CIFAR-10 Recognition

Author: Yihang Zhou

Email:zhouyiha@usc.edu

1 Abstract and Motivation

Now pattern recognition is famous in our life. It is used in many different areas, such like face recognition, unmanned supermarket and some other areas. Convolutional Neural Network (CNN) is feedforward neural network which is inspired by Receptive Field. In object recognition, action recognition, image recognition and other fields, it is widely used.

2 Approach and Procedures

2.1 CIFAR-10 Classification

The principle of CNN is using four type layers to extract features from images and build connection to train images. Then use these connections to justify unknow images. The layers are convolution layer, max-pooling layer, fully connected layer and SoftMax layer. I will explain more details in discussion.

I use online source PYRORCH to solve this problem.

First, I build a net structure and load data.

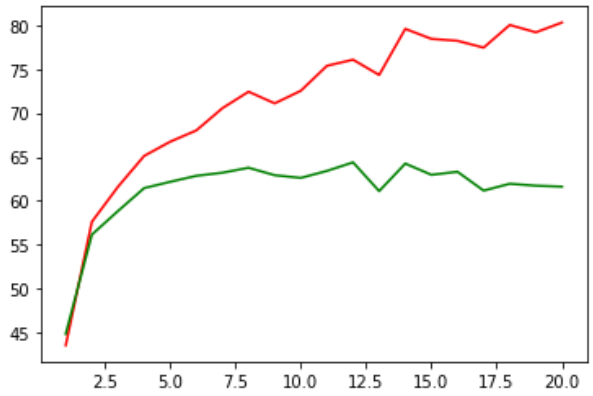
Then I use train data to get optimize weights.

Finally, Using optimize weights to the test data to calculate the accuracy. I adjust different parameter to see the influence of these parameter.

3 Result

Red: training data Green: testing data

X label: epoch Y label: Accuracy

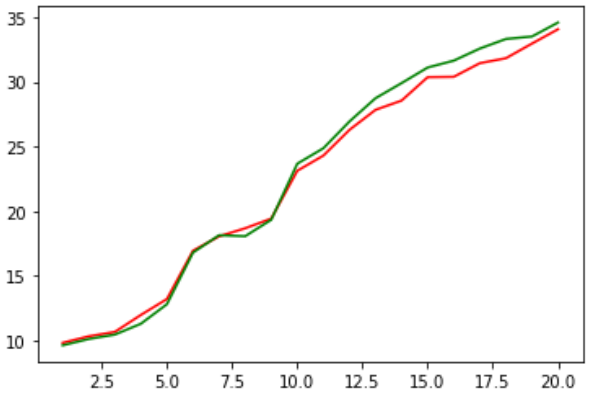


1 Learning rate=0.01 Momentum=0.92

Weight-decay=0 Batch size=64 Epoch=20

Highest Accuracy of test :63.76

Highest Accuracy of train :80.28

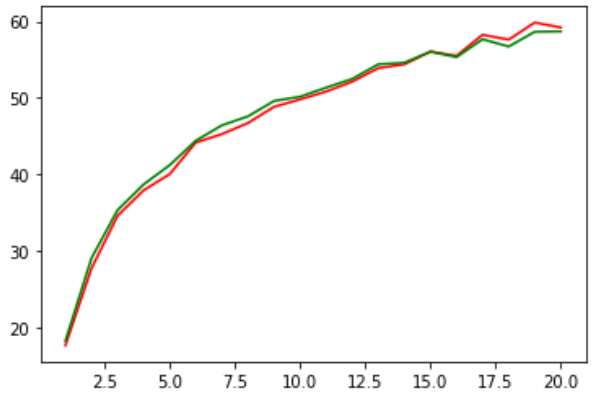


2 Learning rate=0.001 Momentum=0.3

Weight-decay=0 Batch size=64 Epoch=20

Highest Accuracy of test :34.60

Highest Accuracy of train :34.08

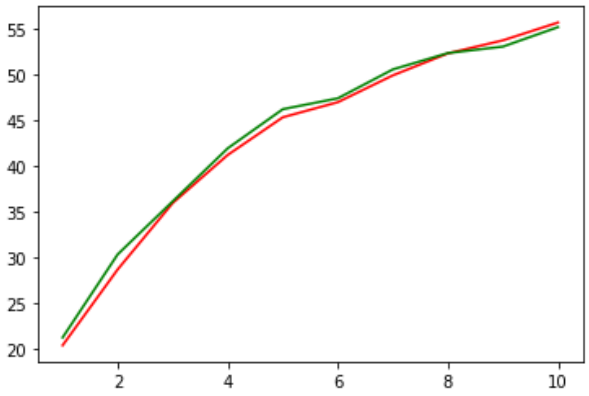


3 Learning rate=0.001 Momentum=0.92

Weight-decay=0.01 Batch size=64 Epoch=20

Highest Accuracy of test :58.70

Highest Accuracy of train :59.20

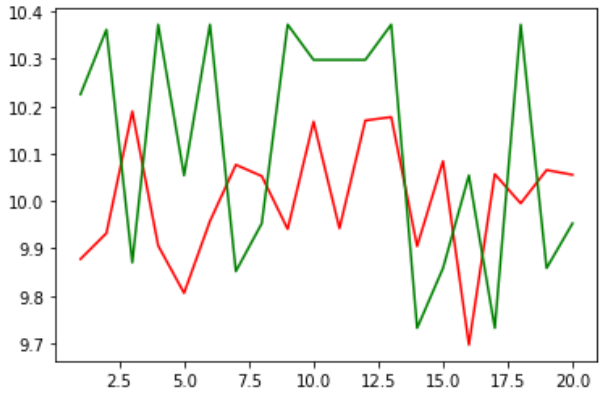


4 Learning rate=0.001 Momentum=0.92

Weight-decay=0 Batch size=64 Epoch=10

Highest Accuracy of test :55.13

Highest Accuracy of train :55.63



5 Learning rate=0.01 Momentum=0.92

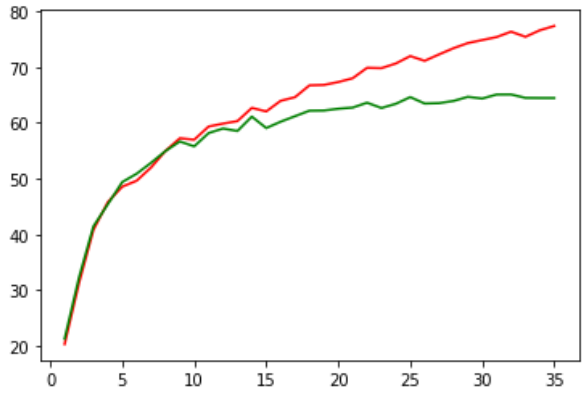
Weight-decay=0 Batch size=4 Epoch=20

Highest Accuracy of test :10.38

Highest Accuracy of train :10.18

Figure 1

Final Result



5 Learning rate=0.0005 Momentum=0.97

Weight-decay=0 Batch size=64 Epoch=35

Highest Accuracy of test :65.09

Highest Accuracy of train :77.39

Figure 2

4 Discussion

4.1 Structure comprehension

The fully connected layer is to connected features before and classifies these features. The method is use convolution kernels to shift above filtered image to 1 value which get a n*1 vector. The connection combines with weights and bias. The reason is to reduce feature location influence on classification. It can integrate feature representation. Usually, there are two fully connected layer because it can solve non-linear classification.

The convolutional layer is a feature layer. Usually, we use some convolution kernels to extract local features from an image to get convolution layer. Sometimes there will be more than one convolution progress which purpose is to extract deeper features. Kernels will be updated by backpropagation.

The max pooling layer is to reduce features dimension which can reduce training parameters and times. It can also reduce overfitting. The method is choosing the maximum value in a 2 by 2 window.

The purpose of activation function is to transform linear mapping to non-linear mapping which is especially usefully when data is nor linear separable. Adding an activation function can make model classify complex data more correctly. Some activation function is sigmoid, Tanh, ReLu and so on.

The SoftMax function is the final step to transfer the non-normalized data to a probability distribution as output classes' probability. It makes large probability class usually become output, but sometimes low probability can also be output.

4.2 Overfitting

Overfitting issue in model learning is classification to training data is too precise. Sometimes some noisy data or not typical data are considered in weights chosen. If we use these weights in test data, the accuracy will not high.

1. Weight-decay can make weights become small which avoid complex discriminant functions. 2. Max-pooling can reduce the number of parameters which also to avoid overfitting. 3. If we train the training data too much time, it will lead overfitting, so, choosing proper epoch times is also a good way. 4. Data augmentation. Using some methods like data modification to generate more data is also a good method.

4.3 Compare with traditional computer vision

In image classification, CNN imitates human vision. The structure is convolution, max pooling, fully connected and SoftMax layer. 1. Following this structure, it can extract image features more efficiently comparing than traditional computer vision. The reason is it is data driven which can update weights by backpropagation and some other ways. So, it modifies weight by computer itself. As for traditional computer vision, human need to adjust weights by themselves which cost a lot of time and the result is not as good as machine learning. 2. Because of CNN's character, it can use more data to train which can help it get more accurate parameter for test data. 3. It also uses max pooling and weight decay to avoid overfitting and reduce computation load. It uses fully connected and SoftMax to give a more reasonable and efficient classify to images which accuracy is higher than traditional computer vision.

4.4 Loss function and backpropagation

Loss function is used to measure the difference between real value and predicted value. We want the loss cost is small which mean the output result is what we expect.

Backpropagation is an efficient method to train neural network which need to use loss function and chain rule. It can avoid complex intermediate term calculation. The purpose is to by update weights to get small loss. The procedure is by loss function, we get lost value in the output. By calculate gradient of loss function with each weight and chain rule, we can get each gradient layer from back to front. Then, we can update each weight which can help us get smaller loss when doing front propagation again.

5 Result comprehension

From picture in Figure 1, I change 5 different parameters to see the influence on the training and testing data.

First is learning rate. It is used to adjust learning speed. We can see that if learning rate is large, it will soon convergence, but it cannot find best weights. In the first picture, we can see that test accuracy stop increasing when epoch equal to 7. However, if learning rate is too small, it needs long lime to converge.

As for momentum, it is used to find global minimum. If it is large, it will make weights converge quickly but may miss global minimum. If it is too small, it may not avoid local minimal and convergence become slow. From second picture, we can see that when epoch equal to 20, the accuracy is still increasing. Hence, find a balance between learning rate and momentum is important.

For weight decay, from picture 3 we can see that the line is smooth. The reason is it can help to avoid complex model which can reduce overfitting influence.

For epoch, we can see that the epoch size is important. If size is too small, the data may not be trained well. From picture 4, the epoch is 10, but the accuracy is just 55.13% for test data and it is still increasing. If it is too large, it will result in overfitting.

Batch size is also important. If batch size is too small, the parameter is hard to converge. From picture 5 we can see that the result is bad because it cannot find right gradient direction. However, if batch size is too large, it will cost more time and weight modify will be slow.

Reference

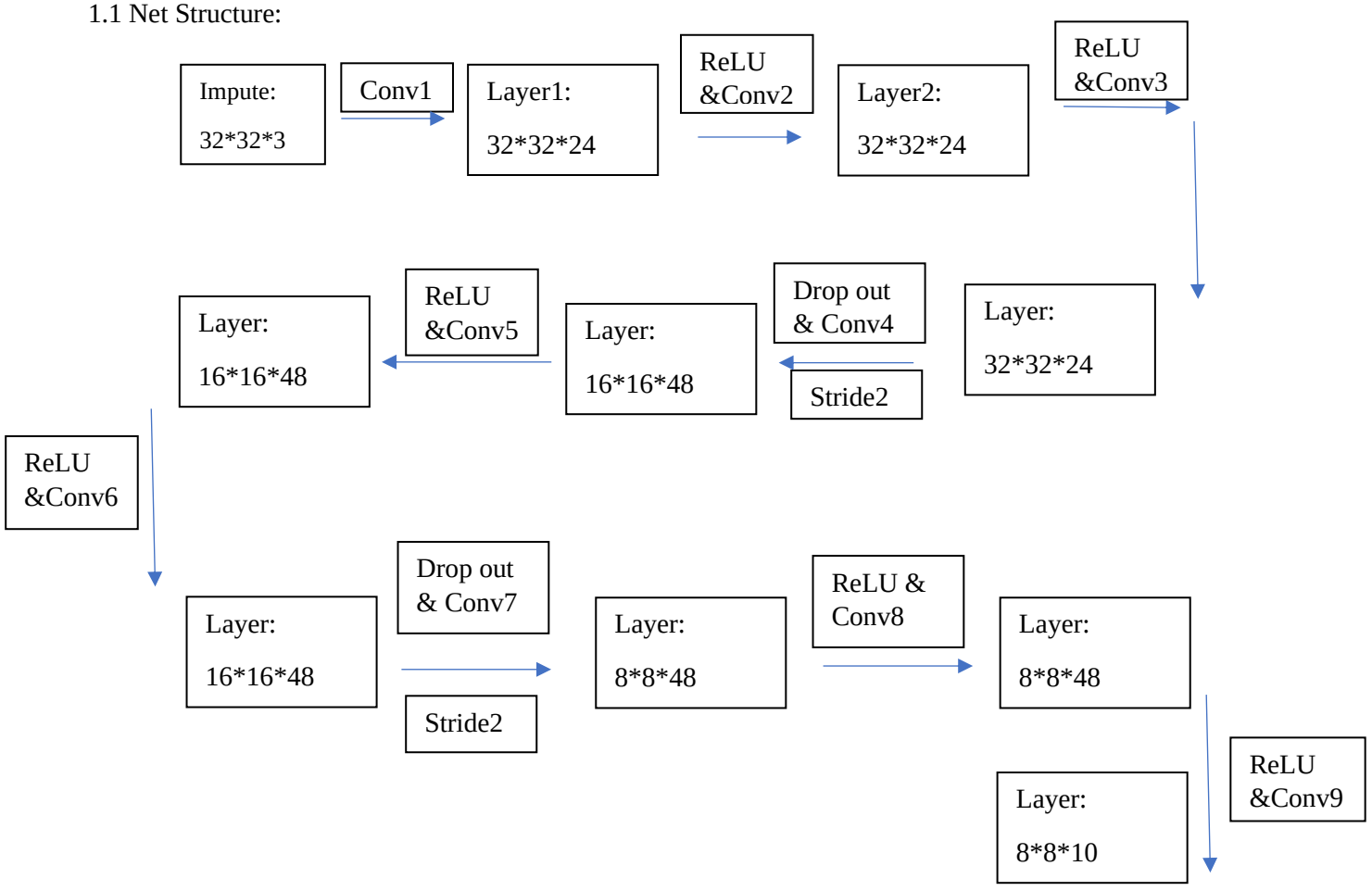
[1] [Online] https://en.wikipedia.org/wiki/Convolutional_neural_network

[2] Y. LeCun. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, November 1998.

Improve method

1 Motivation and logics

1.1 Net Structure:



This network I inspired by All Convolution Net [1]. It consists of 9 convolution layers and 6 ReLU activate functions. In the third and six convolutions, I use 2 strides to shrink filters space. After shrink, I use drop out to randomly set some units be zeros. In the last, I use global average pooling to connect all features and use SoftMax to map features.

1.2 Training parameter setting:

Learning rate: 0.006

I use a low learning rate because it can achieve higher accuracy. If learning rate is large, I can converge fast but can not find the highest accuracy.

Momentum: 0.93

I set a large momentum which can help me find global minimum and converge faster.

Weight decay: 0

I set zero because I already use drop out to reduce overfitting. If I set weight decay here, it will converge slowly.

Drop out: 0.5

I change 50% unit to zero to avoid overfitting.

Epoch:150

Because of low learning rate, I need more epochs to get higher accuracy. If it is too large, just waste time because no accuracy improvement. If it too small, can't achieve its highest accuracy.

Batch size: 64

If batch size is too small, it needs more time to calculate in each epoch. If batch size is too large, the accuracy does not raise stably, so I choose 64 which works well for me.

1.3 Comparing with LetNet5:

This method has smaller model size and higher accuracy comparing with LetNet5 (model size: 135062, accuracy: 63.76%).

Layer number: I use more convolution layers and convolution steps comparing with LetNet5, so it can extract features more deeply. This is an important part that can make it has higher test accuracy than LetNet5.

Shrink method: Instead of using max pooling in LetNet5, I use another convolution layer with stride 2 to substitute it. Hence it not only can reduce the filters dimension, but also reduce accuracy loss.

Connected layer: I use global average pooling to substitute fully connected layer. It can transform features more natural and simpler. It also reduces parameters in this layer which makes it more robust.

Overfitting: In my network, I add drop out function to with transform high probability units to zero which can reduce overfitting. In LetNet5, I use low probability which is not enough.

2 Classification accuracy:

I use GPU to calculate follow result.

2.1

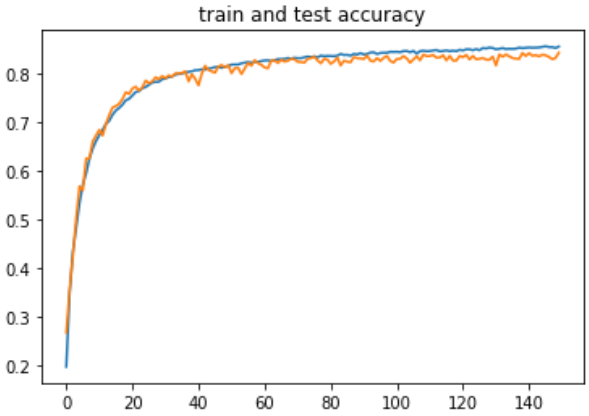


Figure 1

Best accuracy: 84.36%

Training time: 1055.6/s

Inference time: 85/us

2.2

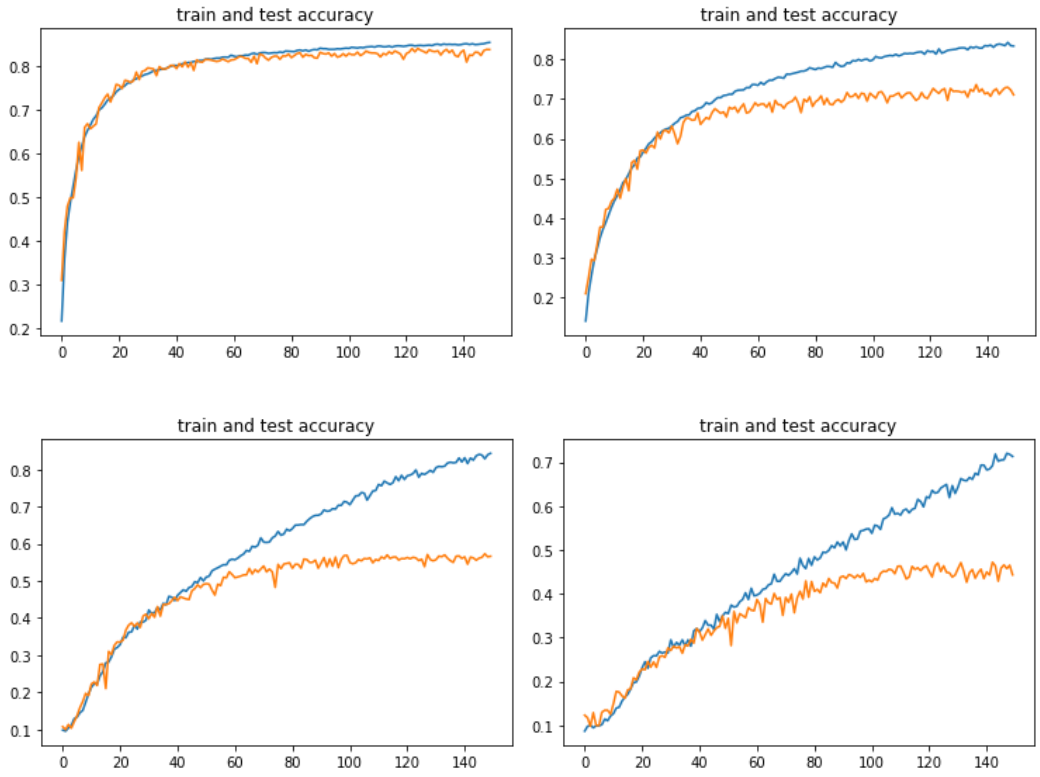


Figure 2 left-top(12500) right-top(6250) left-bottom(3125) right-bottom(1563)

train data	highest accuracy	train time	inference time
50000	84.36	1055.6/s	85/us
12500	72.99	332.4/s	90/us
6250	64.77	209.6/s	92/us
3125	57.28	150.4/s	90/us
1563	46.59	125.2/s	91/us

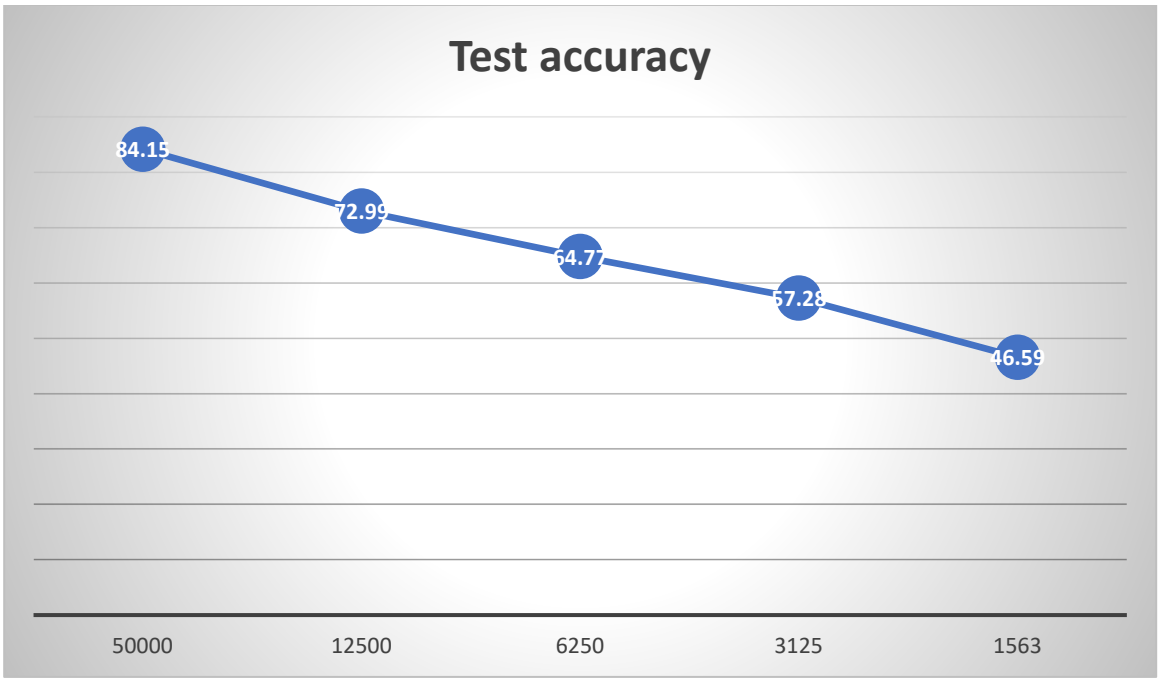


Figure 3

We can see that it decreases slowly.

Screen shoot:

```
train total time: 1055.5921124099996
10000/10000 [==============================] - 1s 89us/step

train total time: 332.3850951029999
10000/10000 [==============================] - 1s 90us/step
train total time: 209.55698842699985
10000/10000 [==============================] - 1s 92us/step

train total time: 150.4410597699998
10000/10000 [==============================] - 1s 90us/step

train total time: 125.22549054899991
10000/10000 [==============================] - 1s 91us/step
```

3 Model size:

Total: 86698

conv2d_57 (Conv2D)	(None, 32, 32, 24)	672
activation_43 (Activation)	(None, 32, 32, 24)	0
conv2d_58 (Conv2D)	(None, 32, 32, 24)	5208
activation_44 (Activation)	(None, 32, 32, 24)	0
conv2d_59 (Conv2D)	(None, 16, 16, 24)	5208
dropout_13 (Dropout)	(None, 16, 16, 24)	0
conv2d_60 (Conv2D)	(None, 16, 16, 48)	10416
activation_45 (Activation)	(None, 16, 16, 48)	0
conv2d_61 (Conv2D)	(None, 16, 16, 48)	20784
activation_46 (Activation)	(None, 16, 16, 48)	0
conv2d_62 (Conv2D)	(None, 8, 8, 48)	20784
dropout_14 (Dropout)	(None, 8, 8, 48)	0
conv2d_63 (Conv2D)	(None, 8, 8, 48)	20784
activation_47 (Activation)	(None, 8, 8, 48)	0
conv2d_64 (Conv2D)	(None, 8, 8, 48)	2352
activation_48 (Activation)	(None, 8, 8, 48)	0
conv2d_65 (Conv2D)	(None, 8, 8, 10)	490
global_average_pooling2d_7 (	(None, 10)	0
activation_49 (Activation)	(None, 10)	0

Total params: 86,698

References:

[1] https://arxiv.org/pdf/1412.6806.pdf