Exploratory meeting on enabling AI in HEP: Fast-ML/FPGA Discussion

Alex, Alex, Dave, Sudan

Charge

- What are the HEP use cases for fast-ML/FPGAs and what is the current status
  - Trigger/DAQ applications: O(us) latency – “traditional” use on custom hardware
  - Wider use as accelerator for CPU based workloads – emerging – less UK involvement?
  - Status
    - First BDTs and NNs running in ATLAS and CMS hardware triggers deployed
    - Large range of examples planned for HL-LHC upgrade
    - Development pipelines for ML on FPGAs – established but further development needed
    - Next Gen trigger project at CERN platform for development (within ATLAS UK) of ML pipelines for FPGA

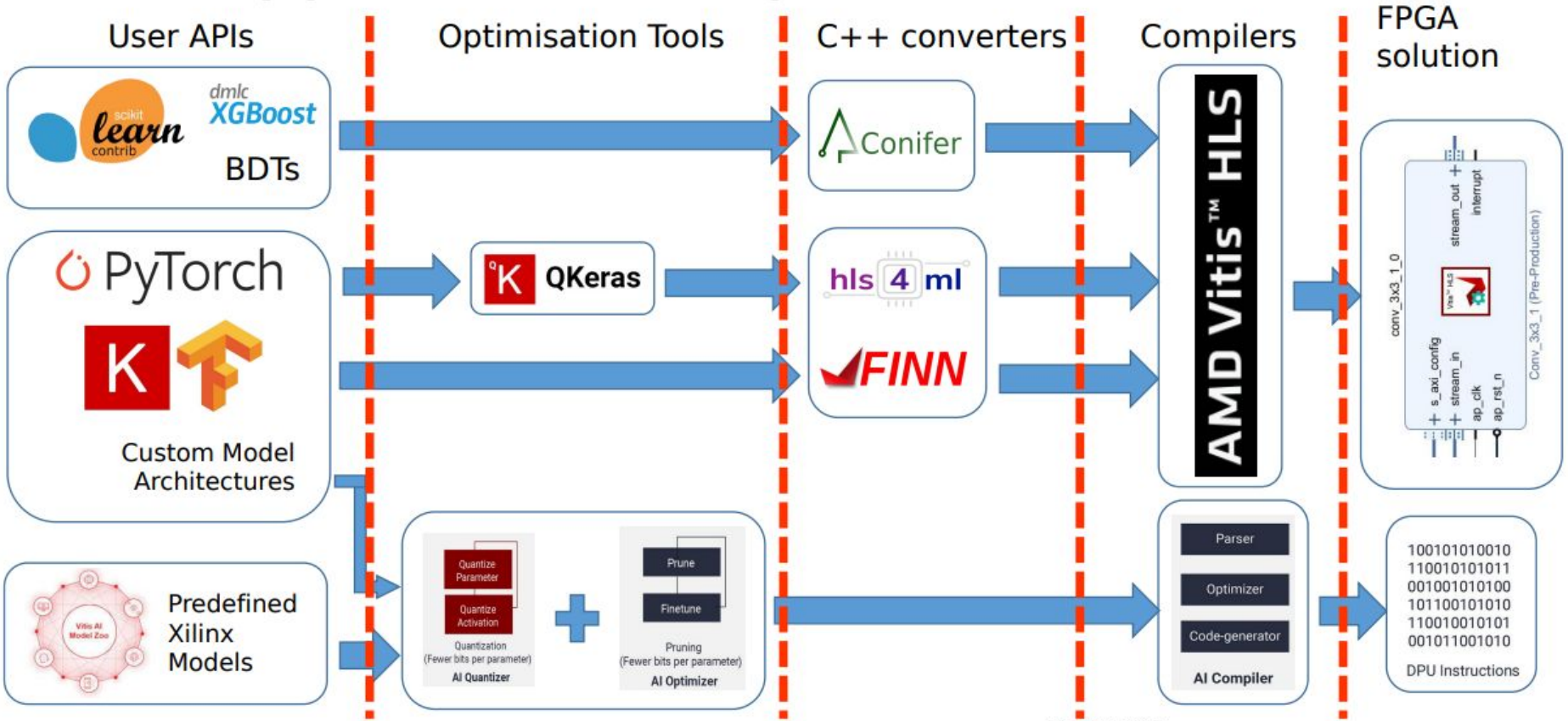
Charge

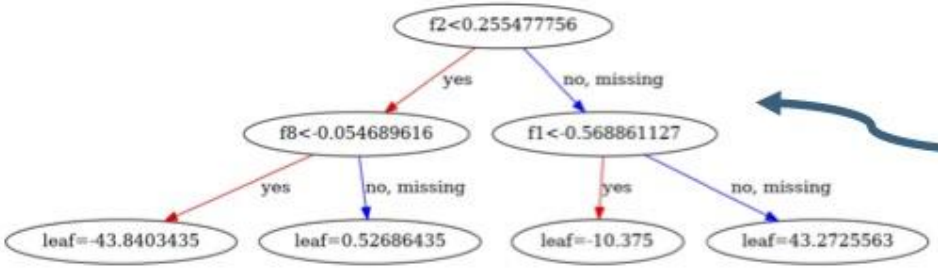
- What are the challenges/barriers that we are facing in this area (e.g. software, training, skills-capacity, hardware, etc.)?
  - Steep learning curve – albeit not as steep as VHDL
  - Understanding and tools e.g. hls4ml, pruning, quantisation
  - Bottleneck is (skilled, technical) effort – technical PhD funding, DRD funding ….
  - Consolidate UK skills across experiments – currently quite siloed
  - Evaluate industry directions e.g. Versal AI engines vs custom electronics (DRD7 etc.)
  - Challenges with obsolescence and software tools support
  - Explainability a key topic for confidence in tools and algorithms – in general but especially so in real time where data is lost

Charge

- What are the opportunities in this area for HEP (e.g. enhanced outcomes, wider connections, funding, industry engagement, knowledge exchange etc.)?
  - Highly trained people useful to science and UK generally – many people go on to data science and elsewhere e.g. CERN
  - Niche in AI – not well covered or supported by industry in general
  - Need to prepare case(s) to allow access to wider funding – examples from science & beyond
  - Industry connections – good connections in this area to key players and UK SMEs
  - Community tool hls4ml – UK contributes, plans driven by US towards formal basis
  - …

ML based pipelines for FPGA implementations

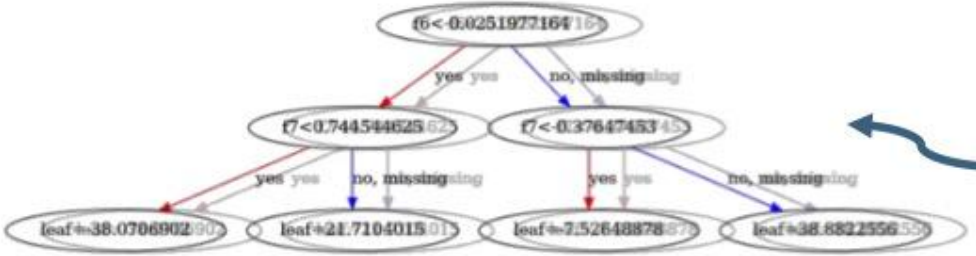




What we can do now in Conifer:

<u>Single scalar-leaf trees</u>

- Two-class classifiers
- 1-target regressors

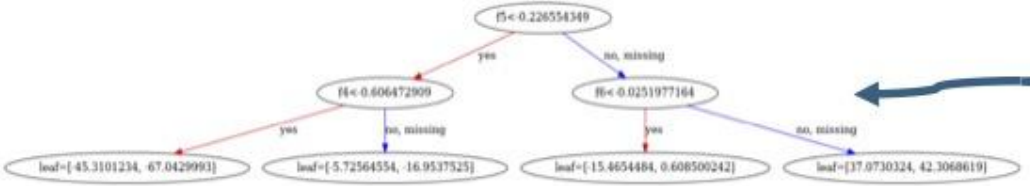


We would like multi-output trees!

- This can mean:

<u>One-output-per-tree</u>

- Each target requires a separate binary BDT
- Resources scales with $N_{targets}$
  - Large $N_{targets}$ = large resources



<u>Vector-leaf trees (preferred)</u>

- Targets share single common tree
- Leaf is vector of length $N_{targets}$
- Small resource overhead for adding extra targets
  - Large $N_{targets}$ ≠ large resources!