Optimization of ML-Based BSM triggering with Knowledge Distillation for FPGA implementation

Dr. Marco Lorusso [1]

[1]National Institute for Nuclear Physics - CNAF Division

20th March 2025

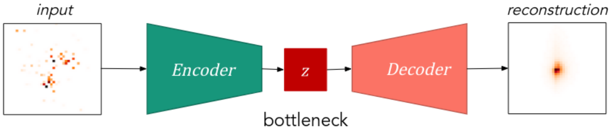
Introduction

- **Machine Learning** strongly considered to process the **increased** amount of **data** in next phases of LHC $\rightarrow$ major focus on **Artificial Neural Networks**;
- Trigger **latency and energy constraints** are quite **unique** $\rightarrow$ need for **specific software** development and strategies to deploy ANN **efficiently** on the **hardware** available on-site, like **FPGAs**.
- An important candidate for using ML for triggering is the research for **Beyond Standard Model** events:
    - Networks like **Autoencoders** are **unbiased** algorithm which can **select** events based on their degree of **abnormality**, **without** theoretical **priors**;
- Need to **optimize** and **compress** these kind of algorithm to make them **suitable** for **trigger** environments.

Anomaly Detection with AEs

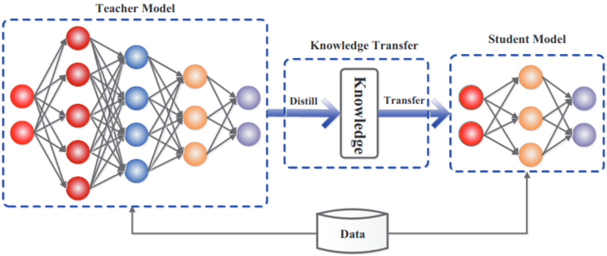
Auto-Encoders (AEs) are neural network models trained to **reconstruct** their **inputs**:

- The **encoder compress** the input into a smaller latent space.
- The **decoder reconstructs** from the compressed representation.



- Idea: **background** samples are associated to **low** anomaly scores;
- Anomalies can be either erroneous, rare or interesting events;
- This approach is **self-supervised**, requiring only background samples in the data;
- **No assumption** on the type of signal.

Knowledge Distillation

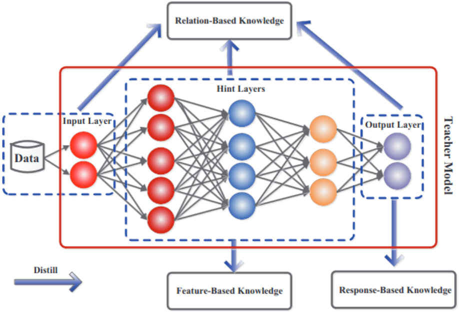


Source: *Knowledge Distillation: A Survey* (arXiv:2006.05525)

- Knowledge distillation trains a **smaller** "student" neural network to **emulate** the behavior of a **larger** "teacher" network. The teacher network, usually more complex and accurate, guides the student network to learn from its knowledge, enhancing generalization;
- The student network can match the teacher's performance with **fewer parameters** → Valuable for resource-constrained environments or where computational efficiency is vital.

Knowledge Distillation - Different Knowledge

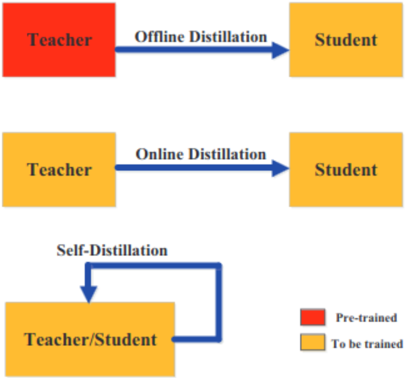
- Response-Based Knowledge: directly mimic the **final prediction** of the teacher model;
- Feature-Based Knowledge: **both** the output of the **last layer** and the output of **intermediate** are used as the knowledge to supervise the training of the student model;
- Relation-Based Knowledge: the student learns the **relations** between intermediate representations of data between layers of the teacher.



Source: *Knowledge Distillation: A Survey* (arXiv:2006.05525)

Knowledge Distillation - Different Learning Schemes

- Offline Distillation: the knowledge is transferred from a **pre-trained** teacher model into a student model;
- Online Distillation (co-training): **both** the teacher model and the student model are **updated simultaneously**, and the whole knowledge distillation framework is end-to-end trainable;
- Self-Distillation the **same networks** are used for the teacher and the student models. This can be regarded as a special case of online distillation.



Source: *Knowledge Distillation: A Survey* (arXiv:2006.05525)

The AD task

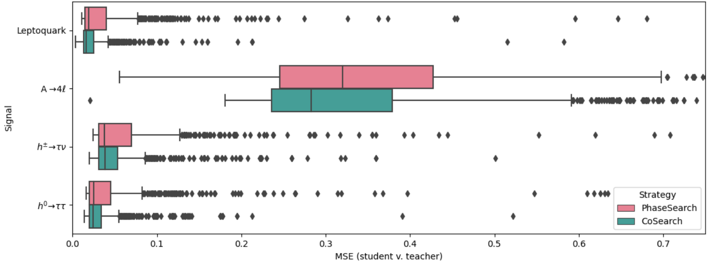
- Dataset: a typical p-p collision dataset pre-filtered requiring an electron or a muon with a $p_T > 23$GeV and a $|\eta| < 3$ (electron) and $|\eta| < 2.1$ (muon);
- Injected signals:
  - *Leptoquark* (LQ) with a mass of 80 GeV, decaying to a b quark and a τ lepton;
  - Neutral scalar boson (A) with a mass of 50 GeV, decaying to two off-shell Z bosons, each forced to decay to two leptons: $A \to 4$
  - Scalar boson with a mass of 60 GeV, decaying to two tau leptons: $h_0 \to \tau\tau$
  - A charged scalar boson with a mass of 60 GeV, decaying to a tau lepton and a neutrino: $h_\pm \to \tau\nu$
- Knowledge Distillation used to create a **smaller network** to fit in e.g. FPGAs which behaves similarly to the beefier network;
- The student should be optimized to learn from the teacher as much as possible, keeping in mind the **hardware restrictions** of deploying NN on FPGAs.

Student Optimization - What should we do first?

- 2 "dimensions" for optimizing a NN for FPGAs: architecture and quantization;
  - quantization ≈ converting all parameters (e.g. weights) to fixed-point numbers, better handled by FPGAs
- Is there a **difference** in searching for the **best candidate with or without** the **quantization process in mind**?

Strategy: first a simple hyperparameter search with no quantisation; then we repeat the hyperparameter search + quantisation search.

CoSearch v. PhaseSearch

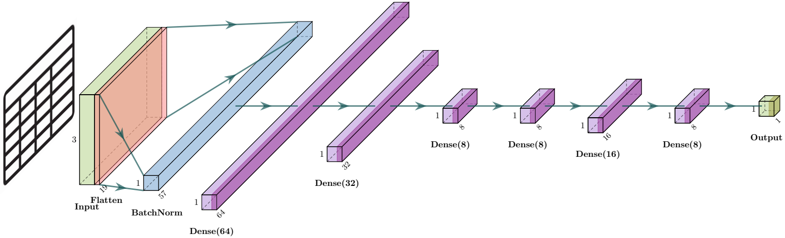


CoSearch — Architecture and Quantization parameter space searched simultaneously;

PhaseSearch — Quantization parameter space searched after optimal Architecture;

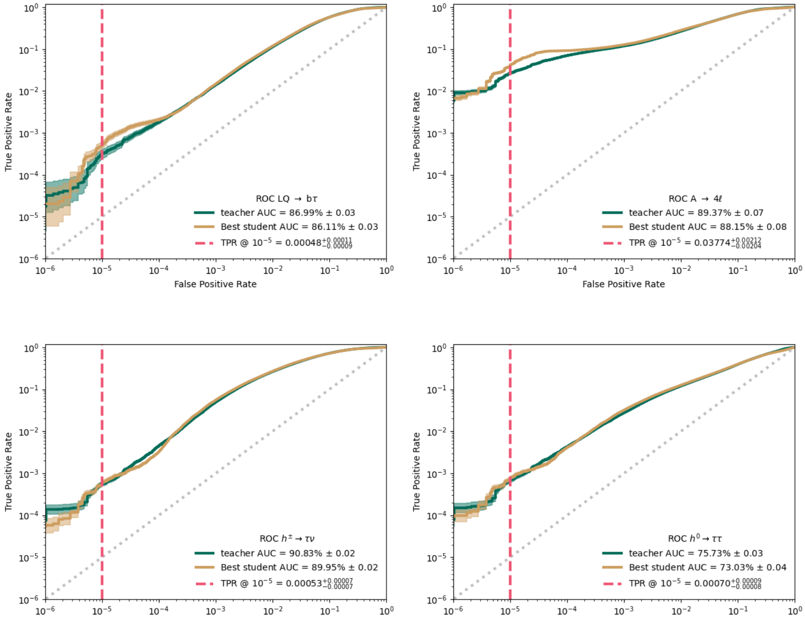
Results are consistent but **CoSearch** more peaked to lower values.

AD performance - Best student



- 57 input nodes;
- $\approx$ 10% teacher model parameters

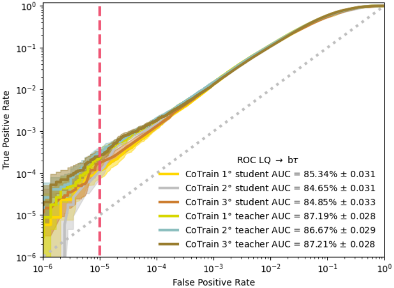
- Hidden layers:
  - 64 nodes, 16 bits per w/b (6 bits integer part);
  - 32 nodes, 16 bits per w/b (6 bits integer part);
  - 8 nodes, 16 bits per w/b (6 bits integer part);
  - 8 nodes, 8 bits per w/b (2 bits integer part);
  - 16 nodes, 16 bits per w/b (6 bits integer part);
  - 8 nodes, 16 bits per w/b (6 bits integer part);

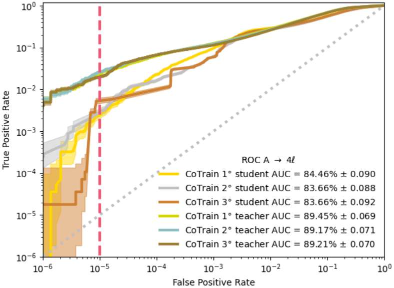

True Positive Rate
False Positive Rate
ROC LQ → bτ
teacher AUC = 86.99% ± 0.03
Best student AUC = 86.11% ± 0.03
TPR @ 10^{-5} = $0.00048^{+0.00011}_{-0.00009}$
ROC A → 4ℓ
teacher AUC = 89.37% ± 0.07
Best student AUC = 88.15% ± 0.08
TPR @ 10^{-5} = $0.03774^{+0.00212}_{-0.00204}$
ROC $h^{\pm} \to \tau\nu$
teacher AUC = 90.83% ± 0.02
Best student AUC = 89.95% ± 0.02
TPR @ 10^{-5} = $0.00053^{+0.00007}_{-0.00007}$
ROC $h^{0} \to \tau\tau$
teacher AUC = 75.73% ± 0.03
Best student AUC = 73.03% ± 0.04
TPR @ 10^{-5} = $0.00070^{+0.00009}_{-0.00008}$

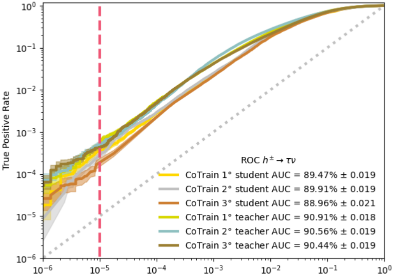
Testing Co-training Distillation

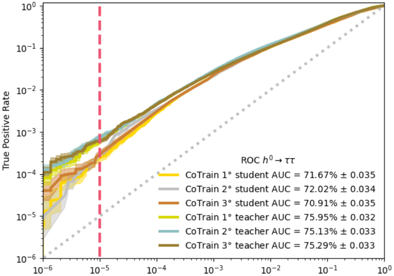
Co-training distillation sees the teacher and student training **at the same time**. The training step of this procedure sees multiple terms:

- The teacher objective is still to **perform the task** as best as possible;
- The student tries to **emulate** the teacher even in the learning procedure;
- In this way the pace of "emulating" and learning the task **can be tuned**;
- An element of noise can be added right in the objective function to avoid overfitting.


ROC LQ → bτ
CoTrain 1° student AUC = 85.34% ± 0.031
CoTrain 2° student AUC = 84.65% ± 0.031
CoTrain 3° student AUC = 84.85% ± 0.033
CoTrain 1° teacher AUC = 87.19% ± 0.028
CoTrain 2° teacher AUC = 86.67% ± 0.029
CoTrain 3° teacher AUC = 87.21% ± 0.028
True Positive Rate
False Positive Rate


ROC A → 4ℓ
CoTrain 1° student AUC = 84.46% ± 0.090
CoTrain 2° student AUC = 83.66% ± 0.088
CoTrain 3° student AUC = 83.66% ± 0.092
CoTrain 1° teacher AUC = 89.45% ± 0.069
CoTrain 2° teacher AUC = 89.17% ± 0.071
CoTrain 3° teacher AUC = 89.21% ± 0.070
True Positive Rate
False Positive Rate


ROC $h^{\pm} \to \tau\nu$
CoTrain 1° student AUC = 89.47% ± 0.019
CoTrain 2° student AUC = 89.91% ± 0.019
CoTrain 3° student AUC = 88.96% ± 0.021
CoTrain 1° teacher AUC = 90.91% ± 0.018
CoTrain 2° teacher AUC = 90.56% ± 0.019
CoTrain 3° teacher AUC = 90.44% ± 0.019
True Positive Rate

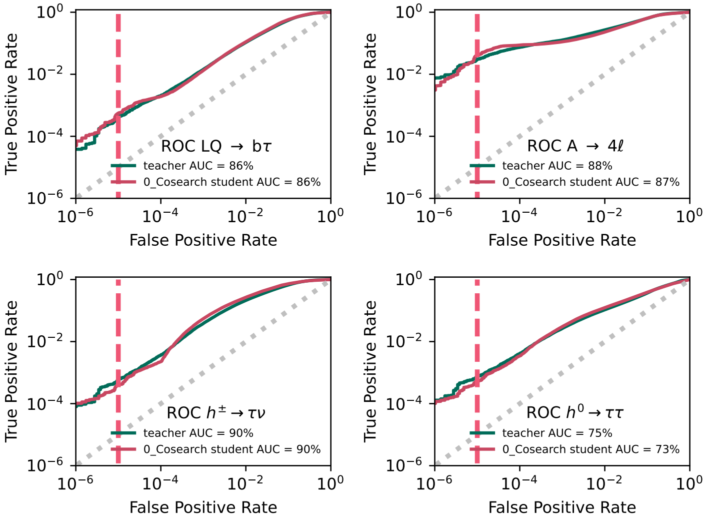

ROC $h^0 \to \tau\tau$
CoTrain 1° student AUC = 71.67% ± 0.035
CoTrain 2° student AUC = 72.02% ± 0.034
CoTrain 3° student AUC = 70.91% ± 0.035
CoTrain 1° teacher AUC = 75.95% ± 0.032
CoTrain 2° teacher AUC = 75.13% ± 0.033
CoTrain 3° teacher AUC = 75.29% ± 0.033
True Positive Rate

Conclusions

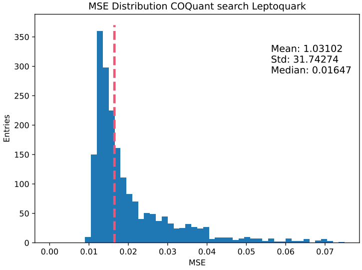
- **Knowledge Distillation** was tested as a way to **optimize** an AE for search of physics events not explained by the SM;
- In the comparison between the CoSearch and PhaseSearch quantization approaches, the former offers a set of students **more peaked** around the best one. This suggests that this procedure should yield better results faster then the latter.
- The co-training technique for knowledge distillation was also tested. The first results see an overall worse performance in the anomaly detection task w.r.t. post-training distillation;

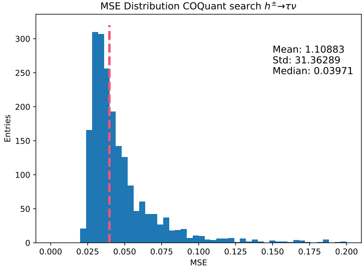
Thank you

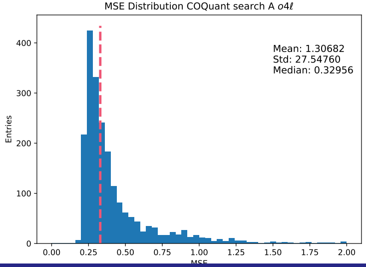
Best Postsearch student (Validation set)

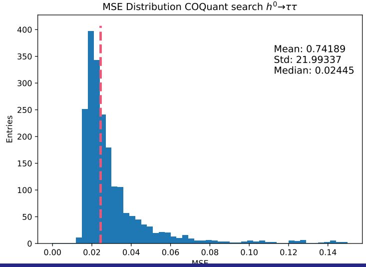


Losses' MSE distribution Cosearch

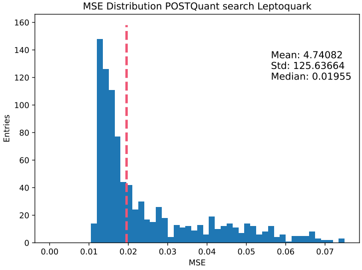


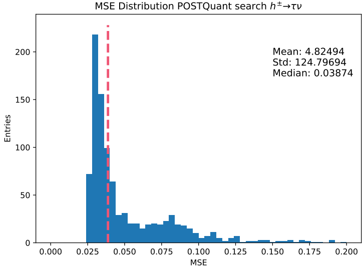


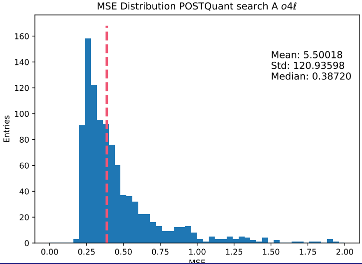


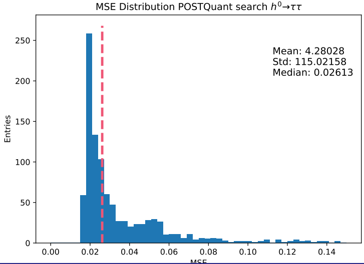


Losses' MSE distribution Post search

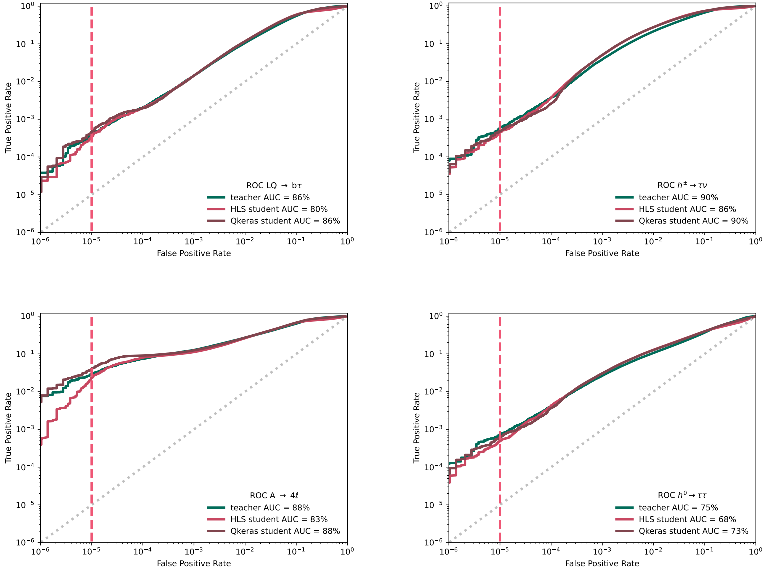








Best Co-quantization student ROCs



Hardware footprint after Synthesis

Name	BRAM_18K	DSP48E	FF	LUT	URAM
DSP	-	-	-	-	-
Expression	-	-	0	120	-
FIFO	-	-	-	-	-
Instance	-	850	24674	214625	-
Memory	-	-	-	-	-
Multiplexer	-	-	-	267	-
Register	-	-	1924	-	-
Total	0	850	26598	215012	0
Available	2688	5952	1743360	871680	640
Available SLR	1344	2976	871680	435840	320
Utilization (%)	0	14	1	24	0
Utilization SLR (%)	0	28	3	49	0

Latency (cycles)		Latency (absolute)	
min	max	min	max
100	100	0.500 us	0.500 us

- Target platform: Alveo U50 for testing purposes;
- Data only about the NN kernel;
- The NN sits comfortably in a single "slice" (SLR) of the board;
- With further optimization these figures could be reduced even more;
- The same can be said for the latency (500 ns).