# Kears LSTM model documentation

**batch\_size** denotes the subset size of your training sample (e.g. 100 out of 1000) which is going to be used in order to train the network during its learning process. Each batch trains network in a successive order, taking into account the updated weights coming from the appliance of the previous batch.

**return\_sequence** indicates if a recurrent layer of the network should return its entire output sequence (i.e. a sequence of vectors of specific dimension) to the next layer of the network, or just its last only output which is a single vector of the same dimension. This value can be useful for networks conforming with an RNN architecture.

**batch\_input\_shape** defines that the sequential classification of the neural network can accept input data of the defined only batch size, restricting in that way the creation of any variable dimension vector. It is widely used in stacked LSTM networks.

we will require fine-grained control over when the internal state of the LSTM is updated. Normally LSTM state is cleared at the ***end of each batch*** in Keras, but we can control it by making the ***LSTM stateful and calling model.reset\_state()*** to manage this state manually.

# RMSE vs MAE

RMSE has the benefit of penalizing large errors more so can be more appropriate in some cases, for example, if being off by 10 is more than twice as bad as being off by 5. But if being off by 10 is just twice as bad as being off by 5, then MAE is more appropriate.

From an interpretation standpoint, ***MAE is clearly the winner***. RMSE does not describe average error alone and has other implications that are more difficult to tease out and understand.

On the other hand, one distinct advantage of RMSE over MAE is that RMSE avoids the use of taking the absolute value, which is undesirable in many mathematical calculations