How to set sample_weight in Keras?

Keras August 29, 2021 April 28, 2020

Imagine you create a binary classification model and you have 100 images of Dog and 10000 images of Cat. In this case, it has a very different relative importance for each class. The correctness is more important for some samples than others. For instance, it might be 1-100x more important to correctly classify some samples than other samples.

In imbalanced classification problems, one class is more important (or expensive) than another then you should be **weighing the data**. The idea is to give more weight to rarely-seen classes.

`sample_weights` functionality adds more importance to some samples than others during training.so that the loss function used by the model is weighted per-sample not per class. It changes the way the loss is calculated.

Using the sample weight

A "sample weights" array is an array of numbers that specify how much weight each sample in a batch should have in computing the total loss.

```
sample_weight = np.ones(shape=(len(y_train),))
sample_weight[y_train == 3] = 1.5
```

Here's we use sample weights to give more importance to class #3.It is possible to pass sample weights to a model when using fit:

```
model.fit(x_train, y_train,
          sample_weight=sample_weight,
          batch_size=64,
          epochs=5)
```

The fit method of the classifier accepts a sample_weight array which assigns weights to individual samples.

Different between class_weight and sample_weight

`class_weight` regards the weights of all classes for the entire dataset and it is fixed whereas the `sample_weight` regards the weights of all classes for each individual batch of data created by the generator. `sample_weight` is defined on a per-sample basis and is independent of the class.

`class_weight` is useful when training on highly skewed data sets, for example, a classifier to detect fraudulent transactions. `sample_weight` is useful when you don't have equal confidence in the samples in your batch. A common example is performing regression on measurements with variable uncertainty.

Using the `sample_weight` we can **weight newer data more than old**, forcing the model do adapt to new behavior more quickly, without ignoring valuable old data.

The conclusion is that both `class_weight` and `sample_weight` only affect training loss, no effect on any metrics or validation loss.

Run this code in Google colab

Recent Posts