TPC looper finder status

J. A. Briffa, B. Costa, G. Valentino
University of Malta

*Thanks to M. Ivanov, D. Rohr, K. Schweda, R. Shahoyan, J. Wiechula
for valuable discussions*

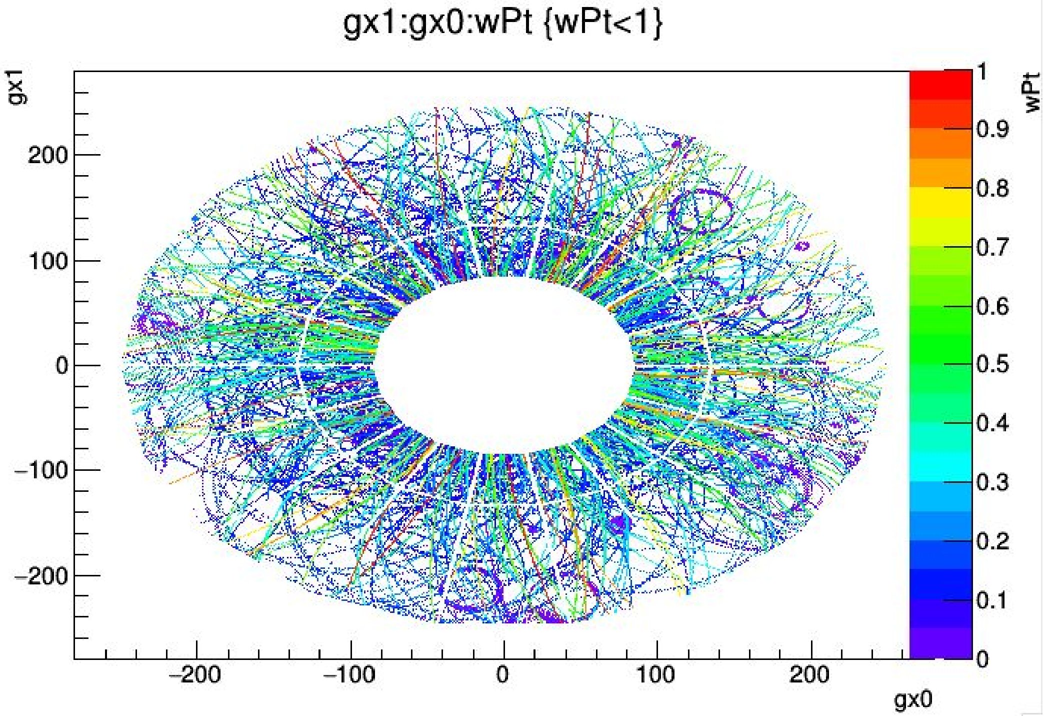
ALICE Offline week, November 2017

Outline

- Introduction
- Feature extraction based on HT (2D) and Helix fit (3D)
- Feature selection
- Machine Learning with GBC and Neural Networks
- Results
- Conclusions

Introduction

- The output from the TPC is currently ~5 PB/year, and due to increase in Run 3.
- Very low momentum tracks (e.g. <50 MeV) are not used for physics analysis, called loopers.
- Loopers make up a significant proportion of the raw data
- It is desirable to remove these loopers in order to achieve better data compression.



pT (GeV/c) = 0.3 * B(Tesla) * r(m)

Ground truth - theoretical

Definitions, assuming MC data has perfect truth:

- Looper: any portion of a track with pT < 40 MeV (~10% in the available MC data)
- Physics track: any portion of a track with pT > 50 MeV
- True positive: a cluster correctly tagged as belonging to a looper
- False positive: a physics track cluster wrongly tagged as being part of looper

In reality, uninteresting data (junk) can be one of:

1) Noise: coming from electronics, not part of any track
2) Clusters belonging to low pT (< 40 MeV) loopers
3) Clusters from loopers of physics tracks with too large inclination angle above the pad row. These are not good for tracking.

The proposed looper finder is only looking for (2), which is 70-80% of junk.

Looper finder algorithm - 2D

- The Hough transform accumulator space A is computed using a Gaussian kernel, and normalizing for circle arc length and radial distance from main beam axis:

$$A(x,y,r) = \sum_{i=1}^{n} G(r - \sqrt{(x_i - x)^2 + (y_i - y)^2}, \sigma) \times \frac{(x_i^2 + y_i^2)^{0.75}}{2\pi r}$$

where: $$G(x,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}$$

- The accumulator space is discretized with a resolution of 1cm in *x,y,r*
- The limits of *x,y* correspond to the detector geometry
- The upper limit of *r* corresponds to the user-defined threshold on Pt, while the lower limit is fixed at 1 cm
- In general we use a Gaussian kernel with fixed σ = 1 cm, truncated to ±2 cm
- Each entry in A represents the likelihood for a looper with axis at *x,y* and radius *r*

Looper finder algorithm - 2D

- Then, the label or likelihood for a particular cluster is computed as:

$$\lambda_i = \max_{x,y,r} A(x,y,r) G(r - \sqrt{(x_i - x)^2 + (y_i - y)^2)}, \sigma)$$

- This represents the likelihood that the given cluster forms part of any looper within the parameter range of the accumulator space.
- Note that the algorithm does not associate a cluster to a specific looper, but only computes the aggregate likelihood over all considered loopers.
- This allows to carry all information until the final threshold decision is taken per cluster.

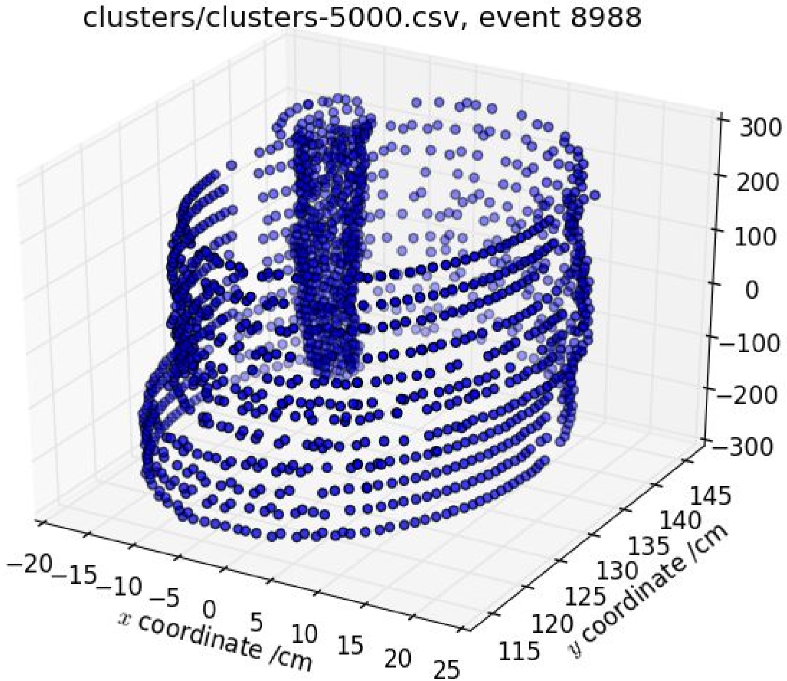
Looper finder algorithm - 3D

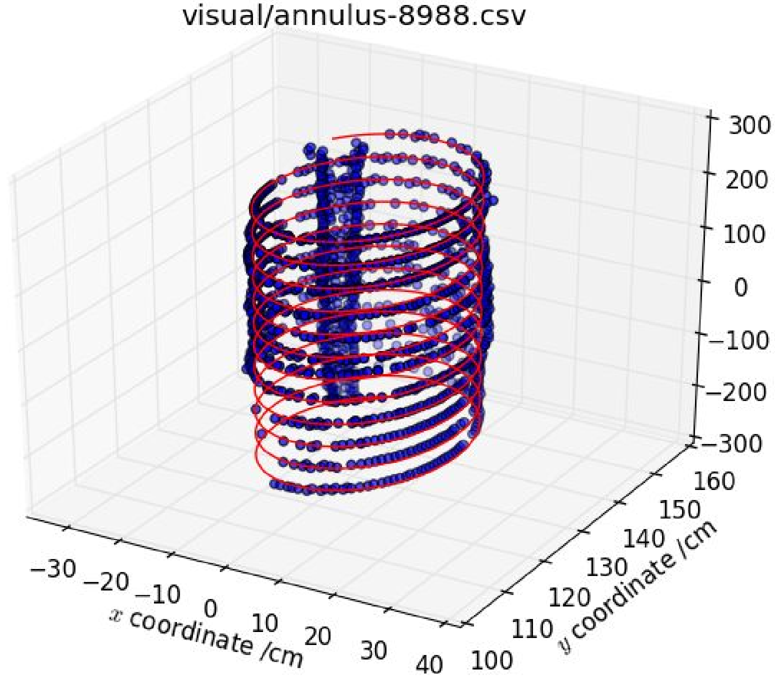
- In the 3D version, after computing the 2D accumulator space A, we select the set of candidate circles as the parameter tuples (*x,y,r*) that correspond to the highest contribution from at least one cluster.
- This is determined by:

$$U(\arg\max_{x,y,r} A(x,y,r)G(r-r_i,\sigma)) = 1 \forall i \in \{1,\ldots,n\}$$

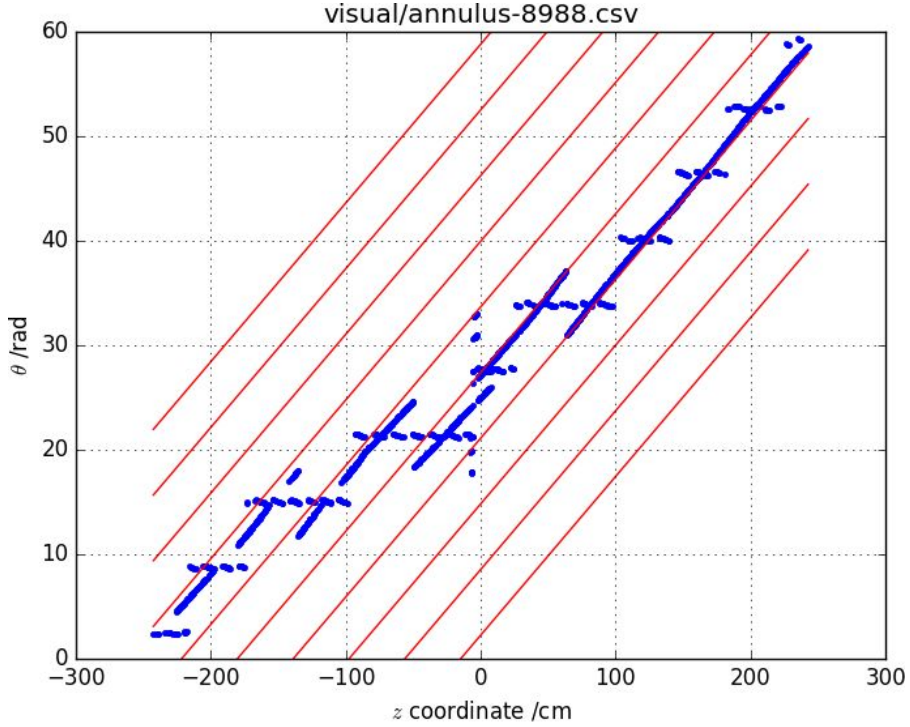
- This step effectively maps a cluster to a 2D candidate looper.
- Next, we iterate over the parameter space for tuples where *U*(*x*,*y*,*r*) = 1; for each:
  - We extract the list of clusters that fit within an annulus of ±2 cm
  - We determine the helix parameters that best fit the extracted clusters

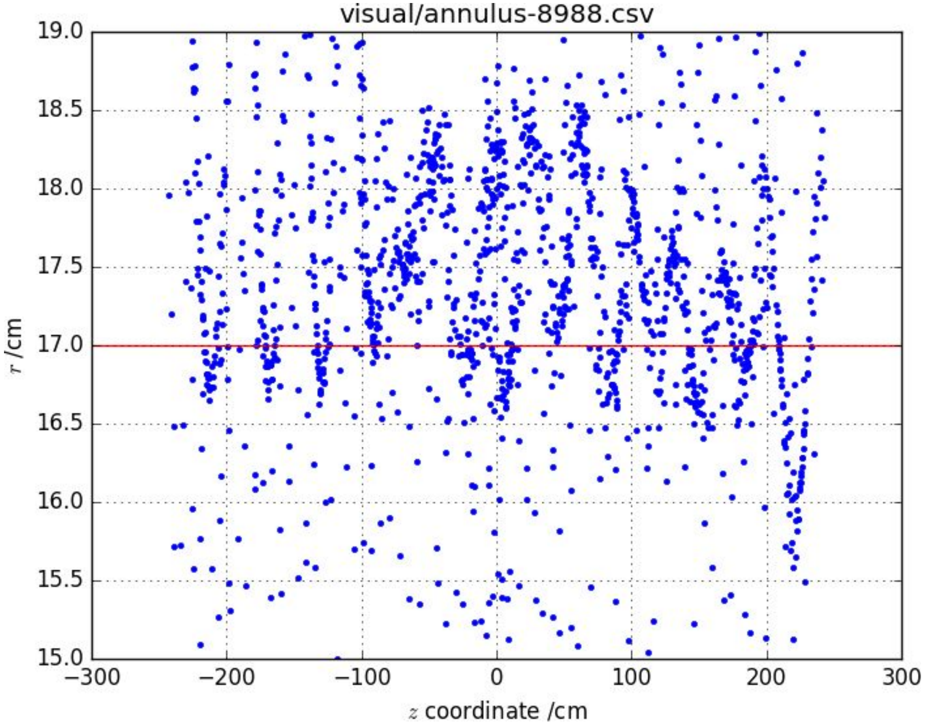
Looper finder algorithm - 3D



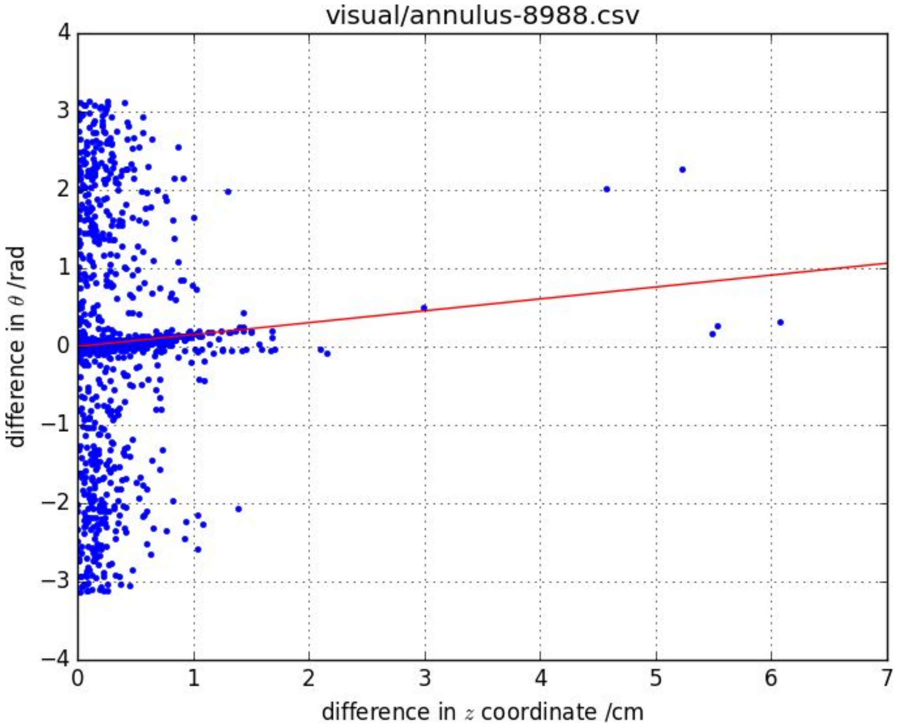


Looper finder algorithm - 3D



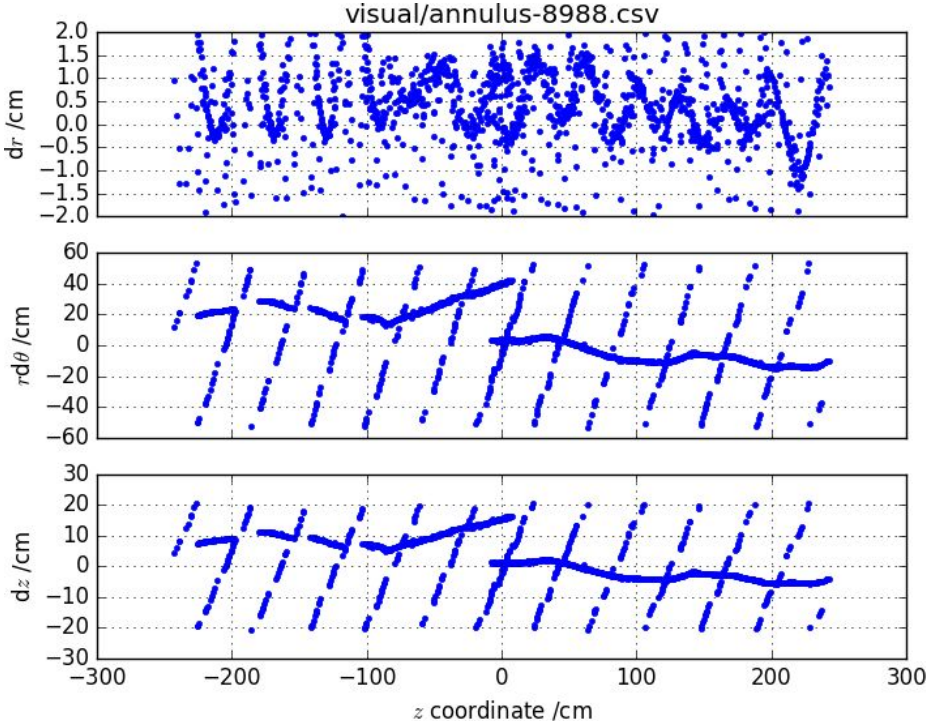


Looper finder algorithm - 3D



- We sort these in increasing *z* and calculate the finite difference in *z* and θ for adjacent clusters, where θ is the radial angle of the looper.
- We fit a straight line through the origin to δθ:δz to fit helix pitch
- From this we can fit starting angle
- Equations:
  - $\theta = a_1.z + a_0 \bmod 2\pi$
  - $a_1 = \text{median}(\delta\theta/\delta z)$
  - $a_0 = 2\pi - \text{median}(a_1.z-\theta \bmod 2\pi)$

Looper finder algorithm - 3D



For each cluster we determine goodness of fit with obtained helix.

Can compute discrepancy in:

- Radius (r_i-r)
- Angle, Arclength (assuming correct z)
- z coordinate (assuming correct θ)
- Closest distance (complicated)

Looper finder algorithm - 3D

- Each cluster is associated with the helix to which it gives the highest 2D contribution
- For this association, the cluster is labelled with its contribution to and discrepancies with the associated helix
- In principle, we have now accumulated several features which could be used by a machine learning algorithm to classify a cluster as being looper or non-looper
- In practice, a reduced feature space (13→5) was found to be sufficient

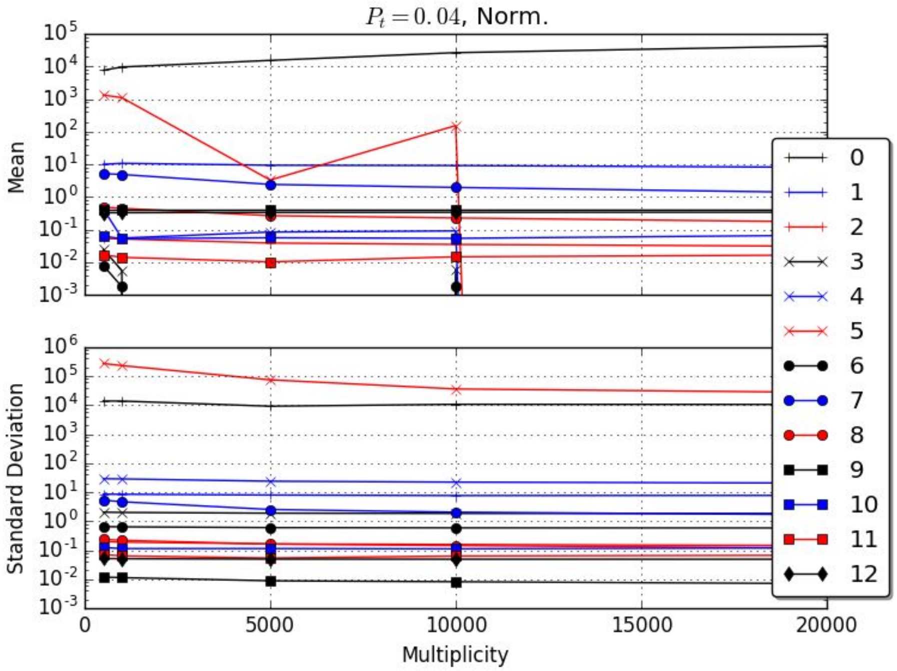
Features Selected

- Feature vector per cluster:
  1. **A(x, y, r): 2D HT accumulator value**
  2. **r**
  3. **δr**
  4. $\delta\theta$
  5. $r\delta\theta$
  6. δz
  7. **$\delta z/z_T$**
  8. **Center of gravity error**
  9. Center of gravity error / r
  10. $G(\delta r)$
  11. G(darc)
  12. $G(\delta z)$
  13. $G(\delta z/z_T)$

These elements are only computed at the end after assigning each cluster to one circle / helix

Features - Dependence on Multiplicity

- Features:
  0. **A(x, y, r): 2D HT accumulator value**
  1. **r**
  2. **δr**
  3. δθ
  4. rδθ
  5. δz
  6. **δz/z_T**
  7. **Center of gravity error**
  8. Center of gravity error / r
  9. G(δr)
  10. G(darc)
  11. G(δz)
  12. G(δz/z_T)



Feature Normalization

- Initially, we normalized all features to zero mean, unit variance
- Now:
  - **A(x,y,r):** normalized by number of clusters in file
  - **dr:** absolute value
  - **r:** normalized by radius corresponding to looper threshold [40 MeV]
  - **cog error:** normalized by radius corresponding to looper threshold [40 MeV]
  - **dz/zT:** absolute value
- Normalization is now independent of data
- Robust to training and testing at different multiplicity

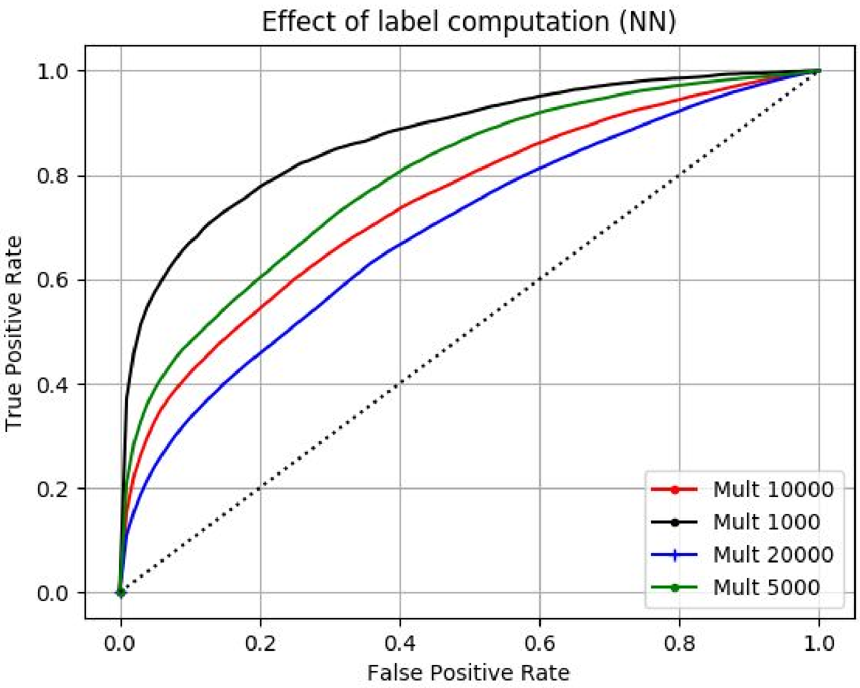
Machine Learning algorithms

- Supervised learning ML algorithms were used to train a classifier on the features to classify between clusters which belong to loopers and those which belong to non-loopers.
- Gradient Boosting Classifier, Support Vector Machines and Neural Networks.
- Cross-validation used to determine best parameters in each case

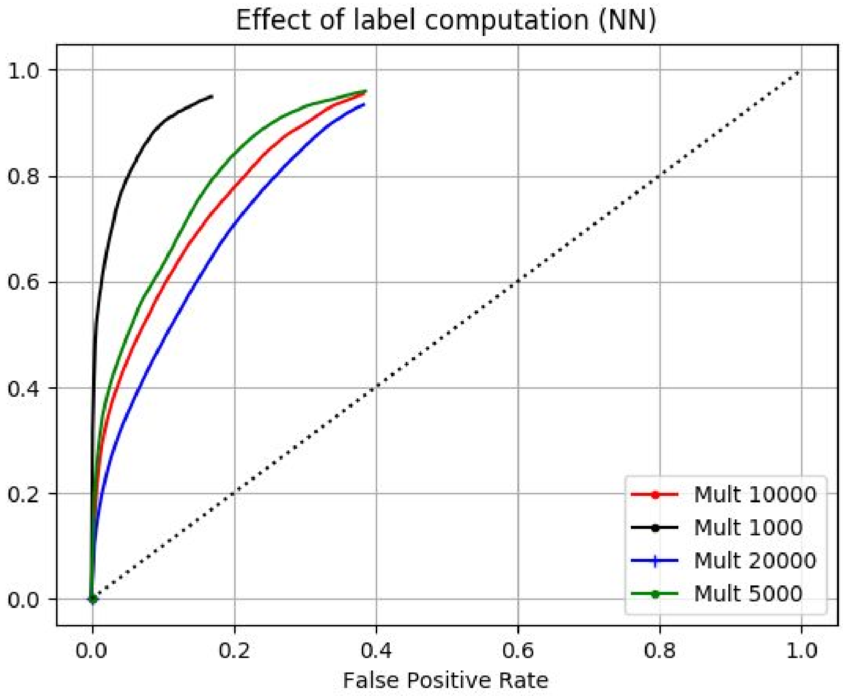
- Bottom line: Neural networks worked best
  - Simple topology (2 hidden layers)
  - Much faster training and testing

Results using Neural Networks

Without pre-filtering of tagged-as-physics clusters

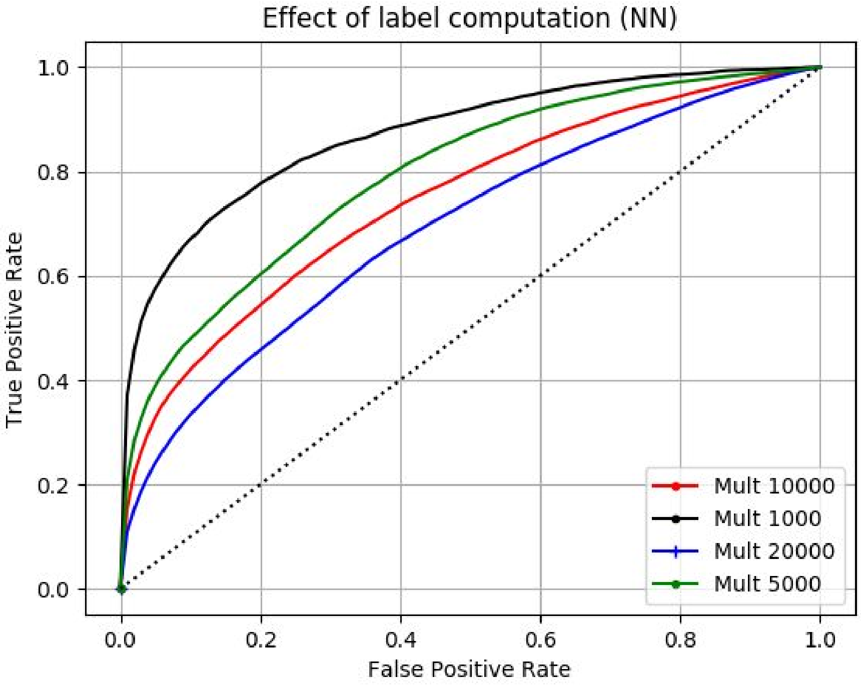


Pre-filtering of tagged-as-physics clusters in ML

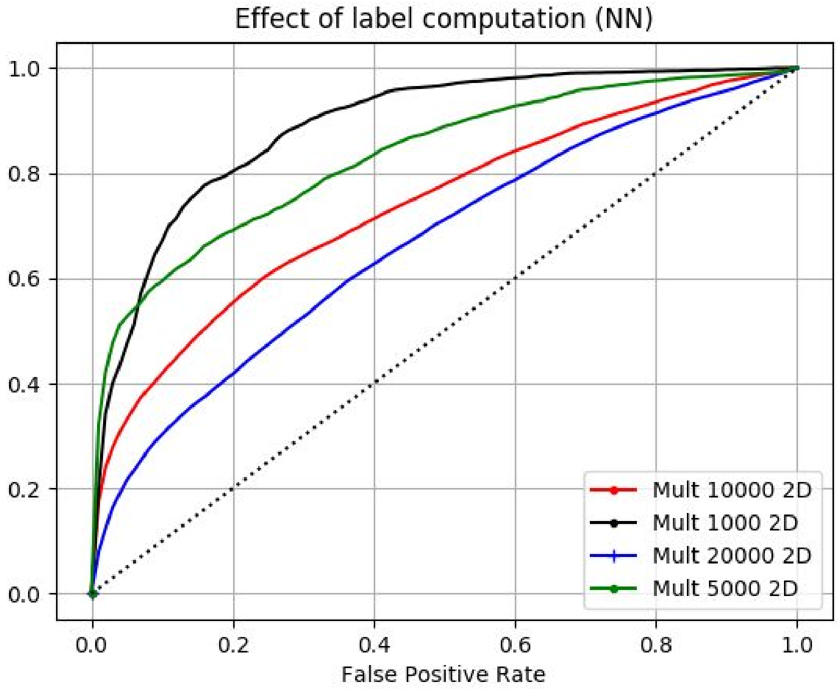


Results using Neural Networks

With 5 features (2D+3D)

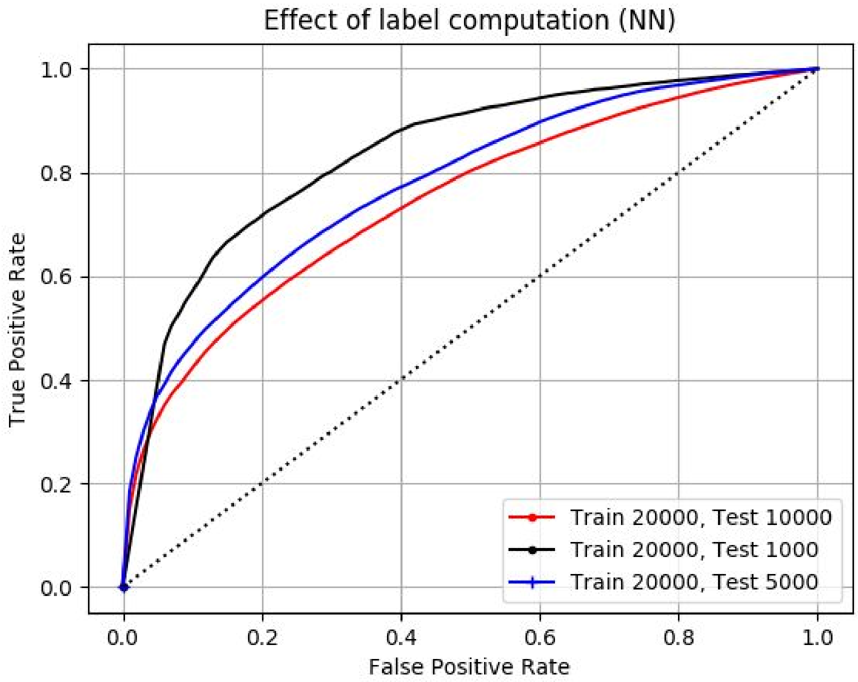


With 4 features (2D only)



Robustness of Classifier

- Training on one data set (mult 20000)
- Testing on other data sets (1000-10000 multiplicity)



Conclusions

- Is 3D worth it?
  - 2D-only features gives similar performance to 2D + 3D features
  - Do we need to re-examine helix fitting algorithm?
  - Consideration of computational load
- Classifier performance
  - Run algorithm on newly available data set (full timeframe of 25 ms)
- Computational performance & implementation issues
  - Optimizing existing GPU implementations for speed
  - Using a GPU implementation of NN classifier
  - Timings for full timeframe
  - Porting GPU algorithms to OpenCL for use on AMD GPUs
  - Update to use newly defined O^2 data formats