Introduction to Machine Learning

Problem Set 4

UT EID – oap338

1. Artificial Intelligence (8 points)
    a. Describe what is natural language processing and how it relates to machine learning.

       Natural Language processing is a special field in Artificial intelligence which deals with raw text data. It is called as Natural language because using this technique we communicate with the computer system using the language used by the humans. So, using Natural language processing we are building a computer system that can understands instructions given in human readable language.

       But to make a machine understand human language, we must teach that machine and allow it to learn about how to do this. And therefore, for this learning phase we are using machine learning for natural language processing. So, overall learning procedure involves, convert the given text instructions into a format which machine can understand (numeric format) so for this we can use different data modelling machine learning algorithms like bag of words, word2vector model etc. and then apply machine learning algorithms based on the required output task. Ex. Classification algorithms can be used to teach a machine how to classify the given text data.

    b. Describe what is computer vision and how it relates to machine learning.

       Computer vision is based on two factors, first is to identify the object and second is to interpret it. So, I can imagine, as NLP is used for textual data, computer vision mainly deals with images, videos, digital conversions. So, computer vision mainly deals with building a machine that can read images, videos. Which can identify objects present in the given image and even further make some interpretations based on the detected features.

       As machine or machine learning algorithms can't understand the visual effects directly, we must convert these images into some quantitative measures like pixel values. This is called as image processing. So, what I understood is, computer vision is a two-stage task i.e. image processing and machine learning together forms computer vision. Here, image processing is used for feature extraction i.e. creating new set of input features for the further analysis. And in next stage we can implement machine learning algorithm to build a model using these extracted features. So, machine learning is used in computer vision to teach a machine about how to read and understand visual effects/ images by building a model.

c. Describe what are low-level, mid-level, and high-level vision features and give two examples for each (6 examples in total).

   Low level features – This includes basic outlined and core features of an image. These features can be color of an image, edges of an object present in that image or you can say number of lines present in that image.

   Mid-level features – This kind of features can be used to connect low level and high-level features. Features like shape of an object, identifying an object from the shape of its shadow.

   High level features – High level features are the features which are easily identified or interpreted by humans. So, unlike standard feature extraction algorithms like HOG feature extraction algorithm, neural networks use several hidden layers to understand high level features present in an image. So, algorithms basically identify the high-level features to understand actions or what exact objects are present in an image. Features like classification of human actions, face detection, object detection can be a high level features.

2. Artificial Neurons (10 points)
   a. Describe what is Perceptron and Adaline and discuss what makes them similar as well as what makes them different.

      Perceptron – It is a supervised learning approach. Mainly used for classification type of problem. It is first artificially created neuron with a single layer network. It uses probabilistic approach to solve the classification problems. Just like an actual neuron, it has one activation function, electronic signals travels through a single layer network through dendrites and based on the value output of an activation function it decides to what output signal it will transfer to output layer neuron. So, in this case if perceptron received a correct output given weights to the connections remains unchanged. But if cost function calculates any error then it backpropagates back into the network and adjust the weights to reduce the value of error function.

      Adaline – As perceptron output function can predict either 1 or -1 based on the learning function/ activation function. It is a clear case of linear function. But we cannot assume that every time we are going to deal with linear function only.

So, Adaline algorithm is used to tackle linearity assumption in every case. So, as Adaline works on continuous values to predict, rather than just giving binary output as 1 or 0 or -1 it defines what is the probability of an output which is correct.

Similarities –

- Both are used as classifiers for binary (yes/No) classification
- Both are based on assumption of linear boundary
- Both are single layered network

Differences –

- Adaline – use Batch gradient descent – update on batches
  Perceptron – use Stochastic gradient descent – update after every training sample
- Adaline uses continuous predicted values.

3. Gradient Descent (7 points)
    a. Describe what is stochastic gradient descent, batch gradient descent, and mini-batch gradient descent and identify what makes them different from each other.

    Gradient Descent is a searching strategy. In case of neural networks, it is used in error backpropagation algorithms used to redefine the weights to minimize the error reported by the cost function. Different variations of gradient descent implementation strategy available -

    Stochastic gradient descent – In this case, cost function is calculated, and weights are redefined at neural network layer levels of each training input sample. In this way it can train large datasets and each iteration for each training dataset sample is faster in execution. As it keeps on updating the gradient value for each input sample, results in unusual gradient bounce. Stochastic gradient decent is used by Perceptron algorithm.

    Batch gradient descent – Unlike Stochastic gradient descent which backpropagates the error by calculating cost function after completing cycle for every training sample, batch gradient descent works on batches of training samples. So, first the large given dataset is divided into number of batches. And then after processing the whole batch of training samples, it uses the cost function calculations to backpropagate the error to adjust the weight to reduce the value of a cost function. As it cumulatively works on all the training samples from a whole batch together, it is very slow and even not recommended when given dataset is large. But as it works batch by batch, unlike Stochastic gradient descent, it avoids unusual gradient bounce.

Mini-batch gradient descent – Mini-batch gradient descent also works on subset of training samples for each update. But in this case, batches are not to big (as a smaller number of training samples per batch compare to number of samples in a batch for a case of batch gradient descent). In this case gradient bounces less erratically as compare to it bounces in case of stochastic gradient descent. And as batches are smaller, I think it will take less time and can be more computationally more efficient as compare to batch gradient descent. So, unlike batch gradient descent, we can implement mini-batch gradient descent on large datasets just by storing some observation in memory.

b. What gradient descent approach does Perceptron use?

   Perceptron uses Stochastic Gradient Descent. So, in this case, cost function is calculated, and weights are redefined every time for each training sample. It is vulnerable to unusual gradient jumps.

c. What gradient descent approach does Adaline use?

   Adaline uses Batch Gradient Descent strategy. So, in this case, first the whole dataset is divided into number of batches. And cost function is calculated, and weights are redefined only after the whole batch has been processed. It avoids unusual gradient jumps.