UNIVERSITY OF MARYLAND, BALTIMORE COUNTY

In-execution classifier visualization

Prepared for: Jian Chen, Assistant Professor in CSEE @ UMBC

Prepared by: Sushant Athley, Sushant Chaudhari, Harsh Vashishtha

October 16, 2016

Team name: “VizGurus”

EXECUTIVE SUMMARY

Motivations

Machine learning algorithms have made great strides in the last decade. We are at a very good place and the future looks very promising with exciting applications such as chat bots and self driving cars on the horizon. As students who has been recently introduced to such classes of algorithms, we have observed certain themes underlying typical supervised classification techniques, which make up the motivation behind this project.

Machine learning algorithms are getting more and more accurate but at the cost of increasing complexities. These complexities come from their sheer size and computational depth. But with recent advancements in offloading critical computations to GPUs, such large scale techniques are becoming trivial everyday. These algorithms work in ‘magical’ ways to learn weights & biases for their internal structures (perceptrons, decision nodes etc.) based on the training data they see. They are then able to classify new incoming data with great precision.

But there is still some amount of hit-and-trial and guesswork involved when creating classifiers. The primary reason behind this is the many varieties of classification techniques available. Once a particular technique has been decided upon, there are still lot of knobs/variables (number of layers, activation functions, initial weights and other hyper parameters) that need to be tuned and tweaked before you start to see any decent accuracy. These algorithms perform tens or hundreds of computations for every training data item. This makes the entire lifecycle slow and requires experience and patience.

Goals & Research Questions

We would like to present an analogy between baking a cake and creating a classifier. Just like when baking a cake, you can taste the amalgamation of ingredients during the process, and also eyeball the unfinished cake during its several recipe stages, we would also like to reduce the guesswork as much as possible from building

classifiers. This would improve efficiency and confidence and there is a higher chance of ending up with a tastier cake. Visualization would be our eyes, ears and taste buds for this exercise.

This project aims to come up with a visual representation of the classifier while it is being built on a per iteration basis. This will allow researchers and end users to better understand the intricacies of these algorithms. A per iteration approach breaks down the learning aspect of the algorithm into its most primitive unit. The user can, after every iteration, get a sneak peak at the inner workings of the classifier.

Proposed Methods

One of the most critical components of building classifiers is the Gradient Descent algorithm. We will visualize the input space of all possible weight vectors and trace the path which the gradient descent algorithm takes in determining the minima. This tracing will give deeper understanding to users about their choice of hyper parameters.

Literature Review

Gradient Descent is a very well studied algorithm for minimizing functions. Sebastian Ruder from AYLEIN has done an excellent job at summarizing the most popular gradient descent techniques [1], which will provide us with a good starting point for this project.

Scikit-learn [2] and TensorFlow [3] are popular open source machine learning tools available for easily implementing gradient descent in any classification or regression problem.

Wikipedia has summarized the work around Gradient Descent well with certain key examples [4]. Although the visualizations presented so far are good and intuitive, but they do not address the problem at the time of execution. They provide a delayed feedback. We intend to bridge this gap with an in-execution animated visualizer of gradient descent.

The authors in [5] have provided an in-depth study of visualizing multi dimensional hyper surfaces with an interest in minimizing or maximizing them. Fig. 1 in [5] where a hybrid approach using contours/3d point plot is presented, This shall serve as an inspiration for the team in designing the solution for this term project.

References

1. http://sebastianruder.com/optimizing-gradient-descent/
2. http://scikit-learn.org/stable/modules/sgd.html#
3. https://www.tensorflow.org/versions/r0.11/api_docs/python/train.html#GradientDescentOptimizer

4. https://en.wikipedia.org/wiki/Gradient_descent

5. P.A. Simionescu, D. Beale: Visualization of hypersurfaces and multivariable functions