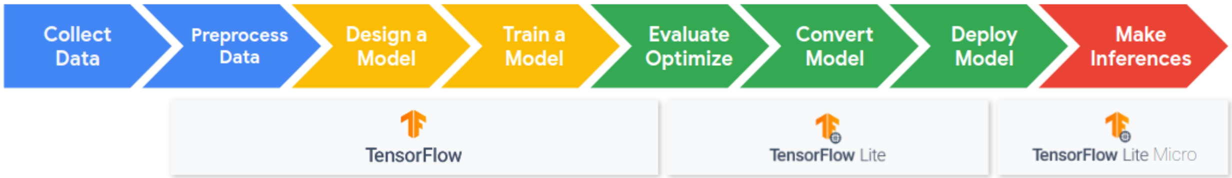
Visual Wake Words Summary

In this section we explored the TinyML flow and data engineering in the context of a Visual Wake Words application, focusing on some unique challenges presented by this computer vision application. Visual Wake Words represents a common TinyML visual use case of identifying whether an object (or a person) is present in the image or not.



Challenges

You explored how latency is a strong constraint for VWW applications as images are large (much larger than spectrograms from KWS) and take a long time to send to the cloud. You also explored how memory is also a strong constraint as images, and their traditional models, take up a lot of memory. Finally, you saw how false positives and false negatives can be particularly challenging in the context of computer vision

Datasets

In the context of datasets your explored three main topics: licensing, privacy, and reuse. Licensing is a major issue for computer vision applications as while images are readily available, they cannot often be used freely without violating copyright. In terms of privacy, you explored how image metadata can often leak significant information about people. Finally, you explored how the VWW dataset reuses the COCO dataset by sub-sampling images using bounding boxes.

MobileNets

You explored how MobileNets use Depthwise Separable Convolutions to reduce the model size and number of operations, at the cost of a decrease in maximal expressiveness. This reduction is dependent on the size of the kernel and the number of kernels used according to the following formula:

$$\frac{\text{Depthwise Separable}}{\text{Standard}} = \frac{1}{N} + \frac{1}{D_K^2}$$

You also explored how reducing the depth multiplier and the input image size can also be used to trade off model memory and latency with accuracy.

Transfer Learning

You next explored how transfer learning can be used to drastically reduce the training time and data requirements. Transfer learning works by reusing the first few layers of a pre-trained model for a task using a similar dataset and re-training only the later layers. This works if the two datasets are close enough together as the first few layers learn to extract generic features from the data while the latter layers learn to extract task specific features. You also learned, however, that transfer learning is not a panacea and is ultimately also limited by the structure of the pre-trained model. You then got to explore using transfer learning to train your own VWW application in Colab!

Metrics

Finally, you explored both qualitative and quantitative metrics for VWW applications. In particular we discussed the latency accuracy tradeoffs often found in VWW applications and the challenges of getting fair results across data represented by both the majority and minority of training samples.