Object Recognition in Images

Wenqing Yang{wenqing@stanford.edu}, Harvey Han{hanhs@stanford.edu}

***Abstract*—The purpose of this project is to build an object recognition system that can accurately classify images using CIFAR-10, a benchmark dataset in image recognition. We applied knowledge of machine learning to this computer vision system. Particularly, we investigated Softmax Regression, SVM and Convolutional Neural Networks to build this model, among which CNN generated the best result.**

I. Introduction

Object recognition is an important subfield in computer vision. It is easy for humans to recognize and classify objects in images, but usually not for machines. There are various obstacles in object recognition. For example, a picture only shows an object in 2D dimension but the angle of viewpoint can vary. There are also scale, color and illumination differences in a picture. Moreover, the intersection, deformation and intra-class variation in the objects themselves also make the problem difficult to handle.[1]

However, object recognition has made great progress out of machine learning techniques. Over the past few decades, a bunch of algorithms and methods have been created to solve the problem, among which deep learning theory especially Neural Network generates the best performance. Thanks to the advancement in machine learning area, recently object recognition has thrived in a variety of commercial areas such as Automatic Focus, MobilEye and Google Goggles. It will further provide more applications in industrial and medical fields including manufacturing quality control and medical imaging.[2]

Our project is from Kaggle competition and the dataset is publicly available. We focus object recognition particularly in color images.

II. Dataset

A. Raw Data

CIFAR-10 is publicly available online.[3] The dataset consists of 60,000 32x32 color images used for object recognition. We keep the split of train and test set in the official data. There are 50,000 images in the training set and 10,000 in the test set.

In addition, there are 10 object classes in total and one image belongs to a certain class. There are no intersections among the 10 classes. The label classes are namely airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. Both the training and test set are labeled for training and testing.

Notably, the dataset from Kaggle is different from the online offical dataset in [3] in that the organizers mixed some junk data in the test set in order to guarantee the justice of the competition. Since we will not submit the result on Kaggle, we use the original test set without junk data directly to provide an estimation of our prediction accuracy.

B. Data Preprocessing

CIFAR-10 gives us natural color images. The color information may not be very essential in some algorithms. However, if we apply the simplest way to process the data (averaging RGB to grayscale), we will definitely lose information. Moreover, in natural images, color may be related to different objects. For example, flowers tend to have bright warm colors while trucks have relatively cold colors. Hence, we determined not to convert the pixels to grayscale.[4]

In addition, the raw data can be redundant, because adjacent pixels have highly correlated RGB values. We want to remove this kind of redundancies while not losing useful information. ZCA whitening[5] provides a great method to preprocess the raw data. This is also a rough model of how biological eyes process the images. We obtained the ZCA-whitened data by following UFLDL tutorial implementation.

III. Experimental Methods

A. Softmax Regression

We chose logistic regression as our first attempt. Since there are 10 different classes in our dataset, we in turn applied softmax regression[6], which is multi-class version logistic regression, to classify and predict object labels. In order to gain a clear understanding of the algorithm, we wrote some codes with the assistance of UFLDL rather than call functions directly from Matlab. We maintained the color information and didn't convert the pixels to grayscale, expecting obtaining higher accuracy.

B. SVM

From the results of softmax regression, it turned out not to be satisfactory. We gained more insight of the data and then we turned to SVM. We used liblinear[7] package in Matlab to train and test our dataset, just as what we did in Problem Set 2. The only difference lied in the fact that we had 10 instead of 2 labels classes.

C. Naive Bayes

We tended to form a classification method with naive Bayes. Considering each pixel of the image has 3 integers to represent three colors and an integer containes 8 bits of information, we could not simply inspect the existence of each integer. Instead, we should consider each bit as a word in a dictionary since each pixel has 24 bits of information. The dictionary size would be as huge as 24x32x32 that equals to 24576. Moreover, there are ten classes in total, which means we need to calculate a series of probabilities. Thus, this method turns out to be cumbersome and is not suitable for object detection. Taken all these into consideration, we abandoned this idea of naive Bayes.

D. Convolutional Neural Network

The field of image recognition has made great progress since the application of the Convolutional Neural Network. LeNet, AlexNet and GoogleNet are examples of milestones in the development of CNN. There are various platforms to build and train CNNs conveniently such as Caffe, Theano, Cuda-Convnet2. We applied MatConvNet for self-defined CNN and C++ for Spatially-sparse CNN.

1) Self-defined CNN: We built up a 12 layer Convolutional Neural Network inspired by the examples in MatConvNet[8] and Caffe[9]. It is composed of layers of Convolution, Rectified Linear Units, Max and Average Pooling. There are also two Fully Connected Layers at the bottom. The architecture is shown in Fig. 1.

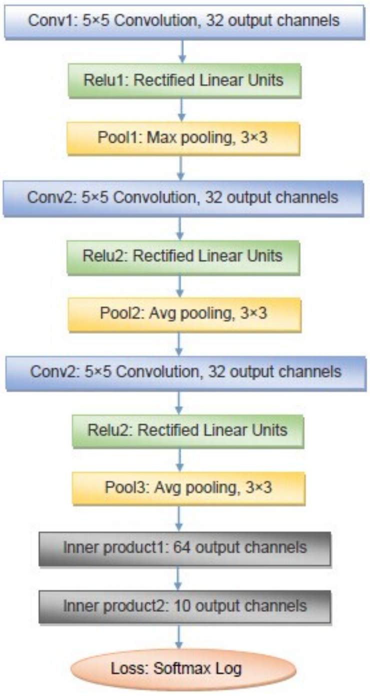
Conv1: 5×5 Convolution, 32 output channels
Relu1: Rectified Linear Units
Pool1: Max pooling, 3×3
Conv2: 5×5 Convolution, 32 output channels
Relu2: Rectified Linear Units
Pool2: Avg pooling, 3×3
Conv2: 5×5 Convolution, 32 output channels
Relu2: Rectified Linear Units
Pool3: Avg pooling, 3×3
Inner product1: 64 output channels
Inner product2: 10 output channels
Loss: Softmax Log

Fig. 1: Architecture of Self-defined CNN.

2) Spatially-sparse CNN: Although the CNN above dramatically improved the accuracy, we could still modify the architecture to further improve our neural network. Motivated by recent spatially-sparse CNN[10], we decided to implement the sparse network to lower the test error. Taking advantage of sparsity of input, we were able to train and test deep CNNs more efficiently. This method turned out to be the best algorithm for our project so far.

a) Sparsity of CIFAR-10 Images: Imagine we feed all-zero arrays into the input layer. As it goes through the hidden layers of CNN, it generates non-zero outputs. We consider receiving no meaningful information as a ground state and then will be able to utilize this sparsity. Handwriting characters with thin pen can be considered sparse; however, naturual images like CIFAR-10 images cannot be regarded as sparse. To make it spatially sparse, we need to add paddings to the surroundings of the images. This has a second advantage: data augmentation on training data can be carried out more easily by adding translations, rotations, or elastic distortions to the input images.[10]

b) Architecture: Sparse CNN takes the advantage of the good performance of sparse dataset on deep neural network, and the combination of multi-column CNN and network in network architecture. Multi-column deep neural network has alternating convolutional and max-pooling layers.[11] In addition to this, we add network in network structure[12] after each maxpooling. The network architecture is shown in Fig. 2.

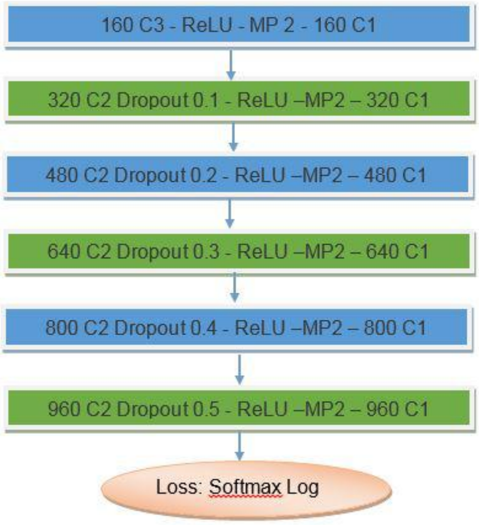
160 C3 - ReLU - MP 2 - 160 C1
320 C2 Dropout 0.1 - ReLU –MP2 – 320 C1
480 C2 Dropout 0.2 - ReLU –MP2 – 480 C1
640 C2 Dropout 0.3 - ReLU –MP2 – 640 C1
800 C2 Dropout 0.4 - ReLU –MP2 – 800 C1
960 C2 Dropout 0.5 - ReLU –MP2 – 960 C1
Loss: Softmax Log

Fig. 2: Architecture of Spatially-sparse CNN.

IV. Results

We made learning curves for Softmax Regression and SVM. See Fig. 3 and Fig. 4. The results of different experimental methods are listed in Tab. I showing both training and test accuracies of different models.

TABLE I: Accuracy Rate for Different Models

Experimental Methods	Training Set Accuracy	Test Set Accuracy
Softmax Regression	53.86%	33.55%
SVM	17.59%	27.62%
CNN	59.9%	58.6%
CNN (MatConvNet)	88.34%	77.64%
Spatially-sparse CNN	92.34%	91.61%

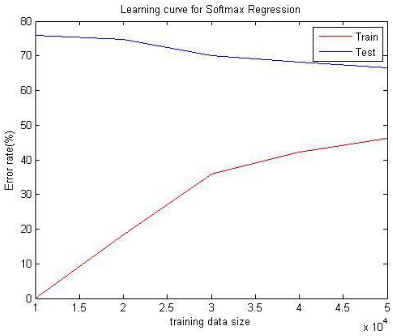

Fig. 3: Learning Curve for Softmax Regression

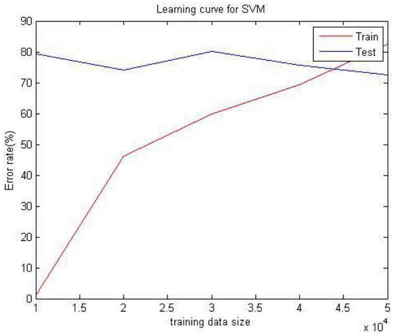

Fig. 4: Learning Curve for SVM

V. Comparison and Analysis

For the data preprocessing, we successfully implemented the ZCA whitening and mapped the raw data to whitened data. The dimensions remained the same and the redundancy was removed.

Softmax Regression and SVM with linear kernel both led to low accuracy. From the learning curve we can see the two models have high bias and overfitting. With the whole training set being 50,000 images, the accuracy will not exceed 50% in 10,000 test images. This means only with linear classifiers will not generate good performance in the object recognition. The results aroused our reflection and drove us to investigate Neural Networks especially Convolutional Neural Networks.

With a 12-layers CNN designed by ourselves, we see the dramatic improvement in accuracy. It is stable and the accurarcy rate is about 60% both in training and test dataset. It proves that CNN is well suited for image recognition. But further refinement can still be made to enhance the performance. There is one difficulty with our selection of CNN structures and layers that the model needs a long training time about 5 hours. So few tests could be made due to time limit. We further conducted literature review in the field and trained two well-designed CNNs. One is designed in MatConvNet, which emulates the layer structures from Caffe. The other is more complex and advanced state-of-art spatially-sparse CNN, which is developed recently by Benjamin Graham. Both of the two networks improved the accuracy rate greatly and the latter one had the accuracy rate of over 90%. This is a very high accuracy in the field of multiclass object recognition. It demonstrates the great potential of CNNs in image recognition.

VI. Future Work

There are several benchmark datasets in image recognition including MNIST, NIST, CIFAR-10, CIFAR-100, ImageNet etc. We will further explore more larger datasets with more categories. Also, we will turn to GPU to train our models, which is promising to greatly shorten the training time. Furthermore, there are a large pool of Convolutional Neural Networks as well as other Neural Networks and involved features can be added to both increase the accuracy and efficiency. We are to learn more varieties in designing Neural Networks.

VII. Conclusion

We first tried out Softmax Regression and SVM and gained a general view of the image recognition problem. It was non-linearly separable and suffered overfitting. Then we designed our own Convolutional Neural Networks and improved the performance greatly. We further implemented two well designed CNNs and obtained the high accuracy rate of 90%. We are fascinated by the great potential of Convolutional Neural Networks and we will dig more into the great variety of Neural Networks and train and test networks using GPU.

Acknowledgement

Thanks for Professor Andrew teaching this excellent course. Thanks for TA's suggestions on our project. Thanks for all those who gave a comment on our project at Poster Session!

References

[1] Wiki. Outline of Object Recognition. http://en.wikipedia.org/wiki/Outline_of_object_recognition.
[2] http://people.csail.mit.edu/klbouman/pw/papers_and_presentations/ObjectRecognitionDetection-11-25-12.pdf.
[3] http://www.cs.toronto.edu/~kriz/cifar.html
[4] https://www.kaggle.com/c/cifar-10/forums/t/7168/converting-the-images-to-grayscale.
[5] http://ufldl.stanford.edu/wiki/index.php/Whitening
[6] http://ufldl.stanford.edu/wiki/index.php/Softmax_Regression
[7] http://www.csie.ntu.edu.tw/~cjlin/liblinear/
[8] http://www.vlfeat.org/matconvnet/
[9] http://caffe.berkeleyvision.org/gathered/examples/cifar10.html
[10] B. Graham. Spatially-sparse convolutional neural networks. http://arxiv.org/pdf/1409.6070.pdf
[11] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural net-works for image classication. In Computer Vision and Pattern Recognition $CVPR$, 2012 IEEE Conference on, pages 3642{3649}, 2012.
[12] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. CoRR, abs/1312.4400, 2013.