



# Visual analytics system for understanding DeepRL-based charged particle tracking

Raju Ningappa Mulawade<sup>1</sup> · Christoph Garth<sup>2</sup> · Alexander Wiebel<sup>1</sup> ·  
on behalf of the Bergen pCT collaboration

Accepted: 25 January 2024  
© The Author(s) 2024

## Abstract

In this work, we propose a visual analytics system to analyze deep reinforcement learning (deepRL) models working on the track reconstruction of charged particles in the field of particle physics. The data of these charged particles are in the form of point clouds with high-dimensional features. We use one of the existing post hoc saliency methods of explainable artificial intelligence (XAI) and extend its adaptation to compute saliency attributions for the input data corresponding to the output of the model. Our proposed system helps users to explore these saliency attributions corresponding to the high-dimensional input data of the machine learning model and interpret the decision-making process of the model. In particular, we provide the users with multiple task-oriented components, different types of linked views and interactive tools to analyze the model. We explain how to use the system by outlining a typical user workflow and demonstrate the system's usefulness using several case studies which address specific analysis tasks.

**Keywords** Machine learning · Particle physics · Explainability · High-dimensional data · Visual analytics

## 1 Introduction

In recent years, the use of machine learning (ML) algorithms has witnessed rapid growth in various fields of applications such as computer vision and natural language processing. Particle physics, a branch of physics that deals with the properties, relationships, and interactions of elementary particles has also seen this trend with respect to the track reconstruction of moving charged particles. This further influences the

field of healthcare due to the use of particle therapy in cancer treatment. Understanding the trajectories of the charged particles allows medical experts to focus the energy deposition of these charged particles at a certain depth in the human body where the tumor is located without damaging the neighboring healthy cells. Farrell et al. [1] proposed an image-based method for reconstructing the tracks utilizing image segmentation and image captioning methods. The use of point cloud data of the particle hits, which is transformed into graph data for ML algorithms, has shown promising results in track reconstruction [2–6].

The ability of these ML models to learn nonlinear features enables them to learn complex features from the data, but it also makes the interpretability and the debugging of these models difficult. When it comes to the implementation of these models in the field of healthcare, the interpretability becomes extremely important. This need for the methods to interpret ML models has led to the increase in the amount of research work going into the field of XAI. Some of the significant works in this area are gradient-weighted class activation mapping (Grad-CAM) [7] and SHAP [8]. However, a large portion of the XAI work caters to the analysis of supervised learning algorithms compared to the analysis of reinforcement learning which is another type of ML. A similar trend

---

Christoph Garth and Alexander Wiebel have contributed equally to this work.

---

✉ Raju Ningappa Mulawade  
mulawade@hs-worms.de

Christoph Garth  
garth@rptu.de

Alexander Wiebel  
wiebel@hs-worms.de

<sup>1</sup> Department of Computer Science, Hochschule Worms  
University of Applied Sciences, Erenburgerstr. 19, 67549  
Worms, Rheinland-Pfalz, Germany

<sup>2</sup> Department of Computer Science, RPTU  
Kaiserslautern-Landau, Gottlieb-Daimler-Str., 67663  
Kaiserslautern, Rheinland-Pfalz, Germany

is observed when we look at the data types these XAI methods take into consideration. Most of the XAI methods are implemented on the image and text data compared to point cloud data.

In this work, we focus on the interpretability of deepRL models working on point cloud data. Here, the point cloud data represent the trajectories of charged particles. The task of the deepRL model is to construct the trajectories of these charged particles using the point cloud data provided. We use post hoc saliency mapping methods for the analysis. These are the methods used for interpreting a pre-trained model. In our previous work [9], we proposed the adaptation of two gradient-based saliency methods for the analysis of deepRL models working on high-dimensional feature data [6]. We analyze both of these adaptations and use one of these adaptations for the analysis of the model considered in this work. The output of this method is referred to as the *saliency attributions* in the paper. We interpret the model using these saliency attributions produced. The SmoothGrad [10] method is one of the gradient-based saliency methods of XAI to analyze ML models. Gradients in ML represent the flow of data from the output layer to the input. Analyzing these gradients can provide the model developers and the users with greater insights into the working of the neural networks. Our previous work [9] mainly focused on the preliminary analysis of the track-reconstruction model based on the saliency attributions of a smaller subset of the input features using a number of static plots. In this work, we take the whole set of input features into consideration. This results in a high-dimensional data analysis problem. To address this issue, this paper introduces a visual analytics system leveraging linked plots, interactivity and dimensionality reduction techniques. The introduced system provides users with the ability to interactively analyze the deepRL model in detail. The main contributions of our work are:

- A visual analytics system supporting high-dimensional saliency attribution data for post hoc analysis of a deepRL model working on point cloud data of charged particles
- An extension of an existing saliency mapping methods to high-dimensional input data of a deepRL model
- Description of a user workflow for the mentioned analysis
- Case studies applying the system and the workflow to support verification, debugging and analysis of a deepRL model

The rest of the paper is organized as follows: Section 2 contains the related work in the field of visual analytics (VA) for machine learning algorithms working on high-dimensional point cloud data. Section 3 gives a brief introduction to the data and ML model used in this work. Section 4 lists the requirements based on which the VA system was designed and developed. Section 5 describes the VA system developed

in this work. In Sect. 6, we describe the user workflow for the proposed system. Section 7 contains several case studies demonstrating the usefulness of the system in model analysis. Section 8 contains the qualitative analysis of the system. Section 9 corresponds to the discussion and conclusion parts of the paper.

## 2 Related work

In the last decade, the field of *visualization* has seen a significant rise in the amount of research work dedicated to the area of VA for understanding and explaining machine learning algorithms [11]. Many VA systems have been developed to analyze and understand the input data, the training process and the ML model after the completion of the training process. XAI-based VA systems can be categorized based on the type of ML models (supervised, unsupervised and RL) being interpreted and the input data (image, text, depth images, point clouds, etc.) taken into consideration. Some examples of XAI-based VA systems working on image data have been presented by Xuan et al. [12] and Hohman et al. [13]. However, the ML models taken into consideration in these works belong to the supervised learning type. When we look for the combination of RL and point cloud data in the VA field, we observe that it is difficult to find any literature that addresses this specific combination. One of the primary reasons for this lack of implementations is the limited amount of work in the development of point cloud-based XAI methods. In the following part of this section, we review the XAI methods found in the current literature for point cloud data.

The methods proposed by Zheng et al. [14] and Zhang et al. [15] are two of the earliest works toward the interpretability of ML models working on point cloud data. Zheng et al. [14] proposed a method to generate a saliency map for the input point cloud data of a classification-oriented ML model. Here, the point clouds represent 3D objects such as tables, humans and airplanes and the ML model tries to recognize these point clouds. The method assigns a score to each point in the input data which reflects the contribution of the corresponding point to the model-recognition loss. They further aggregate the highly scored points to highlight important segments in the input point cloud data. While the very basics of the presentation of the importance is similar to some illustrations in our work, the very different meaning of the point clouds in our data results in additional visual representations which are rather different.

Zhang et al. [15] proposed PointHop, an explainable machine learning method for ML models performing point cloud classification where point clouds represent 3D objects. It consists of two stages. The first stage corresponds to the local-to-global attribute building through multi-hop information exchange and the second stage corresponds to classifi-

cation and ensembles where classifiers such as the support vector machine (SVM) and the random forest (RF) classifiers are adapted to perform classification task on the feature vector obtained from the first stage. This explainability method corresponds to the type of XAI methods used in the preprocessing stage of the ML pipeline. Tan et al. [16] proposed a point cloud-applicable explainability approach based on a local surrogate model-based method to indicate which components in a point cloud representing a 3D object contribute to the classification. This is based on LIME [17] which is an explanation technique that explains the predictions of a given classifier by learning an interpretable model locally around the prediction.

In recent years, gradient-based saliency methods of XAI have become a popular tool to analyze a trained model by highlighting the input features that influence the output of the model. These methods are categorized into post hoc interpretability methods in XAI. Some of the prominent gradient-based methods are guided backpropagation [18], deconvolution [19], integrated gradients [20], SmoothGrad [10], class activation mapping (CAM) [21], gradient-weighted class activation mapping (Grad-CAM) [7] and SHAP [8]. Some of the authors have attempted to extend/adapt these gradient-based saliency methods originally developed for image data to point cloud data. Gupta et al. [22] extended vanilla gradients, guided backpropagation and integrated gradients to explain classification networks working on point cloud data as well as voxel data representing 3D objects. The clearest results were obtained for the integrated gradients method which highlighted the corners and edges as important features and flat surfaces as less important. Matrone et al. [23] proposed BubbleX, a multi-modal fusion framework to learn 3D point features. The point cloud data considered in this work represents 3D objects. They extend Grad-CAM for point cloud to generate saliency maps and interpret the classification network. Schwegler et al. [24] adapted the integrated gradients method to a large-scale urban point cloud dataset to get better insight into how the decision-making process takes place in the model and what influence the input features have on the output prediction.

However, none of the point cloud-based XAI literature mentioned so far deals with the explainability of RL models. In addition, the point cloud data considered in this work are quite distinct as it does not represent models/objects in 3D but it represents particle hits and tracking these particle hits leads to forming tracks. In our previous work [9], we tried to address this issue by proposing XAI methods to analyze a deepRL model working on point cloud data. We proposed visual analysis methods adapting integrated gradients and SmoothGrad methods to interpret a DeepRL model working on point cloud data. Both integrated gradients and SmoothGrad methods produced promising results compared

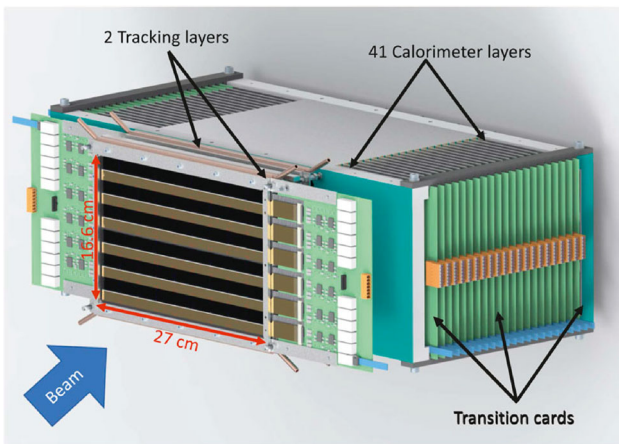
to vanilla gradients by highlighting some of the very important features in the input data. However, we considered only a small subset of the saliency attributions corresponding to the input features for the analysis. In this work, we look into the integrated gradients and SmoothGrad methods adapted for point cloud data in our previous work [9] and select the best method for analyzing the deepRL model working on the track reconstruction task. In addition, we extend the selected method by adapting it to the other two input data (the model takes in three input data). We address the issue of utilizing only a small subset of saliency attributions for deepRL model's analysis by utilizing multiple plots and linked plots to analyze the saliency attributions corresponding to *all* the input features. To further address the issue of visualization of high-dimensional data, we leverage one of the most adopted dimensionality reduction methods, t-distributed stochastic neighbor embedding (t-SNE) [25], which performs better than other dimensionality reduction methods over a set of different data types as demonstrated in [26].

### 3 Data and model

In this section, we provide a brief introduction to the data and the pre-trained model considered for analysis in this work. For a more detailed explanation about the data, model and training process, we refer the readers to the work of Kortus et al. [6].

#### 3.1 Data

The data considered in this work are a point cloud containing the information of particle *hits* in a proton computed tomography (pCT) scanner [27]. A prototype pCT scanner with a high granularity digital tracking calorimeter (DTC) is being designed and built by the *Bergen pCT Collaboration* [28]. The DTC is a multilayer structure consisting of two tracking layers followed by multiple detector/absorber sandwich layers referred to as the calorimeter layers (see Fig. 1) with each layer containing multiple strips of ALICE pixel detector (ALPIDE) silicon sensors [29]. These strips capture the high-energy protons passing through the patient in an imaging setup, providing the users with both spatial information of each proton hit and the corresponding energy deposited by the proton at this location. Hundreds of trajectories, or tracks, of the protons that enter and traverse the detectors must be fully and correctly reconstructed from the sensor hits. Using these discrete tracks, with a corresponding energy loss, the user can reconstruct in in-vivo, patient-specific 2D images of the water-equivalent path length or 3D images of the proton stopping power. These images can be used for increased treatment accuracy for planning or monitoring of proton therapy for cancer. Note that a high hit density (i.e.,



**Fig. 1** The general structure of the Bergen pCT system. Image from Alme et al. [28] (CC BY)

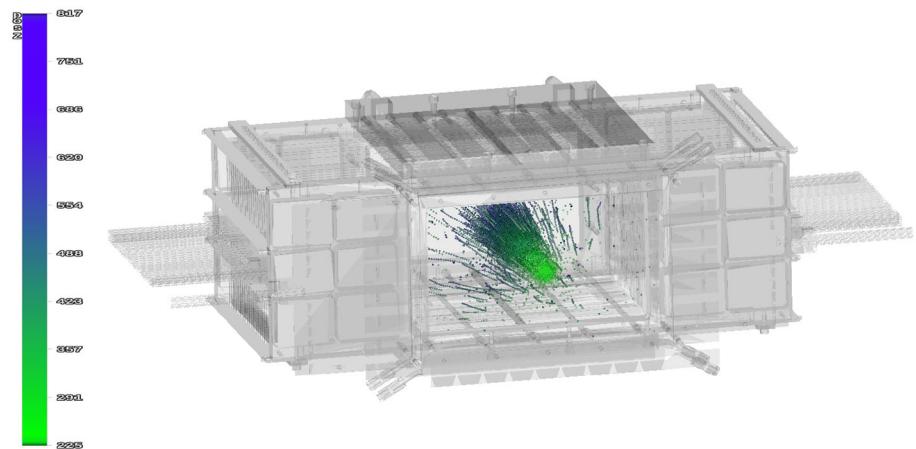
proton intensity) drastically increases the complexity of the track-reconstruction process.

To be able to understand the comparisons we make throughout the paper it should be noted that the point cloud data considered in this paper comes from a simulation of the actual detector [30]. The actual detector is in the developmental phase. While in the future the model will be applied to real detector data, the virtual detector data (hits and their properties) obtained from the simulation are helpful for

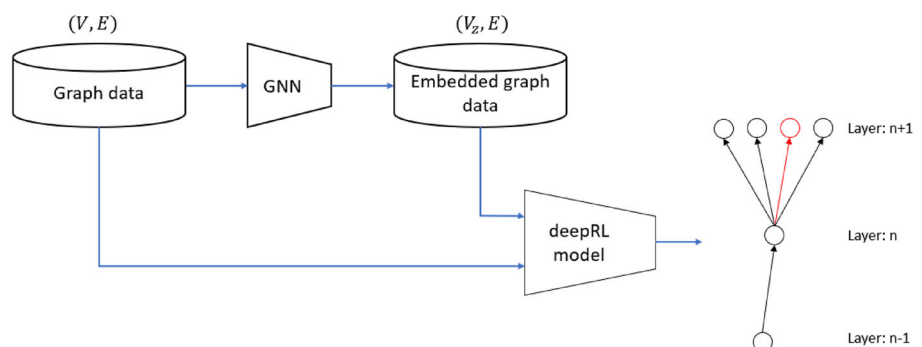
developing the tracking model. Additional information such as the ground truth provided by the simulation makes the model analysis part much easier. An example of the simulation data generated for the protons is shown in Fig. 2. Given this point cloud data, the task of the deepRL model is to construct the trajectories or paths of the protons. The *hits* form a layered point cloud (due to the use of multilayer DTC to record data) which is converted into graph data  $G = (V, E)$  where the nodes ( $V$ ) of the graph are defined by the centroids of the *hits* and the edges ( $E$ ) connect existing neighboring hits over subsequent layers in a direction opposite to the path of the particles [6].

The nodes ( $V$ ) of the graph data  $G$  are embedded to  $V_z$  using a graph neural network (GNN) to capture the structural information in the graph data [6]. In particular, the GNN attempts to capture relations between the nodes in subsequent detector layers. These embedded data  $G_z = GNN(\vec{v}) = (V_z, E)$  are used in defining the input data for the deepRL model as shown in Fig. 3. Each node  $\vec{v}_i \in V$  is parameterized by the energy deposition  $\Delta E_i$ , the  $x$  and  $y$  coordinates of the pixel position  $(x_i, y_i)$  and the one-hot encoded 50-dimensional vector  $z\_enc_i$  representing the index of the detector layer. Thus,  $\vec{v}_i$  is represented by 53-value vector. Each edge  $\vec{e}_{jk} \in E$  is given in spherical coordinates as radius  $r_{jk}$ , azimuthal angle  $\theta_{jk}$  and polar angle  $\phi_{jk}$  describing the connection of a transition hypothesis as its parameters.

**Fig. 2** Point cloud data of the simulated protons. This rendering shows hits of a subset (2000) of simulated protons entering the calorimeter layers. The hits are color-coded according to the distance from their entry position into the detector ( $z$ -coordinate)



**Fig. 3** Overview of the use of graph data and the embedded nodes as input for the track-reconstruction process



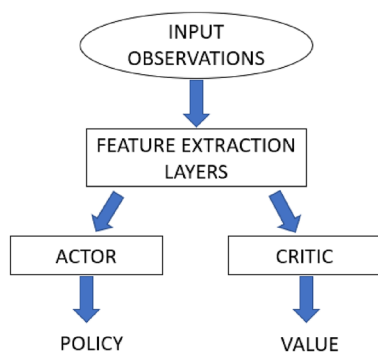


Fig. 4 Actor-critic architecture of RL

The polar angle  $\phi_{jk}$  corresponds to the deflection angle of particles.

In addition to the virtual detector data, the simulation can provide the actual paths  $p^l$  of the particles, i.e., the correct connections of the hits. It is important to note that the model does *not* have access to the paths  $p^l$ , neither during training nor during inference, because the paths are what the model should predict. Instead, the paths  $p^l$  serve as *ground truth* or *label* to check how good the predicted tracks  $p$  resemble the goal.

### 3.2 Model

Reinforcement learning is a branch of machine learning where the learning agent learns to perform a set of sequential actions by interacting with the environment. The learning agent tries to maximize a reward function by trying out multiple actions and discovering the set of actions that yield the maximum reward. This characteristic of RL distinguishes it from the other two branches of ML referred to as *supervised learning* and *unsupervised learning*. In this work, we try to explain the decision-making process of a pre-trained deepRL model (based on the model proposed in [6]) working on the track-reconstruction task of the charged particles. The task of the model is to use the point cloud data of a set of charged particles captured by the detector layers and construct their trajectories/path/track by tracking each of these charged particles along these detector layers. The model has an actor-critic architecture as shown in Fig. 4.

The actor-critic model consists of three parts:

- *Feature extraction layers*: This is the common network branch that holds the shared parameters of the actor-critic model. The features learned from the input data instance by these layers are utilized by both policy and value networks.
- *Policy network*: Policy ( $\pi_\theta$ ) refers to the action that a learning agent takes at a given state. Policy network yields for each input data instance associated with a state  $s_t$  a probability distribution over all the possible actions ( $a_t^{(i)}$ ).

In our case, the possible actions refer to the number of hits belonging to a particular layer whose features are provided as the input for the actor-critic model. The action with the highest probability is chosen as the best possible action.

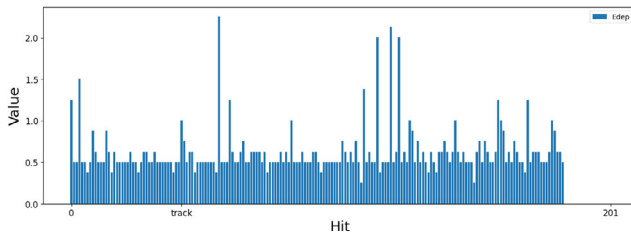
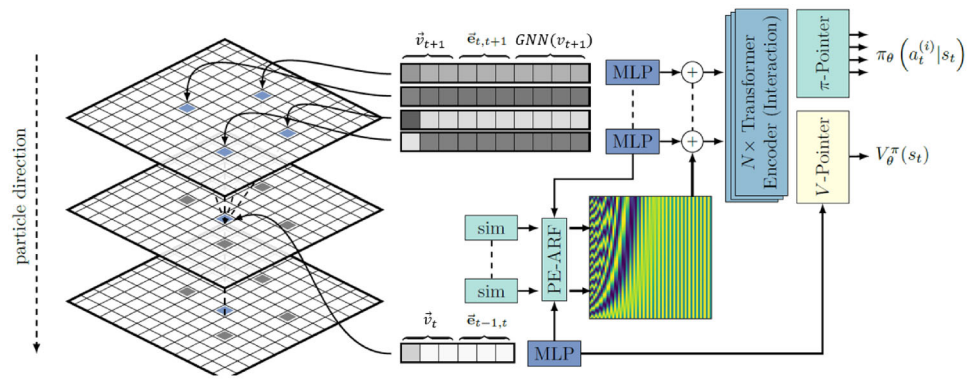
- *Value network*: Estimates the value for the current state ( $s_t$ ). Here, the value ( $V_\theta^\pi$ ) refers to the total amount of reward that the learning agent can expect to accumulate over the feature starting from this current state.

The shared network architecture of the pre-trained deepRL model considered for analysis in this work is shown in Fig. 5. It consists of multiple multilayer perceptron (MLP) layers, N stacked Transformer multilayer perceptron (MLP) layers, N stacked Transformer encoder and two separate branches of attention-based decoder for value and policy estimate, respectively [6]. To track a hit  $H_t^p$  from detector layer  $t$  to  $t + 1$ , the actor-critic model is provided with *action features*, *observation features* and positional encoding with adaptive receptive field (PE-ARF) based on cosine similarities as the input. The *action features* refer to the features of the hits belonging to detector layer  $t + 1$  in  $G$  and  $G_z$  that are taken into consideration by the model and the features of the edges in  $G$  that connect these hits with  $H_t^p$ . This is represented by  $(\vec{v}_{t+1}, \vec{e}_{t,t+1}, GNN(v_{t+1}))$  in Fig. 5. The size of *action features* input is  $(m, 120)$  where  $m$  is the number of hits considered in detector layer  $(t + 1)$  ( $m = 4$  in Fig. 5) and 120 corresponds to the number of features representing each of these hits and their corresponding edges connecting them to the hit being tracked from detector layer  $t$ . The *observation features* (represented by  $(\vec{v}_t, \vec{e}_{t-1,t})$  in Fig. 5) refer to the features of  $H_t^p$  in  $G$  along with the features of the edge connecting  $H_t^p$  to  $H_{t-1}^p$  in  $G$  which is a part of the reconstructed track  $p$ . The size of *observation features* is  $(1, 56)$  where 56 represents the number of features mentioned before. The policy network of the model outputs the probability distribution for the input hits belonging to  $t + 1$  and the hit with the highest probability is chosen as the next hit in the track  $p$ . The direction of the track reconstruction process is opposite to the path of the particles.

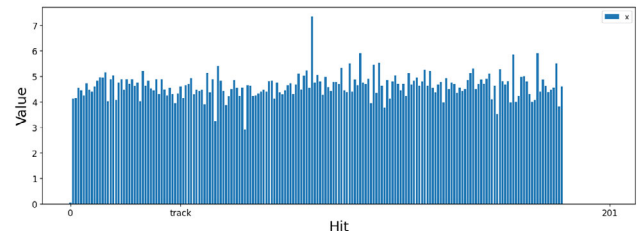
### 3.3 Saliency mapping method

Gradient-based saliency mapping methods generate saliency maps using gradients. Gradients indicate the change in output value corresponding to a particular class of interest with respect to the change in input data. In our previous work [9], we adapted two gradient-based saliency methods to explain the decision-making process of the point cloud-based deepRL model taken into consideration. We demonstrated the usefulness of integrated gradients and SmoothGrad methods over vanilla gradients using the saliency maps generated. However, in this section, we look into the results of these

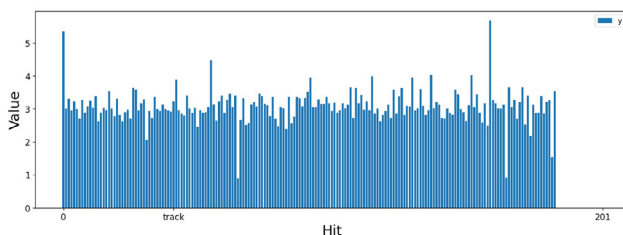
**Fig. 5** Network architecture of the deepRL model from Kortus et al. [6] (CC BY 4.0) with minor modifications in the notations to suit our description



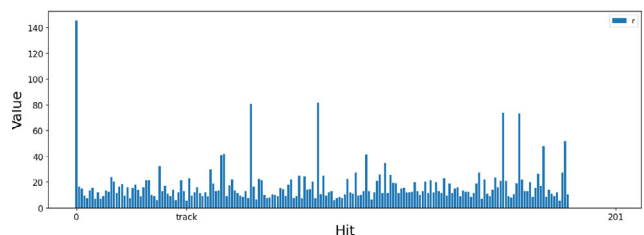
(a) Energy deposition



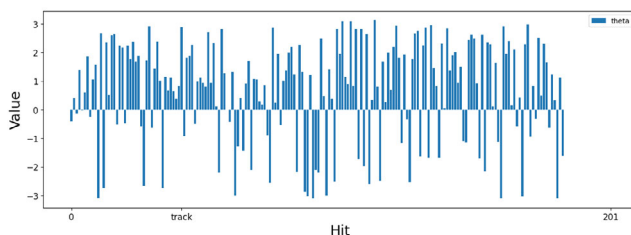
(b) x



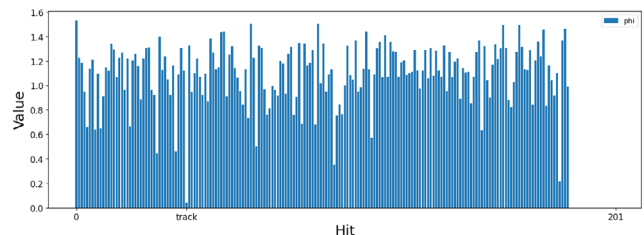
(c) y



(d) r



(e)  $\theta$



(f)  $\phi$

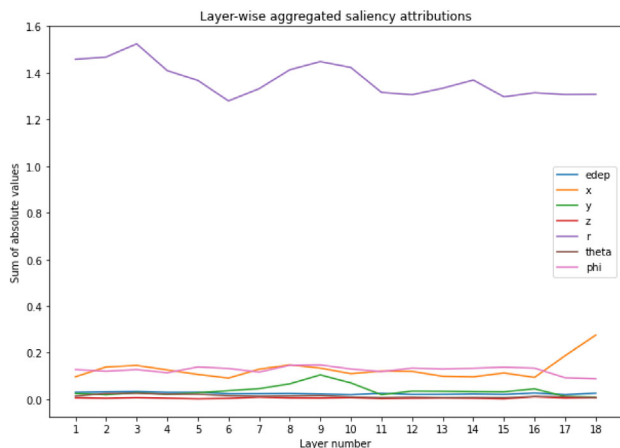
**Fig. 6** Input values of some of the features belonging to the *action features* of an example detector layer (*x*-axis refers to the hits in the detector layer to which the feature belongs and *y*-axis corresponds to the value of the input feature). The index marked *track* on the *x*-axis refers to the hit chosen by the model

adaptations of integrated gradients and SmoothGrad methods to determine the best method for model analysis.

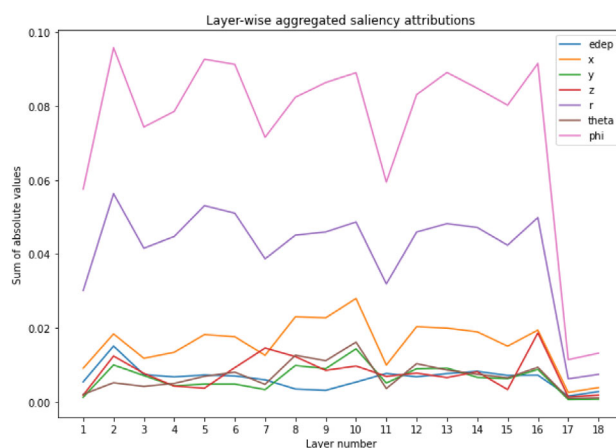
For a deep neural network represented by a function  $F : \mathbb{R}^n \rightarrow [0, 1]$ , input  $x \in \mathbb{R}^n$  and its corresponding baseline  $x' \in \mathbb{R}^n$ , integrated gradients are computed as:

$$M_{intGrad}(x) ::= (x - x') * \int_{\alpha=0}^1 \frac{\partial F(x' + \alpha * (x - x'))}{\partial x} d\alpha \quad (1)$$

From the above equation, we observed that the input values have a higher influence on the gradients integrated for the intermediate inputs when the baseline  $x' \approx 0$  leading to  $(x - x') \approx x$ . We used 2% noise generated for each input variable in *action features* as the baseline resulting in  $(x - x') \approx x$ . This indicates that the saliency map generated by the integrated gradients method depends on the input values to a great extent. Figure 6 shows an example of input values of six features of each node considered in *action fea-*



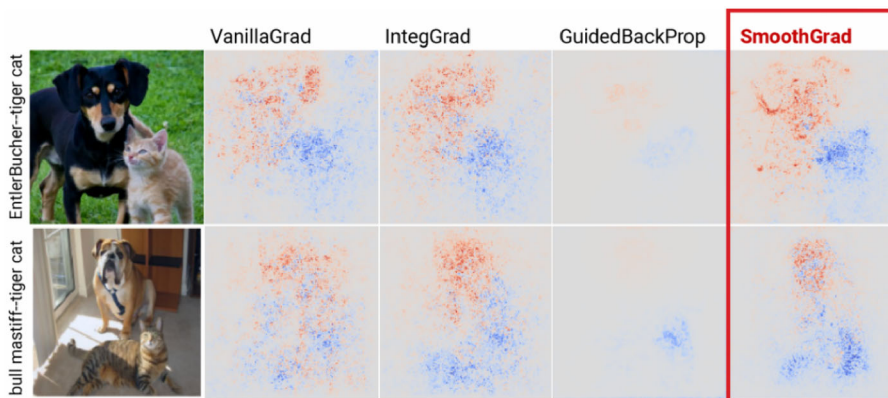
(a) Example track 1: Aggregated integrated gradients



(b) Example track 1: Aggregated SmoothGrad gradients

**Fig. 7** Aggregated saliency attributions corresponding to seven of the *action features* for two example tracks taken from our previous work [9] (CC BY). The saliency attributions are aggregated by adding the magnitude of saliency attributions for each detector layer

**Fig. 8** Comparison of saliency maps produced for two input images (on the left) with the gradient values normalized to be between [0; 1]. The values are plotted using a diverging color map [−1; 0; 1] ↔ [blue; gray; red]. Each method is represented in columns. Image from Smilkov et al. [10]



tures for a particular detector layer. We observe that the values corresponding to the polar radius,  $r$  (Fig. 6d) have significantly higher magnitude compared to the other five features. This is reflected in the saliency attributions produced by the integrated gradients in [9] where polar radius,  $r$  is shown to be highly influential compared to deflection angle,  $\phi$ . This is shown in Fig. 7 where the saliency attributions corresponding to each of the seven features taken into consideration are aggregated for each detector layer.

However, we consider deflection angle,  $\phi$  to be the most important feature as it directly defines the trajectory of the particle. This information is critical in tracking a charged particle and constructing the corresponding track.

The adapted SmoothGrad method, however, provides better results as it relies completely on the gradient values. It indicates the high influence of deflection angle,  $\phi$  followed by polar radius,  $r$  (shown in Fig. 7). The SmoothGrad method sharpens the gradient-based saliency maps by reducing the

noise in these saliency maps. For a given input  $I$  and corresponding output class score  $S_c$  of class  $c \in C$ , the gradients of class score  $S_c$  with respect to input  $I$ ,  $M_c(I)$  are computed as  $\partial S_c / \partial I$ . The SmoothGrad saliency map  $M_{SG}$  for input  $I$  and output class score  $S_c$  is computed as:

$$M_{SG} = \frac{1}{m} \sum_1^m (M_c(I + N(0, \sigma^2))) \tag{2}$$

where  $N(0, \sigma^2)$  represents the Gaussian noise with given standard deviation  $\sigma$  and  $m$  represents the number of noisy samples of input  $I$  generated. The saliency map,  $M_{SG}$  has the same dimension as input,  $I$ . Figure 8 shows the comparison of saliency maps generated by SmoothGrad and three other saliency mapping methods. We use the adaptation of this method from our previous work [9] to analyze the point cloud-based deepRL model in this work. We extend the adaptation of this method to compute saliency attributions of

remaining input features consisting of *observation features* and *PE-ARF*. We compute  $M_{SG}^{act}(policy)$ ,  $M_{SG}^{obs}(policy)$  and  $M_{SG}^{PE-ARF}(policy)$  for *action features*, *observation features* and *PE-ARF* input data, respectively, taking *policy* output into consideration. In our previous work, we adapted the SmoothGrad method for computing saliency attributions with respect to the best possible action of *policy* output. In addition, we extend the adaptation to compute saliency attributions corresponding to the *value* output ( $M_{SG}^{act}(value)$ ,  $M_{SG}^{obs}(value)$  and  $M_{SG}^{PE-ARF}(value)$ ) to allow the users to understand the model through its *value* output.

## 4 Requirements analysis

We developed the VA system based on the user requirements acquired during the discussion with around ten experts belonging to the fields of physics, ML and visualization from the pCT collaboration. Through the use of this system, the users from physics intend to understand the working of the model and verify if the model is paying attention to the important features in the input data during the decision-making process. They also intend to understand the behavior of the model when it fails to reconstruct the tracks correctly. The users from the ML field which also include model developers, want to understand and debug the model and they are looking for assistance in feature engineering during the developmental phase. Thus, through our system, we try to address the following requirements:

*R1: Provide an overview of the reconstructed tracks with the possibility to select a track interactively:* Since we try to analyze the reconstruction process of multiple tracks, it is useful for the users to have an overview of all the tracks reconstructed. The users can examine the reconstructed tracks and choose the track of their interest for the analysis through a mouse-click event.

*R2: Visualize saliency attributions corresponding to multiple input features:* When analyzing the reconstruction process of a track, it is important to take a larger set of saliency attributions into consideration and provide the users with an overview and the option to select a subset of the data for analysis. For example, *action features* input is represented as  $(v_{t+1}, \vec{e}_{t,t+1}, GNN(v_{t+1}))$  with  $v_{t+1}$  consisting of 53 features,  $\vec{e}_{t,t+1}$  consisting of three features, and  $GNN(v_{t+1})$  consisting of 64 features for each hit. Therefore, each hit considered in *action features* along with the edge connecting it to the hit being tracked in layer  $t$  is represented by 120 features. It is important to consider the saliency attributions of all these 120 features of each hit when analyzing the model. Considering only a small subset of saliency attributions is insufficient.

*R3: Allow interactive exploration of saliency attributions:* Interactive tools are one of the important parts of a VA system. They help users in analyzing the data in detail. Since we deal with multiple hits and features, providing users with interactive tools to explore the data would improve the analysis process.

*R4: Analysis of a single step of the reconstruction process:* Since the reconstruction of each track involves multiple steps, the users found it important to analyze these multiple steps individually. This is important because the analysis of individual steps allows the users to understand the decision-making process of the model at different detector layers such as the tracking layers and calorimeter layers.

*R5: Analysis of detector layers of interest in incorrectly reconstructed tracks:* Understanding the decision-making process of the deepRL model at instances (or at detector layers) where the reconstruction goes wrong is important for understanding and debugging the model. It can also give us an insight into the input examples where the model finds it difficult to reconstruct tracks. Thus, many users recommended having a separate part in the VA system to analyze only these special detector layers.

## 5 Visual analytics system

Taking the requirements mentioned in Sect. 4 into consideration, we developed the VA system. We use the tab functionality to allow the users to easily navigate between different parts/sections and analyze these parts without the need to re-apply the dimensionality reduction method and regenerate corresponding figures for analysis. The system is divided into two main parts which can be selected as main tabs (see the top line in Fig. 9):

1. *Overall Analysis*
2. *Wrong Layer Analysis*

The main tabs contain multiple components to address requirement **R2**. The linked views and the *control panel* containing multiple widgets provided in these parts are based on requirement **R3**. The following subsections explain these main parts and their components and how they differ from each other.

### 5.1 Overall analysis

The *Overall Analysis* part focuses on the analysis of all the tracks taking into consideration all the detector layers. It consists of four tabs with corresponding widgets for the analysis of the model reconstructing the tracks.

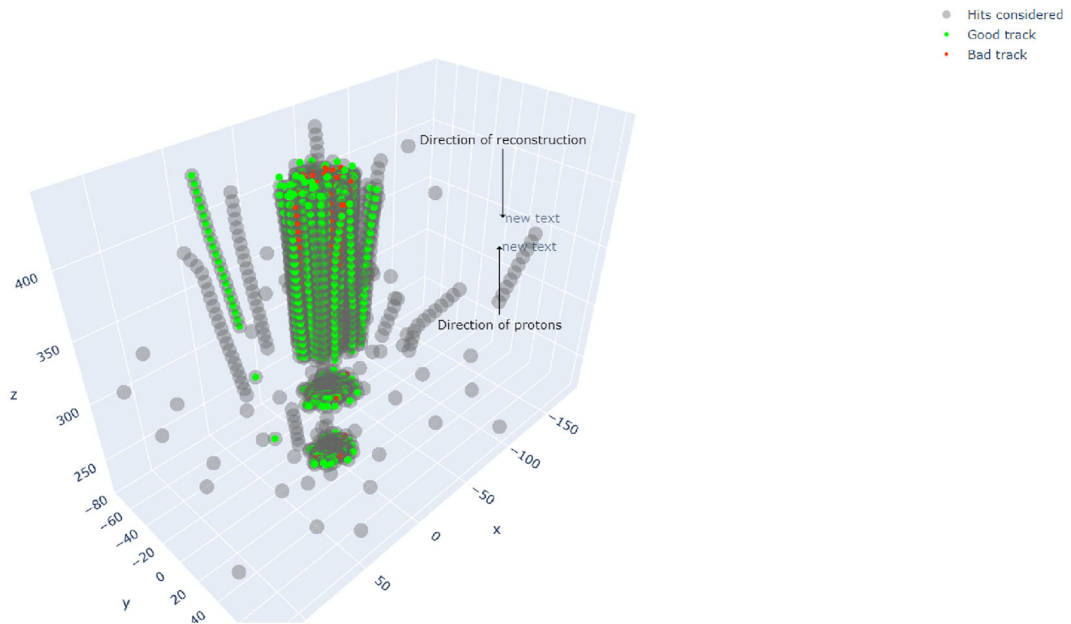


Overall Analysis Wrong Layer Analysis

## Analysis of Saliency Attributions through Dimensionality Reduction

Reconstructed Tracks Feature Analysis Layer Analysis Saliency Attributions

Visualization of the reconstructed tracks



**Fig. 9** 3D visualization of the reconstructed tracks. This view provides an intuitive way to select the track to be analyzed

- Tab: Reconstructed Tracks
- Tab: Feature Analysis
- Tab: Layer Analysis
- Tab: Saliency Attributions

In the following subsections, we describe the set of widgets (collectively referred to as *Control Panel*) provided in the above-mentioned tabs and the remaining components of these tabs in detail.

### 5.1.1 Control panel

The *control panel* provided in each tab (see, e.g., Fig. 10 above marked frames) consists of multiple widgets that help the users to modify the data which is being visualized and to modify the way of interaction with the plots present in two of the four tabs described below.

The *Plot data* drop-down menu allows the users to switch between different types of saliency attributions such as policy-based or value-based saliency attributions. The *Track number* (in feature analysis tab) and *Layer number* (in layer analysis tab) sliders allow the users to select different tracks  $p$  and detector layers in this selected track  $p$ , respectively, for analyzing corresponding saliency attributions in the connected tabs. The *Track type* drop-down menu allows the users

to select the types of tracks (correctly reconstructed, incorrectly reconstructed or all tracks) they intend to analyze. The *Track number* slider is updated to allow only a selection of the type of tracks being analyzed. The *Feature to embed* drop-down menu controls the set of saliency attributions that are used by a dimensionality reduction method. The *Show track and ground truth* checkbox allows the users to indicate the points corresponding to the reconstructed track and its ground truth in the 2D plots visualizing the embedded data.

The default color-coding of the 2D plots in linked plots utilizes a large number of discrete colors to color-code hits based on their ground truth value. This is useful in identifying tracks in 2D (in projection views) and their corresponding points in the embedded space (in TSNE plot). However, it is difficult to differentiate some of the colors due to the very minute difference between their appearances during interactive analysis. To address this issue, we allow the users to modify the color-coding of these 2D plots. Users can modify the color coding mechanism of the 2D plots in the linked plots using the *Switch to single color* checkbox. Selecting this option leads to color-coding of the selected points to orange and the unselected points to blue teal color. We choose this pair of colors as it provides good hue contrast to the selected and unselected points making the interpretability of the plots better. The *Selection mode* and the *Cross-filter mode* drop-

down menus correspond to the selection operation through which the plots are linked (the details regarding which of the plots are linked are provided in the description of corresponding tabs below). The *Selection mode* corresponds to the type of interaction between multiple select operations in a single plot. It allows the users to either overwrite or combine or consider the intersection of multiple selections in a particular plot. The *Cross-filter mode* drop-down menu corresponds to the type of interaction between the selection operations across different plots. It allows the users to either overwrite or consider the intersection of the selection operations performed on two different plots. The following subsections describe the tabs containing these control panel widgets.

### 5.1.2 Tab: reconstructed tracks

This tab, which is shown in Fig. 9, includes the visualization of all reconstructed tracks in three dimensions. It is provided to make it easier for the users to view and select an interesting track from the set of reconstructed tracks for analysis and thereby addressing requirement **R1**. The users can select the track that they wish to analyze by clicking on one of the hits belonging to that track. The remaining tabs connected with the track number will be automatically updated for the analysis of the saliency attribution data corresponding to this selected track. The color-coding of the reconstructed tracks is based on whether they are correctly or incorrectly reconstructed by the model. Alternatively, users can use the *Track number* slider provided in the feature analysis tab to select the desired track. Figure 9 shows the component visualizing an example dataset.

### 5.1.3 Tab: feature analysis

As explained before, the reconstruction of a single track involves multiple steps. The RL model starts reconstructing the track from one end and moves to the other end detector layer by detector layer. The *feature analysis* tab (see Fig. 10) deals with the visual analysis of the whole track. This means the saliency attributions of all the *action features* ( $M_{SG}^{act}(policy)$  and  $M_{SG}^{act}(value)$ ) used in reconstructing the whole track (in all detector layers) can be visually analyzed. This is important when analyzing the saliency attributions of input features such as  $z_{enc}$  and  $GNN(\vec{v}_i)$  which are intended to capture and provide structural information from the graph data.

The tab contains three plots (marked as **A** and **B** with **A** containing one plot and **B** containing two plots) as shown in Fig. 10 with the option to switch between  $M_{SG}^{act}(policy)$  and  $M_{SG}^{act}(value)$  using the *Plot data* drop-down menu.

- **A**: The saliency attributions of the *action features* of all the hits taken into consideration by the RL model in reconstructing a track are embedded in 2D space using the t-SNE dimensionality reduction method. This use of dimensionality reduction method allows us to analyze saliency attributions corresponding to multiple features instead of analyzing them individually. The default color-coding of the points in the plots is based on the index  $k$  of the ground truth path  $p_k$  of the tracks they belong to.
- **B (two plots)**: *3D plot* of all the hits taken into consideration for reconstructing the selected track and a *projection* of the 3D plot for interaction. The 3D visualization allows the user to see hits in spatial dimension that correspond to the points visualized in the embedded space described in **A**. The users can use the checkbox to switch between two projections (front and top) of the 3D plot. The projection plot on the right is intended for selecting desired hits.

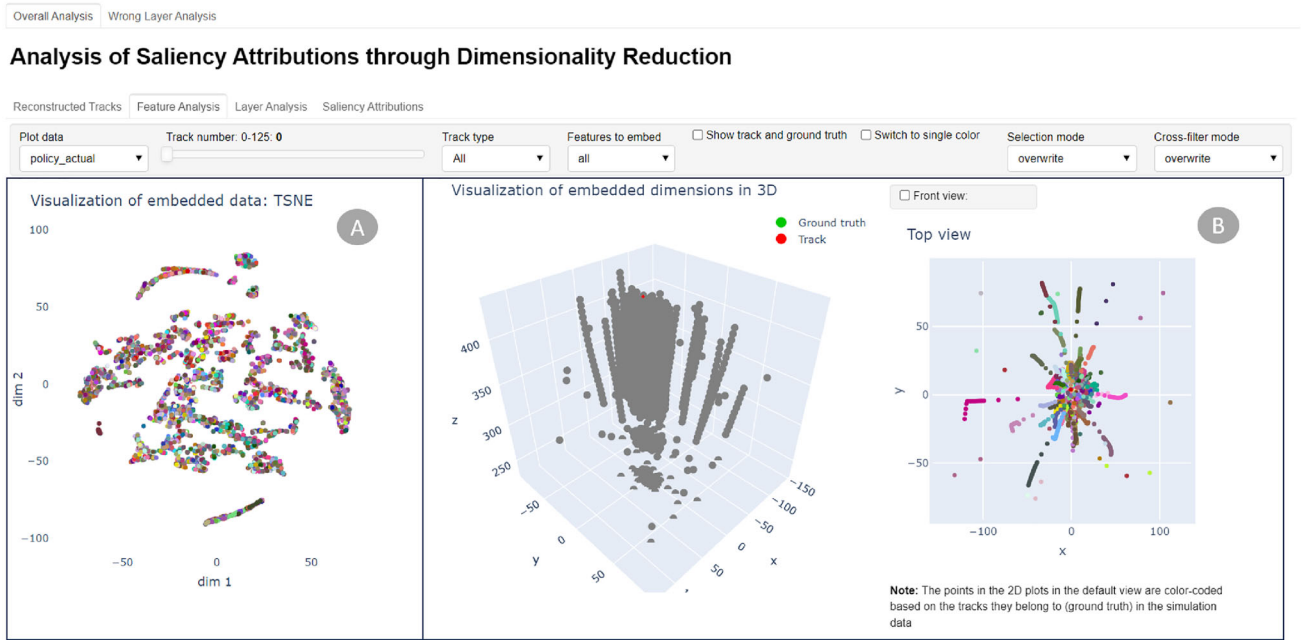
All the three plots are linked to each other using the selection operation. This allows the users to interact and examine these plots simultaneously. The box selection tool allows the users to select a set of points in one of the 2D plots and the corresponding points in the remaining plots will be highlighted. This tool is available on the top right corner of all the 2D plots.

### 5.1.4 Tab: layer analysis

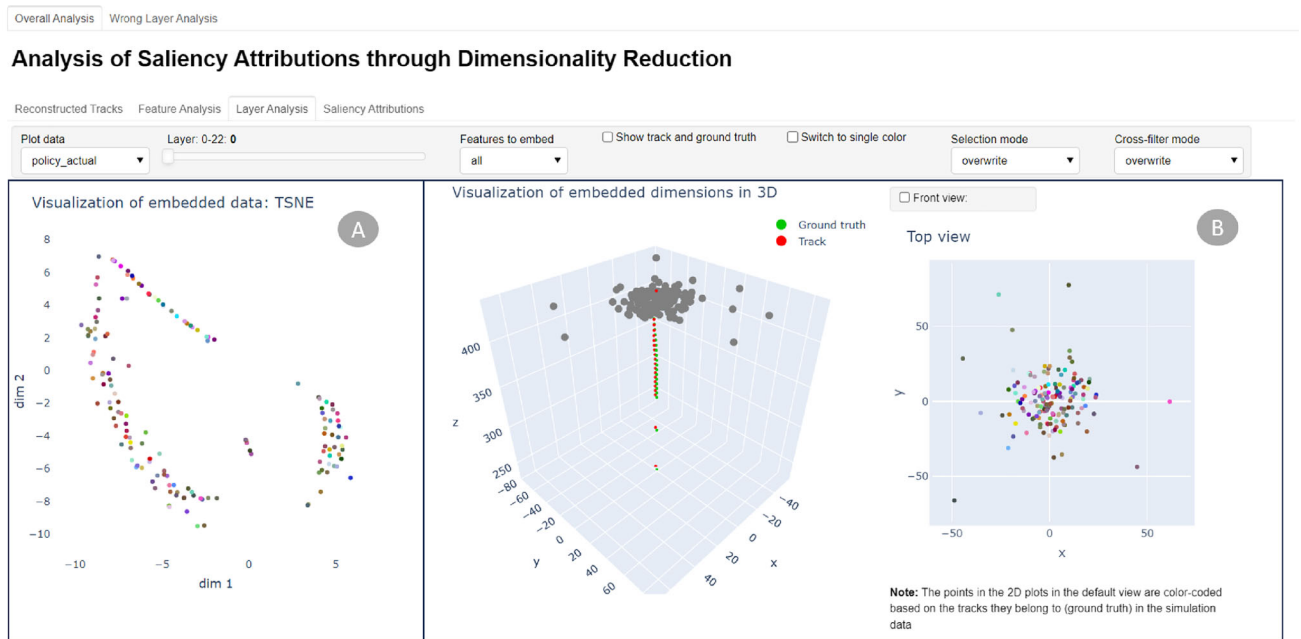
The *layer analysis* tab in Fig. 11 is intended for the instance-based analysis of the track-reconstruction model. It specifically addresses requirement **R4** concerning the analysis of a single step of the reconstruction process. The 3D plots in the center of Figs. 10 and 11 highlight the difference between analyzing multiple detector layers together and a single detector layer in the track reconstruction process. The users can choose a particular detector layer in the reconstructed track using the *Layer number* slider and analyze the behavior of the deepRL model when choosing a hit from the detector layer as part of the track. The tab contains the same set of plots as described in 5.1.3 (marked as **A** and **B** in both). The only difference is, the saliency attribution data considered. In this tab, we use the saliency attributions corresponding to the *action features* ( $M_{SG}^{act}(policy)$  and  $M_{SG}^{act}(value)$ ) of a single detector layer for analyzing the decision-making process of the model.

### 5.1.5 Tab: saliency attributions

The saliency attributions tab contains the plots to analyze the saliency attributions of each input feature of the hits belonging to a particular detector layer (see Fig. 12) instead of analyzing them in the embedded space using a dimensional-



**Fig. 10** An overview of the *Feature Analysis* tab of *Overall Analysis* part of the VA system: (A) Saliency Attributions of the features embedded using t-SNE dimensionality reduction technique visualized in 2D embedded space. (B) 3D visualization of all the hits used in reconstructing a track and one of the 2D projections of this 3D visualization



**Fig. 11** Layer-wise analysis of the saliency attributions corresponding to the input *action features* of a reconstructed track

ity reduction method. This allows the users to compare the influence of individual features on the output of the model based on the corresponding saliency attributions. In addition, the tab also provides 2D plots to analyze *observation features* and *positional encoding with adaptive receptive field (PE-ARF)*. This is done by providing two sub-tabs:

- *Action features* sub-tab
- *Observation and PE-ARF* sub-tab

The *Action features* sub-tab corresponds to the analysis of the deepRL model based on the saliency attributions  $M_{SG}^{act}(policy)$  and  $M_{SG}^{act}(value)$ . It contains three plots (marked as **A** and **B** in Fig. 12):

- **A (two plots)**: The scatter plot on the right is color coded based on the saliency attributions corresponding to one or a set of features from the *action features* of all the hits belonging to the selected detector layer. It helps the users in understanding how the deepRL model pays attention to the hits in detector layer  $(t + 1)$  based on one or a set of corresponding features in *action features* when tracking a hit from detector layer  $t$ . The drop-down menu *Action feature/s* on the left provides the users with the option to select these features. We use a diverging color map  $[-1; 0; 1]$  to plot the values. The scatter plot on the left indicates the hit selected by the model (labeled *Track neighborhood*) of the selected hit defined by the users using the *Radius* slider on the left side of the tab. These scatter plots are linked with *pan* and *zoom* functionalities. This connection provides the users with the information regarding the location of the hit selected by the model and the corresponding ground truth when analyzing the scatter plot on the left.
- **B**: The bar plot at the bottom is utilized to analyze the variation in the saliency attributions corresponding to the *action features*, ( $M_{SG}^{act}$ ) of the hit selected by the model, its ground truth and the hits in the neighborhood of the hit selected by the model (color-coded with translucent blue color). The neighborhood is defined by the *Radius* slider on the top left and the selected region (marked by a circle) can be seen in the scatter plot on the left side of the connected plots described in **A**.

The *Observation and PE-ARF* sub-tab corresponds to the analysis of the deepRL model based on the saliency attributions computed for *observation features* ( $M_{SG}^{obs}(policy)$  and  $M_{SG}^{obs}(value)$ ) and *PE-ARF* ( $M_{SG}^{PE-ARF}(policy)$  and  $M_{SG}^{PE-ARF}(value)$ ). It provides two plots (as shown in Fig. 13), one for the saliency attributions corresponding to the *observation features* of the hit being tracked and the other for the saliency attributions of *PE-ARF*. In this work, we focus

primarily on the saliency attributions corresponding to *action features* input.

## 5.2 Wrong layer analysis

The focus of the *wrong layer analysis* part is to analyze the detector layers of interest in the incorrectly reconstructed tracks. With this part, we try to address requirement **R5** regarding the analysis of detector layers of interest in incorrectly reconstructed tracks. We investigate the detector layers where the model has reconstructed the track  $p$  incorrectly and include the detector layer (if present) after which the RL model deviates from the corresponding ground truth path  $p^l$ .

This part contains two tabs with each tab containing the necessary widgets to analyze corresponding saliency attribution data. The tabs mentioned in Sects. 5.1.4 and 5.1.5 are used in this part to analyze the saliency data. Figure 14 shows the *layer analysis* tab of *wrong layer analysis*. The *Track number* drop-down menu in the *control panel* consists of only the incorrectly reconstructed tracks and the *Layer number* slider contains the list of all the detector layers where the reconstruction went wrong and the last detector layer (if present) until which the reconstruction was correct. The functions of the remaining widgets are similar to the widgets mentioned in Sect. 5.1.

## 6 User workflow

In this section, we describe the user workflow to demonstrate the working of the VA system. Figure 15 shows an overview of the user workflow for the *Overall Analysis* part in the system. For the *Wrong Layer Analysis* part, users follow the workflow depicted inside the region marked with red boundary.

The VA system starts with the default view of the *reconstructed tracks* tab. The users can explore the 3D visualization of all the reconstructed tracks in the data and select the desired track for analysis by clicking on the track. This interactive visualization addresses requirement **R1** mentioned in Sect. 4.

Once the track is selected, users switch to the *feature analysis* tab for the analysis of saliency attributions corresponding to the *action features* ( $M_{SG}^{act}(policy)$  and  $M_{SG}^{act}(value)$ ) of all the detector layers of the selected track together addressing requirement **R2**. The users are provided with the option to analyze saliency attributions corresponding to all the features of *action features* or select a predefined subset of these features.

To analyze a particular detector layer of interest among these layers of the selected track, users switch to the *layer analysis* tab. Here, the users can slide through the list of detector layers to analyze *action features'* saliency attributions,  $M_{SG}^{act}(policy)$  and  $M_{SG}^{act}(value)$ , corresponding to a

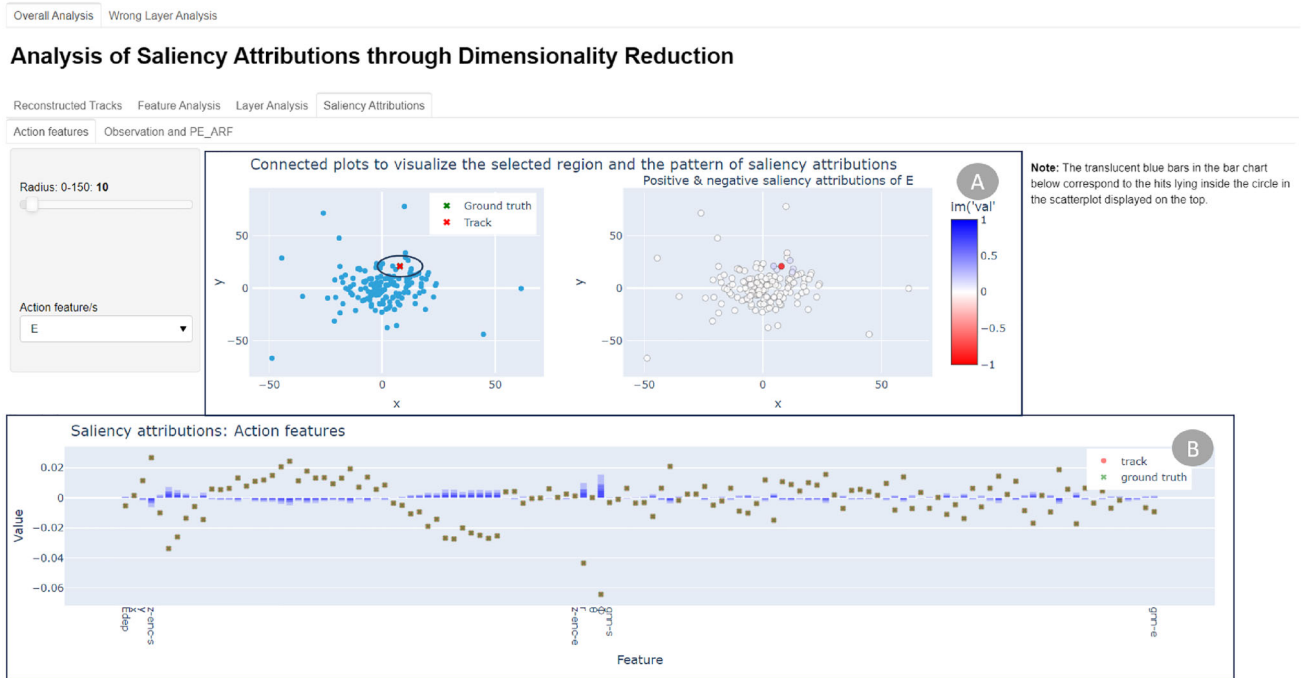


Fig. 12 Saliency attributions corresponding to the *action features* at the chosen detector layer of the reconstructed track

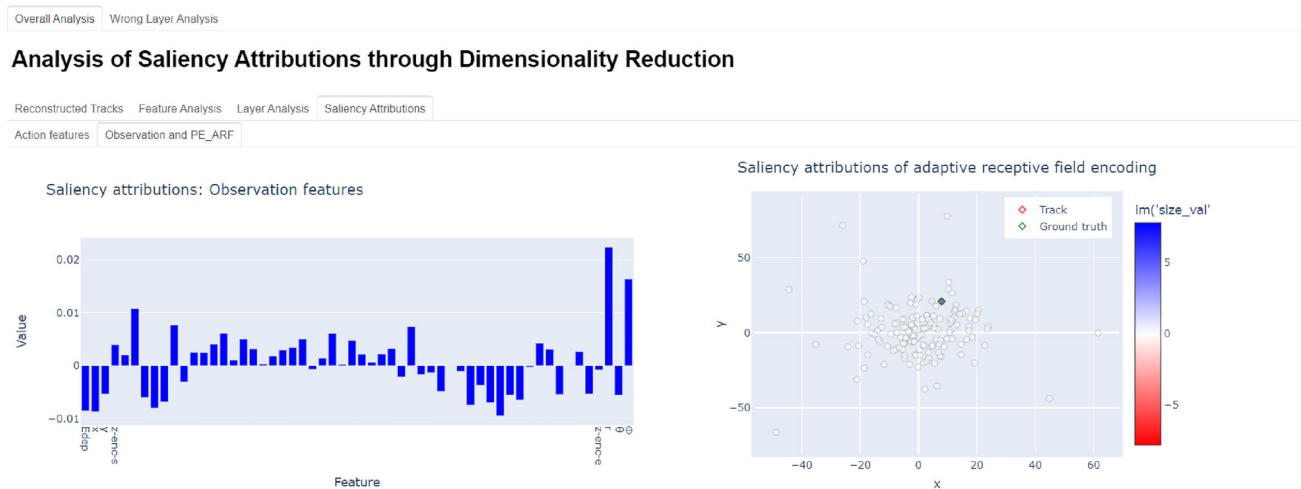


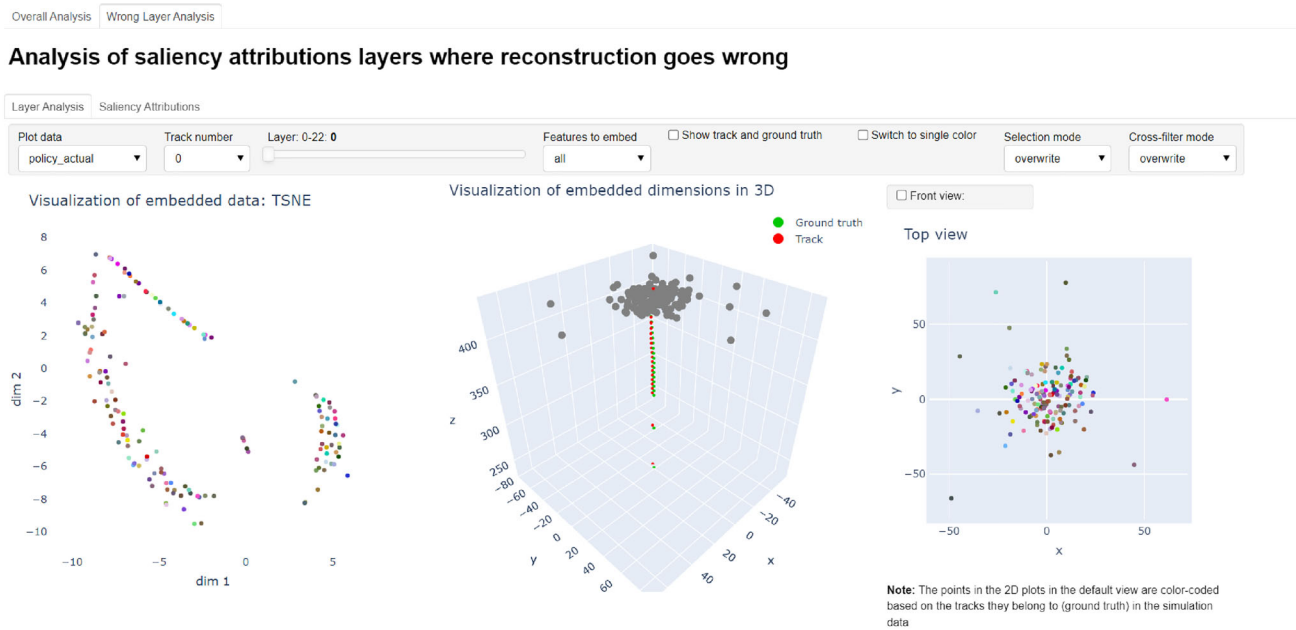
Fig. 13 Saliency attributions corresponding to the *observation features* and *PE-ARF* input at the chosen detector layer of the reconstructed track

particular detector layer. The selected dimensionality reduction technique is used in this tab to analyze large numbers of saliency attributions. The interactive tools associated with the figures in all the tabs and the tools provided in the *control panel* are designed to address requirement **R3**.

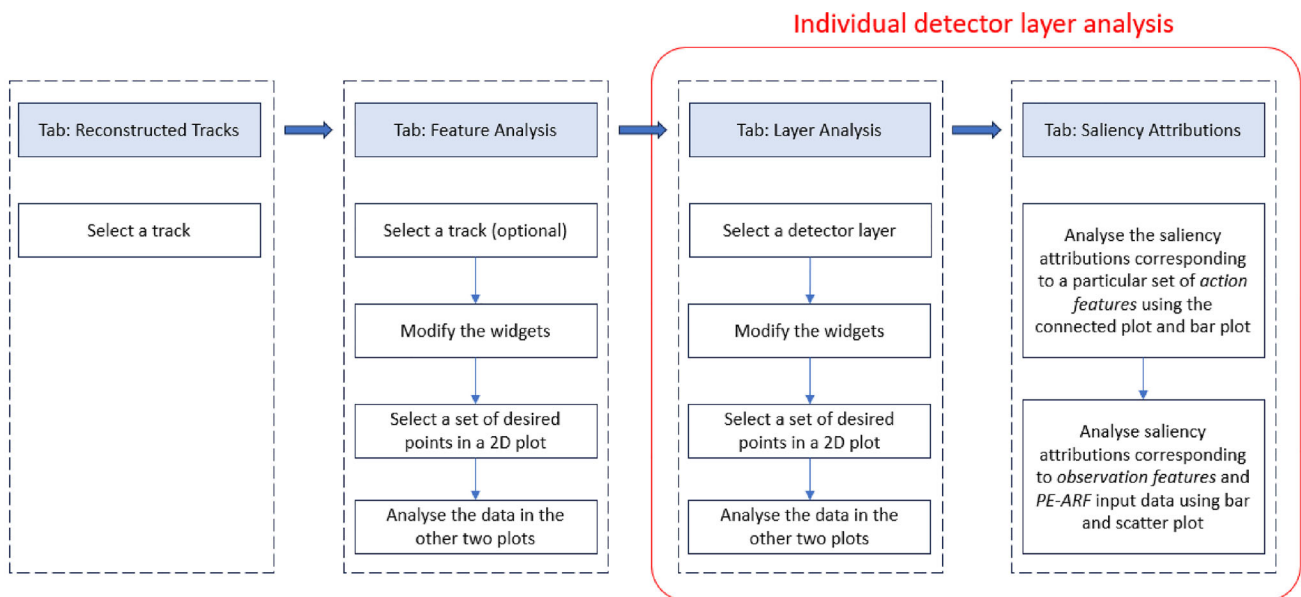
To analyze the actual saliency attributions  $M_{SG}^{act}(policy)$ ,  $M_{SG}^{act}(value)$ ,  $M_{SG}^{obs}(policy)$ ,  $M_{SG}^{obs}(value)$ ,  $M_{SG}^{PE-ARF}(policy)$  and  $M_{SG}^{PE-ARF}(value)$  corresponding to the selected detector layer, users switch to the *saliency attributions* tab. The *layer analysis* and *saliency attributions* tabs correspond to the analysis of a single step of the reconstruc-

tion process addressing requirement **R4**. It consists of two sub-tabs. In the *Action features* sub-tab, users can analyze the saliency attributions corresponding to the *action features* input. The *Observation and PE-ARF* sub-tab contains two plots each visualizing the saliency attributions corresponding to *observation features* and *PE-ARF* input data. We demonstrate the user workflow through two case studies in the video provided as supplemental material.

Reconstructed tracks consist of both correctly and incorrectly reconstructed tracks. To study the decision-making process of the model when it fails to reconstruct the track



**Fig. 14** Wrong layer analysis section. Only detector layers where the tracking deviates from ground truth (plus last correct detector layer) can be selected here



**Fig. 15** User workflow for the proposed VA system

correctly, users switch to the *Wrong Layer Analysis* tab on the top of the system. This segment of the system fulfills requirement **R5**. They follow the same workflow described for *Layer Analysis* tab and *Saliency Attributions* tab (marked by red colored rectangle in Fig. 15) above to analyze the model.

## 7 Case studies

In this section, we look at several case studies to demonstrate the usefulness of the proposed VA system. We focus on the saliency attributions corresponding to the *action features* input computed for *policy* output.

## 7.1 Case 1: Pattern of saliency attributions with respect to input features

### Aim

To investigate the pattern of positive and negative saliency attributions and compare it to the pattern observed for ML models working on other spatial data.

### Analysis

For this case study, we select a track and use the layer slider in the *layer analysis* tab to choose a particular detector layer for analyzing the *policy* output of the model reconstructing the selected track at this detector layer. We observe that the saliency attributions corresponding to the *action features* ( $M_{SG}^{act}(policy)$ ) of the hits present in the neighborhood of the hit selected by the model display distinct nature compared to the this selected hit. Most of the points corresponding to the hits in the neighborhood of the predicted hit lie on the other end of the trail of points in the embedded space. An example of the analysis is shown in Fig. 16 where we select the hits in the neighborhood of the hit selected by the model in the *Top view* plot (on the right side of the figure) and observe the above-mentioned distinctiveness in the t-SNE plot. To understand this behavior in detail, we investigate the model prediction at this detector layer in the *Saliency Attributions* tab. In particular, we study the pattern of saliency attributions corresponding to some of the important features belonging to *action features*.

In the *Saliency Attributions* tab, we focus on the connected plots provided on top for our analysis. We observe a pattern of positive and negative attention in Fig. 17a (neighborhood of the hit is zoomed-in in Fig. 17b) when we look at the saliency attributions of feature  $\phi$  (one of the most important input features) belonging to the *action features* of a detector layer. A similar pattern is observed for the polar radius,  $r$  which is another important feature. Rosynski et al. [31] also observed a similar pattern of positive and negative gradients when analyzing the usefulness of the gradient-based saliency maps in deepRL for image classification. We provide an illustrative example of saliency maps generated for ML classification model working on MNIST data in Fig. 17c. We observe that the pixels representing the numbers in MNIST images (Fig. 17c) have very high positive (with respect to the influence on the output) attention and they are surrounded by the pixels having high negative attention. This is similar to the pattern observed in Fig. 17b. Here, the white pixels representing the number in the input MNIST image correspond to the features of the hit in *action features* chosen by the deepRL model in detector layer  $(t + 1)$ . The black colored pixels surrounding these white pixels in the immediate neighborhood correspond to the hits (defined by their features in *action features*) surrounding the chosen hit in our scatter plot example.

### Result

The observations indicate that the deepRL model is showing a similar pattern of saliency attributions to that of machine learning algorithms working on other spatial data, there image data. This insight supports the idea that the model is showing an usual behavior when detecting the desired hit. This case is demonstrated in the video provided as supplemental material.

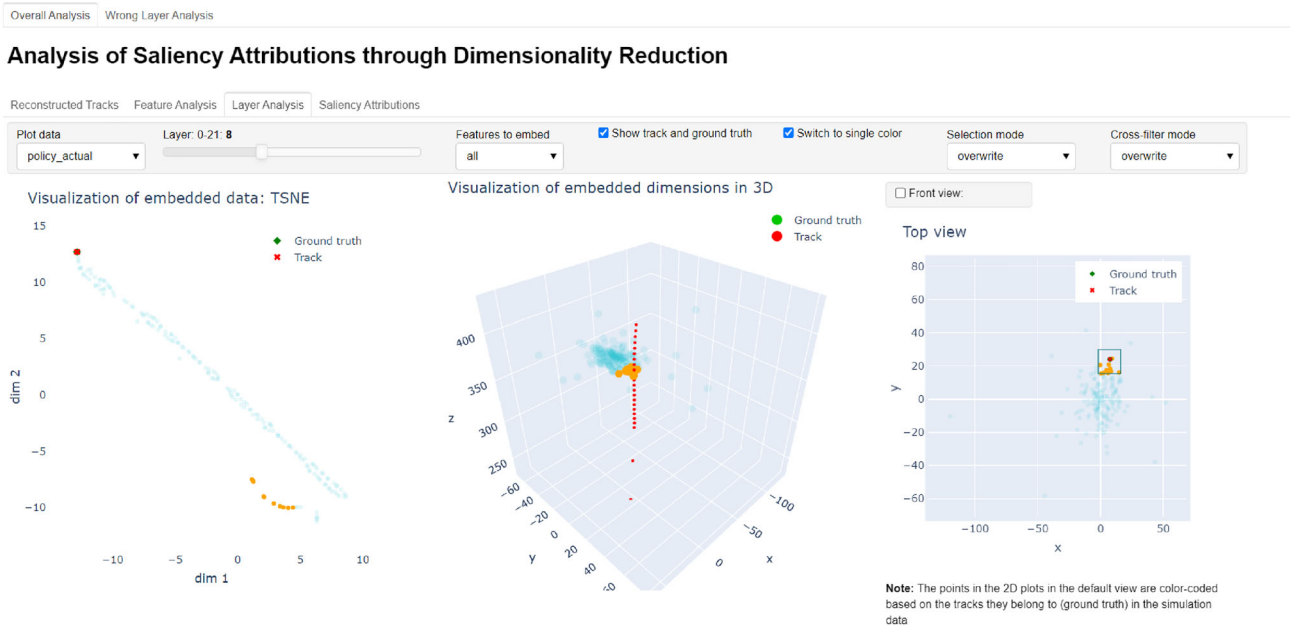
## 7.2 Case 2: Behavior of the model when the track has high deflection angle

### Aim

To investigate the action of the model when it fails to track a charged particle that has high deflection angle at a particular detector layer.

### Analysis

We look into the *layer analysis* tab of the *wrong layer analysis* part to investigate the model behavior when reconstructing a particular track (selected by choosing the track from the *Track number* drop-down menu) with a high deflection angle in a high-density region of particle hits. The deflection angle  $\phi_{jk}$  is one of the major features of particle hits that defines the difficulty of the track-reconstruction process of charged particles. Another important factor affecting the track reconstruction process is the density of particle hits. An increase in the density of particle hits increases the difficulty of reconstructing tracks significantly. Reconstructing tracks with high deflection angles can be very challenging for both humans and machines when we are dealing with the data of thousands of tracks simultaneously. In Fig. 18, we try to analyze the performance of the deepRL model in reconstructing a track with a high deflection angle at a particular detector layer. We observe that the model reconstructs a part of the track correctly but fails when the track deflects with a higher angle. The RL model follows an approximately straight-line path in reconstructing the track even at the first few detector layers (from the top of the 3D plot) after the reconstruction has gone wrong. Here, the reconstruction process following a straight-line path is desirable up to the first nine detector layers from the top but undesirable at the tenth detector layer. In the absence of the ground truth, it would be extremely difficult to classify the action of the model at the detector layer with a high deflection angle as incorrect. Even human observers would not be able to choose the *correct* action and would select the straightest but wrong continuation. This observation highlights the complexity of the point cloud data. It also indicates the need for a VA system to understand the behavior of the model without relying exclusively on the accuracy value of the model to evaluate its performance.



**Fig. 16** Layer-wise analysis of *action features*' saliency attributions corresponding to *policy* output

### Result

Knowing how the model decides wrong suggests specific changes in the strategy. In this case, it could, e.g., be helpful to optimize the connection of the hit for all tracks simultaneously [6].

### 7.3 Case 3: Positive gradients versus negative gradients

#### Aim

The saliency attributions generated show that the model pays positive attention to some features and negative attention to the other features of the selected hit belonging to a detector layer. The aim of this case study is to understand why the deepRL model has positive and negative saliency attributions for individual features in  $(\vec{v}_{t+1}, \vec{e}_{t,t+1}, GNN(v_{t+1}))$ .

#### Analysis

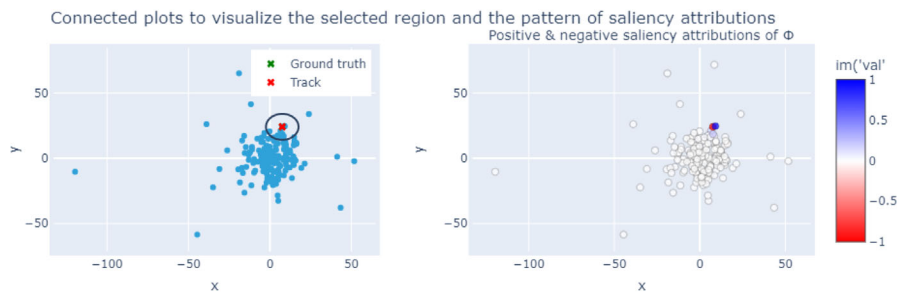
The SmoothGrad method involves computing the gradients of the output class score with respect to the input data. The sign of the gradient value indicates the direction in which the corresponding input value needs to be changed to observe an increase in the class score. The interpretation of the positive and negative gradients depends on the characteristics of the dataset being used. The complexity of the input data makes the interpretation of these gradients much harder.

We observe a similar ambiguity during the analysis of the SmoothGrad saliency attributions corresponding to a particular detector layer computed for the input data with respect to *policy* output in *saliency attributions* tab. We select a track using the *Track number* slider in the *Feature Analysis* tab and

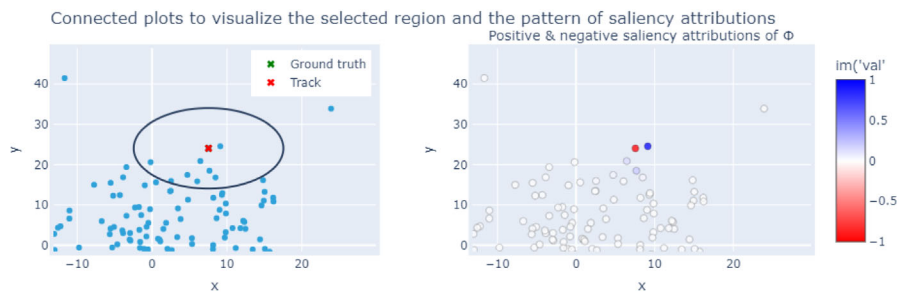
choose a layer using the *Layer* slider provided in the *Layer Analysis* tab. Then, we switch to *saliency attributions* tab to study the pattern of saliency attributions  $M_{SG}^{act}(policy)$ . The saliency attributions corresponding to the features  $r$  and  $\phi$  (see the region marked by red colored rectangle in the center of Fig. 19) in  $M_{SG}^{act}(policy)$  have the highest attention for the hit selected by the model (labeled as track) in the direction of negative gradients compared to the other hits in the layer (refer Fig. 19). However, the saliency attributions corresponding to feature  $y$  (marked by red colored rectangle on the left in Fig. 19) in  $M_{SG}^{act}(policy)$  show high attention for the reconstructed track in the direction of positive gradients with respect to the other hits. This difference in the saliency attributions is due to the properties of the input features. The features of the hits are quite distinct from each other with respect to the desired values of these features in selecting a particular hit. Lower values of deflection angle ( $\phi$ ) and polar radius ( $r$ ) indicate the straightness of the track. Therefore, having lower values of these features for track reconstruction is desirable. However, this is not the case with other features such as energy deposition  $E$  or feature  $x$ . This difference in the desired values of the input features results in model paying attention to these features differently.

Figure 6 shows the values of some of these input features belonging to *action features* at a given detector layer and the index marked *track* indicates the hit chosen by the model. Figure 6d, f indicates that lower values of the features  $r$  and  $\phi$  are desired by the deepRL model when choosing a hit as the part of a track.

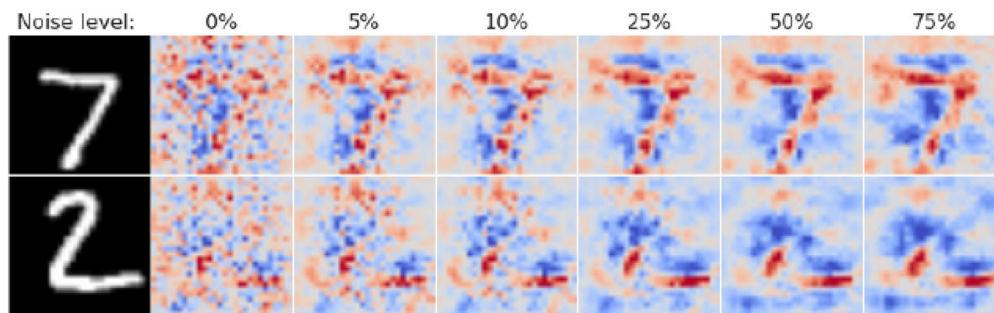




(a) Saliency attributions corresponding to  $\phi$  feature of the *actionfeatures* of hits in a detector layer.



(b) Neighborhood of the hit selected by the model in 17a



(c) Saliency attributions of five MNIST images computed using SmoothGrad saliency method with varying noise levels. Image from Smilkov et al.[10]

**Fig. 17** Comparison of the pattern of saliency attributions at a detector layer with the MNIST images

*Result*

We observed that due to the distinctiveness of the input features, for some of them high values are desirable, for other low values are desirable, the model learns to pay attention differently while reconstructing a track. Thus, the issue concerning the interpretation of positive and negative saliency attributions was successfully verified by analyzing the input features to understand their corresponding saliency attributions generated.

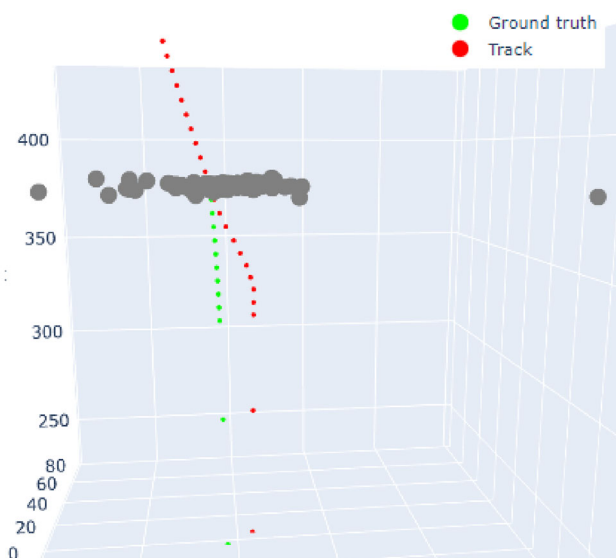
**7.4 Case 4: Observation regarding GNN embeddings**

*Aim*

GNN embeddings are intended to capture structural information in the graph data. Here, the aim is to verify if  $GNN(v)$  captures relations between the nodes in subsequent detector layers.

*Analysis*

Here, we analyze the influence of GNN embeddings ( $GNN(v)$ ) belonging to  $G_z$  that are used in defining the *action features* input for the reconstruction model. In particular, using the *Feature Analysis* tab from *Overall Analysis*



**Fig. 18** Incorrectly reconstructed track  $p$  (in red) which starts to deviate from ground truth path  $p^l$  where  $p^l$  exhibits a high deflection angle

part, we try to verify if the embedding values capture any structural information in the graph data  $G$ . We select a track (numbered 55 in this case) using the *Track number* slider provided on top the *Feature Analysis* tab. To analyze GNN embeddings, we select *gnn\_emb* from the *Feature to embed* drop-down menu. We also check both *Show track and ground truth* as well as *Switch to single color* checkboxes for better analysis. Figure 20 shows the analysis of the saliency attributions corresponding to the GNN embeddings in  $M_{SG}^{act}(policy)$ .

During the analysis of clusters, we select a set of points forming a cluster in the TSNE plot (on the left side of Fig. 20) for GNN embeddings using the *box selection* tool as shown in Fig. 20. The hits corresponding to these selected points (orange-colored hits in the 3D plot) appear to lie in the neighborhood of the reconstructed track. This is also highlighted by the *Top view* plot on the right side.

### Result

We observe that the  $GNN(v)$  from the graph data  $G_z$  does capture relations between the nodes in subsequent detector layers. Thus, we successfully verified that the GNN embeddings do capture structural information in the graph data. This case is demonstrated in the video provided as supplemental material.

## 7.5 Case 5: Saliency attributions of features in tracking layers and calorimeter layers

### Aim

To verify if the deepRL model recognizes the difference between tracking and calorimeter layers and adapts its strategy accordingly.

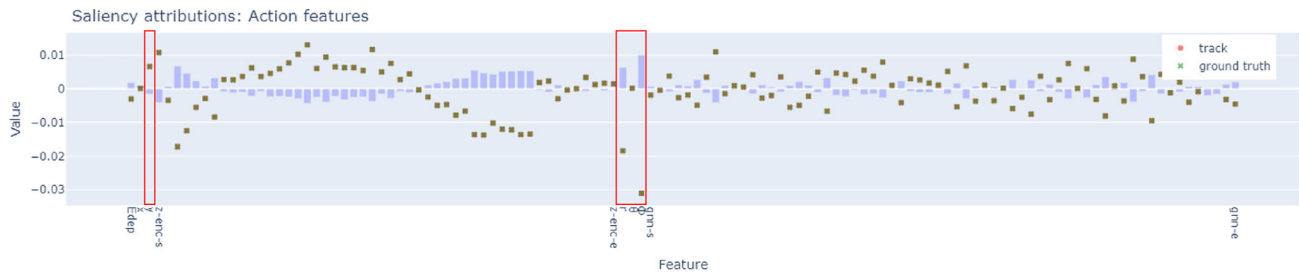
### Analysis

As mentioned in the description of data in Sect. 3, the pCT scanner consists of two tracking layers followed by multiple calorimeter layers. We can distinguish these layers in the 3D plot of Fig. 10. The two tracking layers can be spotted at the bottom of the scattered points. In this case study, we try to analyze the influence of *action features* on the *policy* output when reconstructing a track at the calorimeter layer and the tracking layer. We select a track in *Reconstructed Tracks* tab (or in *Feature Analysis* tab using the *Track number* slider provided). Then we move to the *Layer Analysis* tab to select a layer. Initially, we select the calorimeter layer adjacent to the tracking layer and switch to *Saliency Attributions* tab. Figure 21a shows the bar plot from *Saliency Attributions* tab corresponding to this calorimeter layer. Similarly, we obtain the bar plot corresponding to the tracking layer (Fig. 21b) by selecting the tracking layer adjacent to the calorimeter layer selected.

Using the plots of saliency attributions for *action features* from the *saliency attributions* tab in Fig. 21, we compare the saliency attributions of these two types of detector layers. Figure 21a shows a high influence of  $r$  and  $\phi$  (shown in the region marked by red colored rectangle) features of selected (by the model) hit's *action features*. However, the difference in attention with respect to other features is not very significant. However, Fig. 21b shows a high influence of the feature  $\phi$  (shown in the region marked by red colored rectangle) with a significant difference compared to the attention the other features of the hit selected by the model (labeled *Track*) receive at the tracking layer. This observation indicates that the  $\phi$  feature of *action features* is significantly more influential than other features in tracking layers compared to calorimeter layers. We observe a similar pattern when analyzing the saliency attributions corresponding to the value output of the reconstructing model.

### Result

We observed that the model learns to pay significantly higher attention to the deflection angle,  $\phi$ , when reconstructing a track in tracking layers compared to the calorimeter layers. Thus, we successfully verified the ability of the model to modify its strategy based on the change in situation.



**Fig. 19** Saliency attributions of the hit selected by the model, the ground truth and the hits in the neighborhood (color-coded with translucent blue) of the hit selected by the model on a given detector layer of an example track. *Note:* The model reconstructs the track correctly on these

detector layers. Therefore, we have an overlap of points representing the hit belonging to the reconstructed track and its ground truth. The use of red boxes is intended only to highlight the region being referred to in the description of case studies



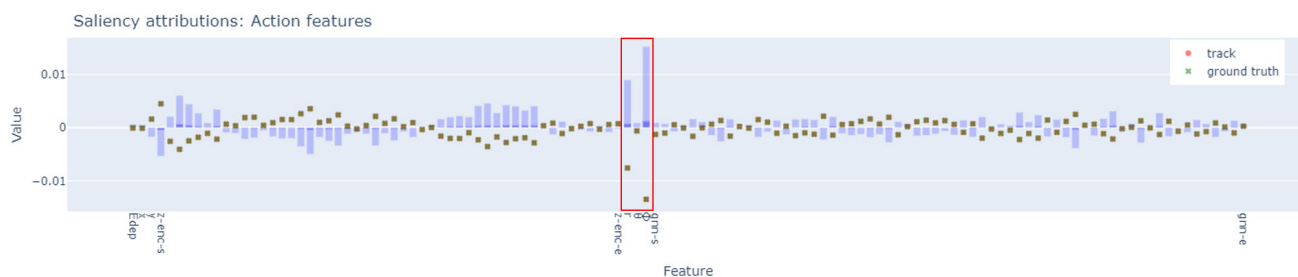
**Fig. 20** Analysis of saliency attributions corresponding to the GNN embeddings in  $M_{SG}^{act}(policy)$

### 8 Expert feedback

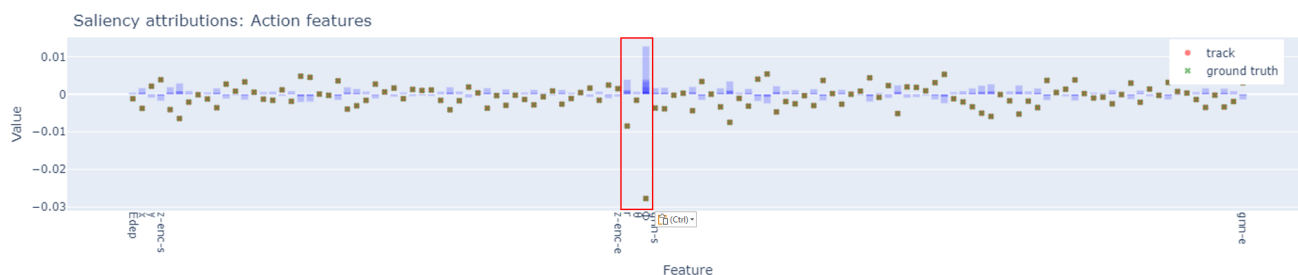
The primary objective of our VA system is to assist ML experts in understanding the model during the developmental phase. We utilize visualization tools to make the interpretation of the ML model easier. To understand the usefulness and the limitations of the developed VA system, we invited two experts (Expert1 from ML and Expert2 from the visualization field) to use our VA system. Expert1 is quite familiar with the data and the model whereas Expert2 is unfamiliar with the same. We explained the particle tracking mechanism, the tracking model, the data used and the parts of the VA system to analyze the saliency attributions computed by adapting the SmoothGrad method.

Expert1 found the overall VA system quite useful in exploring the large amount of saliency attributions data. He

mentioned that the layout of the system helped him explore the data in detail as he navigated from the *Reconstructed Tracks* tab to the *Saliency Attributions* tab in the *Overall Analysis* part. Due to his familiarity with the data and the model, he was able to understand the plots without much difficulty. He appreciated the use of linked plots to analyze the saliency attributions in the *Feature Analysis* and *Layer Analysis* tabs. He said that the use of the front view and top view of the 3D plots in these two tabs improved the exploration of the data as they allowed the users to select hits based on their spatial position. The use of *Layer Analysis* and *Saliency Attributions* tabs helped him get better insights into the saliency attributions as they deal with the analysis of a single instance of track reconstruction. In particular, he found the *Saliency Attributions* tab very informative due to the visualization of saliency attributions corresponding to all



(a) Policy output-based saliency; calorimeter layer



(b) Policy output-based saliency; tracking layer

**Fig. 21** Comparison of saliency attributions of the hit selected by the model and the hits in the neighborhood (color-coded with translucent blue color) in a calorimeter layer and tracking layer for policy output. *Note:* The model reconstructs the track correctly on these detector

layers. Therefore, we have an overlap of points representing the hit belonging to the reconstructed track and its ground truth. The use of red boxes is intended only to highlight the region being referred to in the description of case studies

the input features at the chosen detector layer. He said that the system can also be used as a tool for feature engineering.

In addition to the above-mentioned comments regarding the VA system, he also provided us with some recommendations to improve the system. He recommended highlighting the track in *Reconstructed Tracks* tab when exploring and selecting a desired track for analysis to reduce the visual clutter. He suggested including a lasso selection tool to improve the interactivity in the linked plots used in *Feature Analysis* and *Layer Analysis* tabs. He found the default color-coding of the points (based on the ground truth) in the 2D plots in these two tabs to be quite overwhelming due to the use of a large number of discrete colors. We encountered this issue during the developmental phase and addressed it by providing users with the *Switch to single color* checkbox option. For the 3D plots used in these two tabs, he recommended further reduction in the opacity value of the hits taken into consideration (color-coded with gray) to improve the visibility of the reconstructed track and its ground truth. However, reducing the opacity value also influences the interactions (in linked plots) as it makes it difficult to differentiate between the selected and not selected hits. In the *Saliency Attributions* tab, he recommended replacing the selection of neighborhood using a circle centered at the hit selected to a region based on the incident angle of the track. He also recommended highlight-

ing this region on the scatter plot visualizing the saliency attributions of PE-ARF. Another important recommendation was to visualize the output of the models that could provide additional insights. These recommendations will be a part of our future work to improve the VA system.

*Expert2* liked the overall structure of the VA system. He said the progressive arrangement of the tabs is helpful in exploring the data in a structured manner. He found the linked plots very useful in exploring the data. He appreciated the availability of *Selection mode* and *Cross-filter mode* drop-down menus to modify the type of interaction when using the selection tool. He also found the use of connected plots to explore data very useful as the scatter plot on the left side provided context to the color-coded scatter plot on the left even during the use of pan and zoom functionalities.

However, due to the complexity of the data and the model being used for analysis in the VA system, he found it difficult to interpret some of the plots. This is understandable as it requires a strong understanding of the data and the ML model to understand all the plots provided in the VA system. For the 3D plots in *Feature Analysis* and *Layer Analysis* tabs, he recommended providing users with the option to change the projection type to orthogonal which would allow them to distinguish between detector layers easily. The 2D projection views (Front and Top view) on the right side of these 3D plots

do address this issue. In future, we will work toward integrating these views in 3D plots. He also recommended providing the users with the possibility to use different dimensionality reduction techniques as they could provide different but interesting insights into the data. However, providing multiple dimensionality reduction techniques would complicate the data analysis process due to too many varying insights. This is one of the primary reasons to provide only one dimensionality reduction method for data analysis in our system.

Based on the detailed feedback of both experts, we derived the following conclusions:

- The VA system is a useful tool in exploring the saliency attribution data to understand the decision-making of the deepRL model and can also be used as a tool for feature engineering.
- Interaction and visualization of some of the plots can be improved to improve analysis.
- Providing additional information such as model output could further improve the analysis.

## 9 Discussion and conclusion

Previous work in the field of visual analytics has shown that VA systems can be a very useful tool in understanding and debugging ML algorithms. The need for such systems increases when we try to analyze ML models working on high-dimensional data. Through our work, we try to address one such need in the field of RL models working on high-dimensional point cloud data.

### 9.1 Discussion

The proposed system is designed for the analysis of deepRL models working on the reconstruction of the tracks of charged particles. The system takes every input value into consideration to allow the users to analyze the ML model in detail. It also provides the users with the possibility to analyze the model using the saliency attributions corresponding to one of the important input data in two different stages, one taking the reconstruction of the whole track (consisting of multiple reconstruction steps) into consideration while the other provides analysis of single reconstruction step at individual detector layers. While the system shows promising results (demonstrated through case studies) in analyzing the reconstruction model considered, it also has some limitations associated with it.

*Generalizability:* We considered point cloud data with multiple features in this work. The current system can be used to analyze deepRL models working on similar point cloud data with varying numbers of features. This property is mainly

useful for model developers in feature engineering of the input data as pointed out by Expert1 in Sect. 8.

The system uses an adaptation of SmoothGrad, a post hoc saliency mapping method, for analyzing the deepRL model. It can be extended to analyze the model using similar saliency mapping methods that produce saliency maps for input point cloud features. One of the important properties of the SmoothGrad method is that it is model-agnostic, which indicates that the method is independent of the model architecture. This property is useful for the model developers in analyzing the reconstruction model at multiple stages of model building. In this work, we considered a deepRL model with *actor-critic* architecture as mentioned in Sect. 3.2. However, the system can be used to analyze deepRL models with different architectures or *supervised learning* models that are modeled to track charged particles using the point cloud data described in this work. The system is also applicable to ML models dealing with other layered point clouds, e.g., time-based data, where a time step or time slice would correspond to a layer.

*Limitations:* In the current implementation of the system, we consider point cloud data with multiple features to analyze a deepRL model. The system does not support any other data formats such as image or text data for the analysis. We also rely on the ground truth provided by the simulation data for analyzing the model even though this information would be missing in the real-world data. Therefore, the VA system is currently limited to the analysis of deepRL model working on simulation point cloud data or methods for producing a different ground truth, like the one by Eschbach et al. [32], need to be employed.

Another limitation of the system is related to interactivity. This is corresponding to the limitation of providing only *box selection* tool for the selection of points in the linked views and the inability to select points in 3D plots visualized in the *feature analysis* and the *layer analysis* tabs.

### 9.2 Conclusion

In this paper, we proposed a VA system to analyze deepRL models working on the reconstruction of tracks of charged particles. The system assists the users in analyzing the model by providing the tools to utilize saliency attributions corresponding to every input data used. We presented several case studies that provide useful insights into the working of the ML model taken into consideration and thus showed that the system can be used by the model developers to verify, understand and debug the model and by end-users to understand the decision-making process of the model. The qualitative analysis of the system based on the feedback of two experts also highlighted the usefulness of the system proposed.

We developed the system using the high-level visualization tools provided by HoloViz [33]. The source code of the developed system is available at [34].

Our future work will be to implement the recommendations of experts mentioned in Sect. 8 and address the limitations mentioned above.

**Supplementary Information** The online version contains supplementary material available at <https://doi.org/10.1007/s00371-024-03297-3>.

**Acknowledgements** This work was partly supported by the German federal state of Rhineland-Palatinate (*Forschungskolleg SIVERT*). We thank all members of the *Bergen pCT Collaboration* for providing consulting and especially Tobias Kortus for providing the data and the model for analysis. Members of the Bergen pCT Collaboration: Johan Alme<sup>a</sup>, Gergely Gábor Barnaföldi<sup>c</sup>, Rene Barthel<sup>d</sup>, Viatcheslav Borshchov<sup>l</sup>, Anthony van den Brink<sup>d</sup>, Mamdouh Chaar<sup>a</sup>, Viljar Eikeland<sup>a</sup>, Georgi Genov<sup>a</sup>, Ola Grøttvik<sup>a</sup>, Håvard Helstrup<sup>e</sup>, Ralf Keidel<sup>k</sup>, Chinorat Kobdaj<sup>m</sup>, Shruti Mehendale<sup>a</sup>, Ilker Meric<sup>e</sup>, Odd Harald Odland<sup>a,b</sup>, Gábor Papp<sup>g</sup>, Thomas Peitzmann<sup>d</sup>, Helge Egil Seime Pettersen<sup>b</sup>, Pierluigi Piersimoni<sup>a</sup>, Attiq Ur Rehman<sup>a</sup>, Matthias Richter<sup>h</sup>, Dieter Röhrich<sup>a</sup>, Andreas Tefre Samnøy<sup>a</sup>, Joao Seco<sup>i,j</sup>, Hesan Shafiee<sup>a,f</sup>, Arnon Songmoolnak<sup>a,m</sup>, Jarle Rambo Sølief<sup>a,f</sup>, Ganesh Tambave<sup>a</sup>, Ihor Tymchuk<sup>l</sup>, Kjetil Ullaland<sup>a</sup>, Monika Varga-Kofarago<sup>c</sup>, Lennart Volz<sup>i,j</sup>, Boris Wagner<sup>a</sup>, RenZheng Xiao<sup>a,n</sup>, Shiming Yang<sup>a</sup>, Christoph Garth<sup>o</sup>, Nicolas R. Gauger<sup>p</sup>, Steffen Wendzel<sup>k</sup>, Alexander Wiebel<sup>k</sup>, Tobias Kortus<sup>k</sup>, Alexander Schilling<sup>k</sup>, Raju Ningappa Mulawade<sup>k</sup>, Sebastian Zillien<sup>k</sup>, Max Aehle<sup>p</sup>, Viktor Leonhardt<sup>o</sup>

a) Dep. of Physics and Technology, University of Bergen, 5020 Bergen, Norway. b) Dep. of Oncology and Medical Physics, Haukeland University Hospital, 5021 Bergen, Norway. c) Dep. for Th. Phys., Heavy-Ion RG, Wigner RCP of the Hung. Acad. of Sci., Budapest, Hungary. d) Institute for Subatomic Physics, Utrecht University/Nikhef, Utrecht, Netherlands. e) Dep. of Comp., Math. and Phys., West. Norway University of Applied Science, Bergen, Norway. f) Dep. of Electrical Engineering, West. Norway University of Applied Sciences, Bergen, Norway. g) Institute for Physics, Eötvös Loránd University, 1/A Pázmány P. Sétány, Budapest, Hungary. h) Dep. of Physics, University of Oslo, Oslo, Norway. i) Dep. of Biomed. Phys. in Rad. Oncology, German Cancer Research Center, Heidelberg, Germany. j) Dep. of Physics and Astronomy, Heidelberg University, Heidelberg, Germany. k) Center for Technology and Transfer, University of Applied Sciences Worms, Worms, Germany. l) LTU, Kharkiv, Ukraine. m) Institute of Science, Suranaree University of Technology, Nakhon Ratchasima, Thailand. n) College of Mechanical & Power Engineering, China Three Gorges University, Yichang, China. o) Scientific Visualization Lab, TU Kaiserslautern, Kaiserslautern, Germany. p) Scientific Computing, TU Kaiserslautern, Kaiserslautern, Germany.

**Funding** Open Access funding enabled and organized by Projekt DEAL.

**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copy-

right holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

## References

- Farrell, S., Anderson, D., Calafiura, P., Cerati, G., Gray, L., Kowalkowski, J., Mudigonda, M., Spentzouris, P., Spiropoulou, M., Tsaris, A., et al.: The HEP.TrkX Project: deep neural networks for HL-LHC online and offline tracking. In: EPJ Web of Conferences, vol. 150, p. 00003. EDP Sciences (2017). <https://doi.org/10.1051/epjconf/201715000003>
- Biscarat, C., Caillou, S., Rougier, C., Stark, J., Zahreddine, J.: Towards a realistic track reconstruction algorithm based on graph neural networks for the HL-LHC. In: EPJ Web of Conferences, vol. 251, p. 03047. EDP Sciences (2021). <https://doi.org/10.1051/epjconf/202125103047>
- Elabd, A., Razavimaleki, V., Huang, S.-Y., Duarte, J., Atkinson, M., DeZoort, G., Elmer, P., Hauck, S., Hu, J.-X., Hsu, S.-C., et al.: Graph neural networks for charged particle tracking on FPGAs. *Front. Big Data* **5**, 828666 (2022). <https://doi.org/10.3389/fdata.2022.828666>
- DeZoort, G., Thais, S., Duarte, J., Razavimaleki, V., Atkinson, M., Ojalvo, I., Neubauer, M., Elmer, P.: Charged particle tracking via edge-classifying interaction networks. *Comput. Softw. Big Sci.* **5**, 1–13 (2021). <https://doi.org/10.1007/s41781-021-00073-z>
- Farrell, S., Calafiura, P., Mudigonda, M., Anderson, D., Vlimant, J.-R., Zheng, S., Bendavid, J., Spiropulu, M., Cerati, G., Gray, L., et al.: Novel deep learning methods for track reconstruction. *arXiv preprint arXiv:1810.06111* (2018)
- Kortus, T., Keidel, R., Gauger, N.R.: on behalf of the Bergen pCT collaboration: towards neural charged particle tracking in digital tracking calorimeters with reinforcement learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **45**(12), 15820–15833 (2023). <https://doi.org/10.1109/TPAMI.2023.3305027>
- Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-CAM: visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 618–626 (2017). <https://doi.org/10.1109/ICCV.2017.74>
- Lundberg, S.M., Lee, S.-I.: A unified approach to interpreting model predictions. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (eds.) *Advances in Neural Information Processing Systems*, vol. 30, pp. 4765–4774 (2017). <https://proceedings.neurips.cc/paper/2017/file/8a20a8621978632d76c43dfid28b67767-Paper.pdf>
- Mulawade, R.N., Garth, C., Wiebel, A., on behalf of the Bergen pCT collaboration.: Saliency clouds: visual analysis of point cloud-oriented deep neural networks in DeepRL for particle physics. In: Archambault, D., Nabney, I., Peltonen, J. (eds.) *Machine Learning Methods in Visualisation for Big Data*. The Eurographics Association, Eindhoven (2022). <https://doi.org/10.2312/mlvis.20221069>
- Smilkov, D., Thorat, N., Kim, B., Viégas, F., Wattenberg, M.: SmoothGrad: removing noise by adding noise. *arXiv preprint arXiv:1706.03825* (2017)
- Yuan, J., Chen, C., Yang, W., Liu, M., Xia, J., Liu, S.: A survey of visual analytics techniques for machine learning. *Comput. Vis. Media* **7**, 3–36 (2021). <https://doi.org/10.1007/s41095-020-0191-7>
- Xuan, X., Zhang, X., Kwon, O.-H., Ma, K.-L.: VAC-CNN: a visual analytics system for comparative studies of deep convolutional neural networks. *IEEE Trans. Vis. Comput. Graph.* **28**(6), 2326–2337 (2022). <https://doi.org/10.1109/TVCG.2022.3165347>
- Hohman, F., Park, H., Robinson, C., Chau, D.H.P.: Summit: scaling deep learning interpretability by visualizing activation and attri-

- bution summarizations. *IEEE Trans. Vis. Comput. Graph.* **26**(1), 1096–1106 (2019). <https://doi.org/10.1109/TVCG.2019.2934659>
14. Zheng, T., Chen, C., Yuan, J., Li, B., Ren, K.: Pointcloud saliency maps. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (2019). <https://doi.org/10.48550/arXiv.1812.01687>
  15. Zhang, M., You, H., Kadam, P., Liu, S., Kuo, C.-C.J.: Pointhop: an explainable machine learning method for point cloud classification. *IEEE Trans. Multimed.* **22**(7), 1744–1755 (2020). <https://doi.org/10.1109/TMM.2019.2963592>
  16. Tan, H., Kotthaus, H.: Surrogate model-based explainability methods for point cloud NNS. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 2239–2248 (2022). <https://doi.org/10.48550/arXiv.2107.13459>
  17. Ribeiro, M.T., Singh, S., Guestrin, C.: “Why Should I Trust You?”: explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD ’16, pp. 1135–1144. Association for Computing Machinery, New York (2016). <https://doi.org/10.1145/2939672.2939778>
  18. Springenberg, J.T., Dosovitskiy, A., Brox, T., Riedmiller, M.: Striving for simplicity: the all convolutional net (2014). arXiv preprint [arXiv:1412.6806](https://arxiv.org/abs/1412.6806)
  19. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European Conference on Computer Vision, pp. 818–833. Springer (2014). [https://doi.org/10.1007/978-3-319-10590-1\\_53](https://doi.org/10.1007/978-3-319-10590-1_53)
  20. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: International Conference on Machine Learning, pp. 3319–3328. PMLR (2017). <https://doi.org/10.48550/arXiv.1703.01365>
  21. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: Proceedings of the IEEE CVPR, pp. 2921–2929 (2016). <https://doi.org/10.1109/CVPR.2016.319>
  22. Gupta, A., Watson, S., Yin, H.: 3D point cloud feature explanations using gradient-based methods. In: 2020 International Joint Conference on Neural Networks (IJCNN), pp. 1–8. IEEE, Glasgow (2020). <https://doi.org/10.1109/IJCNN48605.2020.9206688>
  23. Matrone, F., Paolanti, M., Felicetti, A., Martini, M., Pierdicca, R.: BubbLEX: an explainable deep learning framework for point-cloud classification. *IEEE J. Select. Top. Appl. Earth Observ. Remote Sens.* **15**, 6571–6587 (2022). <https://doi.org/10.1109/JSTARS.2022.3195200>
  24. Schwegler, M., Müller, C., Reiterer, A.: Integrated gradients for feature assessment in point cloud-based data sets. *Algorithms* (2023). <https://doi.org/10.3390/a16070316>
  25. Maaten, L., Hinton, G.: Visualizing data using t-SNE. *J. Mach. Learn. Res.* **9**, 11 (2008)
  26. Espadoto, M., Martins, R.M., Kerren, A., Hirata, N.S.T., Telea, A.C.: Toward a quantitative survey of dimension reduction techniques. *IEEE Trans. Vis. Comput. Graph.* **27**(3), 2153–2173 (2021). <https://doi.org/10.1109/TVCG.2019.2944182>
  27. Johnson, R.P.: Review of medical radiography and tomography with proton beams. *Rep. Prog. Phys.* **81**(1), 016701 (2017). <https://doi.org/10.1088/1361-6633/aa8b1d>
  28. Alme, J., Barnaföldi, G.G., Barthel, R., Borshchov, V., Bodova, T., Brink, A., Brons, S., Chaar, M., Eikeland, V., Feofilov, G., et al.: A high-granularity digital tracking calorimeter optimized for proton CT. *Front. Phys.* (2020). <https://doi.org/10.3389/fphy.2020.568243>
  29. Mager, M.: ALPIDE, the monolithic active pixel sensor for the alice its upgrade. *Nucl. Instrum. Methods Phys. Res. Sect. A* **824**, 434–438 (2016). <https://doi.org/10.1016/j.nima.2015.09.057>
  30. Kortus, T., Schilling, A., Keidel, R., Gauger, N.R.: on behalf of the Bergen pCT collaboration: particle tracking data: Bergen DTC prototype. Zenodo (2022). <https://doi.org/10.5281/zenodo.7426388>
  31. Rosynski, M., Kirchner, F., Valdenegro-Toro, M.: Are gradient-based saliency maps useful in deep reinforcement learning? (2020). arXiv preprint [arXiv:2012.01281](https://arxiv.org/abs/2012.01281)
  32. Eschbach, R., Messerschmidt, K., Keidel, R., Wiebel, A.: on behalf of the Bergen pCT Collaboration: semi-automatic particle tracking for and visualization of particle detector data. In: Bender, J., Botsch, M., Keim, D.A. (eds.) *Vision, Modeling, and Visualization*. The Eurographics Association, Eindhoven (2022). <https://doi.org/10.2312/vmv.20221210>
  33. HoloViz. <https://holoviz.org/>. Accessed 27 Mar 2023
  34. Mulawade, R.N., Garth, C., Wiebel, A., on behalf of the Bergen pCT collaboration.: Source code for paper ‘Visual analytics system for understanding deeprl-based charged particle tracking’. Zenodo (2024). <https://doi.org/10.5281/zenodo.10491504>

**Publisher’s Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Raju Ningappa Mulawade** received his bachelor of engineering (B.E) degree in Mechanical Engineering from Sri Jayachamarajendra College of Engineering, Mysore, India, in 2013. He received his M.S. degree in Mechatronics from Technische Universität Hamburg (TU Hamburg) in 2021. He is currently pursuing his PhD at the University of Applied Sciences Worms. His research interests include artificial intelligence (AI) with a focus on explainable AI (XAI) and scientific visualization.



**Christoph Garth** received the PhD degree in computer science from Technische Universität (TU) Kaiserslautern in 2007. After four years as a postdoctoral researcher with the University of California, Davis, he rejoined TU Kaiserslautern where he is currently a full professor of computer science. His research interests include largescale data analysis and visualization, in situ visualization, topology-based methods in visualization, and interdisciplinary applications of visualization.



**Alexander Wiebel** received his PhD from Universität Leipzig in 2008 for his research on flow visualization. He is currently a professor of visual computing at the University of Applied Sciences Worms where he is co-heading the research group UX-Vis and serves as deputy scientific director of the Center for Technology and Transfer. His research interests include scientific visualization, explainability, and AR/MR/VR. As a postdoctoral researcher, he worked at Max

Planck Institute CBS and Zuse Institute Berlin (ZIB). During this time, he also lectured at Universität Leipzig and Freie Universität Berlin. Between 2013 and 2015, he was a professor at Coburg University.