HEP.TrkX
Charged Particle Tracking using Graph Neural Networks

TrackML Challenge Grand Finale

CERN, July 1-2 2019

Jean-Roch Vlimant for the HEP.TrkX project

Outline

- Forewords on tracking with ML
- Dataset and graph neural network models
- Results and outlooks

Motivations

Current algorithms for tracking are highly **performant physics-wise** and scale **badly computation-wise**

Faster implementations are possible with **dedicated hardware**

Turn to deep learning for new approaches

Machine Learning for Tracking

Zagoruyko et al, [1604.02135](1604.02135)

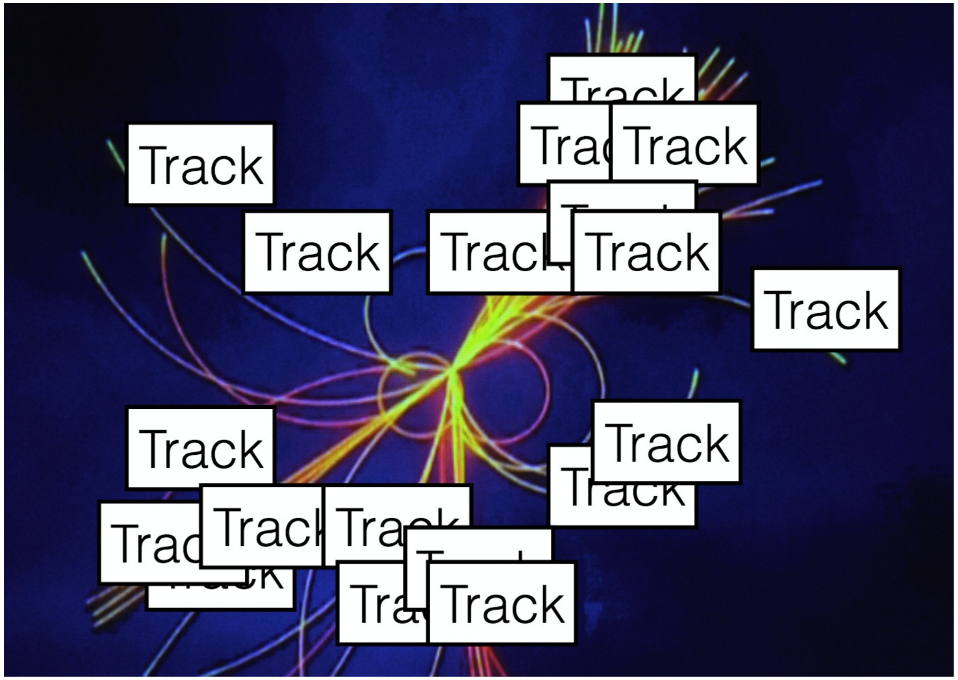


Photo by Pier Marco Tacca/Getty Images

Many possible ways to cast the algorithm of tracking, or part of the current algorithms in a machine learning problem

HEP.TrkX
J.-R. Vlimant

BERKELEY LAB

HEP.TrkX Project

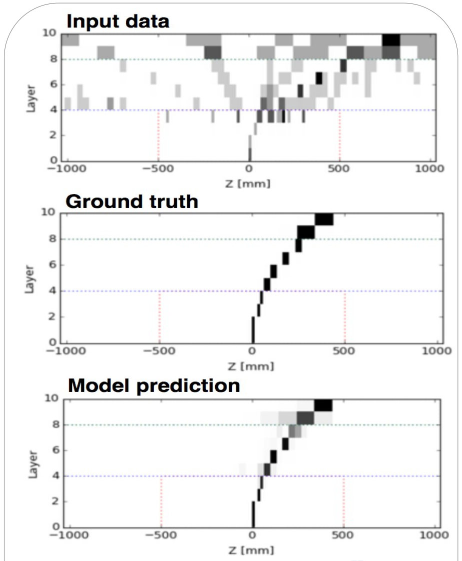
- Pilot project funded by DOE ASCR and COMP HEP
- Part of HEP CCE
- Mission
  - Explore deep learning techniques for track formation

- People
  - **LBL :** Paolo Calafiura, Steve Farrell, Mayur Mudigonda, Prabhat
  - **FNAL :** Giuseppe Cerati, Lindsey Gray, Jim Kowalkowski, Panagiotis Spentzouris, Aristeidis Tsaris
  - **Caltech :** Dustin Anderson, Josh Bendavid, Pietro Perona, Maria Spiropulu, Jean-Roch Vlimant, Stephan Zheng

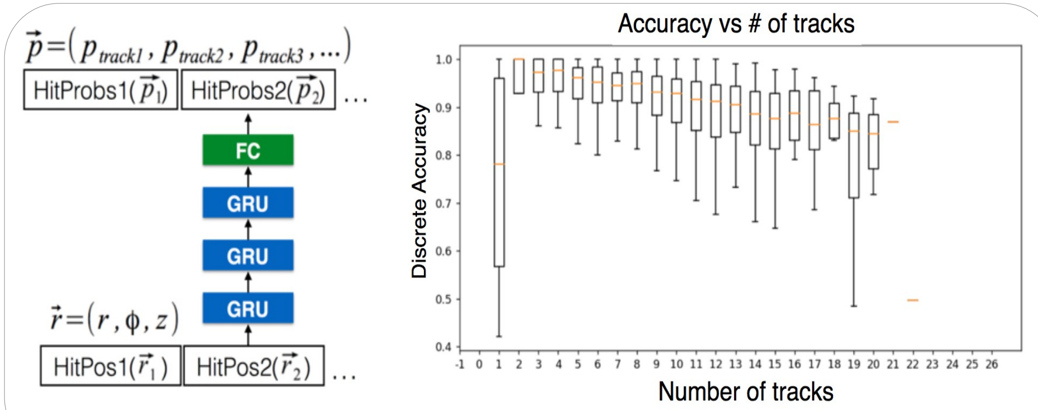
- All material available under https://heptrkx.github.io/

Data Representation

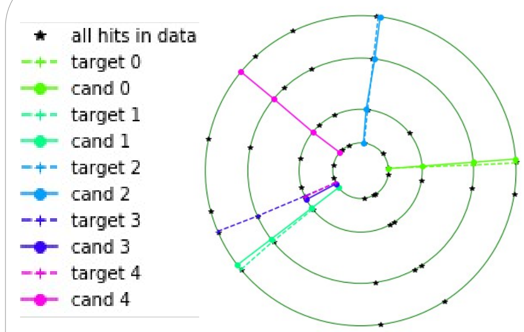
HEP.TrkX Approaches



Track following with RNN



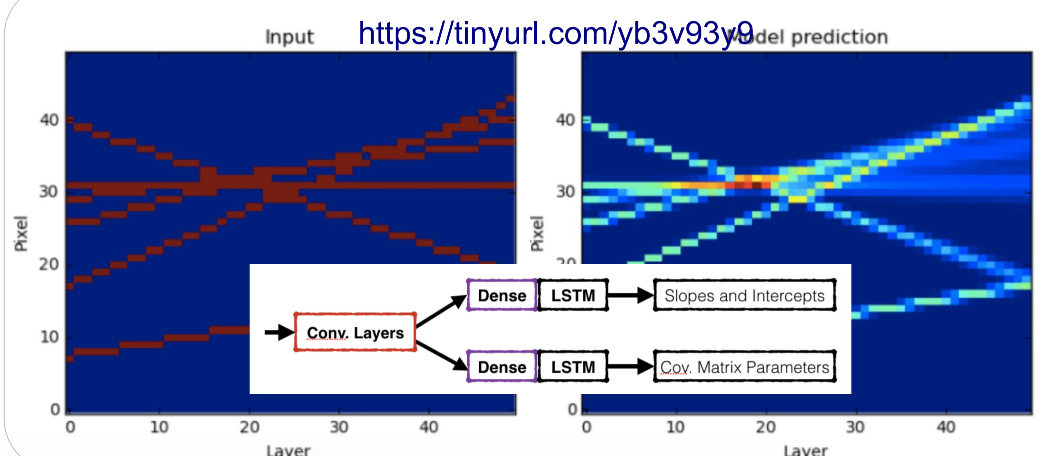
End-to-end hit assignment

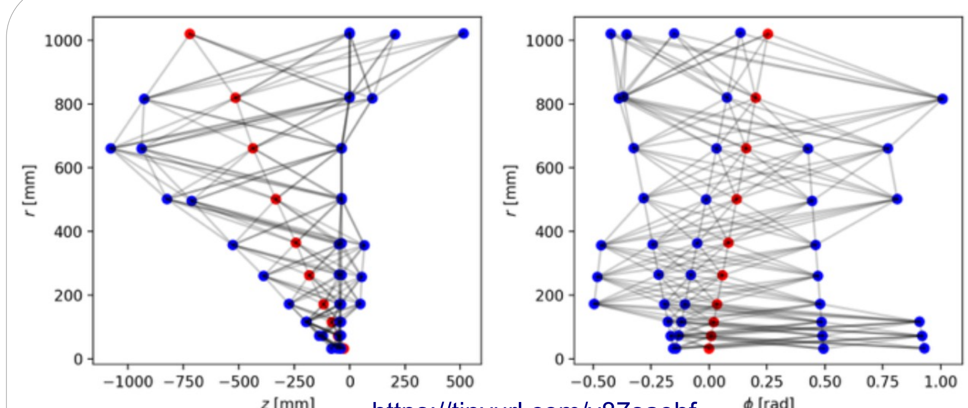


Seq-to-seq track finding

https://heptrkx.github.io/

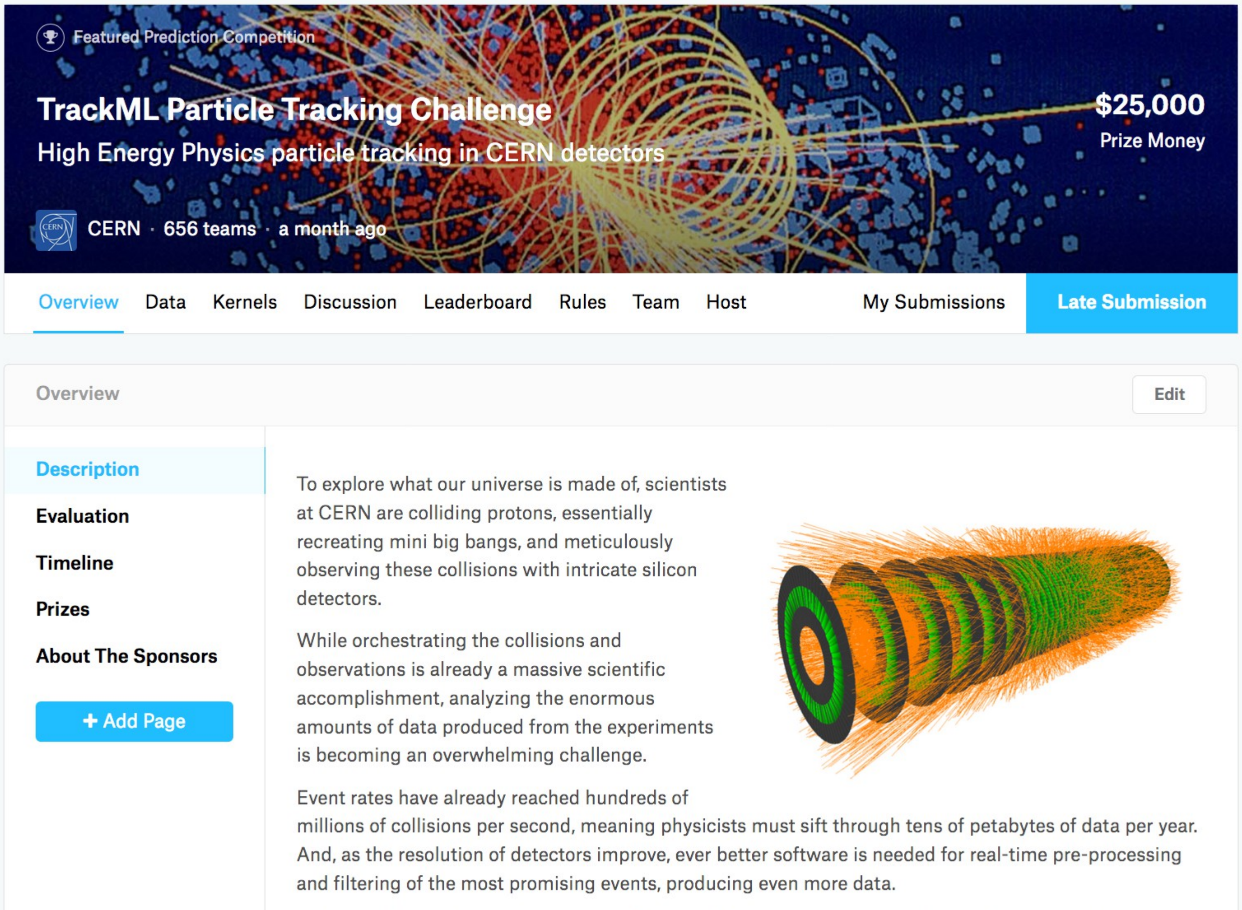
https://tinyurl.com/yb3v93y9





https://tinyurl.com/y87saehf

Charged Particle Tracking Dataset

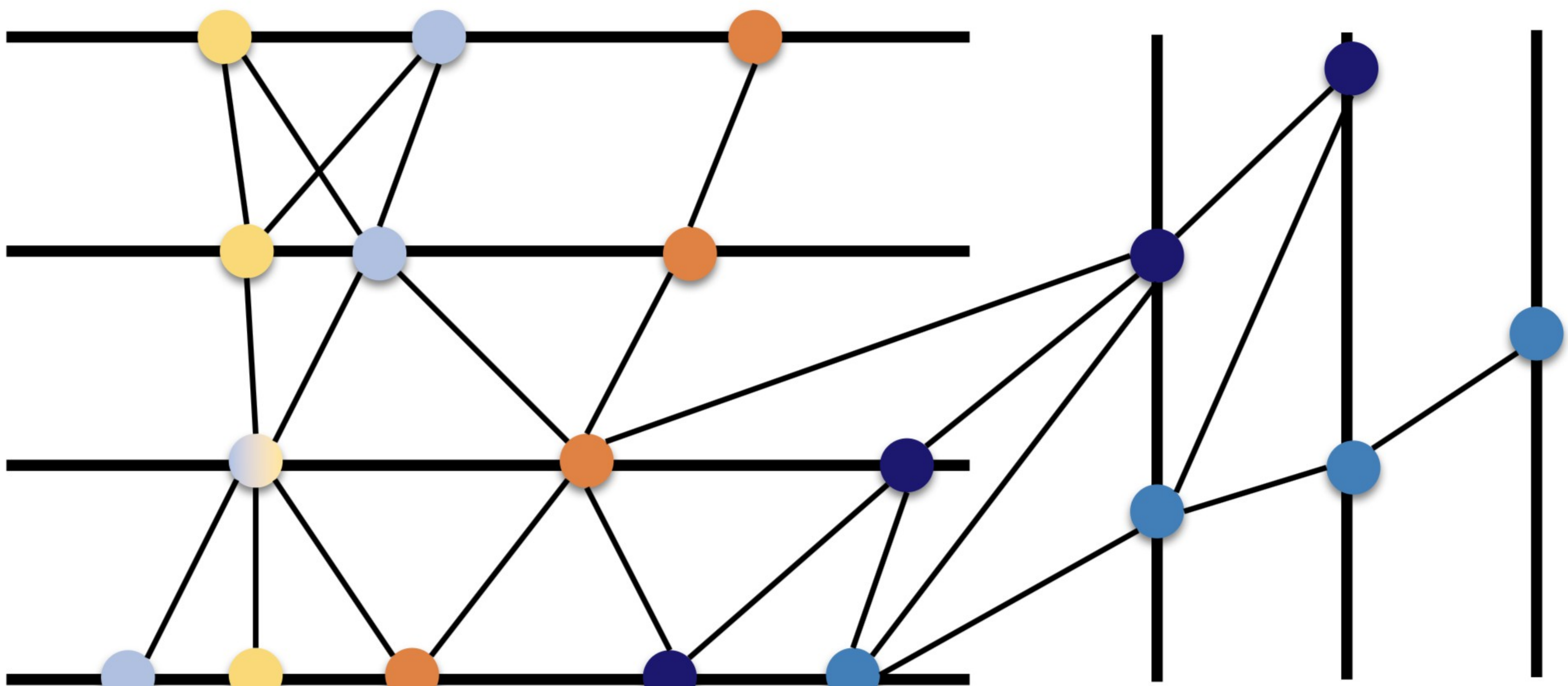


https://www.kaggle.com/c/trackml-particle-identification
https://competitions.codalab.org/competitions/20112

- This work uses the public dataset of the TrackML Particle Tracking Challenge (Kaggle, codalab).
- Simulating the dense environment expected for HL-HLC. Average of 200 proton-proton interaction per bunch crossing.

TrackML sponsors

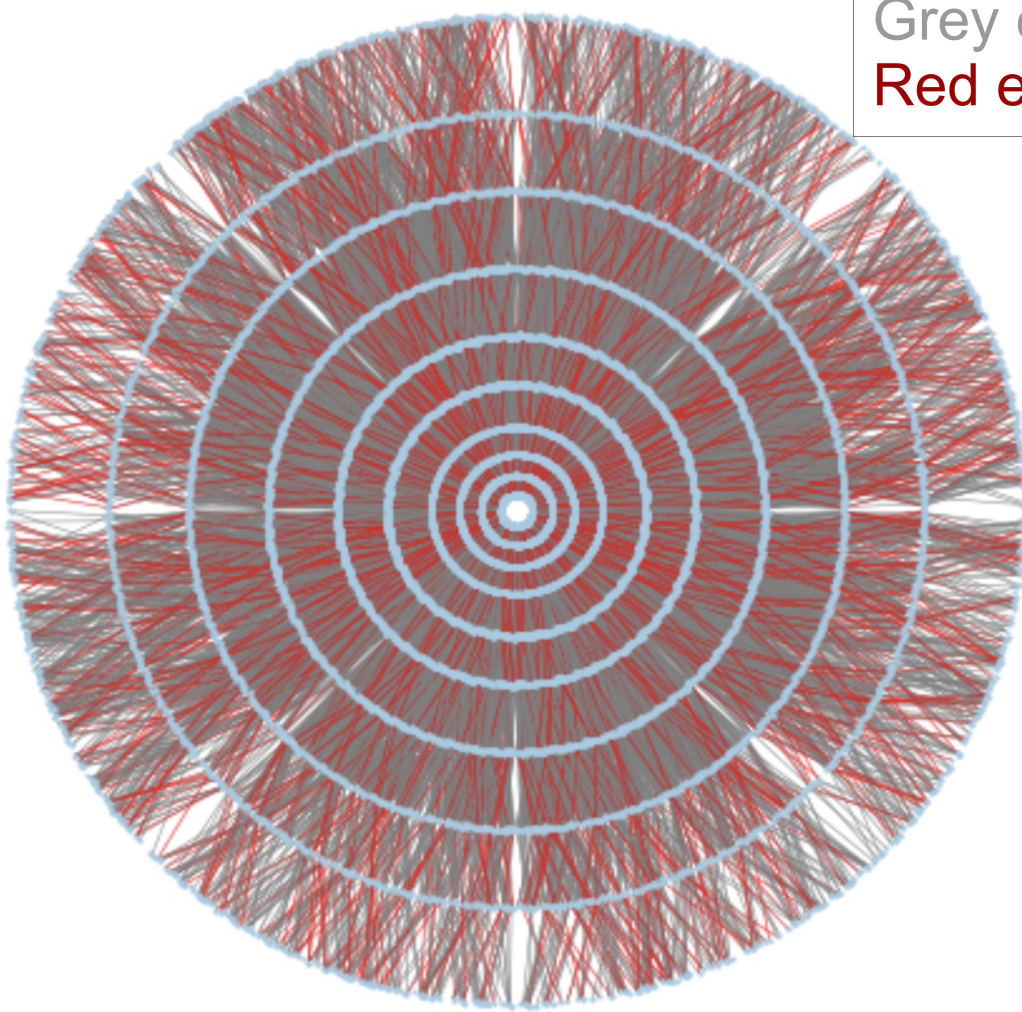
Tracker Hit Graph



Directed graph constructed

- One tracker hit per node
- Direct edge inside-out

Edge Classification

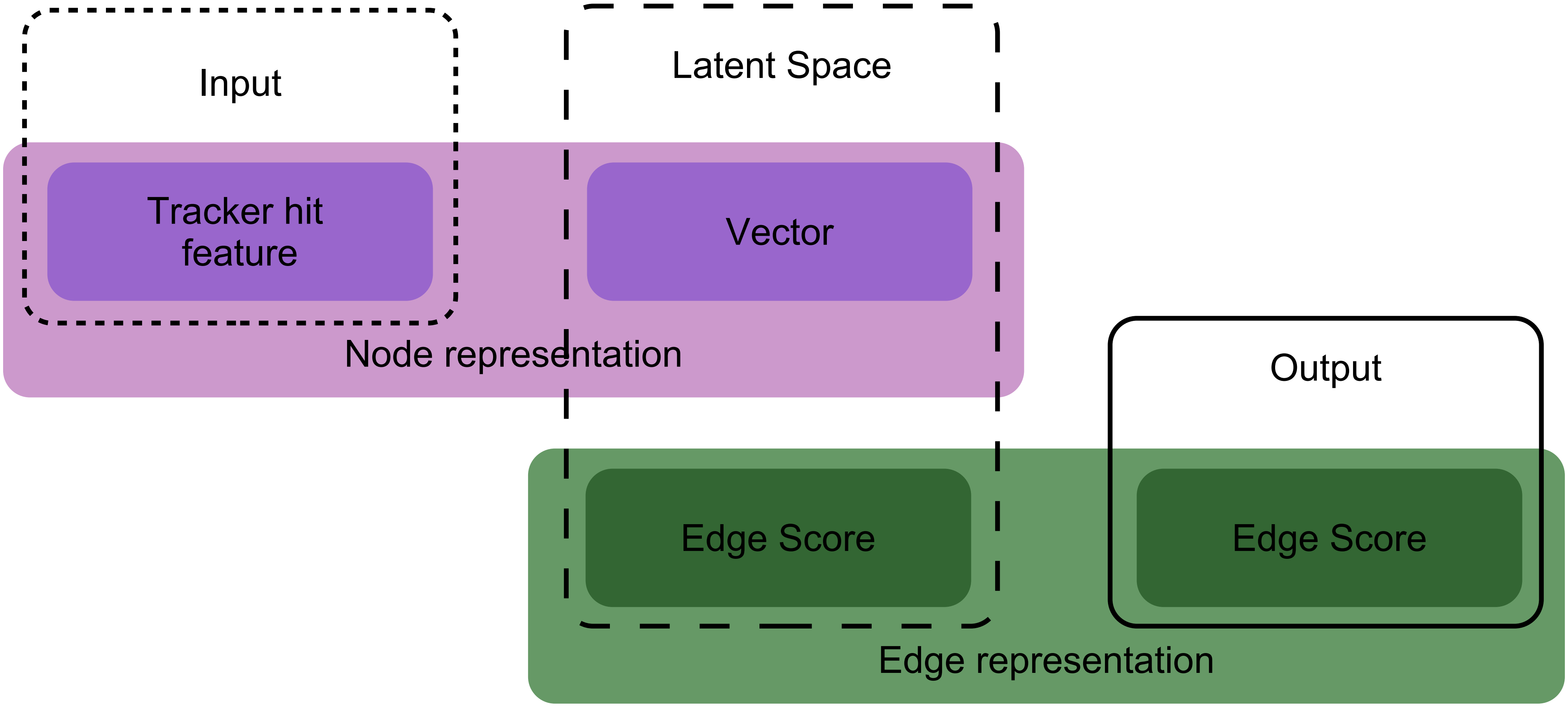


Grey edge: fake
Red edge : true

HEP.TrkX
J.-R. Vlimant
BERKELEY LAB

Edge Classification with Graph Neural Network

Node & Edge Representations

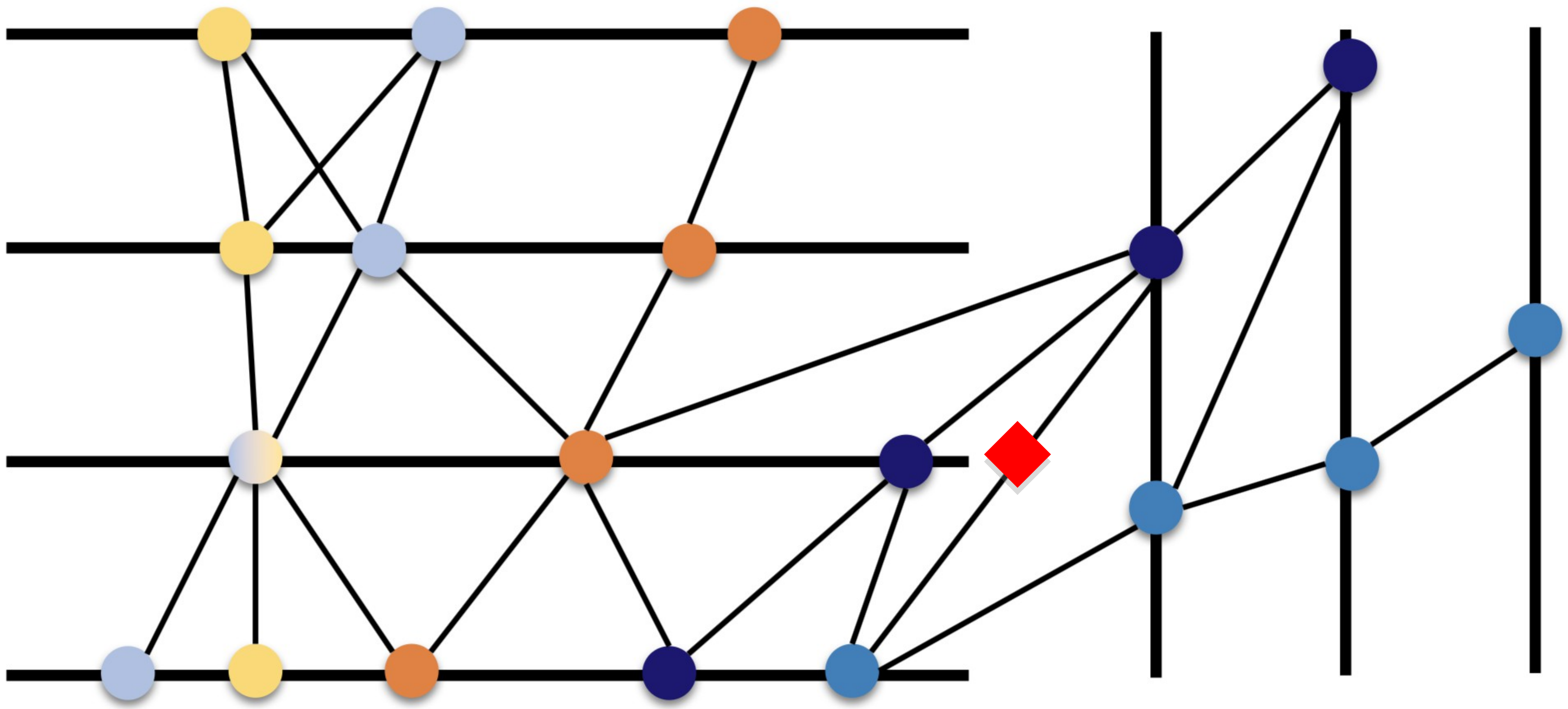


Latent edge representation taken to be the classification score.
“attention” mechanism to the relevant edges.

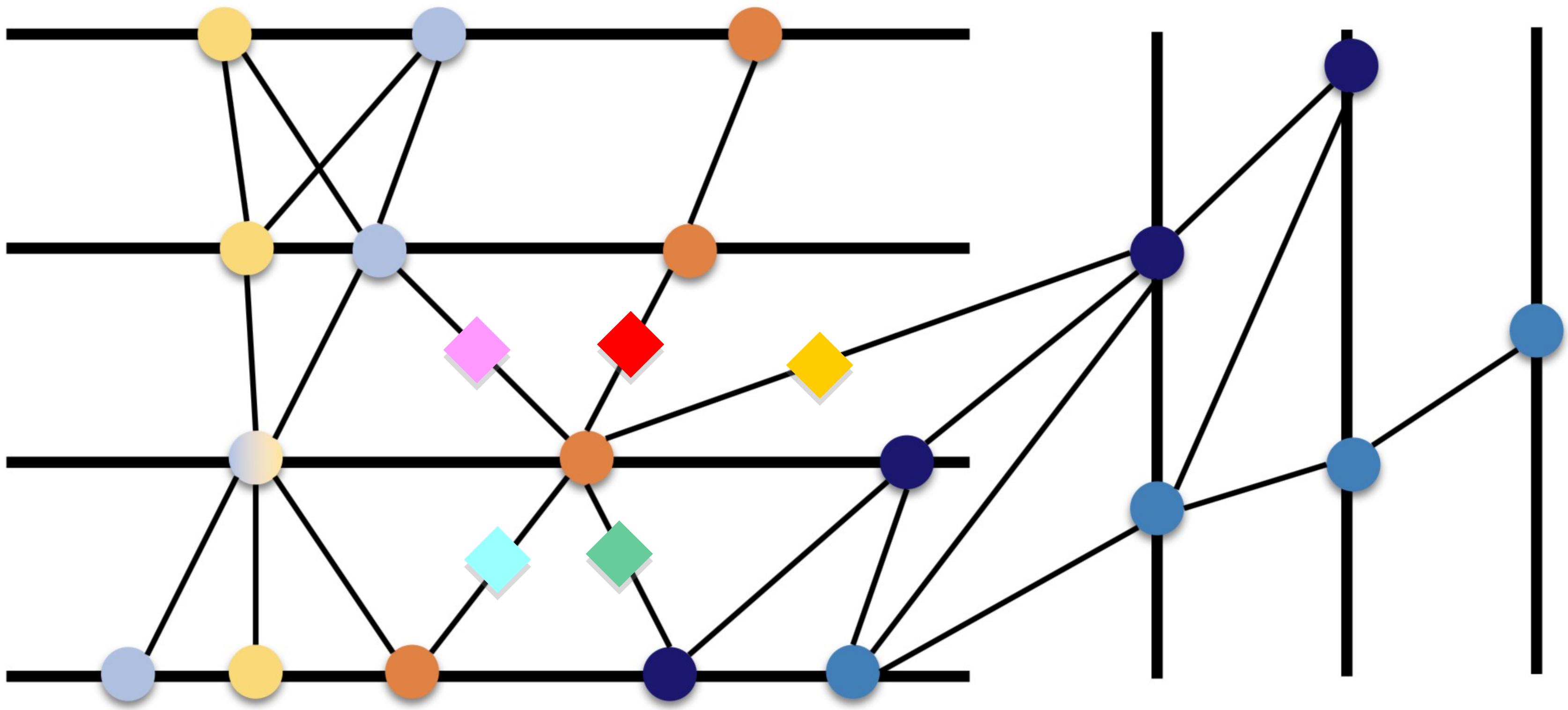
Neural Networks

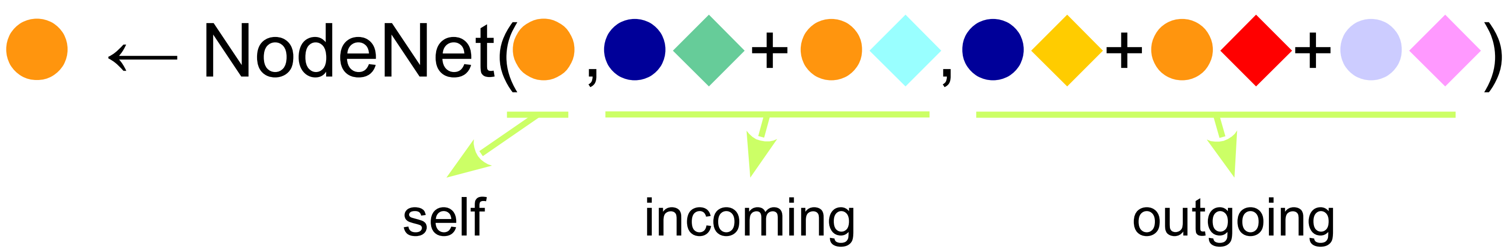
- **Input Network**
  - ➢ Transforms from hit features (r,φ , z) to the node latent representation (N for 8 to 128)
  - ⬩ Dense : 3→...→N
- **Edge Network**
  - ➢ Predicts an edge weight from the node latent representation at both ends
  - ⬩ Dense : N+N→...→1
- **Node Network**
  - ➢ Predicts a node latent representation from the current node representation, weighted sum of node latent representation from incoming edge, and weighted sum
  - ⬩ Dense : N+N+N→...→N

Edge Network



Node Network



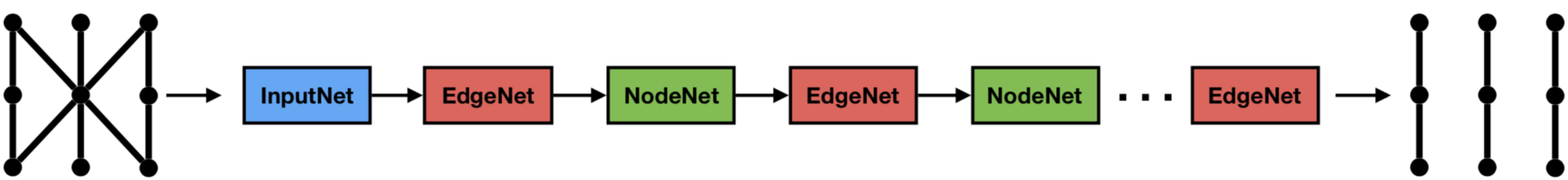


Information Flow

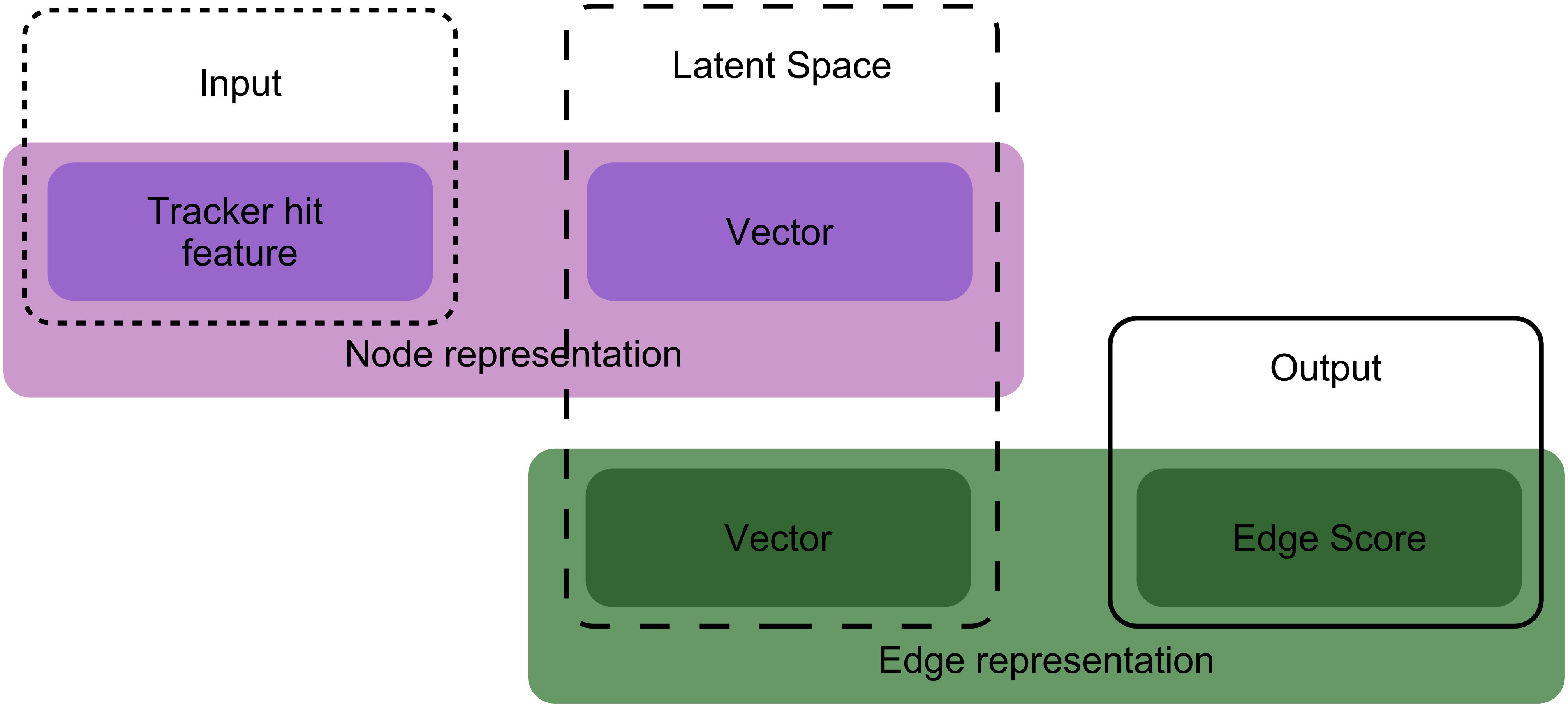
- Graph is sparsely connected from layer to layer
- InputNet + EdgeNet + NodeNet only correlates hits information on triplet of layers
  - The information from the outer hits and inner hits are not combined

→ Correlates hits information through multiple (7) iterations of (EdgeNet+NodeNet)

→ Implemented in Torch https://github.com/HEPTrkX/heptrkx-gnn-tracking

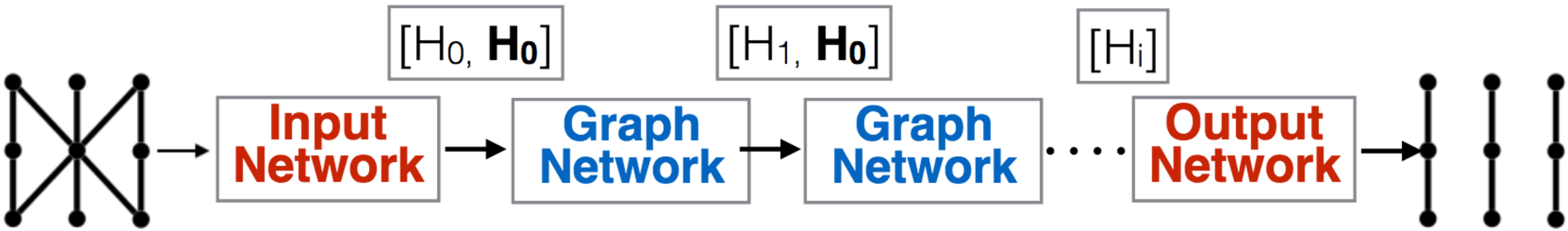


Node & Edge Representations



Edge representation is not the edge score.
Final edge score extracted from the latent edge representation.

Message Passing Model

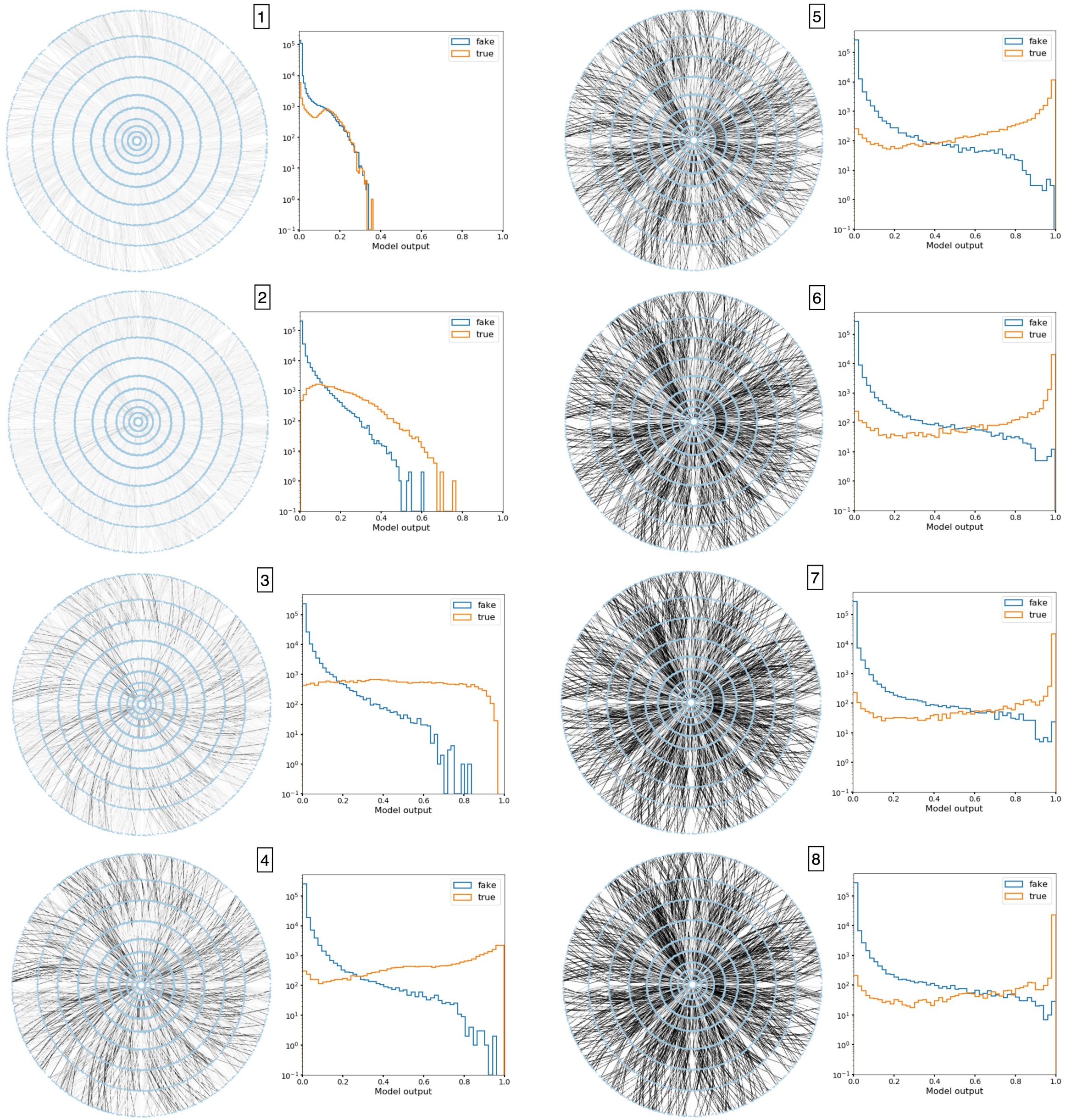


- Same graph connectivity
- No explicit attention mechanism
- Edge representation computed from end-nodes features
- Node representation computed from the sum over all connected edges

→ Correlates hits information through multiple (8) iterations of (Graph Network)

→ Uses https://github.com/deepmind/graph_nets TF library

Information Flow



- Checking edge score after each step of graph network.
- Effective output of the model is in step 8.
- Full track hit assignment learned in last stages of the model.
- Tracklets learned in intermediate stages.

Problem Size Considerations

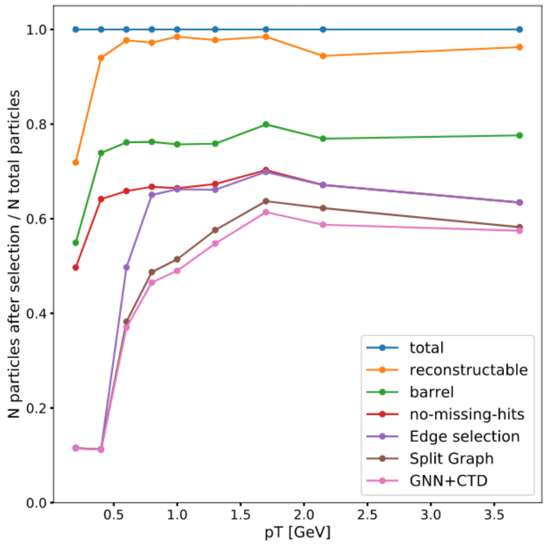
Dealing with Large Graphs

- Full event embedding
  - A graph with ~120k nodes (14.4B edges) and ~1M potential edges
  - Very large graph

- Split the problem
  - currently using 8/16 sectors in φ
- Identify disjoint sub-graphs
  - Geometrical cuts, segment pre-classifier, ...
- Implement distributed learning of large graphs
  - Scope of the **Exa.TrkX Project**

Performance

one-event	N-particles	ratio w.r.t Total	ratio w.r.t Reconstructable	relative ratio
Total	11170	100%		100%
Reconstructable	9635	86%	100%	86%
Barrel	7492	67%	78%	78%
No-missing hits	6600	59%	69%	88%
Edge selection	3114	28%	32%	47%
Split graph	2668	24%	28%	86%
GNN	2590	23%	27%	97%



- Tracks formed with a simple algorithms that traverse the hit graph over high-score edges.

→ Promising performance, once passed acceptance cut for training purpose (to be solved)

→ Further details in Xiangyang talk at Connecting the Dots 2019 https://indico.cern.ch/event/742793/contributions/3274328/

HEP.TrkX
J.-R. Vlimant
BERKELEY LAB

Summary and Outlook

- Graphs is the most nature data representation we could come across.
- Promising performance of graph networks at doing pattern recognition of tracks.
- Finalizing end-to-end solutions.
- Multiple ways to improve the model ; also using domain knowledge.
- Computationally intensive task that requires further work on scaling.

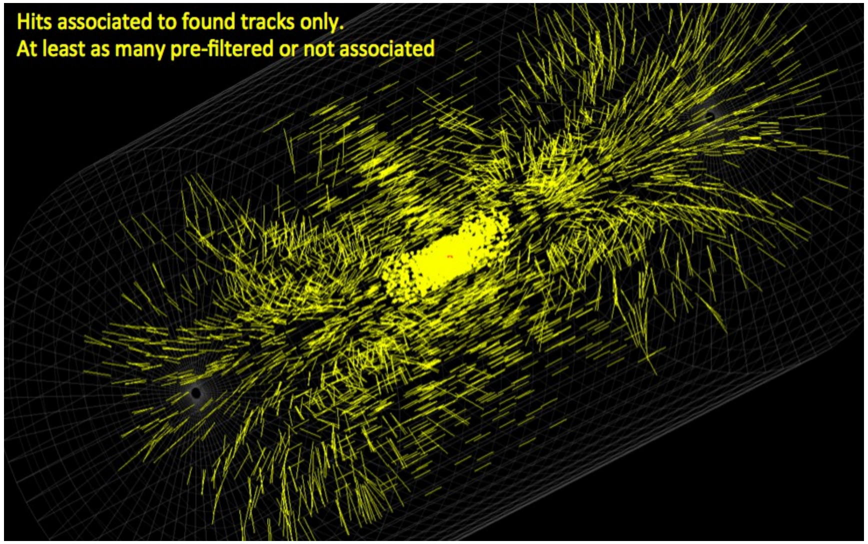
Extra material

Acknowledgments

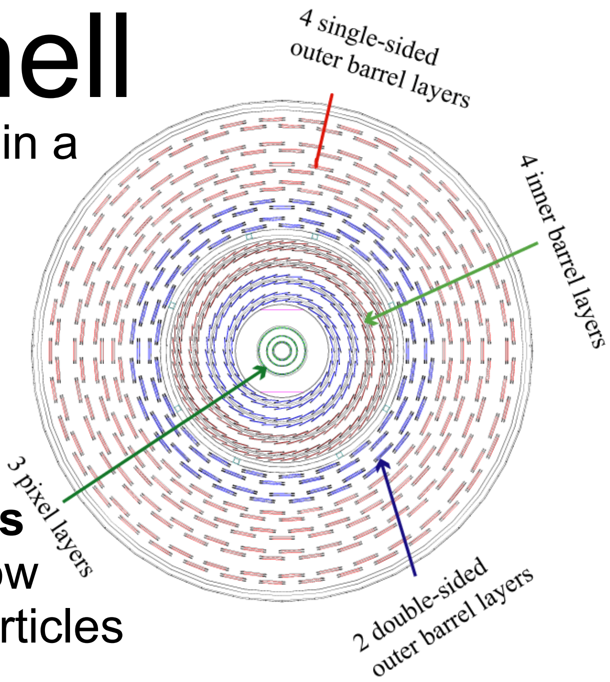
Part of this work was conducted at "iBanks", the AI GPU cluster at Caltech. We acknowledge NVIDIA, SuperMicro and the Kavli Foundation for their support of "iBanks".

HEP.TrkX
J.-R. Vlimant

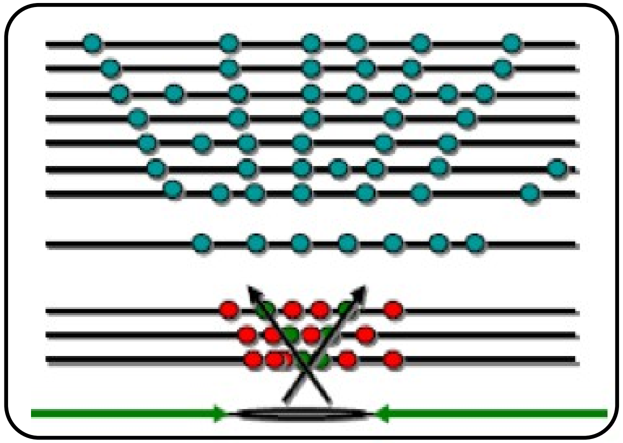
Tracking in a Nutshell



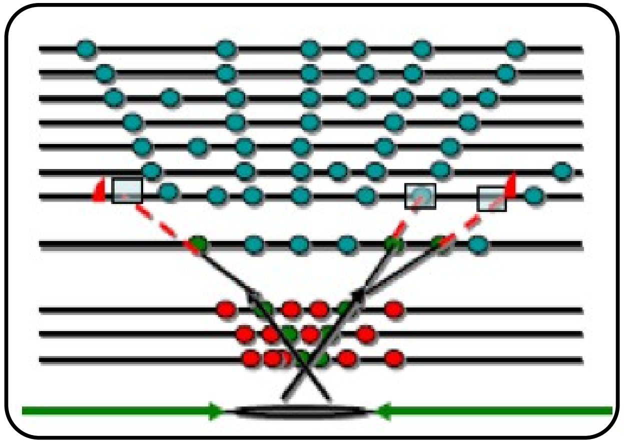
- Particle trajectory bended in a solenoid magnetic field
- Curvature is a proxy to momentum
- Particle ionize silicon pixel and strip throughout several concentric layers
- **Thousands of sparse hits**
- Lots of hit pollution from low momentum, secondary particles

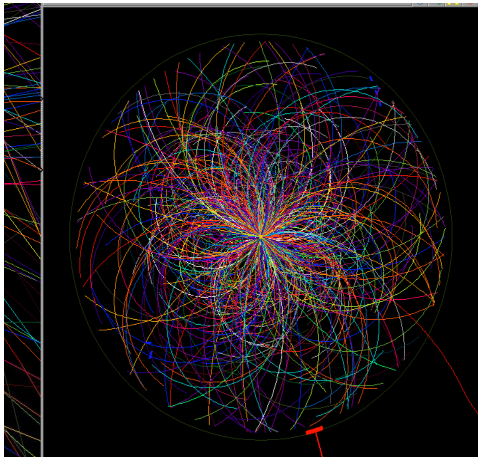


Seeding



Kalman Filter

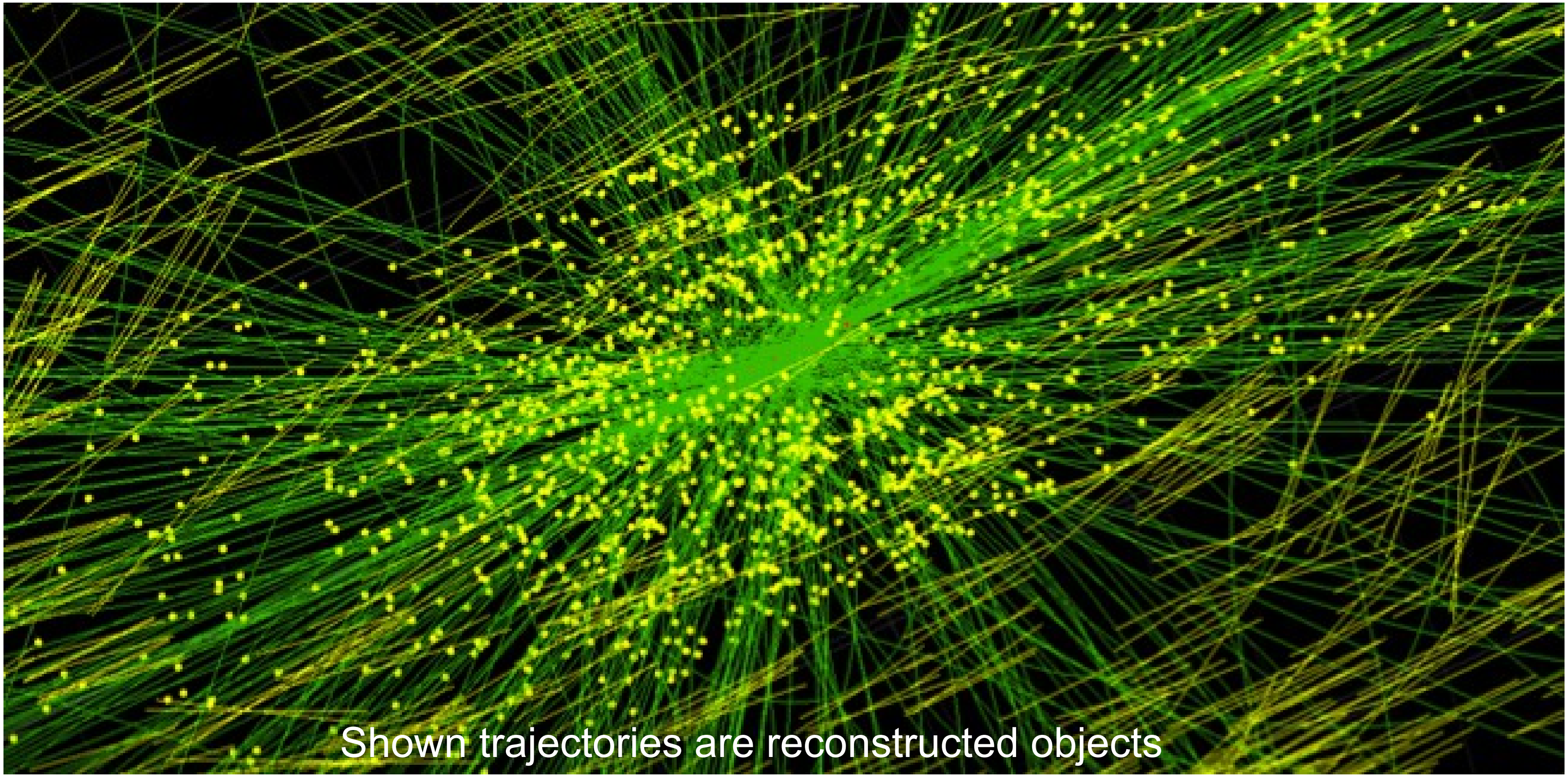




- **Explosion of hit combinatorics** in both seeding and stepping pattern recognition
- **Highly time consuming task** in extracting physics content from LHC data

HEP.TrkX
J.-R. Vlimant
BERKELEY LAB

Complexity and Ambiguity

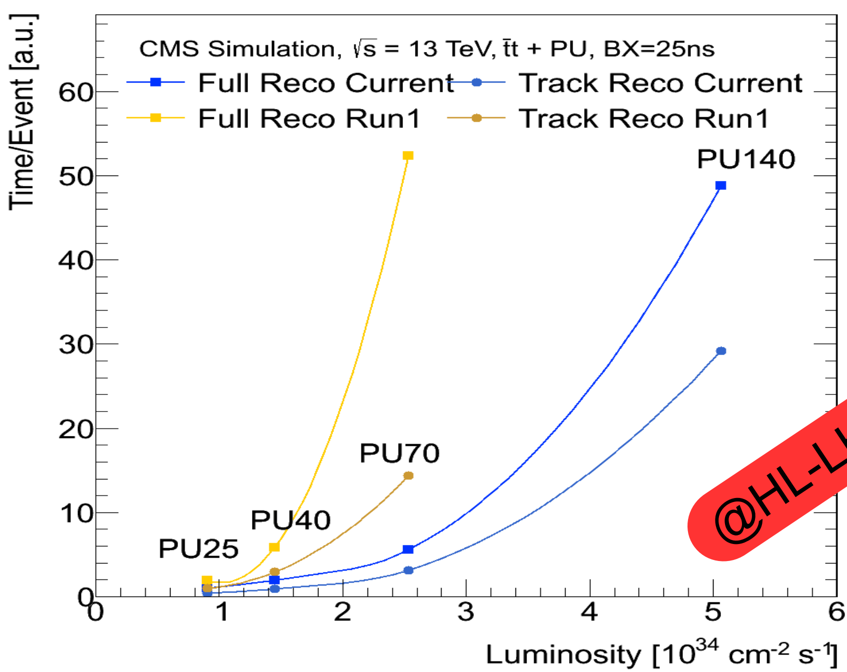


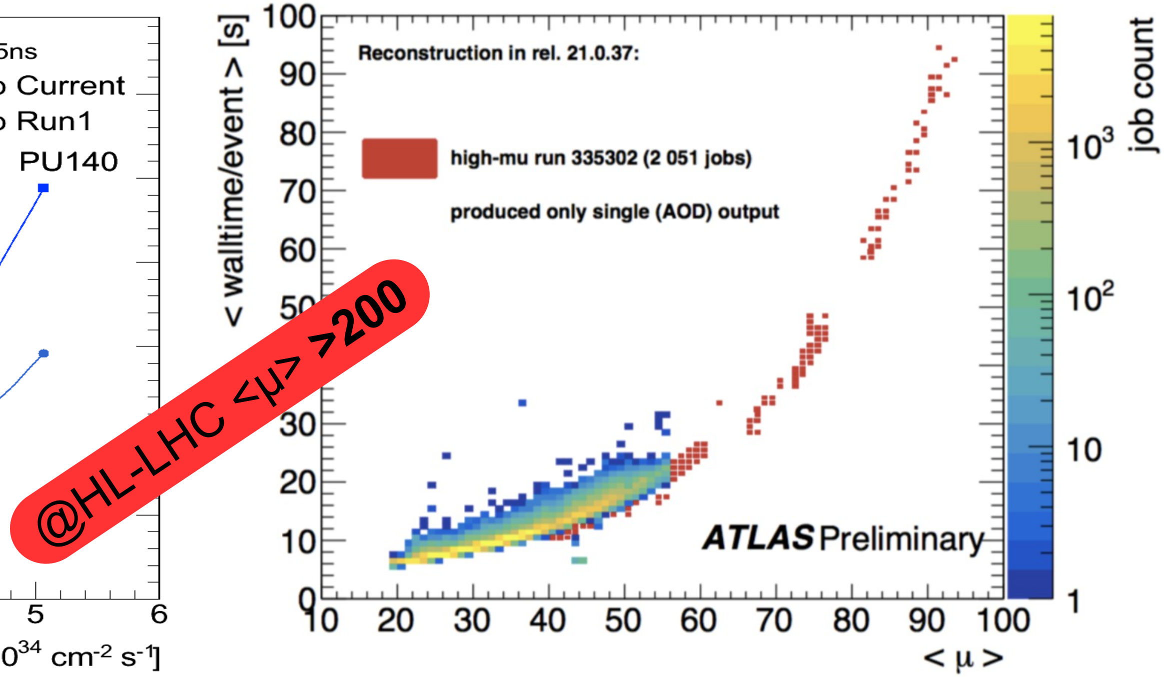
The future holds **much more hits**

High Luminosity LHC
The Challenge

Cost of Tracking

- CPU time consumption in HL-LHC era **surpasses computing budget**
  - → Need for **faster algorithms**
- Charged particle track reconstruction is one of the most **CPU consuming task** in event reconstruction
  - → Optimizations **mostly saturated**
- Large fraction of CPU required in the HLT. **Cannot perform tracking inclusively**
  - → **Approximation** possible in the trigger

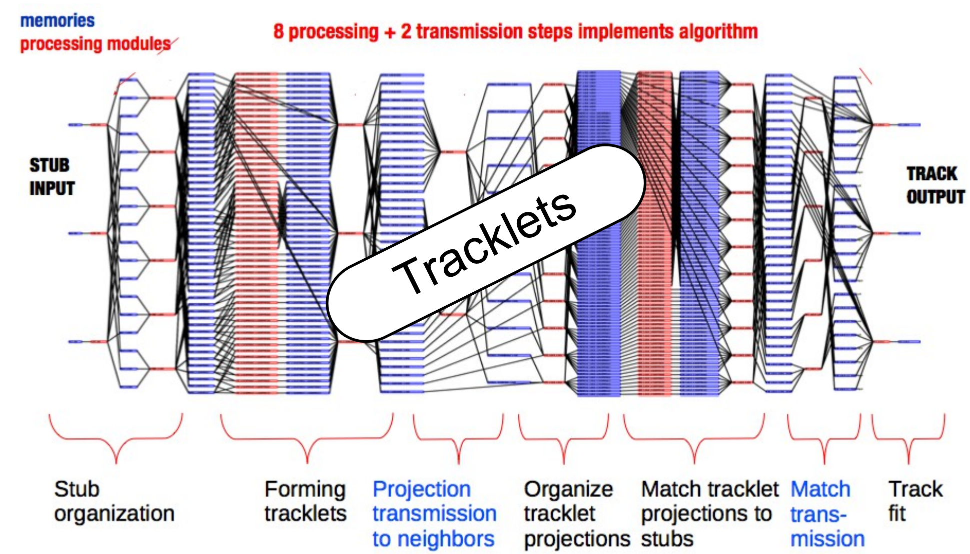


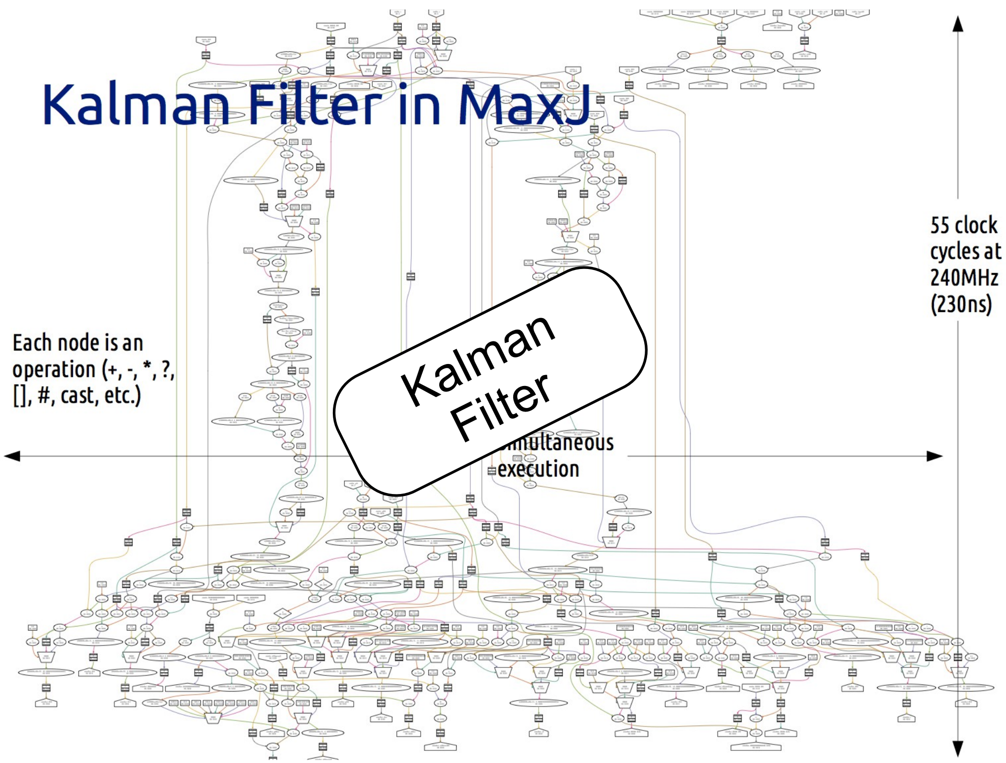


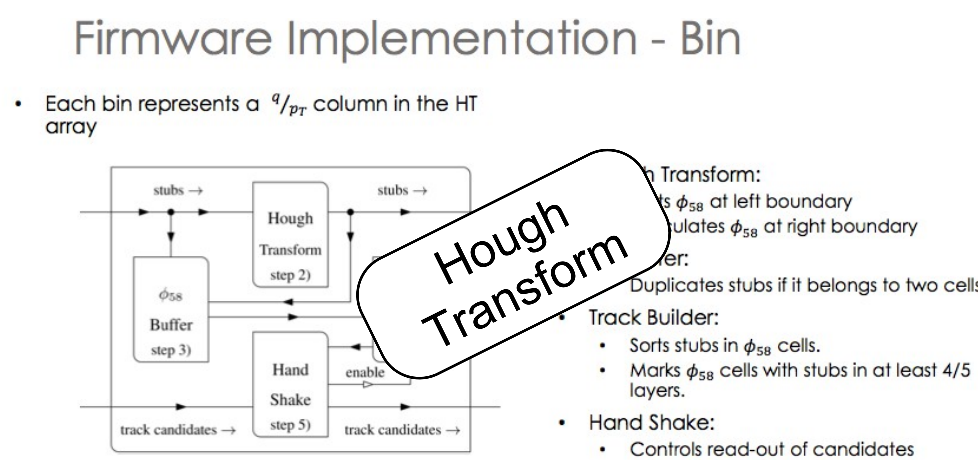
HEP.TrkX J.-R. Vlimant

Fast Hardware Tracking

- Track trigger implementation for Trigger upgrades development on-going
- Several approaches investigated
- **Dedicated hardware is the key** to fast computation.
- **Not applicable for offline** processing unless through adopting heterogeneous computing.





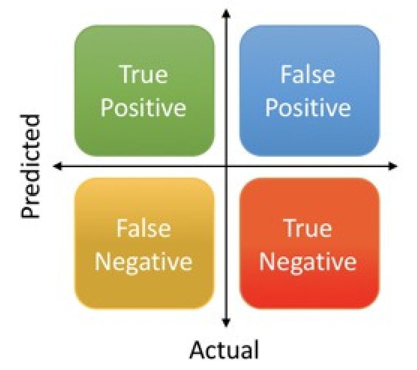


Recall & Precision

$$\textbf{Precision} = \frac{\text{True Positive}}{\text{Actual Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Positive}}$$

$$\textbf{Recall} = \frac{\text{True Positive}}{\text{Predicted Results}} \quad \text{or} \quad \frac{\text{True Positive}}{\text{True Positive + False Negative}}$$

$$\textbf{Accuracy} = \frac{\text{True Positive + True Negative}}{\text{Total}}$$



Precision ≡ Efficiency

Recall ≡ Purity ≡ 1-(Fake rate)

Accuracy ≡ How much do we get it right

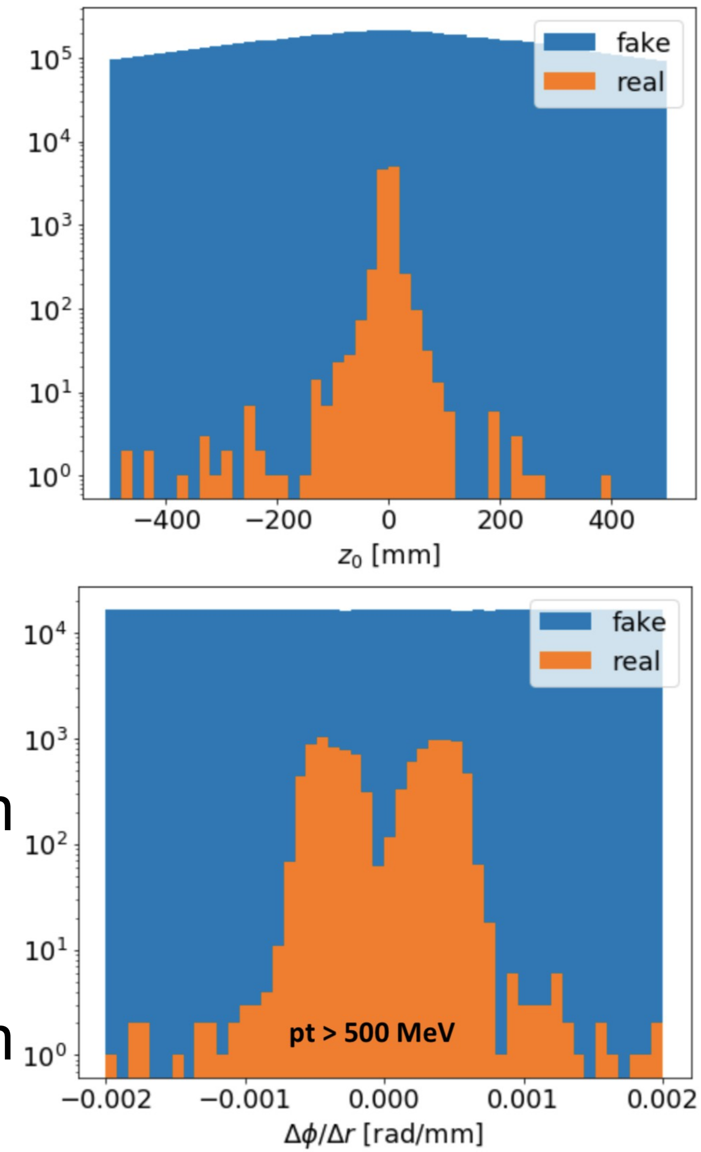
HEP.TrkX
J.-R. Vlimant

BERKELEY LAB

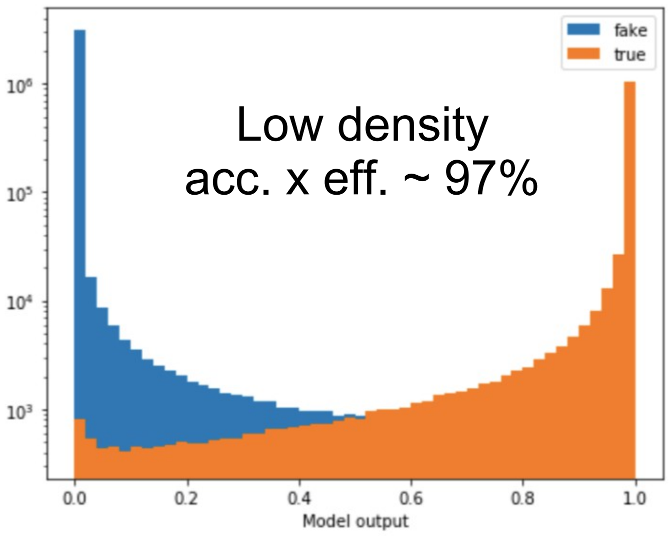
Downgraded Complexity

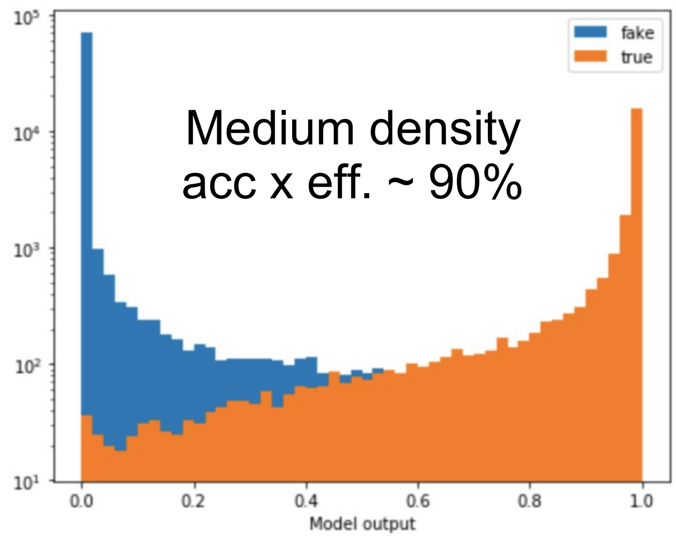
- TrackML dataset generated from ... with an average of 200 pileup events.
- Not computational possible at this time to embed the smallest relevant sector of full event on a graph

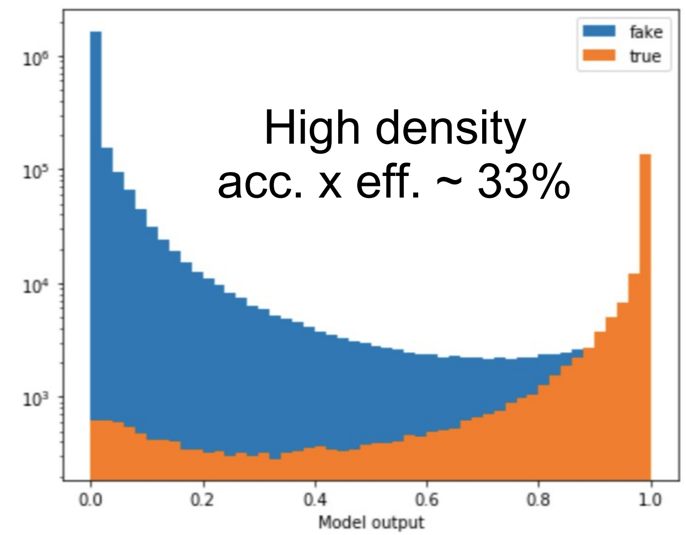
➔ Sub-dataset are constructed by
  - Low density
    - p_T>1 GeV, $\Delta\varphi$<0.001, Δz_0<200mm
    - acceptance: 99%, purity: 33%
  - Medium density
    - p_T>500 MeV, $\Delta\varphi$<0.0006, Δz_0<150mm
    - acceptance: 95%, purity: 25%
  - High density
    - p_T>100 MeV, $\Delta\varphi$<0.0006, Δz_0<100mm
    - ➔ acceptance: 43%, purity: 9%



Performance





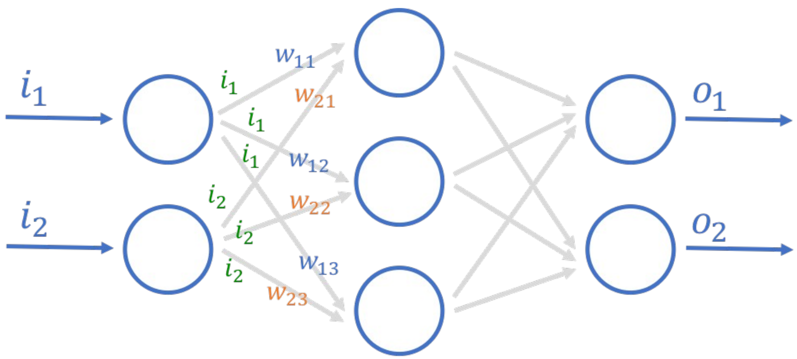


Summary

- Pilot project to explore new ideas for charged particle track reconstruction
- Graph neural network show promising results even in increasingly dense event
- Post-processing, pre-processing, using domain knowledge, ... : work in progress
- Optimizing such models requires training at scale : issues to be tackled, stay tuned

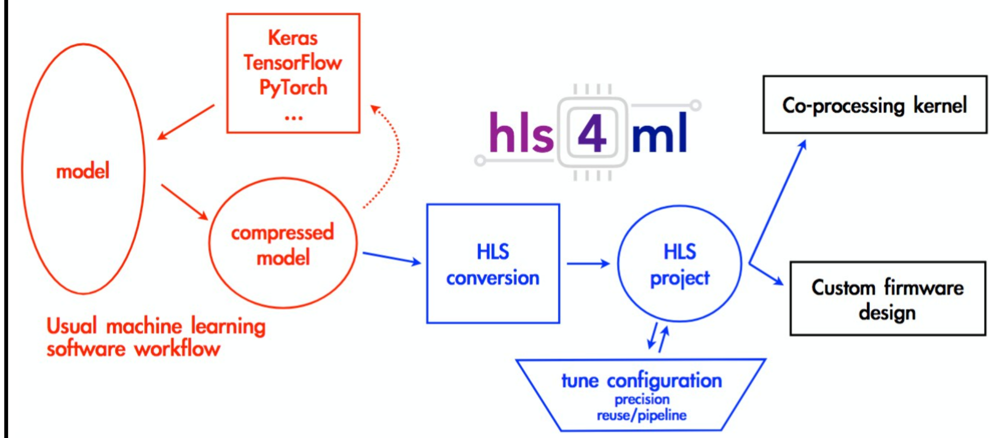
The Case for Machine Learning

Computational Aspect



ANN ≡ matrix operations ≡ parallelizable

$$\begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \\ w_{13} & w_{23} \end{bmatrix} \cdot \begin{bmatrix} i_1 \\ i_2 \end{bmatrix} = \begin{bmatrix} (w_{11} \times i_1) + (w_{21} \times i_2) \\ (w_{12} \times i_1) + (w_{22} \times i_2) \\ (w_{13} \times i_1) + (w_{23} \times i_2) \end{bmatrix}$$



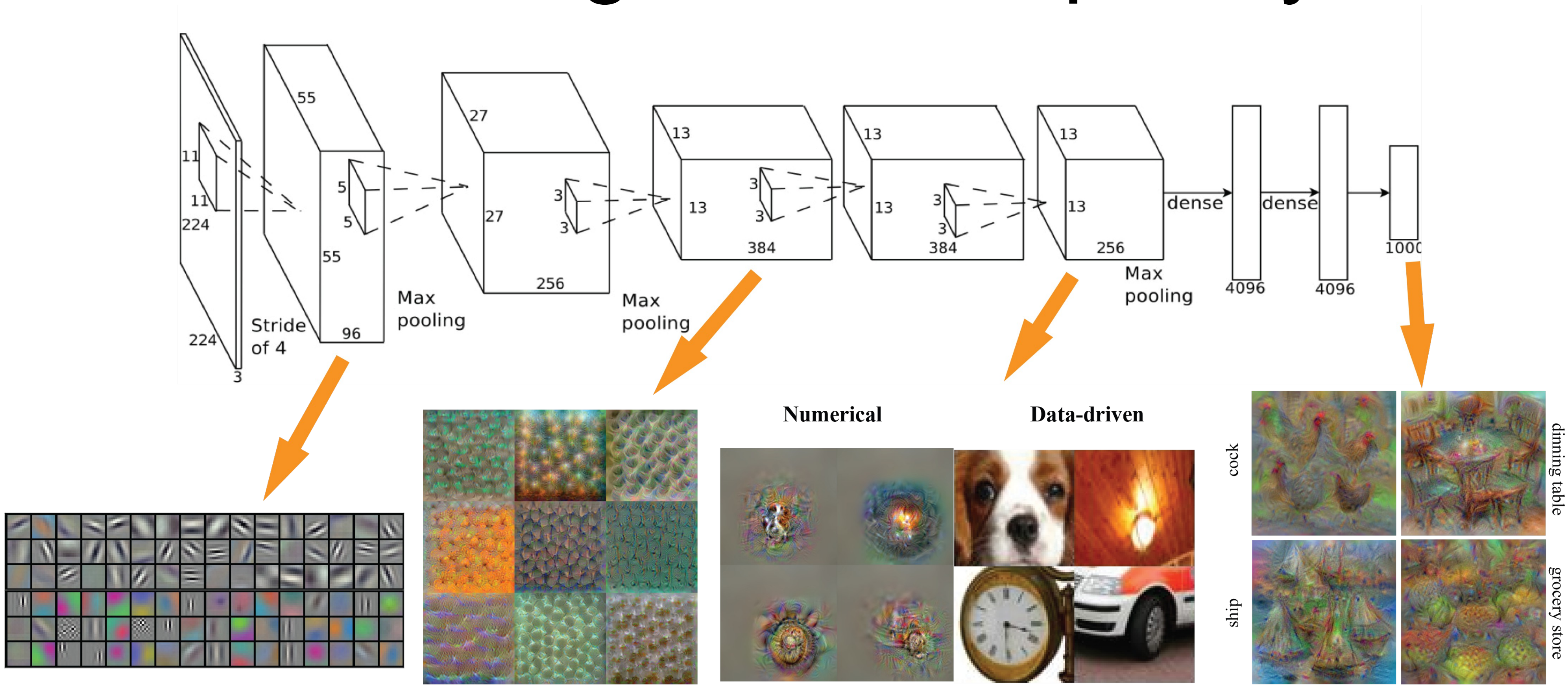
Synthesis FPGA firmware
from trained ANN

https://hls-fpga-machine-learning.github.io/hls4ml/

Bonsai BDT, contained tree growth and feature discretization ; fast classification
https://arxiv.org/abs/1210.6861

- Computation for machine learning prediction from a trained model is parallel and can be fast

Learning from Complexity



Conv 1: Edge+Blob **Conv 3: Texture** **Conv 5: Object Parts** **Fc8: Object Classes**

- Machine learning can extract useful information from complex underlying data structure
- Classical algorithm counter part may take years of development

HEP.TrkX
J.-R. Vlimant
BERKELEY LAB

Pattern Recognition With Deep Learning

Machine Learning for Tracking

Zagoruyko et al, [1604.02135](1604.02135)

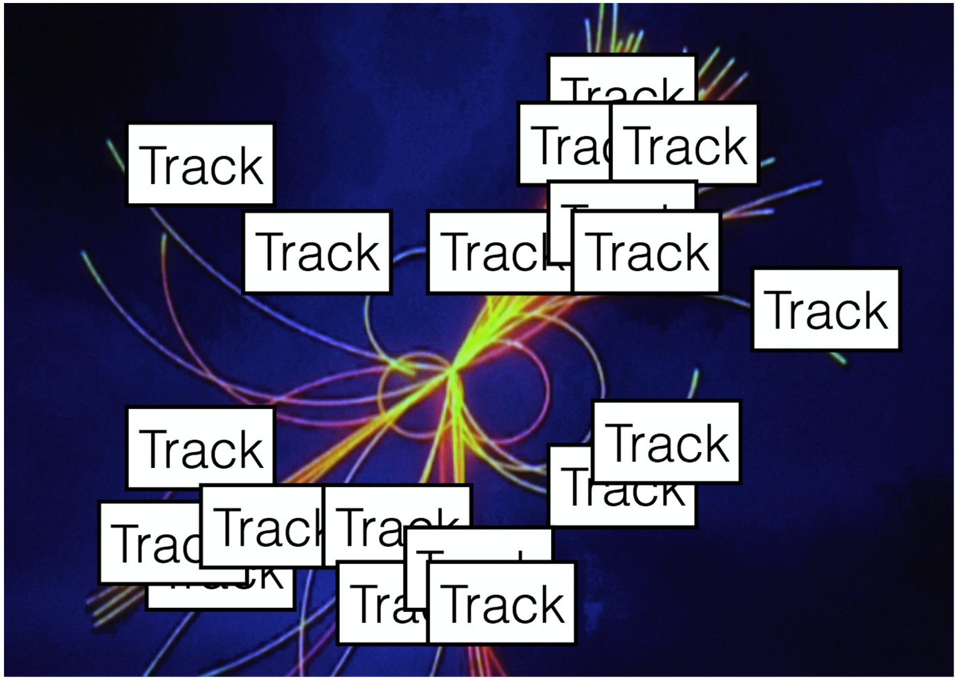


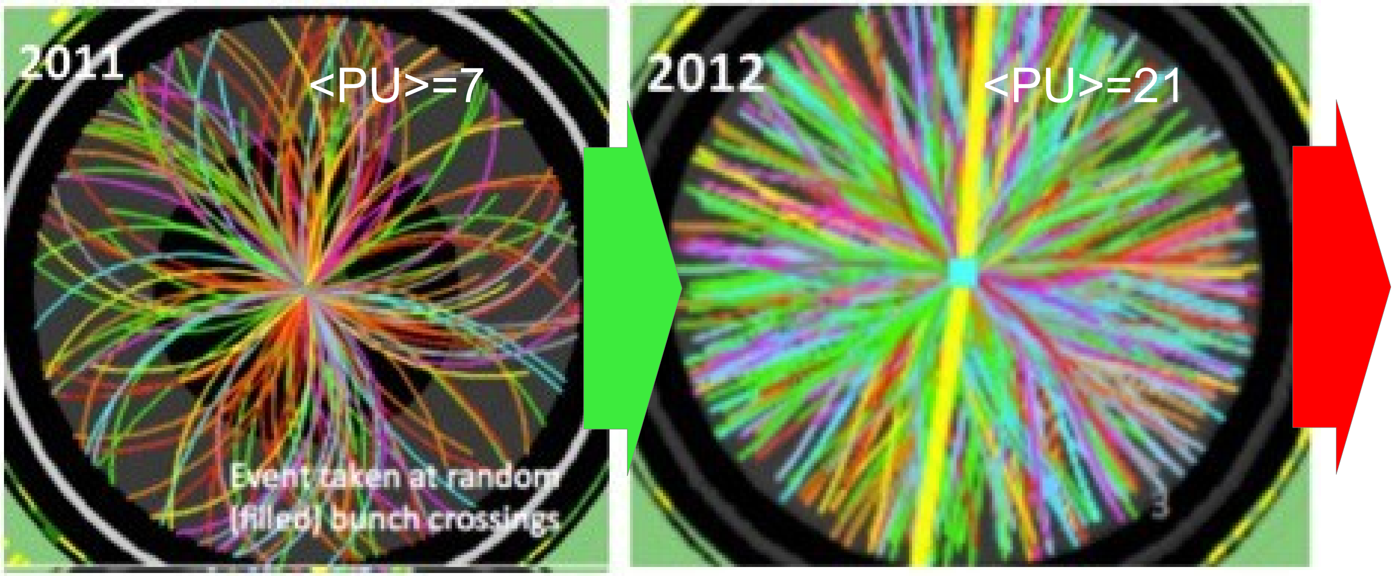
Photo by Pier Marco Tacca/Getty Images

Many possible ways to cast the algorithm of tracking, or part of the current algorithms in a machine learning problem

Similarities and Challenges

- Particle tracking is an active field in data science
  - Different type of particles
  - Not oriented to code performance
- Making a track is a pattern recognition problem
  - Not the usual one in data science
- Tracking data is much sparser than regular images
  - Test and adapt methods
- Tracking device may have up to 10M of channels
  - Scale up deep learning models
  - Perform tracking by sector
- Underlying geometry of sensor more complex
  - More than a simple picture
  - Barrels and end-caps are not the usual pictures
- Not the regular type of sequences
  - Cover new ground of sequence processing
- Defining an adequate cost function
  - Tracking algorithms are optimized by proxy
- A solution must be performant during inference ...

HL-LHC Challenge



<PU>=140-200
10x more hits
Circa 2025

- CPU time extrapolation into HL-LHC era far **surpasses growth in computing budget**
- **Need for faster algorithms**
- Approximation allowed in the trigger

Scene Labeling

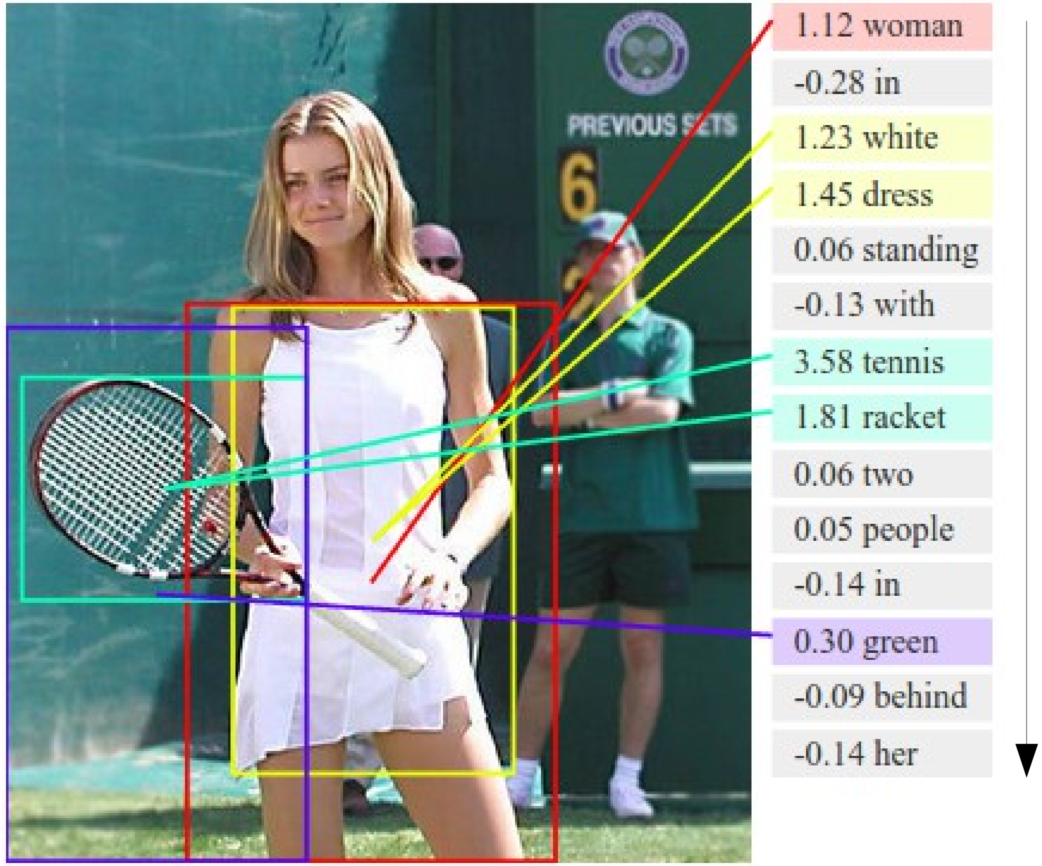
From talk of LeCunn at CERN

Scene Labeling

Farabet et al. ICML 2012, PAMI 2013

- Assign hits to track candidates

Scene Captioning

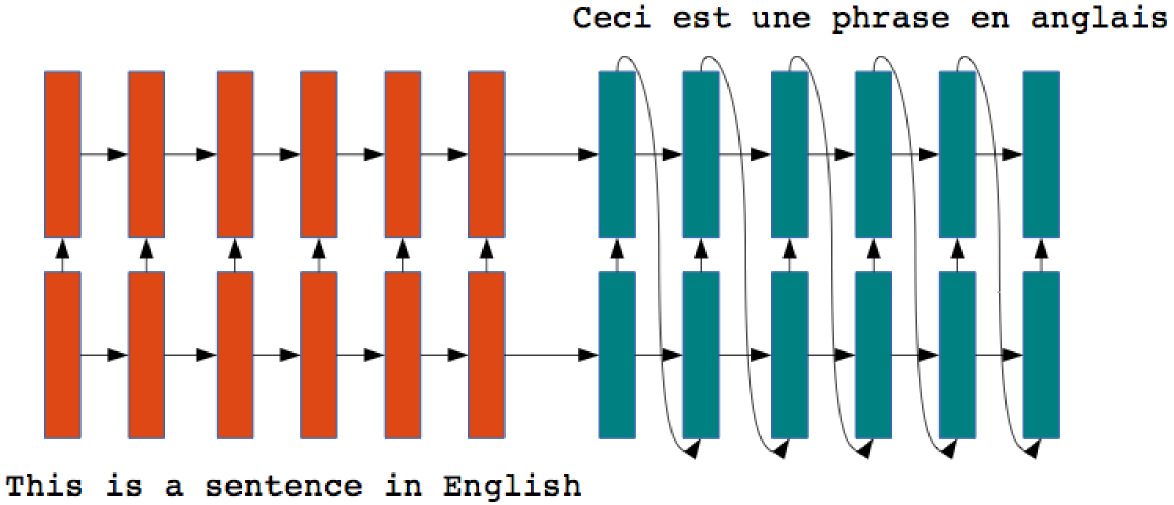


Karpathy, Fei-Fei, CVPR 2015

➔ Compose tracks explanation from image

Text Translation

- [Sutskever et al. NIPS 2014]
  - Multiple layers of very large LSTM recurrent modules
  - English sentence is read in and encoded
  - French sentence is produced after the end of the English sentence
  - Accuracy is very close to state of the art.

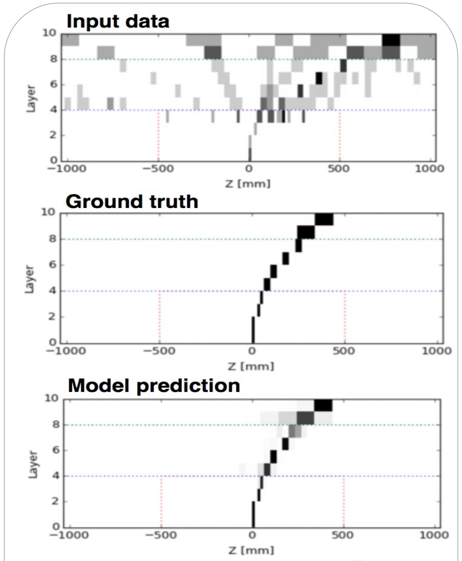


→ From sequence of hits on layer to sequence of hits on track

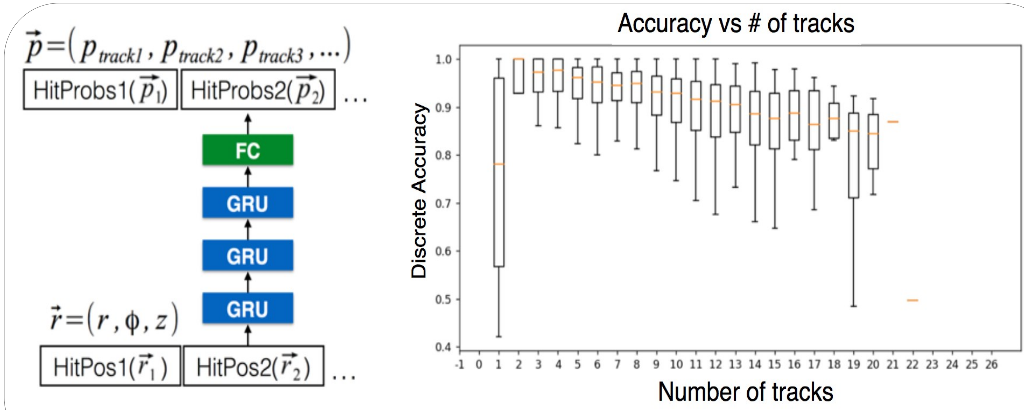
Possible Application to Tracking

- **Track candidate**
  - Finding the hits that belong to a track
  - Seed + hits → tracks
- **Track parameters**
  - Measuring the physic quantity of tracks
  - Hits → track kinematics
- **Seeding**
  - Putting together hits into tracks
  - Hits → track

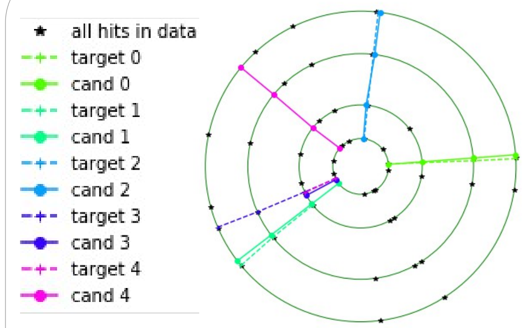
HEP.TrkX Approaches



Track following with RNN



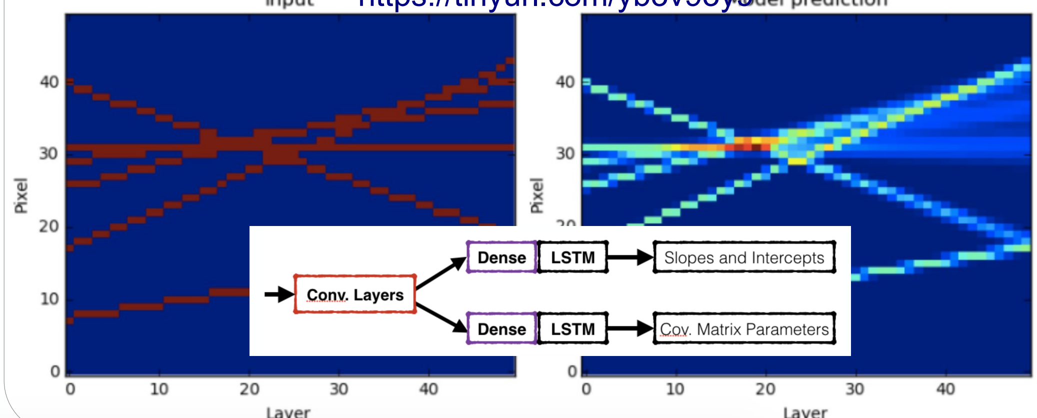
End-to-end hit assignment

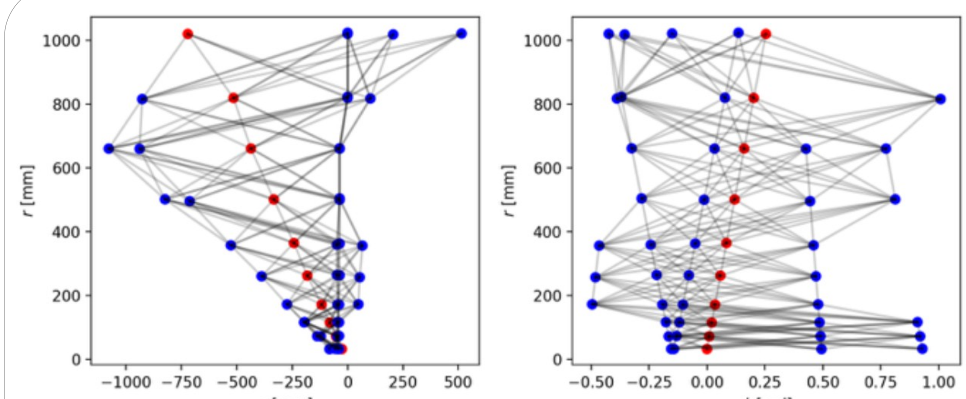


Seq-to-seq track finding

https://heptrkx.github.io/

https://tinyurl.com/yb3v93y9





https://tinyurl.com/y87saehf

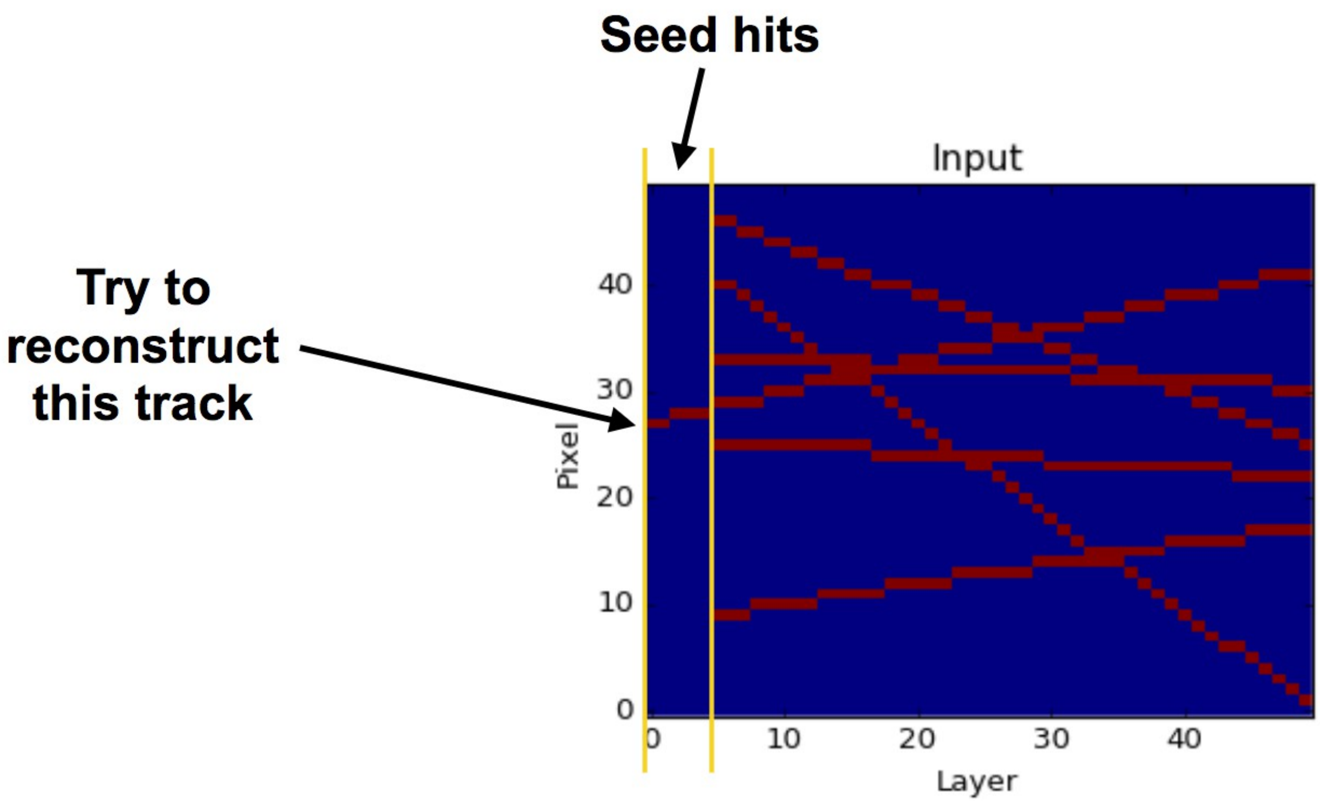
HEP.TrkX
J.-R. Vlimant

BERKELEY LAB

Seeded Track Candidate Making

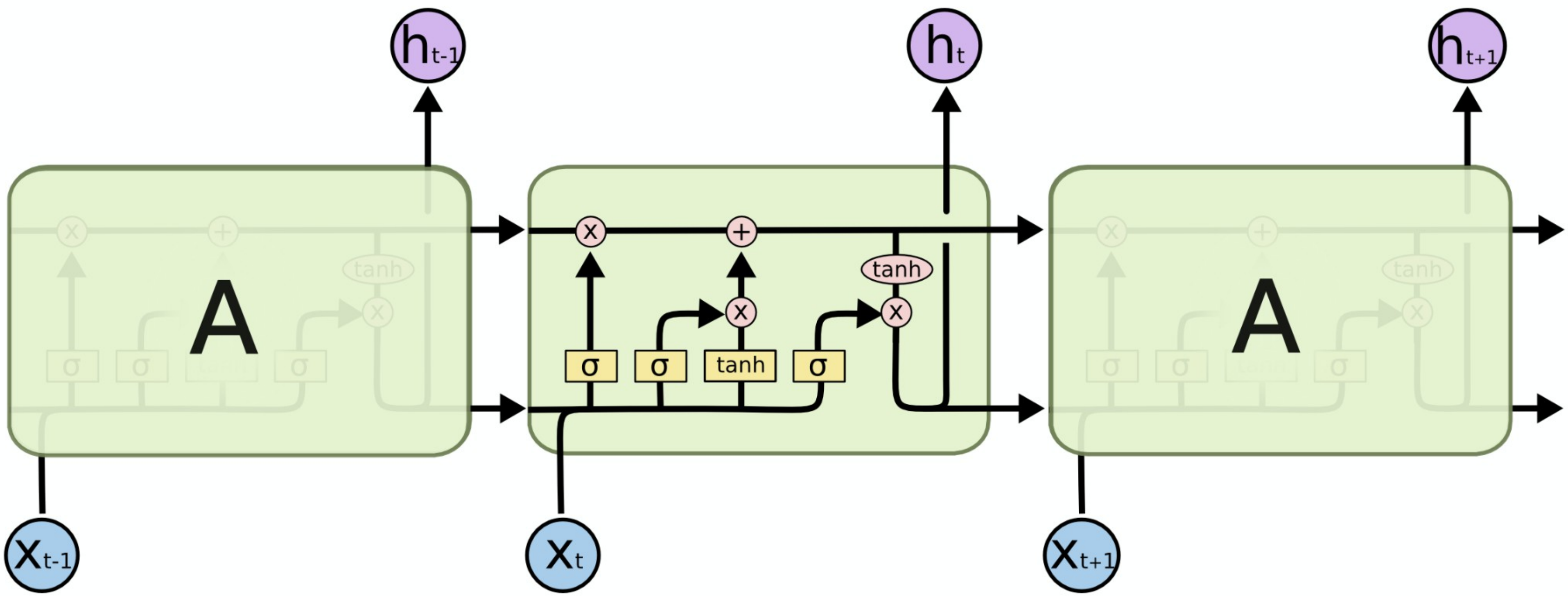
Seeded Pattern Prediction

- Hits on first 3 layers are used as seed
- Predict the position of the rest of the hits on all layers



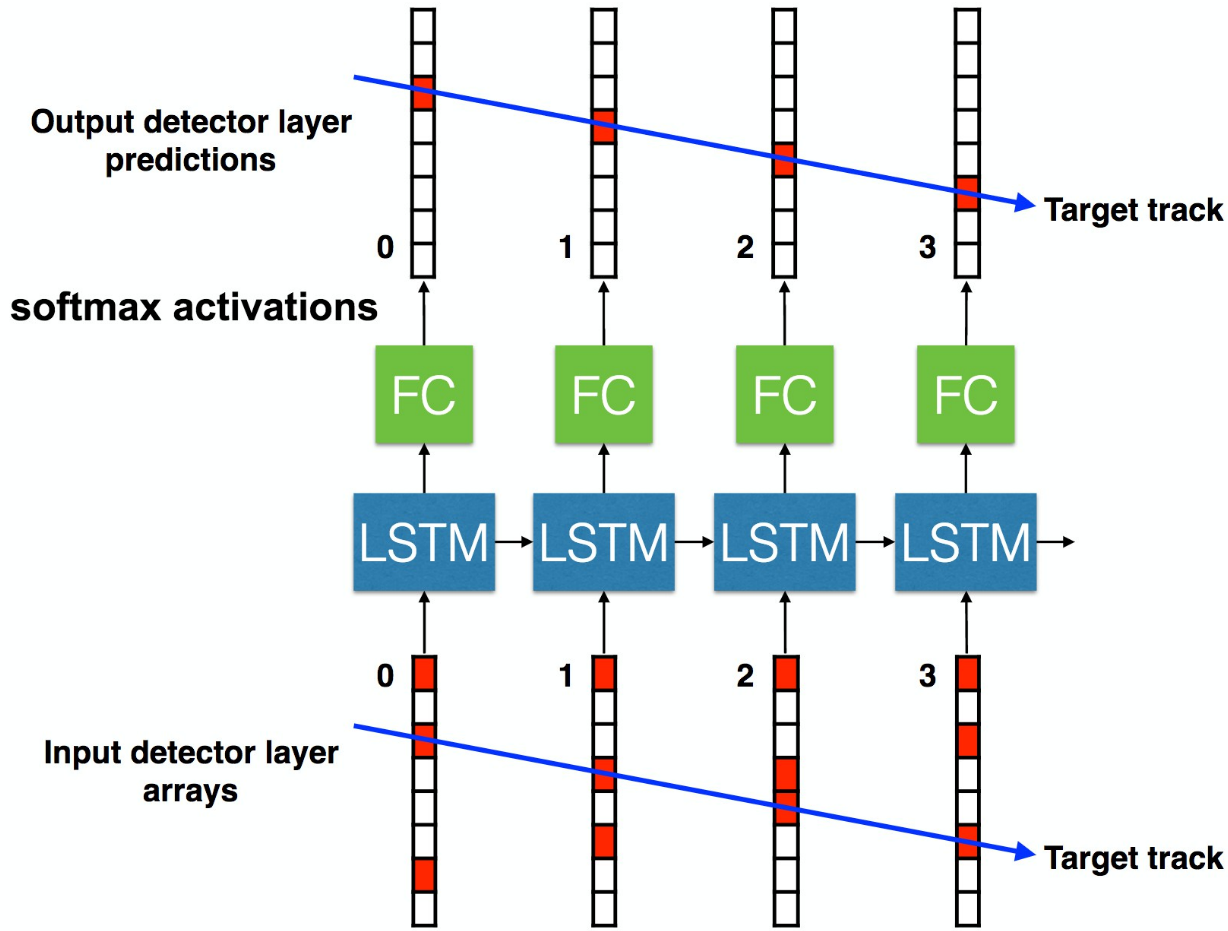
Long Short Term Memory - LSTM

Breakthrough in sequence processing by carrying over an internal state, “memory” of the previous items in the sequence, allowing for long range correlation



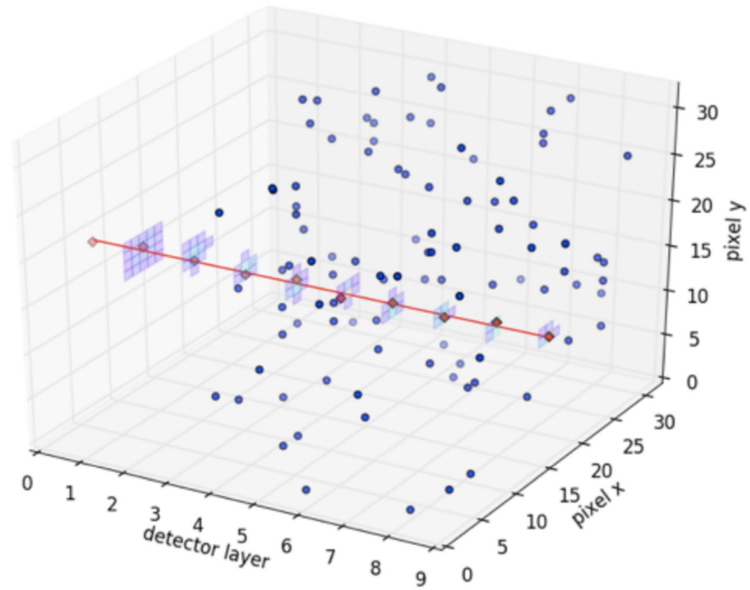
http://colah.github.io/posts/2015-08-Understanding-LSTMs/

LSTM ≡ Kalman Filter

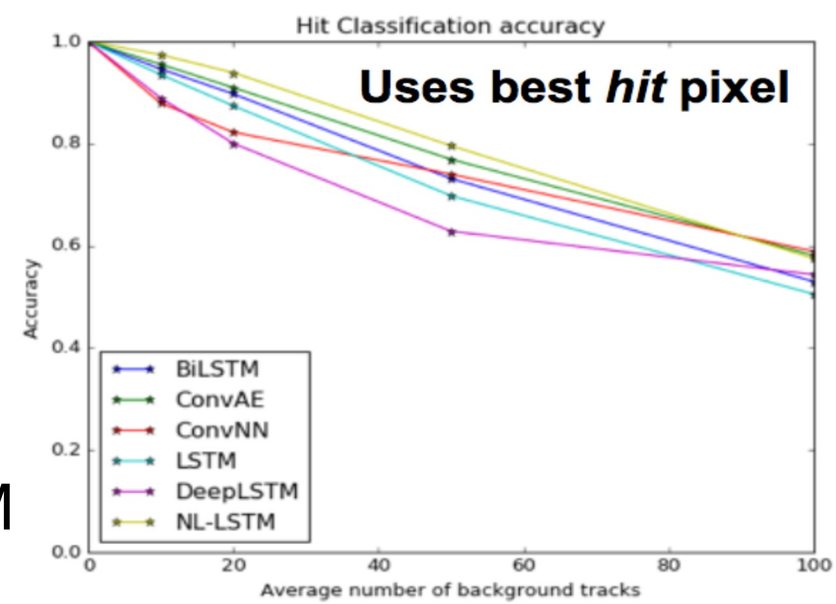


Seeded Pattern Recognition Insights

- For a simplified track models, predicting the track pattern from the seed works
  - In 2D and 3D
  - With some level of noise
  - With other tracks present
  - On layers with increasing number of pixels

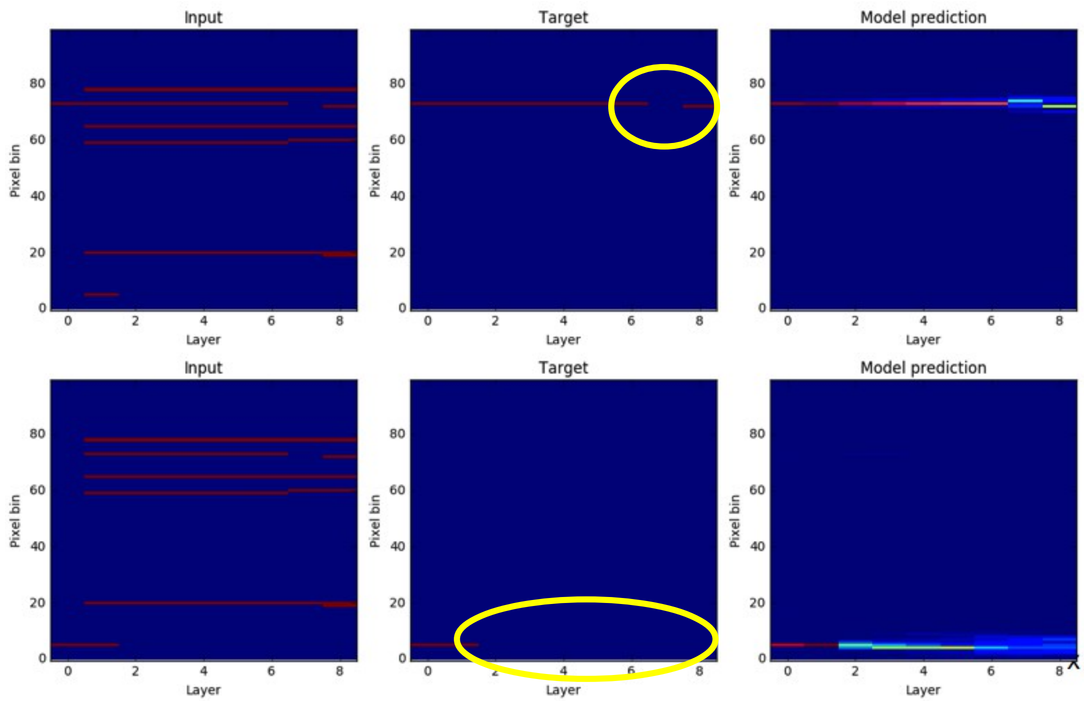


- Several other architectures tried
  - Convolutional neural nets (no LSTM)
  - Convolutional auto-encoder
  - Bi-directional LSTM
  - Prediction on next layer with LSTM



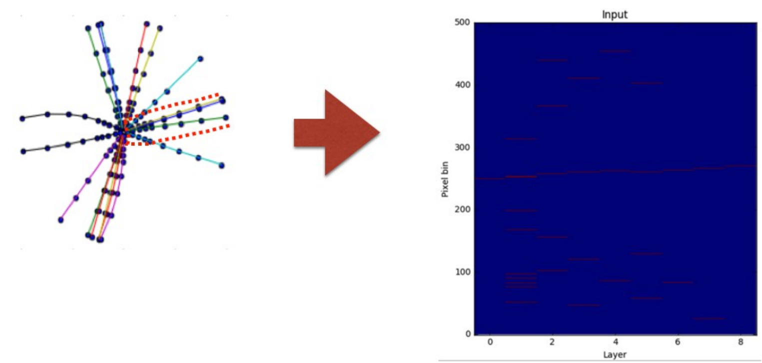
Tracking RAMP at CtD

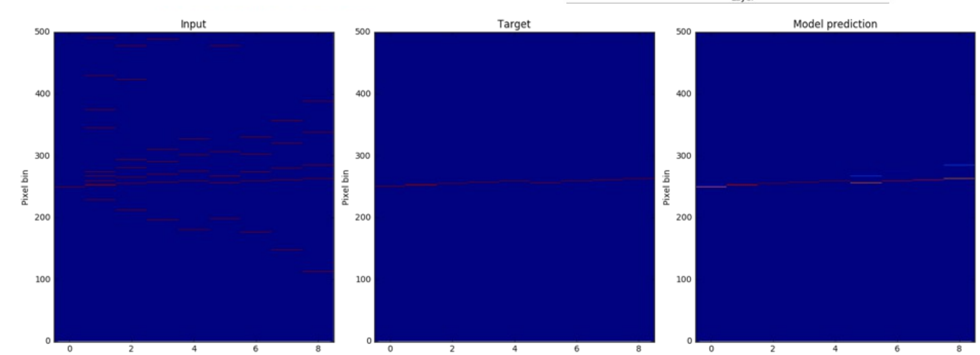
S. Farrell : Best solution in the Machine Learning category
https://indico.cern.ch/event/577003/contributions/2509988/



- Down-sampling layer to 100 bins
- LSTM for hit assignment
- 92% efficiency
- Robust to holes and missing hits

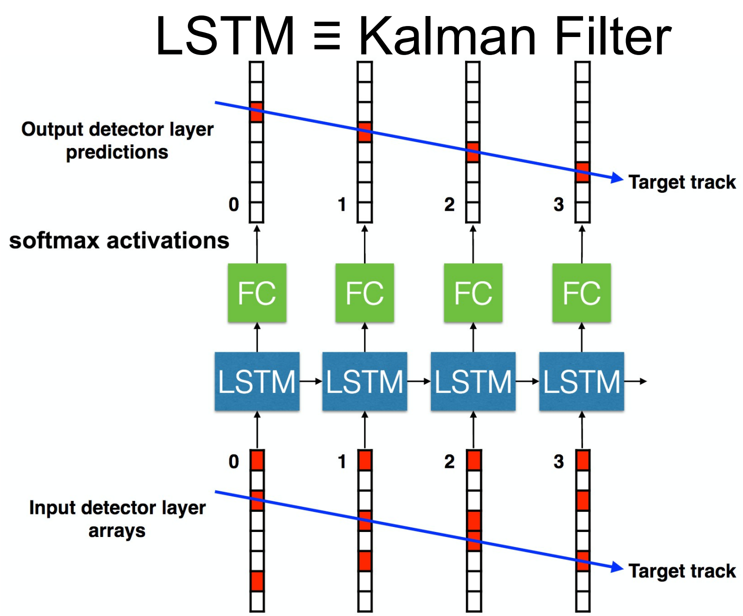
- Increased granularity in “road”
- LSTM for hit assignment
- 95% efficiency

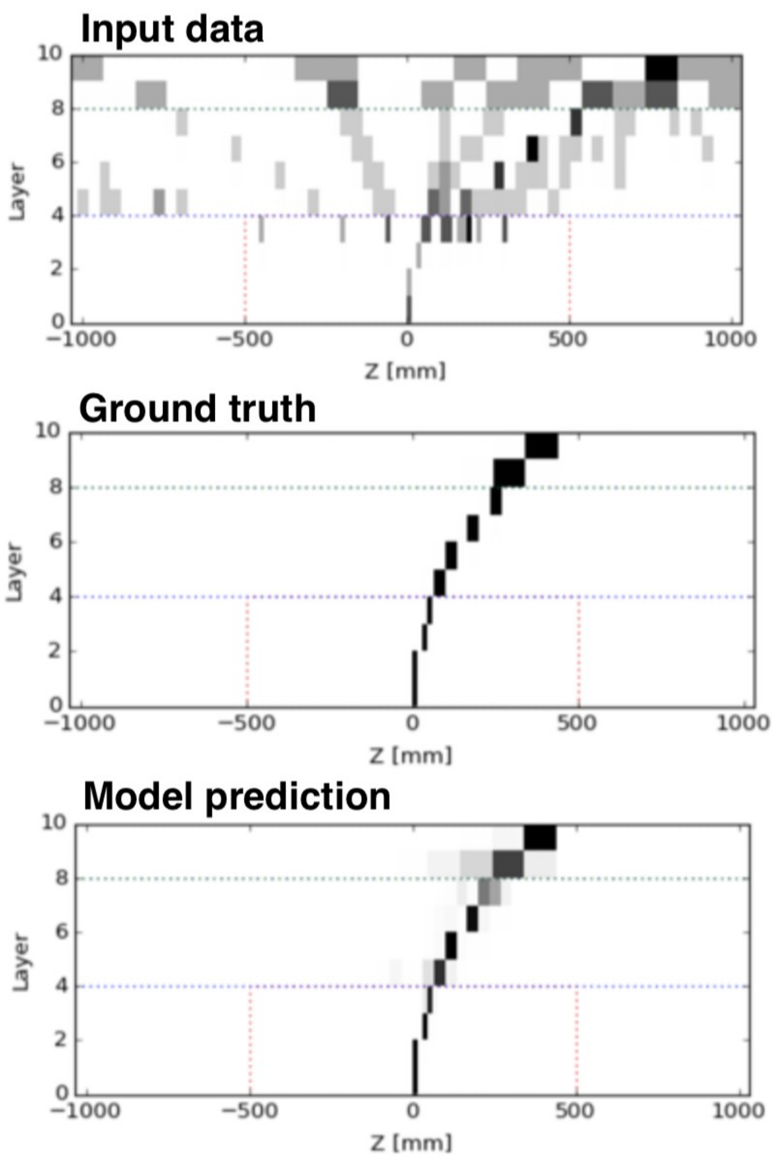




HEP.TrkX
J.-R. Vlimant
BERKELEY LAB

Finding Tracks with LSTM





- Search seeded from a known tracklet
- Hit location is discretized to fixed length
- Model predicts the binned position of the hit on the next layer

HEP.TrkX
J.-R. Vlimant
BERKELEY LAB

Hit Prediction with Gaussian Model

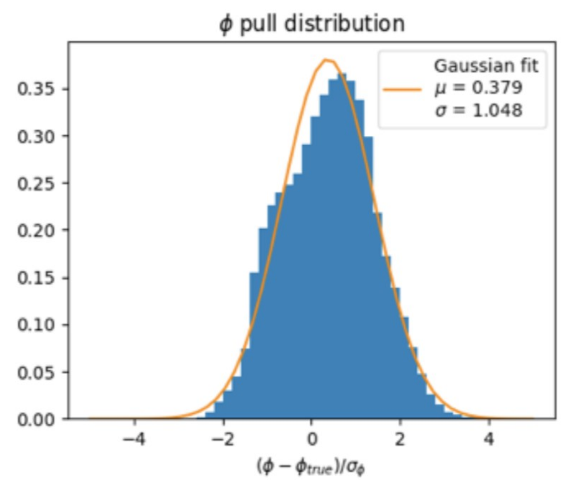
$$\boxed{\vec{r}_0, \vec{r}_1, \ldots, \vec{r}_{N-1}} \rightarrow \boxed{\text{LSTM}} \rightarrow \boxed{\text{FC}} \rightarrow \boxed{(\hat{\vec{r}}_1, \Sigma_1), (\hat{\vec{r}}_2, \Sigma_2), \ldots, (\hat{\vec{r}}_N, \Sigma_N),}$$

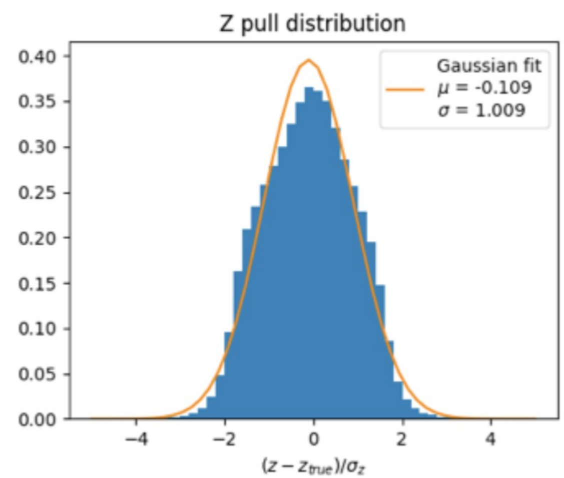
$$\vec{r} = (r, \phi, z) \qquad \hat{\vec{r}} = (\hat{\phi}, \hat{z}) \quad \Sigma = \begin{pmatrix} \sigma_\phi^2 & \sigma_{\phi z}^2 \\ \sigma_{\phi z}^2 & \sigma_z^2 \end{pmatrix}$$

Loss function incorporates the position and the predicted uncertainty

$$L(x, y) = \log|\Sigma| + (y - f(x))^{\mathrm{T}} \Sigma^{-1} (y - f(x))$$

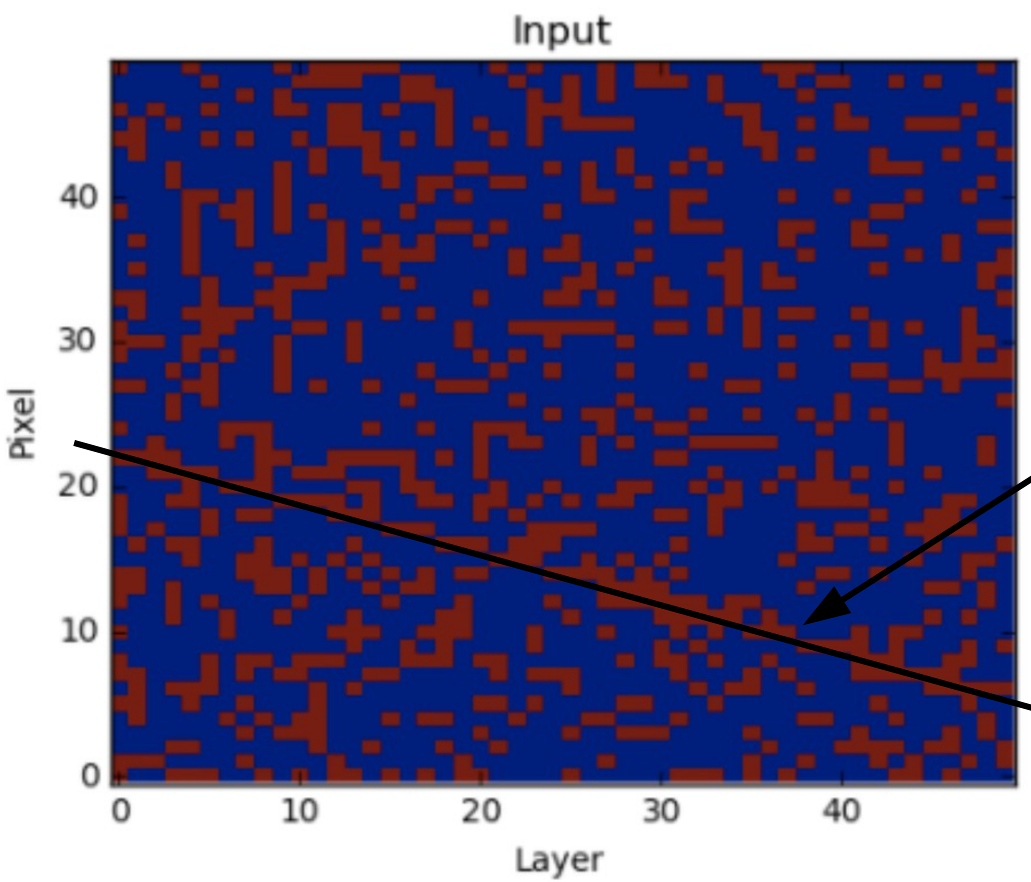
- Search seeded from a known tracklet
- Hit positions taken in sequential input
- Model predicts the position of the hit on the next layer





Track Parameters Measurement

Track Parameter Estimation

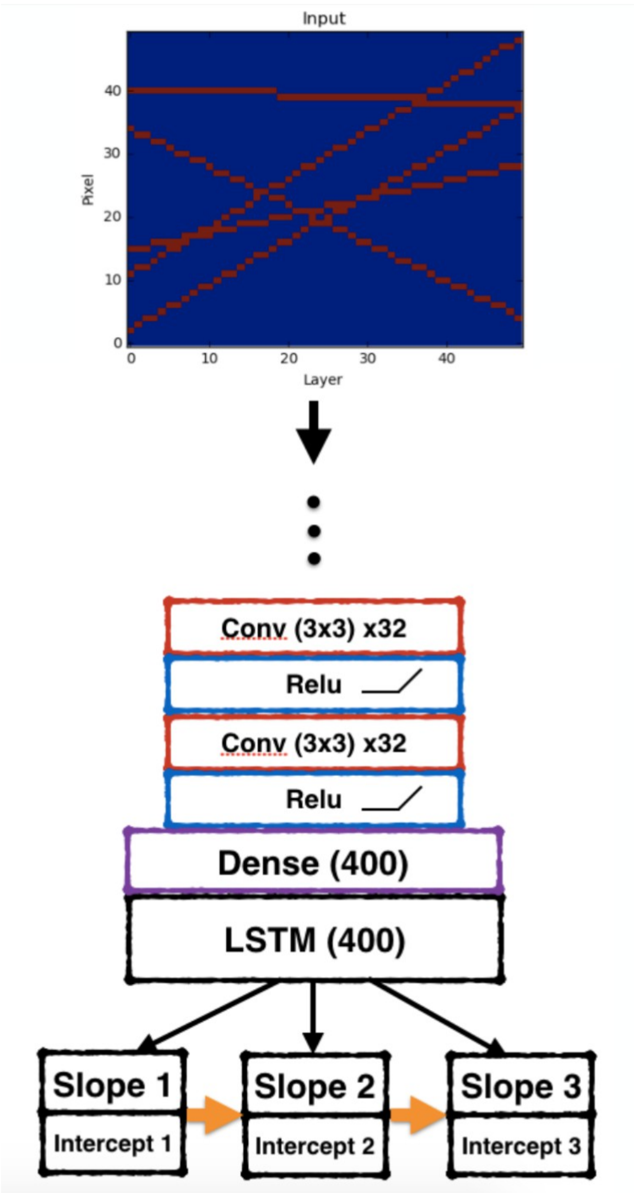


Try to predict the slope and intersect of this track

Multi-Track Prediction with LSTM

- Hit pattern from multiple track processed through convolutional layers

- LSTM Cell runs for as many tracks the model can predict.



Predicting Covariance Matrix

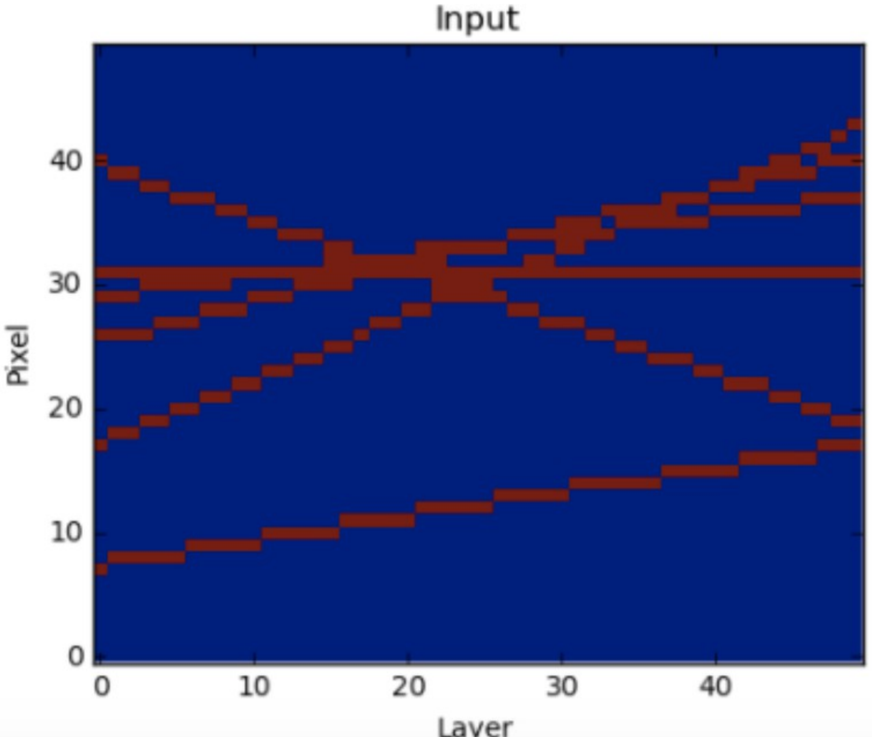
- The observed hit pattern from multiple track processed through convolutional layers
- LSTM cells are ran multiple time in order to predict a list of particles
- Model is able to predict the covariance matrix of track parameters, incorporated in the loss function

$$L(\boldsymbol{x},\boldsymbol{y}) = \log|\boldsymbol{\Sigma}| + (\boldsymbol{y}-\boldsymbol{f}(\boldsymbol{x}))^T\,\boldsymbol{\Sigma}^{-1}\,(\boldsymbol{y}-\boldsymbol{f}(\boldsymbol{x}))$$

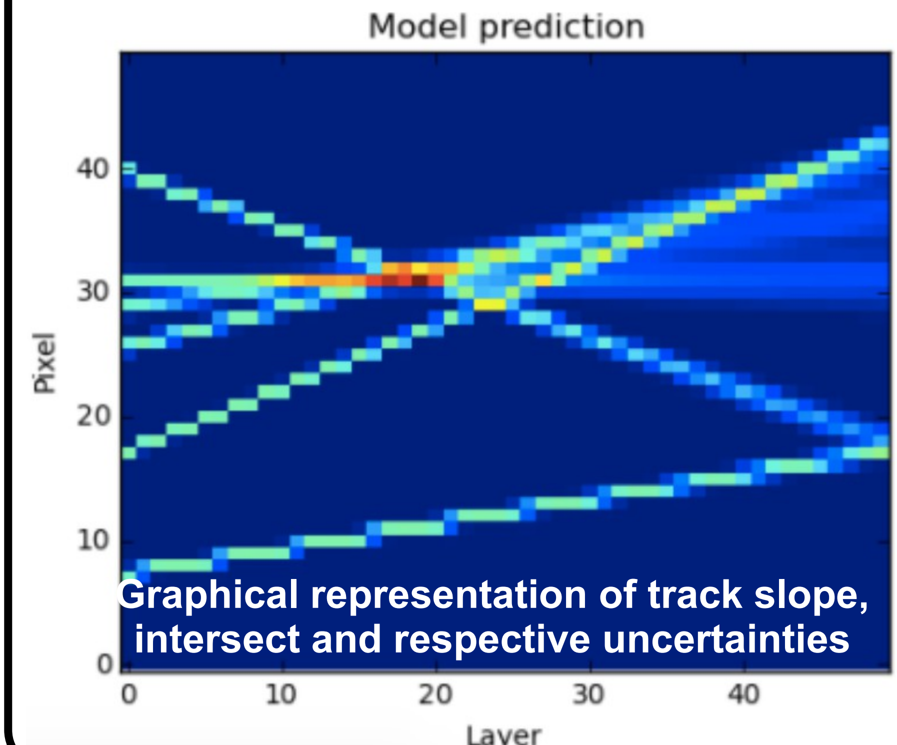
HEP.TrkX
J.-R. Vlimant

Track Parameter Prediction

Hit pattern in the detector

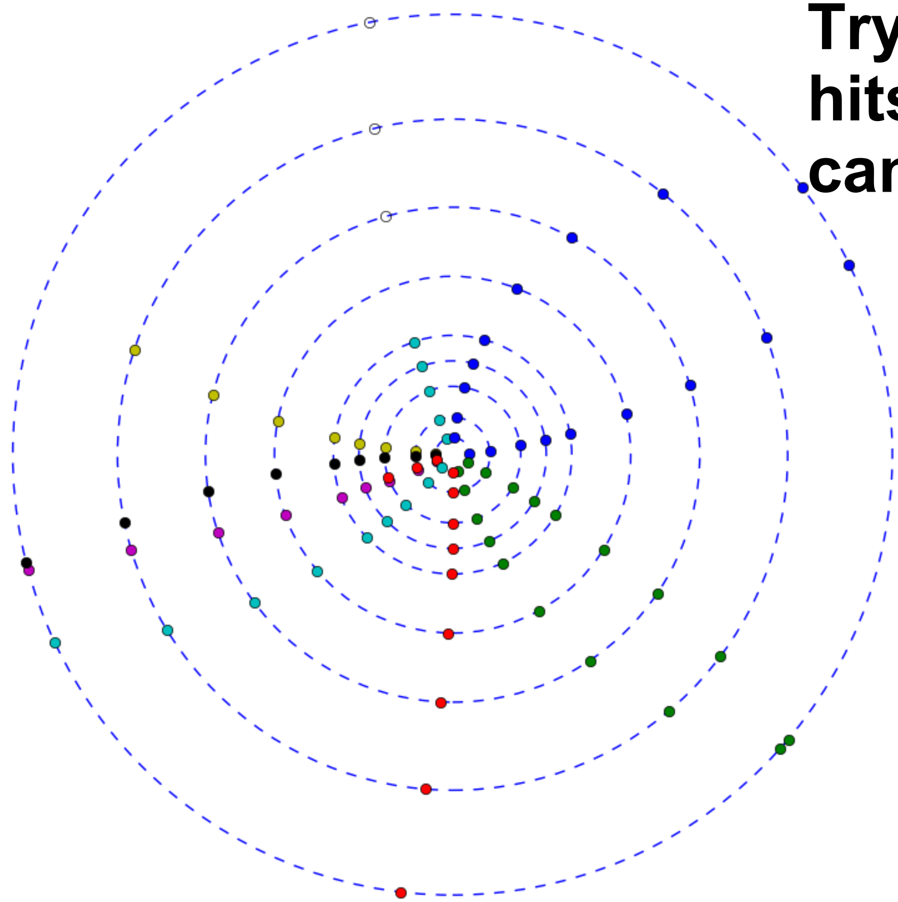


Track parameters and corresponding uncertainties



Hit Assignment Approaches

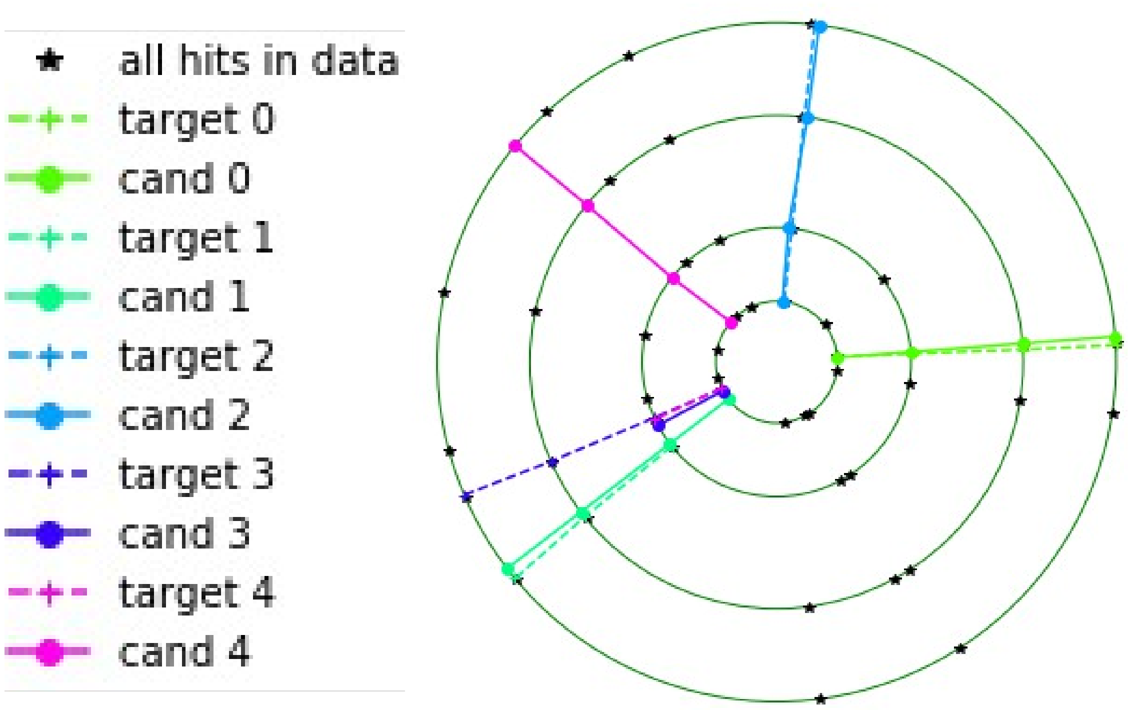
Pattern Recognition



Try to assemble hits into track candidates.

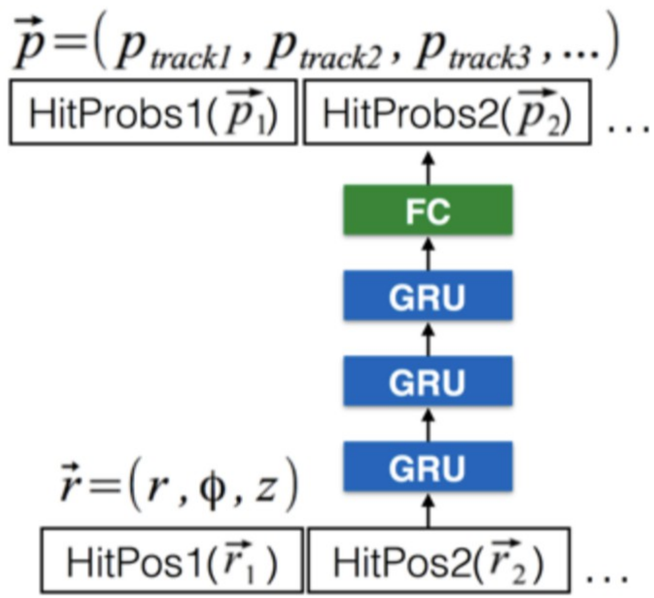
seq-2-seq tracking

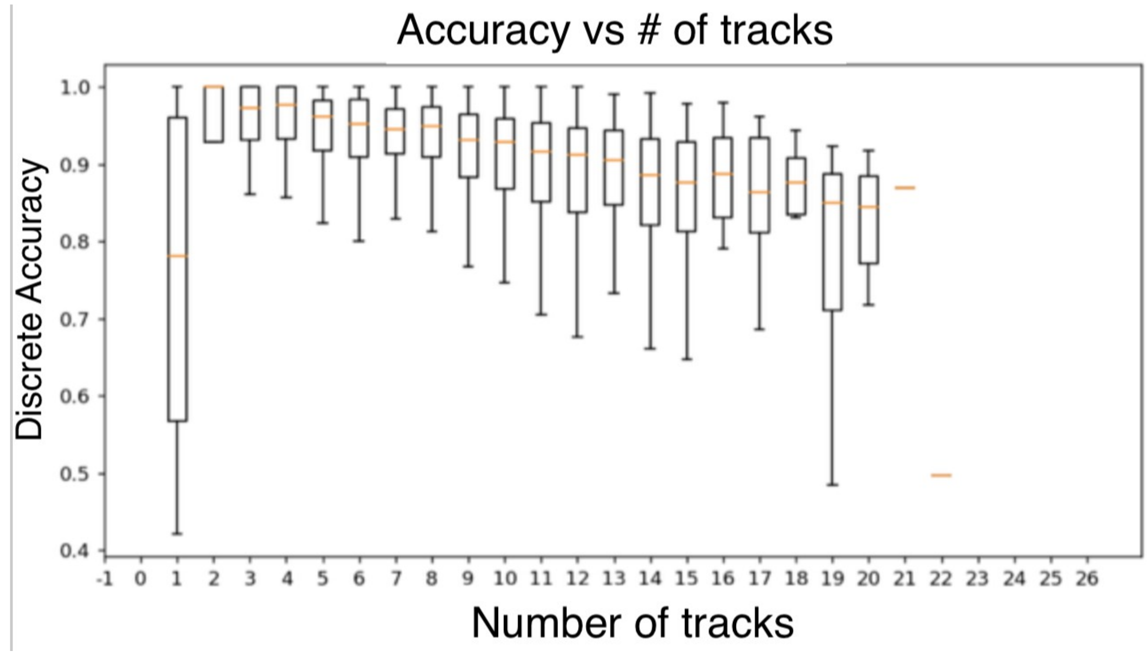
- Input sequence of hits per layers (one sequence per layer)
  - One LSTM cell per layer
- Output sequence of hits per candidates
  - Final LSTM runs for as many candidates the model can predict



- Restricted to 4 layers (with seeding in mind)
- Full performance evaluation still to be done

Hit Assignment Algorithm

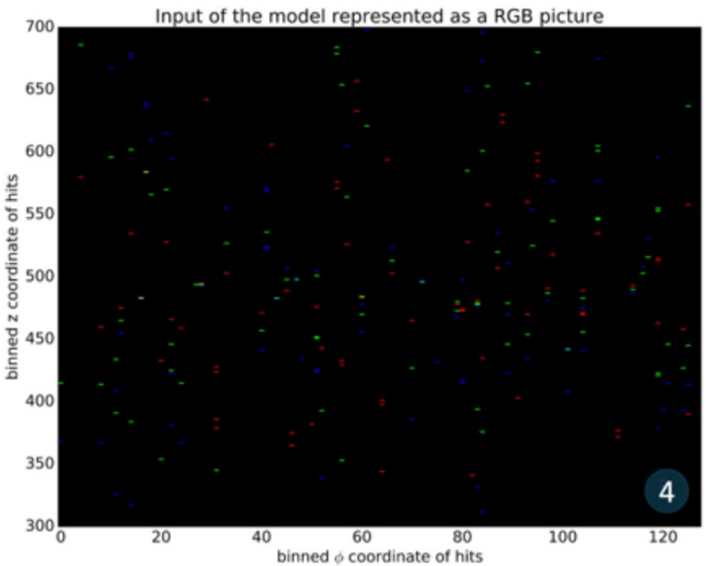


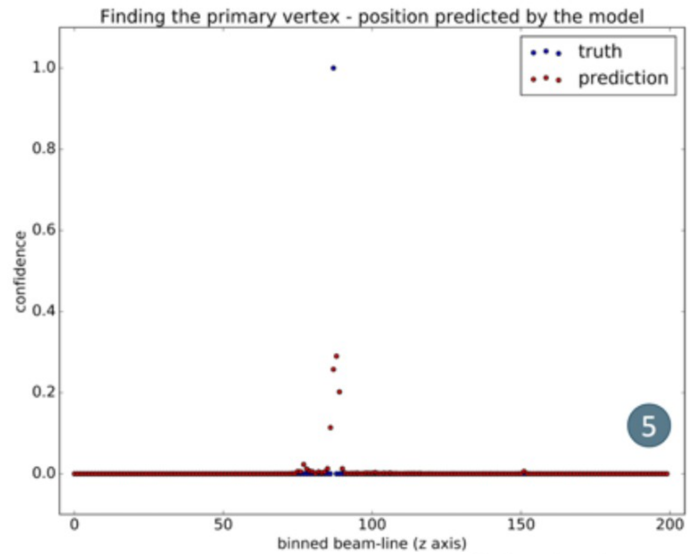


- Unseeded hit-to-track assignment (clustering)
- Hit positions taken in sequential input
- Model predicts the probability that a hit belongs to a track candidate

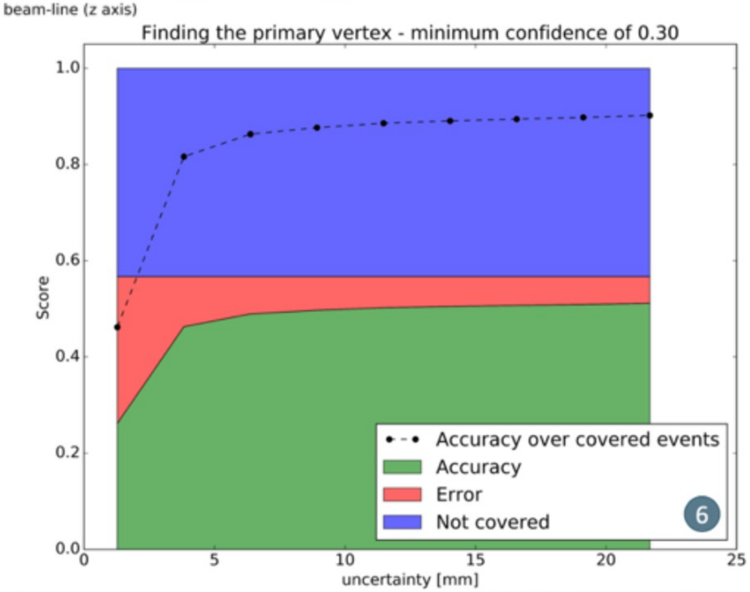
Vertexing

Vertexing with CNN





- Using hits binned (η, φ) map in input for a regression of the primary vertex position
- Modest success

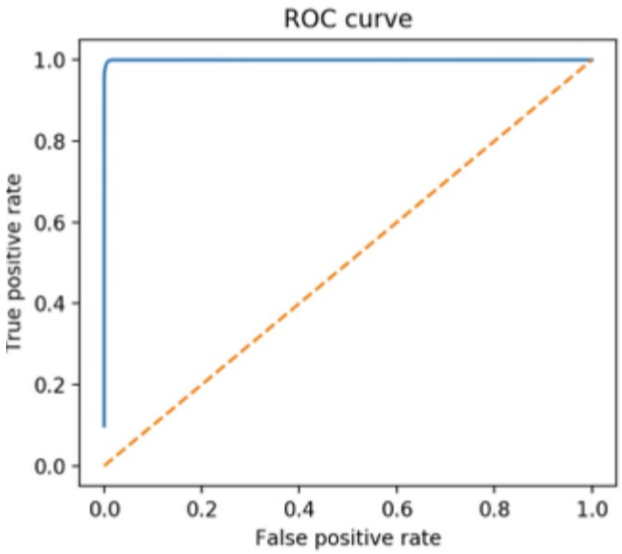


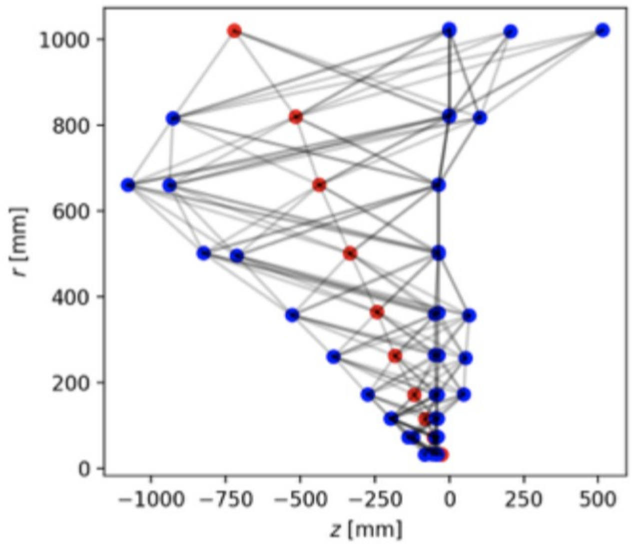
Graph Networks Approach

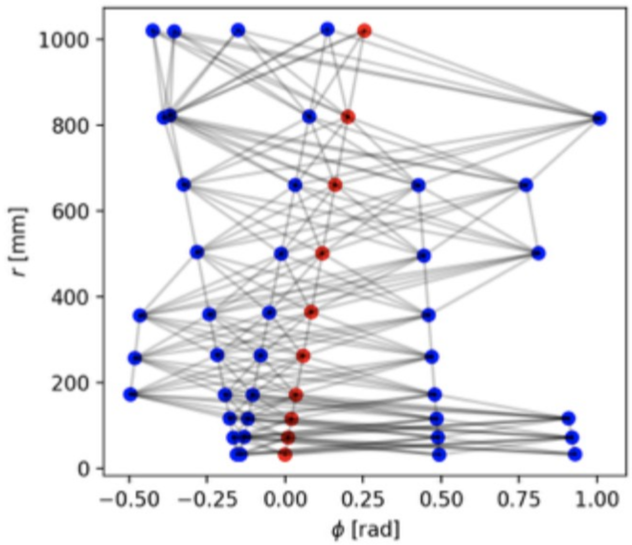
J.-R. Vlimant

Seeded Hit Classification with GNN

- Seeded hit classification
- Model predicts whether hits belong to the given seed

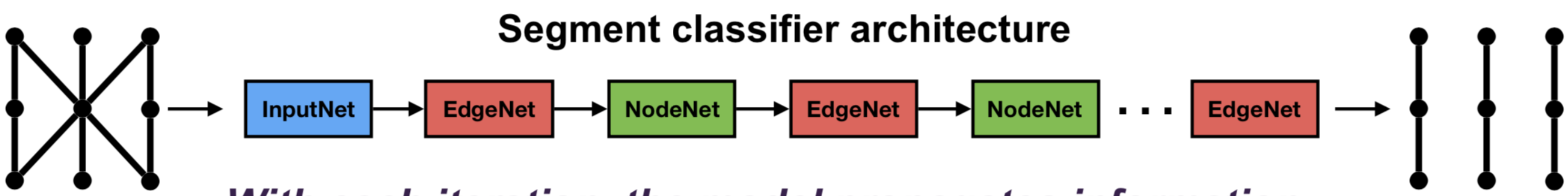






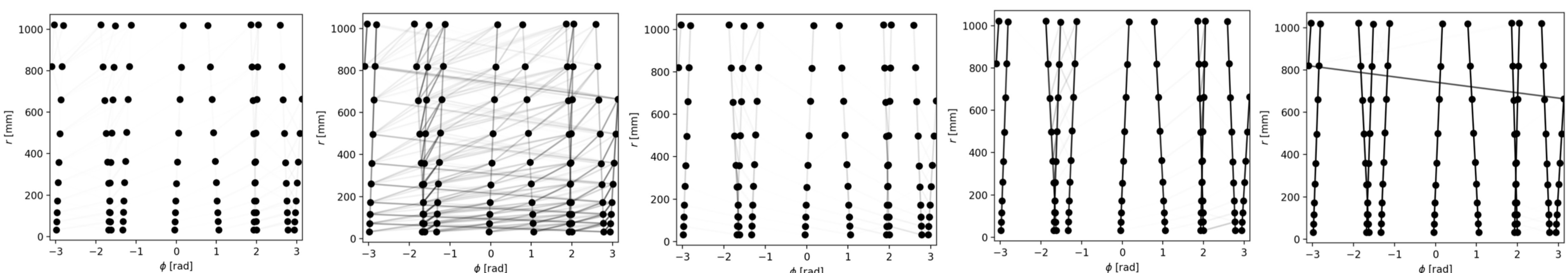
HEP.TrkX
J.-R. Vlimant

Track Building With GNN



With each iteration, the model propagates information through the graph, strengthens important connections, and weakens useless ones.

- Unseeded hit-pair classification
- Model predicts the probability that a hit-pair is valid



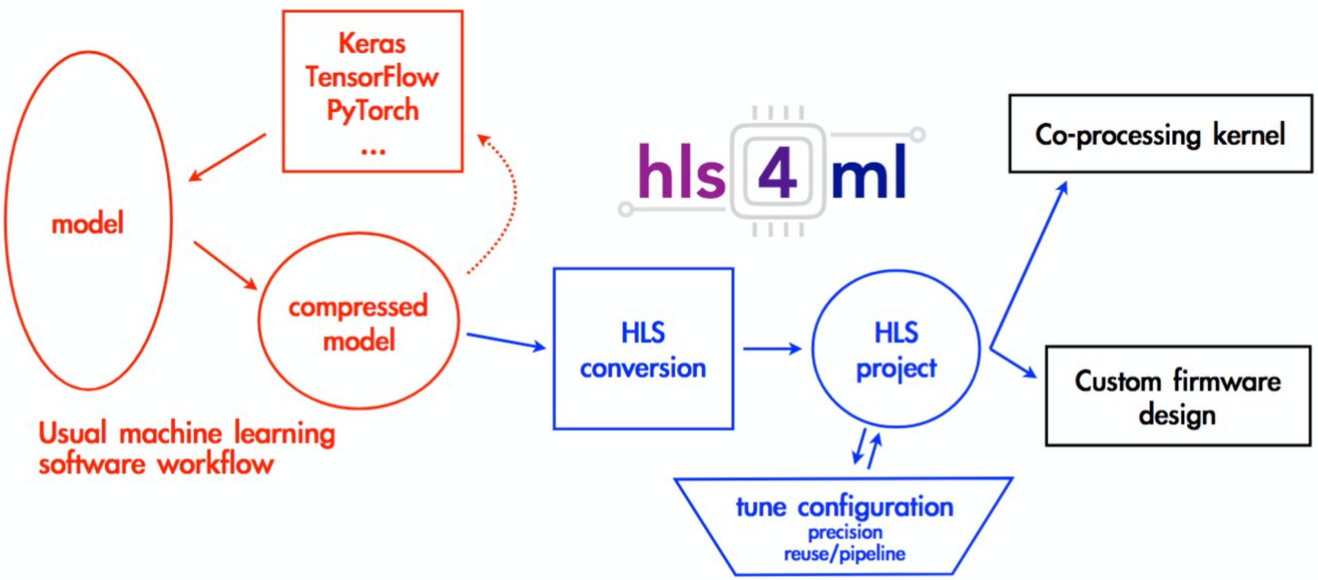
Successive iterations on a selected event

See our poster on Track 6 for more details
https://indico.cern.ch/event/587955/contributions/2937570/

Hardware Consideration

Inference on FPGA

- Demo at NIPS 2017 of implementing neural networks on FPGA
- Collaborating with hls4ml team to push the graph neural networks models to the nexts level



See Jennifer's talk during this event
https://indico.cern.ch/event/587955/contributions/2937529/

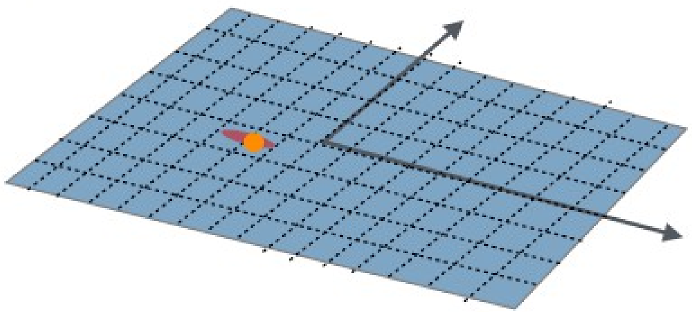
Tracking **Not** In a Nutshell

Several Times

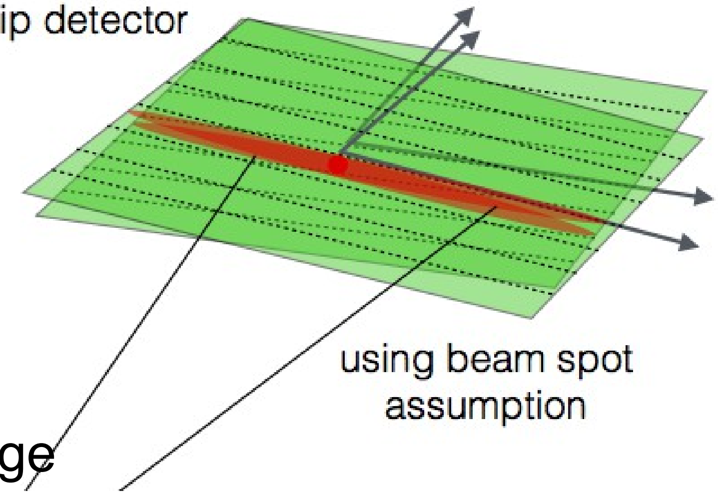
- **Hits preparation**
- **Seeding**
- **Pattern recognition**
- **Track fitting**
- **Track cleaning**

Hit Preparation

pixel detector

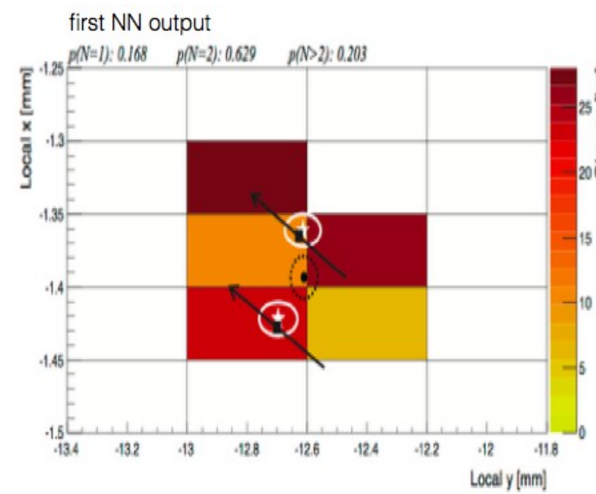


strip detector



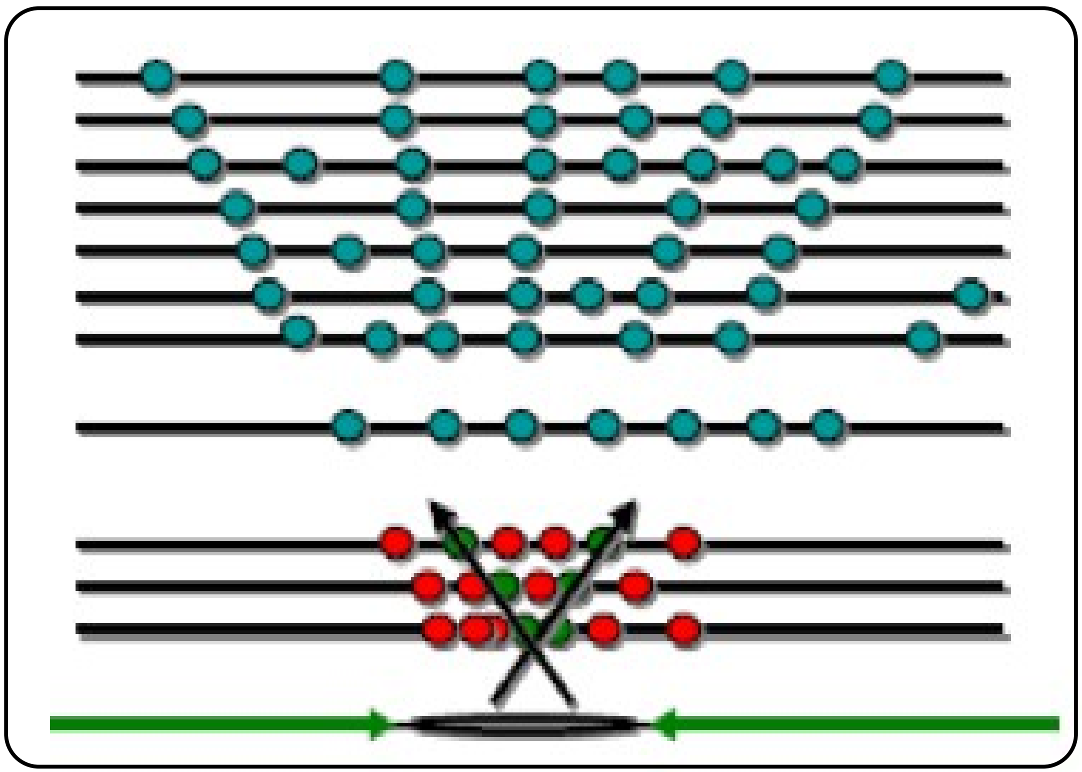
using beam spot assumption

- Calculate the hit position from barycenter of charge deposits
- Use of neural net classifier to split cluster in ATLAS
- Access to trajectory local parameter from cluster shape
- Remove hits from previous tracking iterations
- HL-LHC design include double layers giving more constraints on the local trajectory parameters



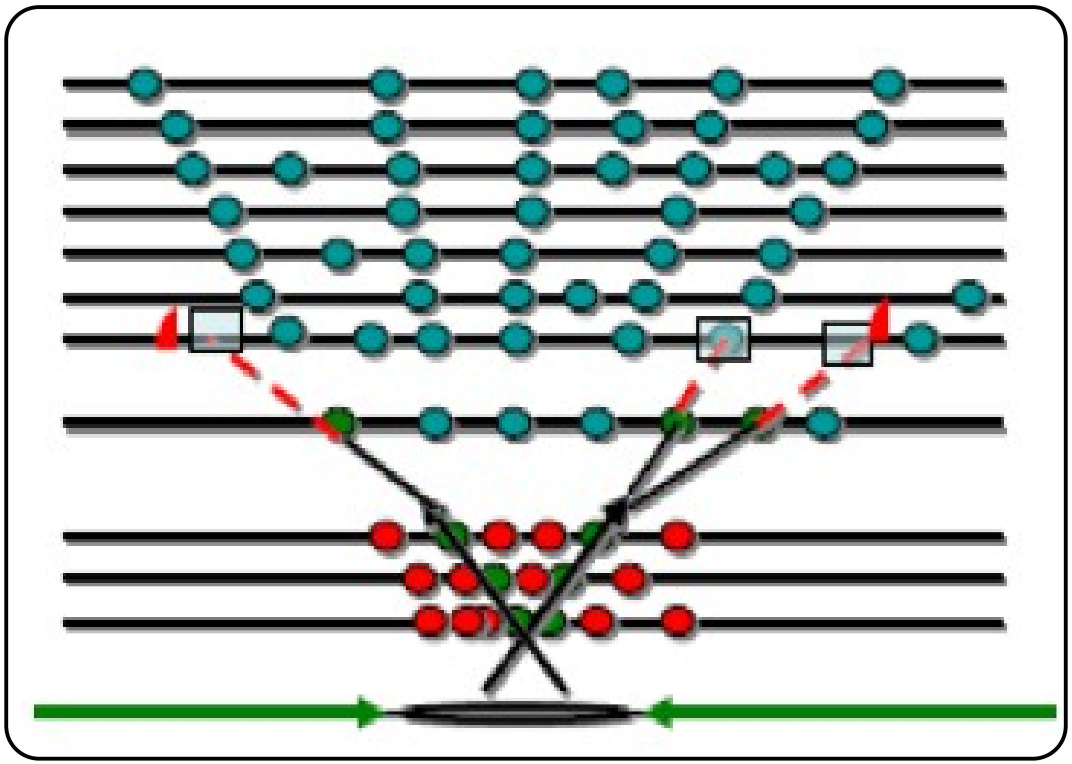
Example of cluster split

Seeding



- Combinatorics of 2 or 3 hits with tight/loose constraints to the beam spot or vertex
- Seed cleaning/purity plays in an important in reducing the CPU requirements of sub-sequent steps
  - ➔ Consider pixel cluster shape and charge to remove incompatible seeds
- Initial track parameters from helix fit

Pattern Recognition



- Use of the Kalman filter formalism with weight matrix
- Identify possible next layers from geometrical considerations
- Combinatorics with compatibles hits, retain N best candidates
- No smoothing procedure
- Resilient to missing modules
- Hits are mostly belonging to one track and one track only
- Hit sharing can happen in dense events, in the innermost part
- Lots of hits from low momentum particles

HEP.TrkX
J.-R. Vlimant

Kalman Filter

$$K_k = C_{k|k-1} H_k^{\top} (V_k + H_k C_{k|k-1} H_k^{\top})^{-1}$$
$$p_{k|k} = p_{k|k-1} + K_k (m_k - H_k p_{k|k-1})$$
$$C_{k|k-1} = (I - K_k H_k) C_{k|k-1}$$

H_k is the projection matrix

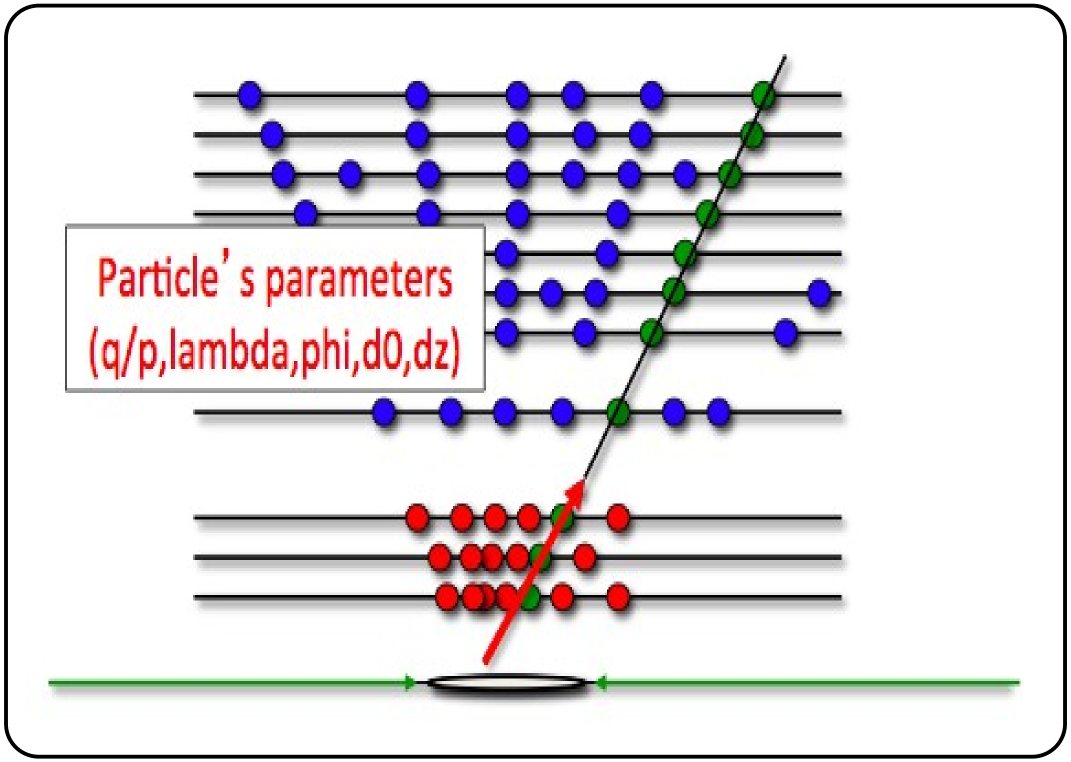
V_k is the hit covariance matrix

$p_{i|j}$ is the trajectory state at i given j

$C_{i|j}$ is the trajectory state covariance matrix at i given j

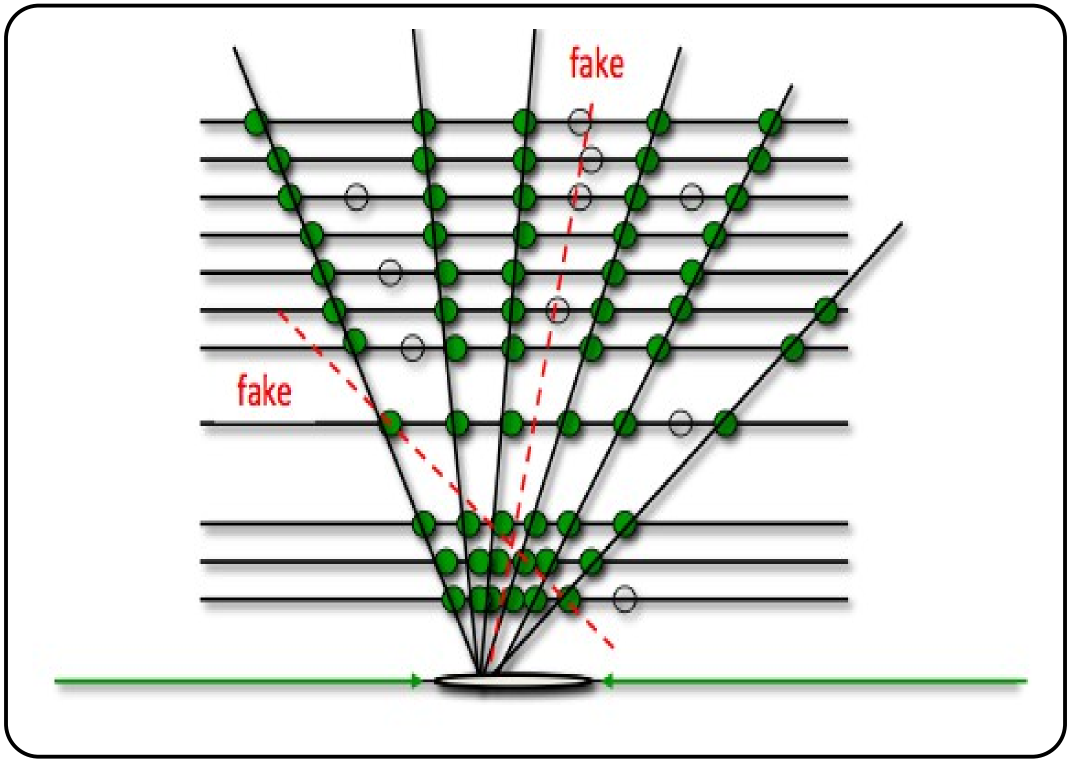
- Trajectory state propagation done either
  - ✓ Analytical (helix, fastest)
  - ✓ Stepping helix (fast)
  - ✓ Runge-Kutta (slow)
- Material effect added to trajectory state covariance
- Projection matrix of local helix parameters onto module surface
  - → Trivial expression due to local helix parametrisation
- Hits covariance matrix for pixel and stereo hits properly formed
  - ✗ Issue with strip hits and longitudinal error being non gaussian (square)

Track Fitting



- Use of the Kalman filter formalism with weight matrix
- Use of smoothing procedure to identify outliers
- Field non uniformity are taken into account
- Detector alignment taken into account

Cleaning, Selection



- Track quality estimated using ranking or classification method
  ➔Use of MVA

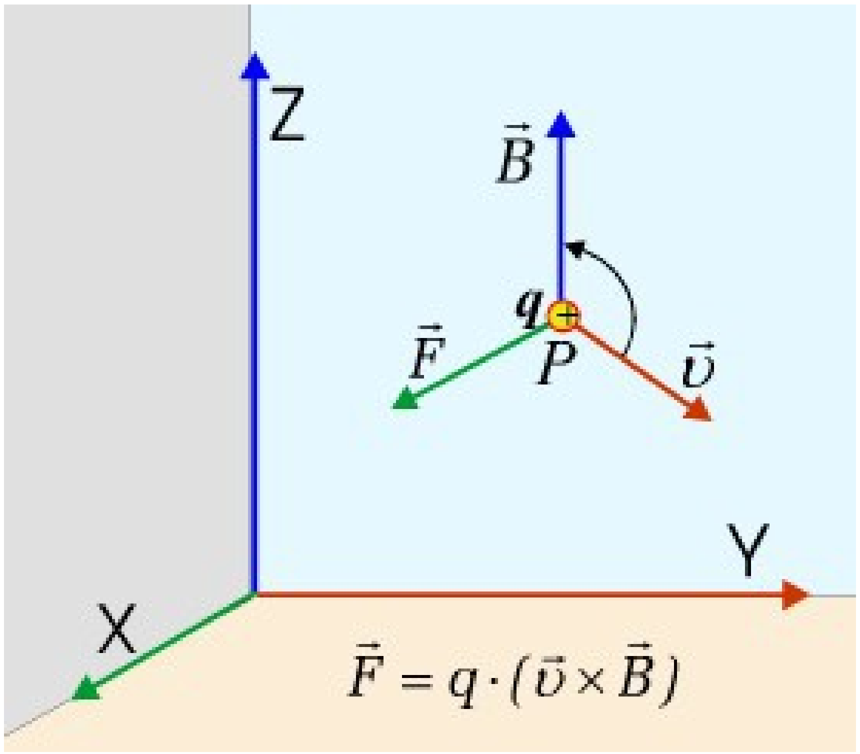
- Hits from high quality tracks are remove for the next iterations where applicable

A Charged Particle Journey

First order effect : electromagnetic elastic interaction of the charge particle with nuclei (heavy and multiply charged) and electrons (light and single charged)

Second order effect : inelastic interaction with nuclei.

Magnetic Field

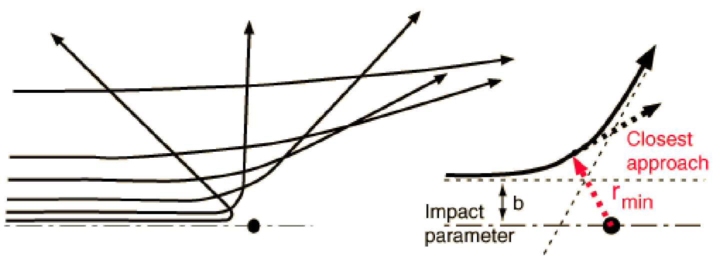


- Magnetic fieldB acts on charged particles in motion : Lorentz Force
- The solution in uniform magnetic field is an helix along the field : 5 parameters
- Helix radius proportional to the component of momentum perpendicular to B
- Separate particles in dense environment

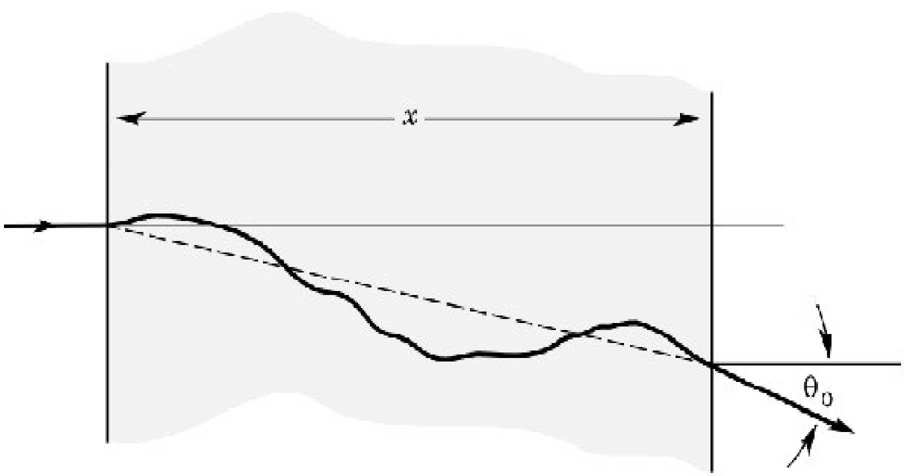
➔ Bending induces radiation : bremsstrahlung

➔ The magnetic field has to be known to a good precision for accurate tracking of particle

Multiple Scattering



- **Deflection on nuclei** (effect from electron are negligible)
- Addition of scattering processes
- Gaussian approximation valid for substantial material traversed



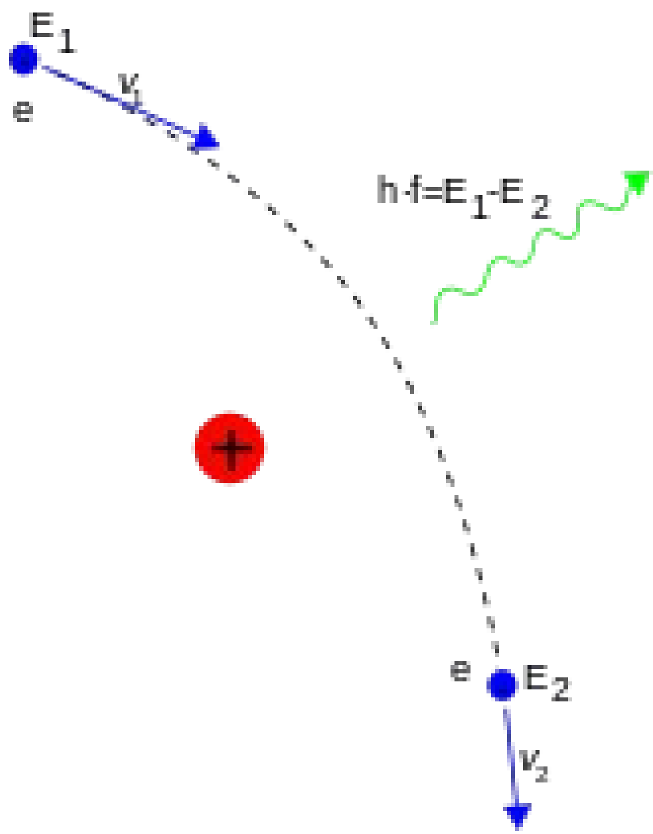
Gaussian Approximation

$$\theta^2 = \left(\frac{13.6MeV}{\beta cp}\right)^2 * \frac{x}{X_0}$$

β -particle velocity
ρ – material density
P - particle momenta

HEP.TrkX
J.-R. Vlimant

Bremsstrahlung



- Electromagnetic radiation of charged particles under acceleration due to nuclei charge
- Significant at low mass or high energy
- Discontinuity in energy loss spectrum due to photon emission and track curvature

➔ Can be observed as kink in the trajectory or presence of collinear energetic photons

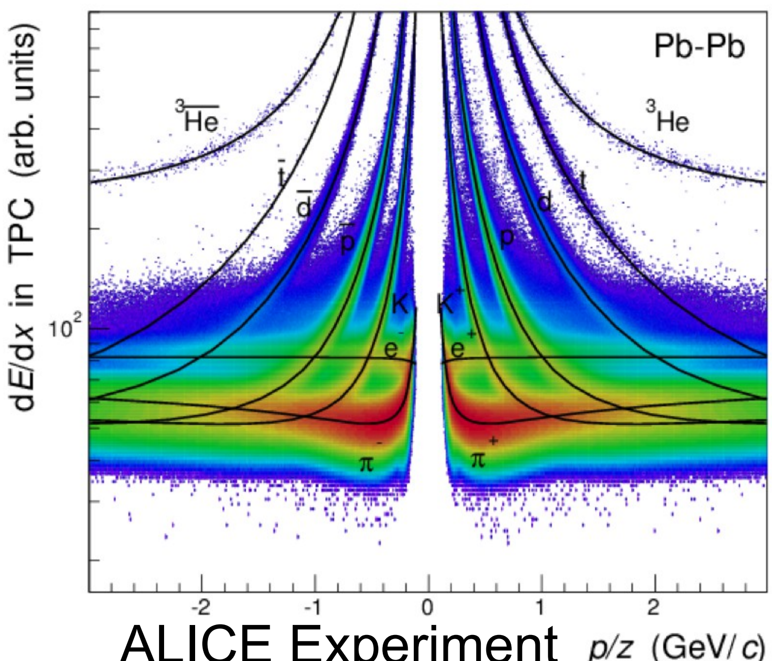
Energy Loss

- Momentum transfer to electrons when traversing material (effect of nuclei is negligible

$$dE/dx = k_1 \frac{Z}{A}\frac{1}{\beta^2}\rho\left(\ln\left(\frac{2m_e c^2\beta^2}{I(1-\beta^2)}\right) - \beta^2 - \frac{\delta}{2}\right)$$

- Energy loss at low momentum depends on mass : can be used as mass spectrometer

β -particle velocity
ρ – material density
Z - atomic number of absorber
A – mass number of absorber
I – mean excitation energy
δ – density effect correction factor – material dependent and β dependent



ALICE Experiment

HEP.TrkX
J.-R. Vlimant

Summary on Material Effects

- Collective effects can be estimated statistically and taken into account in how they modify the trajectory
- Bremstrahlung and nuclear interactions significantly distort trajectories