## FINDING THE BEST FOOD IMAGE DETECTION SYSTEM

## ABSTRACT

With the improvement of living standards, people tend to put more attention on foods. There are more and more situations, for example a website for food sharing, that need to detect a image is food or not. In this report, we tried to use several well known machine learning algorithms as Single Layer Perceptron , Multiple Layer Feed Forward Perceptron , Support Vector Machine and Convolutional Neural Network with other techniques to solve the problem and to figure out which algorithm has the best effect and what contributes to it. Besides, we use SIFT to extract significant features in those images to promote the accuracy of some of our algorithms. The result shows that the Convolutional Neural Network with dropout turns out to be the algorithm having highest accuracy at xx% and Multi-Layer Feed Forward Perceptron be the second at xx%. Feature detection technique we used did no help. For further study, we can add some state of art techniques to improve Convolutional Neural Network and we can look deeper into not only judging whether an image contains food or not but also what kind of the food is.

**Keyword:** Food image detection, SLP, MLP, SVM, CNN, SIFT

## 1 INTRODUCTION

With the improvement of living standards, people begin to put more attention on delicacy and health care on foods. People like sharing their life these days by uploading photos they take in websites, yummy foods is one of the most popular part. Of course, these websites prefer user upload the images exactly belong to their categories. Or in other case where people want an automatic analyzing of how healthy they eat, grabbing photo records of foods will be necessary. In all these cases, we need a technique to tell whether a certain image contains foods or not.

With this requirement, building a machine learning model to predict is what we can do. So another question can be drawn here that which algorithm in the machine learning area is most effective and what makes it.

Since there are so many algorithms now can be used to do the task, we picked several well known ones as Single Layer Perceptron (SLP)[], Multiple Layer Feed Forward Perceptron (MLP)[], Support Vector Machine (SVM)[] and Convolutional Neural Network (CNN)[]. We not only used almost the basic setting for those methods to do the comparison, we went further to add pre-training feature detection or dropout in certain learning algorithms to improve the prediction accuracy and to find out which adjustment will bring improvement in this case. By running all these algorithms, we get an order of them ranked by test accuracy as …., which tells us that...

## 2 DATASET

We combined UECFOOD100, STL-10 and CVPR-09 for training and testing in this paper. The images are of different sizes, we used a thumbnail function to resize and crop them into 96x96 pixels. Namely, we rescale the image so that its width or height, whichever is smaller, into 96 pixels, and crop out the centered 96x96 square. This is the way the Photo app in Mac generates thumbnails, and it almost always preserves foods in the images according to our experience.

3.1 Food Image

UECFOOD100 contains 100 categories of food images, about 14000 images in total. We took 6/7 images out of each category and put them into our training dataset as positive data.

3.2 Non-food Image

We have 12000 non-food images in the training dataset. To avoid bias from lighting condition, we decide to select 7000 indoor scenes and 5000 images of outdoor or animals.

3.2.1 Outdoor Image

STL-10 has 5000 training images, which are outdoor views or animals. We took all of them.

3.2.2 Indoor Image

CVPR-09 has 67 indoor categories and 15620 images. We removed 13 categories such as bar, buffet, bakery and etc, which may contain foods. From the rest we randomly chose 7000 images and use them as negative data.

## 3 METHODS

(If you are using previously developed algorithms, describe them briefly, and provide references to complete descriptions.)

3.1 Basic Neural Network

3.1.1 Single Layer Perceptron

In order to figure out what helps to get better prediction accuracy, we decided to start from the basic neural network, the single layer perceptron. Single layer perceptron is a kind of neural network that only has two layers as input layer and output layer. Two layers are fully connected.

We used the pixels of images with all three red, green, blue channels as inputs. For output layer, in order to figure out what help improving the accuracy, we decided to adopt softmax[] in SLP as the same as in CNN which is more effective with two output units. Here, we use backpropagation algorithm[] to adjust the weight.

3.1.2 Multi-Layer Feed Forward Perceptron

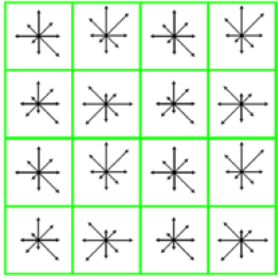
After we test SLP, we add more hidden layers, which use sigmoid function[] as activation function, to the model to form a Multi-Layer Feed Forward Perceptron, still using backpropagation algorithm to optimize the network. Here, we add three hidden layers as we test the CNN.

Multi-Layer Feed Forward Perceptron is a fully connected neural network that has no recurrent structure, which means we add up all outputs of lower layer and transfer it the higher layer. The result could be very large, so we set a column norm to hidden layers, the norm for the add up of all lower layer nodes to one higher node, to prevent overfitting of the network. What’s more, we also add momentum[] to avoid local minima and to speed up convergence as the same as CNN we will discuss later.

We also used the pixels of images with red, green, blue channels as inputs for MLP and the same output layer that contains two softmax nodes at the first test of MLP. Hoping to get better performance and compare to the SVM, we also used detected features as inputs to run the MLP. Since the vector of the keys of detected features are much smaller than original image matrix, we only use two hidden layers this time.

3.2 Feature Detection

We used SIFT (Scale Invariant Feature Transformation) to extract features from the images. SIFT is a widely used method for object recognition. To detect the same object recognition, For a given image, the algorithm goes through four steps to find a collection of key points, and produce a list of properties for each key point. These key points are invariants under rescale, rotation and change of illumination, which are the usual transformations . The output can be considered as a table. Each row of the table corresponds to one identified key point. The first two columns stores the location of the key point. The third column is the scale of the key point, meaning that on which scale this point is interesting. It corresponds to the size of the yellow circle in Figure x. The fourth column is called orientation



3.3 SVM

After we have vectors extracted from the images, we can perform feature based machining learning methods. We start with Support Vector Machine (SVM), one of the most widely used method for feature based supervised learning. SVM addresses binary classification by constructing a hyperplane in the vector space to separate vectors stand for different classes. A good separation is that can maximize the distance to the closest vector in both classes (margin). This optimizes the model in the sense of generalizing the observed data. However, it is not always the case that the data is linearly separable. Two approaches can be used to address the issue. One is to use the soft-margin, which trade-off with maximizing the margin and the violation of some vectors instead of forbidding any of the vectors in the margin. Another one is to transform vectors into higher- or infinite- dimensional spaces so that they are easy to separate, which involves so called kernel functions to compute the inner product of converted vectors.

We use the implementation from Python3 package sklearn[] .

3.4 Convolutional Neural Network (ConvNet)

A convolutional neural network is a multi layer neural network that uses 3 additional ideas: *local receptive fields*, *shared weights and biases*, and *pooling.*

**Local receptive fields**: In a regular MLP, each hidden unit is connected to every neuron in the previous layer. However in a ConvNet, we only make connections in small, localized regions of the previous layer.

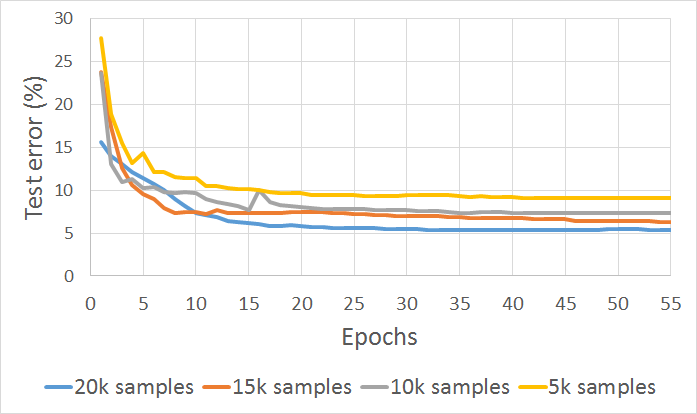
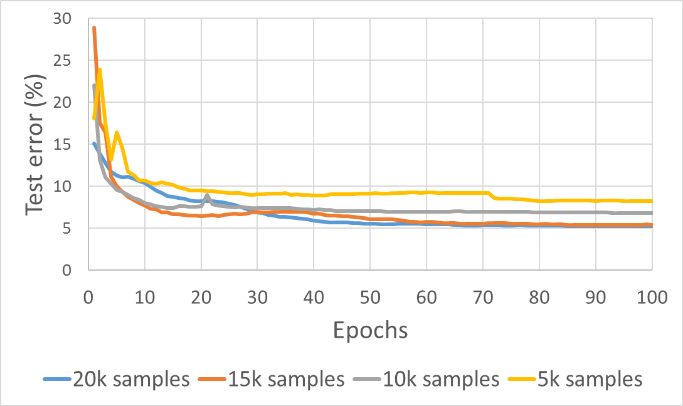
**Shared weights and biases**: Each hidden unit in a layer in a CNN shares the weights and biases (of its local receptive field) with every other neuron in the same layer.

**Pooling**: A pooling layer in a CNN takes each feature map output from the convolutional layer and prepares a condensed feature map.

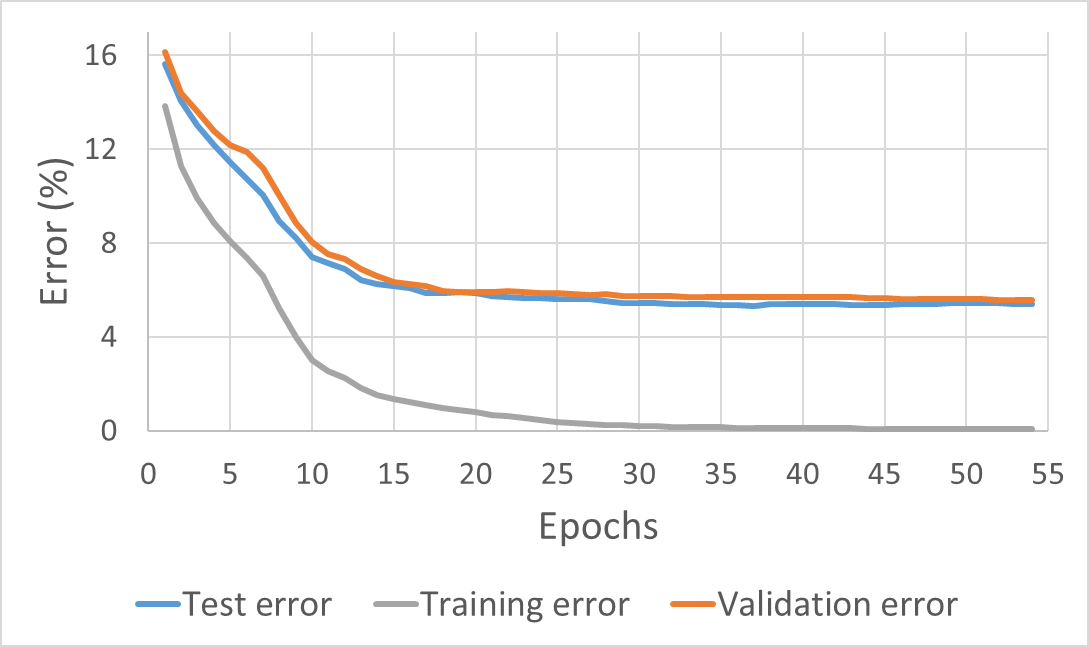
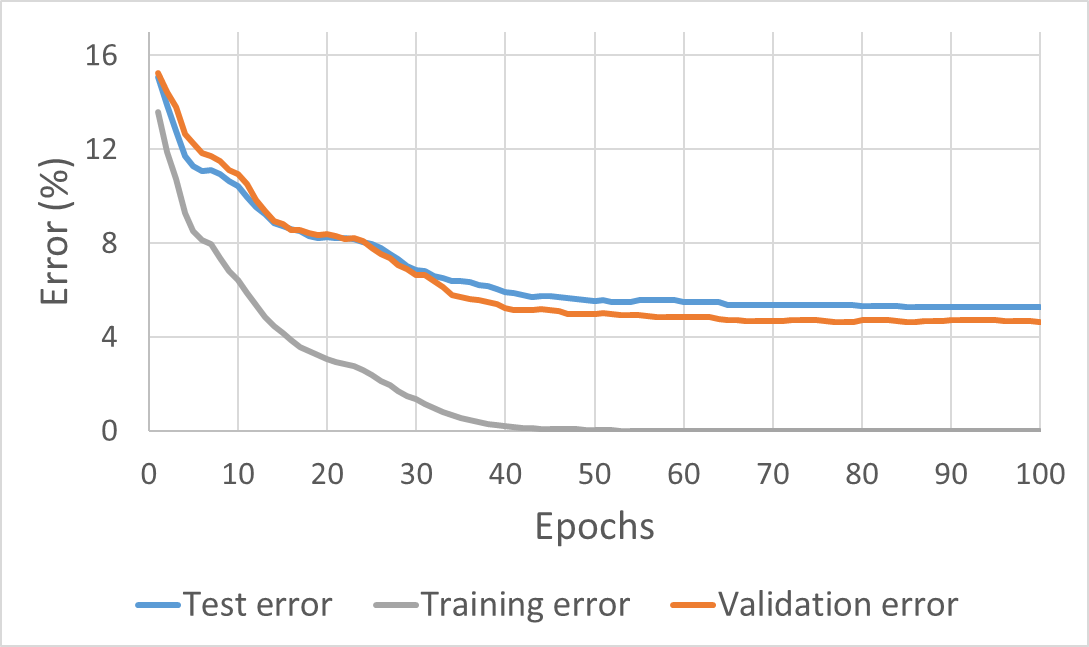
## 4 Experiments

## 5 RESULT

(Describe how you chose settings for parameters of the algorithms? Clearly state what are you trying to test/demonstrate in your experiments.)

|  |  |
| --- | --- |
| Fig : Test error vs epochs for 2 layer CNN | Fig : Test error vs epochs for 3 layer CNN |

|  |  |
| --- | --- |
| Fig : Error metrics vs epochs for 2 layer CNN with  20k samples | Fig : Error metrics vs epochs for 3 layer CNN with 20k samples |

|  |  |
| --- | --- |
| Fig : Test error vs epochs for 3 layer CNN with dropout | Fig : Test error vs epochs for 3 layer CNN |

|  |  |
| --- | --- |
| Fig : Test error vs epochs for 2 layer Fully connected MLP | Fig : Test error vs epochs for 3 layer Fully connected MLP |

## 6 DISCUSSION

## 6 CONCLUSION

In this report, we tried 4 well known machine learning algorithms to do the food detection task. We also added some techniques like momentum, normalization, feature detection and dropout to some of our approaches. We found...

We hope that this study can serve as a springboard for further study adding some state of art techniques to Convolutional Neural Network to get better performance or categorizing the foods in the images directly.

## REFERENCES