbook-sales-survive-shift-digital-media-n900621

[3] https://firstwefeast.com/eat/2013/06/the-most-iconic-food-photographs-of-all-time/72330

[4] https://en.wikipedia.org/wiki/Food_porn

Data Wrangling

Using the Pushshift API I collected data on the submissions between 1/1/2015 and 9/31/20 from each subreddit including post title, score, id, image URL, post URL, number of comments and creation timestamp. The data gathered represents 333,866 submissions to FP and 317,486 SFP submissions during this time. From there I narrowed down the dataset by selecting the submissions in the 80th percentile by number of votes. As some submissions have been removed, or are not images, the final dataset resulted in 22,899 FP and 18,682 SFP images.

Exploratory Data Analysis

To explore this data set I examined 500 images from each FP and SFP. The images selected were those with the most votes from their respective subreddits. First I begin by examining 25 sample images from each class.

FP Sample Images

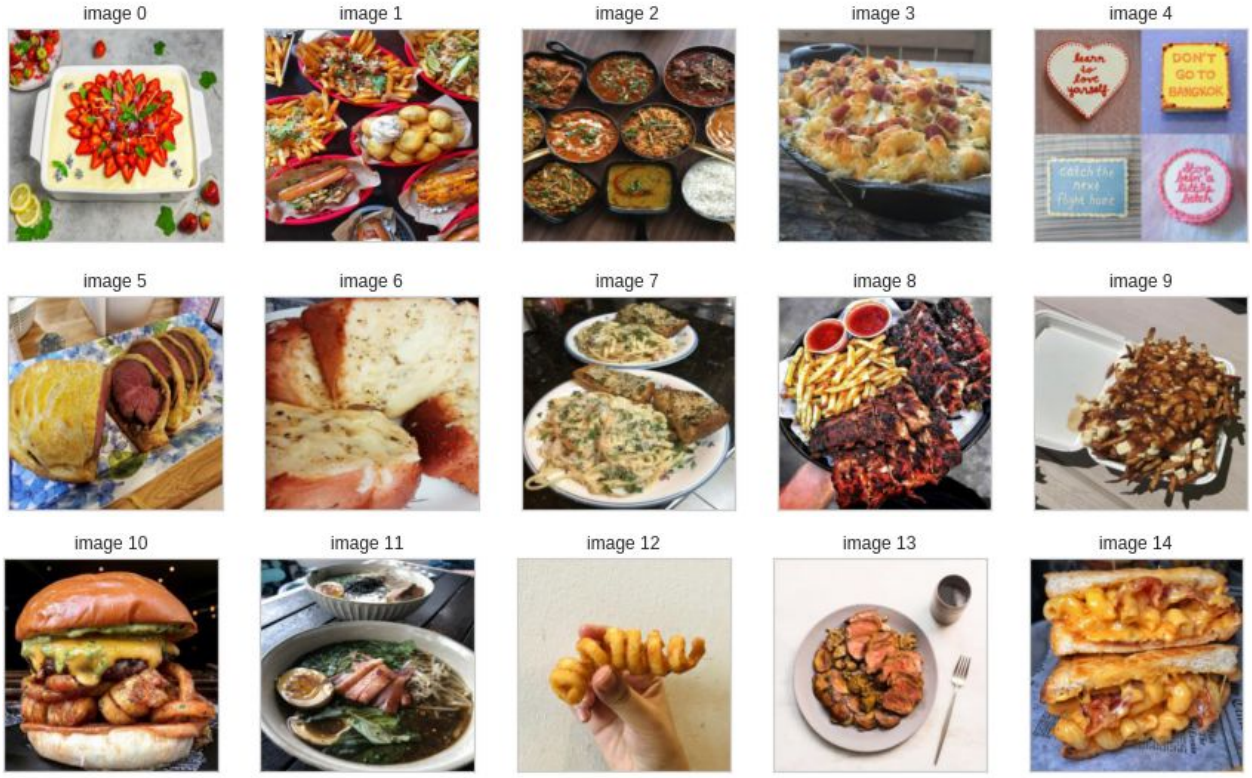


image 15

image 16

image 17

image 18

image 19

image 20

image 21

image 22

image 23

image 24

Starting with FP images, they are appetizing, colorful, and aesthetically pleasing. Even the poutine (image 9) or the mac & cheese bacon sandwich (image 14). Some are even artistic (image 0 and 18). But we also see some that are simply good food that the user wished to share, such as the perfectly fried curly fry (image 12), with no attention to the presentation - this fry is in the user's hand and their shadow is cast over the image.

SFP Sample Images

image 0

image 1

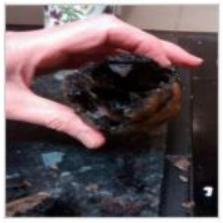


image 2

image 3

image 4

image 5

image 6

image 7

image 8

image 9

image 10

image 11

image 12

image 13

image 14

image 15

image 16

image 17

image 18

image 19

image 20

image 21

image 22

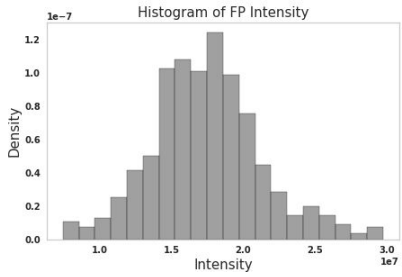
image 23

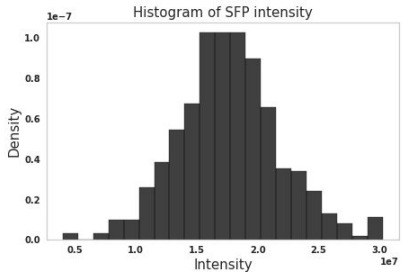
image 24

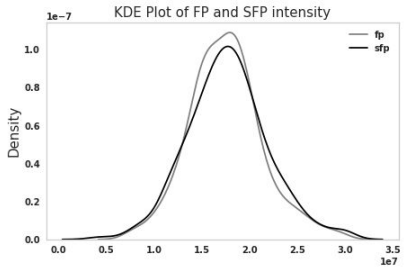
However, the posts on SFP have a different thematic variety to them. They tend to be darker with muted colors and little attention paid to the ascetics of the image. There are odd/gross combinations, such as chocolate and sprinkle chicken wings (image 18) or pasta shells filled with peanut butter/cheese spread/chicken paste/red pepper hummus/ketchup (image 10). Naturally there are "fails" (images 12 & 14), food gone bad (image 21) and undercooked food (image 7). Also present are lazy or "poor man's dinners". Examples of these are pizza with no crust (image 11), or a single piece of american cheese on white bread (image 15). Another theme in the data set is images of food that is well executed but a terrible idea (not present in these samples) such as, one of my personal favorites, a pasta and hot dog jello mold:

Image intensity

The below charts show image intensity calculated by summing total RGB pixel value (all images have been resized to 224x224). The intensity histograms show that the FP images are more centered along the spectrum whereas the SFP images have more images towards the tails of the distribution.



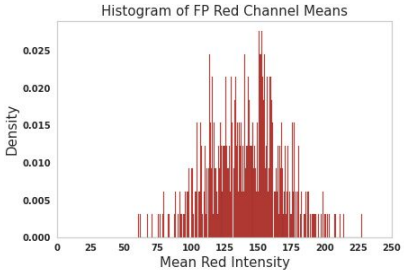


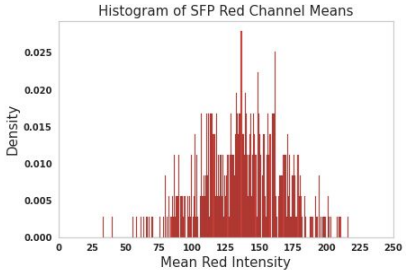


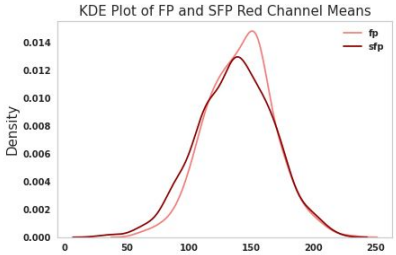
Color Channel Histograms

Red Channel

FP shows a higher proportion of deeper red values peaking around 155, whereas SFP peaks around 130. SFP has more images in the 50 to 120 range. FP then overtakes through 170 where they then become relatively even in distribution.

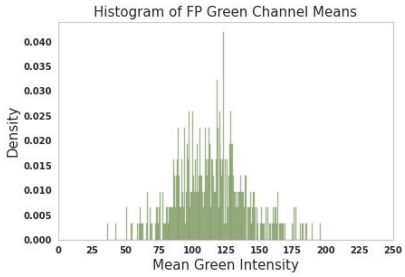


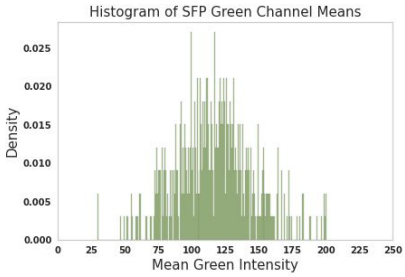


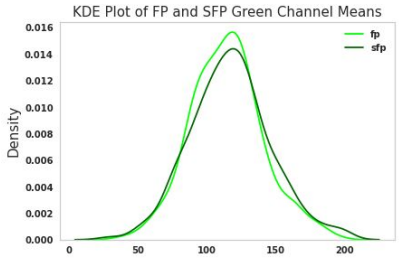


Green Channel

Both FP and SFP have a similar green distribution, with SFP having slightly more spread with more images at the higher and lower ends of the distribution.

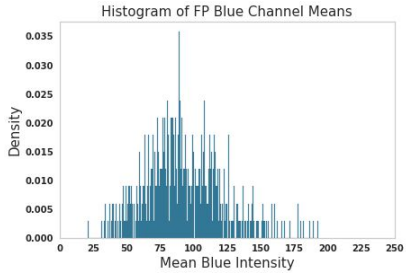


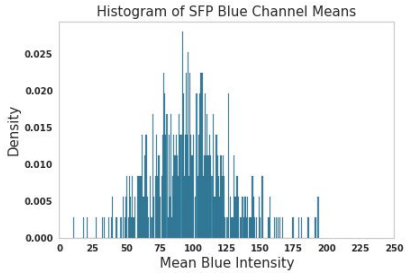


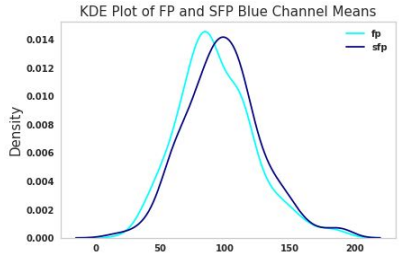


Blue Channel

Of all the color channels there is the largest difference within the blue channel. FP images clearly favor less blue with a peak frequency around 80 compared to the SFP peak at 100.







Average Image

The average images show that FP images take up more of the frame. Additionally the FP average image shows a higher contrast between the center of the image and the outer edges than seen in the SFP average image suggesting that the focus point of SFP images is not as centered.

Model Selection

To start, I created my own convolutional neural network with the below architecture to serve as a baseline and point of comparison.

Baseline Model Architecture

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 224, 224, 3)]	0
rescaling (Rescaling)	(None, 224, 224, 3)	0
conv2d (Conv2D)	(None, 222, 222, 64)	1792
max_pooling2d (MaxPooling2D)	(None, 111, 111, 64)	0
conv2d_1 (Conv2D)	(None, 109, 109, 64)	36928
max_pooling2d_1 (MaxPooling2	(None, 54, 54, 64)	0
conv2d_2 (Conv2D)	(None, 52, 52, 32)	18464
global_average_pooling2d (Gl	(None, 32)	0
flatten (Flatten)	(None, 32)	0
dense (Dense)	(None, 10)	330
dropout (Dropout)	(None, 10)	0
dense_1 (Dense)	(None, 1)	11

Total params: 57,525
Trainable params: 57,525
Non-trainable params: 0

Transfer Learning

Next, I used transfer learning to compare the results of pretrained networks versus a model of my own creation. Transfer learning is a method that takes a network that has been previously trained on a separate data set as a starting point and then training on a new data set. Typically the original data set is comparatively large and is generalized. In this case the original data set is ImageNet, with over 14 million images across 20,000 categories. The idea is that the

pretrained network has already learned general image features that we can take advantage of without having to start training from square one with a large data set of our own. Only the final prediction layer is trained while all other layers are frozen.

The pretrained networks I tested were **ResNet50**, **InceptionV3**, and **VGG16**. Both ResNet50 and InceptionV3 have shown extraordinary results on the Food-101 image dataset[5], and VGG16 has strong results in general image recognition with the ImageNet dataset.

During model selection I leveraged the Hyperband algorithm via Keras Tuner to optimize the learning rate and dropout rate hyperparameters. Hyperband is an hyperparameter optimization algorithm designed to speed up evaluation with adaptive resource allocation to focus on promising results while quickly eliminating unpromising ones[6]. The Hyperband trials were conducted with 4,500 training images and 1,000 validation images. Each model completed 30 trials across the below parameter space and was then trained with the best parameters for 100 epochs with early stopping using a patience of 10 while monitoring validation loss. The best performing model is ResNet50 with a 0.4 dropout rate and a 0.001 learning rate, resulting in a 0.4060 validation loss with a 0.8160 validation accuracy.

Hyper Parameter Space

Parameter	Values
Dropout Rate	0.0, 0.1, 0.2, 0.3, 0.4, 0.5
Learning Rate	0.1, 0.01, 0.001, 0.0001

Hyperband Trial Results

Model	Best Parameters	Validation Loss	Validation Accuracy
Baseline	Dropout = 0.1 Learning Rate = 0.01	0.5184	0.7480
VGG16	Dropout = 0.3 Learning Rate = 0.01	0.4877	0.7690
ResNet50	Dropout = 0.4 Learning Rate = 0.001	0.4060	0.8160
InceptionV3	Dropout = 0.4 Learning Rate = 0.001	0.7270	0.6670

[5] https://github.com/Pyligent/food101-image-classification
[6] http://web.eecs.umich.edu/~mosharaf/Readings/HyperBand.pdf

Training & Results

After selecting the ResNet50 model and hyperparameters I conducted a first pass of training on 13,000 images, with the same 1,000 image validation set and a separate test set of 1,000 images. Below is a summary of the model architecture with 2,049 trainable parameters.

Transfer Model Architecture

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 224, 224, 3)]     0
_________________________________________________________________
resnet50 (Functional)        (None, 7, 7, 2048)        23587712
_________________________________________________________________
global_average_pooling2d (Gl (None, 2048)              0
_________________________________________________________________
dropout (Dropout)            (None, 2048)              0
_________________________________________________________________
dense (Dense)                (None, 1)                 2049
=================================================================
Total params: 23,589,761
Trainable params: 2,049
Non-trainable params: 23,587,712
_________________________________________________________________
```

The following charts show loss and accuracy during training. I allowed for 1,000 epochs of training while using early stopping to monitor validation loss with a patience of 20. Model checkpoints were used to monitor the loss and save the model with the lowest loss. However, the model only required 3 epochs of training before reaching a local minimum loss. The best validation loss was 0.3812 with a 0.8380 validation accuracy. The test loss was 0.3746 with a 0.8310 test accuracy.

Initial Training Loss and Accuracy

Fine-Tuning

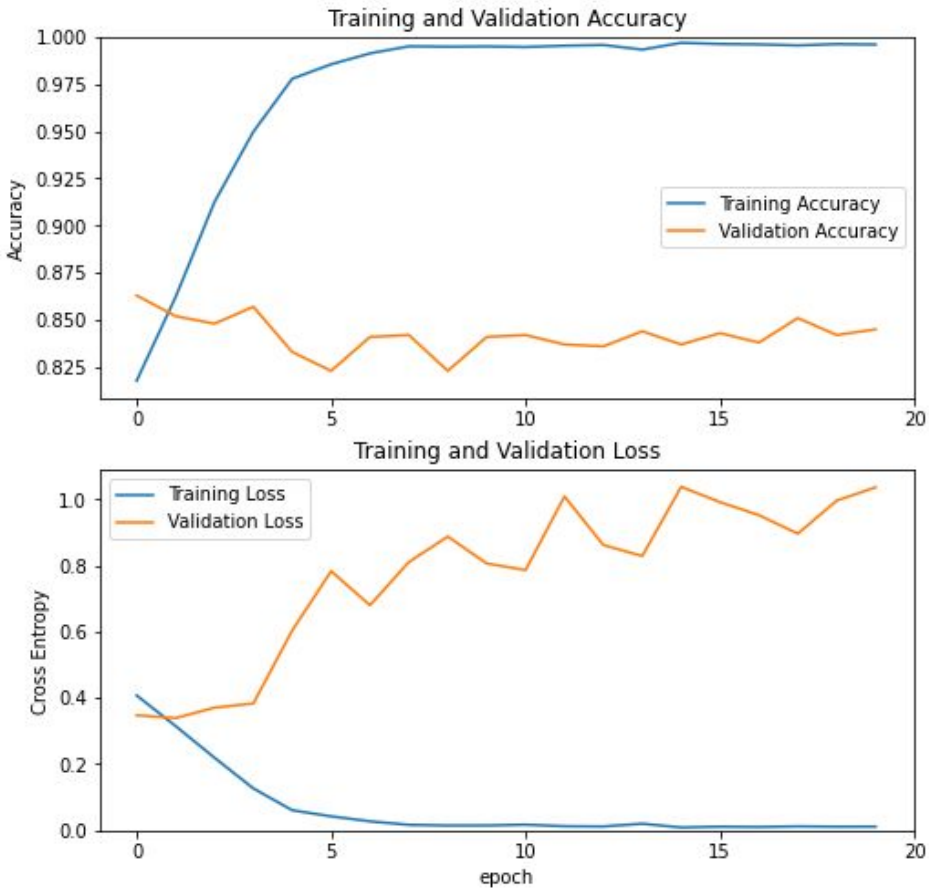
A technique to further improve results after completing the initial training in transfer learning is to unfreeze some of the base layers and conduct a round of fine-tuning with a low learning rate. I tested this fine-tuning with 20, 86, 150, and 214 trainable layers. I allowed this to train for 20 epochs, this time without early stopping, but still utilizing checkpoints. I found that fine-tuning training on 150 of the base layers was optimal.

Fine-Tuning Layer Selection

# of Trainable Layers	Best Validation Loss	Validation Accuracy	Test Loss	Test Accuracy
214	0.3484	0.844	0.2102	0.914
150	**0.339**	**0.852**	**0.1814**	**0.937**
86	0.3478	0.849	0.2118	0.917
20	0.3687	0.834	0.2368	0.904

Once again, minimal training was required. After fine-tuning 150 layers, validation loss was reduced to 0.3390 and validation accuracy improved to 0.8520. Test accuracy improved to 0.9370 with a 0.1814 loss. Training accuracy and loss charts are below followed by a summary of all training.

Fine-Tuning Loss and Accuracy

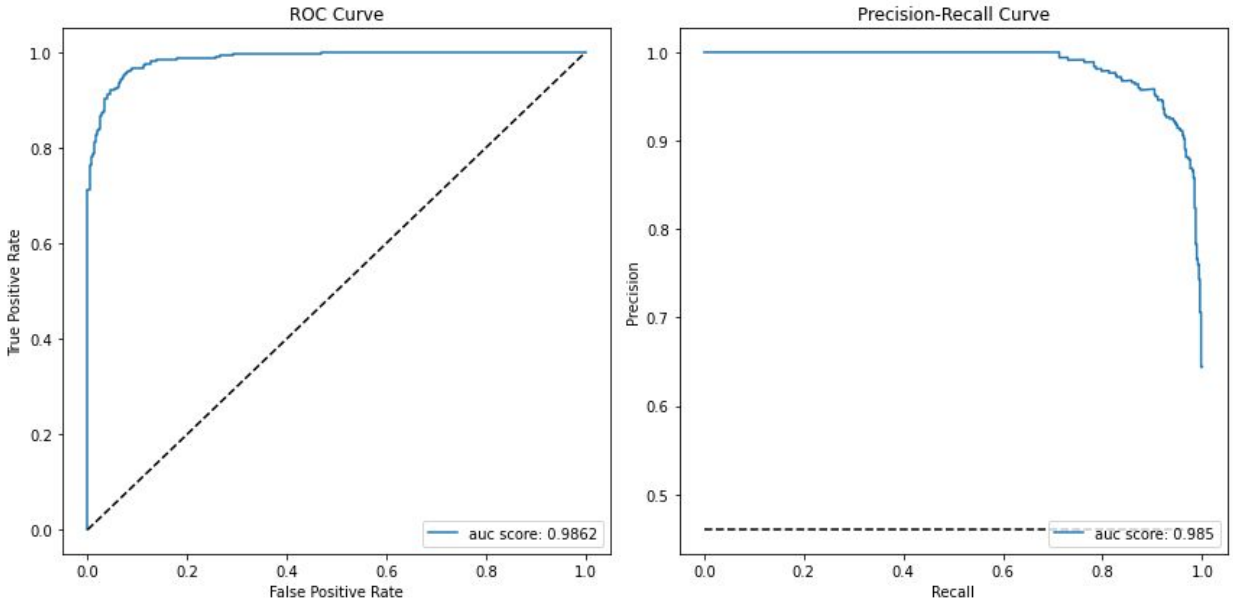


Training Results

	Validation Loss	Validation Accuracy	Test Loss	Test Accuracy
Initial Training	0.3812	83.80%	0.3746	83.10%
Fine-Tuning (150 layers)	0.3390	85.20%	0.1814	93.70%
Percent Change	-11.07%	1.67%	-51.58%	12.76%

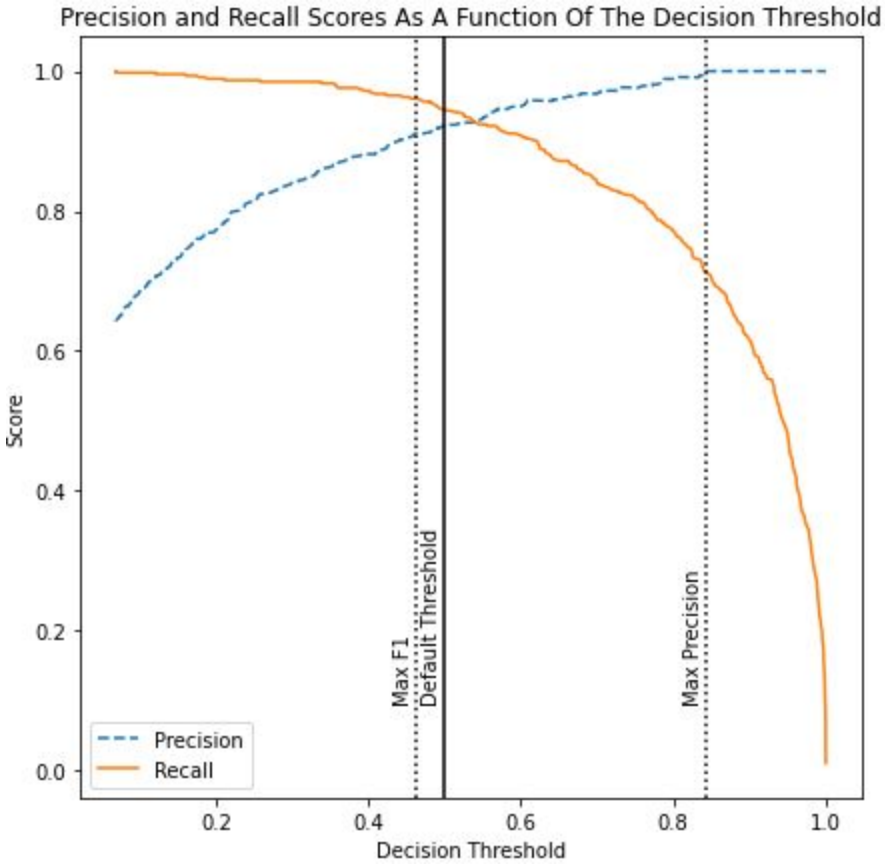
Model Validation

The fine tuned model scores well on the test data set with a ROC auc score of 0.9862 and a Precision-Recall auc score of 0.985 with 1 representing a perfect score for both metrics.

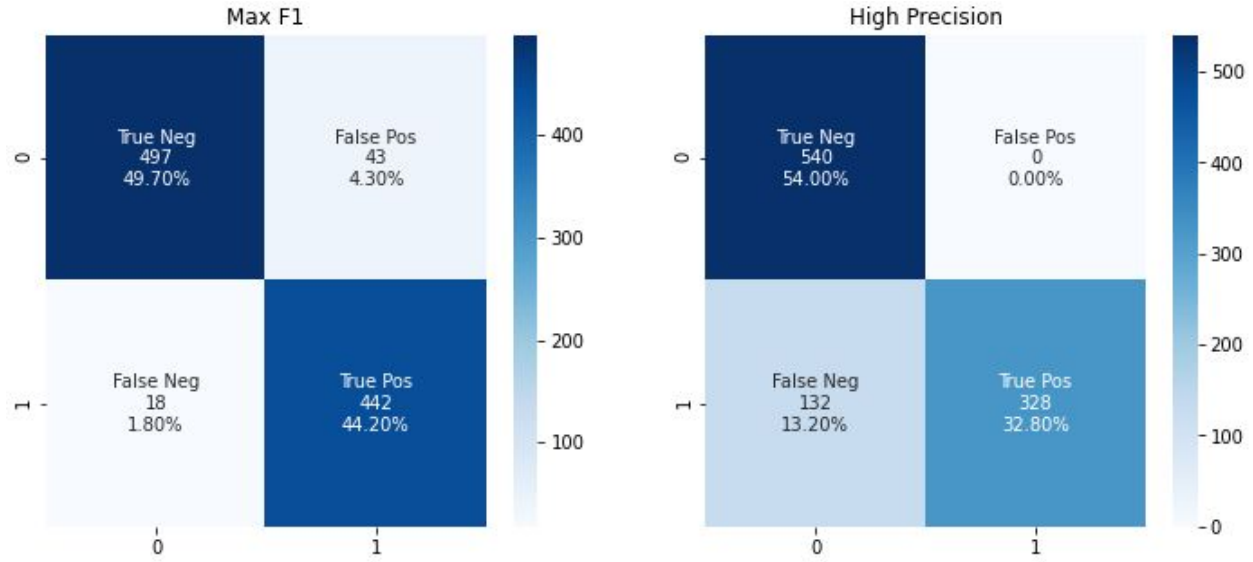


Decision Threshold

In choosing the decision threshold for the model I propose two options. The first increases the decision threshold to 0.84 to reach a 1.00 precision at the expense of a 0.71 recall. This model would be for food professionals such as advertisers, authors, or chefs who require assurance that their food photographs are of the best quality. The second model reduces the threshold to 0.46 in order to maximize both precision at 0.91 and recall at 0.96 (effectively maximizing the F1 score). This is for general purpose users such as bloggers or influencers where content creation and volume is more important.



Below are the confusion matrices for both scenarios. Maximizing the F1 score minimizes the overall error rate while the high precision scenario has a 0% false positive rate in exchange for a 13.20% false negative rate.



Model Exploration

To get a better understanding of this model we'll look at class maximization and the misclassified images. However, it is first important to dig into what makes ResNet different from other deep learning architectures.

ResNet is a residual learning framework designed for easier training of very deep neural networks. The creators of ResNet note that a deep network is of critical importance, but they identify a phenomenon that as networks get deeper, adding more layers leads to higher training error not caused by overfitting. With the intuition that a deeper network should perform no worse than a shallower network, the solution used in ResNet is to create a "shortcut" which adds the output from previous layers to the outputs of subsequent layers. By feeding forward this identity mapping, the network learns the residuals (difference between error and prediction) and can preserve the performance of the lower layers. The identity mapping does not add parameters or computational complexity and allows ResNet to be both deeper and less complex than other state of the art architectures.[7]

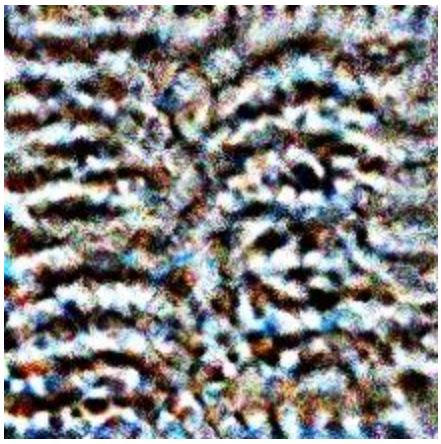
Class Maximization

A common technique to try to understand what a model "sees" is to generate images that maximize convolution filters. The final layers of networks learn complex patterns and resemble a representation of the visual world. Here are a few examples from VGG16 pretrained on ImageNet which resemble fish.

However, due to the structure of ResNet, this technique does not provide the same visual understanding. Instead, we can generate a loss function that optimizes a particular class, take the gradients of an image of random noise with respect to the loss function, and iteratively add the gradients multiplied by the learning rate to the image. Below are generated images.

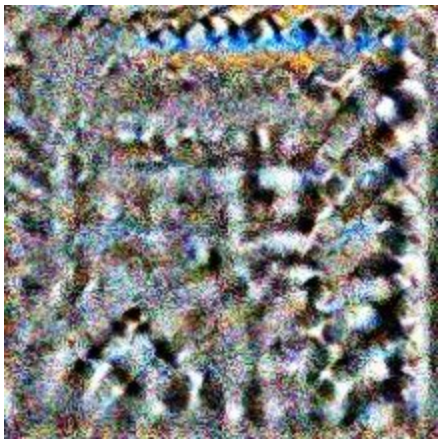
[7] https://arxiv.org/abs/1512.03385

FP



Predicted Probability: 0.00025487

SFP



Predicted Probability: 0.9999112

Empirically we can see the FP image has more texture, less noise, and is brighter overall with more contrast. These observations are validated by the numerical data points in the below table. Conducting Bartlett's Tests on the individual color channel resulted in small p-values indicating that the variances are different. Similarly, a t-test showed small p-values when comparing the red and blue channels means, however the green channel test has a 0.1170 p-value indicating that should not reject that the green means are the same.

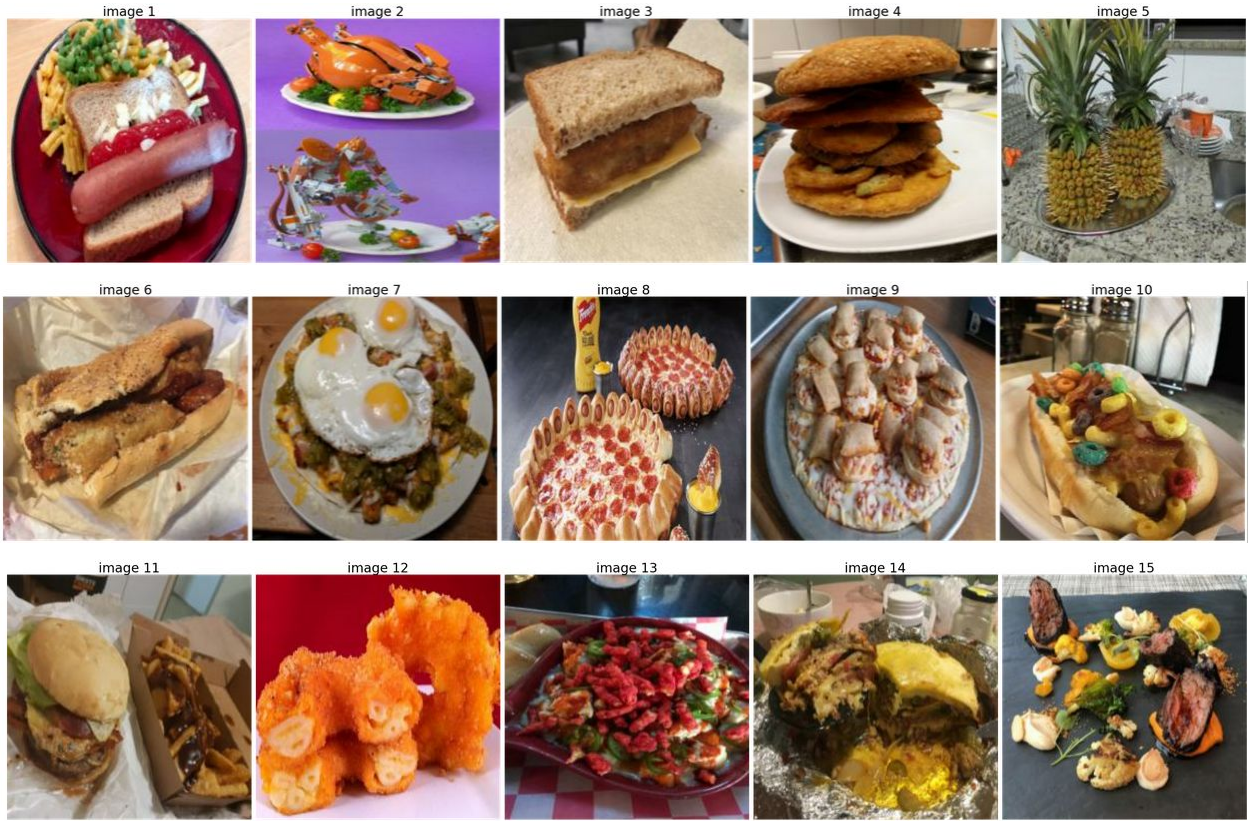
	FP			SFP		
	Red	Green	Blue	Red	Green	Blue
Total	6,074,429	6,028,425	6,040,340	5,966,976	5,992,508	5,952,625
Mean	128.9989	128.0219	128.2750	126.7170	127.2592	126.4122
Standard Deviation	77.7777	83.0088	86.1889	64.0337	65.2497	68.6446

Misclassifications

Below are fifteen false negative and fifteen false positive misclassifications. Without prior examination of the class maximized images it is difficult to tell these misclassifications apart. What we can learn from the false negatives is that the model is missing context. These are generally colorful or bright with a high level of contrast. But the model doesn't know, for example, that image 5 is a pineapple constructed out of toothpicks and olives or that image 10 is a chicken sandwich with children's cereal. Many of the false negatives are images that are of

well executed/constructed and are visually pleasing, but certainly not something you would consider ordering at a restaurant or even consuming at all.

False Negatives (Incorrectly labeled as not SFP)



In looking at the false positives, we can see a different story. Many of these images are blurry, have poor lighting, or are washed out with minimal color and contrast. These items are not immediately recognizable, and the image focus is not on the food. (From EDA we saw that the average FP image took up the majority of the frame.)

False Positives (Incorrectly labeled as SFP)

Beyond the basics of color and contrast we can also see that the model is sophisticated in composition i.e. the elements of the image, such as lines, ratios, colors, or shapes, are arranged in a way that is appealing to the viewer. False negatives 1, 8, 12, and 15 have aspects of professional plating which improves their composition. On the other hand, this is largely missing from the false positives. A prime example is the false positive image 13. These are cakes that are appetizing, beautiful, and clearly made by a skilled hand. However, the image is askew, the only colors are the cakes, and white space takes up the majority of the photo.

Conclusion

Good food images have color contrast, a point of focus that is front and center, and proper lighting and the misclassifications validate that the model has learned this. However, we saw that it can still be tricked when using proper photography techniques on monstrous food creations.

This model utilized the ResNet50 architecture pre-trained on the ImageNet data set. It achieved a 93.70% test accuracy after training the final layer and conducting a second fine-tuning training on the top 150 layers (out of 214). This model is useful for anyone whose profession involves food photography from cookbook publishers and food advertisers to influencers and food bloggers.

Considerations for improvement include training with a larger data set. While I collected 41,537 images, due to computational limitations, I was only able to train with 13,000. In a similar vein, the images were scaled to 224x224, but were originally of a higher resolution; increasing the resolution of the training data set can provide incremental improvements. Finally, additional network architectures can be explored. The ResNet team has shown success with up to 152 layers. On the other hand, if one wished for a more lightweight model, it may be possible to

refine a shallower model and find similar results; the baseline model I created performed within the ranges of the other state-of-the-art pretrained networks while having fewer parameters on the order of magnitudes.