Model analysis

My ConvNet model is made up of 3 Convolution2D layers, with a MaxPooling2D layer following after the second and the fourth convolution. After the first pooling layer, we double the number of kernels (in line with the previously mentioned principle of sacrificing height and width for more depth). Afterwards, the output of the second pooling layer is flattened to 1D (via the Flatten layer), and passed through two fully connected (Dense) layers. ReLU activations will once again be used for all layers except the output dense layer, which will use a softmax activation (for purposes of probabilistic classification).

To regularise our model, a Dropout layer is applied after each pooling layer, and after the first Dense layer. This is another area where Keras shines compared to other frameworks: it has an internal flag that automatically enables or disables dropout, depending on whether the model is currently used for training or testing.

The remainder of the model specification exactly matches our previous setup for MNIST: - We use the cross-entropy loss function as the objective to optimise (as its derivation is more appropriate for probabilistic tasks); - We use the Adam optimiser for gradient descent; - We report the accuracy2 of the model (as the dataset is balanced across the ten classes); - We hold out 10% of the data for validation purposes.

Epoch 34/50

704/708 [============================>.] - ETA: 0s - loss: 0.6536 - acc: 0.6392Epoch 00033: val\_loss improved from 0.53998 to 0.41251, saving model to /tmp/weights.hdf5