Computer Vision and Food Classification
Springboard Datascience Intensive Capstone Report

Simon Bedford

01 September 2016

Contents

1 Introduction

Cooking, and more broadly food in general, is a very popular content category online. Indeed, according to the FT.com, cooking has become a key category to focus on for driving growth in both audiences and advertising alike.

When someone is looking for a specific recipe, or even just inspiration, it is now perfectly natural for them to go online and consult one of the myriad recipe curating and aggregating websites.

From big names such as Allrecipes.com, Food.com and the Food Network, to more focused sites like Epicurious and Serious Eats, there is an increasing proliferation of online properties where consumers can find, view and share recipes and cooking ideas. Nowadays even companies like Youtube and Buzzfeed are trying to drive traffic through cooking videos and websites [1].

One only has to look at some of the numbers to see how big these websites are:

- In December 2016, allrecipes.com drew 50m unique visitors, a 23% increase from the same month the previous year [1]
- Epicurious boasts 30,000 professionally tested and created recipes, along with 150,000 member-submitted recipes.[2]
- The New York Times cooking website receives 7-8m monthly visitors, has 650,000 subscribers and stores more than 17,000 recipes online[1]

Perhaps even more impressive is the related growth in photo and video content:

- There are 168m+ posts on Instagram with the hashtag #food, and 76m+ for #foodporn [3]
- Food related videos were viewed 23bn times in 2015, a 170% increase over 2014 [1]

Furthermore, and inline with broader trends, mobile is where this contents is increasingly produced and consumed. Allrecipes.com for instance states that 66% of their page views are now via mobile devices. [4].

Given both the rapid growth in and focus on food, and the increasing dominance of photo and video media, there is a clear opportunity for computer vision techniques to be applied to the world of food content.

For example, image recognition and classification techniques could be integrated into a seamless mobile experience to enable faster and more accurate recipe search and suggestions, and even enable content producers to display more relevant and targeted advertising specific to the types of food that a user is interested in.

However applications are not just limited to search and advertising, but can also have health benefits too. For example food classification algorithms could help people keep food diaries and daily calorie counts, an idea explored by both Microsoft [5] and Google research teams [6].

The aim of this project is as proof-of-concept, to explore whether it is possible to create a food classification algorithm that could feasibly be implemented into a production environment to enable automated food recognition and recipe retrieval.

As such, it is key to consider the right measures of performance:

1. Classification Accuracy
2. Resources and Cost
3. Ease of Production Deployment

Classification Accuracy

In order to measure performance, we could use a simple metric like:

$$Simple\ Accuracy = \frac{\#\,Correct\ Predictions}{All\ Predictions}$$

However this is a slightly naive measure of accuracy, and instead we will use the F1-score as a more refined measure:

$$F1 - Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

To better understand the F1-score, consider a very simple binary classifier attempting to distinguish between pizza and not-pizza. There are four possible prediction outcomes which we can summarize in the Confusion Matrix below:

		True Categories	
		Pizza	***Not Pizza***
Classifier Predictions	***Pizza***	True Positive (TP)	False Positive (FP)
	Not Pizza	False Negative (FN)	True Negative (TN)

Table 1: Confusion Matrix Truth Table

We can then define:

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \quad \text{and} \quad Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

One way to think about Precision is that it measures, out of all images predicted to be pizza, what proportion actually are pizza. In some senses this tells us how noisy our pizza predictions are, or how much does the classifier confuse other types of food with pizza.

Similarly Recall asks, out of all true pizza images, what proportion were correctly identified as being pizza. This is a measure of how good our classifier is at correctly identifying pizza images and placing them in the correct category.

The F1-score is the harmonic mean and is a measure of the balance between Precision and Recall. It is measured on a scale from 0 to 1, with 1 being the best possible score.

Resources and Cost

For Resources what we care about is how costly is it in terms of both time and money to create an accurate classifier. From a practical perspective, any company evaluating implementing this type of technology would need to perform some type of ROI analysis, including taking into account the costs of training an algorithm with sufficient accuracy.

During our analysis we will measure the time taken in training different algorithms, and also the cost of any external computing resources required. For the sake of this project we will ignore both man-hours and also negligible costs such as electricity etc.

Ease of Production Deployment

It is also important to consider how practical it would be to implement a successful model in a production environment in a way that makes sense and would benefit the customer or user experience. Some possible questions to consider are:

- Can the algorithm perform accurate classification in real time?
- Is it possible to run the model locally on a mobile device, or does it need to be run on a central server?

During this project we will not spend a lot of time analyzing these questions in detail, however in the conclusions and recommendations we will briefly discuss some possibilities for integrating the best model into production systems, along with associated challenges.

Classification Models

Finally, with regards to models, we take three approaches, which will be explored in more detail in later sections:

- Traditional machine learning
- Deep learning with Neural Networks
- A hybrid approach fusing deep-learning with traditional machine learning

As will be seen, deep-learning techniques enable us to achieve impressive results in a relatively short amount of time, with a good possibility of applicability to different problems and contexts.

2 Exploratory Data Analysis

The dataset used is The Food-101 Data Set from the ETH Zurich Computer Vision Laboratory [7]. This dataset contains 101 food categories, with 1,000 images per category for a total of 101,000 images. Each image has a maximum side-length of 512 pixels, and the total dataset comprises approximately 5GB of data.

Early on in the project, it was decided that, given the available computing resources, the original dataset was too large for practical experimentation.

In order to create a smaller dataset the existing categories were compared to the 100 most popular recipe-related Google search terms in 2015 in the US. In total, there was a direct match with 13 classes in the dataset, and for the remainder of the project, the analysis is focused on the top 12 matching categories:

Pork Chop	Lasagna	French Toast
Guacamole	Apple Pie	Cheesecake
Hamburger	Fried Rice	Carrot Cake
Chocolate Cake	Steak	Pizza

Noisy Data

The images in the Food-101 dataset are of mixed quality. Some are very clear, well-lit and framed on the food item in question. Others however are more noisy, poorly lit, containing other irrelevant items and, in some cases, mislabeled. Some examples of both high and low-quality pictures are shown in figure 1.

Image Sizes

Looking at the different image sizes and shapes within the dataset, we see that the top 5 most common shapes are:

Shape	% of Images
(512, 512, 3)	59.4
(384, 512, 3)	15.8
(512, 384, 3)	6.4
(512, 382, 3)	2.8
(382, 512, 3)	2.5
Other	13.2

The most common image size is 512 x 512 accounting for 60% of the images, and so the whole dataset was standardised to this size for future analyses.

RGB Histograms

One of the first exploratory steps was to compare RGB histograms for the images across the different food categories. As can be seen in figure 2, there are clear differences in the distribution of pixel values for Red, Green and Blue histograms between food categories.

Figure 1: Select pictures of Pizza, Hamburger and Steak categories, illustrating both high and low-quality images.

Principal Component Analysis

The next step was to visualize the data by plotting images in 2-D and see if any clear patterns emerged, for example images from the same class being clustered together.

For dimensionality reduction to 2-dimensions, the following approach was used:

1. Initial reduction using Randomized PCA with n_components = 50
2. Further reduction to 2-D using TSNE (t-distributed Stochastic Neighbor Embedding)

Given that in total the dataset comprises 12,000 images, for practical purposes we used only the 40 nearest neighbors to the mean image for each category (480 images in total).

As we can see from figure 3, based upon the extracted components, there are no discernible patterns or groupings in two-dimensions.

3 Traditional Machine Learning

For the purposes of this project, when we refer to Machine Learning, we mean the use of pre-selected features that are used in conjunction with computer-enabled algorithms to attempt to solve our classification problem. In particular, we will focus on using *Supervised Learning* algorithms.

Figure 2: Histograms of RGB pixel values for mean images for the Guacamole, Chocolate Cake and Pizza categories.

One of the key points for machine learning is that we must make a conscious choice regarding the types of features we wish to use prior to training the model. In some cases the features are extracted manually or semi-manually, although in other cases we may also rely on unsupervised models to extract other types of features.

A successful model should be able to create a general rule for correctly differentiating between food classes based upon the provided features. Thus we need to try and identify features that are similar enough for images belonging to the same category, but also different enough between classes to enable differentiation. A description of the types of features used during the project can be found in table 2.

For the very earliest attempts, some features were tested individually, but in most cases, different types of features were chained together before being used for model training.

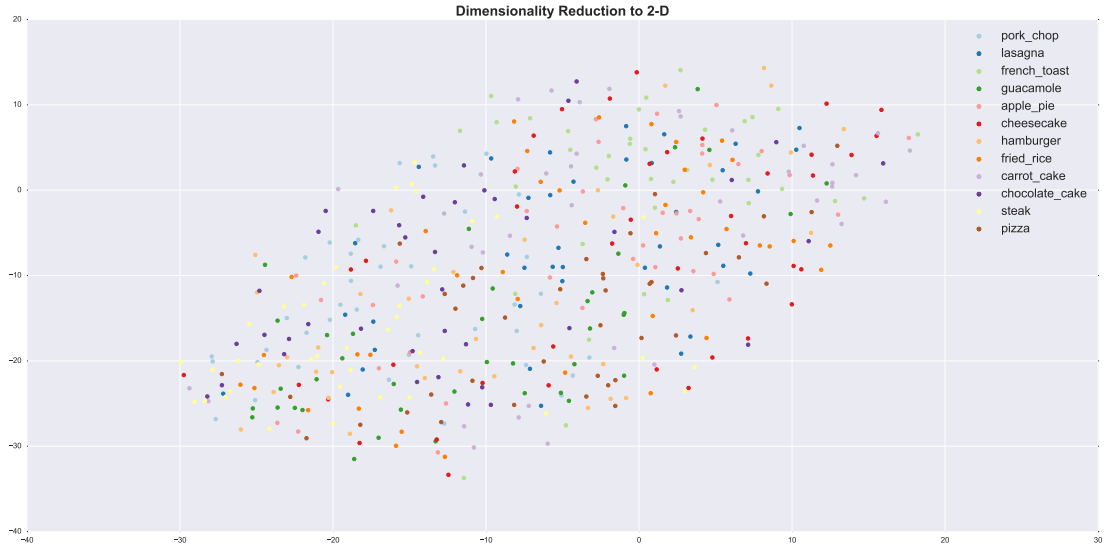
In some cases features were extracted from the image as a whole, and in other cases the images were divided up into sub-sections or windows and then features extracted from each sub-section before being chained together to create one long vector of features for the whole image.

Models

As well as testing a number of combinations of different features, we also started by testing a number of different models on the simplest features, including:

- k-Nearest Neighbours

Figure 3: Images plotted in 2-D; First PCA was applied to reduce from 786,432 to 50 dimensions, and then TSNE further applied to reduce to 2 dimensions.



- SVM: Linear & Polynomial Kernel
- Decision Tree
- Random Forest
- ADA Boost Classifier
- Gaussian, Multinomial and Bernoulli Naive Bayes
- Linear & Quadratic Discriminant analysis

Based on the results of training and testing these models, see figure 4, it was decided to mainly use Random Forest classifiers, as the RF gave the best accuracy and F1 scores using the simple features, and also had a very fast training and testing time.

All models, both supervised and unsupervised, were implemented using the python Scikit-learn library.

In total we tried approximately XX approaches using a mix of features and classifiers. The full list of approaches and results can be seen in Appendix X, however the results of a selection of approaches is shown in table 3.

The best machine learning approach was based upon the following procedure:

Feature extraction

1. Split each image into non-overlapping boxes of side 32 pixels; for our 512x512 images, this results in a grid of 256 boxes per image
2. For each box, extract the following features:
    - Average red pixel value

Table 2: Description of the different types of features used for machine learning attempts.

RGB Histograms	As we saw in figure 2, it initially appears that different types of food images have sufficiently different distributions of Red, Green and Blue pixel values to potentially be able to differentiate between them.
Pixel Values	In some models, we used individual pixel values as features, typically for images scaled to a smaller size.
Edges	These features are extracted using image processing algorithms that look for discontinuities in image brightness.
Corners	The intersection of two edges; also extracted algorithmically.
Meta Approaches	One approach that can sometimes be of benefit is applying unsupervised learning techniques in order to reduce the dimensionality of other extracted features in order to make supervised learning more practical and faster to run. Two approaches tried were Principal Component Analysis as well as K-means Clustering.

- Average blue pixel value
- Average green pixel value
- Number of edges (using skimage canny edges algorithm)
- Number of corners (using skimage corner-fast algorithm)

3. Concatenate all features together into one long feature vector of length 1,280

Classifier

The model used was a Random Forest. Parameter optimization was performed using a grid-search on the number of estimators and maximum tree depth, and then a classifier was trained using the best values. The results obtained are shown in table 4.

It can also be instructive to look at the confusion matrix and see which categories the classifier tends to get right or confuse with other classes (see figure 5).

We can see that Pizza and Cheesecake appear to be the categories where the classifier performs best, although with correct predictions for 133 and 126 instances out of 250 respectively, there is still a long way to go to train a model that could be usefully applicable.

In terms of where the classifier struggles, by looking at the darker shaded cells off-diagonal, we see that the model has particular difficulty distinguishing between Steak and Pork Chops, and also between the different categories of cakes.

4 Deep Learning

The second approach was based upon deep learning techniques, and in particular Convolutional Neural Networks (CNNs).

Currently CNNs are recognised as being state of the art models for image classification [9], and in fact various image classification and captioning competitions have been won consistently over the past few years using CNNs [10].

Figure 4: Comparison of F1 score and training time of different classifiers using RGB Histogram-based features.

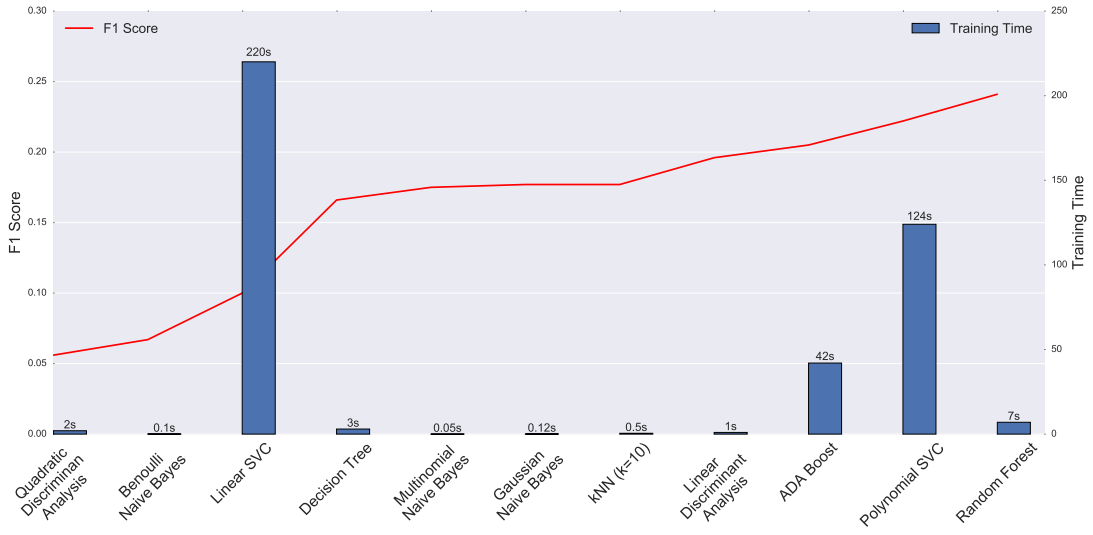
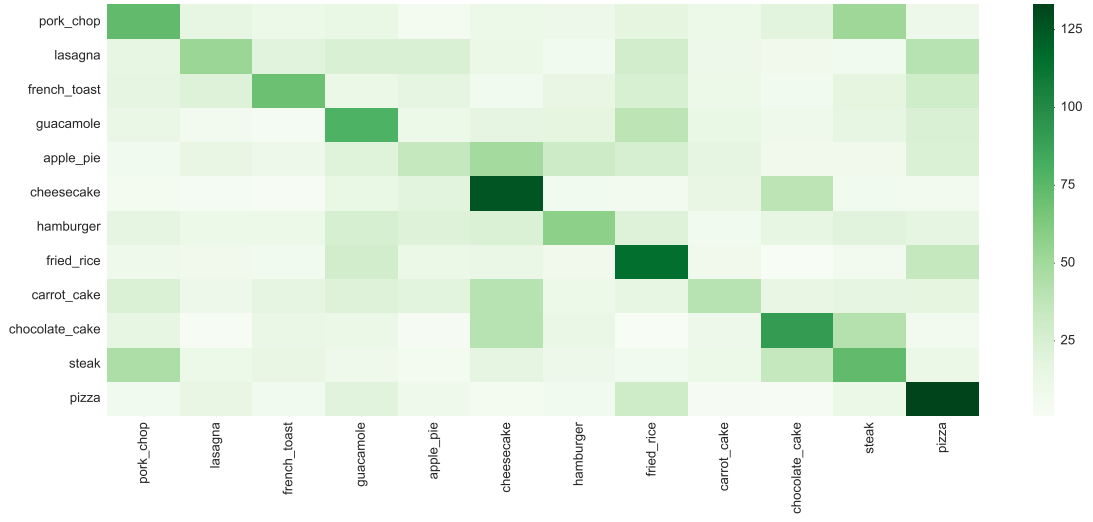


Figure 5: Confusion matrix for best machine learning classifier predictions. Darker shaded cells indicate more correct predictions for that class.



One of the biggest challenges is that training a full CNN from scratch requires a lot of data, and takes a long time. For example, VGGNet, the winning model of ImageNet 2014, was trained on 1.3M images, for approximately 2-3 weeks [9]. However all is not lost as there exist a number of techniques whereby it is possible to take a pre-trained model and adapt it to a different but related problem through a series of techniques called Transfer Learning [11].

Table 3: Results of varying combinations of features and supervised learning models.

Classifier	Features	F1 Score
Random Forest	100-d vector extracted using K-Means Clustering on individual pixels	0.09
Random Forest	100-d vector extracted using PCA on individual pixels	0.07
Random Forest	Images re-scaled to 32 x 32, pixels as features	0.19
Random Forest	RGB Histogram for complete image + individual pixels of 32 x 32 image	0.26
Random Forest + Grid-search	RGB Histogram + Edges + Coreners; zero-variance features removed; PCA to reduce to 300-d vector of features	0.27
Random Forest + Grid-search	Image split into 32x32 boxes; for each box extract Avg Red, Green, Blue pixel value, # Edges, # Corners	0.31
Random Forests trained on each feature type; Bayesian Net training on probabilities from each classifier	Image split into 32x32 boxes; for each box extract Avg Red, Green, Blue pixel value, # Edges, # Corners	0.29
Random forest trained on segments; overall prediction based on average of segment predictions.	Image split into segments (using SLIC algorithm) and for each segment use Avg, Max, Min, Range of Color values & normed histograms for RGB	0.21

Feature Extraction

The very first approach was using pre-trained CNNs for feature extraction, and then training a Linear SVC model using the obtained features.

The pre-trained models used were AlexNet [12] and VGGNet [13], and for each network, three sets of features were used based upon the outputs of Fully-Connected layers 6, 7 and 8.

The model took less than 1 hour to train, and the best result came from features extracted from Fully Connected layer 7 from VGG Net. The results obtained are shown in table 5, and we see that, in a very short space of time, we have achieved significant improvements vs. the top machine learning approach, more than doubling our overall F1 score from 0.32 to 0.69.

Fine Tuning

The final approach was to attempt to fine-tune the weights of a pre-trained network using our problem-specific dataset of food images. For this approach we worked exclusively with the AlexNet model.

The first attempt was based on fine-tuning the whole network using just the original dataset (although taking advantage of Caffe's inbuilt data augmentation during training). The model was trained for 30,000 iterations in total, and appeared to have already converged by iteration 10,000, although the overall F1-score obtained was only 0.50.

We then tested a number of additional strategies using a combination of the following ideas:

- Using only the original dataset vs. using an augmented dataset
- Fine-tuning the whole network vs. fine-tuning only the fully-connected layers

The best results were obtained using the following procedure:

Table 4: Per-class results from the best machine learning classifier.

Class	Precision	Recall	F1	Support
Pork Chop	0.29	0.29	0.29	250
Lasagna	0.31	0.21	0.25	250
French Toast	0.37	0.28	0.32	250
Guacamole	0.28	0.32	0.30	250
Apple Pie	0.19	0.14	0.16	250
Cheesecake	0.35	0.50	0.41	250
Hamburger	0.30	0.23	0.26	250
Fried Rice	0.35	0.46	0.39	250
Carrot Cake	0.26	0.16	0.20	250
Chocolate Cake	0.36	0.36	0.36	250
Steak	0.26	0.29	0.28	250
Pizza	0.38	0.53	0.44	250
Overall	**0.32**	**0.31**	**0.31**	-

Table 5: Per-class results from the Linear SVC + CNN Features.

Class	Precision	Recall	F1	Support
Pork Chop	0.55	0.51	0.53	211
Lasagna	0.67	0.61	0.64	187
French Toast	0.64	0.62	0.0.63	208
Guacamole	0.90	0.89	0.90	189
Apple Pie	0.55	0.60	0.57	194
Cheesecake	0.70	0.67	0.68	206
Hamburger	0.71	0.76	0.73	200
Fried Rice	0.81	0.88	0.84	213
Carrot Cake	0.66	0.65	0.66	205
Chocolate Cake	0.70	0.76	0.73	178
Steak	0.54	0.51	0.52	211
Pizza	0.82	0.78	0.80	198
Overall	**0.68**	**0.69**	**0.69**	-

1. Data Pre-Processing

All images were first re-scaled to 256 x 256, and each class of 1,000 images was split into:

- Training - 664 images
- Validation - 136 images
- Testing - 200 images

2. Data Augmentation

Training and validation images were further augmented to generate a total of 16 images per input image:

- Original image + mirror image
- 3 Lightened Images + their mirror images
- 3 Darkened Images + their mirror images
- Original image rotated by 180 degrees + mirror image

Note: The image lightening and darkening was performed using the adjust_gamma function from the skimage exposure module with fixed gamma values (Lightening = [0.45, 0.65, 0,85], Darkening = [1.25, 1.50, 2.00]).

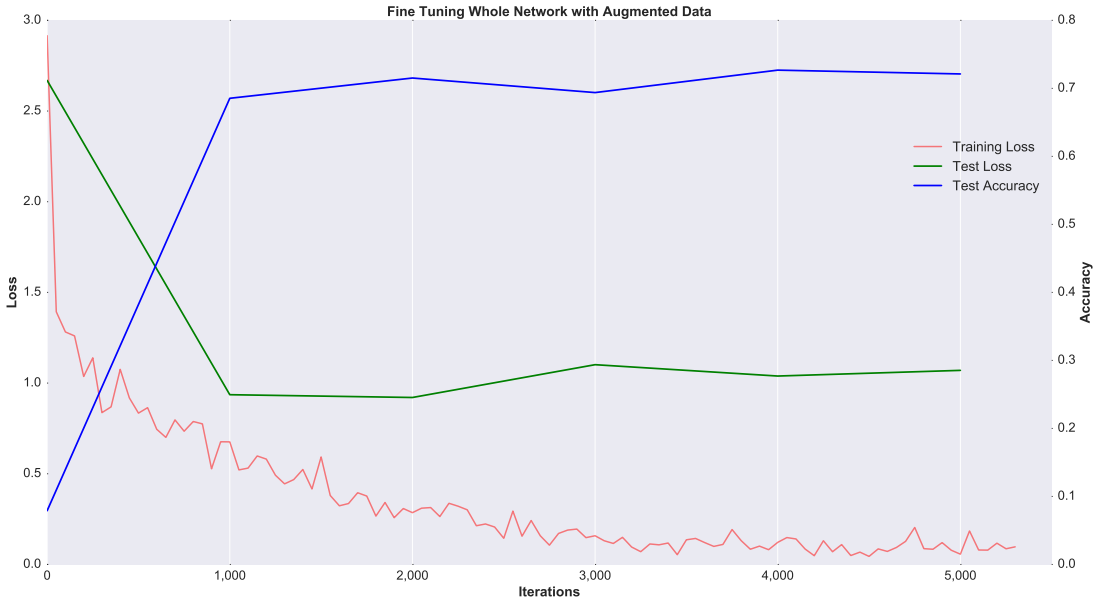
3. Training Parameters

The chosen training parameters were in general based on the default parameters provided with the pre-trained network.

Base Learning Rate	0.001
Learning Rate Update Policy	Step, with stepsize = 3,000
Momentum	0.9
Weight Decay	0.0005
Batch Size	150
Dropout	0.5

The training time was initially set to 30,000 iterations, but training was terminated early once the network appeared to have converged (see figure 6).

Figure 6: Training curve for fine tuning of Alexnet.



The overall test results were:

Precision	Recall	F1
0.71	**0.70**	**0.71**

As a further step, the same deterministic data augmentation strategy was applied to each of the test images, and the predictions for each augmented image were averaged to give an overall prediction. This gave an approximately 2% improvement in the results, as can be seen in the table 6.

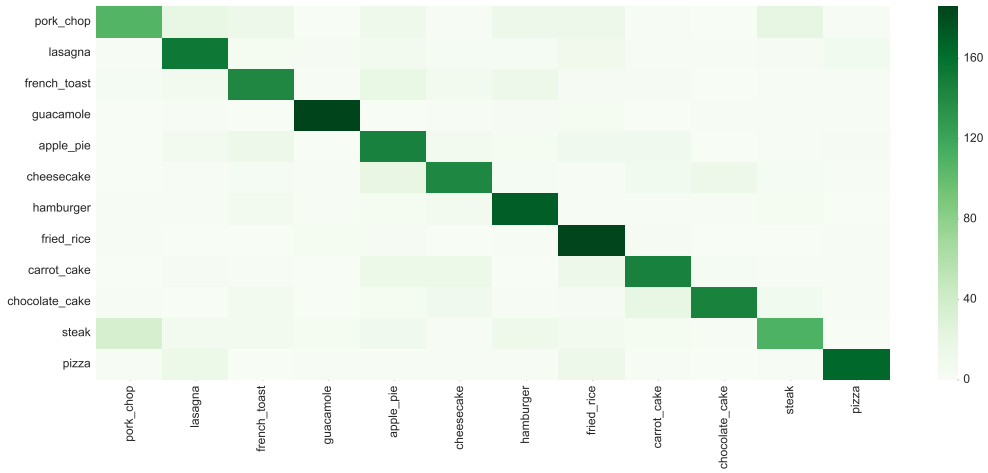
Looking at the new confusion matrix for our best CNN-based model, and we see that in general the per-class results have improved considerably (see figure 7.

Finally, we examine the Precision-Recall curves for this classifier for each of the individual categories

Table 6: Overall results from best fine tuning approach

Class	Precision	Recall	F1	Support
Pork Chop	0.70	0.54	0.61	200
Lasagna	0.73	0.76	0.74	200
French Toast	0.72	0.71	0.72	200
Guacamole	0.90	0.93	0.92	200
Apple Pie	0.61	0.74	0.67	200
Cheesecake	0.73	0.72	0.72	200
Hamburger	0.76	0.86	0.80	200
Fried Rice	0.71	0.92	0.80	200
Carrot Cake	0.74	0.74	0.74	200
Chocolate Cake	0.86	0.73	0.79	200
Steak	0.70	0.55	0.62	200
Pizza	0.89	0.82	0.86	200
Overall	**0.76**	**0.75**	**0.75**	-

Figure 7: Confusion matrix for best overall fine-tuned CNN.



(figure 8). We can see that, for nearly all the categories, it would be theoretically possible to achieve near-perfect Precision or Recall through a suitable choice of threshold. However what interests us in the context of our problem is a balance between the two, with both being as close to one as possible.

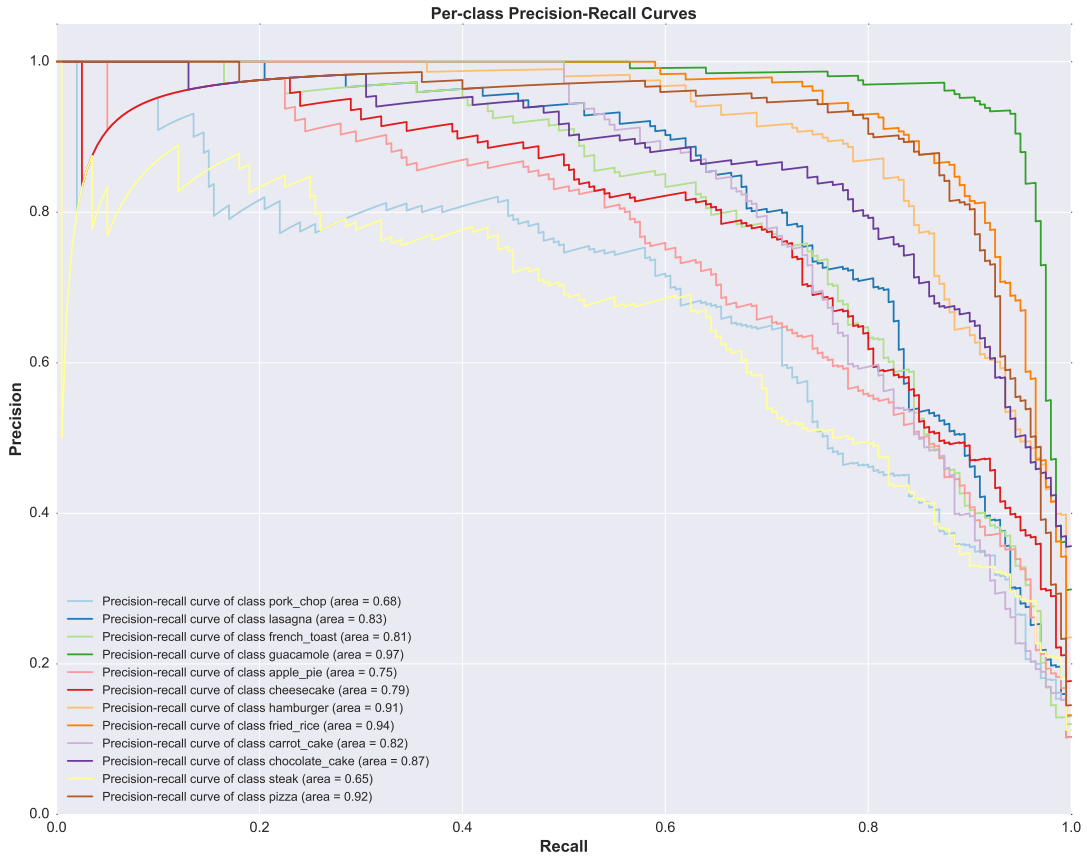
From the curves it is easy to see that the category with the best performance is Guacamole (dark green) with a curve that gets very close to the top right-hand corner. At the same time Steak (yellow) is the category with the worst results, with the best balance between precision and recall at around 0.6.

Additional Optimization

Although we were able to make significant improvements using pre-trained CNNs, we do not believe that our results represent the best possible classifier we could train. In fact, with additional time and resources, it should be possible to continue to improve on the overall and per-class scores by:

1. Additional fine-tuning of the network hyper-parameters (e.g., dropout rate)

Figure 8: Precision Recall curves for the different food classes.



2. Increasing the training batch size
3. Generate additional data by expanding the augmentation techniques used
4. Fine-tuning more recent and better-performing models such as VGGNet or GoogleNet
5. Training a number of models on the data, and aggregating the predictions of each model

Through a combination of these strategies, it should be possible to achieve an F1-score of 0.80 or higher.

5 Fail Cases

It can be illustrative to look at fail cases to better understand the weak points of our model.

In our case, even with our best model we see that Steak and Pork chop continue to present challenges:

- Steak has precision 0.70 but recall 0.55

- Similarly, Pork Chop has precision 0.70 but recall of 0.54
- Out of 200 test images:
  - 21 Steak pictures are classified as Pork Chops
  - 36 Pork Chop pictures are classified as Steak

However, looking at examples of errors, some of these mis-classifications are understandable:

Figure 9: Examples of fail cases confusing Steak and Pork Chops categories.

Another area where the model still has some issues, although to a slightly lesser extent, is in distinguishing between different types of cake, for example confusing chocolate cake with cheesecake.

Should it not be possible to achieve sufficient single-class accuracy through further optimization, one initial approach could be to consolidate some categories into overarching super-categories, for example consolidating all types of cake together into a single Cake category, and combining Steak and Pork Chops into a Meat category. Tests using one such approach resulted in an immediate improvement for F1-score from 0.75 to 0.85.

6 Comparison of Results

Overall, Deep Learning with Convolutional Neural Networks was clearly the better approach, resulting in far superior test outcomes. It was also, a little surprisingly, in some ways a much easier method too, as it avoided the need to manually choose and test different combinations of features.

However, one drawback of CNNs is the amount of computing resources and data required to train a successful model. Some of these difficulties can be mitigated by using Transfer Learning techniques on pre-trained models, and we saw that within a short space of time we were able to significantly improve our results using these methods. However even the small amount of CNN training we carried out would have been too difficult to perform on a standard laptop, and instead we had to resort to

Table 7: Summary of Machine Learning & Deep Learning approaches to the project.

	Machine Learning	**Deep Learning and CNNs**
Classification Accuracy	Relatively poor (<40%)	Excellent (75%+)
Training Speed	Mostly quite fast (<1 hr)	Anything from 1hr for simple transfer learning to 3-weeks+ for training a full network.
Testing Speed	Fast (X per image)	Fast (X per image)
Ease of getting started	Easy: libraries mean time can be spent on feature selection rather than creating models from scratch	Hard: Requires additional tools and resources and more time spent setting-up a model
Resources Required	Low	Medium-High
Feature Selection	Greatest time investment is in selecting and testing different types of features.	Very easy; the model selects features during training.
Overall	**	*****

investing in using a GPU instance from Amazon Web Services, in total spending approximately $50 USD on all of our combined deep learning attempts.

The benefit of standard machine learning models is that it is very easy to begin training and testing models, even on a pretty basic computer, due to the availability of a number of fast and user-friendly standard libraries (e.g., Scikit learn).

An overall summary of the two different approaches based on the experience from this project is shown in table 7.

Whilst it is true that Deep Learning with CNNs requires more computing resources, the truth is that it is now cheaper and easier than ever to be able to access the required processing power through services like Amazon Web Services. During this project, a large part of the incurred cloud computing cost was due to lack of experience with the tools and techniques being used. Without this learning-curve, it should be possible to carry out a similar exercise in far less time.

Additionally, the significant improvement in test results clearly demonstrate that future efforts on this project would be best focused on continuing to optimize CNN-based methods.

This is not to disparage standard machine learning which continues to be a very powerful tool-set and a more than adequate solution for a number of problems. However, when it comes to computer vision, it is clear why such a strong emphasis is currently placed on Deep Learning and CNNs.

7 Conclusion and Recommendations

As a proof of concept, we believe that the results of this project demonstrate that it would be possible to implement image recognition in a recipe search and retrieval user flow.

Although the current best model in itself would not be adequate both due to the small number of categories and further opportunities for improving accuracy, by working with the entire Food-101 dataset and choosing the right categories it should be possible to train a classifier with a high enough accuracy to be effective.

Overall, our recommendations for next steps are:

- Expand the model to include all 100 food categories from the existing dataset.
- Seek to increase the number of images by looking for other sources of data.

- Invest more time in optimizing the model, for example using the suggestions from before
- Consider a pilot based on using a smaller set of 10-15 consolidated food categories.

References

[1] http://www.ft.com/cms/s/0/f609954c-1d46-11e6-a7bc-ee846770ec15.html

[2] http://www.epicurious.com/about/press-center

[3] http://www.business.com/social-media-marketing/food-photo-frenzy-inside-the-instagram-craze-and-travel-trend/

[4] http://press.allrecipes.com/

[5] http://research.microsoft.com/en-us/um/redmond/projects/menumatch/

[6] http://www.popsci.com/google-using-ai-count-calories-food-photos

[7] https://www.vision.ee.ethz.ch/datasets_extra/food-101/

[8] The Japan Reader *Imperial Japan 1800-1945* 1973: Random House, N.Y.

[9] http://www.robots.ox.ac.uk/ṽgg/research/very_deep/

[10] Deep Residual Learning for Image Recognition, arXiv:1512.03385

[11] http://cs231n.github.io/transfer-learning/

[12] https://github.com/BVLC/caffe/tree/master/models/bvlc_alexnet

[13] Very Deep Convolutional Networks for Large-Scale Image Recognition, K. Simonyan, A. Zisserman, arXiv:1409.1556