Samwoo Seong
7953-6137-66
samwoose@usc.edu

EE569 Midterm 2 (HW5 P2)
April 30, 2020

1.

1.1 Motivation and logics behind my design

Each component in a CNN model plays certain role. If we utilize their roles while designing a CNN model, then we can obtain improvement in terms of accuracy. [1]
We have changed following hyper parameters according to their roles.

(1) Kernel/ Filter size: filter size determines whether filter wants to capture local or global information when convolution is operated. Since we are dealing with small size of images, we employ 5x5 filter size in our model.
(2) Padding: padding can bring 2 main effects to a CNN model. First, we can preserve information at border of image after performing convolution. Second, it will help us to maintain spatial dimension of outputs at each layer. It will potentially make our CNN able to be built deeper. In our model, we make a use of padding by size of two at each spatial dimension (i.e., row and column).
(3) Stride: Not to miss details from input images, we build our model with stride 1 at every convolutional layer.
(4) Number of channels: the more channels(filters) we have, the more patterns(features) we are able to capture. However, it also increases chances of overfitting especially when number of training samples is not enough. Therefore, we slightly increase number of filters at each layer. It is shown below at Architecture of CNN section.
(5) Pooling layer: Pooling layer can not only reduce the dimension of current output but also allow a CNN model to retain the most important spectral information at next layer. We insert max-pooling layer after every convolutional layer.
(6) Depth of CNN model: Generally, the deeper CNN goes, the wider and more detailed receptive field can be captured by the model. However, it will lead us to bigger size of model which delays training time. Therefore, we decide to construct only 3 convolutional layers to show slight performance improvement.
(7) Non-linear activation function: role of non-linear activation function is to introduce non-linearity to CNN. We make a use of ELU. This is because generally it causes better performance when max-pooling is employed.
(8) Number of epochs: Generally speaking, accuracy on training or test set converges at some point of epochs. To avoid underfitting caused by non-sufficient number of epochs, we choose 80 epochs in our model.

Since we covered roles of parameters such as learning rate, weight decay, momentum extensively at homework5, we omit details on them. Briefly speaking learning rate should be small enough to converge, weight decay should be large enough to prevent overfitting but small enough to avoid underfitting, and lastly, momentum should be large enough to escape local optima. Chosen pair is shown on Table 2.

Hyper parameter	Chosen value
Filter size	Size of 5x5 at every convolutional layer
Padding	Zero padding. Size of 2 for each dimension
Stride	1 at every convolutional layer
Number of channels	Input: 3 1st convolutional layer: 10 2nd convolutional layer: 20 3rd convolutional layer: 30 1st FC layer: 130 2nd FC layer: 94 3rd FC layer: 10
Pooling Layer	Max-pooling after every convolutional layer
Depth of CNN model	3 convolutional layers & 3 FC layers
Non-linear activation function	ELU
Number of epochs	80

Table1. Summary of chosen hyper parameters.

	Pair (Learning rate, Momentum, Weight decay)
Chosen pair	(0.02, 0.7, 0.01)

Table2. Chosen pair of parameters.

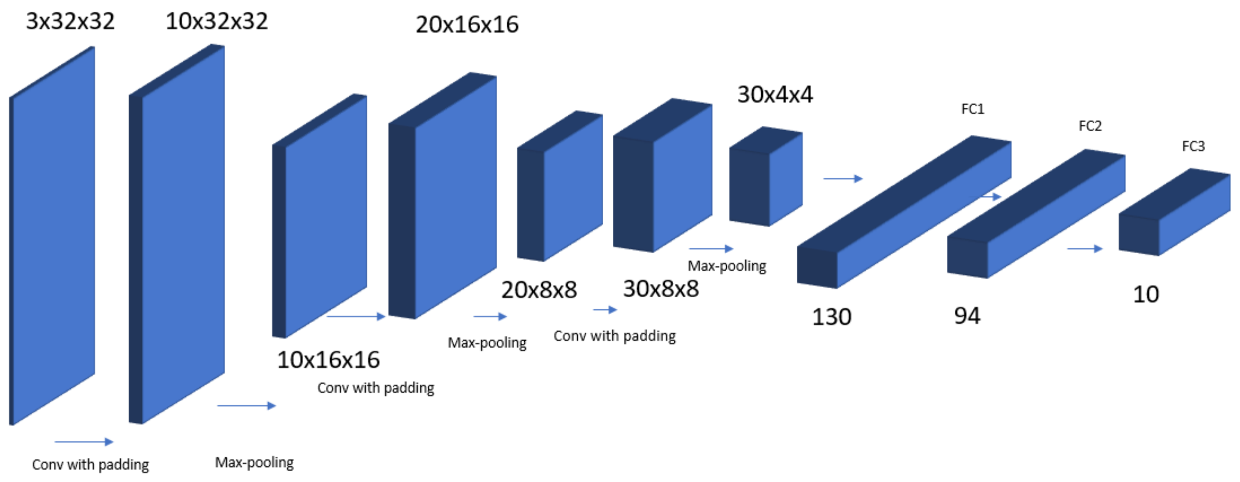

Figure 1. Architecture of CNN

1.2 Experiment results

(0) Hardware Specifications

Memory	8GB RAM
Processor	CPU, Inter core i5

Table 3. Hardwar specs

(1) Best Accuracy

69% on test set

Note: Best accuracy on test set at HW5: 63%

=>6% of improvement in terms of accuracy on test set

(2) Training time

About 3 hours of training 50000 images

(3) Inference time

9.71 seconds for 10000 test images

=> 0.000971 per image

(4) Performance curves

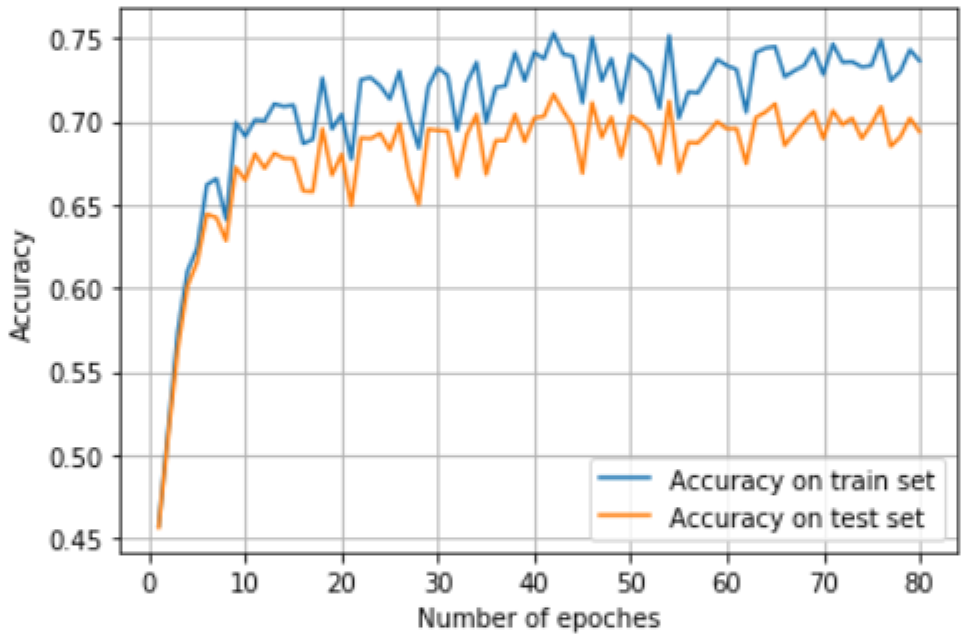

Figure 2. Performance curves on training set and test set

(5) Test accuracy as a number of training samples drops

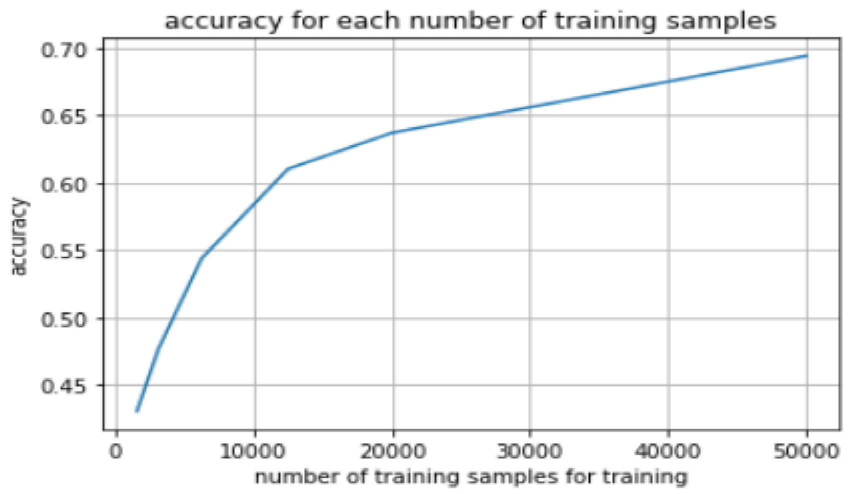

Figure 3. Test accuracy as a number of training samples drops

(6) Model size

	Current dimension	# of learnable parameters
Input	(3,32,32)	0
1st conv	(10,32,32)	(5 * 5 + 1) * 10 = 260
1st pool	(10,16,16)	0
2nd conv	(20,16,16)	(5 * 5 + 1) * 20 = 520
2nd pool	(20,8,8)	0
3rd conv	(30,8,8)	(5 * 5 + 1) * 30 = 780
3rd pool	(30,4,4)	0
1st FC	(130,1)	(30 * 4 * 4) * 130 + 1=62401
2nd FC	(94,1)	130 * 94 + 1 = 12201
3rd FC	(10,1)	94 * 10 + 1 = 941
Total # of learnable parameters	260 + 520 +780 + 62401 + 12201 + 941 = 77103	

Table4. Model size

1.3 Discussion

-Observation on accuracy improvement and its sources

We have modified number of channels, number of convolutional layers, non-linear activation function, and added zero padding method. As we explained earlier, increased number of channels allows our model to capture more patterns at each layer. The deeper convolutional layer causes our CNN to have wider receptive fields. Smoother non-linear activation function helps max-pooling work better. Finally, zero padding method assists our model to retain information at borders. As a result, we could obtain improved accuracy on test data set.

-Observation on training and inference time

As we expected, since model size has increased due to number of filters and number of convolutional layers, it takes quite long time to finish training our model. We can assume that the deeper and larger model are, the longer it takes for training.

-Opinion on designing CNN structure

Even though we could end up getting some improvement by simply making our model deeper, it can be time consuming and waste of computational resource if we blindly build a deeper network because it employs end-to-end optimization method. Therefore, we need to figure out how to extract feature or representation of images like humans do and perform classification without optimization process. One way introduced by Kuo et al is Successive Subspace Learning method. It will be investigated further in HW6 P3 report.

-Robustness

As we can see at Fig. 3, performance drops significantly as number of training samples is reduced. This is because CNN based model is highly relying on labeled data.

-Future work

For the future work, we can build a more complex CNN model by adding more layers and using different max-pooling methods. However, as we mentioned above, we do need to have fundamental understanding of how humans are able to classify objects so quickly and utilize the principle instead of mindlessly finding a “better” architecture of CNN.

References

[1] Shashank Ramesh., May 7, 2018. “A guide to an efficient way to build neural network architectures-Part II: Hyper-parameter selection and tuning for Convolutional Neural Networks using Hyperas on Fashion-MNIST” [Online] Available:

https://towardsdatascience.com/a-guide-to-an-efficient-way-to-build-neural-network-architectures-part-ii-hyper-parameter-42efca01e5d7