Learning to Reweight Examples for Robust Deep Learning

Manish Kumar Krishne Gowda, Purdue University, ECE Department, On Campus MS Student[1]

Abstract

Deep Neural Networks easily overfit to training set biases and label noises.Regularizers and example reweighting algorithms are popular solutions to these problems. But these algorithms require careful tuning of additional hyperparameters. Traditionally, validation is performed at the end of training, which can be prohibitively expensive if the example weights are treated as some hyperparameters to optimize. To circumvent this, the paper proposes to perform validation at every training iteration to dynamically determine the example weights of the current batch. This approach significantly increases the robustness to training set biases.

1. Introduction

The paper proposes a learning algorithm that learns to assign weights to training examples based on their gradient directions. They perform gradient descent on the mini-batch example weights which are initialized to zero. Thereby they minimize the loss on clean unbiased validation set.

The model in the paper is derived from a meta-learning objective towards an online approximation that can fit into any regular supervised training. Instead of minimizing the expected loss for the training set each input example is weighted equally and this reweighted input is used to model the dataset. There are multiple examples of automatic differentiation techniques which will be the correct starting point as it is needed to compute the gradient of the validation loss. The paper itself for example references a couple of other papers which implements it using popular deep learning frameworks such as TensorFlow.

1.1. Unfamiliar Concepts

As of now I have limited knowledge of the concept of deep neural networks and its implementation in software. This paper further narrows down the deep learning problem to batch learning which needs to be practically implemented. The paper itself was read in haste and hence a thorough reading to understand the algorithm mentioned and its translation to a practical dataset needs to be realized.

2. Issues with Implementation

The core of the paper is the Algorithm 1 Learning to Reweight Examples using Automatic Differentiation in page 4. I am trying to understand the different terminologies in the paper like Deep neural networks, hyperparameters, metaa-learning, MNIST and CIFAR benchmarks etc etc. With these I should be able to replicate one of the two experiments mentioned in the paper i.e. MNIST data imbalance experiments or CIFAR noisy label experiments. Both are computationally expensive so, trying on my local system seems infeasible.

With deep neural network to be in agenda of discussion in class next week i hope to get a few questions cleared and get some hints on the algorithm implementation during office hours.

Some information on the MNIST data imbalance experiment implementation can be found at `shorturl.at/iprM7`. The results are well replicated and provide some confidence for implementation.

References

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

A. Do *not* have an appendix here

Do not put content after the references. Put anything that you might normally include after the references in a separate supplementary file.

We recommend that you build supplementary material in a separate document. If you must create one PDF and cut it up, please be careful to use a tool that doesn't alter the margins, and that doesn't aggressively rewrite the PDF file. pdftk usually works fine.

Please do not use Apple's preview to cut off supplementary material. In previous years it has altered margins, and created headaches at the camera-ready stage.