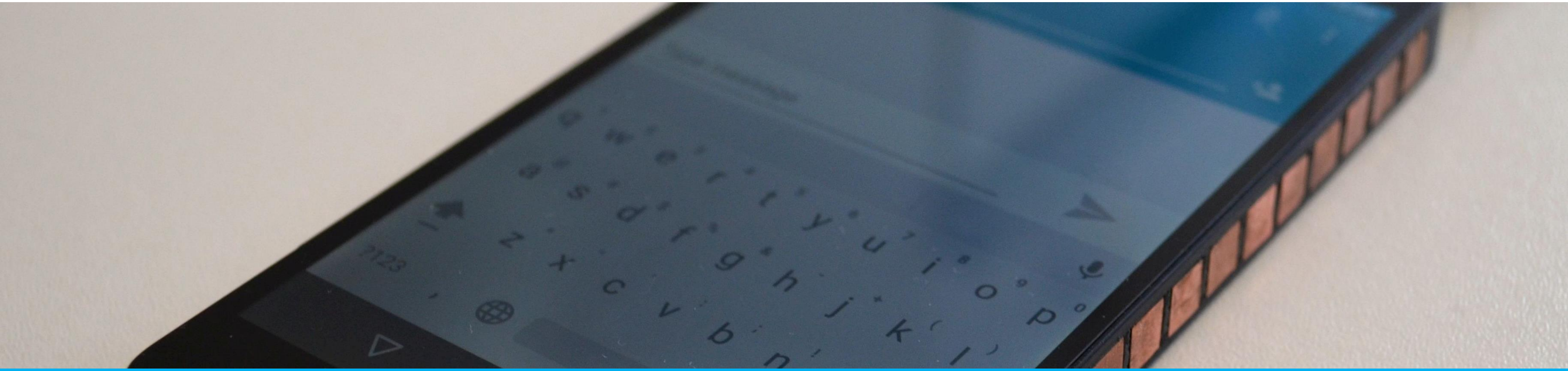
University of Stuttgart
Institute for Visualization and Interactive Systems



# TensorFlow Mobile: Exporting and Optimizing Models for Mobile Devices

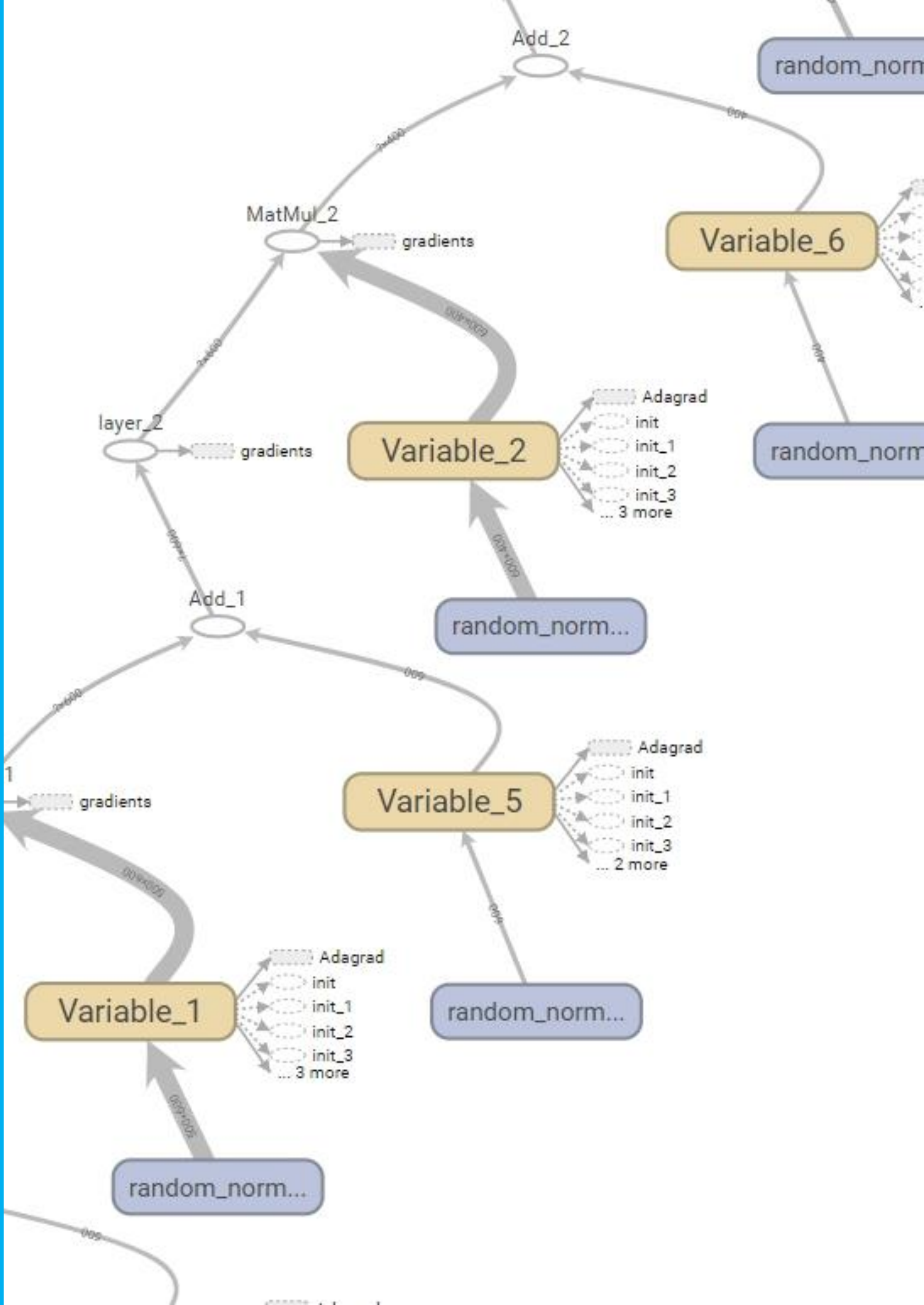
Huy Viet Le, Sven Mayer, Niels Henze

interactionlab.io

## Core Graph Data Structures

- `tf.Tensor`
  - Can be a `tf.variable`, `tf.constant`, `tf.placeholder`, …
  - Example: `c = tf.constant([1,2,3])`
- `tf.Operation`
  - Node in a graph that take `tf.Tensor(s)` as input and return `tf.Tensor(s)` as output.
  - Example: `c = tf.matmul(a, b)`
- `tf.Graph`
  - Contain set of `tf.Operation` and `tf.Tensor` object for using them e.g. in sessions.

Source: https://www.tensorflow.org/api_guides/python/framework



## Data Structures in TensorFlow

### How models are stored

- `NodeDef`: Operations and Fixed Values
  - A single *operation* in a model and the basic unit of computation in TensorFlow.
  - Operation type (e.g. Add, or Mul) and parameters needed to execute the operation.
  - `Const` are also stored in a NodeDef
- `GraphDef`
  - Contains a list of `NodeDefs` to define the computational graph.
  - Results in .pbtxt (text representation) or .pb (binary representation) files.
- `MetaGraphDef`
  - Contains a `GraphDef` and 'signatures' (e.g. checkpoint paths, and I/O node information).
  - Results in .meta files and required for re-training.

Source: https://www.tensorflow.org/mobile/prepare_models

## Data Structures in TensorFlow

### How models are stored

- `Checkpoint`: Variable Values
  - Stores *variables* (whose values are held in RAM for time-critical operations).
  - Examples are weights and biases.
  - Results in `*.data` files.
- `SavedModel`: A bundle of all required data
  - Includes `MetaGraphDef,` checkpoint files, and further asset files (e.g. label names).

Source: https://www.tensorflow.org/mobile/prepare_models

# Porting Models to Mobile and Embedded Devices

## ‘Freezing’ graphs

- Running a computational graph requires `NodeDefs` as well as `Variables`.
- `Variables` (e.g. weights) are stored in Checkpoints and not in `GraphDefs`.
- Inconvenient to have separate files for deployment in production.
- Freezing: Converting all Variables to `Const` to embed them in a single `GraphDef`.
  - Results in ‘`.pbtxt`’ (text format) or ‘`.pb`’ (binary format).

## Using Models on an Android Device

### Overview of procedure

- Add the TensorFlow Inference Interface into the gradle build file: `'org.tensorflow:tensorflow-android:+'`
- Copy model file (*.pb) to the project’s assets folder.
- Initialize `TensorFlowInferenceInterface` with the model file.
- Specify input/output node names, and input data structure.
- Retrieve inference result and decode one-hot encoding (for classification).

## Live Code Demo

## Optimizing Models for Mobile Devices

## Graph Transform Tool

- Needs to be manually built with Bazel ([https://bazel.build/](https://bazel.build/)).
- In case TensorFlow is already built with Bazel:
  - Build: `bazel build tensorflow/tools/graph_transforms:transform_graph`
  - Run: `bazel run tensorflow/tools/graph:transforms:transform_graph --`
- In case TensorFlow was installed with pip:
  - Download TensorFlow sources from github and unpack.
  - Build: `bazel build tensorflow/tools/graph_transforms:transform_graph`
  - SymLink to the executables will be created in the root folder.
  - Run: `bazel-bin/tensorflow/tools/graph_transforms/transform_graph --`

## Getting an Overview of the Graph

### Graph Summary & Benchmark

```
bazel-bin/tensorflow/tools/graph_transforms/summarize_graph
      --in_graph=original_model.pb
```

## Optimizing the Graph for Mobile Devices

### Quantization of Neural Networks

- Weights and biases are stored as 32-bit floating points.
- Idea: Reducing floating point accuracy to eight-bit equivalent for constants.
- 70% compression in comparison to 8% using gzip [1].
  - Accuracy Loss: ~1% [1].

```
bazel-bin/tensorflow/tools/graph_transforms/transform_graph
      --in_graph=original_model.pb
      --out_graph=quantized_model.pb
      --inputs=input
      --outputs=output
      --transforms='quantize_weights'
```

[1] Source: https://codelabs.developers.google.com/codelabs/tensorflow-for-poets-2/#4

# Optimizing the Graph for Mobile Devices

## Removing unused nodes

- Remove `GraphDefs` contain `NodeDefs` that are needed for e.g. summary ops.
  - For inference only, these `NodeDefs` are not needed.

```
bazel-bin/tensorflow/tools/graph_transforms/transform_graph\
      --in_graph=original_model.pb\
      --out_graph=optimized_model.pb\
      --inputs=input\
      --outputs=output\
      --transforms='strip_unused_nodes(type=float, shape="1,784")'
```

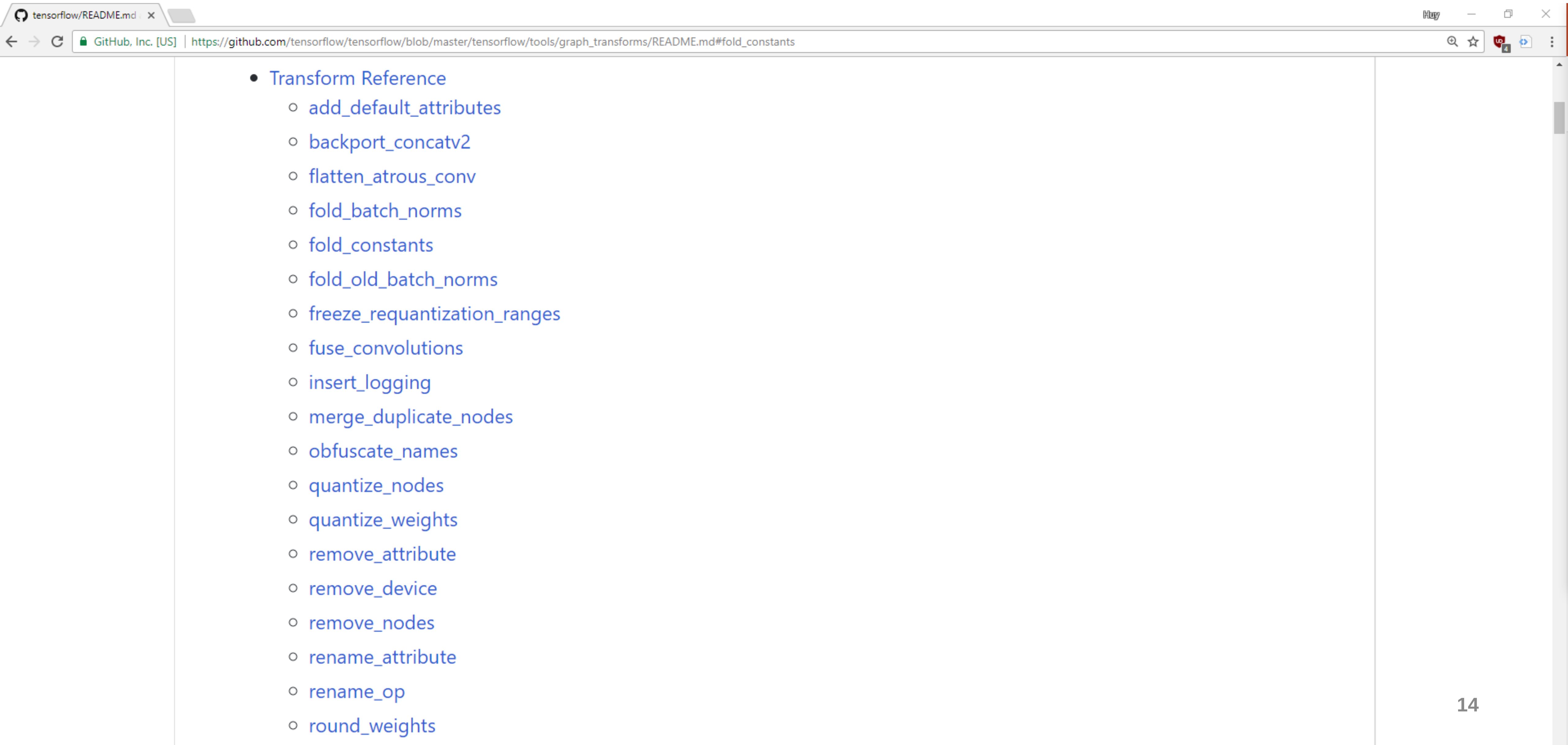
## Optimizing the Graph for Mobile Devices

### Fold constants

- Replaces sub-graphs that always evaluate to constant expressions with actual constants.

```
bazel-bin/tensorflow/tools/graph_transforms/transform_graph\
      --in_graph=original_model.pb\
      --out_graph=optimized_model.pb\
      --inputs=input\
      --outputs=output\
      --transforms='fold_constants'
```

## Further Transformations


tensorflow/README.md
GitHub, Inc. [US]
https://github.com/tensorflow/tensorflow/blob/master/tensorflow/tools/graph_transforms/README.md#fold_constants

- Transform Reference
  - add_default_attributes
  - backport_concatv2
  - flatten_atrous_conv
  - fold_batch_norms
  - fold_constants
  - fold_old_batch_norms
  - freeze_requantization_ranges
  - fuse_convolutions
  - insert_logging
  - merge_duplicate_nodes
  - obfuscate_names
  - quantize_nodes
  - quantize_weights
  - remove_attribute
  - remove_device
  - remove_nodes
  - rename_attribute
  - rename_op
  - round_weights