Learning and Implementation of Artificial Neural Networks(ANN) and Convolutional Neural Network(CNN)

Emanuel macias

Abstract: Artificial neural networks as the naming suggests are an interpretation or inspiration of the neurons in the human brain. Artificial neural networks imitate the brain because the brain is the best learning system known. Since they are artificial they are virtualized using computer hardware. Both neural networks require the input layer and the output. This takes inspiration from neurons which have dendrites that take in information or electrical signals and axions that output electrical signals but with a transformation. The transformation of information that takes place in between neurons is what makes artificial neural networks work. While ANN and CNN differ in their approach they contain a layer in between output and input. Ann is called the hidden layer(s) which implements a linear function to determine the information from the input. It also contains an activation function which determines how influential the information obtained from the input will be to the output. CNN on the other hand uses convolution layers which are designed to process images to recognize content in sed images.

Problem 1:

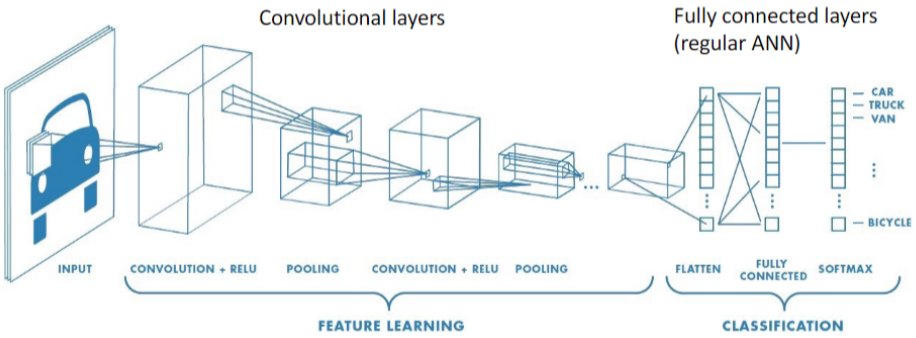
a. If the model has 1 hidden layer and 1 neuron within the hidden layer, what is the prediction accuracy on the test set? Increase the number of neurons in the hidden layer to 100, what is the accuracy now? What could be the reason?
    i. When the model has 1 hidden layer with 1 neuron the accuracy obtained was 0.4545..
    ii. Increasing the number of neurons in the hidden layer gave an accuracy of 0.99
    iii. Increasing the number of neurons allows the hidden layer to more accurately understand the information or data given to it. More weights and biases go into each neuron with its own linear function direction by its own activation function which decides the significance of the data given. Allow from manipulation of the input giving in turn a more accurate output.
b. Build the model with 1 hidden layer and 1,000 hidden neurons, what is the accuracy if we set the constant learning rate as 0.1? What is the accuracy if we change the learning rate to 0.001? What is the reason?
    i. Increasing the neurons but leaving the hidden layer a 1 gave similar results as the 100 neurons; 0.991 when the default constant learning rate is kept which is 0.001.
    ii. When changing the constant learning rate to 0.1 gave an accuracy of 0.993. A slight increase in the accuracy when the learning rate was increased.

iii. This can be because the learning rate is more optimal when closer to 0.35 as stated in the slides. Since 0.34 > 0.1 > 0.001. Being closer to the optimal learning rate prevents errors like overshooting or the convergence taking too long.

c. What is the method to choose the appropriate number of hidden layers, number of hidden neurons and learning rate? Describe.
   i. Depending on the complex of the data depends on the number of hidden layers. If the data is linearly separable then a hidden layer is not needed. If the data is not as complex then a few dimensions are needed, around 1-2. If the data is large with large dimensions or features 3-5 layers can be used. Even though there's no limit to the number of hidden layers, adding more hidden layers can lead to overfitting. To choose the optimal nodes/neurons for hidden layers, it should be in between the size of the input and output layer. The formula for the best number of neurons is sqrt(input layer nodes * output layers nodes). The correct learning rate was listed in problem b.

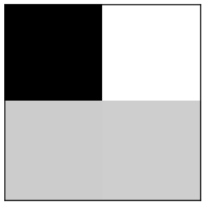
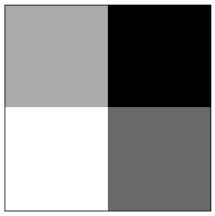
Problem 2:

1. The above code builds a CNN model. What is the final prediction result you get?
   a. The final prediction of the implementation code for the CNN model; gave an accuracy of 0.69~
2. Add one more Convolution operation to the model with the stride (2,2). Add one more Average pooling operation to the model. What is the final prediction result? Any change of the final result? What do you think could be the reason?
   a. The result was similar: the final accuracy of the model CNN was 0.69~ when the implementation of one more convolution and average pooling was added. Although the initial accuracy from Epoch 1 and 2 did differ. 0.4~0.7 respectively for the added content. While the given code with no changes gave an epoch 1: 0.55, and epoch 2: 0.70. Thinking about the purpose of the CNN model and how it works, it makes sense that the accuracy did not really change. Using similar convolution layers back to back, is like using the same formula to find the same result. Some changes or learning is possible but minimal. Same can be said for the pooling operation.



3. Visualize the filter/kernel.

`no changes: 

added changes:   

Appendix:

Hidden layer, hidden neuron:
https://www.linkedin.com/pulse/choosing-number-hidden-layers-neurons-neural-networks-sachdev/

Visualizing the filters:
https://www.kaggle.com/code/arpitjain007/guide-to-visualize-filters-and-feature-maps-in-cnn