fo_outline TensorFlow modules are reusable pieces of a TensorFlow graph Learn more

Version

Documentation

Overview

EfficientNet-Lite are a family of mobile/IoT-friendly image classification models. They are derived from the EfficientNet architecture originally published as:

- Mingxing Tan and Quoc V. Le: EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, ICML 2019.

EfficientNet-Lite runs well on all mobile CPU/GPU/EdgeTPU hardware and is not specialized to EdgeTPU.

Due to the requirements from edge devices, we mainly made the following changes based on the original EfficientNets.

- Remove squeeze-and-excite (SE): SE are not well supported for some mobile accelerators.
- Replace all swish with RELU6: for easier post-quantization.
- Fix the stem and head while scaling models up: for keeping models small and fast.

Training

Training EfficientNet-lite on Cloud TPUs

Please refer to our tutorial: https://cloud.google.com/tpu/docs/tutorials/efficientnet

Post-training quantization

```
$ export MODEL=efficientnet-lite0
$ wget https://storage.googleapis.com/cloud-tpu-
checkpoints/efficientnet/${MODEL}.tar.gz
$ tar zxf ${MODEL}.tar.gz
$ python export_model.py --model_name=$MODEL --ckpt_dir=$MODEL --
data_dir=/path/to/representative_dataset/ --
output_tflite=${MODEL}_quant.tflite
```

To produce a float model that bypasses the post-training quantization:

```
$ python export_model.py --model_name=$MODEL --ckpt_dir=$MODEL --
output_tflite=${MODEL}_float.tflite --quantize=False
```

The `export_model.py` script can also be used to export a tensorflow saved_model from a training checkpoint:

```
$ python export_model.py --model_name=$MODEL --ckpt_dir=/path/to/model-ckpt/ -
-output_saved_model_dir=/path/to/output_saved_model/ --
output_tflite=${MODEL}_float.tflite --quantize=False
```

Usage

This module implements the common signature forimage classification. It can be used like

```
module =hub.Module("https://tfhub.dev/tensorflow/efficientnet/lite3/classifica
tion/2")
height, width =hub.get_expected_image_size(module)
images = ...  # A batch of images with shape [batch_size, height, width, 3].
logits = module(images)  # Logits with shape [batch_size, num_classes].
```

...or using the signature name `image_classification`. The indices into logits are the `num_classes` = 1000 classes of the classification from the original training (see above). The mapping from indices to class labels can be found in the file atdownload.tensorflow.org/data/ImageNetLabels.txt and please ignore the first background class, ie. index 0 corresponds to "tench".

This module can also be used to compute image feature vectors, using the signature name `image_feature_vector`.

For this module, the size of the input image is flexible, but it would be best to match the model training input, which is `height` x `width` = 300 x 300 pixels for this model. The input `images` are expected to have color values in the range [0,1], following the common image input conventions.

Fine-tuning

In principle, consumers of this module can fine-tune it. However, fine-tuning through a large classification might be prone to overfit.

Fine-tuning requires importing the graph version with tag set `{"train"}` in order to operate batch normalization in training mode.

Changelog

Version 1

- Initial release.

Version 2

- Change variable with shape (1, 1, 3) to scalar for normalization. This should make the model run faster on CPU and be compatible with GPU.