How to interpret loss and accuracy for a machine learning model [closed]

Asked 8 years, 11 months ago Modified 1 year, 6 months ago

Viewed 299k times

279

Closed. This question does not meet Stack Overflow guidelines. It is not currently accepting answers.

This question does not appear to be about programming within the scope defined in the help center.

Closed 3 years ago.

Improve this question

When I trained my neural network with Theano or Tensorflow, they will report a variable called "loss" per epoch.

How should I interpret this variable? Higher loss is better or worse, or what does it mean for the final performance (accuracy) of my neural network?

machine-learning neural-network mathematical-optimization deep-learning objective-function

Share

Improve this question

Follow

edited Mar 28, 2021 at 11:44
desertnaut
60.2k ● 31 ● 151 ● 176

asked Dec 29, 2015 at 20:33
mamatv
3,661 ● 5 ● 21 ● 28

2 I'm voting to close this question because Machine learning (ML) theory questions are off-topic on Stack Overflow - gift-wrap candidate for Cross-Validated – Daniel F Feb 10, 2021 at 12:13

3 Answers

Sorted by: Highest score (default)

373

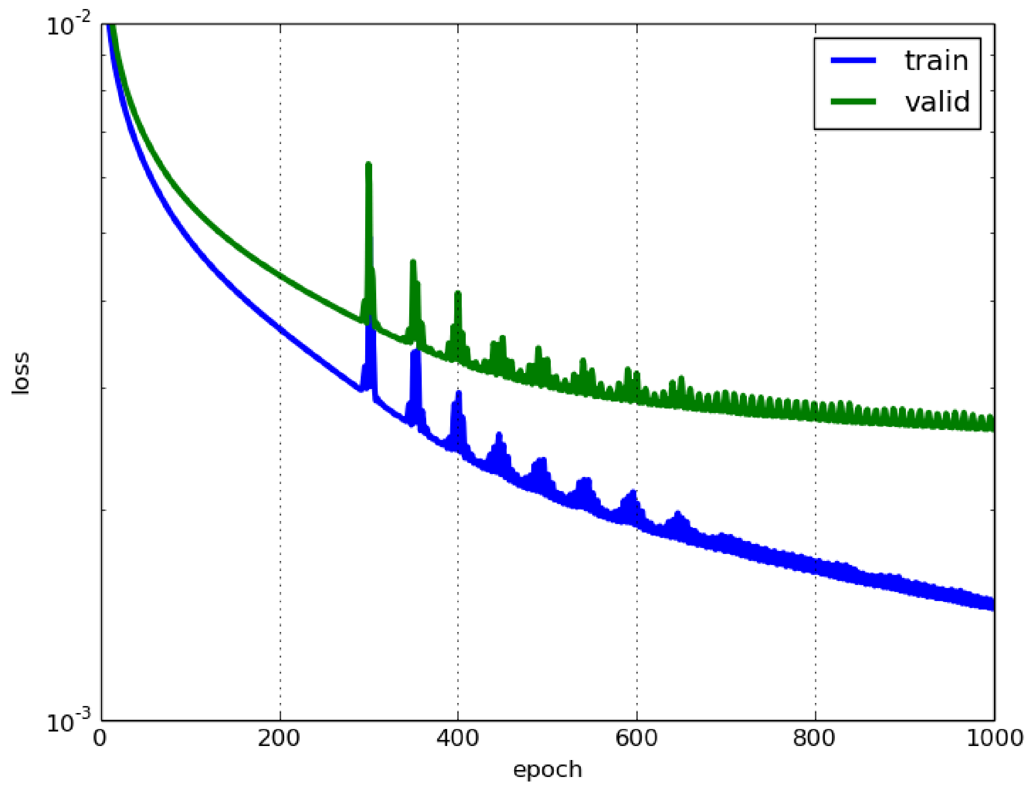
The lower the **loss,** the better a model (unless the model has over-fitted to the training data). The loss is calculated on **training** and **validation** and its interperation is how well the model is doing for these two sets. Unlike accuracy, loss is not a percentage. It is a summation of the errors made for each example in training or validation sets.

In the case of neural networks, the loss is usually negative log-likelihood and residual sum of squares for classification and regression respectively. Then naturally, the main objective in a learning model is to reduce (minimize) the loss function's value with respect to the model's parameters by changing the weight vector values through different optimization methods, such as backpropagation in neural networks.

Loss value implies how well or poorly a certain model behaves after each iteration of optimization. Ideally, one would expect the reduction of loss after each, or several, iteration(s).

The **accuracy** of a model is usually determined after the model parameters are learned and fixed and no learning is taking place. Then the test samples are fed to the model and the number of mistakes (zero-one loss) the model makes are recorded, after comparison to the true targets. Then the percentage of misclassification is calculated.

For example, if the number of test samples is 1000 and model classifies 952 of those correctly, then the model's accuracy is 95.2%.



There are also some subtleties while reducing the loss value. For instance, you may run into the problem of over-fitting in which the model "memorizes" the training examples and becomes kind of ineffective for the test set. Over-fitting also occurs in cases where you do not employ a regularization, you have a very complex model (the number of free parameters `W` is large) or the number of data points `N` is very low.

Share Improve this answer

Follow

edited Jul 7, 2020 at 16:13

VansFannel
45.9k ● 117 ● 373 ● 649

answered Dec 29, 2015 at 21:21

Amir
11.1k ● 10 ● 51 ● 75

2 Hi @Amir, thanks for your very details explanation. However, I have a problem: in my Neural Network, loss always decrease when I trained (when the *epochs* increase), however the accuracy is not better. – mamatv Jan 4, 2016 at 7:43

13 @mamatv As long as the cost is decreasing you should be good to go. Although cost and accuracy normally have a inverse proportionality relationship, but you may note that accuracy is a summation of zero-one errors whereas cost is a summation of floating point numbers. Therefore, 0.001% decrease in the cost does not necessarily mean 0.001% increase in the accuracy. Increasing the accuracy is much harder when the decrement in cost is intangible (the cost is very close to a local minima) – Amir Jan 4, 2016 at 17:34

3 @mamatv I should have said as long as the cost for both training and validation is decreasing you should be good to go. You may also check validation accuracy on each epoch. If it starts going up, then your model might have started to over-fit and you should stop training it. – Amir Jan 5, 2016 at 2:46

1 Why not train the model to increase accuracy rather than to minimise loss? – Bikash Gyawali Aug 28, 2018 at 15:25

7 @bikashg accuracy is not differentiable and therefore you can't backprop on it. – DharmaTurtle Feb 25, 2019 at 2:31

33

They are two different metrics to evaluate your model's performance usually being used in different phases.

Loss is often used in the training process to find the "best" parameter values for your model (e.g. weights in neural network). It is what you try to optimize in the training by updating weights.

Accuracy is more from an applied perspective. Once you find the optimized parameters above, you use this metrics to evaluate how accurate your model's prediction is compared to the true data.

Let us use a toy classification example. You want to predict gender from one's weight and height. You have 3 data, they are as follows:(0 stands for male, 1 stands for female)

y1 = 0, x1_w = 50kg, x2_h = 160cm;

y2 = 0, x2_w = 60kg, x2_h = 170cm;

y3 = 1, x3_w = 55kg, x3_h = 175cm;

You use a simple logistic regression model, that is

$$y = \frac{1}{(1 + e^{-(b1 \cdot x_w + b2 \cdot x_h)})}$$

How do you find b1 and b2? You define a loss first and use optimization method to minimize the loss in an iterative way by updating b1 and b2.

In our example, a typical loss for this binary classification problem can be: (a minus sign should be added in front of the summation sign)

$$\sum_{i=1}^{i=3} y_i \log(\widehat{y_i}) + (1 - y_i) log(1 - \widehat{y_i})$$

We don't know what b1 and b2 should be. Let us make a random guess say b1 = 0.1 and b2 = -0.03. Then what is our loss now?

$$\hat{y_1} = \frac{1}{1 + e^{-(0.1 \cdot 50 - 0.03 \cdot 160)}} = 0.549834 = 0.55$$

$$\hat{y_2} = \frac{1}{1 + e^{-(0.1 \cdot 60 - 0.03 \cdot 170)}} = 0.7109495 = 0.71$$

$$\hat{y_3} = \frac{1}{1 + e^{-(0.1 \cdot 55 - 0.03 \cdot 175)}} = 0.5621765 = 0.56$$

so the loss is

$$-\log(1 - 0.55) - \log(1 - 0.71) - \log(0.56) \simeq 2.6162$$

Then your learning algorithm (e.g. gradient descent) will find a way to update b1 and b2 to decrease the loss.

What if b1=0.1 and b2=-0.03 is the final b1 and b2 (output from gradient descent), what is the accuracy now?

Let's assume if y_hat >= 0.5, we decide our prediction is female(1), otherwise it would be 0. Therefore, our algorithm predicts y1 = 1, y2 = 1 and y3 = 1. What is our accuracy? We make a wrong prediction on y1 and y2 and make a correct one on y3. So now our accuracy is 1/3 = 33.33%.

PS: In Amir's answer, back-propagation is said to be an optimization method in NN. I think it would be treated as a way to find gradient for weights in NN. Common optimization method in NN are GradientDescent and Adam.

Share Improve this answer Follow

edited Jun 23, 2023 at 1:52
There
516 ● 6 ● 21

answered Oct 17, 2017 at 22:46
Undecided
631 ● 9 ● 13

1 thank you for the math. it helped clarifying the concept. – Finn Luca Frotscher Jan 9, 2019 at 23:39

3 Your math equations should be converted to mathjax they are rendering very strangely. – IntegrateThis Sep 11, 2020 at 4:22

10

Just to clarify the Training/Validation/Test data sets: The training set is used to perform the initial training of the model, initializing the weights of the neural network.

The validation set is used after the neural network has been trained. It is used for tuning the network's hyperparameters, and comparing how changes to them affect the predictive accuracy of the model. Whereas the training set can be thought of as being used to build the neural network's gate weights, the validation set allows fine tuning of the parameters or architecture of the neural network model. It's useful as it allows repeatable comparison of these different parameters/architectures against the same data and networks weights, to observe how parameter/architecture changes affect the predictive power of the network.

Then the test set is used only to test the predictive accuracy of the trained neural network on previously unseen data, after training and parameter/architecture selection with the training and validation data sets.

Share Improve this answer

Follow

edited Feb 22, 2021 at 14:26

desertnaut

60.2k ● 31 ● 151 ● 176

answered Nov 1, 2017 at 14:46

user7772648