A DNN for CMS track classification and selection

Leonardo Giannini* on behalf of the CMS collaboration
*UCSD

Introduction

Iterative tracking at CMS

4 main steps repeated in several iterations to ensure the best possible efficiency while keeping low fake rates (see ref. [1])

- **Track seeding**: Different types of seeds by iteration: starting from easier seeds and moving to more complicated seeds
- **Pattern recognition**: Compatible hits associated to the seed, based on Kalman filter techniques
- **Fitting**: Track hits fitted outside-in and inside-out, again using the Kalman filter
- **Track selection**: At the end of each iteration the hits of high purity tracks are removed from the following iterations

The importance of the track selection

- The track selection is an integral part of the iterative tracking, as hits coming from high purity tracks are removed for the subsequent iterations, thus reducing the combinatorics of the pattern recognition
- After all the iterations are completed, the tracks are merged into a single collection. Several types of track selection can be applied to the final collection ("high purity", loose…)

The Track selection Deep Neural Network

The track selection DNN is introduced for Run 3 tracking [2]. Previously a BDT was employed in Run 2 and a parametric selection in Run 1 [1].

DNN inputs/target

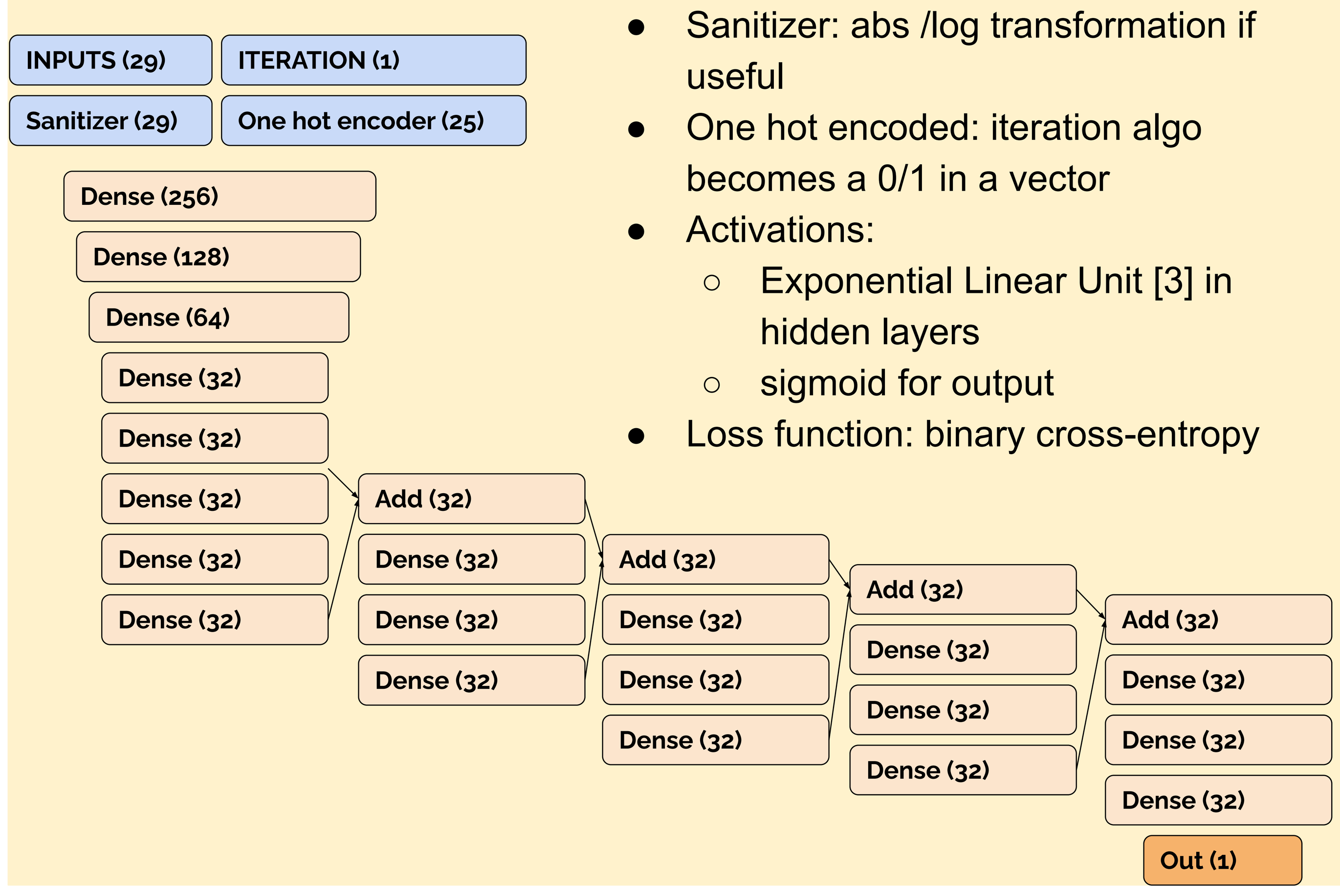
Input Features

- the track p_T, η, φ, and their respective uncertainties δp_T, $\delta\eta$, $\delta\varphi$
- p_x, p_y, p_z, p_T for the innermost and outermost state of the track
- the transverse and longitudinal impact parameters, d_0, d_z, computed both from the beamspot and from the closest primary vertex, and their respective uncertainties δd_0, δd_z
- the track χ^2 and number of degrees of freedom
- number of Pixel hits, number of Strip hits
- number of missing hits inside the innermost hit and outside the outermost hit
- number of inactive layers crossed inside the innermost hit and outside the outermost hit
- number of layers without hits overall
- the iteration flag (integer)

Target

- true/false flag: a true track must have more than 75% of its hits matched to a simulated track.

DNN architecture



- Simple feed-forward network
- Sanitizer: abs /log transformation if useful
- One hot encoded: iteration algo becomes a 0/1 in a vector
- Activations:
  - Exponential Linear Unit [3] in hidden layers
  - sigmoid for output
- Loss function: binary cross-entropy

Training Procedure

- Training performed on tracks, including those from pileup vertices, from several simulated samples generated at a center-of-mass energy of 14 TeV with pileup 20 to 70 (QCD, $t\bar{t}$, Z to electrons, long lived stop-antistop)
- Training in one step: no track selection on previous iterations, but all tracks labeled as "high purity" with consequent hit masking
- Batch size 512, Adam optimizer [4]
- 5 training epochs over 1.3B tracks
- Software package: keras + tensorflow [5,6]

Choice of working points

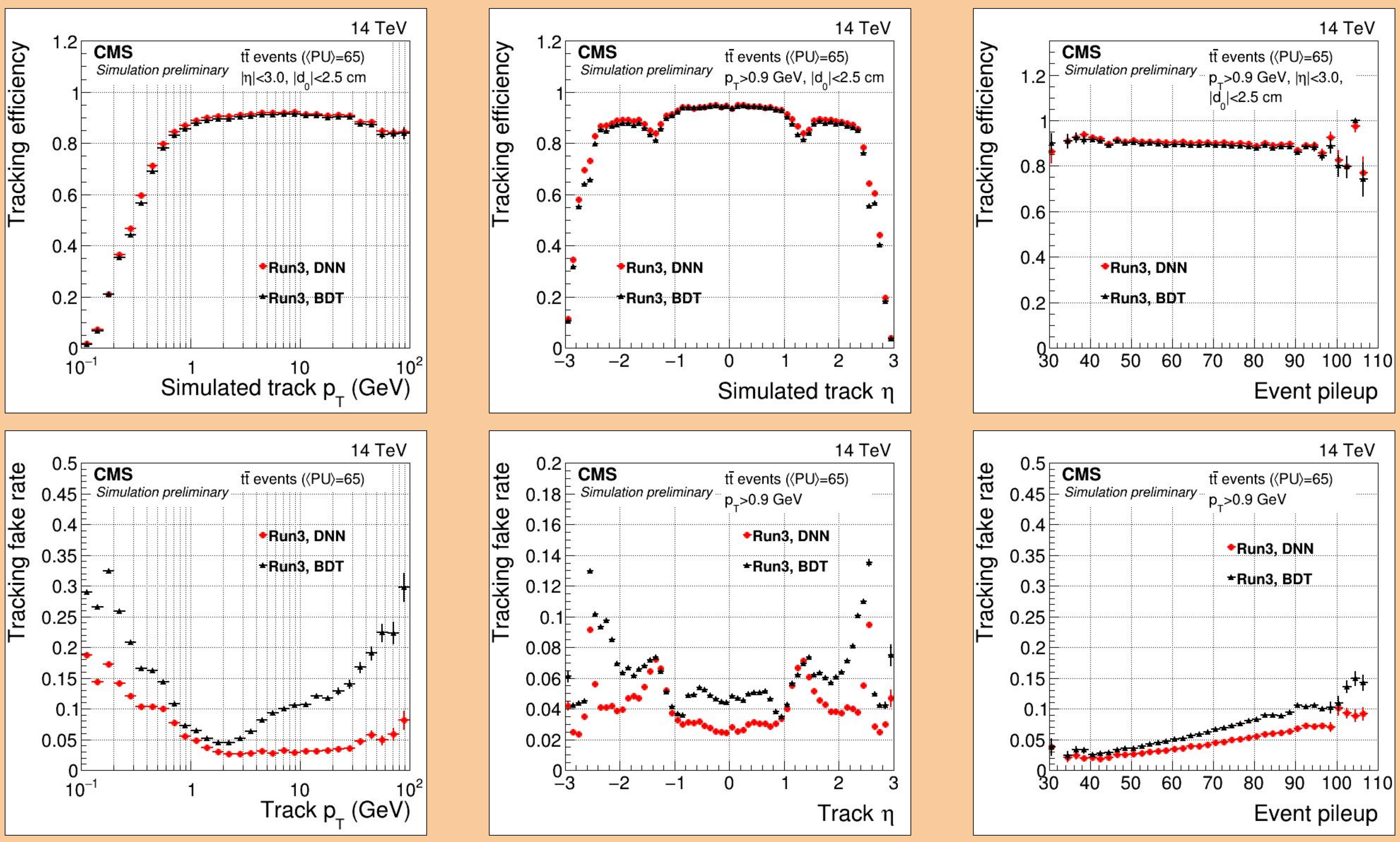
- The working point are chosen iteration by iteration in a validation sample similar to the training one - the efficiency is set to match the Run 2 BDT efficiency
- The choice of the working point is validated in tracking with the hit masking applied

Physics Performance

The mkFit algorithm [7,8] was introduced for a subset of the tracking iterations, replacing the legacy CKF algorithm, at the same time as the DNN

1. The DNN is trained for both mkFit and the legacy CKF and used based on the iteration
2. We compare the performance of the DNN on the current default tracking, including mkFit, to the Run 2 BDT applied on the same tracks

Efficiency and Fake rate in top pair production vs p_T, η, PU

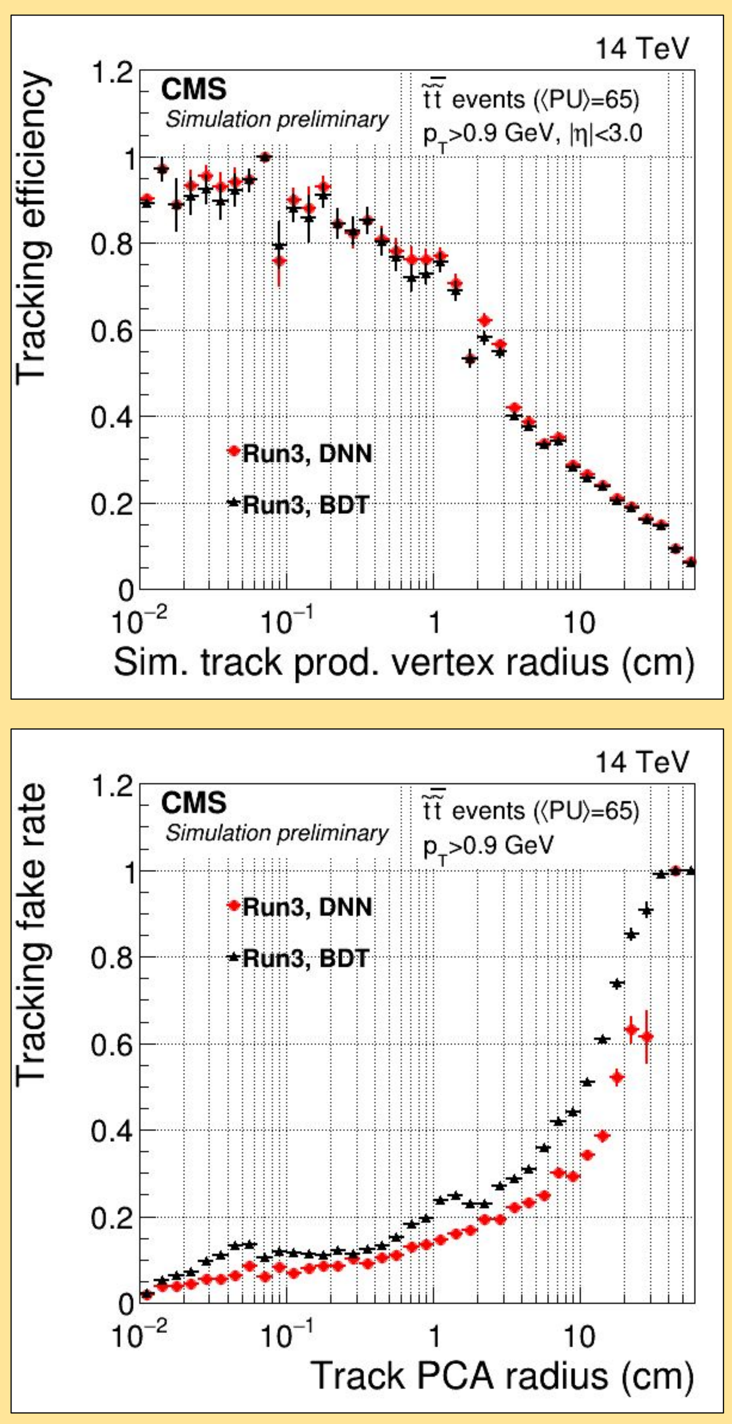


The tracking efficiency is consistent or slightly better, when comparing the DNN to the Run 2 BDT

The tracking fake rate is overall lower. Most notably in the high and low p_T range, in the barrel and encap (|η|<1 or |η|>2) and at higher PU values

The duplicate rate is about 20% higher (see ref. [2]), due to the merging of tracks selected by different DNNs trained on mkFit or legacy CKF reconstruction

Efficiency and Fake rate in long lived stop-antistop production vs displacement



Timing

The evaluation time of the DNN is compared to the one of the Run 2 BDT. A slight speed up is observed with batch size 1, and a larger one with larger batches. A batch size of 16 is chosen for memory footprint constraints.

Method	% of the total tracking time
BDT	4.9
DNN - batch size 1	3.4
DNN - batch size 16	0.9

Summary

CMS improved the track selection by means of a simple feed-forward DNN

The DNN leverages a larger training dataset and improves the efficiency and fake rate with respect to the previous BDT selection

The evaluation time is also faster, thus slightly reducing the total tracking time.

Bibliography

1. Description and performance of track and primary-vertex reconstruction with the CMS tracker, CMS Collaboration, e-Print: 1405.6569 [physics.ins-det], DOI: 10.1088/1748-0221/9/10/P10009, Published in: JINST 9 (2014) 10, P10009
2. Performance of the track selection DNN in Run 3, CMS-DP-23/009, CMS Collaboration
3. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUS), Djork-Arne Clevert, Thomas Unterthiner & Sepp Hochreiter, e-Print: 1511.07289 [cs.LG]
4. Adam: A Method for Stochastic Optimization, Diederik P. Kingma, Jimmy Lei Ba, e-Print: 1412.6980 [cs.LG]
5. https://keras.io. F. Chollet et al., Keras, Software available from https://github.com/keras-team/keras
6. TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems M. Abadi et al., Software available from tensorflow.org, e-Print: 1603.04467 [cs.DC]
7. Speeding up particle track reconstruction using a parallel Kalman filter algorithm, Steven Lantz (Cornell U.), Kevin McDermott (Cornell U.), Michael Reid (Cornell U.), Daniel Riley (Cornell U.), Peter Wittich (Cornell U.) et al., e-Print: 2006.00071 [physics.ins-det], DOI: 10.1088/1748-0221/15/09/P09030, Published in: JINST 15 (2020) 09, P09030
8. Performance of Run 3 track reconstruction with the mkFit algorithm, CMS-DP-22/018, CMS Collaboration

Contact - leonardo.giannini@cern.ch