Week2 - Parameters Tuning and DL Frameworks

Sonntag, 19. Juli 2020 10:11

Parameters Tuning Process

Choose random uniformed values to tune parameters

This method is useful when the two ends of range are on different scales.

Batch Normalization

Speed up learning process by normalize the 'input' of each layer -> normalize z()

$$M = \frac{1}{m} \sum_{i=1}^{m} \frac{1}{2} z^{(i)}$$

$$R = \frac{1}{m} \sum_{i=1}^{m} z^{(i)$$

Batch norm has also a slight regularization effect.

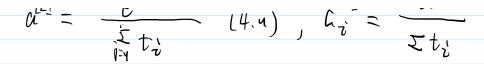
Batch norm at test time: using exponential weighted average through all minibatches to estimate $\frac{1}{2}$ for test time

Multiple classification

Softmax layer as the last layer

CutiVation:
$$t = e^{(\frac{1}{2}i\lambda)}$$

$$c^{i\lambda} = \frac{e^{2i\lambda}(1+\mu)}{\sum_{i=1}^{2} t_{i}} (4, \mu), \quad c^{i\lambda} = \frac{t_{i}}{\sum_{i=1}^{2} t_{i}}$$



Take away from programming assignment:

TensorFlow 1: general idea of runing a TensorFlow pragram:

Create Tensors (variables) that are not yet executed/evaluated.

Write operations between those Tensors.

Initialize your Tensors.

Create a Session.

Run the Session. This will run the operations you'd written above.

Note that, a-d have only created the calculation graph, step e runs it.

A placeholder is an object whose value you can specify only later. To specify values for a placeholder, you can pass in values by using a "feed dictionary" (feed_dict variable).

for multi-class:

LINEAR -> RELU -> LINEAR -> RELU -> LINEAR -> SOFTMAX. The SIGMOID output layer has been converted to a SOFTMAX. A SOFTMAX layer generalizes SIGMOID to when there are more than two classes. for the tf.reduce_mean() to calculate the cost: It is important to know that the "logits" and "labels" inputs of tf.nn.softmax_cross_entropy_with_logits are expected to be of shape (number of examples, num_classes).