TensorFlow: A system for Large-Scale Machine Learning

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, *Google Brain*

Presentation by Antonia Boca | R244 | 19/10/2022

Work at the time

- Good single-machine frameworks:
  - High performance: Caffe
  - High flexibility: Torch, Theano

- Good multiple-machine frameworks:
  - DryadLINQ, Spark
    - Low flexibility: data must be immutable – ML training becomes slow
  - MXNet
    - Similar to TensorFlow
    - *Parameter server* architecture – cannot do sparse gradient updates
  - **DistBelief**
    - Does not scale *down* well

Motivation

- TensorFlow comes from the Google Brain team
- **DistBelief** is its predecessor
  - cannot easily define new types of layers in NN architectures
  - Cannot modify the optimisation algorithm (SGD)
  - Cannot modify the training algorithm (pipeline works only for Feedforward Neural Networks)

- Goal: build a framework that is both **flexible** and **scalable**
  - Platform-agnostic: can scale up to any number/type of device
  - Offer flexibility in the design of ML pipelines – create an API for popular programming languages
  - Use the same programming language for ML design and distributed systems design

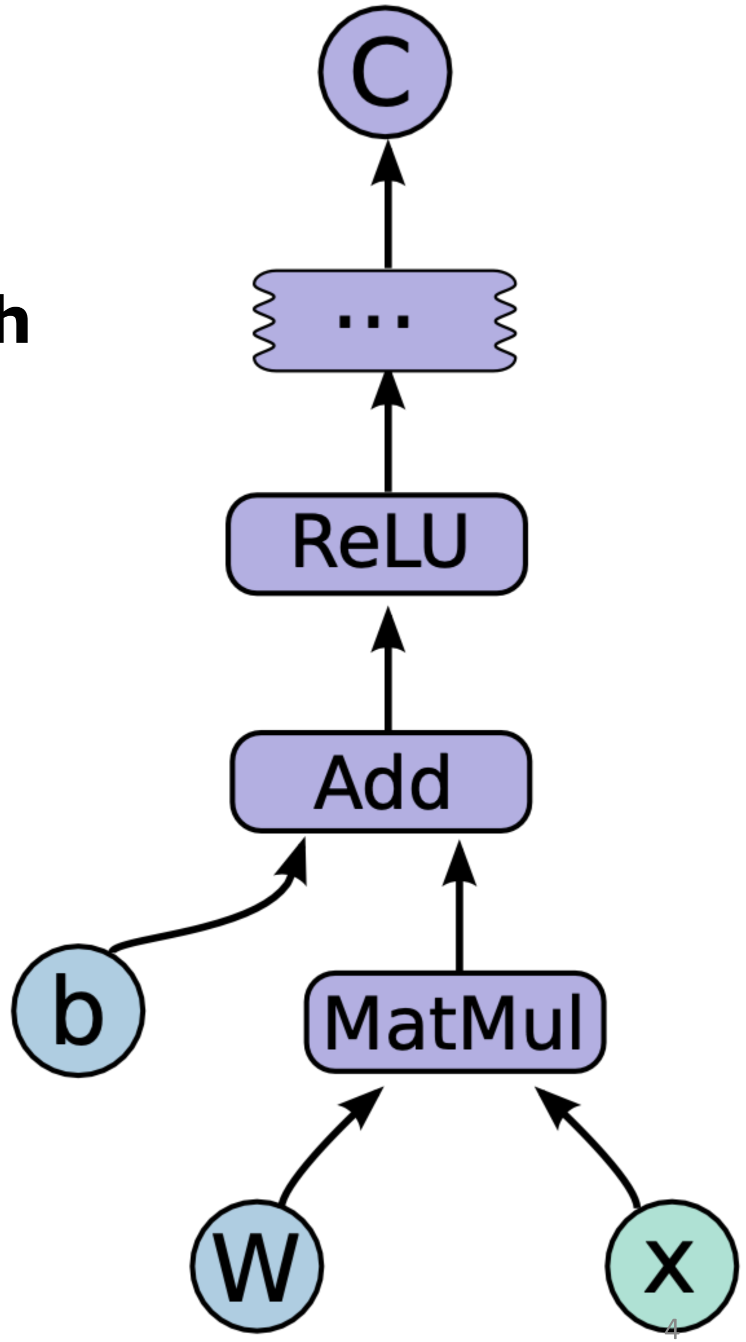
Design principles (1)

- Computation described by a **dataflow graph**
  - Nodes represent **operations**
  - Edges represent **dependencies**
  - `E.g. ReLU(b + W*x)`

- Data flows through the graph using **tensors**
  - Typed, multi-dimensional arrays

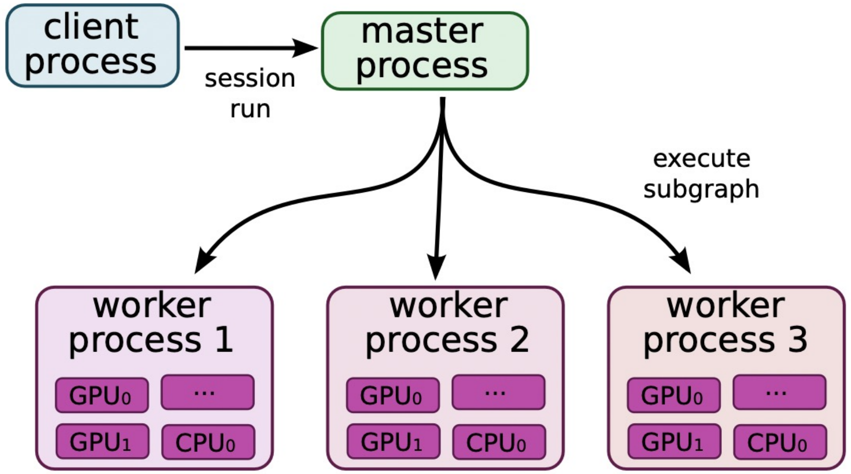
- TensorFlow automatically builds a gradient graph for the Backpropagation algorithm

- TensorFlow optimises the dataflow graph (e.g. with common subexpression elimination)



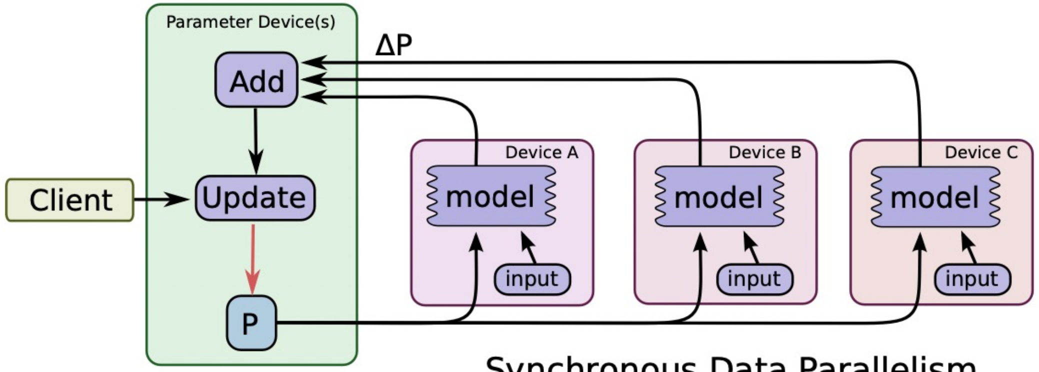
Design principles (2)

- Client communicates with a *master*
- *Master* communicates with worker processes
- Workers control devices (CPUs, GPUs, TPUs etc.)
- Devices get a subgraph of the initial dataflow graph
  - Each device has its **own implementation (called kernel)** of the operation to execute
- Fault tolerance: user-level checkpointing



Distributed execution

- Greedy heuristic used to choose which node to assign to which device
- Workers send data across only once for multiple nodes on a different worker [TensorFlow has **weak consistency guarantees**]
- (A)synchronous replica coordination
  - Can have synchronous ML training
  - TensorFlow proactively prepares backup workers in case stragglers exist



Synchronous Data Parallelism

Key innovations

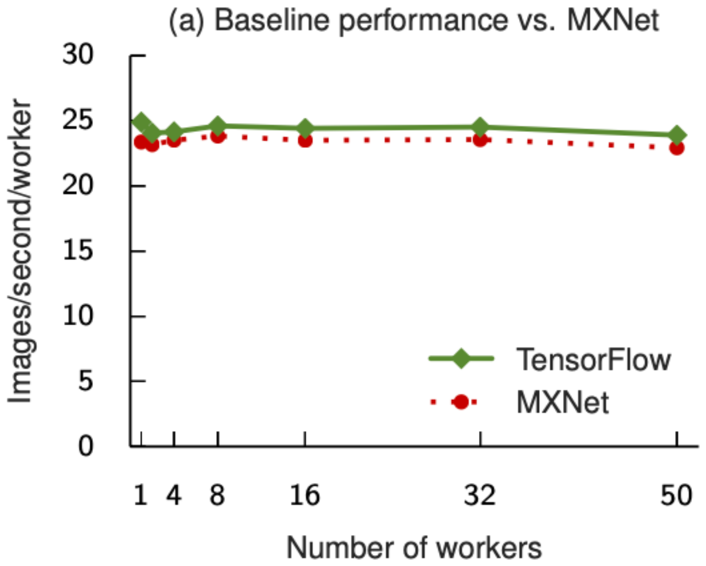
- Dynamic control flow
  - Can have conditional and iterative control flow – now possible to implement RNNs
- Nodes represent **single operations** and can hold and update state
  - Dataflow systems at the time: nodes represent functional computation on immutable data
  - **Abstracted computation kernels** for heterogeneous distributed systems
- Optimisation – can experiment with new algorithms
- Ability to scale up and down
- Really cool visualization system: **TensorBoard**

Experimental results

- Single-machine benchmarks
  - TensorFlow similar to Torch because they use the same **matrix multiplication library**

Library	Training step time (ms)			
	AlexNet	Overfeat	OxfordNet	GoogleNet
Caffe [38]	324	823	1068	1935
Neon [58]	87	**211**	**320**	**270**
Torch [17]	**81**	268	529	470
TensorFlow	**81**	279	540	445

- Multi-machine benchmarks
  - TensorFlow compared to MXNet for the **Inception-v3** model



- **Key takeaway:** TensorFlow has similar performance to its competitors, but it is much more flexible!

Limitations

- Algorithm to schedule nodes uses a greedy heuristic
- Paper does not show how fast ML training converges
- Paper does not show experimental results for RNNs
- Training for Reinforcement Learning is still too limited
- Not suitable for applications with strong consistency requirements
- Training slower than in other frameworks because of the use of cuDNN library for matrix multiplication
- Does **not** have **fine-grained control** over execution order and memory requirements

Impact & Future

- Widely adopted ML framework
- Used in hundreds of research papers
- Downloaded by millions of users

- Nowadays, TensorFlow is losing ground to PyTorch and the new MLGO

Any questions?