The use of new methods for processing data of a physical experiment. Application of machine learning methods on the NICA complex.

Triplet Siamese Network for Event Unraveling in the SPD Experiment

Borisov M., Goncharov P., Ososkov G., Rusov D.

Saint Petersburg, 2023

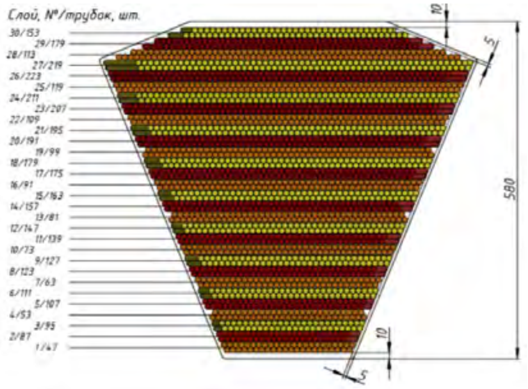
Introduction

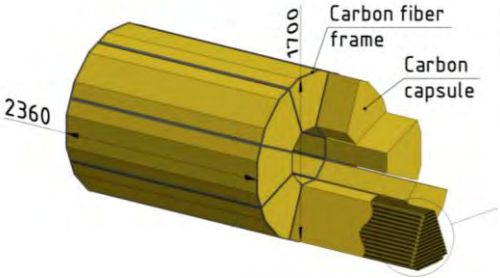
SPD (Spin Physics Detector) is a future experiment of the NICA project.

The main goal of the experiment is to check the predictions of quantum chromodynamics (QCD) and study the spin structure of nucleons through the collision of polarized protons.

The **frequency of events** at the design luminosity of the collider **will reach 3 MHz**

It is planned to store for further processing only 2-5% of the original data stream.





Problem Statement

In the context of the SPD experiment within the NICA project, a significant challenge arises in processing vast amounts of data to **extract valuable events**.

For the SPD experiment, in which events are expected to arrive with a frequency of 3 MHz, the data acquisition is supposed to be performed in time slices, during one **time slice up to 40 events with overlapping tracks** may appear.

The process of extracting **valuable events**:

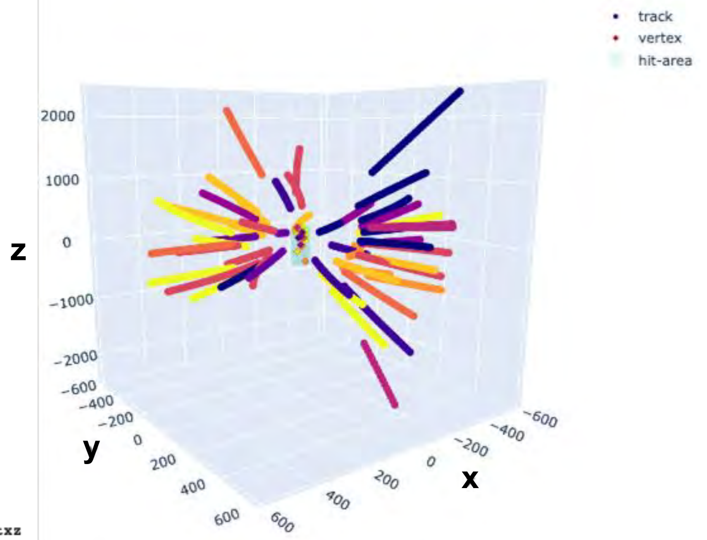
Online tracking (TrackNET) → **Unraveling Time-Slices of Events** → Filtering events of interest

doi.org/10.22323/1.429.0005

*In the present task, it is assumed that the tracks are already recognized.

Simulation of events

- Python script for spiral approximation of particle trajectory.
- The number of tracks in each event is from 1 to 10.
- The transverse momentum of the particle is a random number with a uniform distribution in the range of values from 100 to 1000 MeV/s.
- The coordinates of the vertices are also random and are chosen from the known region of possible particle collisions.
- The trajectory of the particle is represented by a set of points on a spiral segment.
- A detector configuration with **35 stations** is considered.
- Detector inefficiency is modeled as the probability that a hit will be removed from the dataset. **Detector efficiency values of 99% and 98% are used**.



Example of a model time-slice in the SPD experiment
(for 10 events in a slide)

	evt	x	y	z	station	trk	px	py	pz	vtxx	vtxy	vtxz
0	0	-268.018768	33.173191	565.522303	1	0	-507.704732	94.421070	851.451364	-1.253358	-15.544558	120.099724
1	0	-276.910426	34.872703	580.684414	2	0	-507.312956	96.503876	851.451364	-1.253358	-15.544558	120.099724
2	0	-286.027191	36.580065	595.575991	3	0	-506.912583	98.585324	851.451364	-1.253358	-15.544558	120.099724
3	0	-294.664452	38.385470	610.836264	4	0	-506.503616	100.665389	851.451364	-1.253358	-15.544558	120.099724
4	0	-303.774233	40.166026	625.997983	5	0	-506.086055	102.744042	851.451364	-1.253358	-15.544558	120.099724
...	...	...	...	...	...	...	...	...	...	...	...	...

Approaches

to solve the **Unraveling Time-Slices of Events** problem

1) **Predict vertices** → **Clustering**

- regression
- line/spline interpolation

- k-means

2) **Embedding mining** → **Clustering**

- Siamese network

- k-means

** The models are trained on prepared tracks from the simulation, in reality the input will be candidate tracks from the tracking algorithm.

1.Predict vertices

Models:

- **Linear/spline interpolation**
- **Random Forest Regressor**
- **Gradient Boosting Decision Tree (GBDT)**

Data preparation:

```
evt,x,y,z,station,trk,px,py,pz,vtxx,vtxy,vtxz
```

```
x1,x2,xn … ,x35,y1,y2,yn … ,y35, z1,z2,zn … ,z35,evt,trk,vtxx,vtxy,vtxz
```

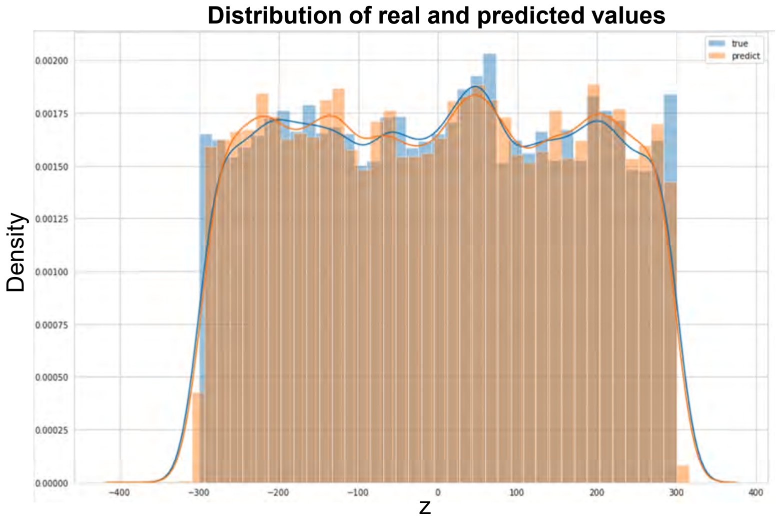
efficiency=1

'loss_function': 'MAE'
'learning_rate': 0.0631
'iterations': 1743
'max_depth': 9
'l2_leaf_reg': 1.029
'bagging_temperature': 4.404

MSE	205.03
MAE	5.258
MAPE	0.208

Yandex CatBoost

OPTUNA



1.Model Interpretation

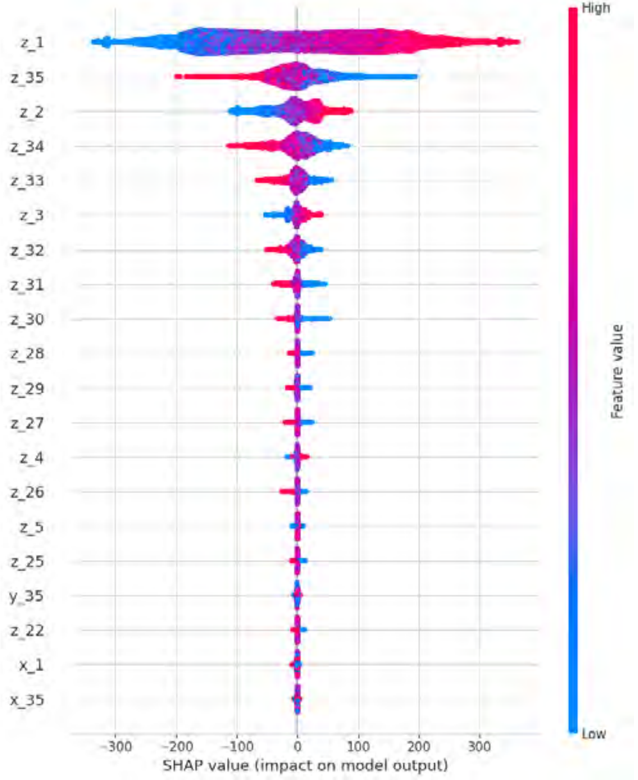
SHAP (SHapley Additive exPlanations)
is a game theoretic approach to explain the output of any machine learning model.

The Shepley values for each trait are calculated by looking at all possible combinations of features and comparing model predictions with and without those features.

x axis - effect on targeting
y axis - sorted by importance features
color - value of the target (blue is the smallest, red is the largest)
thickness - concentration of observations

the most important feature is the z-coordinate.

summary plot



SHAP

1.Clustering

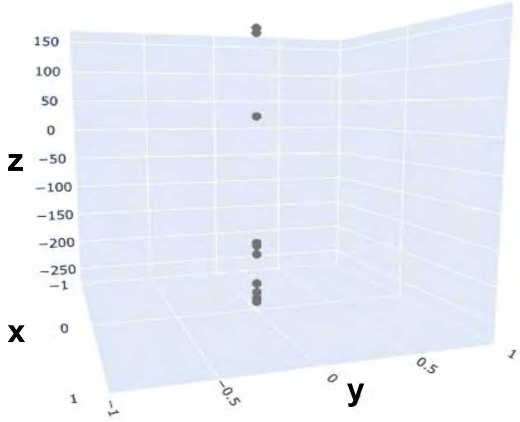
Clustering of vertices predicted by regression

Number of clusters = Number of events in the time-slice

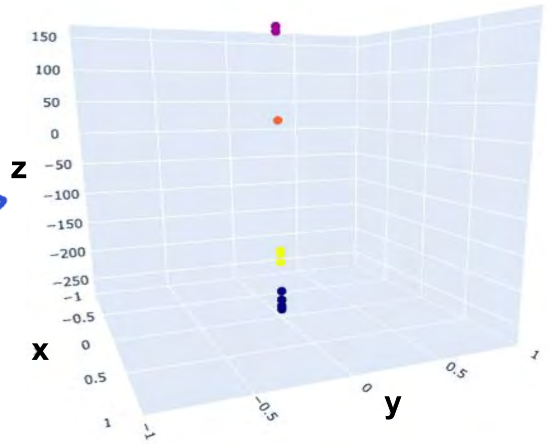
Method:

- **K-means**
  - **cosine**
  - **euclidean**

Clustering is applied to each slice.



Vertex prediction result



Vertex clustering result

* In reality, we do not know exactly how many events are in the timeslice.
In the experiments we use the exact number for proof of concept.

* demonstration for a time-slice with 5 events.

1.Clustering metrics (Internal)

The **silhouette** value shows how similar an object is to its cluster compared to other clusters.

The **Davies-Bouldin Index** calculates compactness as the distance from cluster objects to their centroids, and separability as the distance between the centroids.

	silhouette			davies_bouldin		
slice	**5**	**10**	**40**	**5**	**10**	**40**
samples	2019	1010	253	2019	1010	253
mean	0,82	0,79	0,65	0,25	0,26	0,43
std	0,09	0,05	0,04	0,09	0,08	0,05
25%	0,78	0,75	0,63	0,10	0,21	0,39
50%	0,84	0,79	0,65	0,15	0,25	0,43
75%	0,89	0,83	0,68	0,22	0,31	0,46

1.Clustering metrics (External)

* class labels are known

1.	slice	5	10	40
	samples	2019	1010	253
percentage of correct	tracks	0,67	0,71	0,23
	evts	0,71	0,72	0,24
	slices	0,32	0,46	0,02

1. Metrics are calculated based on the unambiguous matching of the predicted cluster to the event.
2. An event is considered unraveled if it has 1 cluster and is not included in other events.
3. A correctly unraveled slice is a slice with all events correctly unraveled;

2.

slice	5	10	40
samples	2019	1010	253
Precision	0,864	0,355	0,291
Recall	0,886	0,491	0,124
F1-score	0,857	0,401	0,046
Accuracy	0,921	0,629	0,211

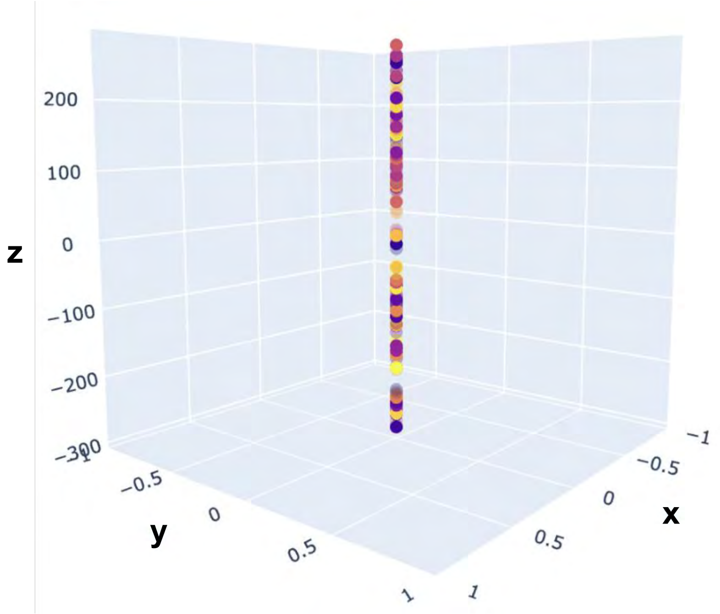
1. For each cluster **build a set of track_id**, which are included in it.
2. Count the **pairwise intersections** of cluster sets and event sets.
3. **Sort** from largest to smallest intersection.
4. **Find** cluster-event pairs with the **largest intersection.**
5. If more than one cluster was found for some event, take only the one with the largest intersection.
6. Assign a label to each cluster, based on the found pair of event;

* in the future we want to test the Hungarian matching algorithm

Problems…

The main problem with the vertex prediction approach and further clustering is that they are **close and overlapping**.

Based on the received metrics, **this approach is not applicable to unraveling 40 events in a slice**.



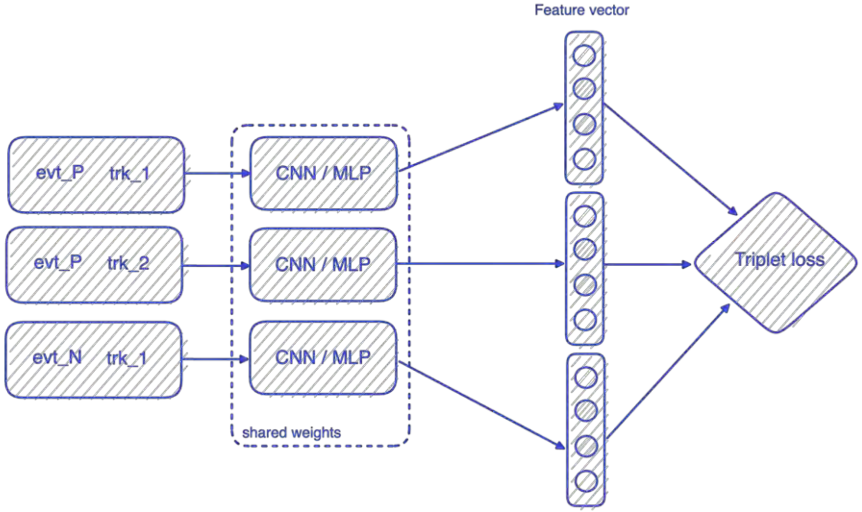
Vertex visualization for 40 events in the time slice.

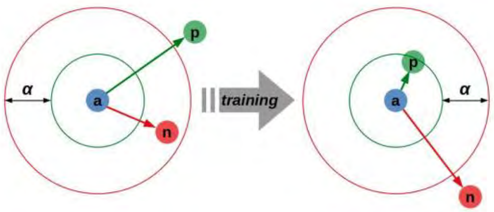
* In a real experiment, x and y will not be fixed as in a simulation.

2.Embedding mining

The idea is that tracks from one event are positive examples and tracks from different events are negative examples.

The Siamese neural network must learn how to extract such vectors of embeddings for tracks. So that the vectors of tracks coming from the **same vertex are close** in the feature space. And vectors of tracks from **different vertices are far away** from each other in the feature space.





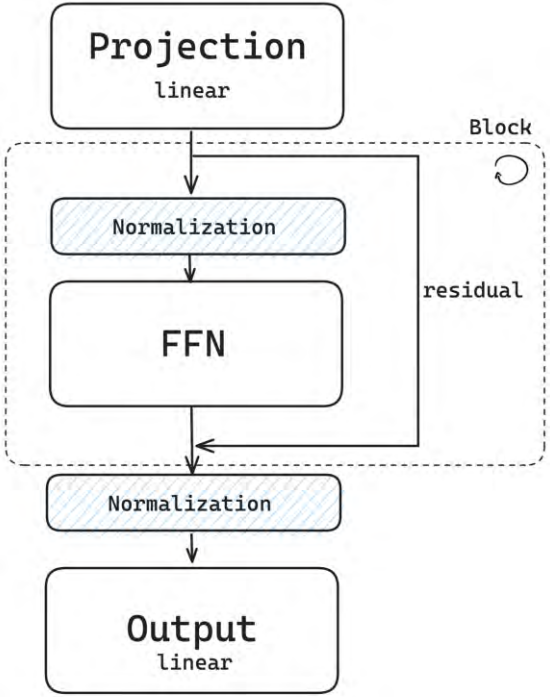
The Siamese network works as a generator of feature vectors.

2.Architecture

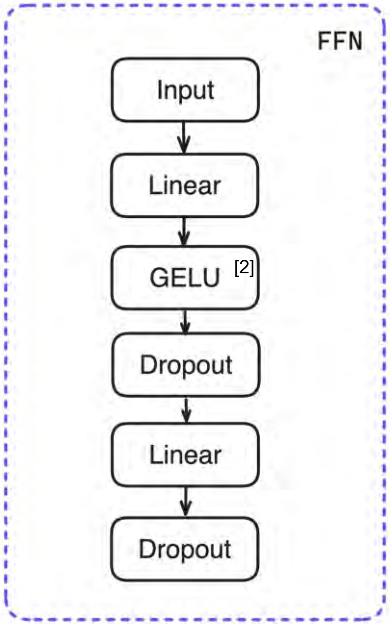
The architecture of the triplet siamese network (TSN) was inspired by FFN layer in transformer models and by MLP block for MLP Mixer [1] model.

For the loss function SNR distance was chosen.

n_stations (35) * 3 coords = 105



(1,32)



FFN - Feed-Forward Network

1. MLP-Mixer: An all-MLP Architecture for Vision
2. Why GELU

2.Training

For training, we used the PLM framework.

PLM has convenient methods for:

- Miner
- Trainer
- Loss
- Tester
- and other…

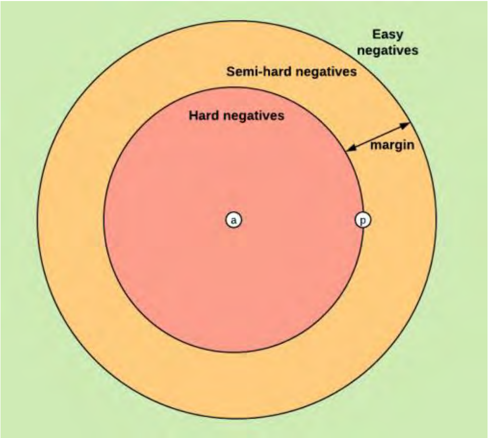
Params (best model):

n_blocks = 5
output_dim = 32

triplet_margin = 0.1
type_of_triplets = all

distance = euclidean/snr dist [1]
normalizer = MinMaxScaler() [-1;1]

train = 10k (timeslices)
test =1k
epoch = 80
time ≈ 9 hr



$$loss = \max(0, dist(A,P) - dist(A,N) + margin)$$

1. Signal-to-Noise Ratio: A Robust Distance Metric for Deep Metric Learning

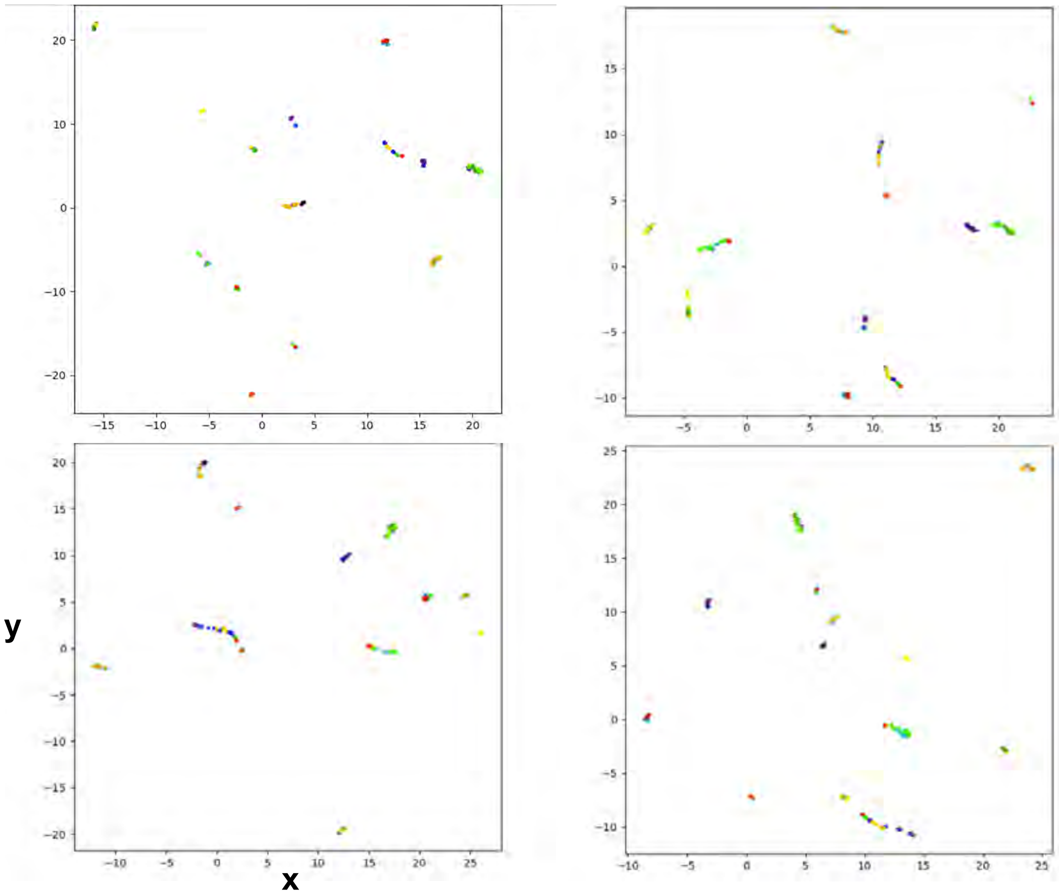
2.Embeddings visualization

For logging, we use Tensorboard.

The result of applying the trained model to tracks from time-slice.

We apply UMAP algorithm to reduce the dimensionality of the original 32D vectors to 2D

Tracks from different events in a time-slice are highlighted with different colors



2.Clustering

Clustering of embeddings received after the network.

Method:

- **K-means**

euclidean distance

Fixed number of events.

- num_cluster=num_evts
- n_init=10

Approach **2 outperforms** approach 1 **in all metrics**.

Internal * mean score

(1) Vertices prediction method

Silhouette	0,291
Davies bouldin	0,124

(2) Embedding clustering

Silhouette	0,725
Davies bouldin	0,655

40 events in timeslice

External * average macro

(1) Vertices prediction method

Precision	0,291
Recall	0,124
F1-score	0,046
Accuracy	0,211

(2) Embedding clustering

Precision	0,811
Recall	0,843
F1-score	0,818
Accuracy	0,895

* The combination approach described on slide 10 was used.

2.Inference time

BATCH_SIZE = 1 time-slice (the number of tracks varies)

Unraveling time per time slice (40 events)

device	embedder (s)	clustering (s)
CPU (Apple m1pro 10 cores)	0.0177 (±0.0059)	0.1884 (±0.0919)
GPU (Apple m1pro 10 cores)	0.0279 (±0.0099)	0.2010 (±0.1031)
GPU Tesla V100 (CPU[1])	0.0076 (±0.0015)	0.1126 (±0.0053)

K-means from scikit-learn package (running on CPU)

Clustering is a bottleneck.

Ideas for acceleration:

- Transfer clustering on GPU device
- Find library with faster clustering
- Architecture optimization (params, layers)
- ONNX/TensoRT

* not included in the measurements:
- transfer time between devices (CPU/GPU)
- time slice generation

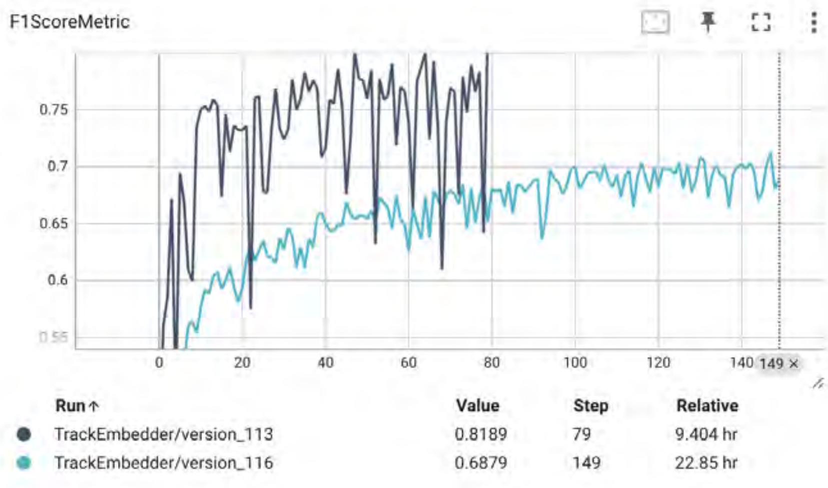
1. Intel(R) Xeon(R) Gold 6148 CPU @ 2.40GHz

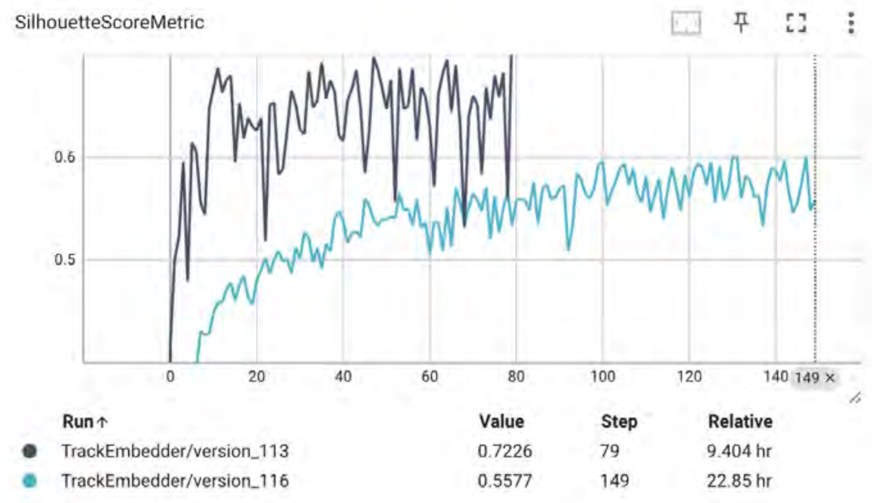
Conclusion and outlook

- An approach for predicting the vertex of an event has been developed.
- Developed an approach to evaluate the quality of clustering.
- Pipeline for unraveling events within a slice has been developed. But this approach turned out to be inapplicable for a large number of events in a slice.
- A disentanglement approach based on clustering of embeddings after Siamese network is developed.

- Processing an unknown number of events (clusters).
- Automatic hyperparameter selection based on Bayesian approaches.
- Processing of skips inside the track.
- Accelerating the clustering step.
- Testing other SOTA architectures.
- Testing the approach on a more physical generator.

Training with missing stations





Missing stations are filled with 0.

GELU

Why GELU