Development of Convolutional Neural Network For Image Classification

Introduction

My final project is to aim to develop a CNN for a simple image classification using the dataset CIFAR-10. Six CNN structures are developed based on a strategy of adding layers step by step to complicate the structure and then check the related accuracy. Number of layers, filter size, FC layers, dropout and batch normalization are investigated here. The highest accuracy of CNN_1-6 is 0.7280, 0.7389, 0.7503, 0.7587, 0.7794 and 0.8183 and the related training time for 20 epochs is 3982, 4497, 11844, 14005, 14332 and 17927s. It is found that the training time stays stable at different epochs, in which convolutional layers contribute to most of the time consumption. More parameters induced by more neurons in hidden layers and filters may contribute to a steady increase in accuracy while too many layers of convolutional filters may not always increase the accuracy. Dropout and batch normalization show a significant enhancement in accuracy.

Structures of the CNN

To achieve a good accuracy, 6 structures of the CNN are developed successively. The idea is to add different functions into the sequence of a simple CNN structure to make it more and more complex step by step. In the process, the old structure is trained to check the variance of the accuracy and the new structure is developed based on the former consequence. Here are the 6 structures of the CNN.

Table 1 Structures of the CNN

	CNN_1	CNN_2	CNN_3	CNN_4	CNN_5	CNN_6
STEP 1	32 filters of size (3*3) with the same size of output	32 filters of size (3*3) with the same size of output	64 filters of size (3*3) with the same size of output	64 filters of size (3*3) with the same size of output	64 filters of size (3*3) with the same size of output	Batch normalization
STEP 2	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	64 filters of size (3*3) with the same size of output
STEP 3	64 filters of size (3*3) with the same size of output	64 filters of size (3*3) with the same size of output	128 filters of size (3*3) with the same size of output	128 filters of size (3*3) with the same size of output	128 filters of size (3*3) with the same size of output	Maxpooling (2,2) with stride (2,2)

STEP 4	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Maxpooling (2,2) with stride (2,2)	Batch normalization
STEP 5	Flatten to 256 neurons	Flatten to 512 neurons	Flatten to 512 neurons	128 filters of size (3*3) with the same size of output	Dropout (0.5)	128 filters of size (3*3) with the same size of output
STEP 6	Softmax function at the output	Softmax function at the output	Softmax function at the output	Maxpooling (2,2) with stride (2,2)	128 filters of size (3*3) with the same size of output	Maxpooling (2,2) with stride (2,2)
STEP 7				Flatten to 512 neurons	Maxpooling (2,2) with stride (2,2)	Batch normalization
STEP 8				Softmax function at the output	Flatten to 512 neurons	Dropout (0.5)
STEP 9					Dropout (0.5)	128 filters of size (3*3) with the same size of output
STEP 10					Softmax function at the output	Maxpooling (2,2) with stride (2,2)
STEP 11						Batch normalization
STEP 12						Flatten to 512 neurons
STEP 13						Dropout (0.5)
STEP 14						Softmax function at the output

In the table above, filters of size (3*3) with the same size of output means that padding is applied to the matrix so that after filtering the shape of the matrix doesn't change. (The related code is '*border_mode = "same"*')

Parameters for variation and Time for training

Here are the basic parameters.

batch_size = 128, num_classes = 10, epochs = 20, img_rows, img_cols = 32, 32

As for the variation of the parameters in the CNN, the following gives a summary of the variation.

In CNN_1, or called the basic structure of the CNN, filtering is applied twice with 32 filters and 64 filters in turn, each followed by max-pooling (2,2) with stride (2,2). Then, flatten the matrix and connect to 256 neurons. Finally, connect to 10 neurons at the output. In CNN_2, the number of neurons in the hidden layer is doubled to 512 versus CNN_1. In CNN_3, the number of filters in both convolutional layers is doubled to 64 and 128 versus CNN_2. In CNN_4, a new convolutional layer with 128 (3*3) filters is applied versus CNN_3. In CNN_5, two dropout layers are added into the structure with a rate of 0.5 versus CNN_4. In CNN_6, batch normalization is applied forth times to normalize the data.

The following table and figures shows the time needed for training.

Table 2 Time needed for training (Unit: s)

Epoch	CNN_1	CNN_2	CNN_3	CNN_4	CNN_5	CNN_6
1	198	223	589	618	723	894
2	198	223	596	709	722	887
3	198	224	598	711	727	888
4	198	226	601	714	717	891
5	198	226	595	731	719	889
6	198	225	595	721	718	893
7	198	225	597	720	711	900
8	198	225	597	712	714	897
9	200	225	594	699	722	893
10	201	225	590	690	715	904
11	201	225	587	695	710	895
12	200	225	588	697	714	886
13	200	225	591	699	716	893
14	199	225	590	696	714	899
15	200	225	594	697	713	904
16	200	225	588	698	713	904
17	200	225	589	701	714	903
18	199	224	588	700	714	901
19	199	225	588	699	721	909
20	199	226	589	698	715	897
Average	199	225	592	700	717	896

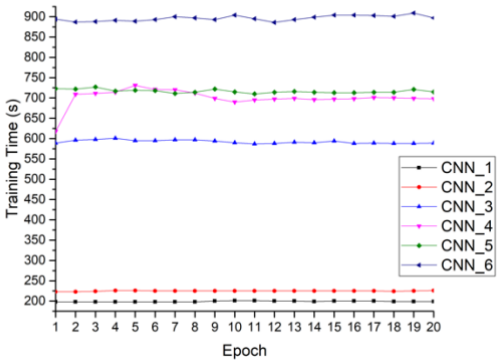

Figure 1 Training time versus epoch

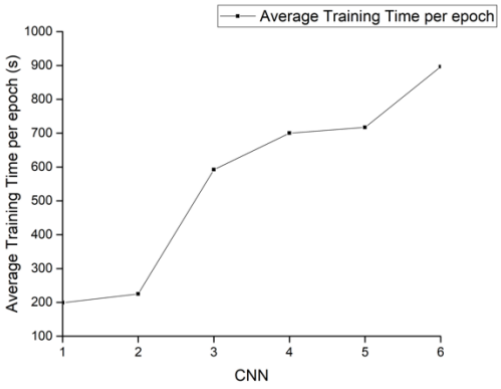

Figure 2 Average training time per Epoch versus CNN

The above figures show that the training time has nearly no difference among different epochs. On the contrast, it ranges a lot in different CNN structures. In general, the training time increases when the steps in the CNN increase. In detail, the increase of the convolutional layers contributes to most of the time consumption while the others have much lower influence.

Results

The accuracy of different CNN structures in 20 epochs are shown below.

Table 3 Accuracy of different CNNs

Epoch	CNN_1	CNN_2	CNN_3	CNN_4	CNN_5	CNN_6
1	0.5477	0.4818	0.5499	0.4124	0.4579	0.631
2	0.6147	0.5933	0.6133	0.6013	0.5354	0.6908
3	0.6145	0.6551	0.6675	0.6744	0.6164	0.7234
4	0.6655	0.6807	0.6561	0.6992	0.6488	0.7447
5	0.6983	0.7036	0.6957	0.7374	0.6902	0.7678
6	0.7169	0.7004	0.7213	0.7267	0.7101	0.7723
7	0.7113	0.7183	0.7243	0.7414	0.7017	0.7811
8	0.6879	0.6965	0.7223	0.7301	0.7097	0.7869
9	0.7015	0.7232	0.72	0.7371	0.7393	0.7808
10	0.721	0.7116	0.728	0.7531	0.7534	0.7938
11	0.7199	0.7168	0.7415	0.7416	0.7553	0.8016
12	0.7139	0.7071	0.7339	0.7462	0.7581	0.802
13	0.7178	0.7346	0.7367	0.7544	0.7534	0.8112
14	0.717	0.7312	0.7503	0.741	0.7689	0.8145
15	0.7203	0.7287	0.7486	0.7356	0.7631	0.8043
16	0.7227	0.7309	0.7461	0.7526	0.7713	0.8026
17	0.6638	0.7317	0.7465	0.744	0.7744	0.8159

18	0.7274	0.7389	0.7489	0.7397	0.7551	0.8155
19	0.728	0.738	0.7483	0.7587	0.7794	0.8183
20	0.7233	0.7379	0.7486	0.745	0.7778	0.8155

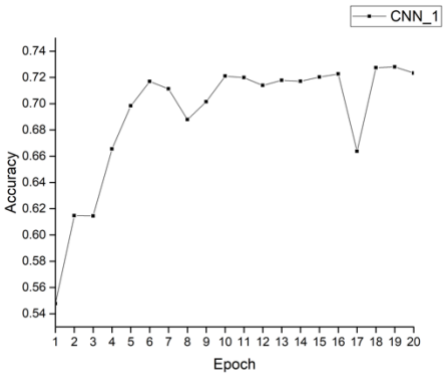

Figure 3 Accuracy of CNN_1

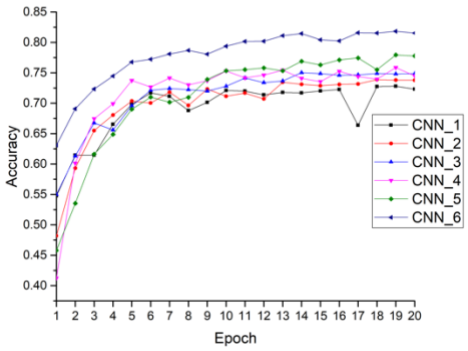

Figure 9 Accuracy of 6 CNNs

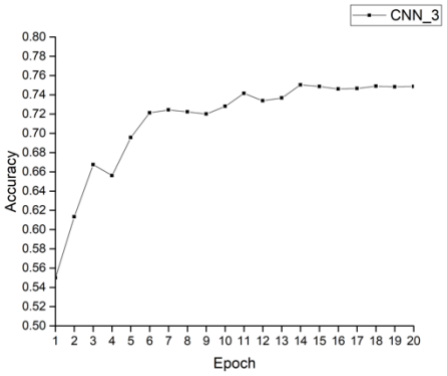

Figure 5 Accuracy of CNN_3

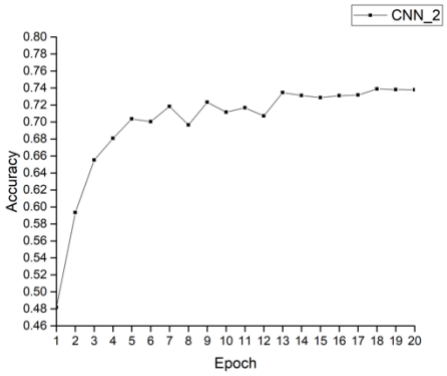

Figure 4 Accuracy of CNN_2

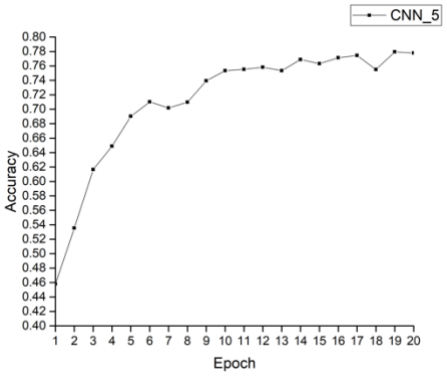

Figure 7 Accuracy of CNN_5

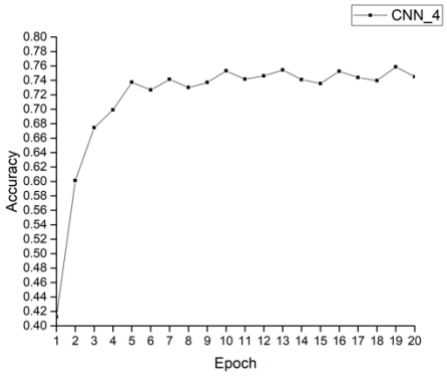

Figure 6 Accuracy of CNN_4

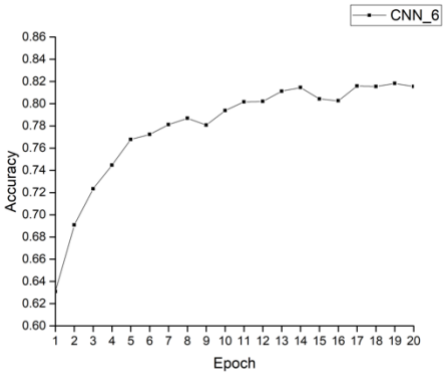

Figure 8 Accuracy of CNN_6

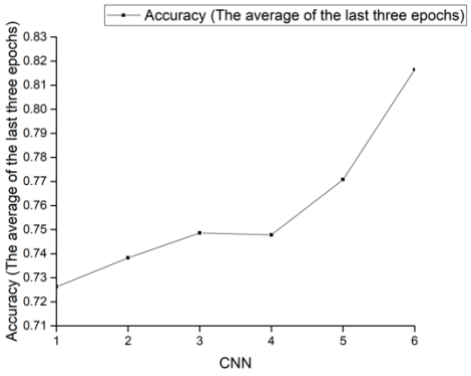

Figure 10 Accuracy (The average of the last three epochs) of 6 CNNs

Figure 3-8 show the accuracy of the 6 CNNs in the first 20 epochs separately. Figure 9 puts 6 CNNs together to compare different CNNs. Figure 10 gives the average accuracy of the last three epochs with the range of CNNs.

Discussion of the results

According to the results above, it is shown that along with the increase of the epoch, the accuracy of all the 6 CNNs increases slower and slower, and finally fluctuates at a fixed level. At the first epoch, different CNNs present quite different accuracy ranged from 0.4124 to 0.6130. This shows that different structures provide quite different training rates at the beginning. Then, the trends of different CNNs seem to be similar to each other at the same epoch. However, at the 17th epoch of CNN_1, the accuracy has a sudden drop and later resumes to the former level at the 18th epoch. This phenomenon also occurs at the 18th epoch of CNN_5. I guess this is an accidental phenomenon at a specific situation because of the interference among different parameters while the network can correct it quickly.

As a result, the accuracy of the 6 CNNs ranges from 0.72 to 0.82. While the highest accuracy of CNN_1 is 0.7280 at the 19th epoch, the highest accuracy of CNN_2 is 0.7389 at the 18th epoch. The highest accuracy of CNN_3 is 0.7503 at the 14th epoch. The highest accuracy of CNN_4 is 0.7587 at the 19th epoch. The highest accuracy of CNN_5 is 0.7794 at the 19th epoch. The highest accuracy of CNN_6 is 0.8183 at the 19th epoch.

Comparing the 6 CNNs with each other, we can find out the influence of different layers to the accuracy. According to the strategy of developing the CNN, the structure is complicated step by step and once only a single kind of layer is added to the structure. From the result, we can see that adding neurons in hidden layer and filters contributes to a steady growth in accuracy. This implies that adding parameters may enable the result more accurate. Meanwhile, comparing CNN_3 with CNN_4, when the

convolutional layers are increased from 2 to 3, the accuracy doesn't increase. This is different from the common sense, which implies that continuing deepen the CNN may not always give a better result. On the contrast, dropout layers and batch normalization layers contribute to a sudden increase of the accuracy! The accuracy increases from 0.7478 to 0.7708 with the help of dropout and increased from 0.7708 to 0.8164 with the help of batch normalization. Maybe the dropout layers deactivate the excessive neurons so that the overfitting is relieved, and the accuracy becomes better as a result. As for the batch normalization layers, the range of the data is controlled at a steady level and makes it easier for training. The higher learning rates can be reached as a result.

Conclusion

The training time stays stable at different epochs, while convolutional layers contribute to most of the time consumption. To reduce the time consumption, the number of the convolutional layers should be controlled. Overall, the time consumption increases when the CNN structure becomes more complex.

As for the accuracy, it increases fast in the first several epochs and then slower and slower and finally fluctuates at a fixed level. Generally, the accuracy becomes higher when the CNN structure becomes more complex. The highest accuracy is 0.8183 at the 19^{th} epoch in CNN_6 and the related training time is 909s at the epoch and 17030s in total. More parameters induced by more neurons in hidden layers and filters may contribute to a steady increase in accuracy while too many layers of convolutional filters may not always increase the accuracy. Dropout and batch normalization show a significant enhancement in accuracy.

Data Source

The CIFAR-10 Dataset

https://www.cs.toronto.edu/~kriz/cifar.html