Training Keyword Spotting in Colab

The code for training your model in Colab is very similar to the code you used to test out the pre-trained model. However, we'd like to point out a couple of things that will hopefully make training in Colab easier (and more intuitive).

Timeouts

After 12 hours your Colab instance will automatically recycle so make sure to download anything you want to keep before then. Also after about 90 minutes of no activity Colab may also recycle the instance as it may deem it "stale." Therefore, during training it is best practice to check in every hour and click around on the Colab to make sure it knows you are still there! But/and beyond that you can absolutely minimize the window and continue with other work.

File Structure

The training process will create three directories of note.

- `/data`: will hold all of the data from the Speech Commands dataset. If you want to take a look at particular audio files used for training this is where you can inspect and download them. We will revisit this in Course 3 when you design your own dataset as you'll need to ensure that it matches this format / file structure.
- `/train`: will hold all of the model checkpoint files. These files describe the state of the model after X steps of training (usually denoted in the file name and the training script auto-saves them every 100 or so steps). If you e.g., trained for 100 steps and wanted to train for another 100 more, you can use the latest checkpoint file at step 100 to start from there instead of re-training from scratch for 200 steps.
- `/model`: will hold all of the final processed model files. There you can find both the frozen Tensorflow model, the TensorflowLite model, and in course 3 you will also find the TensorFlowLite Micro model there. If you'd like to download and save your trained model to compare against another trained model at another time you should make sure to download it from there!

The Training Script

The training script is designed to automate a bunch of the training process. It first sets up the optimizer using a `sparse_softmax_cross_entropy` loss function. It then runs the optimizer (by default stochastic gradient descent) on the training data and automatically preprocesses the audio files as we've discussed in previous sections, such as computing the spectrogram and then the MFCC of the audio stream, before passing them to the model/optimizer.

If you want to take a look at the source code it can be found here: https://github.com/tensorflow/tensorflow/blob/master/tensorflow/examples/speech_commands/train.py. You'll notice that there are even more flags than we set in the colab. If you'd like to customize the training process even more you can!