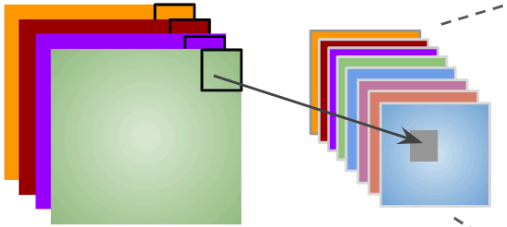
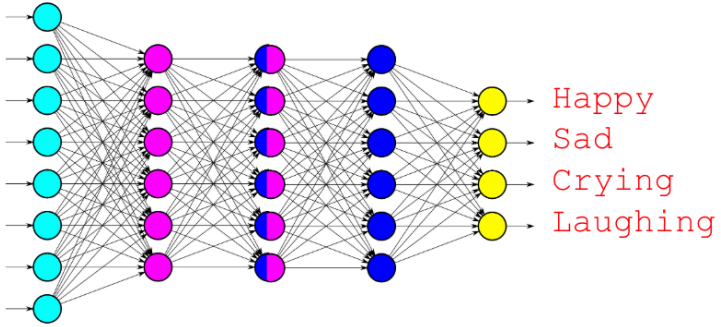
Training must stop!!

Picture of Your Face





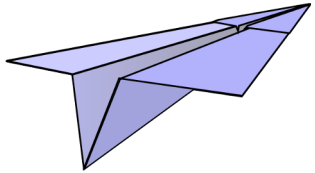
Deep Learning Model

Output


colleague
congratulating me
for achieving
state-of-the-art
me who trained
on test data
MBT thing 10/23/17

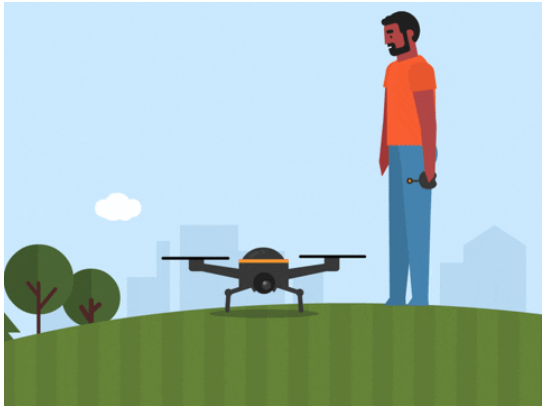
Why do we need callbacks?

Without Callbacks



on a large dataset for tens of epochs - past the initial impulse,
you don't have any control over its trajectory or its landing spot

With Callbacks



Development, training, validation, & test sets

What is the difference between them?

Training set — Which you run your learning algorithm on

Dev (development) set — Which you use to tune parameters, select features, and make other decisions regarding the learning algorithm

Test set — which you use to evaluate the performance of the algorithm, but not to make any decisions regarding what learning algorithm or parameters to use.

Model checkpointing & Early stopping

Model Checkpointing is saving weights of the model during training

```
a = tensorflow.keras.callbacks.ModelCheckpoint(filepath='model.h5',monitor='val_loss', save_best_only=True )

model.fit(x, y, epochs=10, batch_size=32, callbacks=[a], validation_data=(x_val, y_val))
```

```
from google.colab import files
files.download("model.h5")
```

What happens if we monitor ‘loss’ instead of ‘val_loss’?

Early Stopping is interrupting training once a target metric being monitored has stopped improving for a fixed number of epochs

```
b = keras.callbacks.EarlyStopping( monitor = 'val_acc', patience = 100 )

model.fit(x, y, epochs=10, batch_size=32, callbacks=[a, b], validation_data=(x_val, y_val))
```

What happens if we choose ‘acc’ instead of ‘val_acc’?

Reduce learning rate on plateau

- Use this callback to reduce the learning rate when the validation loss has stopped improving
- Reducing or increasing the learning rate in case of a loss plateau is an effective strategy to get out of local minima during training

```
keras.callbacks.ReduceLROnPlateau( monitor = 'val_loss', factor = 0.1, patience = 10 )
```

Divide the learning rate by 10

Do it when ‘val_loss’ does not improve for 10 epochs

Writing Custom Callbacks

`on_epoch_begin`	Called at the start of every epoch
`on_epoch_end`	Called at the end of every epoch
`on_batch_begin`	Called right before processing each batch
`on_batch_end`	Called right after processing each batch
`on_train_begin`	Called at the start of training
`on_train_end`	Called at the end of training

Example: You have a large dataset and would like to plot 'predictions' vs 'true' values after each epoch to observe the progress

Solution: Overwrite the 'on_epoch_end' method with all that you want to do