

installing on personal machine:

1. `conda create -n tensorflow 1.10` # create new vM in conda
1-1. Proceed — 'Yes'
2. `conda activate tensorflow 1.10`
3. `pip3 install tensorflow jupyter matplotlib`
4. `jupyter notebook`

Workshop 5

COMP90051 Machine Learning

Semester 2, 2018



Learning Outcomes

At the end of this workshop you should be able to:

1. explain the fundamental characteristics of the TensorFlow computation model
2. implement logistic regression using low-level TensorFlow APIs (Worksheet 5)

*Compare w/ week 3 implementation
using Numpy*

What is TensorFlow?

“TensorFlow™ is an open source software library for high performance numerical computation.”

– www.tensorflow.org

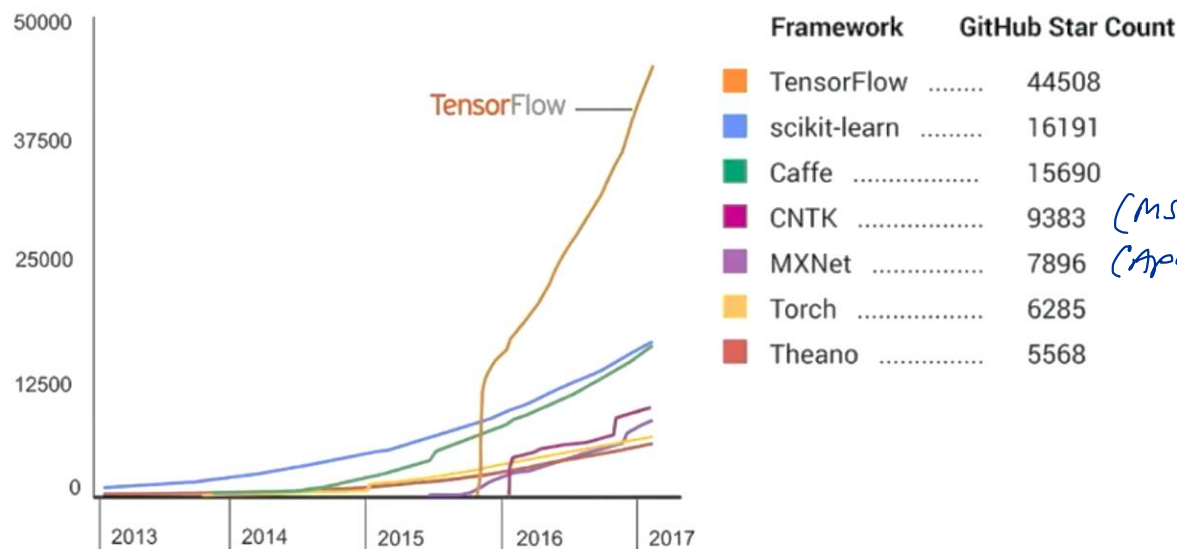
- Project started internally at Google *Brain*
- Open-sourced under the Apache 2.0 License in Nov 2015
- Runs on CPUs, GPUs, TPUs
and can run in parallel



Why TensorFlow?

*implement
novel architectures*

- Large community of users: easy to get help
- Strikes a good balance between flexibility and scalability



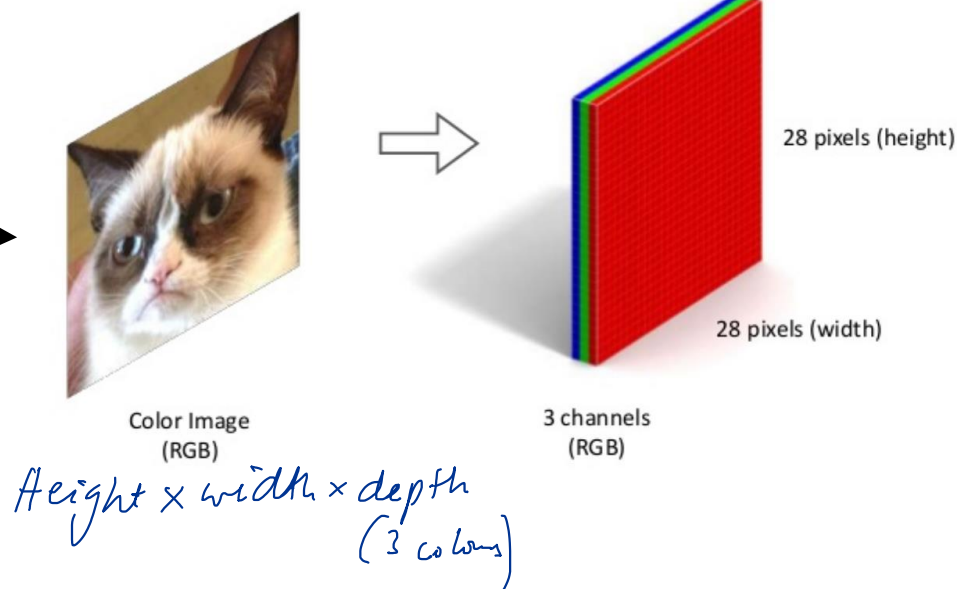
*(MS)
(Apache)* → *very
scalable,
but not
flexible.*

Source: Jiang Jun, GDG-Shanghai TensorFlow Dev Summit 2017

TensorFlow Basics

What's a tensor?

- For computer scientists:
“a multidimensional array”
- [Note: richer meaning in maths/physics]
- Examples:
 - * 0-d tensor: a scalar
 - * 1-d tensor: a vector
 - * 2-d tensor: a matrix
 - * 3-d tensor
 - * ... *(color image)*

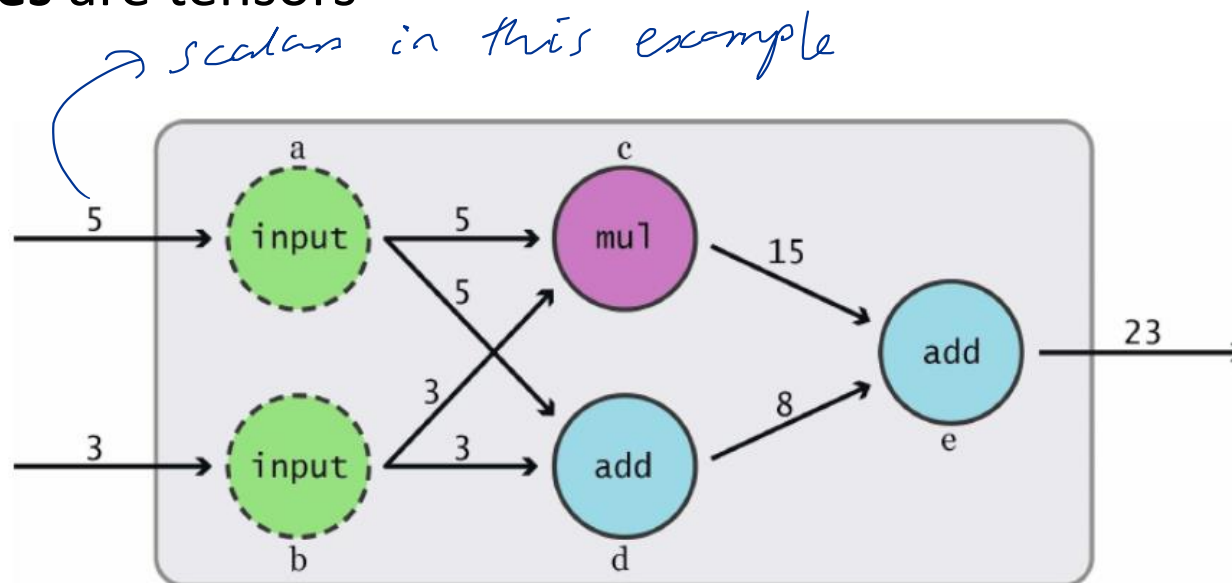


Data flow graphs

- Computation model adopted by TensorFlow

- A directed graph where

- * **Nodes** are operators/variables/constants *(functions or inputs into the graph)*
- * **Edges** are tensors



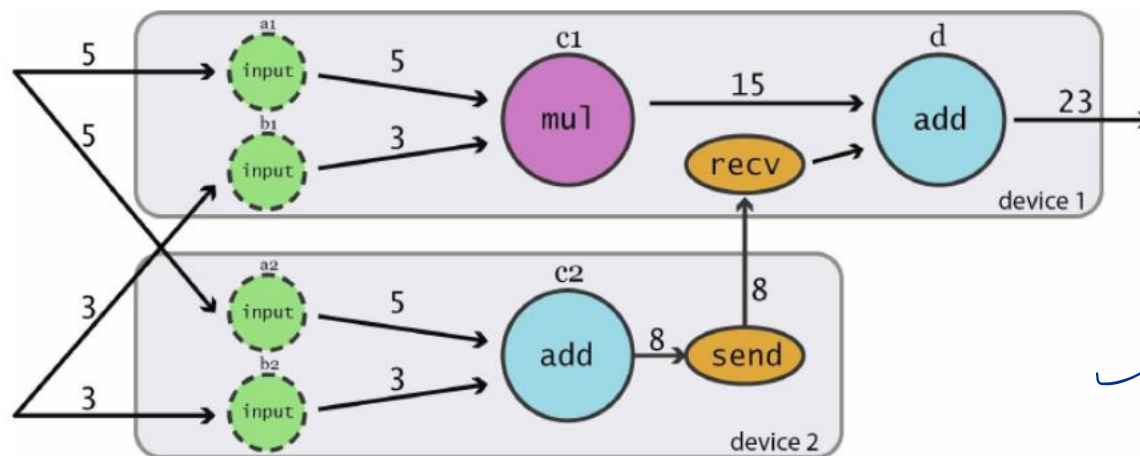
Source: Abrahams et al. *TensorFlow For Machine Intelligence*. 2016

Data flow graphs

naturally corresponds to N.N. notation/diagrams

- Advantages:

- * **Connection to NNs**: also represented as directed graphs
- * **Automatic differentiation**: break up computation into small, differentiable parts
- * **Distributed computation**: can split across CPUs, GPUs, TPUs

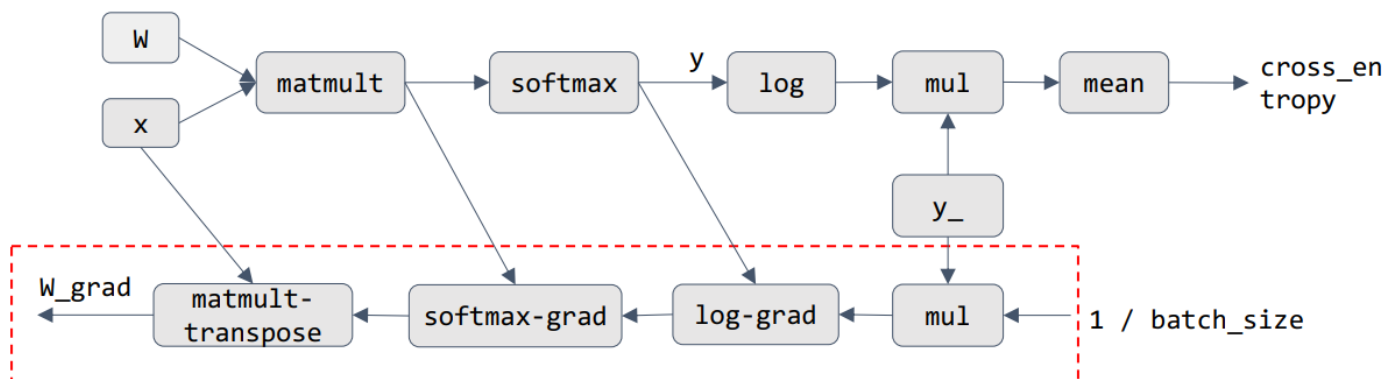


Source: Abrahams et al. *TensorFlow For Machine Intelligence*. 2016

Automatic differentiation

- Given a computation graph f , can automatically generate a *new* computation graph that computes ∇f
- How? Traverse the graph in reverse, replacing each node op by the corresponding gradient op
- Automatic so don't need to worry about it!**


it's all
coded in
C++ behind
the
scenes
so is
very efficient



Basics of graph building

- Create tensors for data input using `tf.placeholder()` *(input node)*
- Create tensors for parameters (to be optimised) using `tf.get_variable()` or `tf.Variable()` *i.e. weights / bias*
- Build up operations on tensors:
 - * Simple: e.g. `a + b`, `tf.matmul(a, b)`
 - * Composite: e.g. `tf.layers.dense(...)`
** Convolution layers.*

Then use a **session** to execute operations on the graph



Might change
with eager mode

Session, run, initialization

- Need a Session to compute outputs of the graph. Can call `tf.Session()` to create one.
- Once you've got a session, use the `run()` method to perform computations:
 - ↳ returns tensors from the graph*
 - * A `feed_dict` is required for placeholders
 - * Only runs subgraphs that lead to outputs you've requested
 - * Can get multiple outputs at once
- If you've defined Variables you need to initialise them. Can use:
 - * `sess.run(tf.global_variables_initializer())`

A simple example

*constant tensors,
remain in memory.*

```
import tensorflow as tf
```

```
tensor  
a = tf.constant(3)
```

```
b = tf.constant(5)
```

```
c = tf.add(a, b)
```

```
with tf.Session() as sess:
```

```
    print(sess.run(c))
```

Operation doesn't
run at define time—
just sets up the graph

Session allows us to
use the defined graph

Requesting the
value of variable c

*return
to Python session.*

starts a session.

High-level APIs on top of TensorFlow

- Keras [tf.keras]

- * High-level API for ANNs
- * Now part of TensorFlow core *(officially)*
- * Supports other (non-TF) backends



- Sonnet

- * Another high-level API for ANNs
- * Supported by DeepMind *(Google) → suits researchers on DeepMind*



- tf.Estimator [tf.estimator.Estimator]

- * API for training, evaluation, prediction
- * Recommended to replace skflow (offered a similar interface to scikit-learn, now deprecated) *→ baby TensorFlow*

Resources

- Official tutorials: <https://www.tensorflow.org/tutorials/>
- Official API docs: https://www.tensorflow.org/api_docs/
- Stanford CS20 course website:
<http://web.stanford.edu/class/cs20si/syllabus.html>
- Books:
 - * Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn and TensorFlow*. (2017)
 - * Bharath Ramsundar and Reza Bosagh Zadeh. *TensorFlow for Deep Learning: From Linear Regression to Reinforcement Learning*. (2018)

TensorFlow on the lab machines

- Open *Start* → *Anaconda3 (64-bit)* → *Anaconda Prompt*
- In the prompt, run the following commands:
 - > `cd "C:\Users\%USERNAME%\Downloads"`
 - > `mkdir workshop05`
 - > `cd workshop05`
 - > `pip install -t . tensorflow "protobuf<3.6.1"`
 - > `jupyter notebook`
- Copy Worksheet 5 into the workshop05 directory
- Open Worksheet 5 from within Jupyter

Note: This is a workaround installation method due to restrictions on the lab VM. machines. On your own device, we recommend following the installation instructions at <https://tensorflow.org/install/>

latest version 1.1.0
recommend install on