Mathematical proof

Zhenghao Peng

April 22, 2018

1 Problem

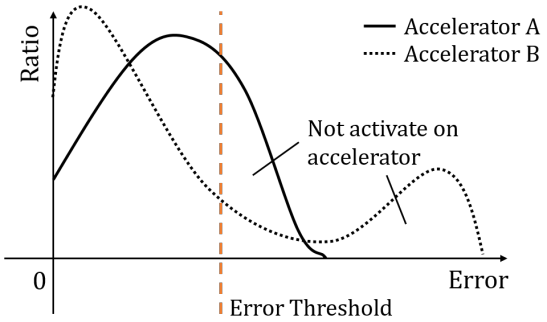

Figure 1: X-axis: the error of approximation value and ground truth. Y-axis: the proportion of data. The solid line: the distribution data before iterative training. The dotted line: the distribution data after iterative training.

Using a neural network (called **approximator**) to regress a function, here we get the solid line in the above figure. It's quite obvious that most of the samples in testing data are located in the medium position, in terms of the error between the output value of NN and the ground truth.

Now, if we set a error threshold (a scalar et), and manually select the samples (x, y) which satisfy:

$$error(f(x) - y) < et \tag{1}$$

wherein $error$ is the error metric, and f is the neural network, y is the ground truth.

Then we found a new dataset $\mathcal{X}$ which only contains the samples that satisfy the above criterion. Using $\mathcal{X}$ to fine-tune the neural network, we get the dotted line in figure.

It's intuitive to get this line: The polarization happens. Some samples shot less error while other shot greater. We can intuitively interpret this phenomena as "The neural network fits (even slightly over-fits) the dataset $\mathcal{X}$ thus performs worse in those samples that are excluded from $\mathcal{X}$".

But we want a mathematical proof.

2 Notation

Here's some notations.

iteration 0: trained approximator by all training data.

h^i: The output value of the model in iteration i. Note that h^0 present the output vector of approximator in iteration 0.

$p(x)$: The probability of x, which would not varies w.r.t. iterations.

$y(x)$: The accurate value w.r.t. x. Sometimes we may simply write as y.

$\mathcal{D}$: The definition domain of all input vector.

$\mathcal{D}^1$: The definition domain of input vector in iteration 1. Note that, apparently, $\mathcal{D}^1 \subseteq \mathcal{D}$

$\widetilde{\mathcal{D}}$: Definition domain that not included in $\mathcal{D}^1$ but in $\mathcal{D}$, namely $C_{\mathcal{D}}\mathcal{D}^1$, the complementary set of $\mathcal{D}^1$.

$\boldsymbol{X^1}$: The probability distribution of input vector in iteration 1.

$\widetilde{\boldsymbol{X}}$: The probability distribution of x defined in $\widetilde{\mathcal{D}}$.

err: The error bound.

$\mathcal{N}(y; h(x); \sigma^2)$: The probability $P(\boldsymbol{y} = y)$ in which $\boldsymbol{y}$ is subject to a normal distribution with mean $h(x)$ and fixed variance σ^2.

3 Assumption of perfect classifier

We assume the classifier can perfectly discriminate the input data which satisfy the error bound and which not.

Since those x that cause approximator produces out-of-bound result will be discard in $\mathcal{D}^1$, so in $\widetilde{\mathcal{D}}$ the absolute error between y and h is greater than err (of course consider relative error is the same):

$$\forall x \in \widetilde{\mathcal{D}},\ |h^0(x) - y(x)| > err$$

Therefore, only take $\widetilde{\mathcal{D}}$ into consideration, we can derive:

$$\underset{x \sim \widetilde{\mathcal{D}}}{E} ||h^0(x) - y(x)|| > K \times err \tag{2}$$

in which $K = \int_{x \in \widetilde{\mathcal{D}}} p(x)dx$.

4 Assumption of perfect approximator

We assume the approximator can perfectly minimize the loss function to a minimal.

We assert, due to the limit scale of the approximator neural network, it can not minimize the loss function to zero, which means it can not perfectly fit the accurate curve of target value.

In the updating phase of approximator in iteration 1, we know that the updating equation is:

$$\boldsymbol{\theta} = \underset{\boldsymbol{\theta}}{argmin} \underset{x\sim \boldsymbol{X^1}}{E} -log\ p(y|x;\theta)$$

If we set $p(y|x)$ to be subject to a normal distribution $\mathcal{N}(y; h(x); \sigma^2)$, in which $h^1(x)$ denotes the mean and σ^2 denotes the variance, which is fixed.

Then the cross entropy (which is the minimization target) can be rewritten as:

$$\underset{x\sim \boldsymbol{X^1}}{E} -log\ p(y|x;\theta) = log\sigma + \frac{log(2\pi)}{2} + \underset{x\sim \boldsymbol{X^1}}{E} \frac{||h^0(x) - y||^2}{2\sigma^2}$$

That means, supposed the minimization process is effective, then of course the cross entropy is declined, and since the first two terms do not correlate with the optimization process:

$$\underset{x\sim \boldsymbol{X^1}}{E} ||h^0(x) - y||^2 \geq \underset{x\sim \boldsymbol{X^1}}{E} ||h^1(x) - y||^2$$

Apparently, this is the square of Euclidean norm. According to the equivalence of norms, the descending of L1 norm is consequently proved. Note that if both sides are multipled by $\int_{x\in\mathcal{D}^1} p(x)dx$, the expectation scope transfers to a part of the expectation of the whole definition domain of x.

$$\underset{x\in\mathcal{D}^1}{E} ||h^0 - y|| \geq \underset{x\in\mathcal{D}^1}{E} ||h^i - y|| \tag{3}$$

This inequality show that the training process of A can decrease the absolute error between the output of neural network and the accurate value.

5 Invariance of minimality

Under the assumption of perfect approximator, in iteration 0 the approximator is sufficiently trained and thus the cross entropy is the minimal. Therefore no matter how to train it in next iteration i, only if the data set is not vary, the cross entropy of approximator provided the same data set $\mathcal{D}$ is still the minimal:

$$\underset{x\sim \boldsymbol{X}}{E} ||h^0(x) - y||^2 \leq \underset{x\sim \boldsymbol{X}}{E} ||h^1(x) - y||^2$$

Thanks for equivalence of norms, the L1 norm still hold above property:

$$\underset{x\sim \boldsymbol{X}}{E} ||h^0(x) - y|| \leq \underset{x\sim \boldsymbol{X}}{E} ||h^1(x) - y|| \tag{4}$$

6 Proof of ascending accuracy in $\hat{\boldsymbol{X}}^{\boldsymbol{i}}$

We want to show that, after a iteration, the absolute error is descending. Moreover, we give a lower bound of the improvement.

First, we simply decompose the expectation of absolute error before a iteration into two part.

$$\mathop{E}_{x\sim \boldsymbol{X}} ||h^0 - y|| = \mathop{E}_{x\in \mathcal{D}^1} ||h^0 - y|| + \mathop{E}_{x\sim \widetilde{\mathcal{D}}} ||h^0 - y|| \tag{5}$$

Second inequality shows after selecting data, even the aprroximator is not trained yet, the error is descending (Equation 2).

$$\mathop{E}_{x\in \mathcal{D}^1} ||h^0 - y|| + \mathop{E}_{x\sim \widetilde{\mathcal{D}}} ||h^0 - y|| \geq \mathop{E}_{x\in \mathcal{D}^1} ||h^0 - y|| + K \times err \tag{6}$$

The third inequality shows that, apparently, fine-tuning approximator can continuously lower the error(Equation 3).

$$\mathop{E}_{x\in \mathcal{D}^1} ||h^0 - y|| + K \times err \geq \mathop{E}_{x\in \mathcal{D}^1} ||h^1 - y|| + K \times err \tag{7}$$

As a reminder, $K = \int_{x\in \widetilde{\mathcal{D}}} p(x)dx$.

7 Proof of decending accuracy in $\widetilde{\mathcal{D}}$

If we simply decompose the expectation in equation 4 :

$$\mathop{E}_{x\sim \widetilde{\boldsymbol{X}}} ||h^1(x) - y|| + \mathop{E}_{x\sim \boldsymbol{X^1}} ||h^1(x) - y|| \geq \mathop{E}_{x\sim \widetilde{\boldsymbol{X}}} ||h^0(x) - y|| + \mathop{E}_{x\sim \boldsymbol{X^1}} ||h^0(x) - y||$$

Transposition:

$$\begin{aligned} &\mathop{E}_{x\sim \widetilde{\boldsymbol{X}}} ||h^1(x) - y|| - \mathop{E}_{x\sim \widetilde{\boldsymbol{X}}} ||h^0(x) - y|| \\ \geq &\mathop{E}_{x\sim \boldsymbol{X^1}} ||h^0(x) - y|| - \mathop{E}_{x\sim \boldsymbol{X^1}} ||h^1(x) - y|| \\ \geq &0 \end{aligned}$$

Combine equation 2 and inequality above, multiply both sides with K, we get:

$$\mathop{E}_{x\in \widetilde{\mathcal{D}}} ||h^1 - y|| \geq \mathop{E}_{x\in \widetilde{\mathcal{D}}} ||h^0 - y|| \geq K \times err \tag{8}$$

in which $K = \int_{x\in \widetilde{\mathcal{D}}} p(x)dx$.