Optimisation with Progressive Sharpening in Deep Neural Networks

Sahil Trivedi

September 2021

1 Abstract

Introducing Progressive Sharpening, a method for optimising convolutional and fully connected layers in neural networks. This is achieved by creating incrementally denoised (blurred) copies of the training data. Training is initialised with the most denoised samples, and every few epochs we progressively sharpen the samples until the network converges on the orginal training data. We discuss the mathematical intuition of our technique and benchmark it against SoTA models on image recognition datasets.

2 Introduction

Gradient descent optimises the weights of a network by minimising the error for it's neurons' activation values. A common tradeoff associated with gradient descent is it's tendency to converge training around local minima. Despite extensive research in the area of learning rate optimisation, avoiding local minima remains a recurring challenge in deep learning.

During denoising, high level features are preserved while obscure features are smoothened out of the data. This is leveraged early on during training as we want the network paramters to distinguish high level features and avoid lower level features associated with local minima. As training progresses and high level features are generalised, we proceed to increase the level of detail in our training data making the network focus on more obscure features towards the end of training.

It is well established that vision in human infants is blurred and improves it's focus as time passes. We discuss the mathematical intuition behind progressive denoising in Section 6 of this paper.

3 Network Architecture

Progressive Sharpening can potentially increase the accuracy of a model from moderately high to extremely high. For this to work we take a model with an acceptable accuracy/loss and make a few design changes based on our understanding of the problem domain along with some trial and error.

In the context of Image Classification on the MNIST dataset, LeNet-5 is a popular architecture with consistently high performance. We modify LeNet-5 by adding dropout to it's dense layers. The model summary is given below.

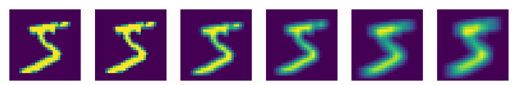
Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 24, 24, 6)	156
average_pooling2d (AveragePo	(None, 12, 12, 6)	0
conv2d_1 (Conv2D)	(None, 8, 8, 16)	2416
average_pooling2d_1 (Average	(None, 4, 4, 16)	0
flatten (Flatten)	(None, 256)	0
dense (Dense)	(None, 120)	30840
dropout (Dropout)	(None, 120)	0
dense_1 (Dense)	(None, 84)	10164
dropout_1 (Dropout)	(None, 84)	0
dense_2 (Dense)	(None, 10)	850

Total params: 44,426
Trainable params: 44,426
Non-trainable params: 0

4 Training

The specifics for implementing Progressive Sharpening largely depend on the problem domain. This paper implements the technique in context of Image Classification, but the principle extends for classifying information with any dimensional encoding.

4.1 Denoising Training Samples



The figure illustrates an original sample on the extreme left followed by it's progressively denoised copies. The original image is from the MNIST dataset and the denoising is carried out using low pass filters. The kernels for these filters are represented by the following matrices:

$$kernel_N = \frac{J_N}{N^2}$$

where J_N is an $N \times N$ matrix with all elements equal to 1.

Since $kernel_1$ denotes an identity transform, the first two images are identical and the latter four are progressively denoised. We apply the said convolutions to the training dataset which returns denoised copies of the training data. These copies are passed to the Convolutional Neural Network while training in a predetermined order.

4.2 Training Set Optimisation

The multiple sets of training data obtained from the previous step are referred to as $train_1, train_2, \ldots, train_K$ where

$$train_n = training\_data \circledast kernel_n$$

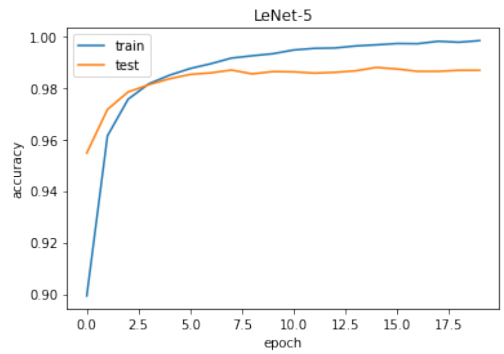
and K is equal to the denoising intervals picked for the data. $\circledast$ is a convolutional operator and the kernel denotes a low-pass filter for the training data.

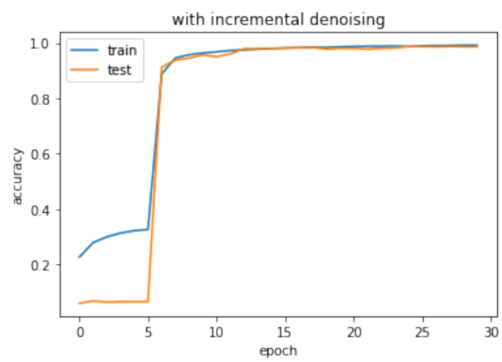
The higher the value of n, the more denoised the data. We begin training with the most denoised copies of the training data. After few epochs we replace the training data with it's next most denoised copy. This process repeats K times before network loss converges for the least denoised samples i.e. the original training data itself.

5 Benchmarking

We compare the performance of our model with the LeNet-5 architecture introduced in the 1999 paper "Object Recognition with Gradient Based Learning" by LeCun et al.

By comparing the training plots of both models we notice areas where our model performs better than LeNet-5.





6 Mathematical Intuition

Gradient descent optimises the weights of a network by minimising the error in it's neurons' activation values. A common tradeoff associated with gradient descent is it's tendency to converge training around local minima. Learning Rate Op-

timisation is generally employed to address this.
- Visualising W vs E hypersurface

7 Future Scope

- Shuffling Denoised Samples with the original samples while training. - Progressive Denoising in Time Variant Signals - Similar Techniques and Approaches