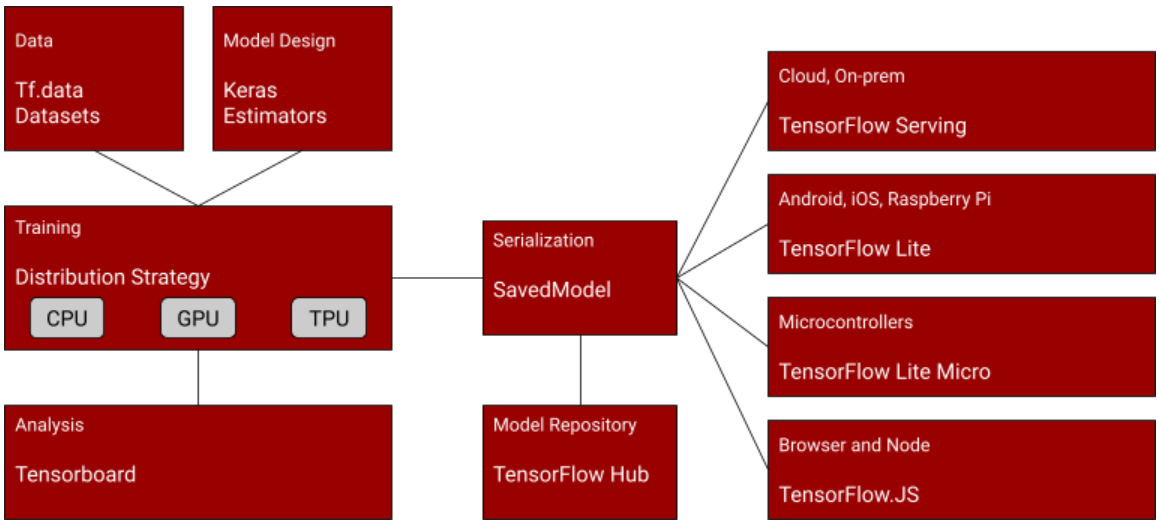
How to use TFLite Models

Previously you saw how to train a model and how to use TensorFlow's Saved Model APIs to save the model to a common format that can be used in a number of different places.

Recall this architecture diagram:



So, after training your model, you could use code like this to save it out:

```
export_dir = 'saved_model/1'
tf.saved_model.save(model, export_dir)
```

This will create a directory with a number of files and metadata describing your model. To learn more about the SavedModel format, take a little time now to read https://www.tensorflow.org/guide/saved_model, and also check out the colab describing how SavedModel works at https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/guide/saved_model.ipynb, and in particular explore 'The SavedModel format on disk' section in that colab.

Once you had your saved model, you could then use the TensorFlow Lite converter to convert it to TF Lite format:

```
converter = tf.lite.TFLiteConverter.from_saved_model(export_dir)
tflite_model = converter.convert()
```

This, in turn could be written to disk as a single file that fully encapsulates the model and its saved weights:

```
import pathlib
tflite_model_file = pathlib.Path('model.tflite')
tflite_model_file.write_bytes(tflite_model)
```

To use a pre-saved tflite file, you then instantiate a tf.lite.Interpreter, and use the 'model_content' property to specify an existing model:

```
interpreter = tf.lite.Interpreter(model_content=tflite_model)
```

Or, if you don't have the existing model already, and just have a file, you can use the 'model_path' property to have the interpreter load the file from disk:

```
interpreter = tf.lite.Interpreter(model_path=tflite_model_file)
```

Once you've loaded the model you can then start performing inference with it. Do note that to run inference you need to get details of the input and output tensors to the model. You'll then set the value of the input tensor, invoke the model, and then get the value of the output tensor. Your code will typically look like this:

```
# Get input and output tensors.
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()

to_predict = # Input data in the same shape as what the model expects

interpreter.set_tensor(input_details[0]['index'], to_predict)

tflite_results = interpreter.get_tensor(output_details[0]['index'])
```

...and a large part of the skills in running models on embedded systems is in being able to format your data to the needs of the model. For example, you might be grabbing frames from a camera that has a particular resolution and encoding, but you need to decode them to 224x224 3-channel images to use with a common model called mobilenet. A large part of any engineering for ML systems is performing this conversion.

To learn more about running inference with models using TensorFLow Lite, check out the documentation at:
https://www.tensorflow.org/lite/guide/inference#load_and_run_a_model_in_python