Melih S. Aslan

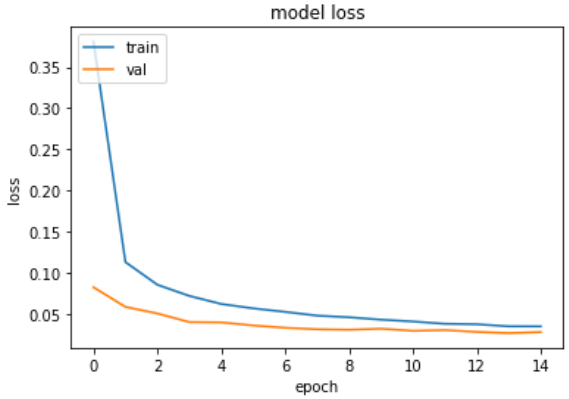
2/26/22

MNIST CLASSIFICATION ASSESSMENT

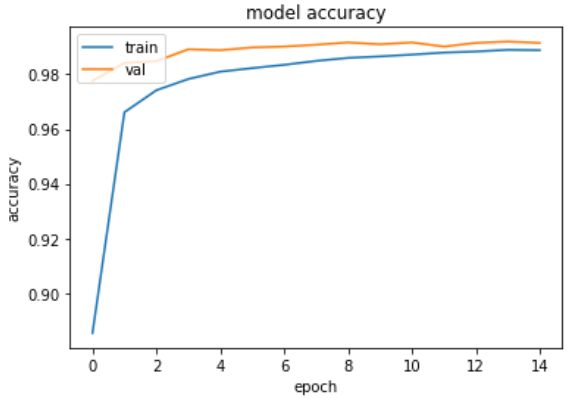
Warmup:

The validation loss after the first epoch less than the train loss. The validation accuracy after the first epoch is higher than the train accuracy. The reason could be the selection of the percentage of train and validation split, 0.9 and 0.1, respectively. Or other reasons could i) the regularization applied on the train, and ii) train loss is reported during the epoch whereas the validation loss is report after the epoch.

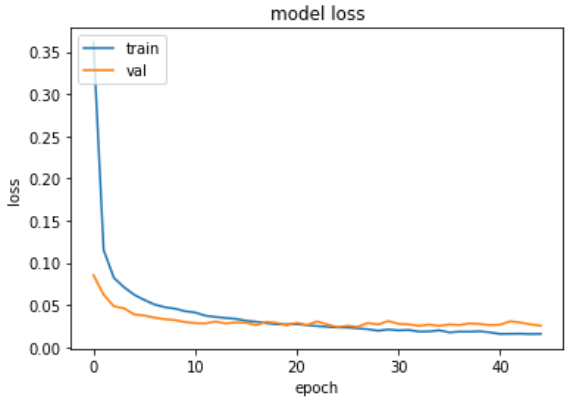
In the 15th epoch, the loss of train and validation get closer to each other like the accuracy values. I suspect if this is a sign of overfitting. I run the experiment again with 45 epochs to see the behaviors of validation and train. I observed that after 25th epoch train loss decreases further while the validation loss becomes saturated. Since the validation loss is around 0.05, this behavior can be neglected since we got >98% accuracy in either case.



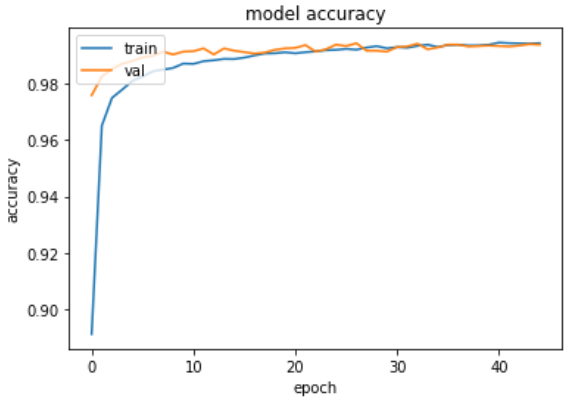
Loss plots of train and validation



Accuracy plots of train and validation



Loss plots of train and validation (**45 epochs**)



Accuracy plots of train and validation (**45 epochs**)

Total parameters used in this warmup training is 34,826.

The memory usage can be considered with respect to other convolutional neural networks such as MobileNetV2, etc. Since the image size and number of training and validation images are small, we do not need to have big network that can cause overfitting too.

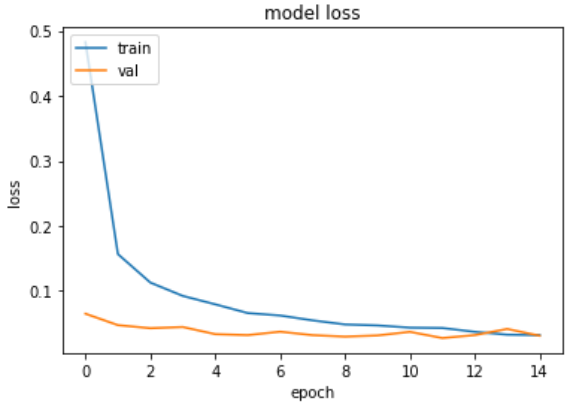
Technical Analysis:

1) MNIST data consists of 10 handwritten digits that contains variations such as shapes, locations, orientations, etc. We can use various machine learning models to achieve the classification. However, convolutional neural networks used to achieve the best accuracy out of other candidates. I imagine, I can analyze the location of digits and intensity information. To save time for other questions and bonus, I would like to give a brainstorming answer for this question.

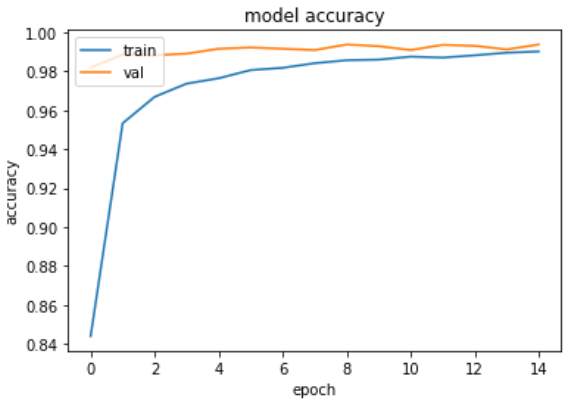
2) When I increase the number of layers, there is a point that I should be careful. Since the number of max-pooling layers increases, the output size decreases towards 0. In order not loosing important information with using more max-pooling layers, I used only convolutional layers in the first 2 layers, then added max-pooling layers after the last two convolutional layers. Code snippets and plots are demonstrated below:

```
model = keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        #layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        #layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dropout(0.5),
        layers.Dense(num_classes, activation="softmax"),
    ]
)
```

The Total params: 130,250.



Loss plots of train and validation



Accuracy plots of train and validation

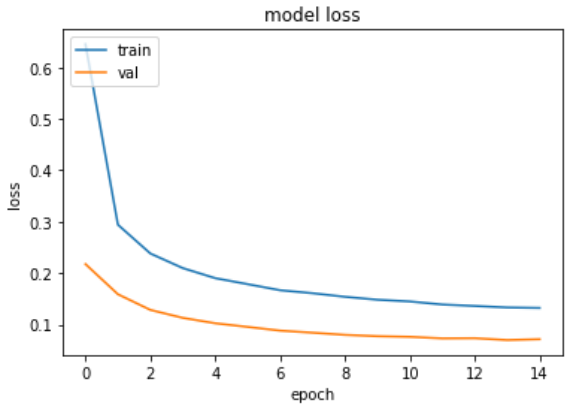
In the figure above, I observe that the validation accuracy rapidly saturated above the point of 98% accuracy just after the first epoch. Since the model is bigger in this experiment, the training time of 15 epochs increased when it is compared with the Warmup model with 15 epochs.

3) I deleted the second convolutional layer and decreased the filter number of the first layer to 8.

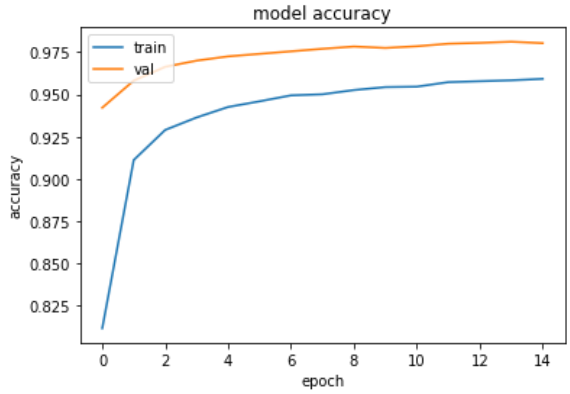
```
model3 = keras.Sequential(

    [
        keras.Input(shape=input_shape),
        layers.Conv2D(8, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dropout(0.5),
        layers.Dense(num_classes, activation="softmax"),
    ]
)
```

I observe that the validation accuracy reaches to the point of >98% accuracy around at the 15th epoch; whereas the bigger network reaches that point just after the 1st epoch.

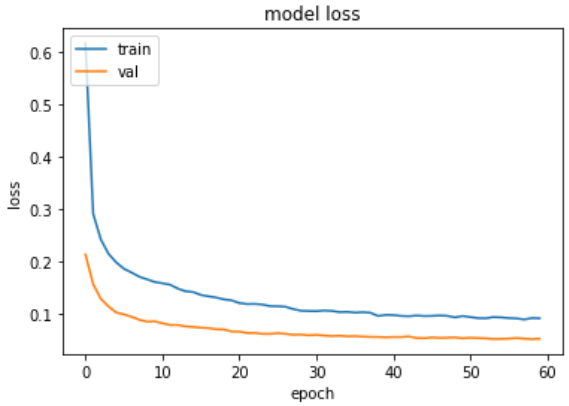
Another interesting point in this experiment is that the difference between the train and validation numbers (both loss and accuracy) are bigger than the other models. So, I run the small model with 60 epochs to see if those gaps would decrease. This gap slightly decreases when I run the experiment for 60 epochs but not completely. Since the train accuracy is still under 98%, there is still room to improve. But I was still able to get >98% validation accuracy. Therefore, I would not spend more time on this question.



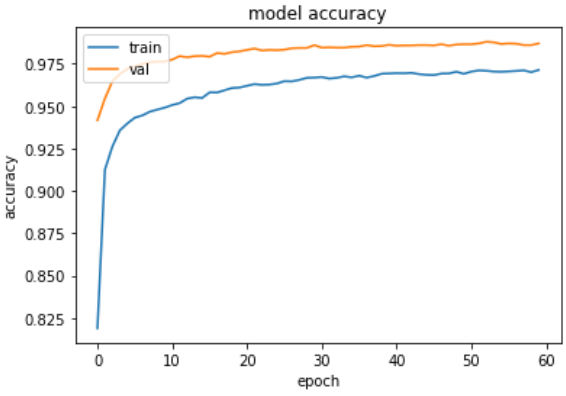
Loss plots of train and validation



Accuracy plots of train and validation



Loss plots of train and validation (60 epoch)



Accuracy plots of train and validation (60 epoch)

Bonus:

I run this experiment on the model in the Warmup question. I enumerated the first 12 filters out of the 32 in the first block and plot them as below. The first layer's weights are responsible to capture fine details such as horizontal, vertical, diagonal, and more various edges from the input images.