Nikhef ML meeting, 31/01/25

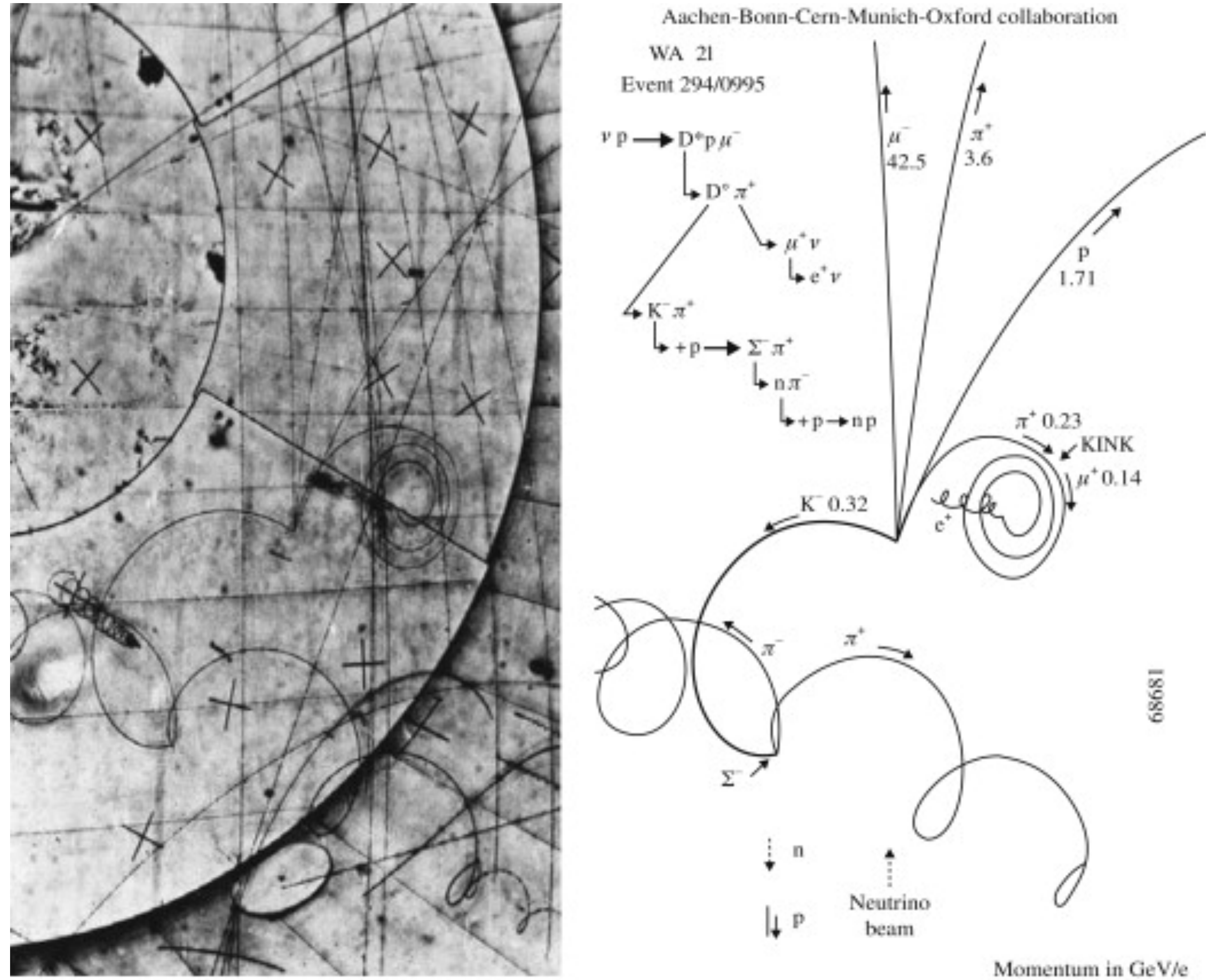
Charged particle tracking with transformers

Zef Wolffs

With work from Nadezhda Dobreva, Sascha Caron, Uraz Odyurt, Yue Zhao, Slav Pshenov

Introduction

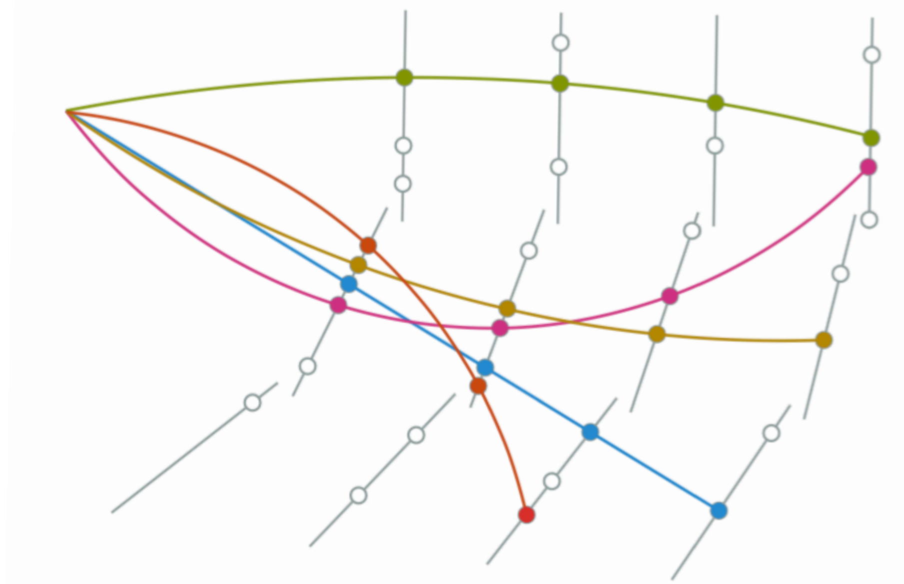
Big European bubble chamber at CERN (operation in 1970s) [1]



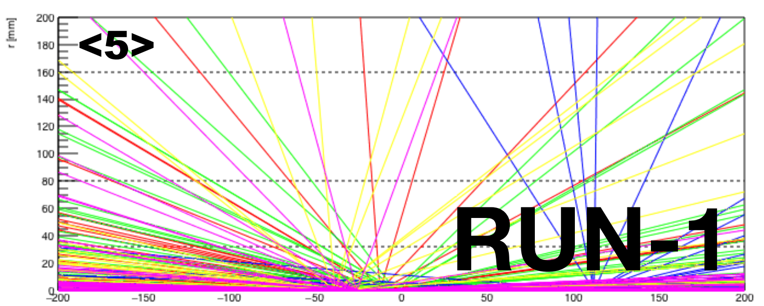
Tracking by meticulously following visible tracks [2]

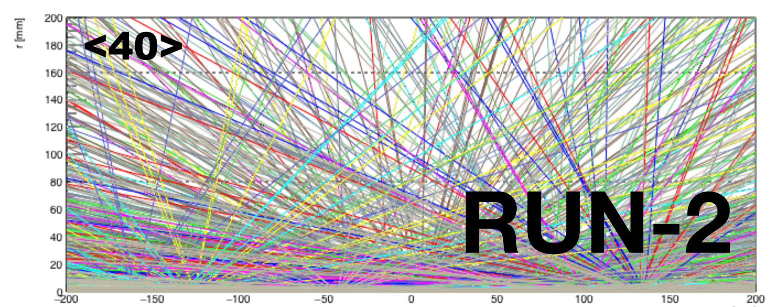
Introduction

- With modern **layered detector design** and **electronic readout systems** the situation looks rather different

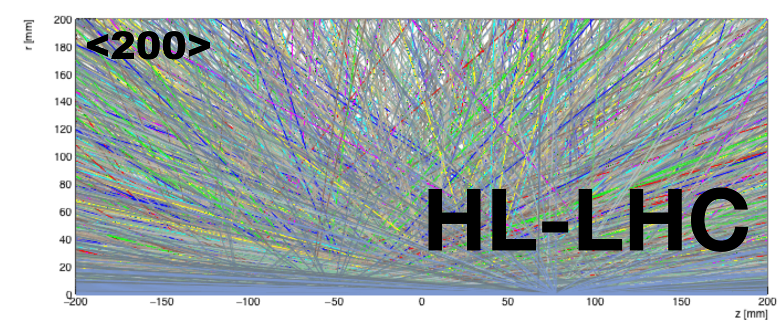


- However, **constructing tracks by hand** has for a while been **completely unfeasible**, and so computational techniques such as the **Kalman filter** have been adopted
- But with **upcoming HL-LHC** even traditional computational techniques such as the **Kalman filter may prove too inefficient**





[4]



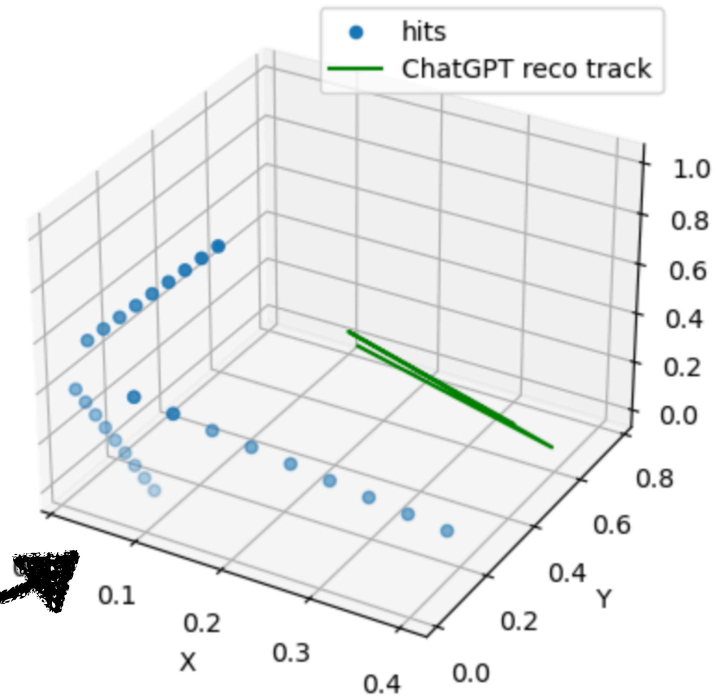
Algorithm Designs

Transformers

Z

Hi ChatGPT, could you convert for me the following matrix of hits in three dimensions (n_hits, 3) to tracks?

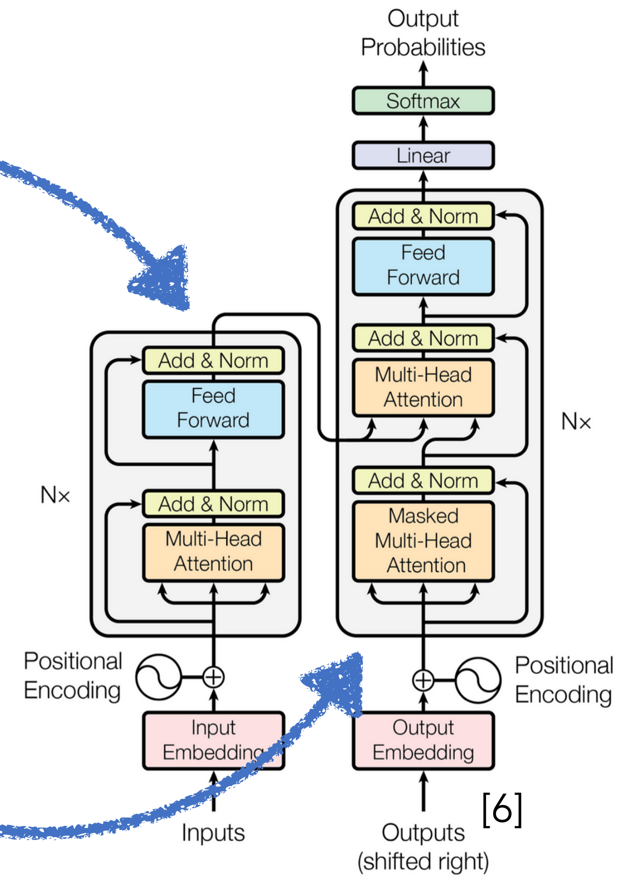
Certainly! Here are the hits that are part of one track based on the calculated z-coordinates:



- Asking ChatGPT is not the way…
  - We have to do something more sophisticated

The encoder model **takes some input and encodes it** into some latent space

The decoder model **takes as input the output of the encoder and some input sequence**, and **outputs the next item in the sequence**



[6]

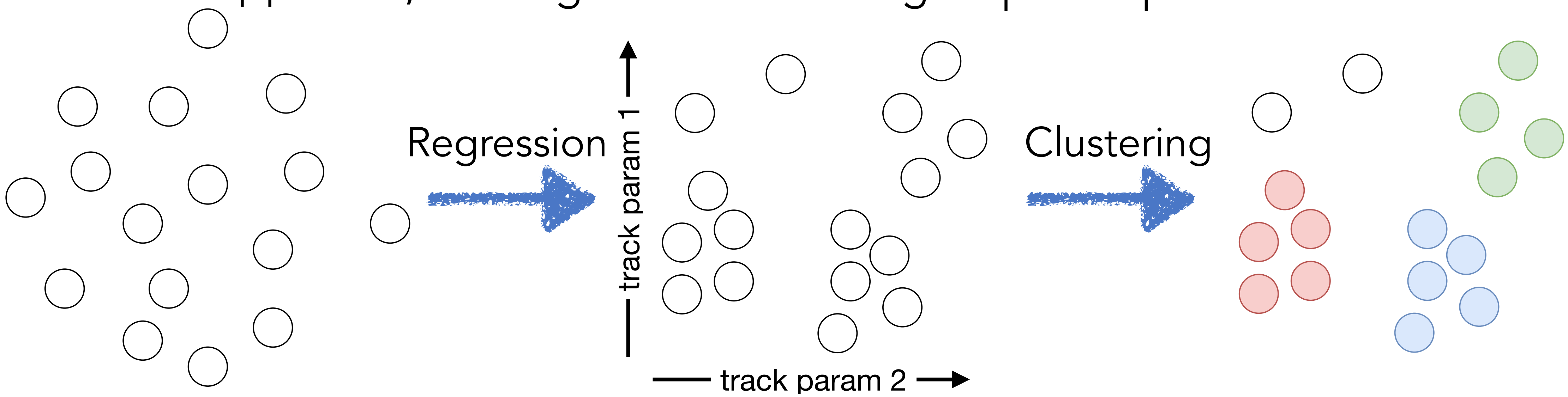
Advantages

- Parallelizable training
- $O(n^2)$ complexity, developments for efficient transformers
- Good at capturing complex nonlinear dynamics

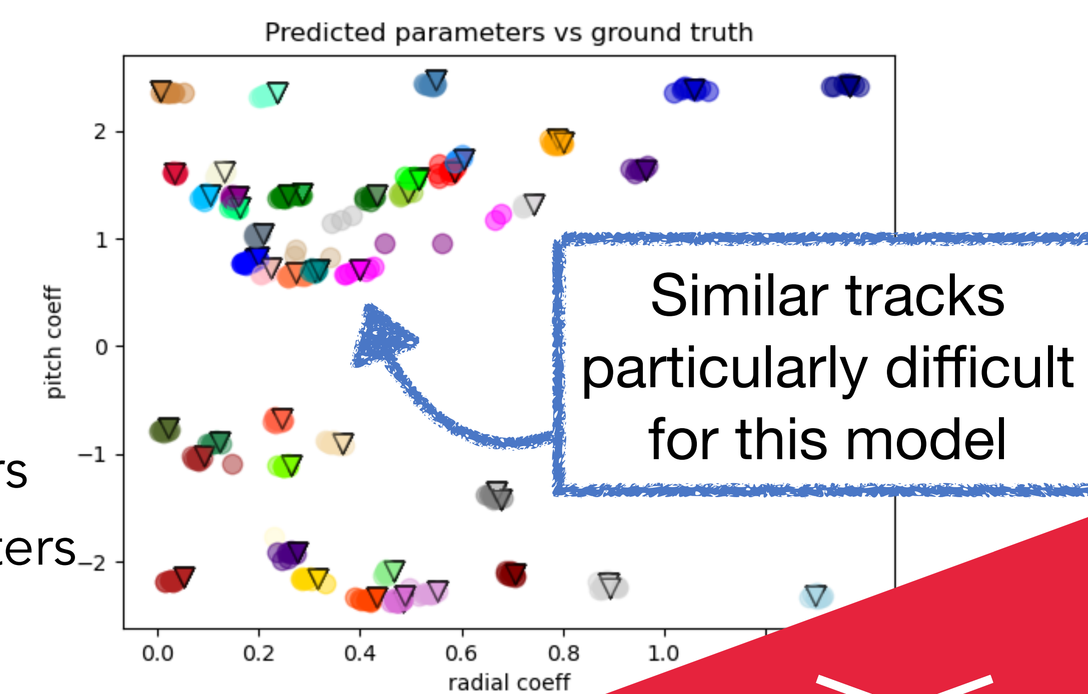
Nikhef

Transformers - Encoder-only Regressor

- This architecture only uses an only an **encoder as a regressor** (sequence to sequence)
  - Regresses track parameters, followed by clustering
  - A one-shot approach, although extra clustering step is required



- Model achieves 90% **accuracy** on trackML data pt > 0.9 GeV and pixel detector only

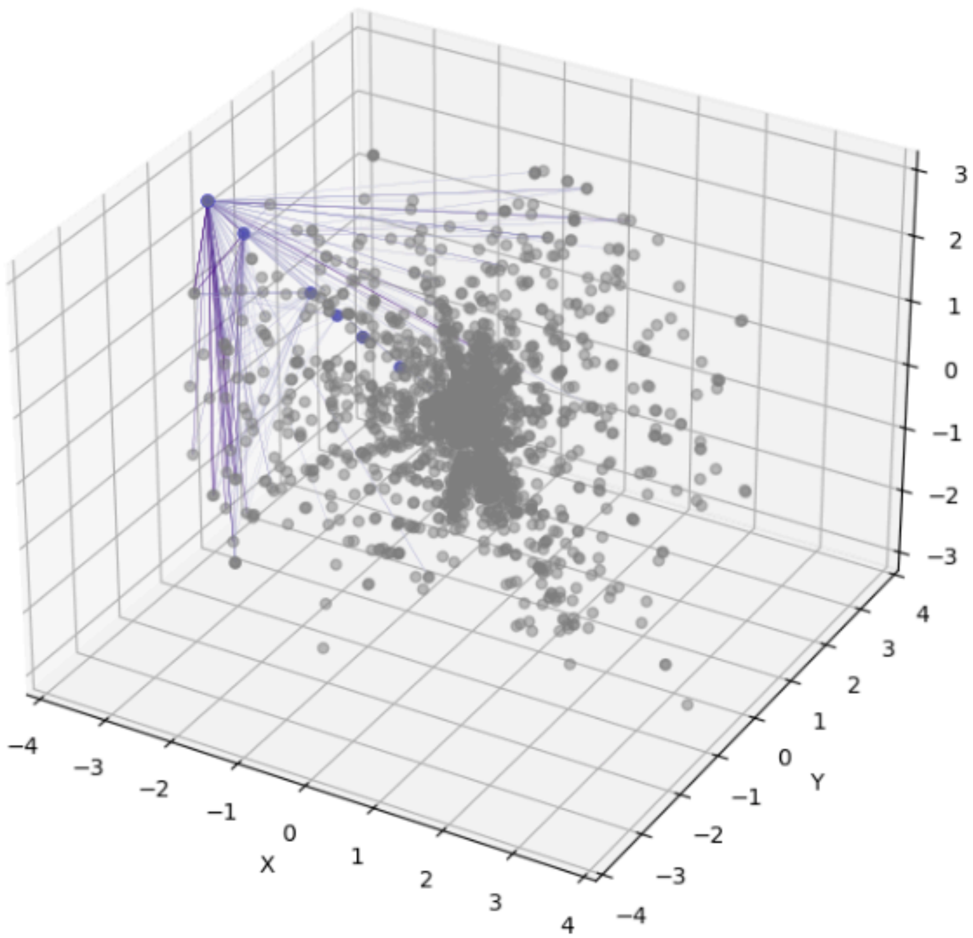


Triangles: True track parameters
Circles: Regressed track parameters
Colors indicate clusters

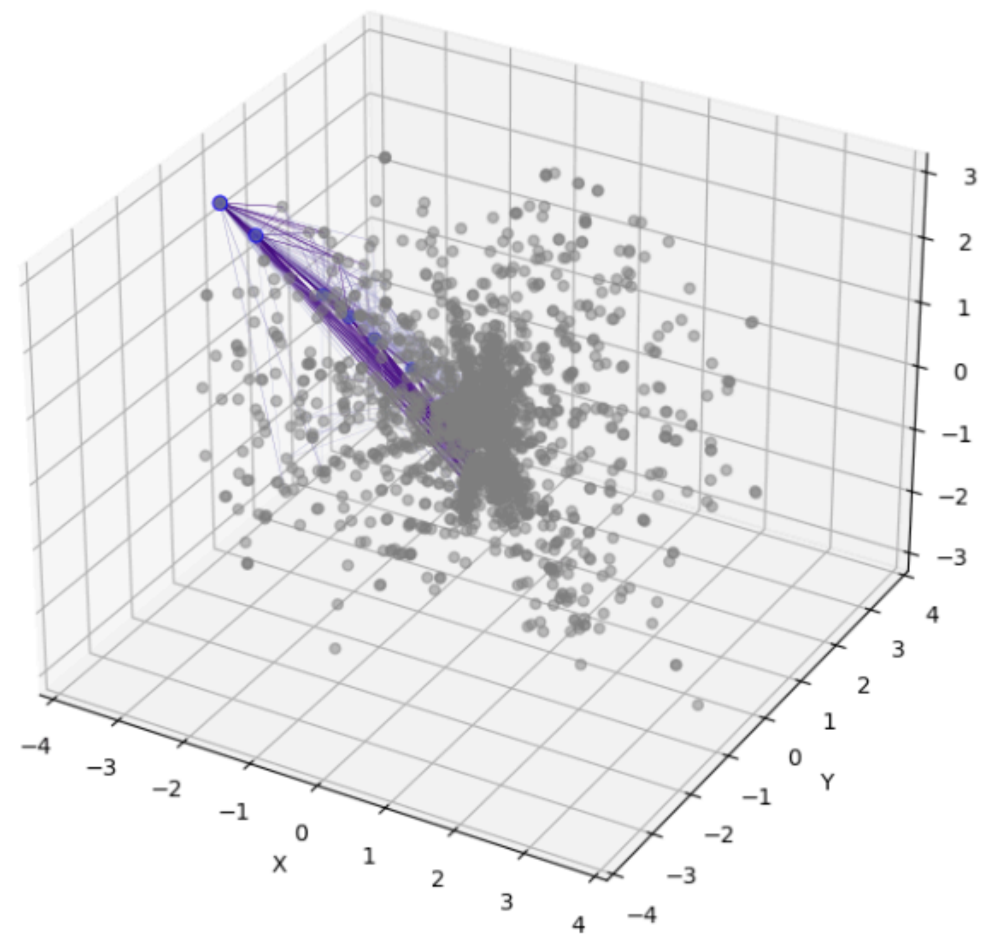
2407.07179

Transformers - Encoder-only Regressor

- What is the model actually learning? We can zoom into and print the attention scores of the model



First attention layer

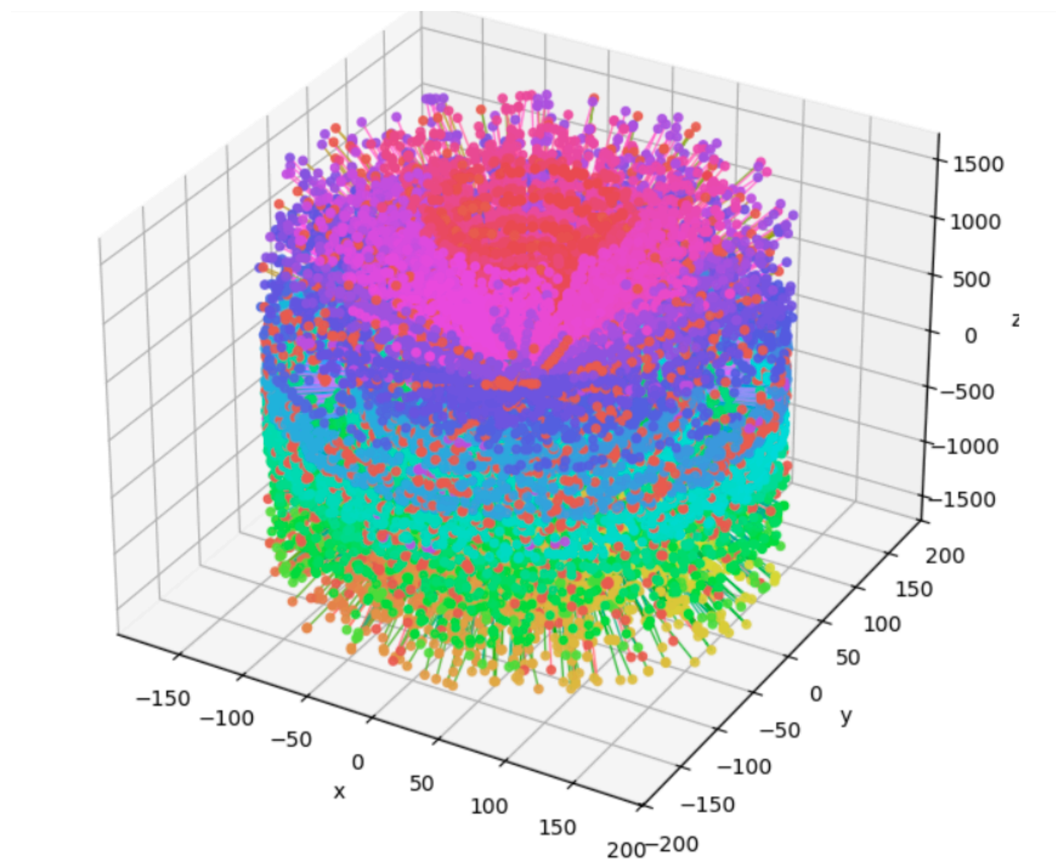


Last attention layer

Nikhef

Current developments

- One of the main challenges is memory, HL-LHC data has O(100k) hits per event, attention matrix 100k x 100k explodes quickly
  - Transformers were originally designed for text processing, with limited max sequence lengths (certainly less than 100k words!)



Nikhef

Current developments

- One of the main challenges is memory, HL-LHC data has O(100k) hits per event, attention matrix 100k x 100k explodes quickly
  - Transformers were originally designed for text processing, with limited max sequence lengths (certainly less than 100k words!)

Maybe we don't have to compute all of the attention scores?

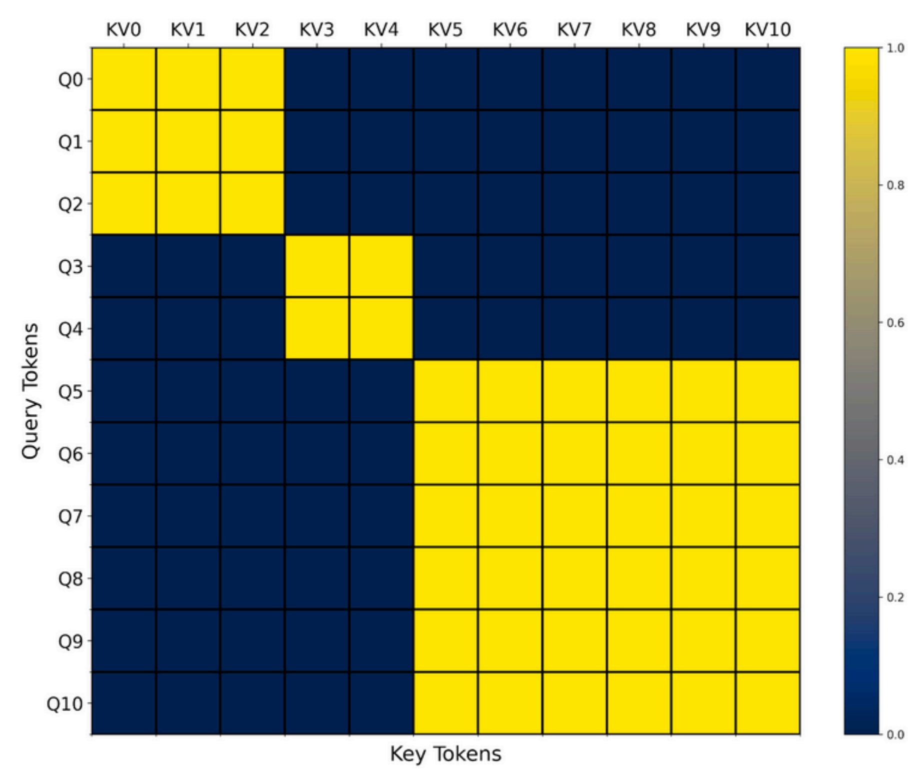
Slav Pshenov

Current developments

- One of the main challenges is memory, HL-LHC data has O(100k) hits per event, attention matrix 100k x 100k explodes quickly
  - Transformers were originally designed for text processing, with limited max sequence lengths (certainly less than 100k words!)

$$\text{Attention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

$$\text{FlexAttention}(\mathbf{Q},\mathbf{K},\mathbf{V}) = \text{softmax}\left(\text{mod}\left(\frac{\mathbf{QK}^T}{\sqrt{d_k}}\right)\right)\mathbf{V}$$

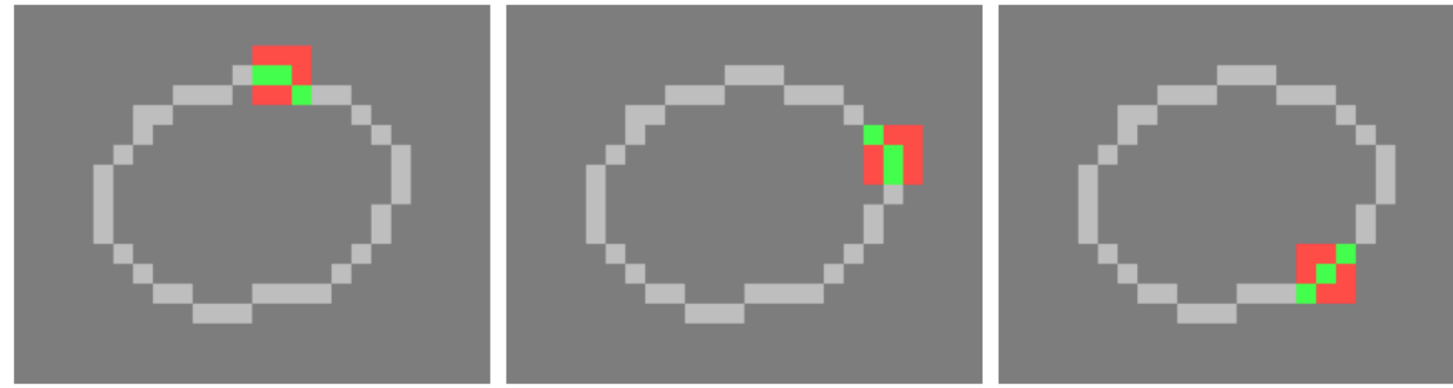


- Currently using preprocessing steps to evaluate what hits are certainly not in the same track, and exclude them from calculation in the attention mechanism

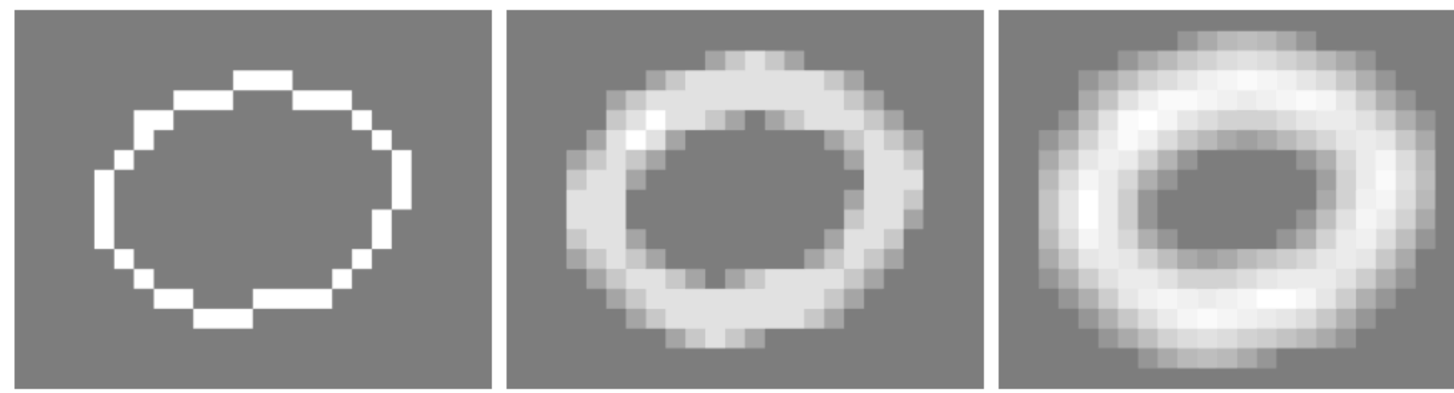
Backup

Submanifold Sparse Convolutions

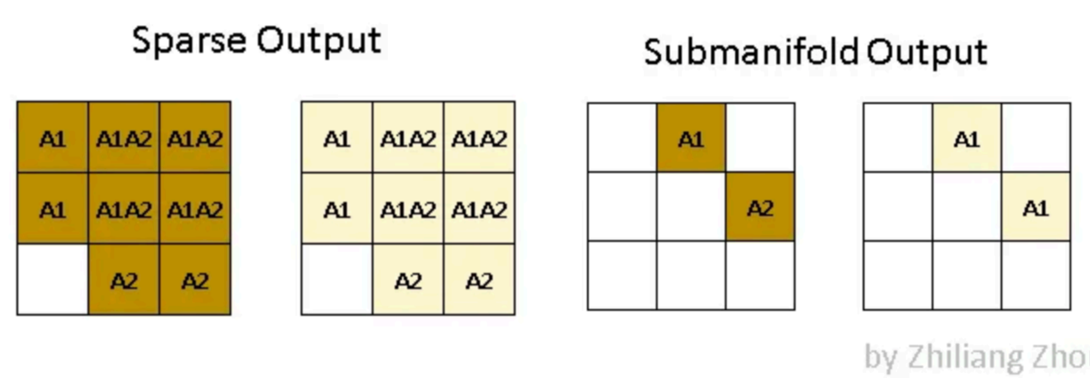
Sparse convolutions only consider input "active sites" and the kernel does process the entire image



This still causes sub manifold dilation



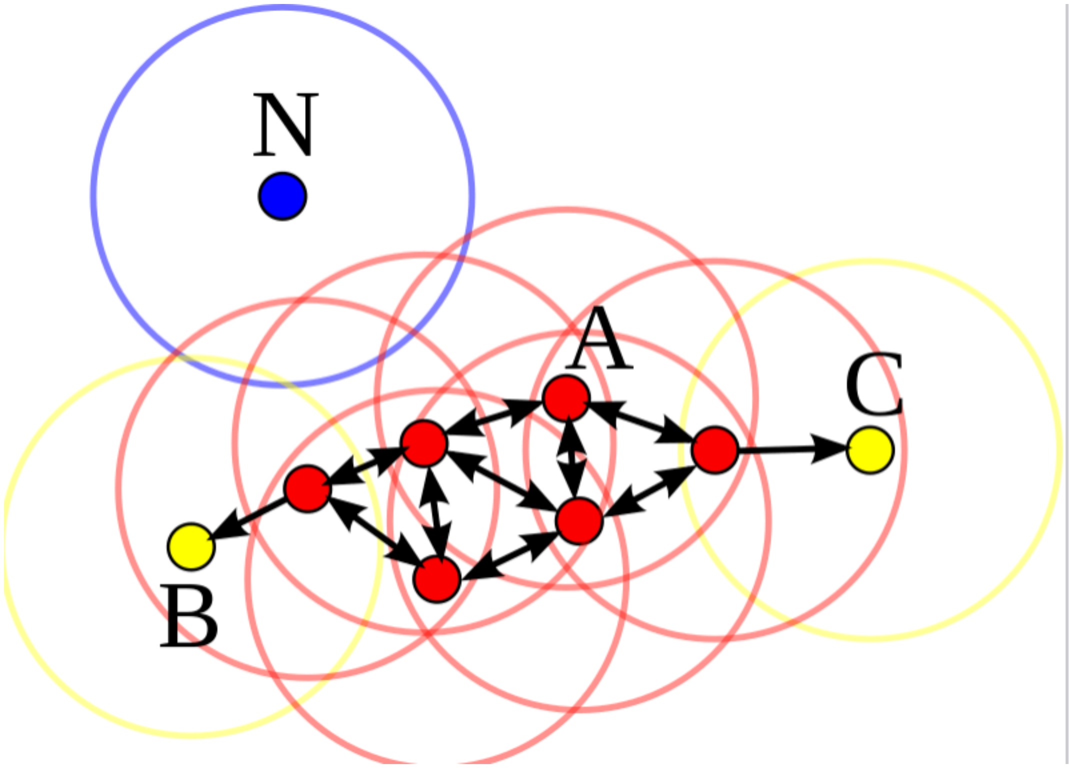
To remedy this, submanifold convolutions are proposed, which only calculate outputs for active input sites, i.e. no dilation



Nikhef

DBSCAN

- A point p is a *core point* if at least minPts points are within distance ε of it (including p).
- A point q is *directly reachable* from p if point q is within distance ε from core point p. Points are only said to be directly reachable from core points.
- A point q is *reachable* from p if there is a path $p_1, \ldots, p_n$ with $p_1 = p$ and $p_n = q$, where each p_{i+1} is directly reachable from p_i. Note that this implies that the initial point and all points on the path must be core points, with the possible exception of q.
- All points not reachable from any other point are *outliers* or *noise points*

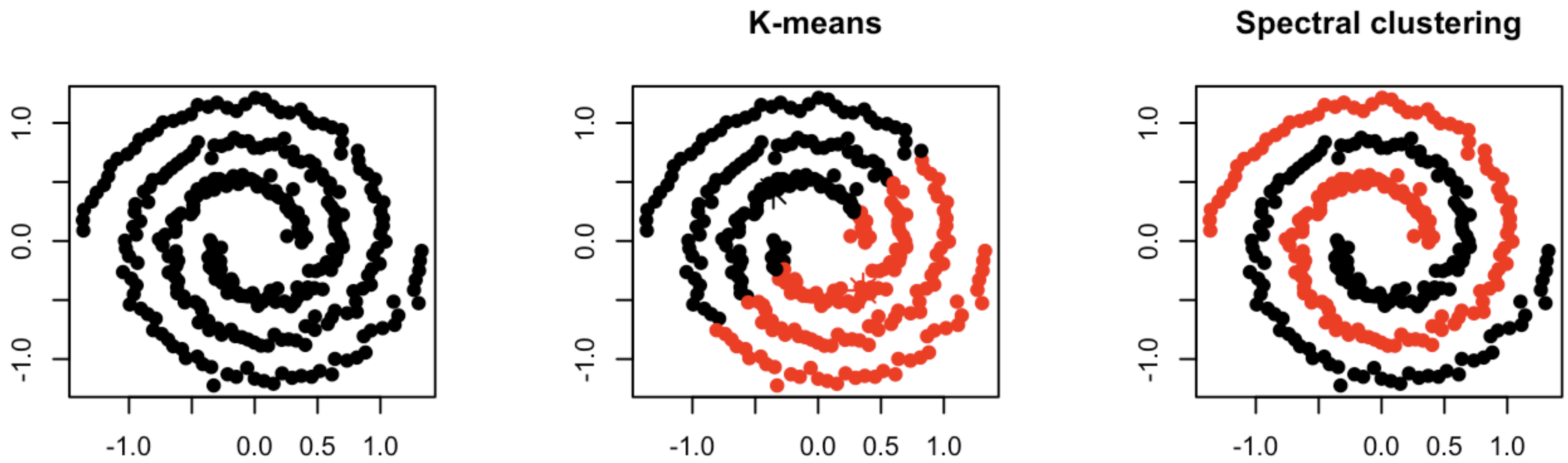


In this diagram, minPts = 4. Point A and the other red points are core points, because the area surrounding these points in an ε radius contain at least 4 points (including the point itself). Because they are all reachable from one another, they form a single cluster. Points B and C are not core points, but are reachable from A (via other core points) and thus belong to the cluster as well. Point N is a noise point that is neither a core point nor directly-reachable.

Spectral Clustering

Clusters uses connectivity between datapoints to create clusters.

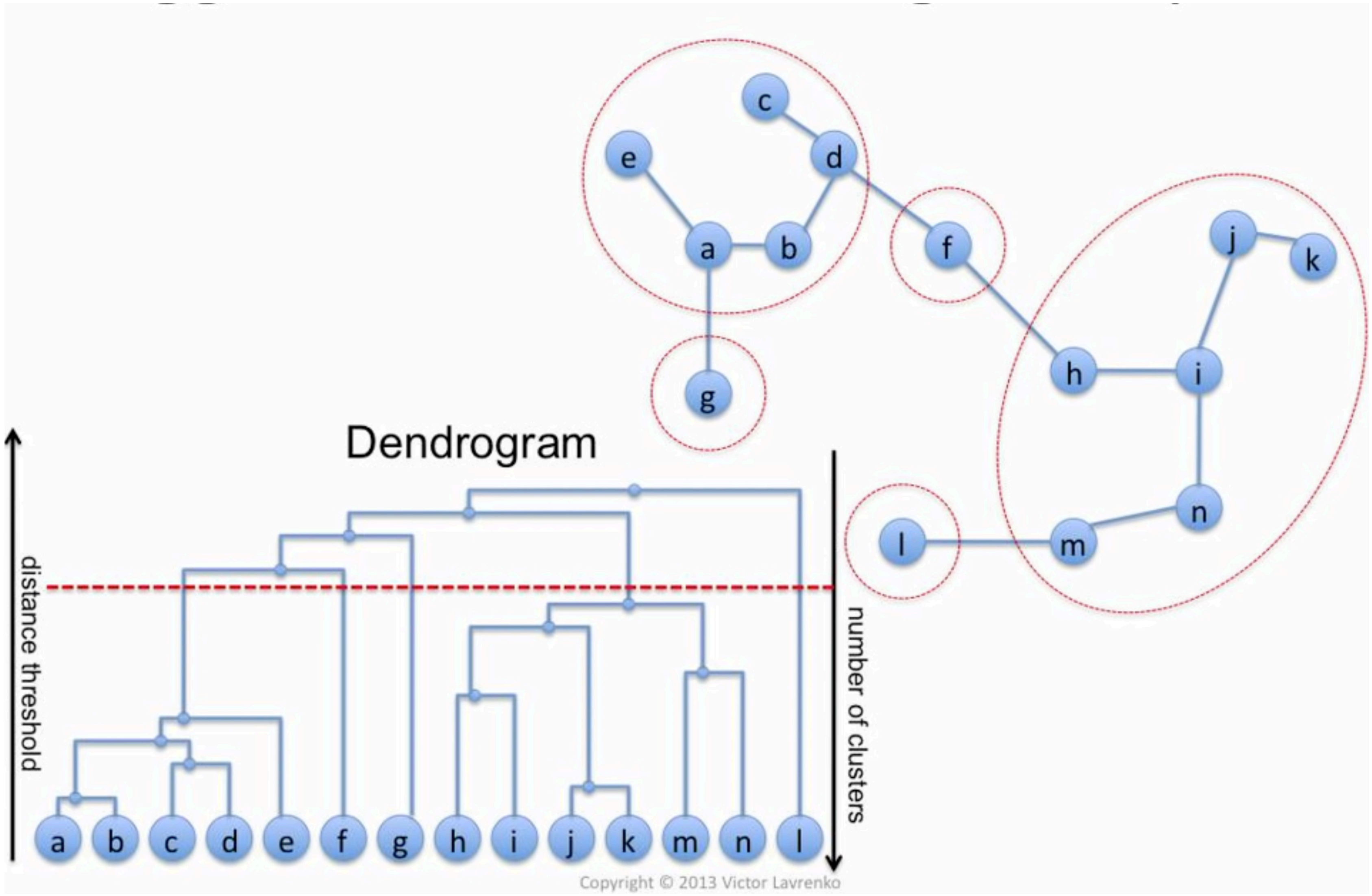
Uses eigenvalues and eigenvectors of the data matrix to forecast the data into lower dimensions space to cluster the data points. Based on the idea of a graph representation of data where the data point are represented as nodes and the similarity between the data points are represented by an edge.



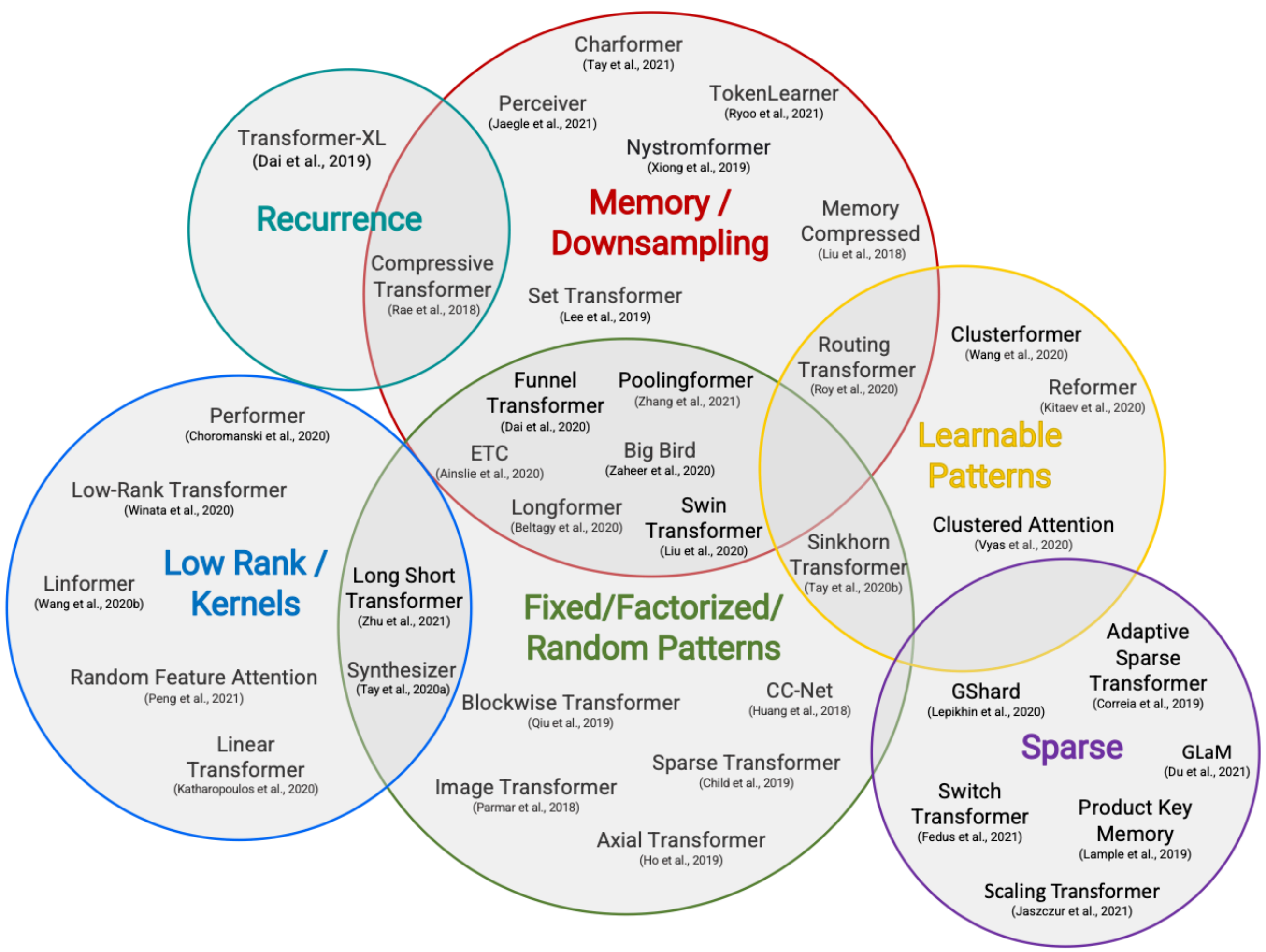
The Davies-Bouldin score is defined as the average similarity measure of each cluster with its most similar cluster, where similarity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score.

Agglomerative Clustering

Agglomerative clustering iteratively adds closest points to clusters, starting with all points as singleton clusters, until all points are connected, at which state a cut in the distance results in a corresponding number of clusters



Efficient Transformers

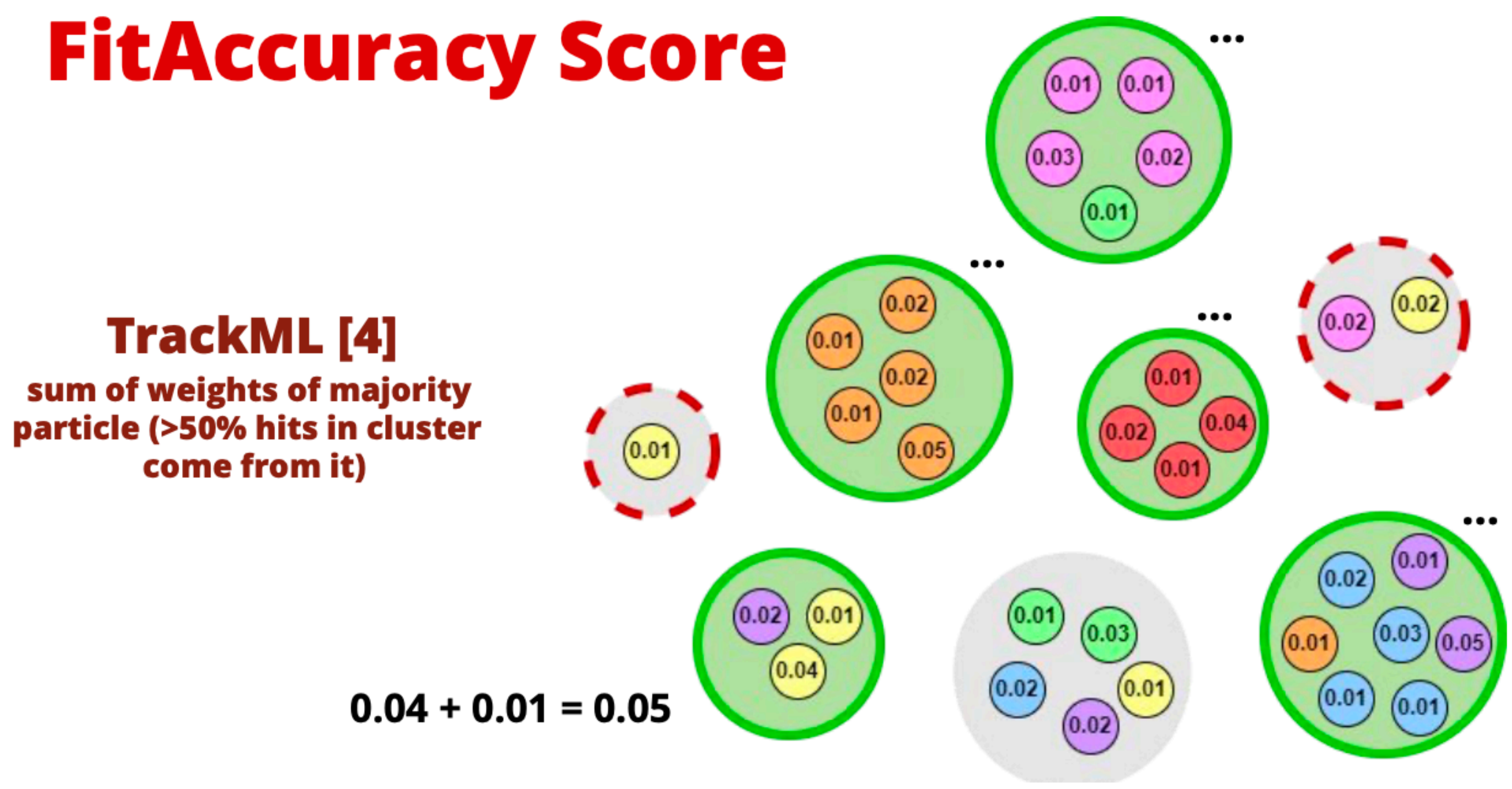


Model / Paper	Complexity	Decode	Class
Memory Compressed (Liu et al., 2018)	$\mathcal{O}(N_c^2)$	✓	FP+M
Image Transformer (Parmar et al., 2018)	$\mathcal{O}(N.m)$	✓	FP
Set Transformer (Lee et al., 2019)	$\mathcal{O}(kN)$	✗	M
Transformer-XL (Dai et al., 2019)	$\mathcal{O}(N^2)$	✓	RC
Sparse Transformer (Child et al., 2019)	$\mathcal{O}(N\sqrt{N})$	✓	FP
Reformer (Kitaev et al., 2020)	$\mathcal{O}(N \log N)$	✓	LP
Routing Transformer (Roy et al., 2020)	$\mathcal{O}(N\sqrt{N})$	✓	LP
Axial Transformer (Ho et al., 2019)	$\mathcal{O}(N\sqrt{N})$	✓	FP
Compressive Transformer (Rae et al., 2020)	$\mathcal{O}(N^2)$	✓	RC
Sinkhorn Transformer (Tay et al., 2020b)	$\mathcal{O}(B^2)$	✓	LP
Longformer (Beltagy et al., 2020)	$\mathcal{O}(n(k+m))$	✓	FP+M
ETC (Ainslie et al., 2020)	$\mathcal{O}(N_g^2 + NN_g)$	✗	FP+M
Synthesizer (Tay et al., 2020a)	$\mathcal{O}(N^2)$	✓	LR+LP
Performer (Choromanski et al., 2020a)	$\mathcal{O}(N)$	✓	KR
Funnel Transformer (Dai et al., 2020)	$\mathcal{O}(N^2)$	✓	FP+DS
Linformer (Wang et al., 2020c)	$\mathcal{O}(N)$	✗	LR
Linear Transformers (Katharopoulos et al., 2020)	$\mathcal{O}(N)$	✓	KR
Big Bird (Zaheer et al., 2020)	$\mathcal{O}(N)$	✗	FP+M
Random Feature Attention (Peng et al., 2021)	$\mathcal{O}(N)$	✓	KR
Long Short Transformers (Zhu et al., 2021)	$\mathcal{O}(kN)$	✓	FP + LR
Poolingformer (Zhang et al., 2021)	$\mathcal{O}(N)$	✗	FP+M
Nyströmformer (Xiong et al., 2021b)	$\mathcal{O}(kN)$	✗	M+DS
Perceiver (Jaegle et al., 2021)	$\mathcal{O}(kN)$	✓	M+DS
Clusterformer (Wang et al., 2020b)	$\mathcal{O}(N \log N)$	✗	LP
Luna (Ma et al., 2021)	$\mathcal{O}(kN)$	✓	M
TokenLearner (Ryoo et al., 2021)	$\mathcal{O}(k^2)$	✗	DS
Adaptive Sparse Transformer (Correia et al., 2019)	$\mathcal{O}(N^2)$	✓	Sparse
Product Key Memory (Lample et al., 2019)	$\mathcal{O}(N^2)$	✓	Sparse
Switch Transformer (Fedus et al., 2021)	$\mathcal{O}(N^2)$	✓	Sparse
ST-MoE (Zoph et al., 2022)	$\mathcal{O}(N^2)$	✓	Sparse
GShard (Lepikhin et al., 2020)	$\mathcal{O}(N^2)$	✓	Sparse
Scaling Transformers (Jaszczur et al., 2021)	$\mathcal{O}(N^2)$	✓	Sparse
GLaM (Du et al., 2021)	$\mathcal{O}(N^2)$	✓	Sparse

Yi Tay, Mostafa Dehghani, Dara Bahri, & Donald Metzler. (2022). Efficient Transformers: A Survey.

Nikhef

Fitaccuracy scores



Yi Tay, Mostafa Dehghani, Dara Bahri, & Donald Metzler. (2022). Efficient Transformers: A Survey.