TensorFlow
For Microcontrollers

Pete Warden

Engineer, TensorFlow

TensorFlow

What are we building?

Demo

Goals: Tiny

- Framework that fits in 5KB of RAM, 20KB of Flash
- Speech demo with 30KB of RAM, 40KB of Flash

Goals: Compatible

- Uses TensorFlow Lite APIs and file format
- Most code shared with TF Lite
- There’s a well-supported path to getting TensorFlow models running

Goals: Extensible

- AKA hackable!
- Works with Keil, Mbed, other IDEs
- Only a small working set of files is needed
- Simple to write specialized versions of ops
- Full set of reference code and tests

TensorFlow

Goals: Extensible

- We’re experts on deploying ML, not MCUs
- We need you!
- We aim to make collaboration as simple as possible
- We will deliver ML examples and benchmarks

Example of Extensibility

Depthwise Conv was too slow!

Start by copying micro/kernels/depthwise_conv.cc to micro/kernels/portable_optimized/depthwise_conv.cc

https://github.com/tensorflow/tensorflow/blob/master/tensorflow/lite/experimental/micro/kernels/portable_optimized/depthwise_conv.cc

```
int32 acc = 0;
for (int filter_y = 0; filter_y < filter_height; ++filter_y) {
  for (int filter_x = 0; filter_x < filter_width; ++filter_x) {
    const int in_x =
        in_x_origin + dilation_width_factor * filter_x;
    const int in_y =
        in_y_origin + dilation_height_factor * filter_y;
    // If the location is outside the bounds of the input image,
    // use zero as a default value.
    if ((in_x >= 0) && (in_x < input_width) && (in_y >= 0) &&
        (in_y < input_height)) {
      int32 input_val =
          input_data[Offset(input_shape, b, in_y, in_x, ic)];
      int32 filter_val = filter_data[Offset(
          filter_shape, 0, filter_y, filter_x, oc)];
      acc += (filter_val + filter_offset) *
             (input_val + input_offset);
    }
  }
}
```

```
// Specialized implementation of the depthwise convolution operation designed to
// work with the particular filter width of eight used by the default micro
// speech sample code. It uses 1KB of RAM to hold reordered weight parameters,
// converted from TFLite's NHWC format to NCHW format, and expressed as signed
// eight bit integers, rather than unsigned. Care must be taken when calling
// this not to use it for more than one node since there's only a single static
// buffer holding the weights. You should use this implementation if depthwise
// convolutions are a performance bottleneck, you have a layer that meets the
// parameter requirements, and the extra RAM usage and additional code size are
// not an issue.
static inline void DepthwiseConvOptimizedForFilterWidthEight(
   TfLiteContext* context, const DepthwiseParams& params,
   const RuntimeShape& input_shape, const uint8* input_data,
   const RuntimeShape& filter_shape, const uint8* filter_data,
   const RuntimeShape& bias_shape, const int32* bias_data,
   const RuntimeShape& output_shape, uint8* output_data) {
...
```

```
// If this is the first time through, repack the weights into a cached buffer
// so that they can be accessed sequentially.
static bool is_reshaped_filter_initialized = false;
if (!is_reshaped_filter_initialized) {
  for (int filter_y = 0; filter_y < filter_height; ++filter_y) {
    for (int filter_x = 0; filter_x < filter_width; ++filter_x) {
      for (int oc = 0; oc < output_depth; ++oc) {
        const uint8* current_filter =
            filter_data + Offset(filter_shape, 0, filter_y, filter_x, oc);
        int8* reshaped_filter =
            reshaped_filter_data +
            Offset(reshaped_filter_shape, 0, oc, filter_y, filter_x);
        *reshaped_filter = (int32_t)(*current_filter) + filter_offset;
      }
    }
  }
  is_reshaped_filter_initialized = true;
}
...
```

```
if ((filter_width == 8) && !is_out_of_x_bounds) {
  int8* current_filter =
      reshaped_filter_data + Offset(reshaped_filter_shape, 0, oc,
                                    filter_y, filter_x_start);
  const uint32_t input_vals0 =
      *reinterpret_cast<const uint32_t*>(current_input);
  current_input += 4;
  const int32_t filter_vals0 =
      *reinterpret_cast<const int32_t*>(current_filter);
  current_filter += 4;
  const uint8 input_val0 = input_vals0 & 0xff;
  const int8 filter_val0 = filter_vals0 & 0xff;
  acc += filter_val0 * input_val0;
  const uint8 input_val1 = (input_vals0 >> 8) & 0xff;
  const int8 filter_val1 = (filter_vals0 >> 8) & 0xff;
  acc += filter_val1 * input_val1;
  const uint8 input_val2 = (input_vals0 >> 16) & 0xff;
  const int8 filter_val2 = (filter_vals0 >> 16) & 0xff;
  acc += filter_val2 * input_val2;
  const uint8 input_val3 = (input_vals0 >> 24) & 0xff;
  const int8 filter_val3 = (filter_vals0 >> 24) & 0xff;
  acc += filter_val3 * input_val3;
} else {
   const uint8* current_filter =
       filter_data +
       Offset(filter_shape, 0, filter_y, filter_x_start, oc);
   for (int filter_x = filter_x_start; filter_x < filter_x_end;
        ++filter_x) {
     int32 input_val = *current_input;
     current_input += input_depth;
     int32 filter_val = *current_filter;
     current_filter += output_depth;
     acc +=
         (filter_val + filter_offset) * (input_val + input_offset);
   }
 }
```

Future?

Depthwise Conv was too slow!

Start by copying micro/kernels/depthwise_conv.cc to micro/kernels/portable_optimized/depthwise_conv.cc

https://github.com/tensorflow/tensorflow/blob/master/tensorflow/lite/experimental/micro/kernels/portable_optimized/depthwise_conv.cc

TensorFlow

Future - Visual Wake Words

Aakanksha Chowdhery

ML Engineer

Future - Visual Wake Words

Popular use-case: classify person/not-person

Initially presence classification

Eventually extend to object counting/localization

Future - Visual Wake Words

Popular use-case: classify person/not-person

ImageNet dataset: classifies 1000 classes

CIFAR10: very low-resolution images

Need ImageNet for microcontrollers !

Future - Visual Wake Words

Open data set based on MS COCO

Labeled images with >5% person

TensorFlow

Future - Visual Wake Words

Need models that fit 250 KB SRAM

Compressed MobileNet architectures to <250KB

Initially presence classification >90% accuracy

TensorFlow

Future - Visual Wake Words

Dataset release and challenge details coming up soon!

More details at the poster session!

Get it. Try it.

Code: github.com/tensorflow/tensorflow/tree/master/tensorflow/lite/experimental/micro

Docs: tensorflow.org/lite/guide/microcontroller

Example: g.co/codelabs/sparkfunTF