- An implementation of the ResNet50 model. The ResNet50 v1.5 model is a modified version of the original ResNet50 v1 model.

7.1.4. Manual Conversion To Mixed Precision In PyTorch

We recommend using AMP to implement mixed precision in your model. However, if you wish to implement mixed precision yourself, refer to our GTC talk on manual mixed precision (video, slides). An example of manual mixed precision for Imagenet training can be found here, and an RNN example can be found here.

7.2. TensorFlow

TensorFlow supports FP16 storage and tensor core math. Models that contain convolutions or matrix multiplications using the **`tf.float16`** data type will automatically take advantage of tensor core hardware whenever possible.

In order to make use of tensor cores, FP32 models will need to be converted to use a mix of FP32 and FP16. This can be done either automatically using automatic mixed precision (AMP) or manually.

7.2.1. Automatic Mixed Precision Training In TensorFlow

The automatic mixed precision feature is available starting inside the NVIDIA NGC TensorFlow 19.03 container. Enable this feature inside the container requires simply setting one environment variable:

```
export TF_ENABLE_AUTO_MIXED_PRECISION=1
```

You can also set the environment variable inside a TensorFlow Python script by calling the following at the beginning of the script:

```
os.environ['TF_ENABLE_AUTO_MIXED_PRECISION'] = '1'
```

When enabled, automatic mixed precision will do two things:

1. Insert the appropriate cast operations into your TensorFlow graph to use float16 execution and storage where appropriate.
2. Turn on automatic loss scaling inside the training Optimizer object.

For more information on automatic mixed precision, see the TensorFlow User Guide.

7.2.2. Success Stories

The models where we have seen speedup using mixed precision are:

Table 2 Mixed precision model speedup

Model	Speedup
BERT Q&A	3.3X speedup