Tracking aware metric learning for particle reconstruction

Sabrina Amrouche* **Tobias Golling**
Université de Genève

Noemi Calace **Moritz Kiehn** **Andreas Salzburger**
European Organization for Nuclear Research CERN

Abstract

We propose a model for learning a metric that simplifies the finding of particle trajectories. A new learning process is designed such that geometrical and clustering constraints from spatial measurements are incorporated. The network allows to map any set of input points into a new space where traces produced by the same particle are clustered. The approach is experiment agnostic and results are demonstrated on the TrackML dataset.

1 Introduction

Particle tracking in high energy physics is a particularly challenging task. At the time of writing, no machine learning based solution was able to solve the TrackML challenge addressing both efficiency and speed [1]. Moreover, applying off-the-shelf deep learning models that do not adequately incorporate prior knowledge or physical constraints leads to poor generalization when confronted with data taken in real world experiments. Point tracking, as a general problem requires a trajectory hypothesis and therefore only enables a combinatorial approach where invalid roads are discarded at a later stage. The complexity of the task is further increased in high energy physics and especially at the High Luminosity Large Hadron Collider (HL-LHC) where the scale of the data produced leads to a combinatorial explosion. A number of studies [1] investigated these challenges using the TrackML dataset as benchmark. In this work, we propose to evolve metric learning based techniques by adding physics driven constraints and noise robustness.

The core idea is to design a model able to map particle hits (technically similar to a 3D point cloud) into a new space where the trajectories are well separated with the euclidean distance. Since we use simulation data, it is possible to train the model in a supervised way with labels attached to the input points. The use of Monte Carlo simulated data to design and validate the track reconstruction algorithms is indeed a common technique in high energy physics. The labels (ground truth) are used in the loss function to incorporate the geometric constraints of the output space. We use Approximate Nearest Neighbors (ANN) as it was established in [2][3] to provide high quality independent bins as model input. ANN(s) [4] are considered the current state of the art techniques for fast similarity search. An ANN index is built for each event and hashes refer to ANN queries : a set of neighboring points along the angular distance. These points (3 dimensional) constitute the input of the model presented in this work.

*Corresponding author

Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020), Vancouver, Canada.

2 Tracking Loss Function: Push and Pull

Charged particle tracking is essentially a clustering problem where the number of clusters is very high and the cluster size relatively low. In any clustering problem, the ideal feature space is one where clusters are *isolated* enough for a distance threshold to split them and *compact* enough for the clusters not to be split themselves. Our loss function is based on these two constraints and an ideal learning strategy finds the balance between pulling and pushing data points until trajectories are well separated. The performance of the model is measured by the total number of well reconstructed particles, i.e. complete clusters. The weighted loss function is of the form :

$$\mathcal{L}_{TrackNet} = (\alpha\mathcal{L}_C + \beta\mathcal{L}_I + \gamma\mathcal{L}_{CL})^{\zeta} \tag{1}$$

$$\mathcal{L}_C = \sum_{i=0}^{K} S(c_i) \;\; ; \;\; \mathcal{L}_I = \frac{1}{S(\mu_{[1..k]})} \;\; ; \;\; \mathcal{L}_{CL} = \sum_{i=0}^{n} \frac{b_i - a_i}{\max(a_i, b_i)} \tag{2}$$

where α, β, γ are weights of the loss function. During the optimization procedure, each weight controls the relative importance and contribution of its associated term. ζ allows the amplification of the total loss. $S(c_i)$ is the variance of cluster i and is computed for each of the K clusters present in the current batch. $S(\mu_{[1..k]})$ is the variance of the K centroids. During the training, clusters are formed from the network predictions using the true labels. To add further constraints on the cluster compactness, we introduce a third loss term $\mathcal{L}_{CL}$ that represents the silhouette coefficient of the true clusters in the learned space. The silhouette coefficient allows for the homogenisation of cluster shapes. It is defined in equation 2 where *a* is the mean intra cluster distance and *b* the mean distance between a point i and every other point located in the nearest cluster. This quantity is average over the n points considered in the model input. $\mathcal{L}_C$ and $\mathcal{L}_I$ alone, act on the centroid of the cluster and would allow a non compact cluster shape (stretched cluster). The silhouette coefficient on the other hand, acts on all the distances within a cluster.

Figure 1 describes the proposed model. A set of hits (as produced by individual particle traces) is given as input. The hits can be associated to multiple particles and for illustration we choose a set with a leading long particle P1 and smaller one P2. In this work, we only rely on the geometrical information of the hits : global coordinates (3D) and the coordinates of the module responsible of the readout (2D). Hence, each data point has five dimensions.

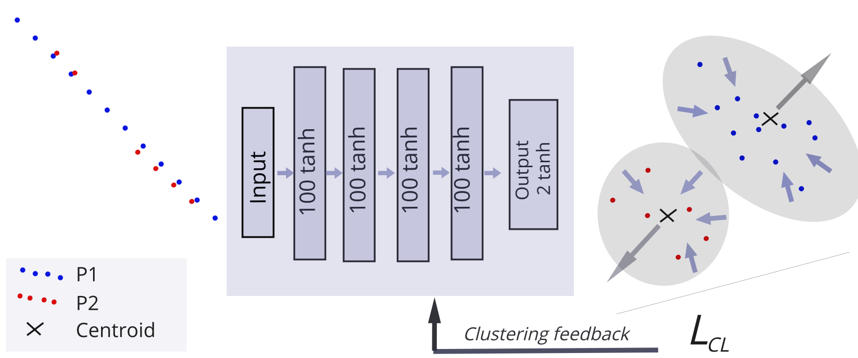
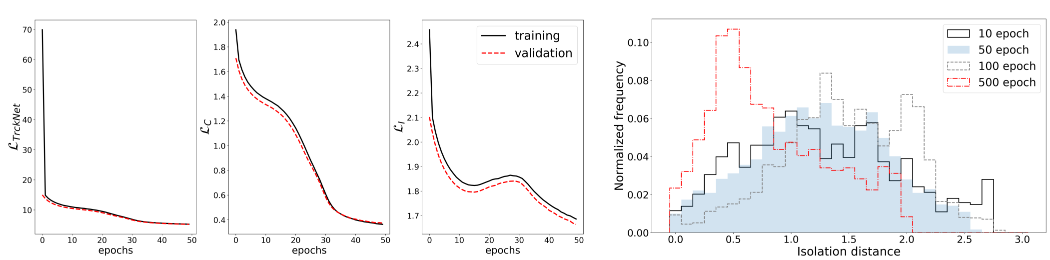


Figure 1: Proposed model architecture: the model takes a bucket from the ANN search as input and maps it into a new feature space where the clusters representing different particles are separated.

The network is composed of 4 dense layers with 50 nodes each. The gradual action of the loss function is presented in the right side of Figure 1. The arrows represent the pushing/pulling strategy of the learning. The main contribution of this work is the integration of the *desired output* shape and constraints into the loss function. The wealth of particle patterns and trajectories is encoded in the labels used to create the clusters during training. The evaluation metric to determine an architecture performance is discussed in Section 3. The best model was obtained through a grid search that

jointly scanned the network hyper-parameters and the loss weights. The best performing model uses 6 dimensions in the output with a tanh activation function that conveniently restricts the space to the range (-1,1). Although the model is not optimized for a specific input size, we demonstrate our approach on hashes of 10 hits since we are interested in *particle seeds* of four hits or more, i.e. innermost part of the particle that allows to recover the full trajectory[1]. The hashes are constructed from the angular distance of the hits.
Figure 2a shows the evolution of the loss function over the epochs as well as the compactness loss $\mathcal{L}_C$ and the isolation loss $\mathcal{L}_I$. The general variance within a cluster is decreasing and so is the inverse of the isolation between clusters. The loss pattern on unseen buckets (validation) shows a similar smooth decreasing tendency. We use early stopping on the validation set to determine the ideal number of epochs.

Clustering the output
The proposed model pushes apart particles to enable an easy and intuitive track clustering. In the learned feature space, it falls upon the clustering algorithm to decide at which distance to stop considering points as similar (same cluster). Indeed, the task of the network is to ensure a large enough distance between groups of particle hits. We propose to use a modified agglomerative clustering to retrieve the particles. Agglomerative Clustering (AC) aligns well with the concept of particle tracking since it starts from a single data point and merges clusters until a stopping condition is met. Since particles have on average 10 data points, AC converges rapidly within a bucket. We use as stopping criteria a threshold distance after which hits are considered to belong to two different clusters. In practice, this distance is the same distance our network learns to increase, i.e. the distance between cluster edges. We will refer to this quantity as the isolation distance. Figure 2b shows to the evolution of the isolation distance of the largest cluster over the epochs. This metric is computed from the network output space. The hits are grouped using their true labels and for each mapped bucket, we record the distance between the largest cluster (largest particle) and the nearest hit that does not belong to this particle. A good model maximizes this distance and pushes the largest particle (the one we are interested to retrieve) away from the rest of the hits. In the first set of epochs (10, 50 and 100) we can observe the general trend of regressing first bins as we have fewer examples that are mapped to a space where particles are close by. The model at epoch 100 shows the best performance : higher isolation distances. At epoch 500 however, the model seem to collapses with a clear shrinking of the isolation distance. This behavior at higher epochs seems independent from the weights assigned to the compactness and isolation terms in the loss function. Once the best model configuration is selected (maximizing the isolation distance), we determine the clustering threshold value that best discriminates clusters. This threshold is highly dependant on the network architecture and loss weights as different architectures produce different mappings and therefore isolation distances.



(a) (b)

Figure 2: Evolution of the model performance over the epochs. (a) shows the convergence of the model via the loss function and its terms $\mathcal{L}_I$ and $\mathcal{L}_C$. (b) describes the isolation distance of the largest cluster per output.

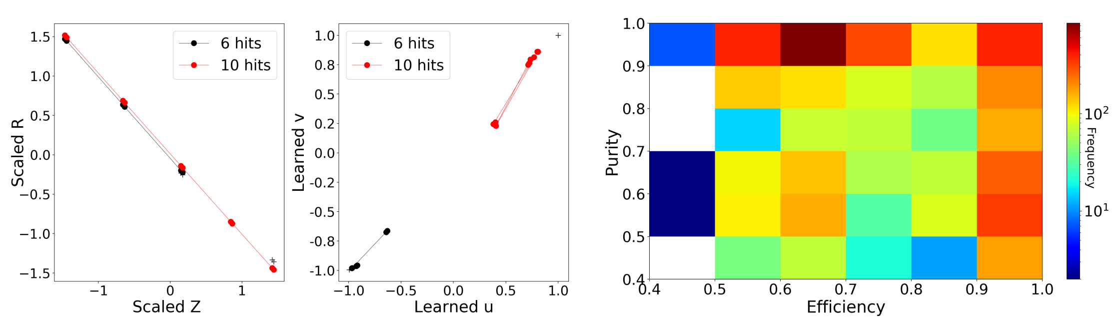
[1]It has been demonstrated that even classical pattern recognition methods are extremely efficient and relatively fast when fed with a high quality seed.

3 Testing and Results

In this study, we consider two performance evaluation metrics. Firstly, the similarity learning model is evaluated on the output representation, i.e. whether particles are clustered and pushed apart (Figure 2b). Secondly, the main particle tracking performance is whether the obtained clusters contain full particles (efficiency) and only full particles (purity). We generate as many hashes as there are data points in a TrackML event : approximately 100 thousand data points. The selected event is unseen during training. Using the trained model, every hash is mapped to the new feature space. There are two scenarios in this application. Either the hash (or bucket) contains a particle trajectory formed by four hits at least or the hits of the bucket do not contain any reconstructable particle (we require four data points to form a trajectory). Since the compactness of particles is enforced through the $\mathcal{L}_C$ in the training procedure, a dense (or containing dense regions) mapping will be an indicator of the presence of a particle. Therefore, we chose to reconstruct buckets that are mapped to clusters of at least four points. This threshold allows to discard noise (fake trajectories).

In the filtered resulting clusters, two metrics are considered : Efficiency and Purity. These metrics are defined at the cluster level. The efficiency of cluster c_i with associated labels $l_i = [p_1, ...p_n]$ is the ratio of the most common label occurrence and the total number of points sharing this label inside a bucket (and not the full event). That is, the fraction of the largest particle per cluster. The purity is the fraction of hits sharing the same label (per cluster). A cluster with a 100% efficiency and 100% purity contains a full particle (in the bucket) and only that. The inefficiency of hashing is not represented in these two quantities. Figure 3a presents an example of a mapping using the network on a 20 hits bucket to highlight the impact on long trajectories. The bucket, in the original scaled detector coordinates (R,Z) contains two close-by tracks (practically merging in this projection), with different colors, evolving from bottom to top. On the right side of the figure is the mapping of the network where the two tracks are clearly separated.

Figure 3b shows the resulting efficiency and purity on a full event for hashes of 10 hits. Each entry in the 2D histogram represents a cluster with at least 4 hits. Both performance metrics include noise buckets with no reconstructable particle. The efficiency (horizontal axis) reflects the network performance while the purity (vertical axis) is a function of the clustering performance. Despite an average of 74% efficiency, overall the inefficiencies are attributed to the clustering (less purity) rather than the similarity learning model. Both the particle size and cluster size in this 2D histogram average 6 hits. Although the trade-off between efficiency and purity is high enough, it is heavily penalized by the fixed nature of the clustering stopping criteria. A dynamic stopping criteria is likely to improve both clusters of 100% efficiency and low purity and those of 100% purity and low efficiency. Additionally, it is worth mentioning that the fraction of filtered clusters and buckets plays a major role in fake reduction and computational speed.



(a) (b)

Figure 3: (a) Example of a mapped bucket (left to right) with colors indicating different particle trajectories. Dark Crosses denote noise hits. (b) Joint distribution of the efficiency and purity per filtered cluster.

4 Conclusion and Outlook

Our work adapts and adjusts powerful deep learning tools to match the complex problems encountered in high energy physics. In this context, we have introduced a tracking targeted similarity learning procedure. The mapping is learned through the incorporation of geometrical and clustering constrains. The proposed model is able to map unseen data points into meaningful and tracking-adapted representations. The variety of particle trajectories in high energy physics makes it compelling to learn a new representation where a combinatorial approach is no longer needed. Deriving the clustering stopping criteria dynamically from the model mapping is an important next step.

Broader Impact

The reasoning behind the design of the loss function can be used in a wide range of point tracking application where expert knowledge is available. Although point tracking can be found in multiple domains, the considerations that guided this work are specific to high energy physics where the data rate and processing constraints are unparalleled. Since detector geometry is not a requirement of the approach, the model and loss design can be exploited in different experiments facing a high multiplicity of the data. At the time of writing, machine learning for charged particle tracking is in its infancy and no negative outcome is foreseen from this research. This paper seeks to promote the adaption of deep learning methods into high energy physics.

References

[1] S. Amrouche, L. Basara, P. Calafiura, V. Estrade, S. Farrell, D. R. Ferreira, L. Finnie, N. Finnie, C. Germain, V. V. Gligorov, *et al.*, "The tracking machine learning challenge: Accuracy phase," in *The NeurIPS'18 Competition*, pp. 231–264, Springer, 2020.

[2] S. Amrouche, T. Golling, M. Kiehn, C. Plant, and A. Salzburger, "Similarity hashing for charged particle tracking," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 1595–1600, IEEE, 2019.

[3] S. Amrouche, M. Kiehn, T. Golling, and A. Salzburger, "Hashing and metric learning for charged particle tracking,"

[4] M. Aumüller, E. Bernhardsson, and A. Faithfull, "Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms," in *International Conference on Similarity Search and Applications*, pp. 34–49, Springer, 2017.