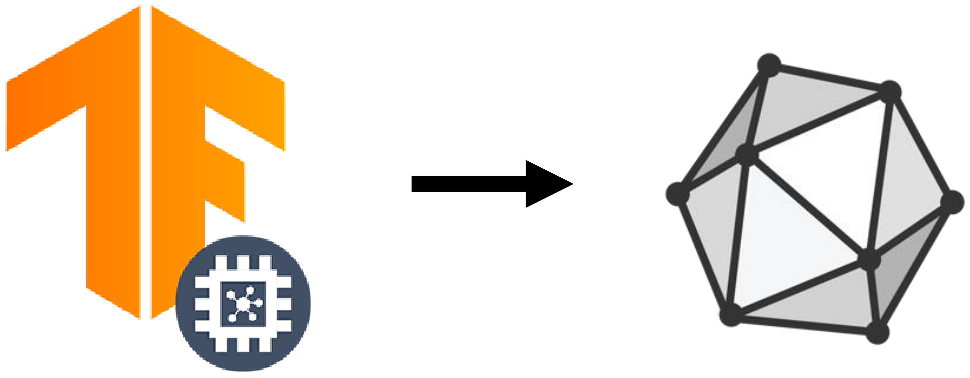
TensorFlow Lite to ONNX Conversion

ONNX Runtime Mobile

TOM WILDENHAIN

SCOTT MCKAY

MICROSOFT



TensorFlow Lite to ONNX

TOM WILDENHAIN SOFTWARE ENGINEER, ONNX CONVERTERS TEAM

MICROSOFT

TensorFlow

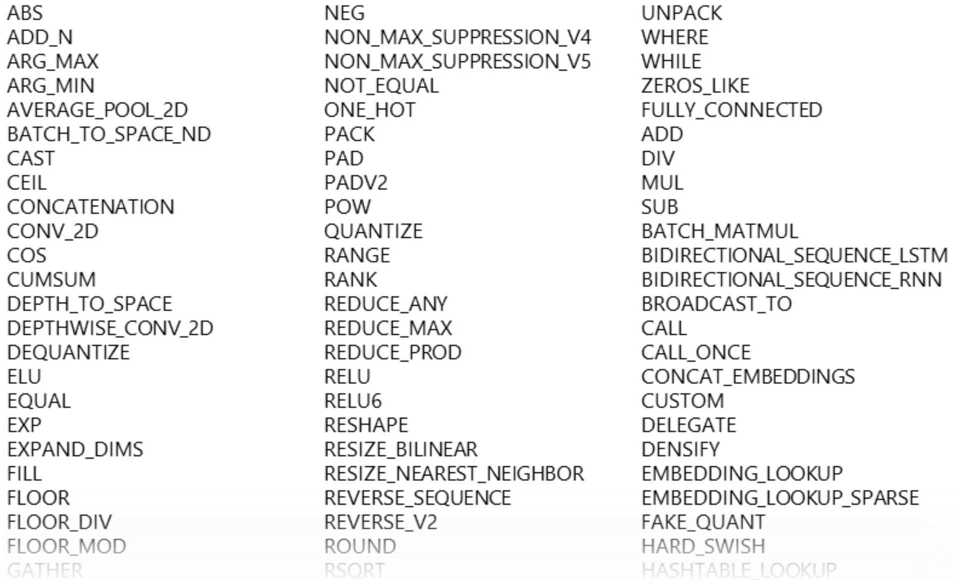
- Used for inference and training
- > 1,000 ops
- Already have conversions for many ops

Abort
Abs
AccumulateNV2
AccumulatorApplyGradient
AccumulatorNumAccumulated
AccumulatorSetGlobalStep
AccumulatorTakeGradient
Acos
Acosh
Add
AddManySparseToTensorsMap
AddN
AddSparseToTensorsMap
AddV2
AdjustContrast
AdjustContrastv2
AdjustHue
AdjustSaturation
All
AllCandidateSampler
AllToAll
Angle
AnonymousIterator
AnonymousIteratorV2
AnonymousMemoryCache
AnonymousMultiDeviceIterator
AnonymousRandomSeedGenerator
AnonymousSeedGenerator
Any
ApplyAdaMax
ApplyAdadelta
ApplyAdagrad
ApplyAdagradDA
ApplyAdagradV2
ApplyAdam
ApplyAddSign
ApplyCenteredRMSProp
ApplyFtrl
ApplyFtrlV2
ApplyGradientDescent
ApplyMomentum
ApplyPowerSign
ApplyProximalAdagrad
ApplyProximalGradientDescent
ApplyRMSProp
ApproximateEqual
ArgMax
ArgMin
CountUpTo
CreateSummaryDbWriter
CreateSummaryFileWriter
CropAndResize
CropAndResizeGradBoxes
CropAndResizeGradImage
Cross
CrossReplicaSum
CudnnRNN
CudnnRNNBackprop
CudnnRNNBackpropV2
CudnnRNNBackpropV3
CudnnRNNCanonicalToParams
CudnnRNNCanonicalToParamsV2
CudnnRNNParamsSize
CudnnRNNParamsToCanonical
CudnnRNNParamsToCanonicalV2
CudnnRNNV2
CudnnRNNV3
Cumprod
Cumsum
CumulativeLogsumexp
DataFormatDimMap
DataFormatVecPermute
DataServiceDataset
DatasetCardinality
DatasetFromGraph
DatasetToGraph
DatasetToGraphV2
DatasetToSingleElement
DatasetToTFRecord
Dawsn
DebugGradientIdentity
DebugGradientRefIdentity
DebugIdentity
DebugIdentityV2
DebugNanCount
DebugNumericSummary
DebugNumericSummaryV2
DecodeAndCropJpeg
DecodeBase64
DecodeBmp
DecodeCSV
DecodeCompressed
DecodeGif
DecodeImage
DecodeJSONExample
DecodeJpeg
Greater
GreaterEqual
GroupByReducerDataset
GroupByWindowDataset
GuaranteeConst
HSVToRGB
HashTable
HashTableV2
HistogramFixedWidth
HistogramSummary
IFFT
IFFT2D
IFFT3D
IRFFT
IRFFT2D
IRFFT3D
Identity
IdentityN
IdentityReader
IdentityReaderV2
If
Igamma
IgammaGradA
Igammac
IgnoreErrorsDataset
Imag
ImageProjectiveTransformV2
ImageProjectiveTransformV3
ImageSummary
ImmutableConst
ImportEvent
InTopK
InTopKV2
InfeedDequeue
InfeedDequeueTuple
InfeedEnqueue
InfeedEnqueuePrelinearizedBuffer
InfeedEnqueueTuple
InitializeTable
InitializeTableFromDataset
InitializeTableFromTextFile
InitializeTableFromTextFileV2
InitializeTableV2
InplaceAdd
InplaceSub
InplaceUpdate
InterleaveDataset
Inv
NonSerializableDataset
NotEqual
NthElement
OneHot
OneShotIterator
OnesLike
OptimizeDataset
OptimizeDatasetV2
OptionalFromValue
OptionalGetValue
OptionalHasValue
OptionalNone
OrderedMapClear
OrderedMapIncompleteSize
OrderedMapPeek
OrderedMapSize
OrderedMapStage
OrderedMapUnstage
OrderedMapUnstageNoKey
OutfeedDequeue
OutfeedDequeueTuple
OutfeedDequeueTupleV2
OutfeedDequeueV2
OutfeedEnqueue
OutfeedEnqueueTuple
Pack
Pad
PadV2
PaddedBatchDataset
PaddedBatchDatasetV2
PaddingFIFOQueue
PaddingFIFOQueueV2
ParallelConcat
ParallelDynamicStitch
ParallelInterleaveDataset
ParallelInterleaveDatasetV2
ParallelInterleaveDatasetV3
ParallelInterleaveDatasetV4
ParallelMapDataset
ParallelMapDatasetV2
ParameterizedTruncatedNormal
ParseExample
ParseExampleDataset
ParseExampleDatasetV2
ParseExampleV2
ParseSequenceExample
ParseSequenceExampleV2
ParseSingleExample
ResourceApplyFtrl
ResourceApplyFtrlV2
ResourceApplyGradientDescent
ResourceApplyKerasMomentum
ResourceApplyMomentum
ResourceApplyPowerSign
ResourceApplyProximalAdagrad
ResourceApplyProximalGradientDescent
ResourceApplyRMSProp
ResourceConditionalAccumulator
ResourceCountUpTo
ResourceGather
ResourceGatherNd
ResourceScatterAdd
ResourceScatterDiv
ResourceScatterMax
ResourceScatterMin
ResourceScatterMul
ResourceScatterNdAdd
ResourceScatterNdMax
ResourceScatterNdMin
ResourceScatterNdSub
ResourceScatterNdUpdate
ResourceScatterSub
ResourceScatterUpdate
ResourceSparseApplyAdadelta
ResourceSparseApplyAdagrad
ResourceSparseApplyAdagradDA
ResourceSparseApplyAdagradV2
ResourceSparseApplyCenteredRMSProp
ResourceSparseApplyFtrl
ResourceSparseApplyFtrlV2
ResourceSparseApplyKerasMomentum
ResourceSparseApplyMomentum
ResourceSparseApplyProximalAdagrad
ResourceSparseApplyProximalGradientDescent
ResourceSparseApplyRMSProp
ResourceStridedSliceAssign
Restore
RestoreSlice
RestoreV2
RetrieveTPUEmbeddingADAMParameters

…

TFLite

- Lightweight runtime used for inference
- ~130 ops
- Models created from TensorFlow

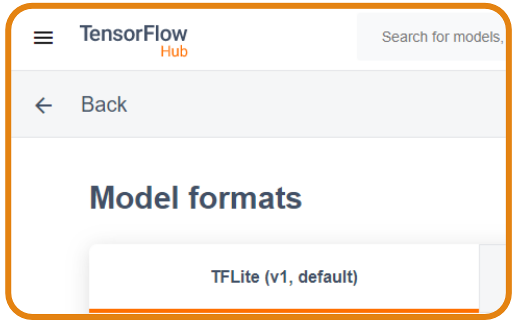


ABS
ADD_N
ARG_MAX
ARG_MIN
AVERAGE_POOL_2D
BATCH_TO_SPACE_ND
CAST
CEIL
CONCATENATION
CONV_2D
COS
CUMSUM
DEPTH_TO_SPACE
DEPTHWISE_CONV_2D
DEQUANTIZE
ELU
EQUAL
EXP
EXPAND_DIMS
FILL
FLOOR
FLOOR_DIV
FLOOR_MOD
GATHER
NEG
NON_MAX_SUPPRESSION_V4
NON_MAX_SUPPRESSION_V5
NOT_EQUAL
ONE_HOT
PACK
PAD
PADV2
POW
QUANTIZE
RANGE
RANK
REDUCE_ANY
REDUCE_MAX
REDUCE_PROD
RELU
RELU6
RESHAPE
RESIZE_BILINEAR
RESIZE_NEAREST_NEIGHBOR
REVERSE_SEQUENCE
REVERSE_V2
ROUND
RSQRT
UNPACK
WHERE
WHILE
ZEROS_LIKE
FULLY_CONNECTED
ADD
DIV
MUL
SUB
BATCH_MATMUL
BIDIRECTIONAL_SEQUENCE_LSTM
BIDIRECTIONAL_SEQUENCE_RNN
BROADCAST_TO
CALL
CALL_ONCE
CONCAT_EMBEDDINGS
CUSTOM
DELEGATE
DENSIFY
EMBEDDING_LOOKUP
EMBEDDING_LOOKUP_SPARSE
FAKE_QUANT
HARD_SWISH
HASHTABLE_LOOKUP

…

TFLite Conversion

```
pip install tf2onnx
python -m tf2onnx.convert --tflite ssdmobilenet.tflite --output ssdmobilenet.onnx --opset 13
```



TensorFlow Hub

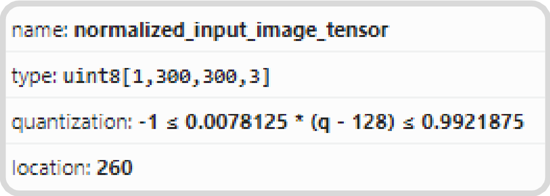
Search for models,

Back

Model formats

TFLite (v1, default)

Some models are only available for TFLite



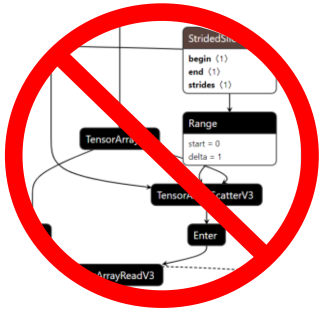
name: normalized_input_image_tensor

type: uint8[1,300,300,3]

quantization: -1 ≤ 0.0078125 * (q - 128) ≤ 0.9921875

location: 260

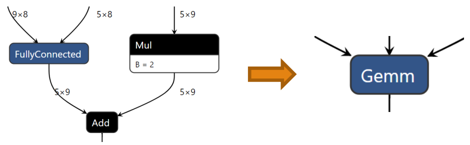
Automatic quantization support!



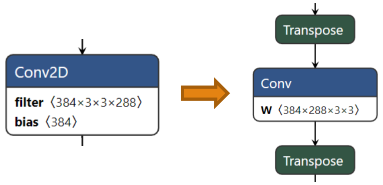
TFLite models are often cleaner

Conversion Process

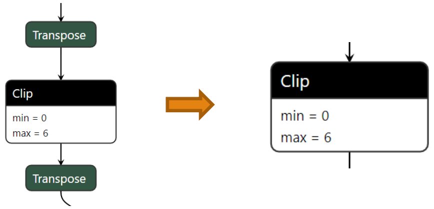
1. Rewriters
   - Convert op patterns



2. Handlers
   - Convert individual ops

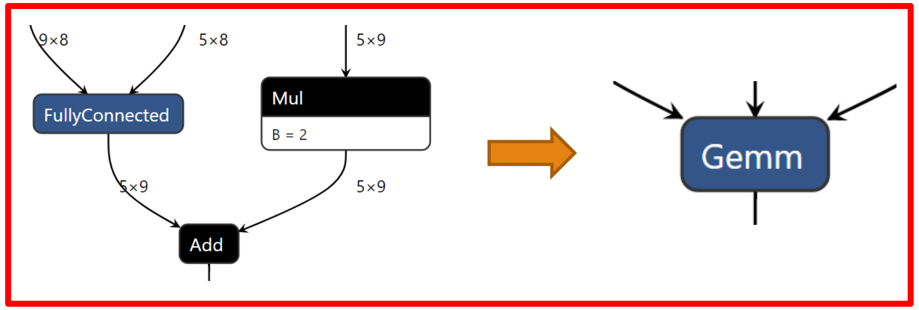


3. Optimizers
   - Remove unnecessary ops

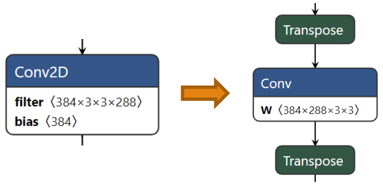


Conversion Process

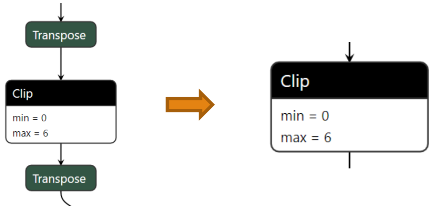
1. **Rewriters**
   - **Convert op patterns**



2. Handlers
   - Convert individual ops

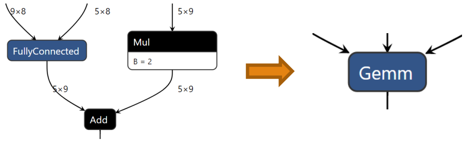


3. Optimizers
   - Remove unnecessary ops

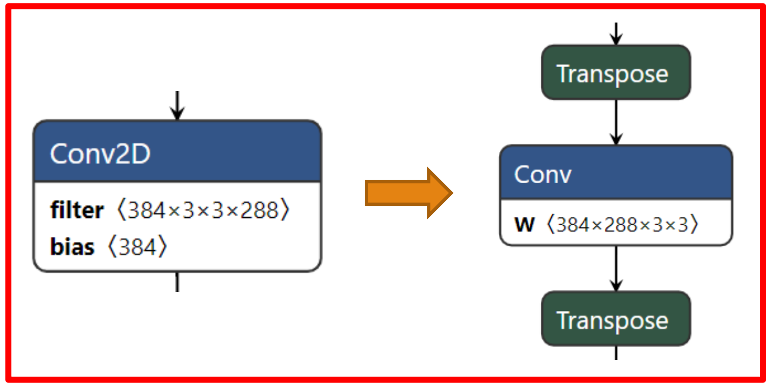


Conversion Process

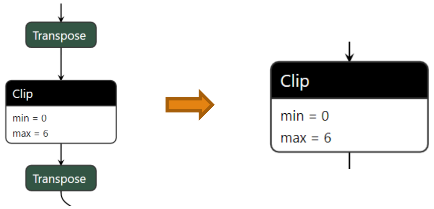
1. Rewriters
   - Convert op patterns



2. **Handlers**
   - **Convert individual ops**

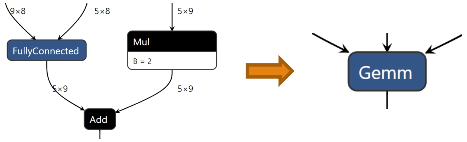


3. Optimizers
   - Remove unnecessary ops

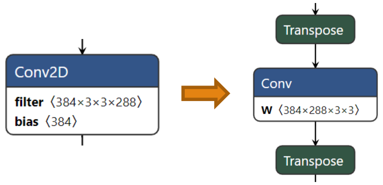


Conversion Process

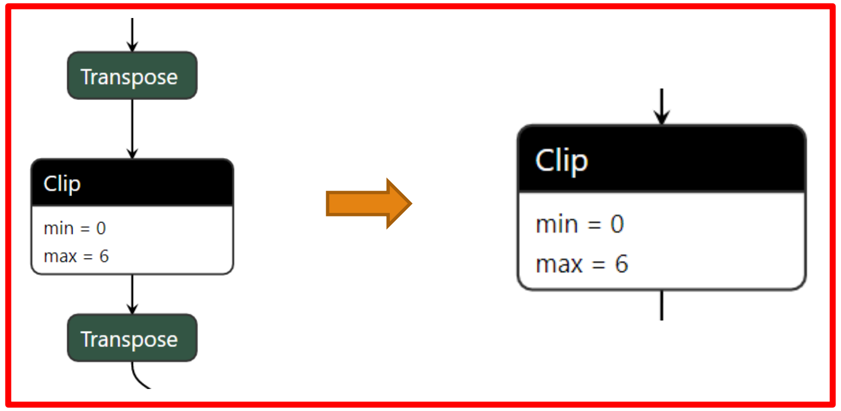
1. Rewriters
   - Convert op patterns



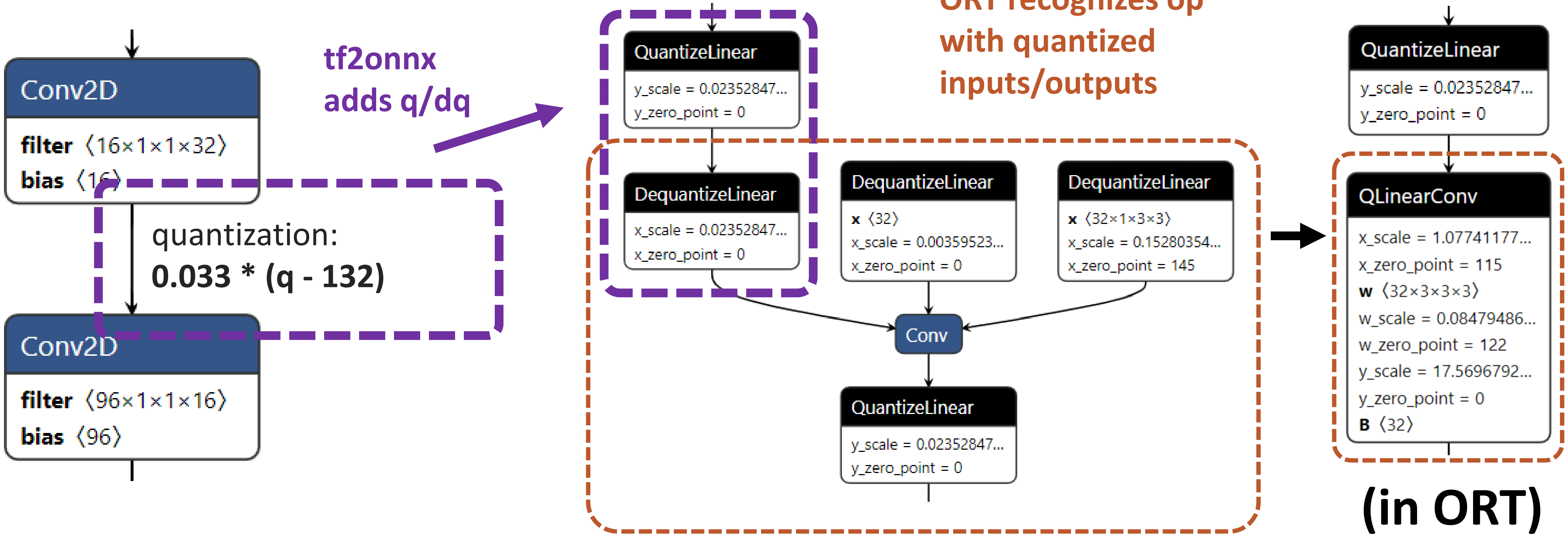
2. Handlers
   - Convert individual ops



3. **Optimizers**
   - **Remove unnecessary ops**



Quantization



Conv2D
filter ⟨16×1×1×32⟩
bias ⟨16⟩

quantization:
0.033 * (q - 132)

Conv2D
filter ⟨96×1×1×16⟩
bias ⟨96⟩

tf2onnx adds q/dq

QuantizeLinear
y_scale = 0.02352847...
y_zero_point = 0

DequantizeLinear
x_scale = 0.02352847...
x_zero_point = 0

DequantizeLinear
x ⟨32⟩
x_scale = 0.00359523...
x_zero_point = 0

DequantizeLinear
x ⟨32×1×3×3⟩
x_scale = 0.15280354...
x_zero_point = 145

Conv

QuantizeLinear
y_scale = 0.02352847...
y_zero_point = 0

ORT recognizes op with quantized inputs/outputs

QuantizeLinear
y_scale = 0.02352847...
y_zero_point = 0

QLinearConv
x_scale = 1.07741177...
x_zero_point = 115
w ⟨32×3×3×3⟩
w_scale = 0.08479486...
w_zero_point = 122
y_scale = 17.5696792...
y_zero_point = 0
B ⟨32⟩

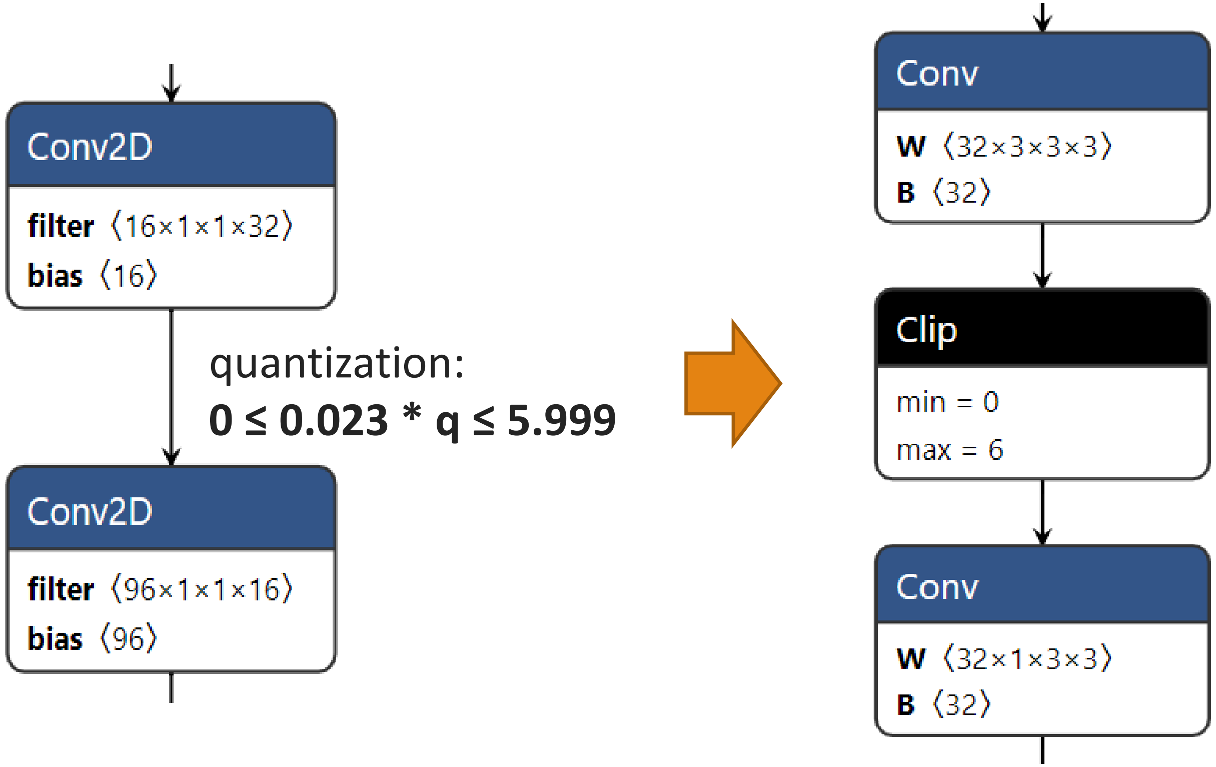
(in ORT)

TFLite graph

ONNX graph

Dequantizing Models `--dequantize`

Detect ReLU and ReLU6 ops from quantization range



Conv2D

filter ⟨16×1×1×32⟩

bias ⟨16⟩

quantization:

$0 \leq 0.023 * q \leq 5.999$

Conv2D

filter ⟨96×1×1×16⟩

bias ⟨96⟩

Conv

W ⟨32×3×3×3⟩

B ⟨32⟩

Clip

min = 0

max = 6

Conv

W ⟨32×1×3×3⟩

B ⟨32⟩

Support and Feature Requests

Please submit feature requests to GitHub

TFLite -> ONNX conversion is new, expect improvements as we support more ops

github.com/onnx/tensorflow-onnx

ONNX Runtime Mobile

SCOTT MCKAY ONNX RUNTIME MOBILE TECHNICAL LEAD

MICROSOFT

ONNX Runtime Mobile

ONNX Runtime Mobile is a variant of ONNX Runtime that minimizes binary size for mobile and edge scenarios

- Same codebase as ONNX Runtime
- Available since ONNX Runtime v1.5, Sept 2020

Includes only required operator kernels in the build

- Can also reduce types supported by operator kernels

Custom format for the model file

ONNX Runtime Mobile

Runtime usage of ONNX Runtime Mobile is the same as regular ONNX Runtime

- C, C++, Python and Java APIs are available

Supports NNAPI Execution Provider on Android

Supports CoreML Execution Provider on iOS (preview)

Documentation:

- ONNX_Runtime_for_Mobile_Platforms.md

ORT format model

Created from an ONNX model

- Python script handles conversion

During conversion:

- ONNX Runtime optimizations are applied
  - e.g. constant folding
- Nodes are assigned to kernels
  - No ONNX schema dependency
    - Significant binary size and memory usage saving

Uses google::flatbuffers

Operator Kernel selection

Configuration file specifies the kernels to include in the build

- Model conversion script will automatically generate configuration file when converting models
- Configuration file can also be manually created/edited

Example config:

- `ai.onnx;11;AveragePool,Conv,Reshape,Shape,Softmax,Squeeze,Transpose`

Reduced Type Support

Can limit types that operator kernels support

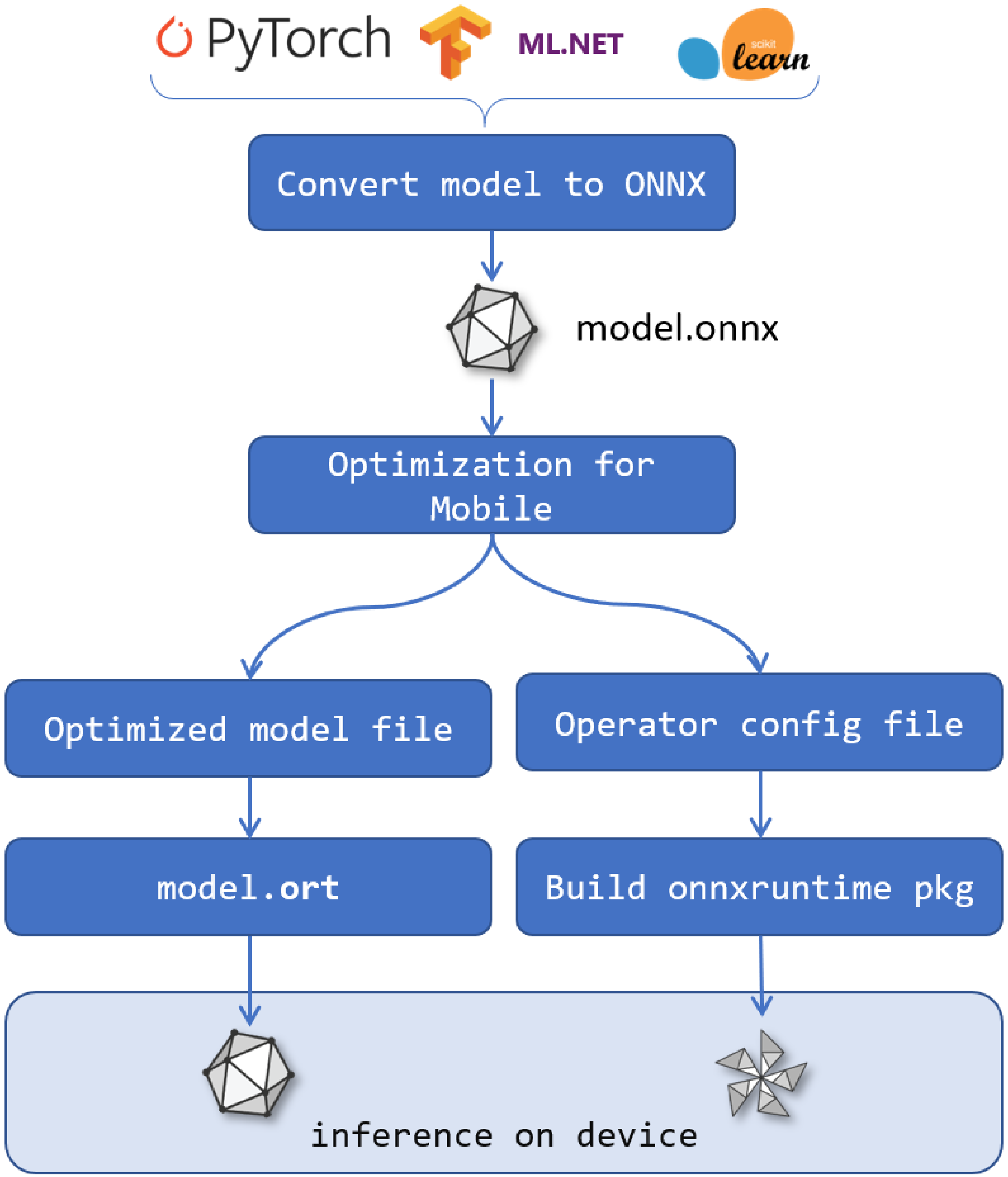
- Model conversion script can automatically detect required types on a per-operator basis
- Alternatively, can specify a global list of types to support

Model based type reduction generally reduces kernel binary size by 25 - 33%

Available in ONNX Runtime v1.7

- March 2021

ORT Mobile Usage



Binary size

Primary choices that determine binary size:

- Operators and types to include
- Enable/disable exceptions
- Enable/disable support for traditional ML operators
- Use static or shared libc++ on Android

Base build size for Android ARM64 *NDK 21.1, no operator kernels, shared libc++, exceptions and traditional ML support disabled*	libonnxruntime.so: 755KB (280KB in AAR)
With operator kernels required by Mobilenet	libonnxruntime.so: 895KB (342KB in AAR)
With reduced type support enabled	libonnxruntime.so: 851KB (325KB in AAR) 31% reduction in size of kernels

NNAPI Support

Usage of NNAPI is determined at runtime

- based on whether NNAPI is available and device capabilities
  - e.g. older version of NNAPI may not support as many operators

Fallback to CPU execution if node cannot be run using NNAPI

Available in ORT v1.6

- December 2020

Questions and Feature Requests

Please reach out to the ONNX Runtime team

- https://github.com/microsoft/onnxruntime/discussions