Deep Learning Assignment 2

Yun-shao Sung* and Chung-Ling Yao†

Abstract. This is the report for deep learning assignment 2

1. More Backpropagation.

1.1. Backpropagation through a DAG of modules. Given each node is sigmoid layers, and the second layer is $O_{min} = min(i_1, i_2)$ and $O_{max} = max(i_1, i_2)$. Therefore, we can rewrite y as:

$$y = min(\frac{1}{1+e^{-x_1}}, \frac{1}{1+e^{-x_2}}) + max(\frac{1}{1+e^{-x_1}}, \frac{1}{1+e^{-x_2}}) \tag{1.1}$$

which can also rewrite as:

$$y = \frac{1}{1+e^{-x_1}} + \frac{1}{1+e^{-x_2}} \tag{1.2}$$

as we taking the deritive of E respect to x_i and with chain rule applied:

$$\frac{\partial E}{\partial x_i} = \frac{\partial E}{\partial y} \frac{e^{-x_i}}{(1+e^{x_i})^2} \tag{1.3}$$

1.2. Batch Normalization.

2. STL-10: semi-supervised image recognition.

2.1. Method: Surrogate Class.

2.2. Method: Kmeans Centroids. We implemented Kmeans methods as one of our model, since based on previous paper that they can reach very good performance based on this simple model. The idea can divided into two part: first for getting centroids, and the second is perform feature mapping and perform classification. Regarding to the centroids identification, we did the following three steps:
1. Randomly select certain amount of unlabeled training images, and extract random patches. The size of patch is 22x22
2. Apply pre-processing to patches, including normalization and whitening
3. Learn feature-mapping using unsupervised method, and here we use kmeans due to the its good performance in the reference paper After centroids are identified, the steps for the second stage is as followed:
1. Extract patches from input images. The size of patches is 22x22, and the gap between patches is 2, and therefore we can get 16 patches from each of imput figure. Then we perforem feature mapping as the following equation:

$$f_k(x) = max\{0, \mu(z) - z_k\} \tag{2.1}$$

*yss265@nyu.edu
†cly264@nyu.edu

where $z_k = \|x - c^{(k)}\|$ and $\mu(z)$ is the mean of the elements of z
2. Pool features together over region and perform 4 quadrants summation to reduce the number of feature values, and therefore we will get the feature in the size of $4k$ per figure, and k is the number of centroids.
3. Perform classification based on the feature vector, and here we used 1-vs-N SVM.
Figure 3.2 showes the process of the centroids identification part, and figure 2.1 shows the feature mapping and classification part

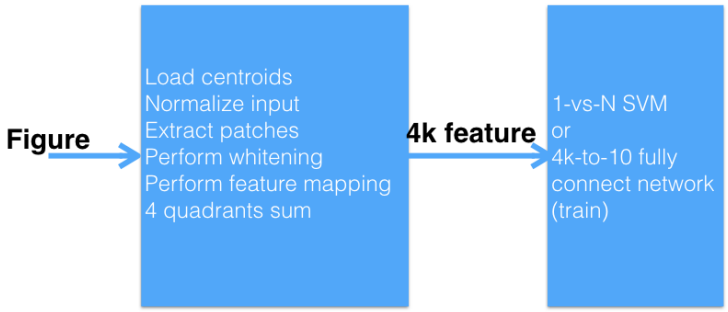

Figure 2.1. *Kmeans model*

2.3. Method: Pseudolabels. Hi

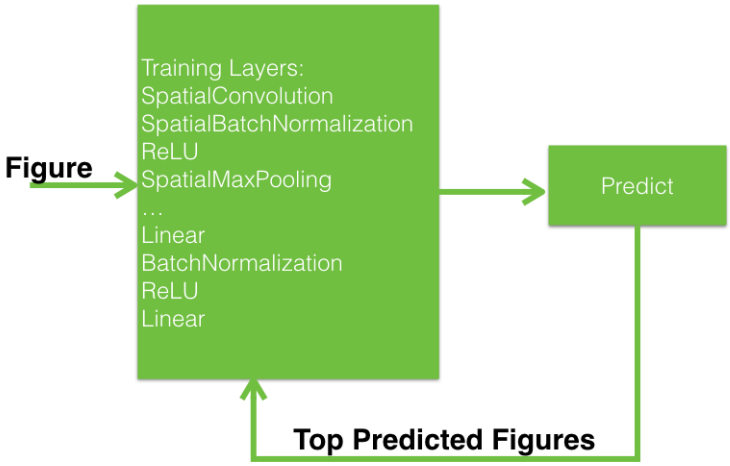

Figure 2.2. *Pseudolabels model*

2.4. Method: Kmeans Centroids + Pseudolabels.

3. Visualization.

3.1. Visualizing filters and augmentations. Initially we were also trying to creat surrogate data set from 4000 unlabel figures, each figure will produced 100 surrogate figures and

therefore we got 4000 class of 400000 figures in total. The initial figre is in the size of 3x96x96, and we created the surrogate figure by the size of 3x32x32 and each will subject in a random degree of rotation between -20 to 20, vertical and horizontal translate between 0 to 0.1, and scale in the range between 0.7 to 1.4. Figure 3.1 is the visualization of the surrogate figures.

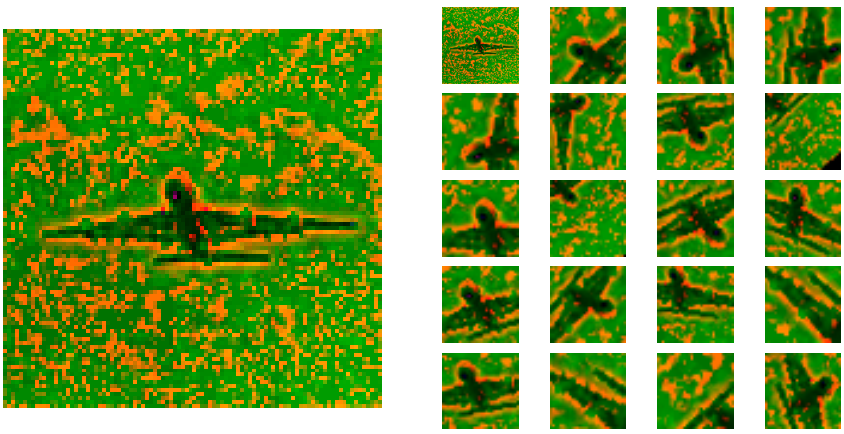

Figure 3.1. *Figures of surrogate set*

Then we implemented the kmeans method. According to the paper, they randomly extracted patches from unlabeled set of data, and performed kmean to find the centroids. The size of patch we extracted is 22x22 and we extracted 16 patches from total of 20000 unlabel figures. Number of centroids we obtained is 1600 because it produced good accuracy as the paper mentioned. As we can see from figure 3.2, most of the centroids are in very sharp of edge and color blobs.

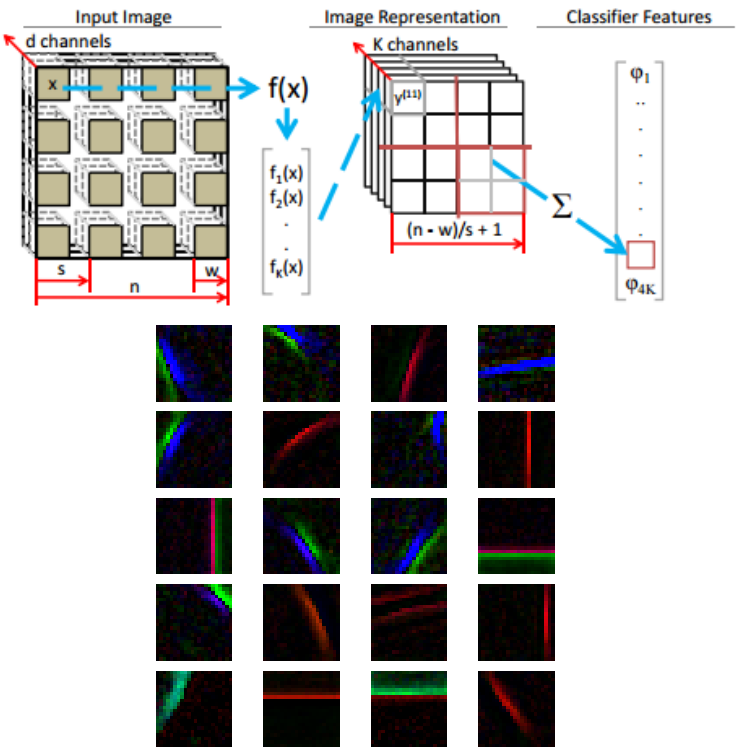

Figure 3.2. *Figure of centroids*

3.2. t-SNE. Here we took all the images from val.t7b, which contains 1000 figures for total and 100 figures in each of class. To generate t-SNE embedding, we used only the first channel of each of the image, which the dimension is 1x96x96, and feed it into manifold.embedding.tsne. Then we will get the mapping result for each of the image onto the 2D space, and we can plot the figures based on the mapped coordinate.

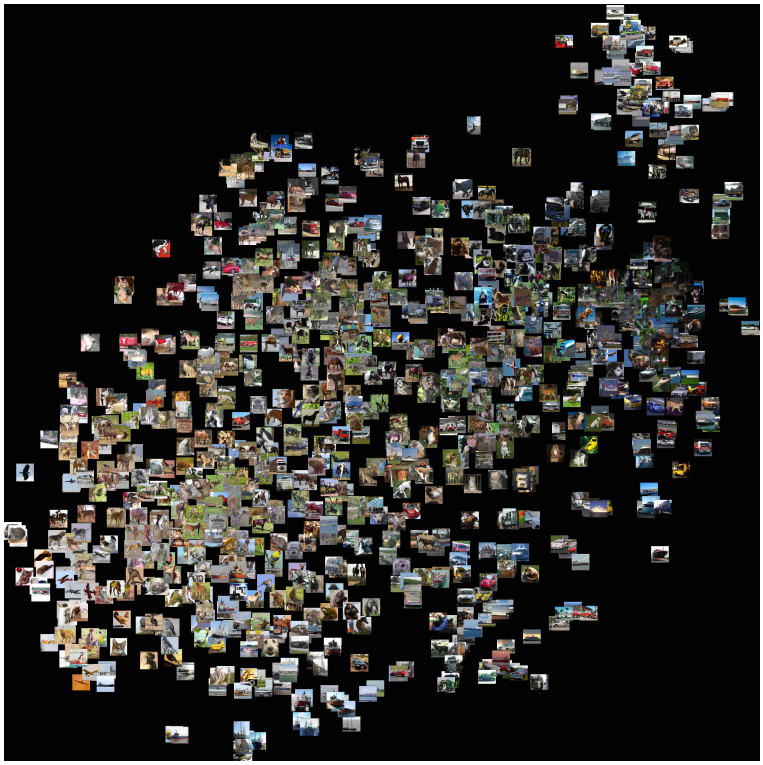

Figure 3.3. *The training and test accuracy of 3-layer versus 2-layer model*