ECE 156B/594BB Homework 2: Neural Networks

Instuctor: Li.C Wang, TA: Jenny Zeng,
University of California Santa Barbara

Due by midnight on Wednesday, Jan. 30th 2019

1 Introduction

In homework 1, you were able to achieve over 80% accuracy based on cross-validation on the training set, simply by calling the built-in modules from the Python scikit-learn library [2]. In fact, feature values in the pokémon data set follow several correlated Gaussain distributions. However, it is more difficult for the machine learning algorithms to capture the underlying data distribution in more complicated applications. In this assignment, you are given a larger data set from such application, and your job, similar to homework 1, is to wisely choose or design a machine learning model to get the best classification accuracy.

2 Data set

*** Make sure you download the latest version of data sets on Piazza.**

Download the training data set: **train_data.npy**. The training data is stored in a 50000-by-3027 matrix as numpy array. Labels (**train_label.npy**) for the training data are stored in another numpy array with 50000 integer values. We provide you sufficient samples, but you can decide the exact amount of training data to use. There are 10 types of labels. Each of the integer 0 to 9 corresponds to a certain class.
Download the unlabeled test set: **test_data.npy**. It contains 10000 test samples each with 3027 'feature values'.

3 Implementaton & Submission

For this assignment, we publish a CodaLab competition (https://competitions.codalab.org/competitions/21455?secret_key=fa96abb4-615a-4824-a7d9-46344afe000f), where you can submit your code and get evaluation scores that will be listed on a leader-board. **The score is determined by how many correct labels you get on the test set.**
If you are not familiar with CodaLab Competitions, check out here: participating in a competition. Register a CodaLab Competitions account and log into CodaLab Compectitions. After your submission finishes running, please choose to submit it to the leader-board. You can use any machine learning toolkits and any resources from the internet.
You must submit **one zip file** via the corresponding submission page. Your results must be in ***.npy*** format.
Submission format: The final submission format to CodaLab should be:
prediction.zip

- prediction.npy: this file should have the same format *train_label.npy*, but with shape (10000,). You can download an example *prediction.npy* file from Piazza to see the format.

Note that to create a valid submission, directly zip *prediction.npy* in its directory. **DO NOT zip a folder containing the file. (THIS IS VERY IMPORTANT!)**

4 Grading Criteria

The grading depends on your score on CodaLab Competition:

- If you achieve more than 80% score on the leader-board, you don't need to do anything. Just submit the report.
  - **Hint: one way to achieve this is to use deep neural networks** [4].
  - See [5] [1] [3] for GPU resources.
- If your score is under 80%, you can only get partial credits depending on your answers to the following questions along with the report: ***Ignore this if score over 80%***.
  1. Same as homework 1, train the following classifiers: Nearest Neighbors (k=3), Linear SVM, Gaussian Naïve Bayes, Neural Net, Decision Tree (depth=5) and QDA from sklearn library with the first 100, 1000, 2000, 5000 training samples, respectively. In a table, illustrate the cross-validation accuracy with respect to different sizes of training samples. What do you observe? Can you train all 50000 samples using this approach? Why?
  2. In class, we discussed the importance of perceptron. How SVM and Neural Networks evolve from the perceptron model? What's the difference between the two algorithms in the view of 'primal' and 'dual' forms?
  3. Solve the problem in Appendix. Provide reasons to all your answers.
  4. In class, we talked about SVM and Multi-layer perceptrons (MLPs) as dominant machine learning algorithms in the past. They also suffered some 'dark age'. Yet, what enables recent upsurge of deep learning (essentially more layers of MLP)?
  5. In terms of domain knowledge, what makes the data set in this assignment different from the Pokémon one? Do you know what are the features in this data set? Why this makes it more difficult to choose the correct algorithm to apply?
  6. What is the range of values in the data set? Can you guess what kind of data it is in the data set? What does this imply on your algorithm choice?
  7. In a paragraph, describe what is a convolution neural network (CNN). Why it is special about 'convolution'?
  8. From all your answers above, in a paragraph summarize what could you do to improve your score on CodaLab?

5 Report

In addition to the result submission on CodaLab Competitions, you are also supposed to report your methodology. In the report, you should:

- Specify your **username and score** on CodaLab competition. **No-show on the leader-board directly translates to zero score**. (10%)
- In a paragraph, introduce the problem. (10%)
- Elaborate your approach. Explain the algorithm you use to get the final score. (70%)
  - ***Ignore this if score over 80%***. Answer questions in 4.

- Conclude with challenges you encounter and what are the possible solutions. (10%)

6 Turn-in Requirements

Please include the following items in a **single** zip folder.

- All code scripts.
- Report.

Name your folder as **ECE156B_HW1_YourName** or **ECE594BB_HW1_YourName**.
Then submit a **single** attachment to: yuelingzeng@ucsb.edu

References

[1] Amazon web services. `https://aws.amazon.com/machine-learning/amis/?hp=tile&so-exp=below`.

[2] Classifiers. `https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html`.

[3] Google clouds. `https://cloud.google.com/gpu/`.

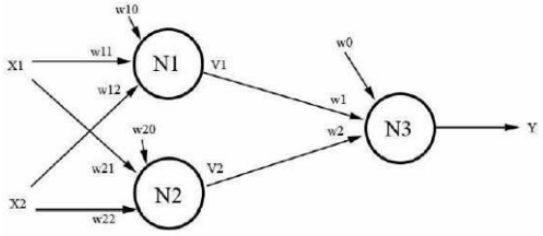
[4] tensorflow with keras api. `https://www.tensorflow.org/guide/keras`.

[5] Ucsb center for scientific computing. `https://csc.cnsi.ucsb.edu/docs/using-gpu-nodes`.

Appendices

See Figure 1.

Here is a simple 2-layer neural network with 2 hidden units and a single output unit.



Consider the linear activation function $y = C \cdot a = C \cdot \sum_i w_i x_i$ where C is a constant multiplied by a which is the weighted sum of its inputs. Also, consider the non-linear logistic activation function $y = \sigma(a)$ where

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$

1. (a) Assume all units are linear. Can this 2-layer network represent decision boundaries that a standard regression model $y = b_0 + b_1 x_1 + b_2 x_2 + \epsilon$ cannot?

   (b) Assume the hidden units use *logistic* activation functions and the output unit uses a *linear* activation unit. Can this network represent non-linear decision boundaries?

   (c) Using *logistic* activation functions for both hidden and output units, it is possible to approximate any complicated decision surface by combining many piecewise linear decision boundaries. Explain what changes you would need to make to the above network so you could approximate any decision boundary.

Figure 1: Neural Network