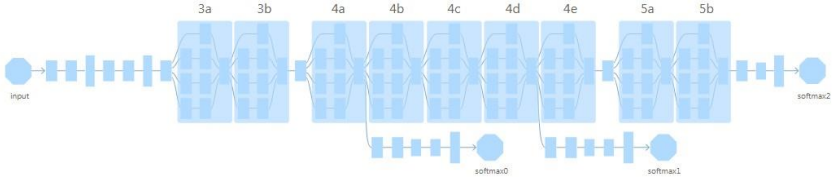
Google TensorFlow.js

浙江大学城市学院 ZHEJIANG UNIVERSITY CITY COLLEGE



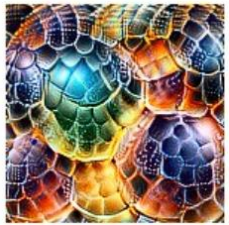

# 深度学习应用开发
## 基于TensorFlow的实践

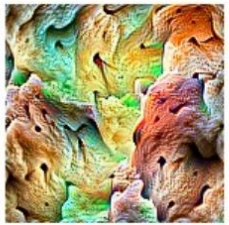
吴明晖 李卓蓉 金苍宏

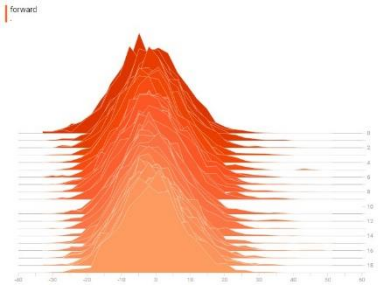
浙江大学城市学院

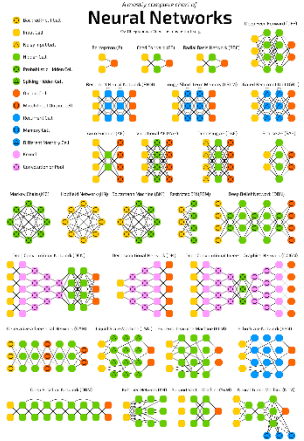
计算机与计算科学学院

Dept. of Computer Science
Zhejiang University City College



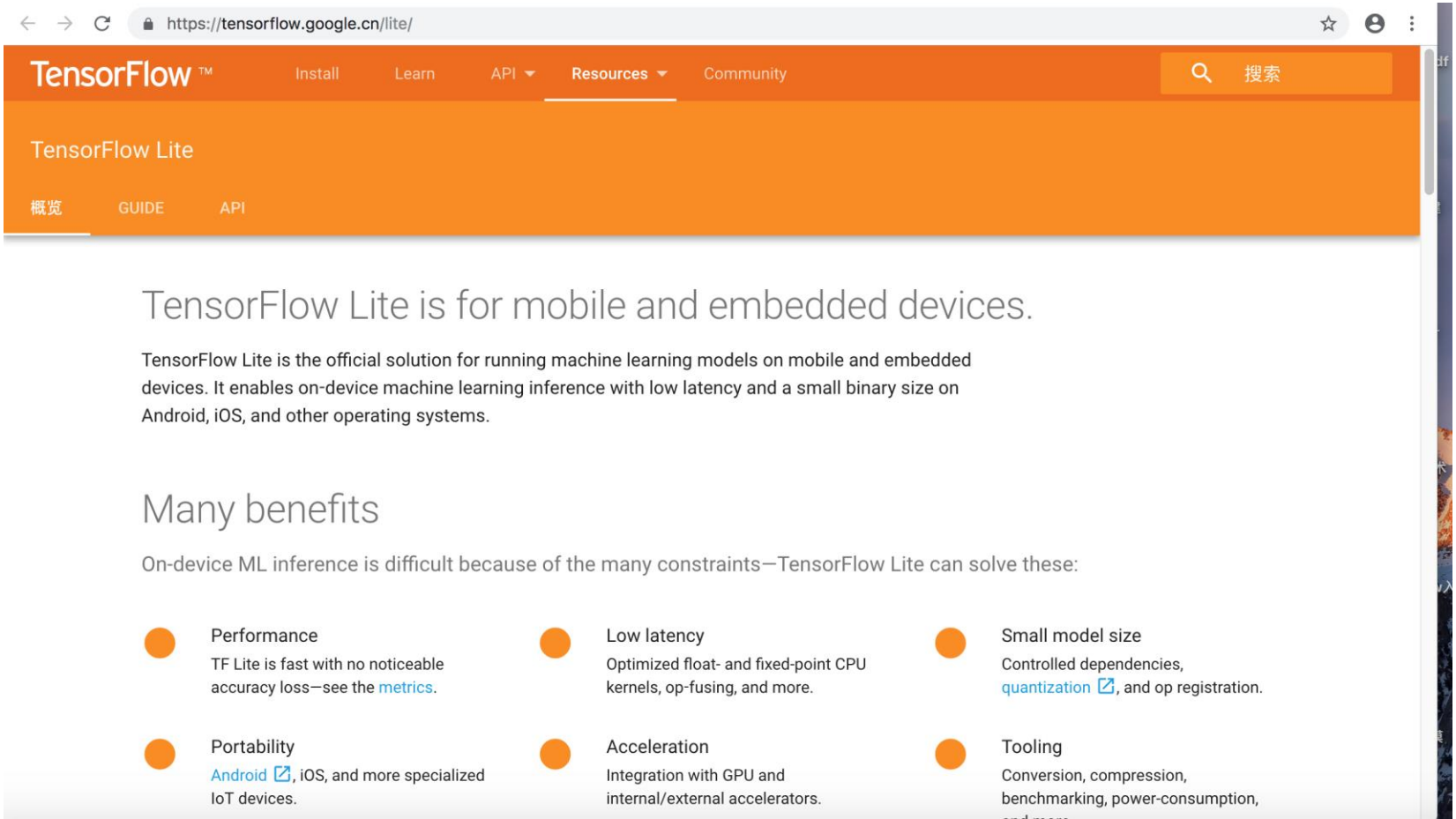




# TensorFlow Lite 开发实践

# TensorFlow Lite基本介绍


https://tensorflow.google.cn/lite/
TensorFlow ™
Install
Learn
API
Resources
Community
搜索
TensorFlow Lite
概览
GUIDE
API
TensorFlow Lite is for mobile and embedded devices.
TensorFlow Lite is the official solution for running machine learning models on mobile and embedded devices. It enables on-device machine learning inference with low latency and a small binary size on Android, iOS, and other operating systems.
Many benefits
On-device ML inference is difficult because of the many constraints—TensorFlow Lite can solve these:
Performance
TF Lite is fast with no noticeable accuracy loss—see the metrics.
Low latency
Optimized float- and fixed-point CPU kernels, op-fusing, and more.
Small model size
Controlled dependencies, quantization, and op registration.
Portability
Android, iOS, and more specialized IoT devices.
Acceleration
Integration with GPU and internal/external accelerators.
Tooling
Conversion, compression, benchmarking, power-consumption,

# 什么是TensorFlow Lite

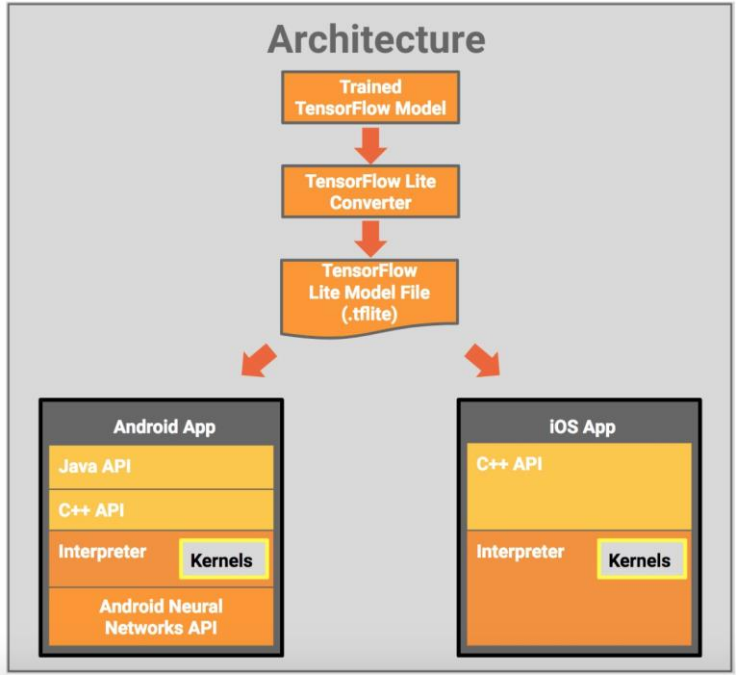


- TensorFlow Lite是TensorFlow在手机、嵌入式设备中的轻量级解决方案。
- 在设备上进行机器学习
- 内核更小、延时更低
- 支持Andorid和iOS系统

# TensorFlow Lite框架



- **Java API**: A convenience wrapper around the C++ API on Android.
- **C++ API**: Loads the TensorFlow Lite Model File and invokes the Interpreter. The same library is available on both Android and iOS.
- **Interpreter**: Executes the model using a set of kernels. The interpreter supports selective kernel loading; without kernels it is only 100KB, and 300KB with all the kernels loaded. This is a significant reduction from the 1.5M required by TensorFlow Mobile.
- On select Android devices, the Interpreter will use the Android Neural Networks API for hardware acceleration, or default to CPU execution if none are available.



# TensorFlow Lite优势



- 核心操作移植到移动端，开发者只需要关注高层接口
- 使用FlatBuffers模型格式、TF Lite更小
- On-device interpreter 在移动端运行更快
- TF converter把TF模型转换成TF Lite模型

# Flat Buffers介绍

- FlatBuffers is an efficient cross platform serialization library for C++, C#, C, Go, Java, JavaScript, Lobster, Lua, TypeScript, PHP, Python, and Rust.
- It was originally created at Google for game development and other performance-critical applications.
- FlatBuffers按照数据组织格式生成Byte数组，并没有做数据解析。因此，比JSON快10倍以上.
- Flat Buffers分为vtable和数据区，存放偏移值和数据.

# TL lite操作和API

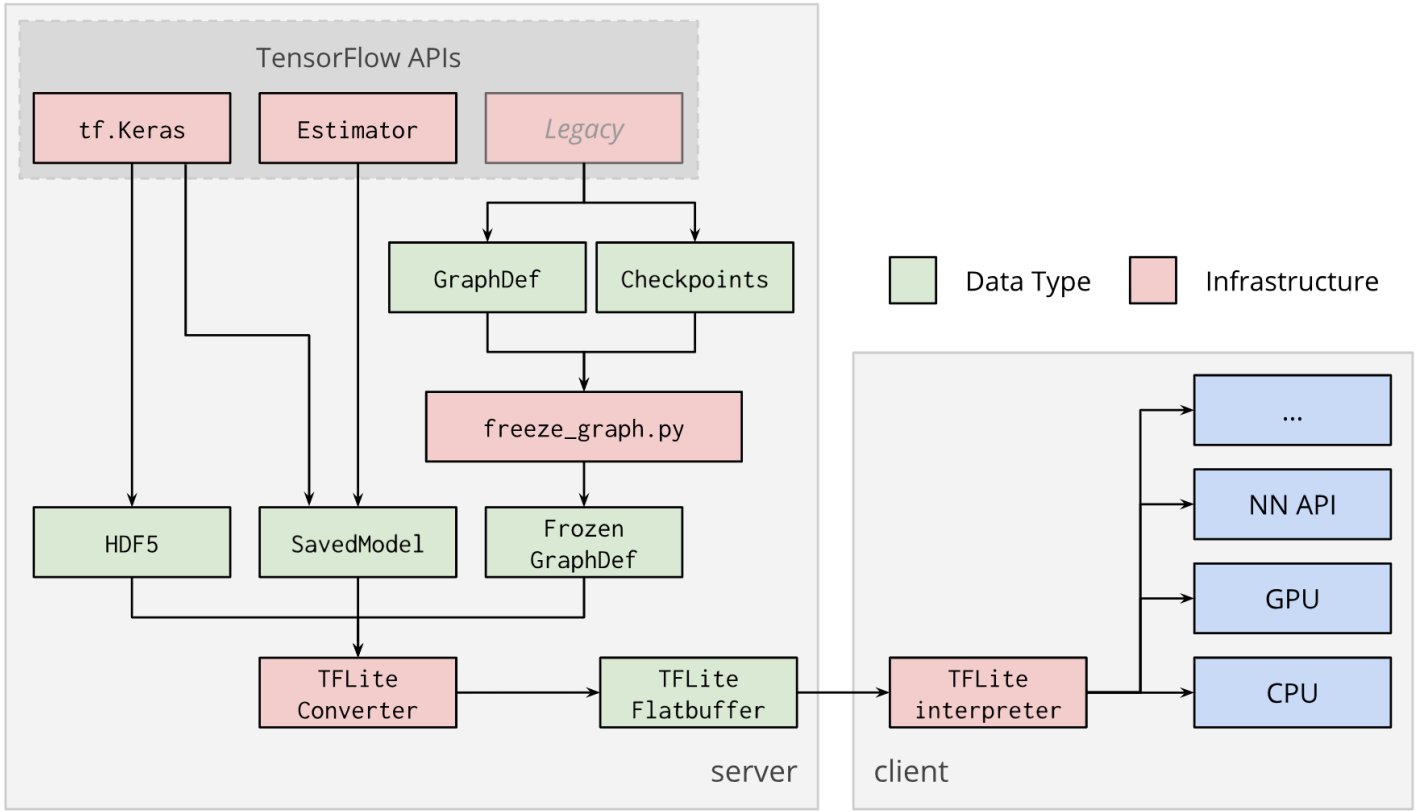
- APIs可以用C++或者Java调用，也有python接口
- 支持基本数据格式float, int, long, byte 和string类型
- TF Lite支持一部分的TF操作，但是支持用户自己构建操作

# TF Lite模型生成步骤

TF训练出的模型不能直接在TF Lite上运行，需要转化成.tflite文件。

- 在算法训练的脚本中保存图模型文件（GraphDef）和变量文件（CheckPoint）。
- 利用freeze_graph工具生成frozen的graphdef文件。
- 利用converter，生成最终的tflite文件。

# TF Lite模型生成步骤



# TF Lite Converter

- 把TF模型转换成TF Lite模型
- tf.lite.TFLiteConverter

使用代码转换的主要方法

1. TFLiteConverter.from_session()
2. TFLiteConverter.from_saved_model()
3. TFLiteConverter.from_keras_model_file()

# 从tf.Session导出GraphDef

```
import tensorflow as tf

img = tf.placeholder(name="img", dtype=tf.float32, shape=(1, 64, 64, 3))
var = tf.get_variable("weights", dtype=tf.float32, shape=(1, 64, 64, 3))
val = img + var
out = tf.identity(val, name="out")

with tf.Session() as sess:
  sess.run(tf.global_variables_initializer())
  converter = tf.lite.TFLiteConverter.from_session(sess, [img], [out])
  tflite_model = converter.convert()
  open("converted_model.tflite", "wb").write(tflite_model)
```

# 从文件中导出GraphDef

- 支持.pb和.phtxt格式文件

```
import tensorflow as tf

graph_def_file = "/path/to/Downloads/mobilenet_v1_1.0_224/frozen_graph.pb"
input_arrays = ["input"]
output_arrays = ["MobilenetV1/Predictions/Softmax"]

converter = tf.lite.TFLiteConverter.from_frozen_graph(
  graph_def_file, input_arrays, output_arrays)
tflite_model = converter.convert()
open("converted_model.tflite", "wb").write(tflite_model)
```

# 从keras文件中导出GraphDef

- 需要安装h5py包，http://docs.h5py.org/en/latest/build.html

```
import tensorflow as tf

converter = tf.lite.TFLiteConverter.from_keras_model_file("keras_model.h5")
tflite_model = converter.convert()
open("converted_model.tflite", "wb").write(tflite_model)
```

# 使用代码进行模型转换

- 需要安装h5py包，http://docs.h5py.org/en/latest/build.html

```
import tensorflow as tf

converter = tf.lite.TFLiteConverter.from_keras_model_file("keras_model.h5")
tflite_model = converter.convert()
open("converted_model.tflite", "wb").write(tflite_model)
```

# 使用命令行进行模型转换

```
jincangongdeMBP:~ jincanghong$ tflite_convert
usage: tflite_convert [-h] --output_file OUTPUT_FILE
                      (--graph_def_file GRAPH_DEF_FILE | --saved_model_dir SAVED_MODEL_DIR | --keras_model_file KERAS_MODEL_FILE
                      [--output_format {TFLITE,GRAPHVIZ_DOT}]
                      [--inference_type {FLOAT,QUANTIZED_UINT8}]
                      [--inference_input_type {FLOAT,QUANTIZED_UINT8}]
                      [--input_arrays INPUT_ARRAYS]
                      [--input_shapes INPUT_SHAPES]
                      [--output_arrays OUTPUT_ARRAYS]
                      [--saved_model_tag_set SAVED_MODEL_TAG_SET]
                      [--saved_model_signature_key SAVED_MODEL_SIGNATURE_KEY]
                      [--std_dev_values STD_DEV_VALUES]
                      [--mean_values MEAN_VALUES]
                      [--default_ranges_min DEFAULT_RANGES_MIN]
                      [--default_ranges_max DEFAULT_RANGES_MAX]
                      [--post_training_quantize] [--drop_control_dependency]
                      [--reorder_across_fake_quant]
                      [--change_concat_input_ranges {TRUE,FALSE}]
                      [--allow_custom_ops] [--target_ops TARGET_OPS]
                      [--dump_graphviz_dir DUMP_GRAPHVIZ_DIR]
                      [--dump_graphviz_video]
tflite_convert: error: the following arguments are required: --output_file
```