Capstone Project

Ryosuke Honda

August 10, 2016

1 Definition

1.1 Project Overview

Student provides a high-level overview of the project in layman 's terms. Background information such as the problem domain, the project origin, and related data sets or input data is given.

The improvement on the camera and the widespread of larger capacity hardwares lead us to store more digital pictures. However computers can't understand the meaning of the pictures, so it is essential for us humans to classify or search images. Image recognition(Figure.1) will help us to classify or search pictures without our intervention. These days, because of the development of the computational capacity, we can process large number of pictures with high level accuracy(nearly the human's recognition). In this project, I'll discuss the image recognition algorithm which will be useful in the classification of large number of images.

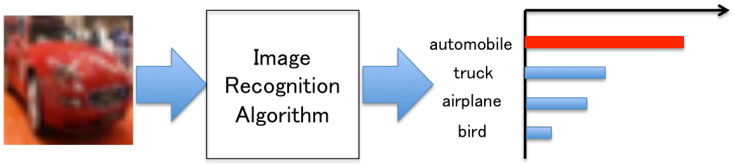

Figure 1: Image of the Algorithm

1.2 Problem Statement

The problem which needs to be solved is clearly defined. A strategy for solving the problem, including discussion of the expected solution, has been made.

The CIFAR-10 dataset

The CIFAR-10 dataset consists of 60,000 32×32×3 (width=height=32 and RGB=3) color images in 10 classes, with 6,000 images. 50,000 images are the training images and 10,000 images are the test images.

The classes are completely mutually exclusive. There is no overlap within each image class.

In this task, I'll discuss the image recognition by Convolutional Neural Network(CNN) and I optimize the hyper-parameters with Bayesian optimization.

1.3 Metrics

Metrics used to measure performance of a model or result are clearly defined. Metrics aper classre justified based on the characteristics of the problem.

The objective in this problem is multi class classification. Therefore,the metrics will be 'cross entropy'. The objective of the learning in this task is to minimize the cross entropy. Cross entropy is defined as the following formula[1].

$$C = -\sum_{j=1}^{n} d_j \log p_j \tag{1}$$

$d_1, ..., d_n$ is the optimal output(Correct output).$p_1, ..., p_n$ is the probability of for the output class. This probability is calculated by softmax function which is defined as below.

$$p_j = \frac{e^{u_j}}{\sum_{k=1}^{n} e^{u_k}} \tag{2}$$

Classification Error is calculated by the difference between the optimal output $d_1, ..., d_n$ and predicted output$p_1, ..., p_n$. The target output$d_1, ..., d_n$ takes the representation of 1-of-n. That means that only the correct label j becomes $d_j = 1$ and the others k($\neq$ j) become $d_k = 0$.

2 Analysis

2.1 Data Exploration

If a dataset is present, features and calculated statistics relevant to the problem have been reported and discussed, along with a sampling of the data. In lieu of a dataset, a thorough description of the input space or input data has been made. Abnormalities or characteristics about the data or input that need to be addressed have been identified.

CIFAR-10 dataset consists of 50,000 images of training data and 10,000 images of test data.For each data, it contains 10 different kind of classes(discussed in the next chapter) and the distribution of each class is the same in both training and test data. That means 5,000 images for each class in the training data and 1,000 images for each class in the test data.(Figure.3 and Figure.4).

2.2 Exploratory Visualization

A visualization has been provided that summarizes or extracts a relevant characteristic or feature about the dataset or input data with thorough discussion. Visual cues are clearly defined.

There are 60,000 of images in total. Figure.2 (50,000 images for training data and 10,000 images for test data) shows the samples of the images. I plotted 10 images for each class.Figure.3 and Figure.4 shows the distribution of the data. For the training dataset, each class has 5,000 images and for the test dataset, each class has 1,000 images. The label 0 to 9 corresponds to 'Airplane','Automobile','Bird','Cat','Deer','Dog','Frog','Horse','Ship, and 'Truck'.

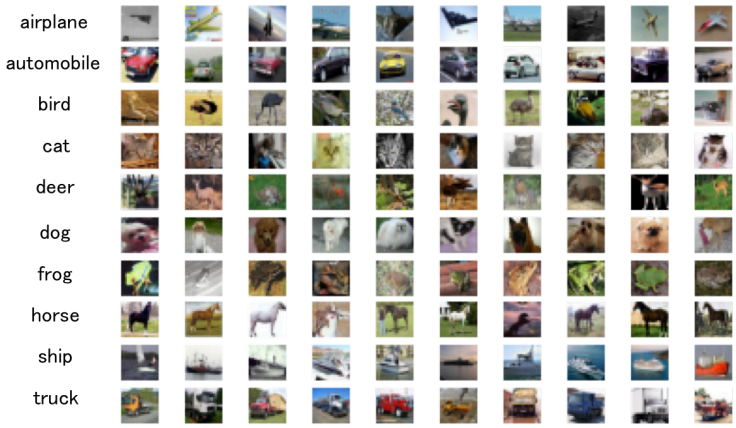

Figure 2: Sample of the Images

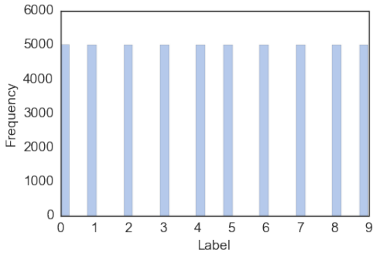

Figure 3: Distribution of Training Data

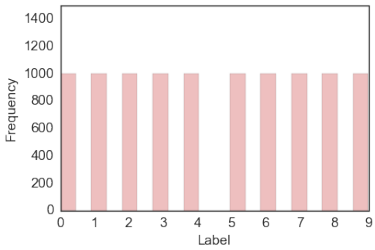

Figure 4: Distribution of Test Data

2.3 Algorithms and Techniques

Algorithms and techniques used in the project are thoroughly discussed and properly justified based on the characteristics of the problem.

In this task, I'll use Convolutional Neural Network(CNN). CNN has been successful in practical applications for image recognition.CNN consists of convolution layer, pooling layer and fully-connected layer and sometimes contains local contrast normalization(LCN). In this chapter, I'll discuss the convolution layer and pooling. As the name 'convolution' suggests, the network employs a mathematical operation called convolution.

Figure.5 is the example of the architecture of the CNN.

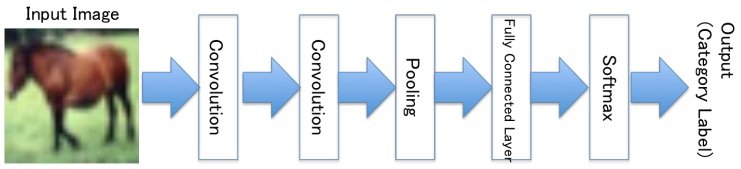

Figure 5: An example of CNN Architecture

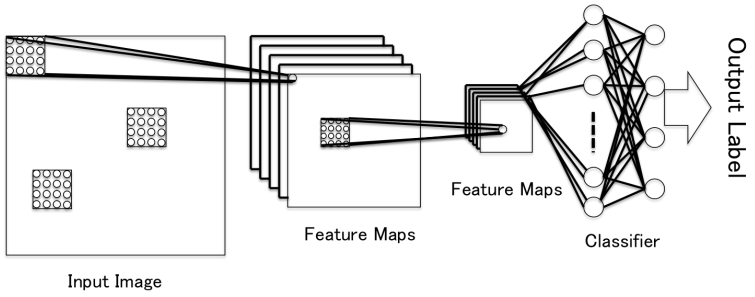

Figure 6: Example of Max Pooling and Average Pooling

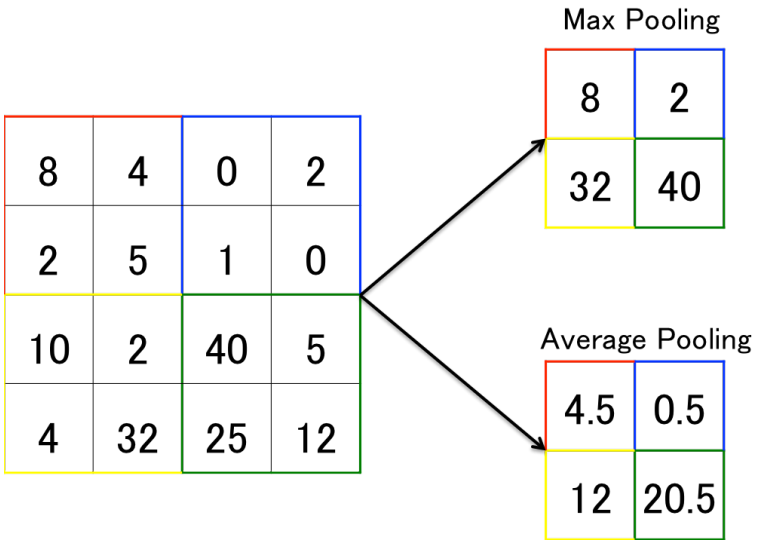

Figure 7: Example of Max Pooling and Average Pooling

Pooling layer is put after the convolution layer. The function of the pooling layer is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network, and hence to also control overfitting. By introducing pooling layer, not only the architecture will be more robust but also the dimensionality will be reduced. Max pooling and

Average pooling are the typical pooling which are generally utilized. Figure.7 is the example of the pooling.The original map is the size of 4×4. The stride for the pooling is 2 and the pooling size is 2×2. "Max pooling" is to extract the maximum pixel from each region and "Average pooling" is to calculate the average value for each region.In this task, I utilized max pooling at the pooling layers.

2.4 Benchmark

Student clearly defines a benchmark result or threshold for comparing performances of solutions obtained.

For the CNNs architecture, it is quite important to decide the number of layers.Therefore, I first choose several convolutional layers and decide one of them as a benchmark.

When deciding the architecture, I set the mutual parameters as follows. The number of batch size is 32.The number of filters of each convolutional layer is 32. and finally the number of epochs is 20.

I tried 4 architecture of CNNs.

1. 1 Convolutional layer & 2 Fully connected layers
   The simplest version in these model.The input and output dimension on each layer are below. The accuracy rate of training and validation data are also below.

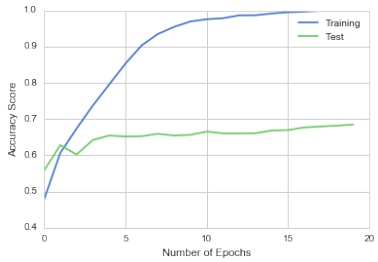

Figure 8: Accuracy rate of training and validation data

   The accuracy rate on the training data is quite high,however the accuracy rate on the validation data is low. This means that this architecture falls into over-fitting.

2. 2 Convolutional layers & 2 Fully connected layers
   The input and output dimension on each layer are below. The accuracy rate of training and validation data are also below.

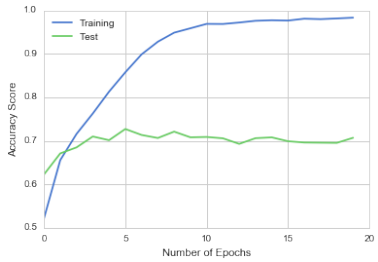

Figure 9: Accuracy rate of training and validation data

This architecture also caused over-fitting as mentioned above.

3. 3 Convolutional layers & 2 Fully connected layers
   The accuracy rate of training and validation data are also below.

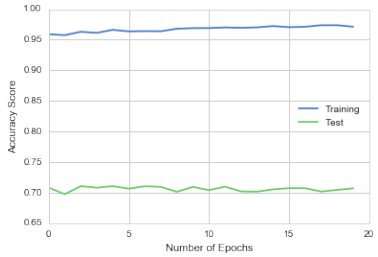

Figure 10: Accuracy rate of training and validation data

This architecture also falls into over-fitting.

4. 4 Convolutional layers & 2 Fully connected layers
   The accuracy rate of training and validation data are also below.

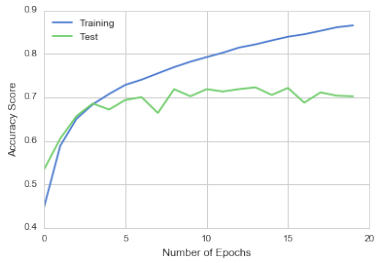

Figure 11: Accuracy rate of training and validation data

This architecture is far better than the others. The accuracy rate of both training and validation data is high as the number of epochs increases. The accuracy rate of this architecture is 70.2%. Actually the highest accuracy rate

in CIFAR-10 is more than 95.0% and the benchmark is quite lower than the highest one. My target in this task is to find the optimal parameters for the architecture. Therefore, I'll utilize this 4 layers convolution & 2 fully connected layers architecture as a benchmark.

From these results, I choose the fourth architecture(4 convolution layers & 2 fully connected layers) as the benchmark for this task.

3 Methodology

3.1 Data Preprocessing

All preprocessing steps have been clearly documented. Abnormalities or characteristics about the data or input that needed to be addressed have been corrected. If no data preprocessing is necessary, it has been clearly justified.

Apart from the models of image processing or computer vision, CNN doesn't need complex preprocessing.However,when analyzing the data, data preprocessing plays a crucial role. One of the first steps is the normalization of the data. This step is essential when dealing with parameters of different units and scales. In this dataset, pixel values range from 0 to 255. I process normalization to this dataset. Normalization scales all numeric variables in the range of [0,1].The formula is given below.

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{3}$$

The minimum pixel value is 0 and maximum pixel value is 255. Therefore, I normalized the data by following.

$$x_{new} = \frac{x}{255} \tag{4}$$

3.2 Implementation

The process for which metrics, algorithms, and techniques were implemented with the given datasets or input data has been thoroughly documented. Complications that occurred during the coding process are discussed.

As I introduced in the section2.4(Benchmark), I utilized the 4 convolutional layers and 2 fully connected layers.In this section, I explain the architecture of the model in more depth.

The objective of the learning is to minimize the cross entropy in the learning process. The architecture of the model is the following.

Layer	Batch	Stride	Map Size	Function
data	–	–	3 × 32 × 32	–
conv1	3 × 3	1	32 × 32 × 32	ReLu
pool1	2 × 2	2	32 × 16 × 16	–
conv2	3 × 3	1	32 × 16 × 16	ReLu
pool2	2 × 2	2	32 × 8 × 8	–
conv3	3 × 3	1	32 × 8 × 8	ReLu
pool3	2 × 2	2	32 × 4 × 4	–
conv4	3 × 3	1	32 × 4 × 4	ReLu
pool4	2 × 2	2	32 × 2 × 2	–
fc5	–	–	512	ReLu
fc6	–	–	10	Softmax

Figure 12: Architecture of the model

As for the optimizer, I utilized 'Adam'. With this setting, I got the accuracy rate of 70.2%

3.3 Refinement

The process of improving upon the algorithms and techniques used is clearly documented. Both the initial and final solutions are reported, along with intermediate solutions, if necessary.

Finding the optimal parameters for deep learning is quite difficult though it is important. When it comes to typical machine learning algorithm(Decision tree, Support Vector Machine etc.),grid search is taken to search the optimal parameters. However, it's almost impossible to apply grid search in deep learning because of the computational time.It has reported that the grid search strategies are inferior to random search.[6] Therefore, other methods are indispensable. A good choice is Bayesian optimization, which has been shown to outperform other state of the art global optimization algorithms on a number of challenging optimization benchmark functions.

Bayesian Optimization[4]

For continuous functions,Bayesian optimization typically works by assuming the unknown function was sampled from a Gaussian process and maintains a posterior distribution for this function as observations are made or, in our case, as the results of running learning algorithm experiments with different hyperparameters are observed.

The power of the Gaussian process to express a rich distribution on functions rests solely on the covariance function.The automatic relevance determination(ARD) squared exponential kernel is often used for Gaussian process regression.

$$K_{SE}(x, x') = \theta_0 \exp\left(-\frac{1}{2} r^2(x, x')\right) \tag{5}$$

$$r^2(x, x') = \sum_{d=1}^{D} (x_d - x'_d)^2 / \theta_d^2 \tag{6}$$

However, sample functions with this covariance function are unrealistically smooth for practical use. Therefore, I use ARD Matern 5/2 kernel.

$$K_{M52}(x, x') = \theta_0 (1 + \sqrt{5r^2(x, x')} + \frac{5}{3} r^2(x, x')) \exp\left(-\sqrt{5r^2(x, x')}\right) \tag{7}$$

I optimized the number of the dimensions of the first fully-connected layer in this task.

4 Result

4.1 Model Evaluation,Validation and Justification

The final model 's qualities — such as parameters — are evaluated in detail. Some type of analysis is used to validate the robustness of the model 's solution.

By utilizing the bayesian optimization, I optimized the number of dimensions of first layer of fully connected layer. I got the optimal number which is 417 and the test accuracy is 71.9% From the result of Bayesian Optimization, I chose —- as the final model parameters. The model improves the accuracy rate by 1.7%

The training and test accuracy in this model is as follows. The loss of the model is shown in Fig.—

The training and validation loss in the benchmark is Fig—. The counterpart in final result is Fig.—

The number of miss-classified data in benchmark is —-. On the other hand, the number in the final result is —-.The distribution difference in miss-classified data between the benchmark and the final result is following.

4.2 Justification

The final results are compared to the benchmark result or threshold with some type of statistical analysis. Justification is made as to whether the final model and solution is significant enough to have adequately solved the problem.

5 Conclusion

5.1 Free-Form Visualization

A visualization has been provided that emphasizes an important quality about the project with thorough discussion. Visual cues are clearly defined.

Some data are miss-classified in this task. I'll look into which label is the poorest result and I'll put some of the images which are miss-classified.

The distribution of the miss-classified images are Fig.—

5.2 Reflection

Student adequately summarizes the end-to-end problem solution and discusses one or two particular aspects of the project they found interesting or difficult.

In this project, I solved the image recognition problem. These days, deep neural networks surpass the other typical image recognition algorithms and even surpass human recognition. However, one of the problems of deep learning is tuning hyper-parameters. As is often said, the architecture of the neural network is black art.Putting more layers may tend to grasp the feature of the input data. However, because of the back-propagation process, the gradient will vanish or explode while calculating. Therefore, the deep neural networks don't always

reach the good result. What's more, the deep networks tend to take much more time than the shallower ones. Personally, I don't have any GPU environment so that trying deep network was quite tough.Therefore, I tried training the shallow networks with hyper-parameter tuning.

First, I decided the number of layers of convolutional layers and then I optimized the parameters with bayesian optimization which is one of the methods to tune hyper-parameters. I chose 4 layers of convolutional layers and 2 layers of fully connected layers for the benchmark, then I optimized the number of dimension of the first layer of the fully connected layer with bayesian optimization.

One of the difficulties of this task was parameter tuning. At first, I tried to tune by grid search which is quite popular in the machine learning field. However, I soon realized the method isn't reasonable because of the number of the parameters and the computational time. Thus I chose other hyper-parameter tuning method which is bayesian optimization.

5.3 Improvement

Discussion is made as to how one aspect of the implementation could be improved. Potential solutions resulting from these improvements are considered and compared/contrasted to the current solution.

In this task, I chose 4 layers of convolutions and 2 layers of fully connected layers. It is reported that the number of layers is the crucial factor in the high accuracy rate. For example, GoogLeNet,VGG,SPP are around 20 convolution layers. In 2015, ResNet which surpassed the human recognition had maximum 152 layers. In fact, deep networks can represent certain function classes far more efficiently than shallow ones. That means basically deeper networks can get the higher accuracy rate.

However, the stacking of several non-linear transformations in convolutional feed-forward network architectures typically result in poor propagation of activations and gradients. Therefore, training deep network is difficult. ResNet takes the strategy of skipping connections between layers. When the number of layers increases, finding optimal parameters becomes more and more important. Bayesian optimization can be one of the methods to find the optimal parameter.

References

[1] 岡谷貴之."機械学習プロフェッショナルシリーズ 深層学習", 講談社,2015

[2] 神嶌敏弘, 麻生英樹, 安田宗樹, 前田新一, 岡野原大輔, 岡谷貴之, 久保陽太郎, ボレガラ ダヌシカ, 深層学習, 近代科学社,2015

[3] 浅川伸一,Python で体験する深層学習–Caffe,Theano,Chainer,TensorFlow–, コロナ社,2016

[4] Jasper Snoek,Hugo Larochelle,Ryan P.Adams,Practical Bayesian Optimization of Machine Learning Algorithms

[5] Diederik Kingma and Jimmy Ba. Adam: a method for stochastic optimization, In the 3rd International Conference for Learning Representations ICLR 2015,2015

[6] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13:281-305,2012