CPSC 4830

Data Mining for Data Analytics

Assignment 2 Part 2

1. Neural Network
2. Draw a fully connected Neural Network with the following specifications:

* Inputs:
* There are totally 3 hidden layers
* For Hidden layer 1, it is made up of and
* For Hidden layer 2, it is made up of and
* For Hidden layer 3, it is made up of and
* The Output layer is made up of and
* Include the bias terms in all layers marking them as with m as their layer number

Make sure you label your drawing.

1. Suppose the activation function for each layer are tanh function and you initialize all weights and bias as zero. You do the forward propagate with an input vector x= (1,2,3,4) in the network. What is the output of o?

* As all weights and bias are zero the output will also be 0 for the network.

1. You are doing stochastic gradient descent using the entire training set. Is it necessary to shuffle the training data? Explain your answer within 2 sentences.

* Yes to prevent the data from being biased and allows the model to learn from different patterns in each epoch. Also, shuffling helps to prevent overfitting specific patterns.

1. What is a loss function?

* A loss function evaluates how well a machine learning model performs on a dataset by quantifying difference between predicted and true values. Loss functions are a measurement of how good your model is in terms of predicting the expected outcome.

1. How to detect if your learning rate being set too large?

->

1. If the loss function rapidly increases or fluctuates wildly during training, it could indicate that the learning rate is too large. The model might be "overshooting" the optimal parameters and failing to converge.
2. Large learning rates can cause gradients to become unstable or explode, leading to NaN (not a number) or infinite values in the loss function or gradients.
3. A loss curve that exhibits a lot of noise or fluctuations instead of a smooth decrease could suggest that the learning rate is too large, causing the optimization process to oscillate around the minimum rather than converging steadily.
4. Ifthe model performs well on the training data but poorly on the validation or test data, it could be a sign of overfitting due to the learning rate being too large. The model may have memorized the training data instead of learning meaningful patterns.
5. If the loss stops decreasing or decreases very slowly over time, it might indicate that the learning rate is too large, preventing the model from making meaningful progress towards minimizing the loss function.
6. You have built a model to classifier if a person is male or female. Your test set (X\_test,Y\_test) is such that the first m images are male, and the remaining images are of female. Now you tried to evaluate your model by min-batch approach. You have shuffled your X\_test and Y\_test, you found that the accuracy of the model with shuffling is better than the accuracy of the model without shuffling.

Question: There are a few wrong statements or conclusions in the above statement. Please point it out.

* + The set should be train set with (X\_train, Y\_train) and should be shuffled.
  + The train set should ideally be random and shuffled first but its described that the first m images are male.
  + The min batch should not be considered during evaluation and should be referred to while train like mini-batch training.
  + The shuffling should occur during train to introduce randomness and so that the training batch is not biased.
  + For testing, we input a single image and try to identify if it’s male or female. No shuffling is required for test.

1. You'd like to train a fully-connected neural network with 10 hidden layers, for which the first 5 hidden layers with 5 hidden units and the remaining hidden layers with k hidden units where k is the k-th hidden layers. The input is 7-dimensional and the output are 2 neurons. What is the total number of trainable parameters in your network?
2. Consider the figure below:

20

20

CONV2D  
f=(5,5)  
# filters = 6  
Stride = (3,3)  
Padding = 5

What is the output shape after performing the convolution step in the above figure? Note we are referring to the image after convolution, but not the activation map. Write your answer in the following format: (nH, nW, nC), where H = height; W = width; C= channels.

1. Suppose the input features = 10. A binary classification problem could be solved with the two approaches described below:

Approach 1: Simple Logistic Regression (1 neuron at the out layer)  
Your output will be , by sigmod function

Classify as 0 if and 1 otherwise.

Approach 2: Simple Softmax Regression (2 neurons at the output layer)

Your output will be

Classify as 0 if and 1 otherwise.

1. Calculate the number of parameters in Approach 1 and Approach 2.
2. Does it mean Approach 2 can learn a more complex model than Approach 1?  
   The answer is NO. Please use the classifier to show that they are equivalent.  
   Note that this exercise shows you the number of parameters is not the only measure for a model complexity. (Nowadays all LLMs are trying to compete in this number.)
3. As a data scientist in NOZAMA Inc., you are asked to build a pipeline to predict the robotic arm to get a parcel in the warehouse from a series of input images. That is Robotic arm problem.
4. You have set up a 3-D camera (for parcel orientation) on each shelf of the warehouse and a camera (for bar code scanning) on the robotic arm. An expert warehouse colleague has collected dataset X in a warehouse by controlling the robotic arm manually where all the parcels are arranged in an orderly manner (i.e. stacked up one on one and bar code always facing perfectly at front like the following picture). He always gets the parcels by removing the parcels on top and put them aside in a perfect way.



1. List any one problem that may occur if you based on this dataset X and the test data in the real world.

* Parcel may or may not be arranged in the orderly mannered in real world compared to the above dataset. Plus the orientation, stacking configuration or environmental conditions like lightning may be different.

1. Without collecting new data, how would you help to cater any of the problem you can think of? What kind of problem you must have to collect the data again? Describe your approach in details.

* 1. To solve we can do data augmentation and perform model building to help in real world.
* 2. Use previously trained models to improve performance with limited data and replicate the same for the real world.
* If the data lacks diversity collecting new data may be necessary, example the dataset X only contains images of parcel arranged in single orientation stacked one above the other while the real world may have different order and needs new data with different orientations to help build a model for real world. Additionally, if dataset X is collected under controlled conditions that do not accurately reflect the complexities of the real-world warehouse environment, collecting new data in the actual operational setting would be necessary to address these limitations.

1. Give any 2 reasons that will stop you from using end-to-end approach to build this model.

* For end to end approaches a deep learning model is required with multiple layers making them highly complex and difficult to interpret. A better way would be to break down problem into smaller insights to better debug and trouble shoot problems.
* This approach requires large amount of data with diverse real-world scenarios and environmental conditions that has high cost. Better would be to use domain knowledge to reduce the requirement and improve generalization.

1. Given your pipeline looks like the following graphs:

Parcel Orientation, P



Movement

Planner,M

Bar Code Detector, B

The input image is given to two modules:  
The Parcel Orientation, O, and the Bar Code Detector B.  
O outputs a set bounding boxes localizing the parcel orientation.  
B outputs a set of bounding boxes localizing the bar code.  
The bounding boxes are then given to a Movement Planner M which outputs the robotic arm movement plan (including remove the parcels on top and get the targeted parcel).  
Assume all these submodules are supervised learning algorithms.

1. What data do you need to collect in order to train the submodules O, B and M?

* 1. For O, data like labels and parcel orientation will be required.
* 2. For B, barcode types and sizes will be required.
* 3. For M, for movement data like the space of the warehouse, size of boxes, orientations, barcode details will be required.

1. Explain how you would collect the data mentioned above.

* Using high quality cameras to collect data in different light conditions and different locations to ensure no biases.
* Using multiple annotation tools to provide training for the dataset from tools like LabelBox, CVAT.
* Capturing arm movement of robot in different lighting conditions.

1. How would you measure human level error for your robotic arm movement task?

* Having different performance metrics like the barcodes not present on the same side of the box maybe on top or bottom and difficult for robot to understand.
* Havind multiple box sizes which will be used as a metric.

1. Let’s get back to how to build the classifier. You use a CNN classifier. Assume all activation function is ReLU and pooling is max pooling. For each layer, state the activation map dimensions (the dimension for the tensor in that layer) and calculate the number of parameters (both weights and biases). Here is the notation follows the convention:

CONV-K-N: denotes a convolutional layer with N filters, each with size K x K filter, with padding =0 and stride = 1.

POOL-K: denotes a K x K pooling layer with stride K and padding 0

FC-N: denotes a fully connected layer with N neurons.

|  |  |  |
| --- | --- | --- |
| Layer | Activation map dimensions | Number of Parameters |
| INPUT | 256 x 256 x 3 | 0 |
| CONV-17-16 | 240 x 240 x 16 | 13888 |
| POOL-2 | 120x120x16 | 0 |
| CONV-11-32 | 110x110x32 | 61984 |
| POOL-2 | 55x55x32 | 0 |
| FC-10 | 55x55x32 | 968010 |

1. Write the model in python for the above model.

model = models.Sequential([

layers.Conv2D(16, (17, 17), activation='relu', input\_shape=(256, 256, 3)),

# Max pooling layer with a 2x2 pool size

layers.MaxPooling2D((2, 2)),

# Convolutional layer with 32 filters, each with an 11x11 kernel

layers.Conv2D(32, (11, 11), activation='relu'),

# Max pooling layer with a 2x2 pool size

layers.MaxPooling2D((2, 2)),

# Flatten layer to convert 3D activation map to 1D

layers.Flatten(),

# Fully connected layer with 10 neurons

layers.Dense(10, activation='relu')

])