# ML infrastructure Panel

**Moderator: Peter Braam**
**Panelists: Maria Girone, Stephen Pawlowski, Yujia Li, Bojan Nikolic**

# 10 mins about Google's TensorFlow Infrastructure

# Background & Developments

Google needed a radically new infrastructure to avoid **doubling** their data centres in the face of modest AI use.

Hundreds of projects have been and will be pursuing ML methodology in Google: **development productivity** is very important.

Google released TensorFlow in 2015 (a 2nd design following DistBelief). TensorFlow's growth and feature far outpace other efforts. 3 years later there are, for example, 3 generations of TensorFlow chips.

It appears to be one of the most successful software - systems - hardware projects I have witnessed in my life.

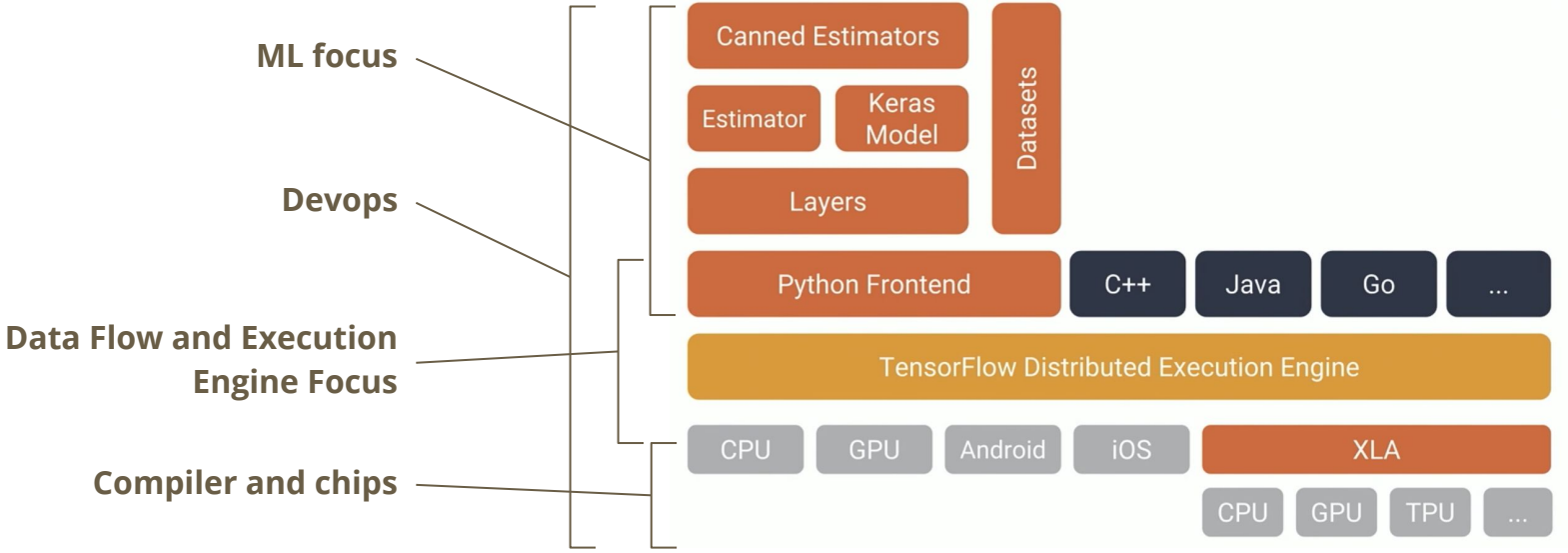
# Programming & Productivity

**High level API's** like Keras allow rapid prototyping of ML models (this is largely maths, not programming). Automatic differentation.

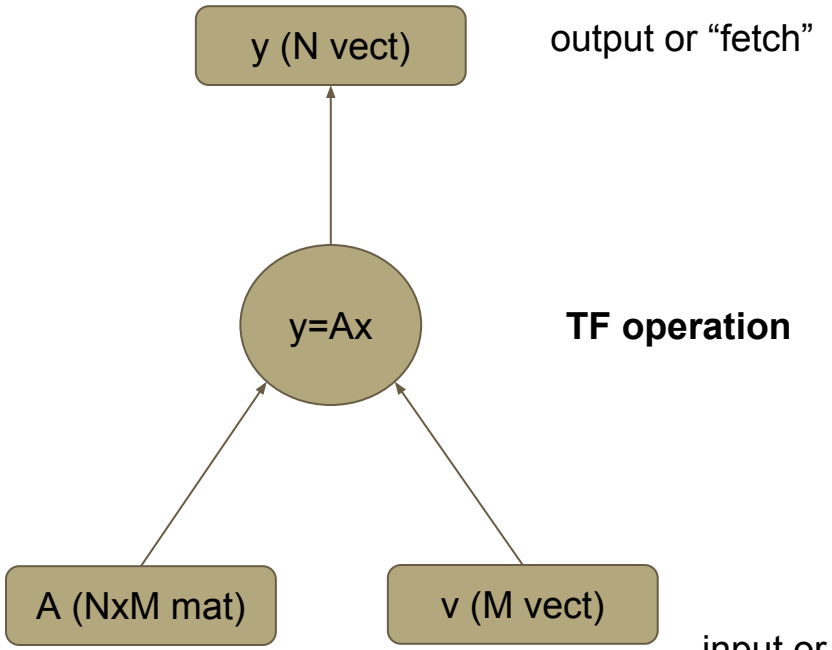
**Debugging and profiling** tools exist, such as tensorboard and a data flow debugger.

**One code base** can be used for development, training, evaluation, inference and snapshotting, and runs on mobile devices through specialized large clusters.

**The architecture reflects a strong separation of concerns**



# Tensorflow Core




Data Flow model with extremely rich features.

Expressions in programming languages define data flow graphs from call graphs and arguments

TF treats graphs **declaratively,** i.e. they are defined but not executed at the same time.

Execution of a graph takes the “fetch”, backtracks its dependencies and computes in parallel.

TF Graphs can be automatically split for distributed execution on multiple devices.

TF Operations Reflect Domain Specific Aspects found througtrout TensorFlow

# Portability with XLA Compiler

- ➔ reorganize a tensorflow graph to
  - ◆ fuse operations, eliminate unused and identity ops, bind constants
  - ◆ introduce communication, and graph partitioning
- ➔ create compiled code for the fused operations
  - ◆ JIT: just in time (during execution) to take full advantage of the sizes of the tensors
  - ◆ AOT: ahead of time to create a standalone binary
- ➔ optimizations:
  - ◆ tiling sizes, threading, data alignment, perform padding, minimize communications, adapt queue lengths
- ➔ compilation targets: CPU's (mobile - server), GPU or TPU processors & clusters of these

# TPU platform (v3)

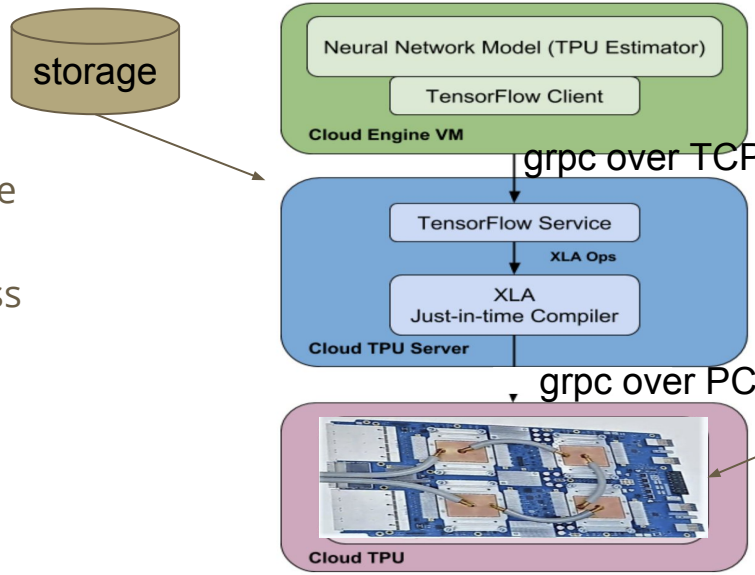
TPU: PCI accelerator card. TPU chips have **systolic** MMU (matrix multiply unit) reducing memory accesses by ~100x: pass data between small processing units.

1 POD has 256 nodes, ~100 PF/sec and 5 PB/sec memory bandwidth. Node has 20TB/sec memory bw. (2nd biggest HBM customer)

Energy efficiency up to 10/100x that of GPU/CPU.

Reuse: **XLA** compiles applicable part of a TF program to machine code. **gRPC** moves data between control node & TPU.

Is this the next HPC platform?

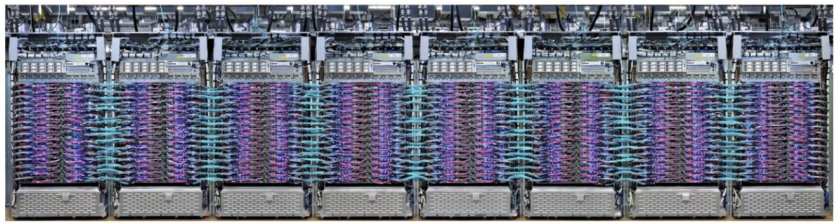



TPU system infrastructure

TPU PCI card.
4 TPU's/card
4 MMU/TPU
1024 TPU/pod

**Operations per clock cycle**

| | |
|---|---|
| CPU | 10's (cores) |
| CPU vectorized | 1000 (core x vector length) |
| GPU | 10K 's |
| TPU | 128K (TPU v1) |



PODS
clusters with TPU's

# Lessons Learned

Significant cost benefits make software and custom HW projects viable solutions

Replicating an effort of this stature is extremely difficult

Domain specific solutions hold a lot of promise.

# Each panelist to share < 3 mins thought about

- Is infrastructure for ML for CERN and SKA special? In what way?
- Will new algorithms be important? What hardware would help, what can be done with current ideas about hardware.
- What qualities of ML infrastructure are highest risk?
- Next steps in development efficiency - both maths and coding aspects
- How can NOC architectures and number formats improve Data movement?
- Other significant developments in this area?