MACHINE (DEEP) LEARNING IN SISTEMI DI TRIGGER

Stefano Giagu

INFN Sezione di Napoli - 21-22.4.2022

INFN

Istituto Nazionale di Fisica Nucleare

PROGRAMMA DEL CORSO

- ieri:
  - **mattina:** richiami elementi essenziali ANN, DNN in sistemi di trigger, tecniche di compressione di reti neurali, acceleratori FPGA per infrenza AI a media letenza, l'ambiente VitisAI
  - **pomeriggio:** (hands-on) training di un modello CNN per un trigger a media latenza su acceleratore FPGA: implementazione e training del modello non compresso, compressione tramite pruning (con tensorflow) e quantizzazione fixed-point INT8 (con tensorflow e con VitisAI) del modello, compilazione su acceleratore Xilinix Alveo U50 con VitisAI
- oggi:
  - **mattina:** tecniche di compressione di reti neurali (2nda parte), inferenza AI ultrarapida su FPGA, la libreria hls4ml e le sue limitazioni, use-case applicativo: trigger hw muonico per l'upgrade di fase2 dell'esperimento ATLAS
  - **pomeriggio:** (hands-on) training di un modello CNN per la ricostruzione a bassissima latenza di muoni nel trigger di livello-0 dell'esperimento ATLAS, quantizzazione estrema con Qkeras, distillazione del modello con teacher-student knowledge-distillation, sintesi del modello con hls4ml vs implementazione custom CNN in VHDL

TECHNICHE DI COMPRESSIONE

- **pruning:** vengono eliminati i pesi del modello che hanno minore effetto sulla risposta del modello oppure che risultano più piccoli in modulo
- **weight sharing (o clustering):** gruppi di pesi vicini vengono raggruppati in cluster e il loro valore viene sostituito con un valore unico
- **quantization:** può essere applicata ai pesi e/o alle uscite delle funzioni di attivazione dei layer della rete neurale. Consiste nel ridurre la precisione con cui si rappresentano le diverse quantità (16bit, 8bit, 4bit … invece che gli usuali 32bit o 64bit)
- **Binary / Ternary net:** versione estrema della quantizzazione (reti con pesi e ativazioni binarie [0,1] o ternarie [-1,0,1]

TECHNICHE DI COMPRESSIONE

- **sparse regularisation:** appartiene allo stesso gruppo delle tecniche di pruning (training aware pruning) e di dropout (post-training pruning), in cui la sparsità del modello è ottenuta applicando tecniche di regolarizzazione L1, L2 o L1+L2 durante il training dell rete stesse

- **distillazione via teacher-student knowledge transfer:** si usa un modello complesso pre-addestrato, come guida durante l'addestramento di un modello semplificato

- **low-rank factorisation:** riduce la dimensionalità delle matrici dei pesi applicate ad ogni layer della rete per velocizzare le operazioni di consolazione nei primi layer (che dominano i tempi di inferenza in una deep CNN)

- **3x3 winograd convolutions (NVIDIA):** versione più efficiente della convoluzione in CNN che riduce il numero di moltiplicazioni floating-point necessarie

WEIGHT SHARING O CLUSTERING

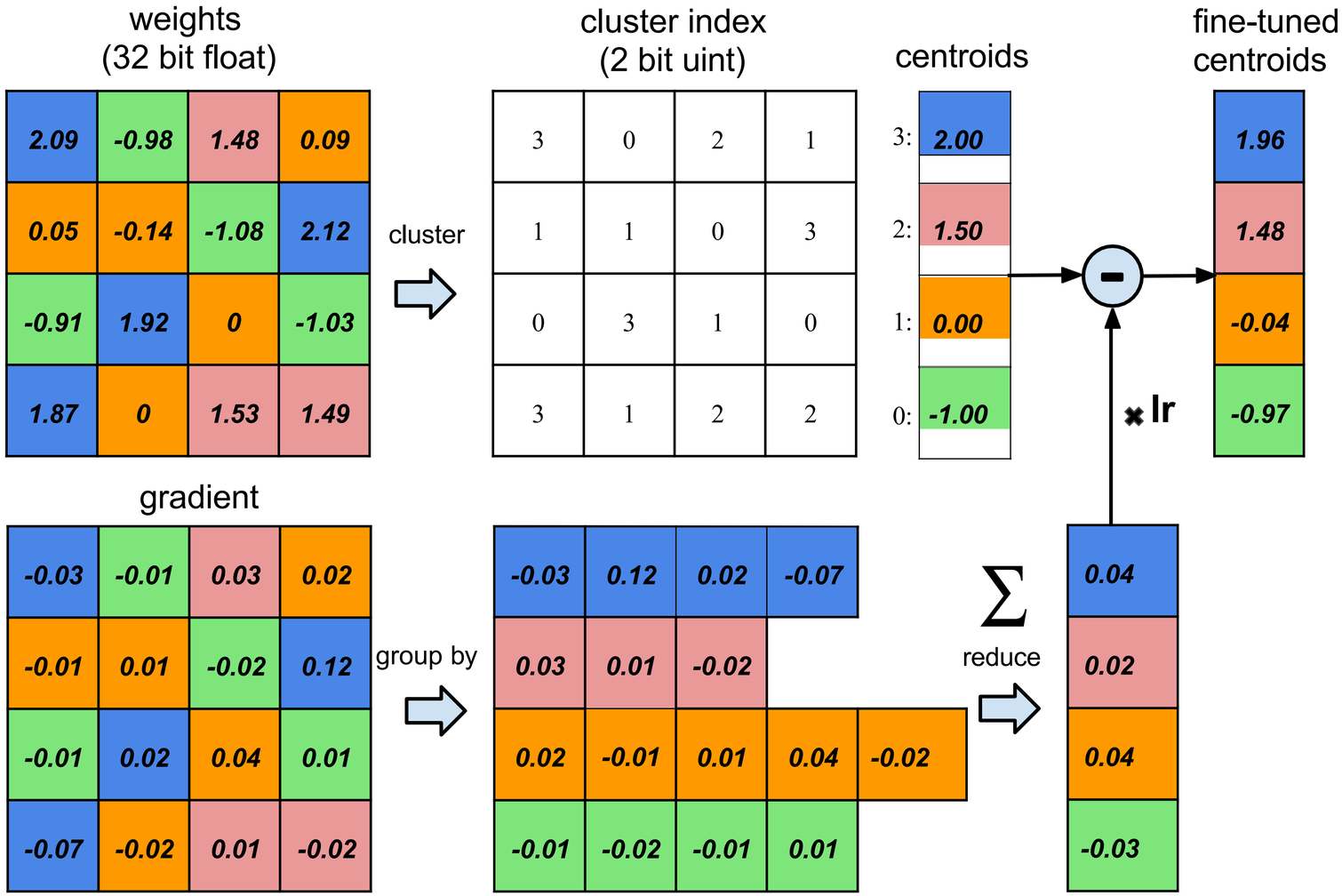
- l'algoritmo di clustering, o di condivisione dei pesi, riduce il numero pesi della rete che hanno valore univoco, sostituendoli a gruppi con un peso unico
  - raggruppa i pesi di ogni layer in N cluster usando algoritmi di clustering (tipicamente density based come dbscan)
  - condivide il valore del centroide del cluster per tutti i pesi appartenenti al cluster
  - spesso si accompagna ad una quantizzazione dei pesi
- ha vantaggi simili a quelli del pruning
- NOTA: non funziona bene per layer che sono preceduti da batch normalisation o layer normalisation

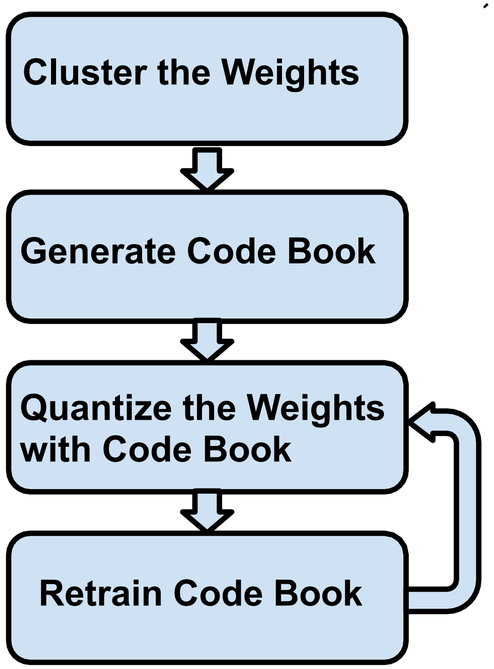
CLUSTERING CON QUANTIZZAZIONE

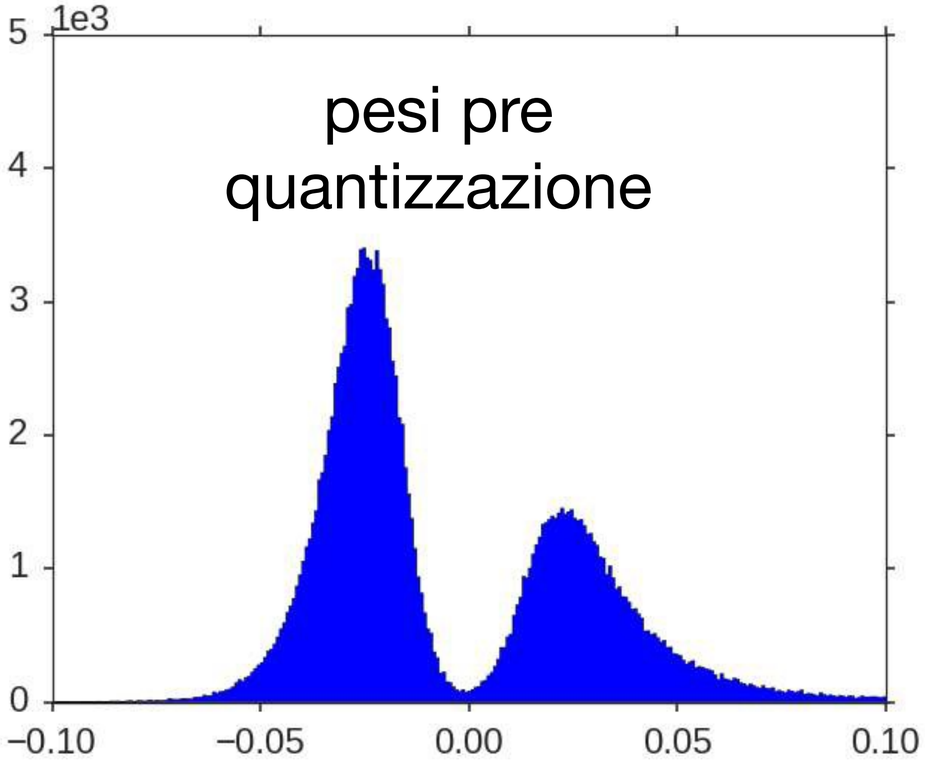
sostituisce i pesi con
i valori degli indici di cluster (2 bit)

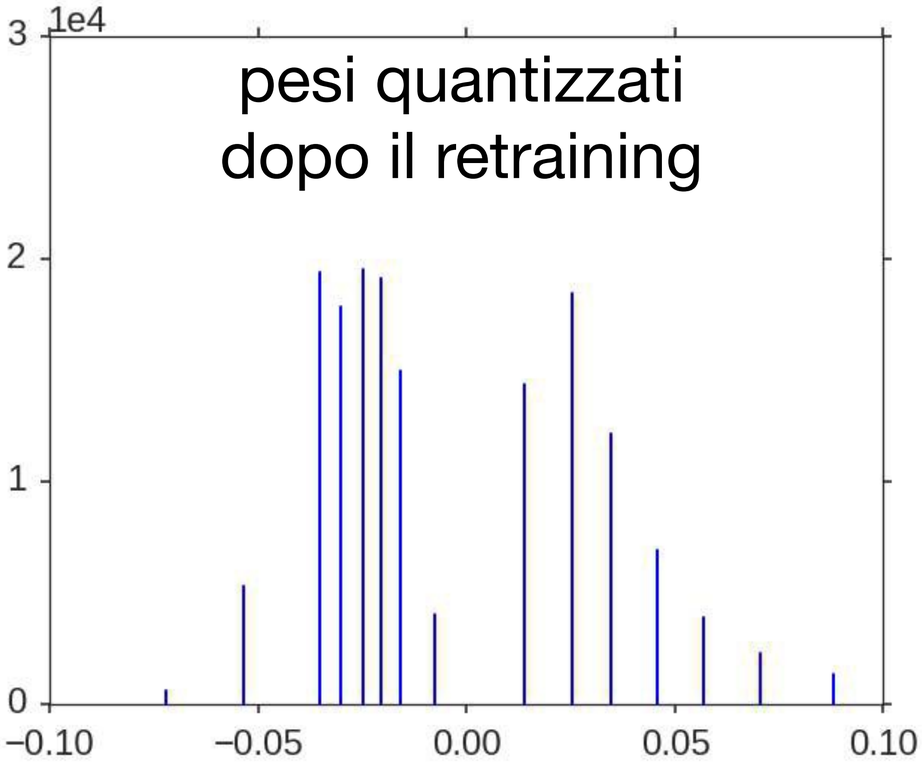
~~2.09, 2.12, 1.92, 1.87~~

2




Cluster the Weights
Generate Code Book
Quantize the Weights with Code Book
Retrain Code Book


pesi pre quantizzazione
1e3
5
4
3
2
1
0
−0.10
−0.05
0.00
0.05
0.10

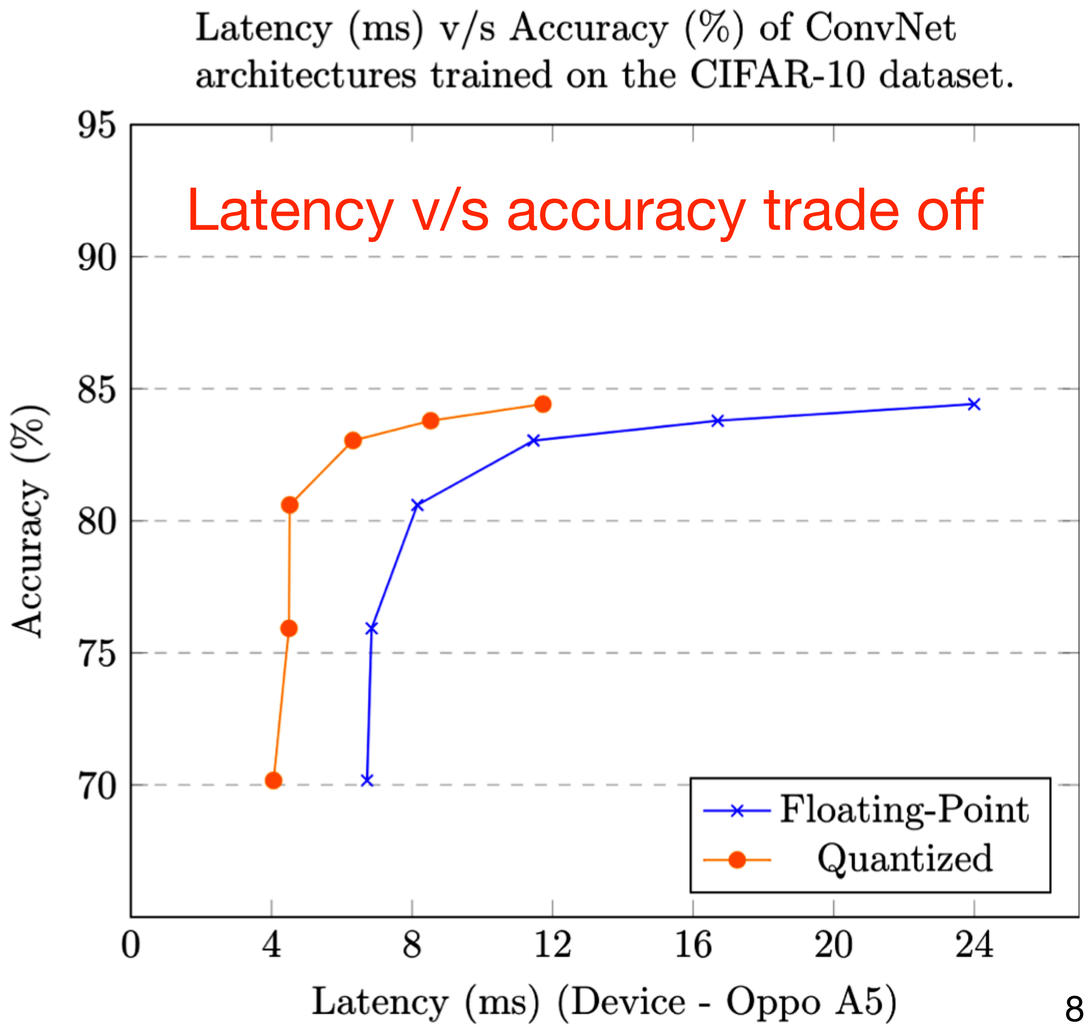

pesi quantizzati dopo il retraining
1e4
3
2
1
0
−0.10
−0.05
0.00
0.05
0.10

COMPRESSIONE VIA QUANTIZZAZIONE

- riduce la dimensione con cui i pesi e le attivazioni della rete vengono rappresentati
- tipicamente usate rappresentazioni fixed-point a 16, 8, 4 bit
- vantaggi:
  - dimensione del modello: riduzione di unfattore x32/n
  - latenza: l'operazione più costosa (a parte l'operazione di lettura/scrittura di memoria) nel calcolo di un layer di un NN è la moltiplicazione tra la matrice dei pesi e la matrice di feature

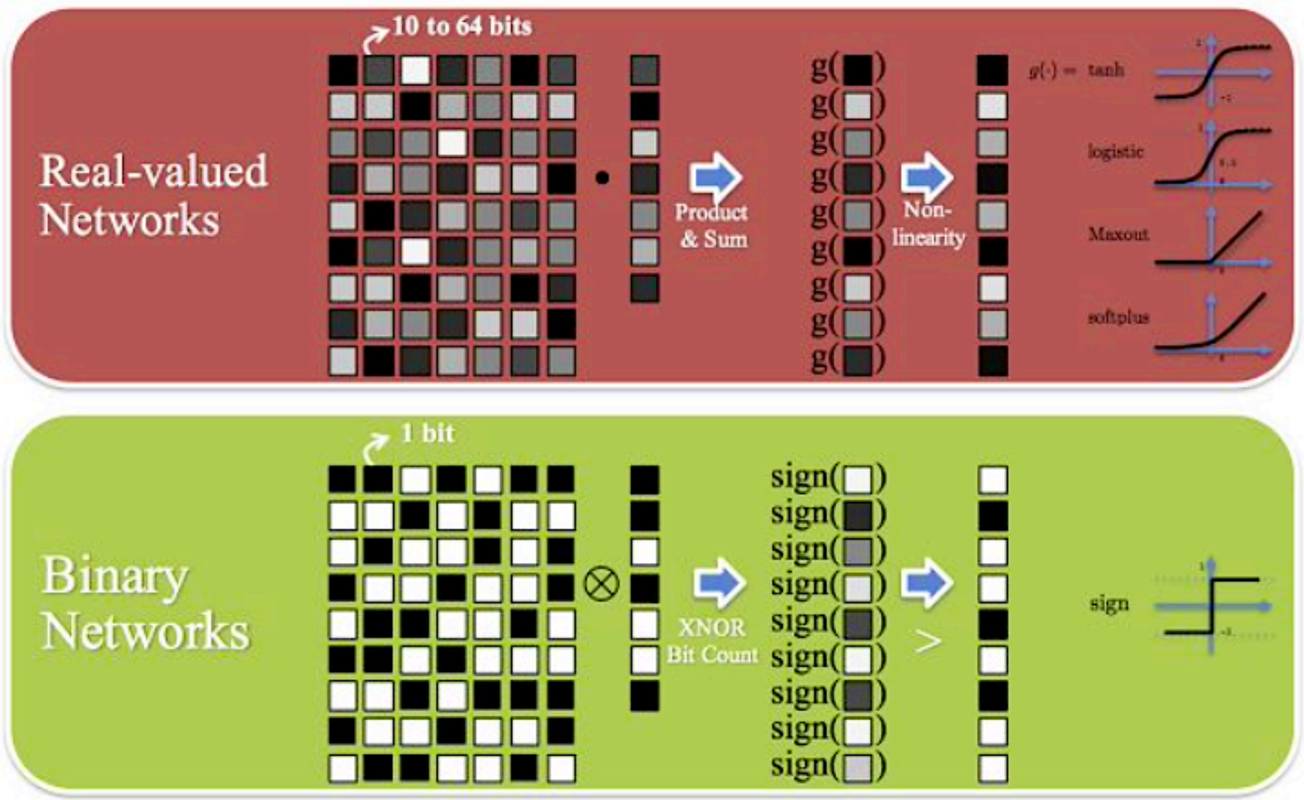
$$\phi(Wx + b)$$

il calcolo in fied_point è più veloce che in floating point



CASO ESTREMO: BINARY/TERNARY NETWORKS

- estremizzazione della quantizzazione:
  - rimpiazza pesi/attivazioni floating/fixed-point con aritmetica a 1 o 2 bit
    - binary net: 1 bit [0,1] (arxiv.:602.02830)
    - ternary net: 2 bit [-1,0,1] (arxiv:1605.04711)



- porta a una perdita delle prestazioni ma la compressione e il guadagno in latenza possono essere notevoli
- moltiplicazioni diventano operazioni di bit-flip
  - binary:

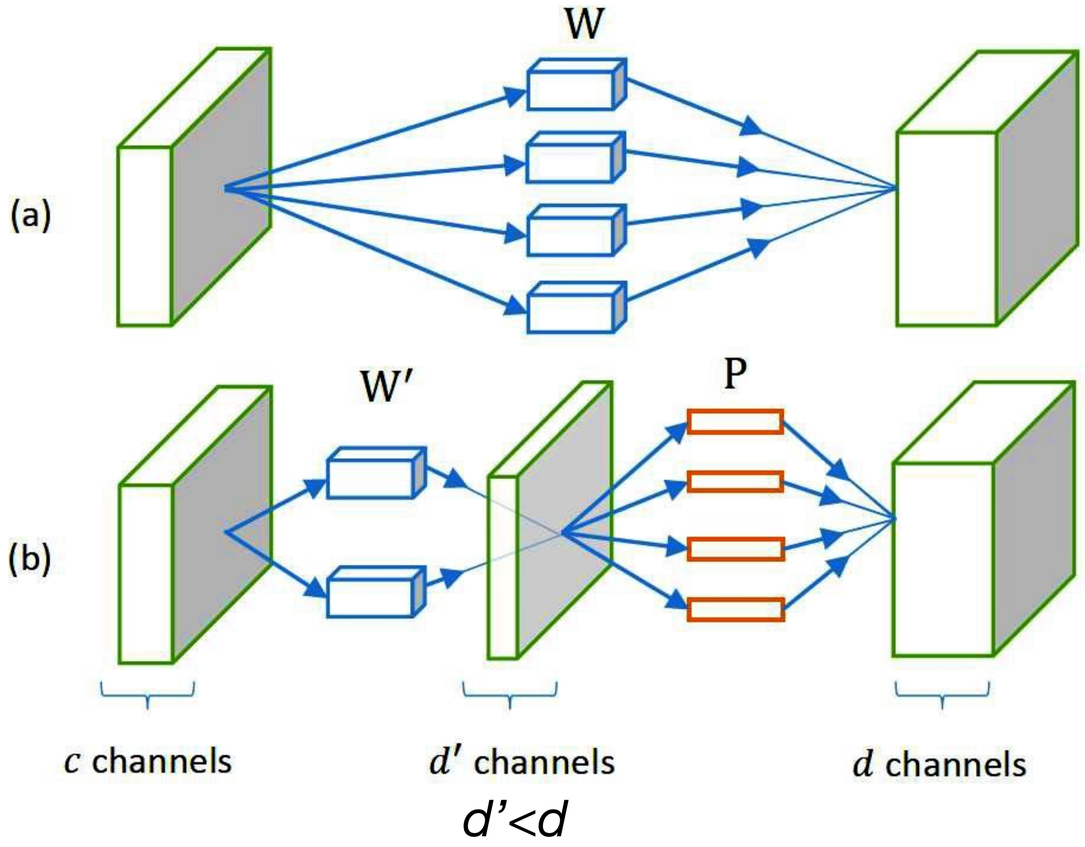
    res = w == 0 ? -d : d;
  - ternary:

    res = w == 0 ? 0 : w == -1 ? -d : d;

LOW RANK FACTORIZATION

- riduce la dimensionalità delle matrici dei pesi applicate ad ogni layer convoluzionale della rete
- sfrutta la ridondanza (overparametrizzazione) presente nei filtri convoluzionali per derivare approssimazioni molto più veloci da calcolare (x2 speedup)

- Decompose a convolutional layer with d filters with filter size $k \times k \times c$ to
  - A layer with d' filters ($k \times k \times c$)
  - A layer with d filter ($1 \times 1 \times d'$)



d'<d

Complessità

$$O(dk^2c)$$

$$O(d'k^2c) + O(d'd)$$

rapporto $\sim \frac{d'}{d}$

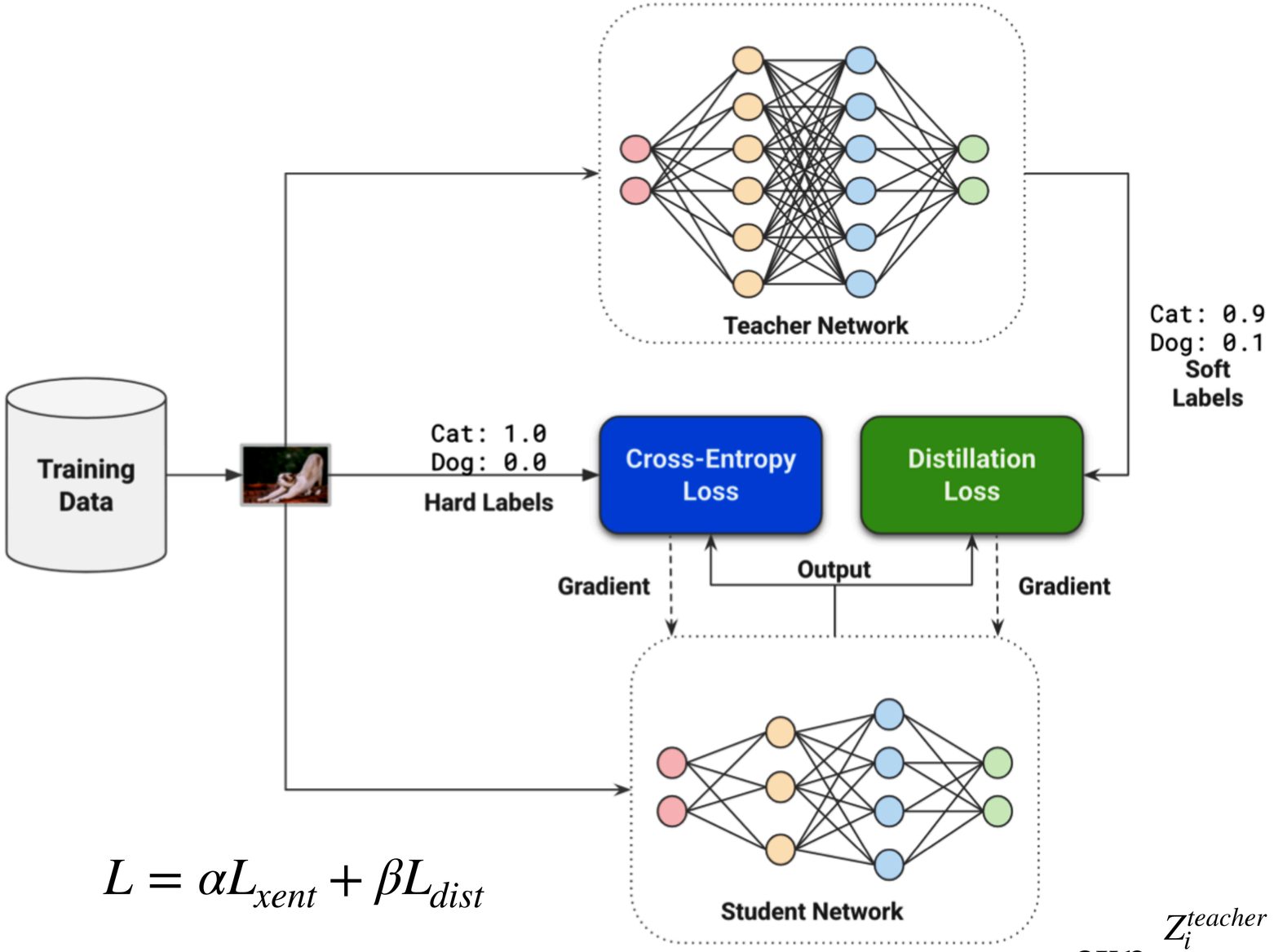
Zhang et al, arXiv:1411.4229
Denton et al, arXiv:1404.0736

KNOWLEDGE DISTILLATION

- Hinton e collaboratori, in un lavoro seminale del 2014 (arxiv:1503.02531) hanno esplorato la possibilità di insegnare a reti neurali lightweight (student) ad estrarre “dark knowledge" da un singolo modello insegnate complesso o anche da un ensemble di insegnanti. L’idea di Hinton (proposta per task di classificazione) era quella di usare il teacher per generare label software (soft label) per i dati di training (simile in qualche modo alla data augmentation in cui si creano dati sintetici nella data augmentation per migliorare la capacità di generalizzazione di una rete neurale).

- le soft label sostituiscono i valori delle label ground-truth con le probabilità che l’input appartenga a ciascuna classe, rpobabilità stimata dal teacher.

- durante il training il modello student può quindi apprendere sia dalle label hard (ground truth) che da quelle soft prodotte dal teacher



- sia il teacher che lo student ricevono lo stesso input
- lo studente viene addestrato utilizzando la normale x-entropy loss incrociata con le etichette hard, ed utilizzando la loss di distillazione che utilizza le soft label
- durante il train i pesi del teacher sono congelati e quindi non vengono aggiornati durante la backprop

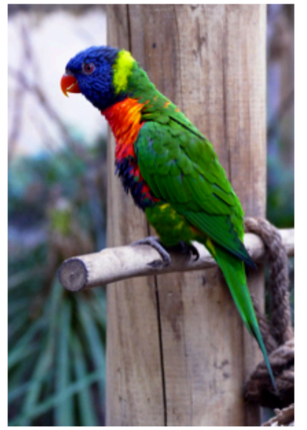
$$L = \alpha L_{xent} + \beta L_{dist}$$

$$L_{dist} = \text{x-entropy}(\hat{y}, y^{soft}; \mathbf{w})$$

$$y_i^{soft} = \frac{\exp \frac{Z_i^{teacher}}{T}}{\sum_j \exp \frac{Z_j^{teacher}}{T}}$$

T: parametro di temperatura che agisce da smmooting per la distribuzione delle soft label

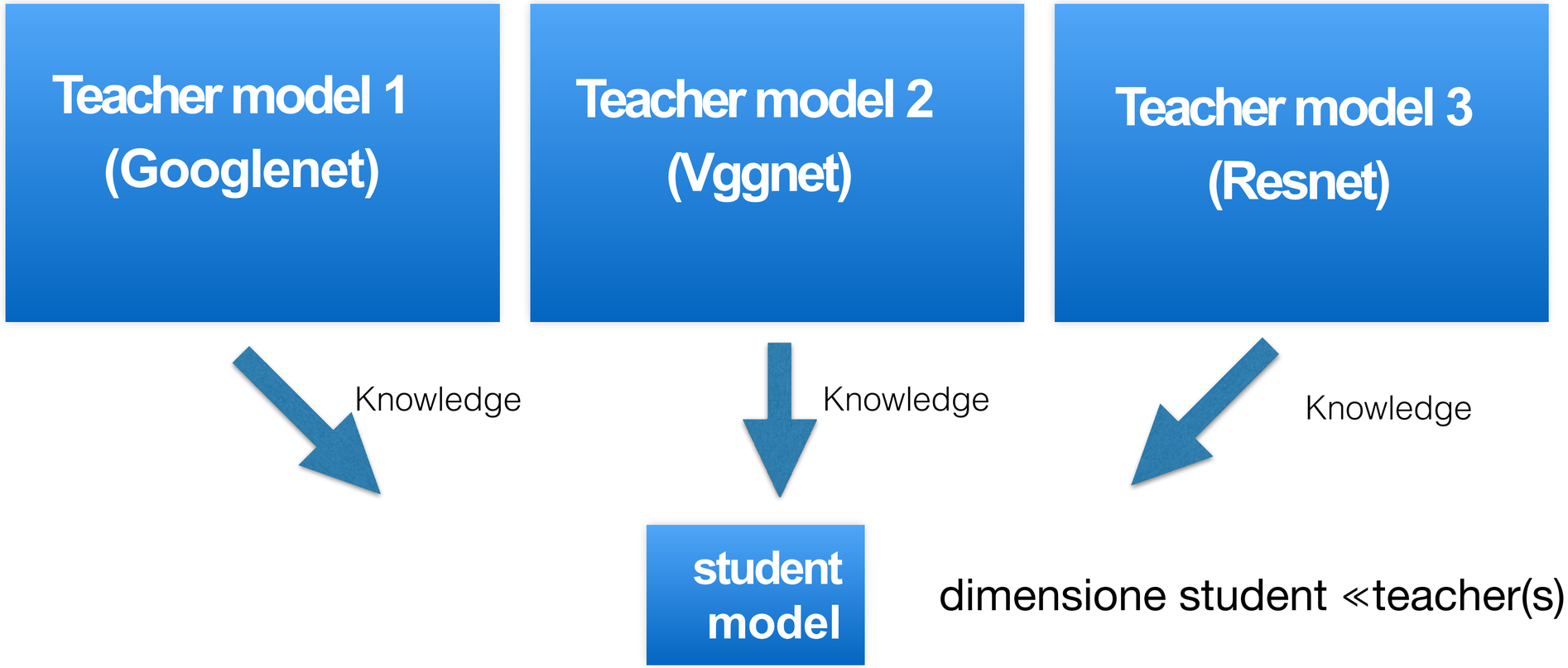
esempio

Labels	Images			
				
Hard[19]	[1, 0, 0, 0]	[0, 1, 0, 0]	[0, 0, 1, 0]	[0, 0, 0, 1]
Soft	[.80, .15, .03, .02]	[.15, .75, .05, .05]	[.03, .02, .85, .10]	[.05, .05, .20, .79]

- il teacher pensa che un gatto sia più simile ad un cane che ad un piccione o a un pappagallo, e che questi ultimi siano più simil tra loro che al gatto o al cane
- la distillazione permette allo student di catturare relazioni tra le classi che non sono rappresentate nelle hard label del dataset di training

KNOWLEDGE DISTILLATION PER LA COMPRESSIONE DI DNN

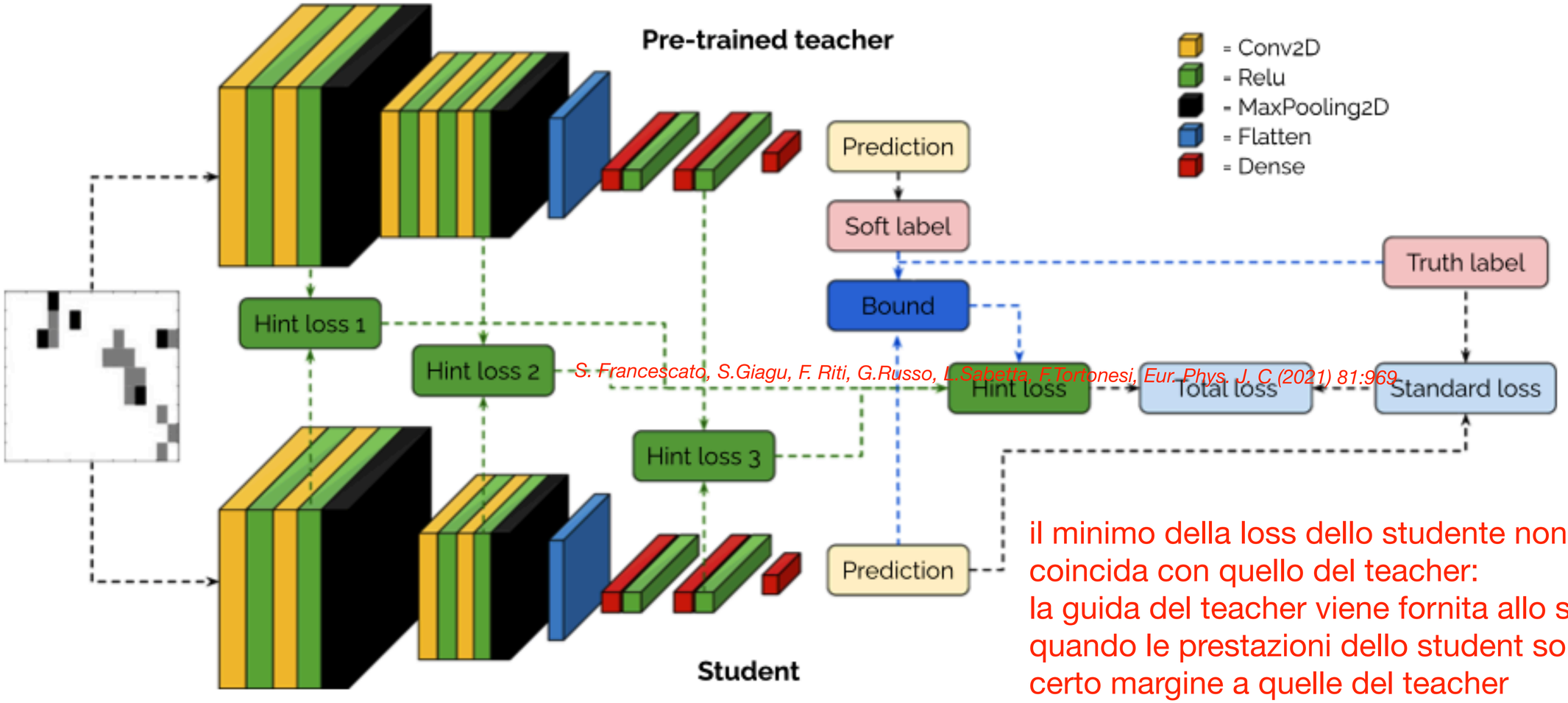
idea: trasferire la conoscenza appresa da uno o più DNN di grande dimensione e potere espressivo, pre-addestrato per la stessa task, ad un modello lightweight eventualmente anche compresso con i metodi visti in precedenza



dimensione student «teacher(s)

Hinton et al, arxiv:1503.02531
Romero et al, arxiv:1412.6550

KNOWLEDGE DISTILLATION PER UNA TASK DI REGRESSIONE



il minimo della loss dello studente non è detto che coincida con quello del teacher:
la guida del teacher viene fornita allo studente solo fino a quando le prestazioni dello student sono inferiori di un certo margine a quelle del teacher

permette di addestrare modelli molto leggeri che sarebbe altrimenti impossibile far convergere in assenza del teacher

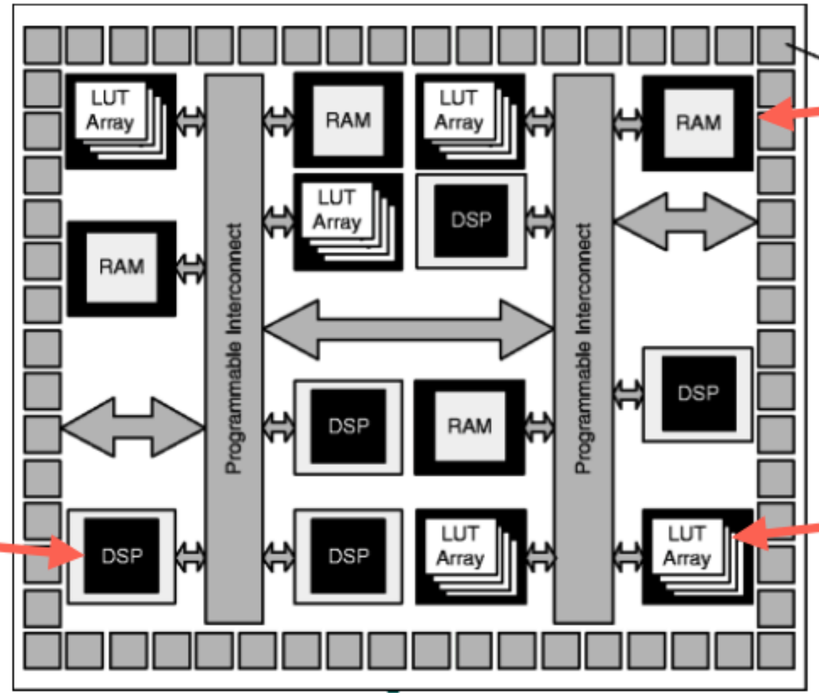
Romero et al, arxiv:1412.6550
S. Francescato, S.Giagu, F. Riti, G.Russo, L.Sabetta, F.Tortonesi, Eur. Phys. J. C (2021) 81:969

DNN SU FPGAs

- Field Programmable Gate Arrays sono circuiti integrati riprogrammabili
- prestazioni simili a quelle degli ASIC ma molto più versatili
- contengono differenti building blocks (“resources”) (DSPs, LUT, FilipFlops, RAM) che possono essere interconnesse tra loro a nostro piacimento

Available resources:

- **Digital signal processors (DSPs):** specialised units for multiplication and arithmetic
  Used a lot in DNNs and easily become scarce for big networks!



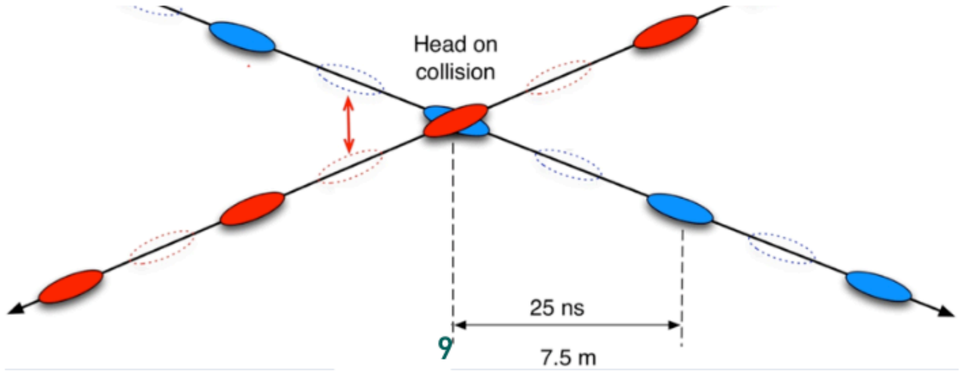
- **memory (BRAM)**
- **Logic cells/lookup tables (LUTs):** perform arbitrary functions (boolean ops, arithmetic, small memory)
- **flip-flops (FF):** registers data in time with clock pulse to keep OPs synchronised

DNN IMPLEMENTATE “ALL ON FPGA”

- vantaggi:
  - permettono un alto parallelismo e quindi consentono di implementare DNN con bassissima latenza (le componenti sequenziali della rete possono in ogni caso essere accelerate sfruttando il parallelismo tramite pipeline)
  - sono intrinsecamente deterministiche: la latenza nel processamento dei dati è ripetibile e predicibile
  - sono più efficienti delle GPU in termini di potenza consumata (migliore rapporto prestazioni/watt)

- prospettiva molto attraente per provare ad implementare un’algoritmo basato su DNN in uno dei trigger hw degli esperimenti LHC

es. ATLAS@HL-LHC
L0 trigger total decision time O(1μs)
latency per l’algoritmo di filtro: <400ns



DNN E FPGA

- l’inferenza con un ANN consiste sostanzialmente in operazioni di moltiplicazioni tra matrici che sono impresentabili in modo molto efficiente n ei DSP delle FPGA

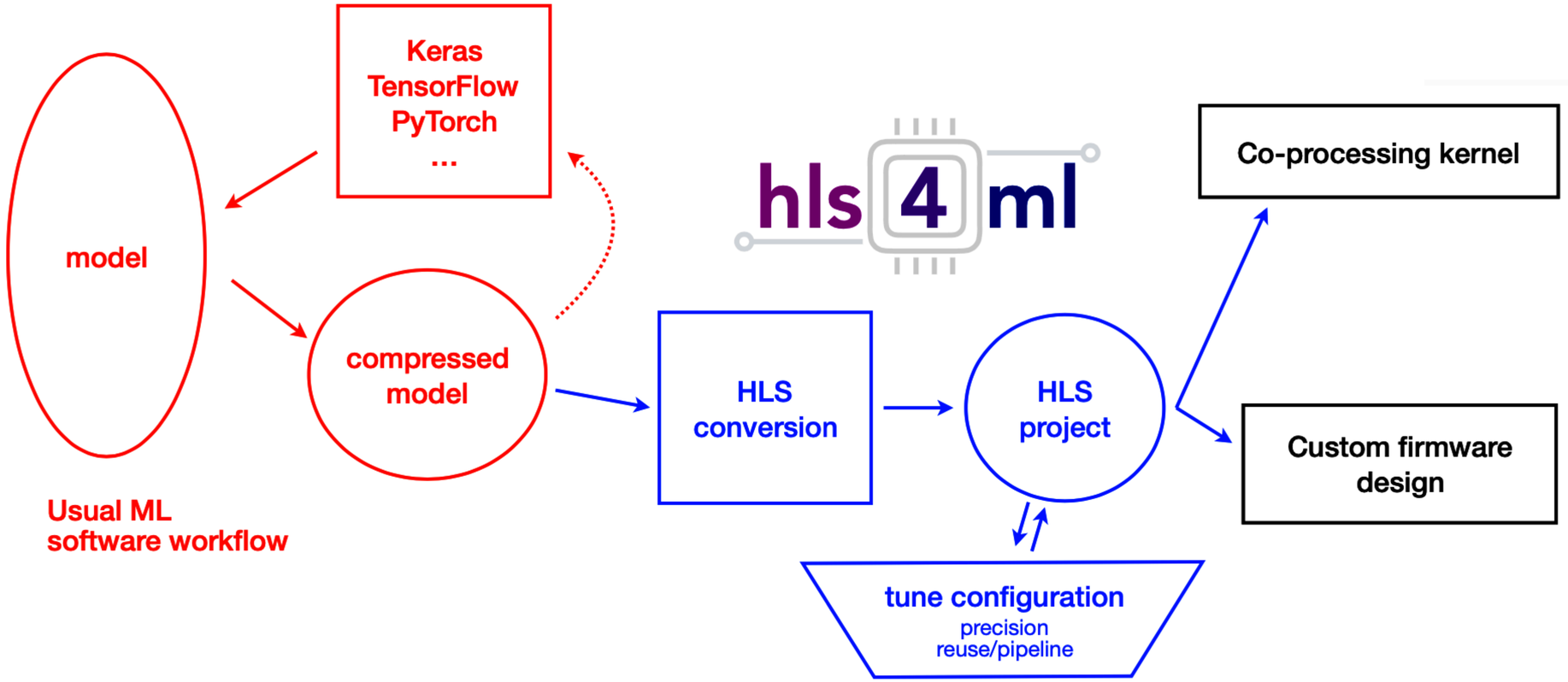
- i DSP diventano la risorsa più preziosa e il vincolo principale quando si vuole sintetizzare i modello neurale nella FPGA

- le ultime generazioni di FPGA hanno 2k-12k DSP (ex. Xilinx Virtex Ultrascale+ XCV2P-12P)

- questo fissa la dimensione massima del modello imlementabile nella FPGA (tenendo conto che una parte delle risorse deve essere necessariamente usata per le altre funzioni richieste dallo specifico trigger)

HLS4ML

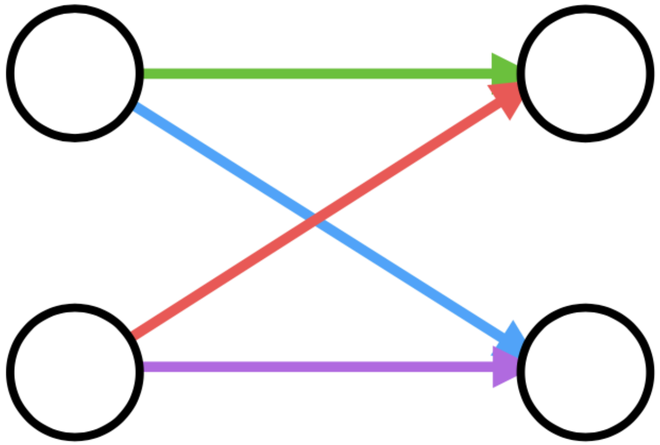
- https://github.com/fastmachinelearning/hls4ml
- libreria per la traslazione automatica di modelli DL in firmware FPGA Xilinx

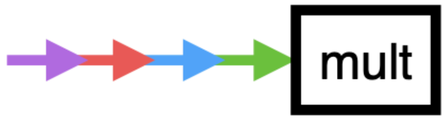


permette di controlare durante la sintesi il tradeoff tra utilizzo risorse e latenza/throughuput tramite un "Reuse Factor" che controlla quanto parallelizzare

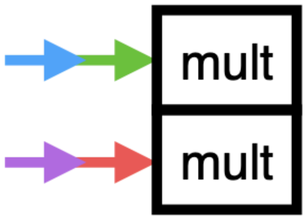
PARALLELIZZAZIONE VS SERIALIZZAZIONE

- **ReuseFactor**: how much to parallelize

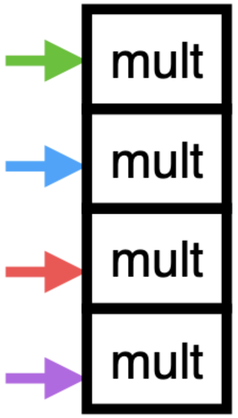




reuse = 4
use 1 multiplier 4 times



reuse = 2
use 2 multipliers 2 times each



reuse = 1
use 4 multipliers 1 time each

LIMITAZIONI

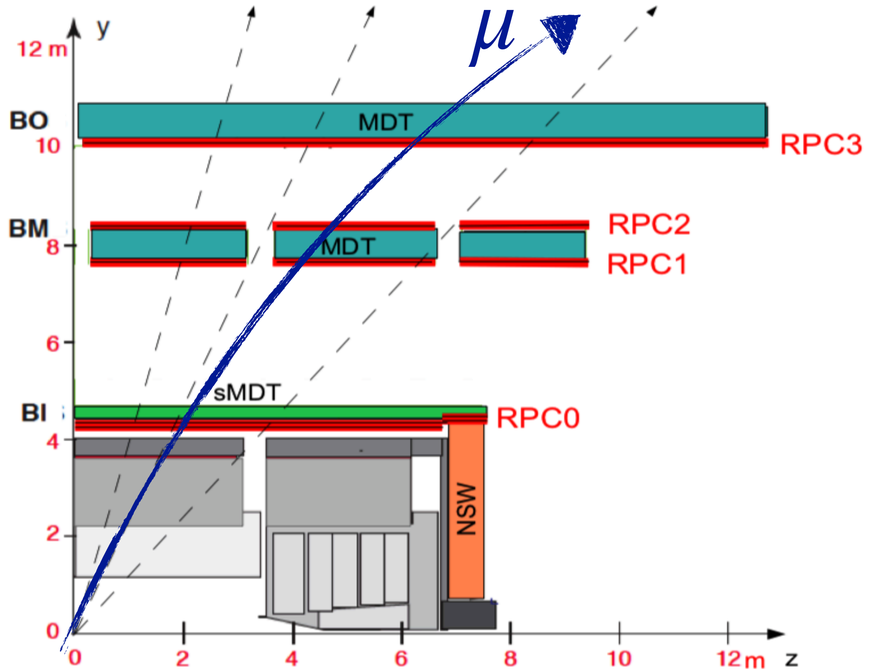
- CNN: input solo seriale (latenza cresce linearmente con la dimensione dell’input)
- non tutte le tipologie di layer, attivazioni etc. supportate
- con input grandi, modelli non semplici spesso fallisce la sintesi
- … vedi Luigi …

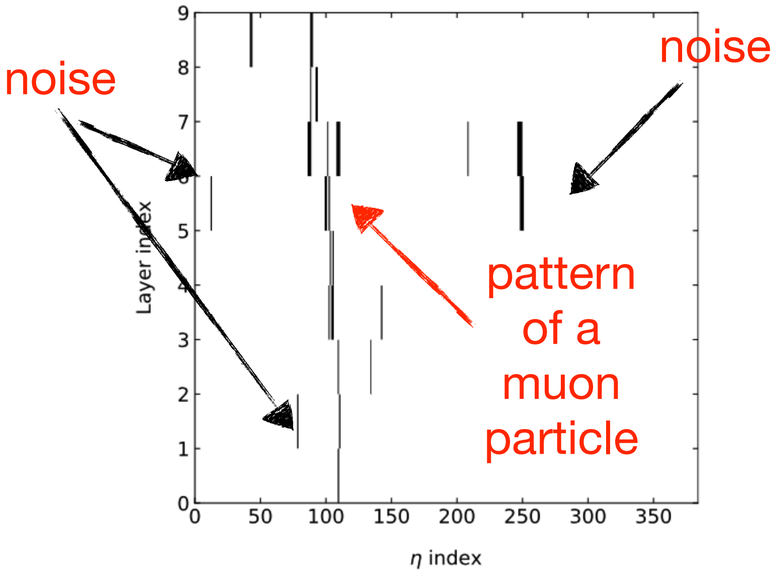
USE CASE: ULTRAFAST CNN SU FPGA PER IL TRIGGER DI L0 MUONICO DI ATLAS@HL-LHC

- Goal: ricostruire pt e pseudorapidità di tracce muoniche a partire degli hit nei rivelatori RPC **in meno di 370 ns**

trade-off tra latenza e occupazione delle risorse in una FPGA, mentre le prestazioni di un DNN dipendono fortemente dalle sue dimensioni

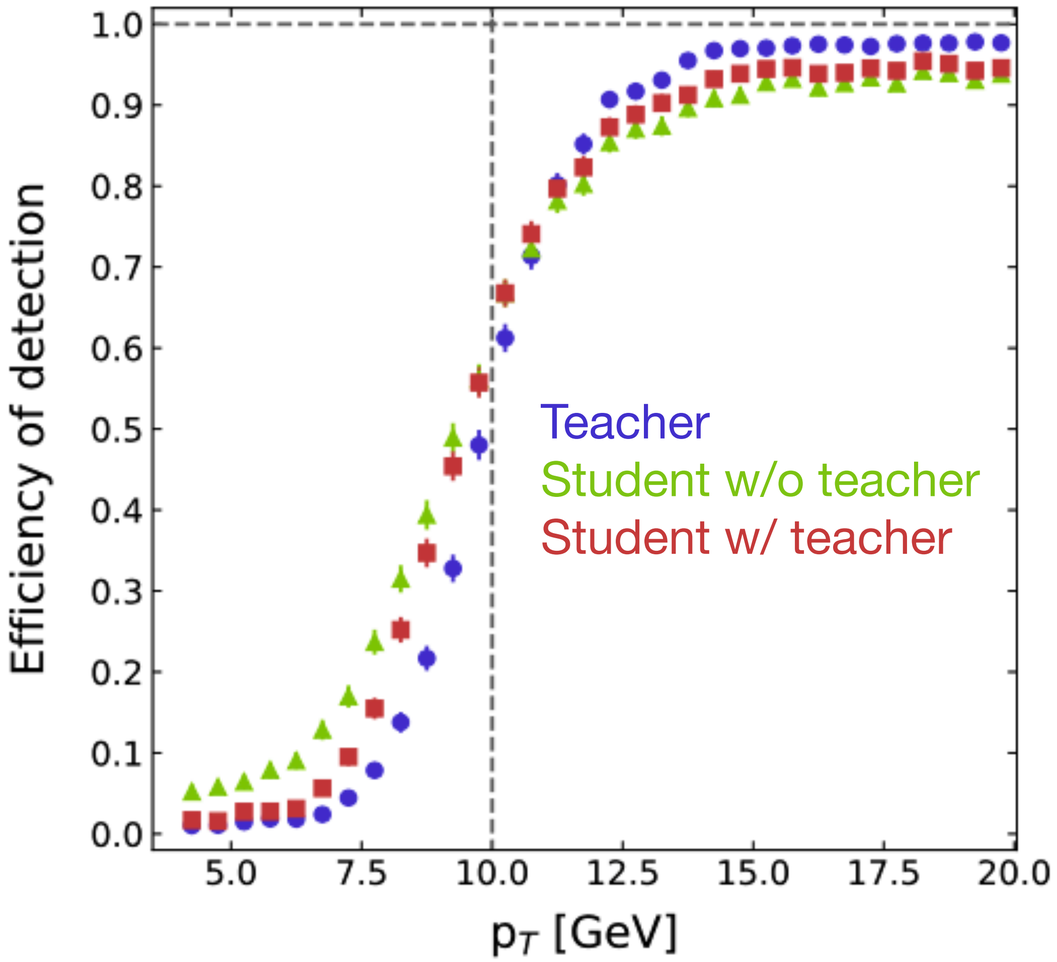
Strategia: multi-stage **AI model compression** e semplificazione basata su **quantizzazione aggreesiva** + **knowledge distillation** per minimizzare la perdita di prestazioni





PRESTAZIONI PRELIMINARI

curva efficienza single muon trigger
soglia nominale pt 10 GeV



FPGA resource occupation

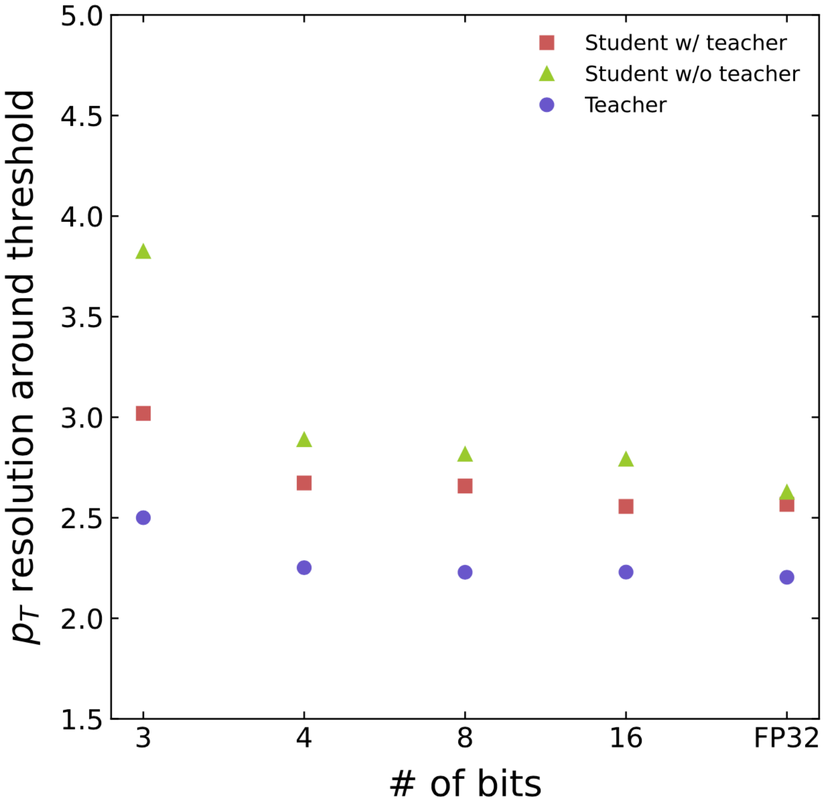
Table 3 Percentage occupancy relative to the total FPGA available resources (model xcvu13p-fhga2104-2L-e [14])

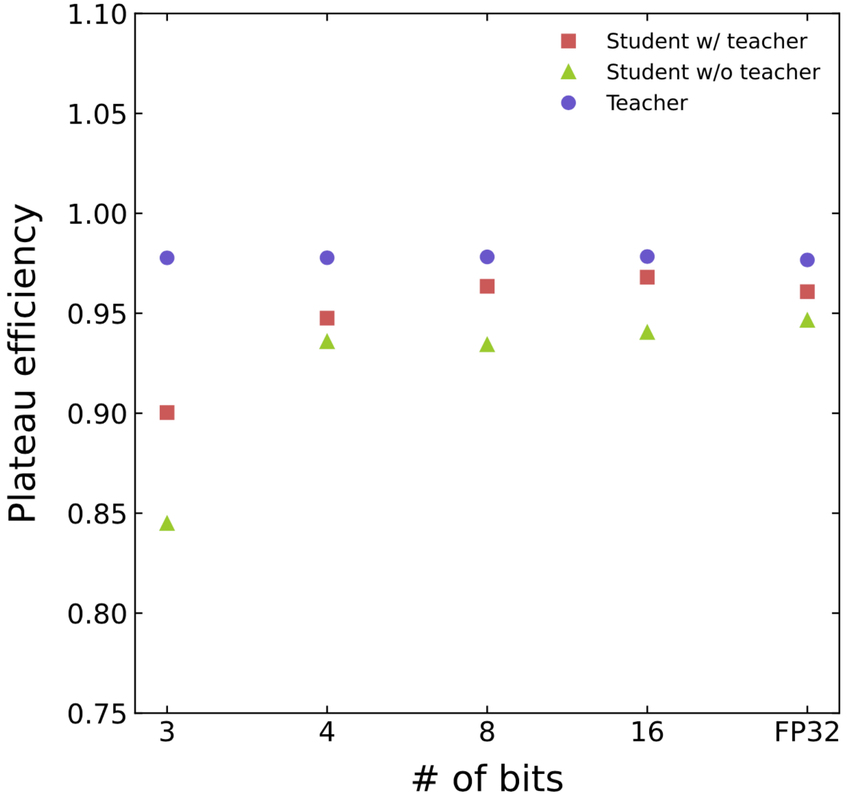
Model (9 × 16)	BRAM	DSPs	FF	LUT
Teacher (%)	20.9	258.0	69.4	15.3
Student 32 bit (%)	3.2	31.0	8.4	2.7
QStudent 4 bit (%)	0.2	0.05	0.4	1.7

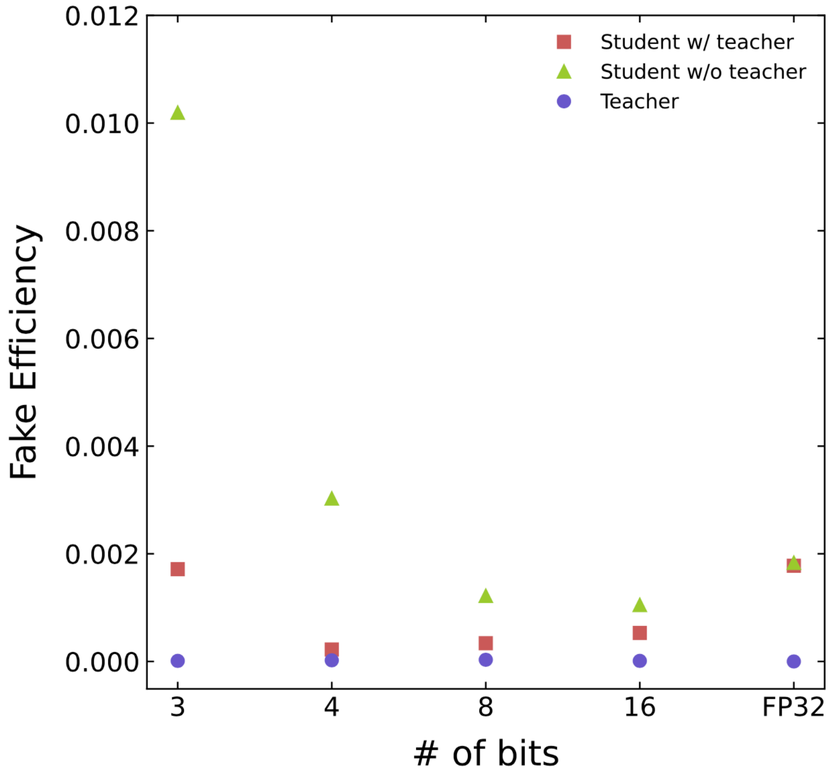
Inference time per event on FPGA
Xilinx Ultrascale+ XCV13P

- Teacher fp32: 5 ms (Tesla V100 GPU)
- Student 4 bit: 438 ns (hls4ml)
- Student 4 bit: 84 ns (our VHDL implementation)

S. Francescato, S.Giagu, F. Riti, G.Russo, L.Sabetta, F.Tortonesi, Eur. Phys. J. C (2021) 81:969


Student w/ teacher
Student w/o teacher
Teacher
p_T resolution around threshold
# of bits
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
3
4
8
16
FP32


Student w/ teacher
Student w/o teacher
Teacher
Plateau efficiency
# of bits
1.10
1.05
1.00
0.95
0.90
0.85
0.80
0.75
3
4
8
16
FP32


Student w/ teacher
Student w/o teacher
Teacher
Fake Efficiency
# of bits
0.012
0.010
0.008
0.006
0.004
0.002
0.000
3
4
8
16
FP32