Multiorder Regression using TensorFlow with Tensorboard

Akshay Arora

June 9, 2017

1 Introduction

This project shows a polynomial regression using gradient descent optimization algorithm.

2 Methodology

2.1 Placeholders in Tensorflow

Tensorflow provides a placeholder method which can be used to feed with data which we need to train. We create two placeholders which reserve place for x and y inputs with size 32 bit of data type float

2.2 Tensorflow Variables

We then initialize tensorflow variables for our weights and biases by randomly choosing weights from a truncated normal distribution.

A truncated normal distribution follow a normal distribution with a specific mean and standard deviation but those values which fall beyond 2 standard deviations are dropped. This helps in avoiding very large weights which in turn helps in gradient descent algorithm .

2.3 Loss Calculation

Loss is calculated by minimizing the objective function or the cost function. The objective function in this case is given by the following equation below :

Loss= $[PredicitedY - ExpectedY]^2$

Our main goal is to minimize this function to reduce the error as much as possible in our prediction by using a gradient descent optimization algorithm which is explained below.

2.4 L2 (Ridge) Regularization

L2 Regularization technique is used to prevent the learning model from over-fitting. It is expressed as follows :

$$Loss_{new} = Loss + \beta * \frac{1}{2} * \|W\|_2^2$$

Here, as we can see L2-norm term is added to our original loss to penalize large weights flowing during gradient descent

2.5 Gradient Descent

Gradient descent is a way to minimize the objective function J(θ) . This is done by updating the parameters in the opposite directions of the gradient of the objective function.

In this project, I have used the stochastic gradient descent method with an Adam (Adaptive momentum) optimizer published by Kingma.This method performs a parameter update for each training example for xs and ys.

The reason behind choosing Adam optimizer is that adam optimizer is because it is an improved version of AdaDelta and RMSProp algorithm as it also calculates the adaptive learning rates for

each parameter update , with addition to this it stores the exponentially decaying past values of the gradients of moments. This it is called Adaptive Momentum .It converges to local minimum with reduced oscillations. The Adam optimizer uses the following steps in gradient descent:
$lr_t <= \eta * \sqrt{(1-\beta_2{}^t)}/(1-\beta_1{}^t)$
$m_t <= \beta_1 * m_{t-1} + (1-\beta 1) * g$
$v_t <= \beta_2 * v_{t-1} + (1-\beta_2) * g^2$
$\theta <= \theta - lr_t * m_t/(\sqrt{(v_t)} + \epsilon)$

Here, m_t and v_t are the estimates of the first moment(mean) and second moment (variance) .

η is the learning rate which is defined as the size of the steps required to reach the local minimum.

Also,$\beta 1$ and $\beta 2$ are the exponential decay rates for the 1st and 2nd moment estimates respectively and ϵ is a small constant for numerical stability. The typical values for η,$\beta 1$,$\beta 2$ and ϵ are 0.0001, 0.9, 0.999, 1e-8 respectively.

2.6 Tensorflow Session

The Tensorflow session is basically a graph that is used to define computation. It doesn't perform any computational operations , it only arranges these operations in the graph. It is used for allocating resources for various results.

3 Tensorboard

Tensorboard is a tensorflow estimator and has been recently widely popular for it's usage in creating a graph of tensors from the code.We can see in the figure below how the tensors flow through the graph.Also, the details of the train block.

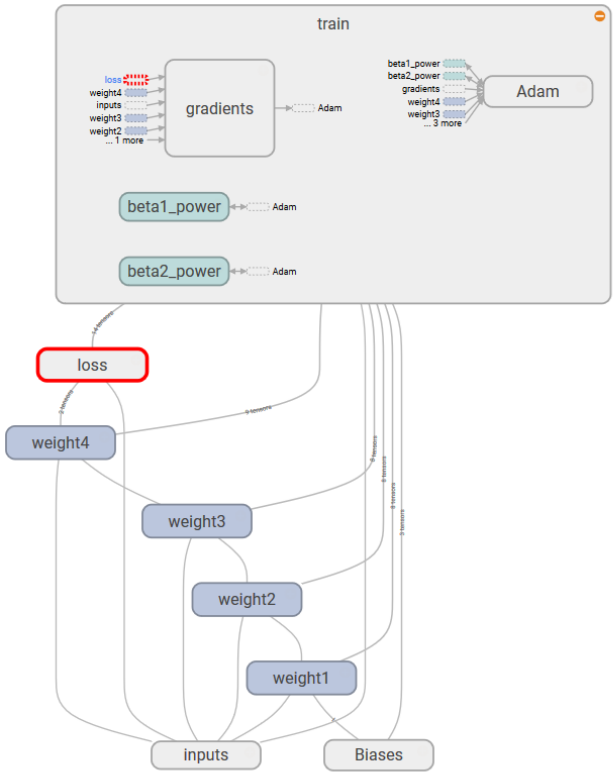

Figure 1: Tensorboard Graph

3.1 Tensorboard Scalar

We can use the "tf.summary.scalar()" command to see the plots of scalar values like the loss over the number of epochs.

We can observe here that the loss is reducing with the increase in number of epochs and stays constant and reaches it's minimum value as 0.000791

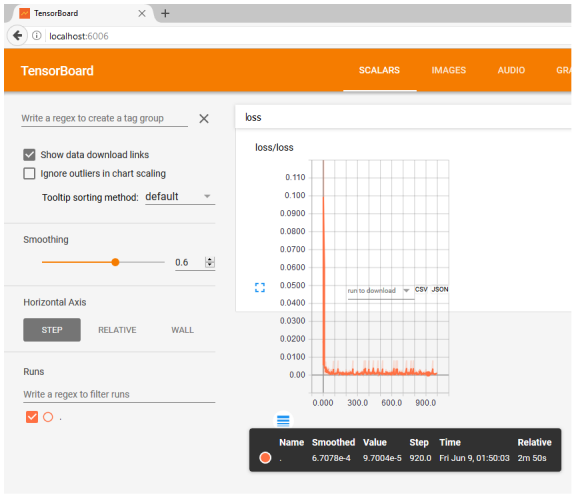

Figure 2: Tensorboard Scalar Loss Plot.

3.2 Tensorboard Histogram

The figure below shows the weights of the histogram

Figure 3: Tensorboard Histogram of weights.

3.3 Tensorboard Distributions

The figure below shows the distibution of these weights

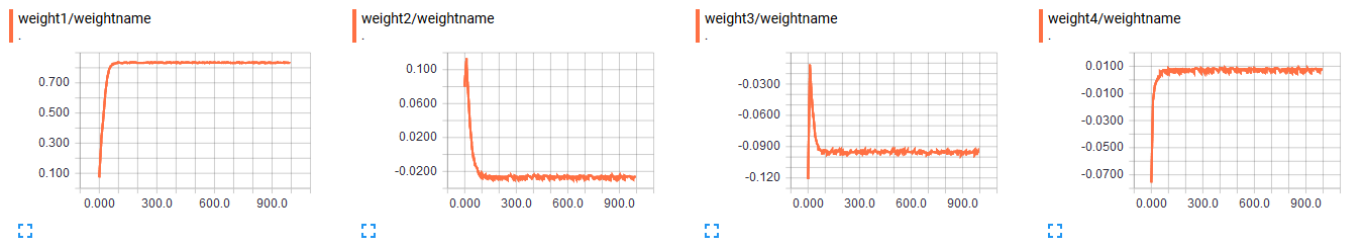

Figure 4: Tensorboard Distribution of weights.

4 Conclusion

Thus, we have successfully implemented a polynomial regression in tensorflow and visualized our network using tensorboard.