Particle-flow reconstruction with Transformer

Paul Wahlen[1,2] and *Taikan* Suehara[3]

[1]ETHZ, Zürich, Switzerland
[2]IP Paris, Paris, France
[3]The University of Tokyo, Tokyo, Japan

Abstract. Transformers are one of the recent big achievements of machine learning, which enables realistic communication on natural language processing such as ChatGPT, as well as being applied to many other fields such as image processing. The basic concept of the Transformer is to learn relation between two objects by a self-attention mechanism. This structure is especially efficient with large input samples and large number of learnable parameters. We are studying this architecture applied to the particle flow, which reconstructs particles by clustering hits at highly-segmented calorimeters. Using datasets consisting of one or two initial photons, the network is asked to predict clusters one by one using hits from the calorimeters as input. Truth clusters information are provided at learning stage to compare with the decoder output. The best models reconstructed one photon events with a relative error on the energy of 5% and direction differing from the ground truth by 2.98°. Moreover, the model achieved an accuracy of 99.6% when asked to separate one and two photons events.

1 Introduction

In order to achieve the precision required by the prospective Higgs factories, several designs of highly granular calorimeters, such as the International Linear Detector (ILD), the Silicon Detector (SiD), the Compact Linear Collider (CLIC) and the CLIC-Like Detector (CLD), are being developed. By increasing the ability of the calorimeters to separate showers from individual particles, high granularity is thus optimised for particle reconstruction using Particle Flow Algorithms (PFAs).

In PFAs, charged particles are completely reconstructed using track information, whereas photons and other neutral particles are reconstructed by clustering their corresponding hits in the calorimeters. Once this is accomplished, the reconstructed particles can be used to identify events where a specific reaction occurred, effectively reducing the background noise or used in more specialised algorithm such as jet tagging to obtain further information about the initial particles produced.

Many of the reconstruction algorithms used in experiments, such as ATLAS [1], CMS [2] and the International Linear Collider (ILC) with PandoraPFA [3], are based on explicitly instructed routine. However, in recent years, machine learning has started to be used extensively in particle physics for various tasks such as track reconstruction, jet clustering and tagging [4–6] and in particular, to develop PFAs [7–9] in the hope of developing a more computationally efficient and accurate program for event reconstruction.

In this work, an implementation in Pytorch's framework of a PFA using a neural network called a transformer is presented. Although first used in language models for translation tasks [10], transformers and their variants have also found applications in particle physics including jet tagging [11], density estimation [12] and particle reconstruction [13, 14].

2 Datasets and input features

2.1 Dataset generation and input features

Raw dataset generation

The datasets are generated using the ILD full simulation ilcsoft v02-03-01 with geometry ILD_l5_o1_v02 [15]. Thus each event in the raw datasets consists of hits from the detectors, referred to as *feats* and their associated Monte Carlo truth particles (MCTruth particles), referred to as *labels* or *clusters*.

Features selection and preprocessing

Each hit is characterised by a vector of the form $(\log_{10} E_h, x, y, z)$ where E_h is the energy deposit of the hit, and (x, y, z), its position in the calorimeter. The MCTruth particles are vectors of the form $(C, |\text{id}|, \log_{10} E_c, n_x, n_y, n_z)$, where C is its charge, $|\text{id}|$ the absolute value of its PDG number [16], E_c its energy and $(n_x, n_y, n_z) = \mathbf{n}$ its momentum at the point of interaction normalised such that $\mathbf{n}^2 = 1$. After preprocessing, the energy and the position of the hits as well as the energy of the labels are normalised over the entire set of generated events according to,

$$f_i' = \frac{1}{\sigma(f)}\left(f_i - \bar{f}\right), \qquad \sigma(f) = \sqrt{\frac{\sum_i |f_i - \bar{f}|^2}{N-1}}, \qquad \text{and} \qquad \bar{f} = \frac{1}{N}\sum_i f_i, \tag{1}$$

with f, a feature to be normalised, N the total number of hits or labels and i running over the entire set of hits or labels. Moreover, clusters are sorted by descending energy order.

2.2 Datasets

Single photon dataset:

This dataset is obtained by simulating a single initial photon passing through the detectors. Its energy follows a logarithmic distribution between 10 and 100 GeV. The direction from the point of interaction are random and 100'000 events were simulated. 80% of the dataset is used for the training set, another 10% for the validation set and the last 10% for the test set.

Due to interactions between the incoming particles and the detectors, pairs of particle/antiparticles generation and other radiations can occur. Some events thus contain more than 1 MCTruth particle and not necessarily only photons or electrons, therefore affecting the performance of the model. To try to maximise the optimisation of the energy and direction accuracy, only events containing a single photon were selected forming true single photons events.

Double photons dataset:

This dataset is obtained by simulating two photons passing through the detectors. As in the previous dataset, their energy follow a logarithmic distribution between 10 and 100 GeV and

their direction from the point of interaction are random. 100'000 events were simulated. Again, 80% forms the training set and the 20% left is shared equally between the validation and the test set. Furthermore, physical interactions can produce other particles leading to several MCTruth particles other than photons.

Mix true single photon - true double photons:

This dataset was created from the datasets containing one and two photons events. Both datasets were mixed together and randomised to create a new dataset of approximately 50% of true single photon events and 50% of true double photons events.

3 Network architecture

3.1 Embedders

Although in machine translation the embedder for the input of the encoder can be the same as for the decoder, it is not possible in this case, since processed feats and labels do not have the same dimensions, nor contain the same kind of information. Therefore, two embedders are created as two different instances of the same `Embedder` class.

The `Embedder` class implements a plain Feed Forward Network (FFN) with a variable number of layers, chosen by the user. At each layer, ReLU is used as an activation function. If the embedder consists of only one layer, the linear transformation is a map from $\mathbb{R}^n \to \mathbb{R}^{d_{model}}$, where n is the number of features of the input vector. Otherwise, the first layer is changed from being a map from $\mathbb{R}^n \to \mathbb{R}^{d_{model}}$, to a map $\mathbb{R}^n \to \mathbb{R}^{d_{ff}}$, where d_{ff} is a hyperparameter, usually chosen as $2d_{model}$. The following layers are maps from $\mathbb{R}^{d_{ff}} \to \mathbb{R}^{d_{ff}}$, except for the last one, for which it has to be $\mathbb{R}^{d_{ff}} \to \mathbb{R}^{d_{model}}$.

Once feats and labels have gone through their associated embedders, they are sent to the encoder and decoder respectively.

3.2 General architecture

The network architecture is built in complete analogy with the original version presented in [10].

In addition to hits and labels in the datasets, special tokens are added for each event. The beginning and end of each event are marked by a `bos` and `eos` token respectively, whilst `pad` tokens are used to pad all events to the same length. Hits/labels are distinguished from other special tokens by adding an extra dimension to their vector, whose last entry is set to 3. The other special tokens are constructed as vectors of the same dimension as the hits/labels and of the form $(0,\ldots,0,s)$, with s equal to 0 for `pad` tokens, 1 for `bos` and 2 for `eos`. The set $\{0,1,2,3\}$ forms the vocabulary for the class of token. Furthermore, two additional vocabularies are obtained by sorting in ascending order the unique values of charges and PDGs found in the dataset and associating them to integers starting from 1 onwards. In these two vocabularies the value 0 is given to all special tokens.

As illustrated in figure 1, both hits and labels first go through the preprocessing phase, described in section 2.1, before being embedded in a higher dimensional space by the embedders. The embedding of the hits are fed to the encoder and the embedding of the labels to the decoder. The output of the encoder supplies the keys and values for the second layer of multi-head attention in the decoder. The final output of the decoder is acted upon by four linear transformations, f_C, f_P, f_T and f_E, to produce the logits corresponding to the charge, PDG, class and continuous DOFs respectively. For f_C, f_P and f_T, their outputs are interpreted

as the unnormalised probabilities that the quantity associated to the index i in the vocabulary is next, such that $f_C : \mathbb{R}^{d_{model}} \to \mathbb{R}^{l_C}$ and $f_P : \mathbb{R}^{d_{model}} \to \mathbb{R}^{l_P}$, where $l_{C/P}$ are the lengths of the charge/PDGs vocabularies and lastly, $f_T : \mathbb{R}^{d_{model}} \to \mathbb{R}^4$. During training, these outputs are compared to the ground truth using Pytorch's CrossEntropyLoss function, ignoring the padding tokens.

As the vector for the cluster direction should be normalised to 1, f_E is defined as a map $f_E : \mathbb{R}^{d_{model}} \to \mathbb{R}^3$, with the first entry of its output interpreted as the normalised log base 10 of the energy, and the two others as θ and ϕ respectively. In order to avoid problems from the non-periodicity of the Mean Squared Error (MSE) loss function, these angles are converted to 3D cartesian coordinates, which are then compared to the ground truth.

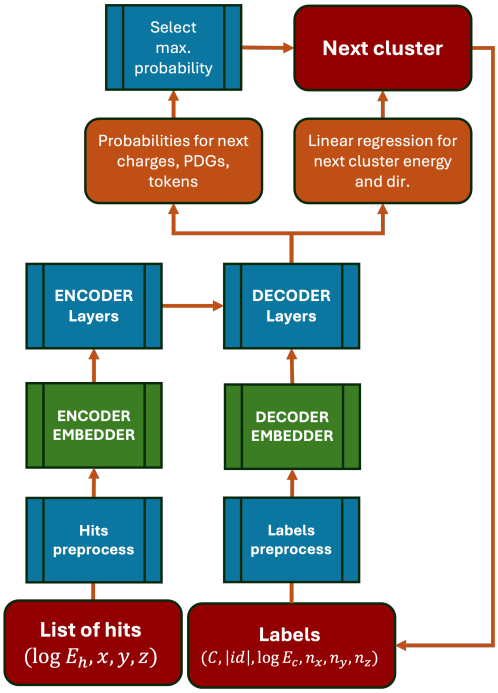

Figure 1: Architecture of the neural network. The hits are first sent to the embedder of the encoder, then supplied to the encoder. The output of the latter is connected to the decoder supplying the keys and values vectors for the second multi-head attention in the decoder. The clusters are first sent through another embedder, before being supplied to the transformer's decoder. The final output of the decoder is used to produce probabilities for the next discrete DOFs, and next energy and spherical angles.

4 Results and discussion

From the predictions obtained by running inference on the test set with the trained model, different quantities were computed to assess the model performance by establishing a one-to-one correspondence between the overlapping sample tokens present in the predictions and labels. If p_{ij} is a predicted feature for the event i and cluster j, then it will be compared with l_{ij}, if it exists, which is the same feature for the j^{th} truth cluster in event i. Since the number

of predicted clusters can be either smaller, equal or larger than the true number of clusters, this correspondence imposes constraints j, whose maximum value is therefore defined as

$$j_{i,max} = \min(n_{ip}, n_{il}), \tag{2}$$

Where n_{ip} and n_{il} are the predicted number of clusters for event i in the predictions and labels respectively. This is justified as the clusters in the labels are sorted in order of descending energy. Note that before establishing this correspondence, the special tokens were already filtered out for both predictions and labels.

From this one-to-one correspondence, several quantities are computed to assess the model performance. The charges and PDGs accuracies for one event are computed by attributing a value of 1 to a pair prediction-label if they have the same charge/PDG and 0 otherwise. Both accuracies are then obtained by summing over these binary values and dividing by the number of pairs in the event. The excess of number of predictions is computed as the difference $n_{ip} - n_{il}$. A positive number therefore indicates that the model has predicted more clusters than what was contained in the labels. Energy accuracy is computed as the absolute relative error between the predicted value and truth. Lastly, the angle θ represents the angle separating the predicted direction and the truth. Its cosine value is directly obtained from the vectors scalar product, since their norm is normalised to 1.

The ADAM optimiser [17] and a constant learning rate of 10^{-4} were used for training. Both embedders had one hidden layer with $d_{ff} = 2d_{model}$ and a final size of d_{model}. The FFNs in the transformer were constructed from one hidden layer of dimension d_{ff} and the size of their final output was set to d_{model}. Morevoer, the four individual losses are summed to form the total loss with weights shared between them according to two scenarios. To improve performance on the prediction of the right number of clusters, a weight of 0.7 is given to the loss associated to the prediction of the class of token. The remaining 0.3 is then shared equally between three other losses. This scenario is later on referred to as the 70% tokens loss function. In the case of the 70% continuous loss function, it is the loss associated to the continuous DOFs that is summed with a weight of 0.7 whilst the remaining 0.3 is shared equally between the other losses.

4.1 True single photon events

The best performances were obtained for $d_{model} = 64$, with a relative error on the energy for the token and continuous losses of 9.7 % and 5 % respectively. Moreover, a mean 3D angle for the token and continuous losses of 4.08 ° and 2.98 ° were obtained respectively.

In figure 2 are shown the relative error on the enery for the predictions obtained with the 70% continuous loss function and classified according to the energy range in which the label is. Indeed, the energy given in the hits is the one as provided by the detector and therefore comes with uncertainty. In other words, taken in their raw forms, a simple addition of all the energy of the hits would not give the same result as the one provided in the labels. Uncertainty on the energy stems from two reasons. First, there is the intrinsic uncertainty on the energy of the photon, which increases for lower energies. Second, there is the detector resolution, σ, defined as the one sigma deviation from the true energy of the particle:

$$\sigma \sim \frac{1}{\sqrt{E}}, \tag{3}$$

with E the energy of the incoming particle. From these considerations, it is expected that photons in the 20-50 GeV range would suffer from larger uncertainty leading to a worse reconstruction by the network and thus a higher RMS value. Inversely, photons in the 50-100 GeV

range are expected to suffer less from this uncertainty and therefore be better reconstructed. The mean, RMS value and number of events for each category of energy range are sum-

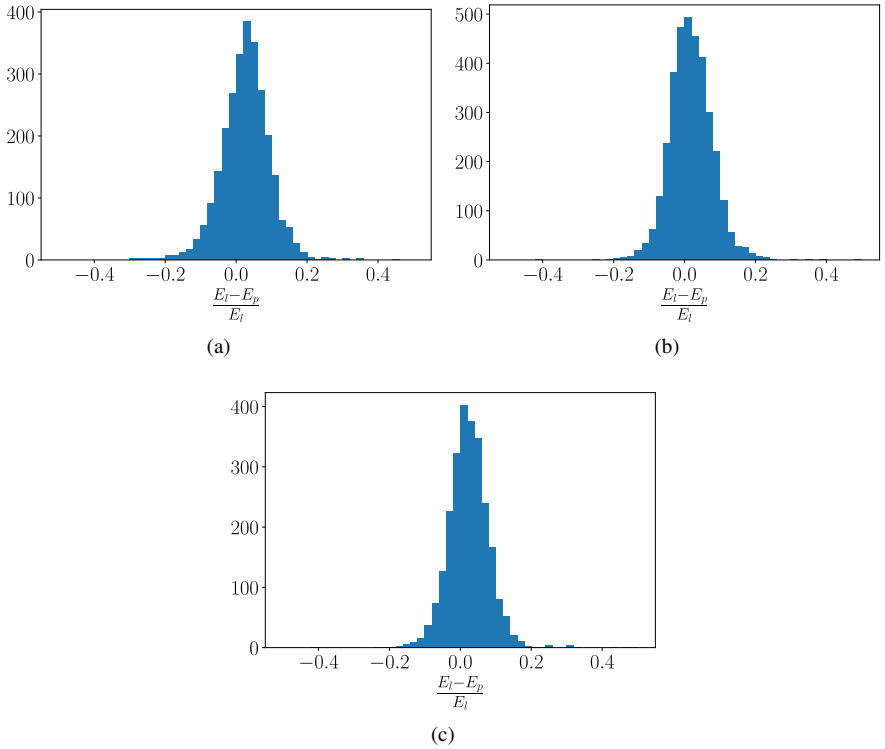

Figure 2: Relative error on the energy computed using predictions from a network with $d_{model} = 64$ trained on true single photon events and classified by energy range a) 10-20 GeV b) 20-50 GeV and c) 50-100 GeV.

marised in table 1. At first, figure 2 and table 1 seem to suggest that both the mean and the RMS values of the 50-100 GeV range are worse than the 20-50 GeV range.

Table 1: Mean and RMS values of the distribution of the relative error on the energy, computed on three energy ranges. Models trained on true single photon events, with 70% of the total loss on continuous DOFs.

Energy Range [GeV]	Mean [%]	RMS [%]	events
10-20	2.25	10.18	2714
20-50	1.83	6.05	3501
50-100	2.45	6.20	2540

Since the energy distribution follows a logarithmic distribution, the higher the energy, the rarer the event, which could indicate that the range 50-100 GeV is underrepresented in the dataset, decreasing the ability of the model to predict the correct energy. Although accounting

for 50% of the energy range, the 50-100 GeV range represents only 29% of the total number of events. This can be easily tested by training the model on a smaller fraction of the dataset.

On the other hand, the evolution of the difference in log values between the predictions and labels plotted as a function of the label energy in figure 3 seems to suggest that the worsening of the mean and the RMS present in table 1 take their roots not in a general worsening of the predictions of the model but rather to a moving bias with a relatively small RMS value. Whereas a higher bias could be linked to the higher energies being less represented in the dataset, it is more difficult to attribute the lower RMS value to a better reconstruction, due to the presence of bias and the small number of events which could create the illusion of a smaller RMS. This logarithmic distribution of the energy was chosen to reflect physical dis-

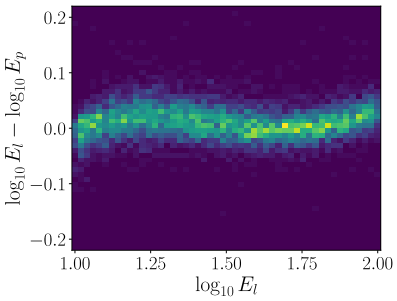

Figure 3: Difference in log values of predicted and label energies as a function of the log value of the label energy. Models trained on true single photon events with 70% of the total loss weight on the continuous DOFs loss function.

tributions, however taking the previous considerations into account, generating datasets with a uniform energy distribution could be used to improve energy accuracy at higher energies.

4.2 Mix true single photon - true double photons

The objective is to observe whether the model is capable of distinguishing one particle events from two particles events. Six models with embedding sizes $d_{model} = 64, 128$ and 256 (two models of each) were created. The first model of each size was trained in the 70% continuous loss function scenario, whereas the second model was trained in the 70% tokens loss function. The performance of the models was assessed by constructing the matrices illustrated on Fig. 4 for the 70% tokens loss function and Fig. 5 for the 70% continuous loss function. The columns of the matrices represent the number of clusters predicted by the labels. The rows are the number of clusters in the labels. Thus, the entry on the first column and first row represents the number of events for which the model correctly predicted one cluster. On the second column on the same row are shown the number of events for which the model predicted incorrectly two clusters. Lastly, on the third column are given the number of events for which the model predicted a number of clusters either smaller than 1 (predicting an `eos` token directly) or larger than 2. The second row is constructed analogously but by considering events containing two clusters in the labels.

All models were able to distinguish between one or two particles events with high accuracy, with the best accuracy obtained for both loss functions for $d_{model} = 128$ at 98.2% for continuous and 99.6% in the tokens case, therefore bringing a significant improvement. Moreover, most of the incorrect predictions were obtained for a number of labels clusters of 2, with the highest values obtained for a prediction of one cluster instead of two. This is

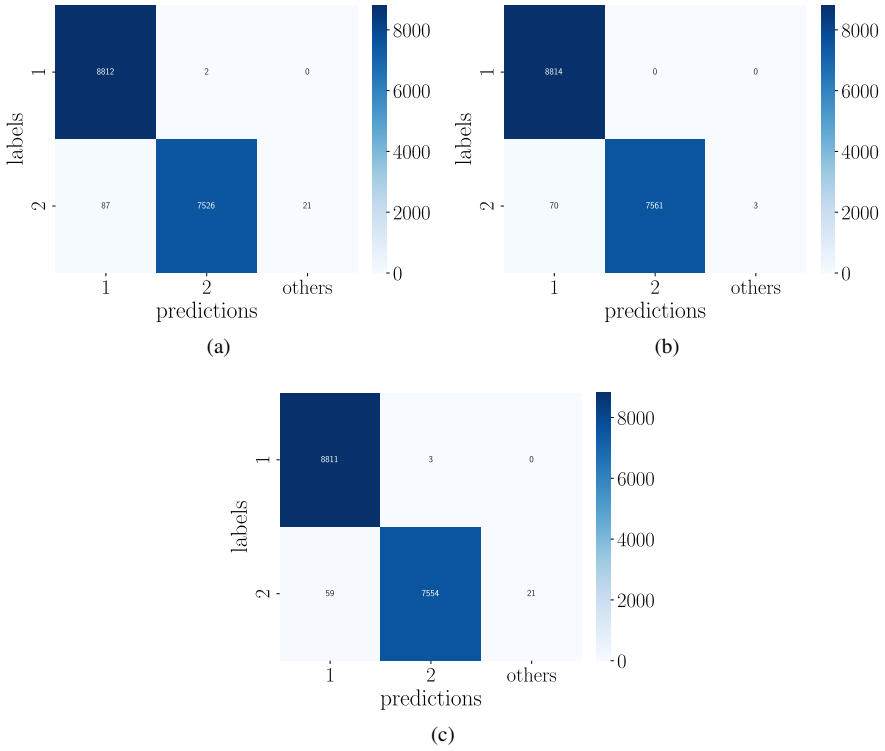

Figure 4: Classifications of the number of events depending on the number of clusters contained in the labels and the number predicted by the models, obtained for the 70% tokens loss and with d_{model} = a) 64 b) 128 and c) 256. Models trained on a mix of true single and true double photons events.

probably due to the fact that in some events, the two clusters are too close to each other to be distinguished by the model.

To measure the ability to distinguish between two close-by clusters, the model obtained for $d_{model} = 128$ with the tokens loss function was used for inference on a dataset containing true double photons whose directions differ by an angle of 100 mrad. The model predicted the two expected clusters in 11.4% of events, with the rest of the predictions being one cluster. In retrospect, such a low accuracy compared to the previous results is not a surprise. Indeed, the average angle accuracy computed for this model on the mixed dataset is 8.64 °, which exceeds the 100 mrad = 5.72 °. For the energy, the mean relative error is 9.1 %. The best model for the continuous loss achieved a mean angle of 8.16 ° and mean relative error on the energy of 11.8 %. This seems to suggest that the predictions of the directions and energy are still somewhat disconnected from the clustering process.

Overall, these results show that clustering from the transformer is possible in this simple scenario, although more precision is still required to resolve correctly clusters close to each other.

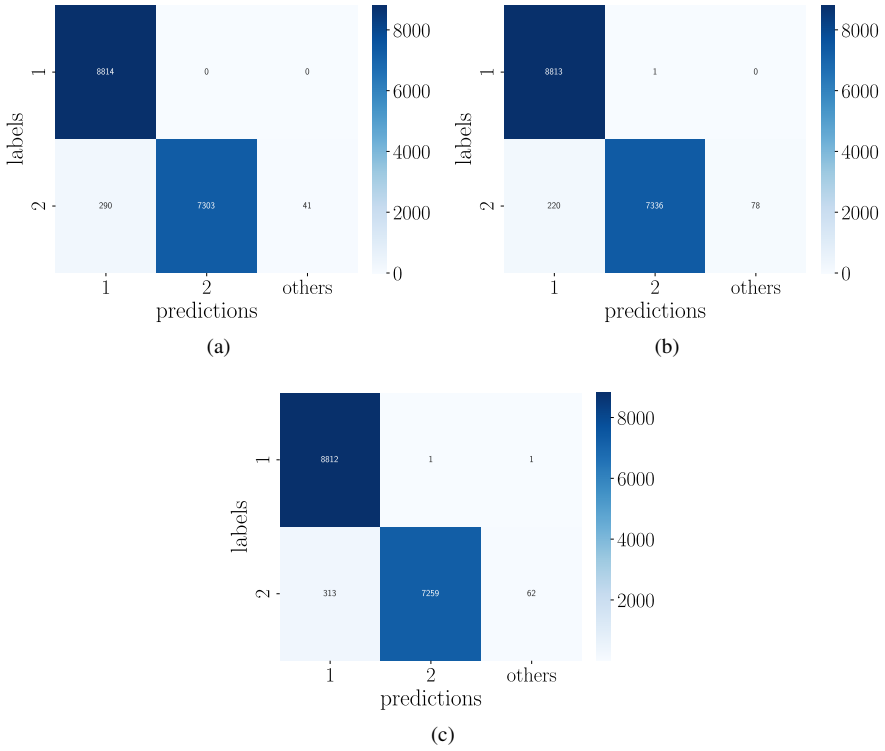

Figure 5: Classifications of the number of events depending on the number of clusters contained in the labels and the number predicted by the models, obtained for the 70% continuous loss and with d_{model} = a) 64 b) 128 and c) 256. Models trained on a mix of true single and true double photons events.

5 Conclusion

This work presented basic implementations of a transformer as a Particle Flow Algorithm using Pytorch's framework, built in complete analogy with the original transformer architecture. The objective was to predict sequentially the visible particles by feeding the network with hits' energy and position from the calorimeters. The input was embedded into higher dimensional vectors by fully connected FFNs before being used as an input to the transformer. To differentiate input data from special tokens used either for indicating the beginning or end of the sequence, or just as padding, an additional integer was added to the tokens initial representation. Labels were characterised by the charge of the cluster, the absolute value of the particle ID, and continuous DOFs: its energy and direction.

As could be expected from the simplicity of the datasets used for training, the model was able to correctly predict, in most cases, the right number of clusters as well as their charge, and PDG number. Clusters produced by physical interactions during the simulation lowered all above accuracies.

The performance for the continuous DOFs showed relatively high bias and variance for models trained on true single photon events, especially at higher energies. It is possible that this stems from the higher energy events being underrepresented in the dataset.

In parallel, the clustering ability of the network was tested by constructing a dataset consisting of a mix of true single photon and true double photon events. The best model was able to predict the correct number of clusters in 99.6% of the cases. Nonetheless, since energy and angle accuracy are relatively biased (9.1% and 8.16 ° resp.), this suggests that the clustering process is still disconnected from the way the model predicts the energy and direction of the cluster. Moreover, two clusters resolution is still limited, with the model being able to resolve showers separated by an angle of 100 mrad in only 11.4% of the cases.

The fact that energy and angle accuracy seems to be disconnected from the clustering process as well as the apparent ineffectiveness of supplied track information could point towards a problem of hidden representation between the hits and labels. For the attention mechanism to make sense, the embedding for both keys and queries needs to be such that it results in a meaningful scalar-product. In the hope to construct a common embedding space between labels and memory, the encoder could be pre-trained to predict the number of clusters using a similar architecture as detailed in [18].

References

[1] M. Aaboud, G. Aad, B. Abbott, J. Abdallah, O. Abdinov, B. Abeloos, S.H. Abidi, O.S. AbouZeid, N.L. Abraham, H. Abramowicz et al., Jet reconstruction and performance using particle flow with the ATLAS Detector, **77**, 466 (????). 10.1140/epjc/s10052-017-5031-2

[2] A. Sirunyan, A. Tumasyan, W. Adam, E. Asilar, T. Bergauer, J. Brandstetter, E. Brondolin, M. Dragicevic, J. Erö, M. Flechl et al., Particle-flow reconstruction and global event description with the CMS detector, **12**, P10003 (????). 10.1088/1748-0221/12/10/P10003

[3] M.A. Thomson, Particle flow calorimetry and the PandoraPFA algorithm, **611**, 25 (????). 10.1016/j.nima.2009.09.009

[4] S. Farrell, P. Calafiura, M. Mudigonda, Prabhat, D. Anderson, J.R. Vlimant, S. Zheng, J. Bendavid, M. Spiropulu, G. Cerati et al., Novel deep learning methods for track reconstruction, 1810.06111

[5] H. Qu, L. Gouskos, Jet tagging via particle clouds, **101**, 056019 (????). 10.1103/PhysRevD.101.056019

[6] S. Thais, P. Calafiura, G. Chachamis, G. DeZoort, J. Duarte, S. Ganguly, M. Kagan, D. Murnane, M.S. Neubauer, K. Terao, Graph Neural Networks in Particle Physics: Implementations, Innovations, and Challenges, 2203.12852, http://arxiv.org/abs/2203.12852

[7] J. Duarte, J.R. Vlimant (????), pp. 387–436, 2012.01249

[8] J. Pata, J. Duarte, J.R. Vlimant, M. Pierini, M. Spiropulu, MLPF: Efficient machine-learned particle-flow reconstruction using graph neural networks, **81**, 381 (????). 10.1140/epjc/s10052-021-09158-w

[9] F.A. Di Bello, S. Ganguly, E. Gross, M. Kado, M. Pitt, L. Santi, J. Shlomi, Towards a Computer Vision Particle Flow, **81**, 107 (????), 2003.08863. 10.1140/epjc/s10052-021-08897-0

[10] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention Is All You Need, 1706.03762, http://arxiv.org/abs/1706.03762

[11] H. Qu, C. Li, S. Qian, Particle Transformer for Jet Tagging, 2202.03772, http://arxiv.org/abs/2202.03772

[12] T. Finke, M. Krämer, A. Mück, J. Tönshoff, Learning the language of QCD jets with transformers, **2023**, 184 (????). 10.1007/JHEP06(2023)184

[13] S. Qiu, S. Han, X. Ju, B. Nachman, H. Wang, A Holistic Approach to Predicting Top Quark Kinematic Properties with the Covariant Particle Transformer, **107**, 114029 (????), 2203.05687. 10.1103/PhysRevD.107.114029

[14] F.A. Di Bello, E. Dreyer, S. Ganguly, E. Gross, L. Heinrich, A. Ivina, M. Kado, N. Kakati, L. Santi, J. Shlomi et al., Reconstructing particles in jets using set transformer and hypergraph prediction networks, **83**, 596 (????), 2212.01328. 10.1140/epjc/s10052-023-11677-7

[15] T.I. Collaboration, International Large Detector: Interim Design Report, 2003.01116

[16] L. Garren, I.G. Knowles, T. Sjöstrand, T. Trippe, Monte carlo particle numbering scheme, **15**, 205 (????). 10.1007/BF02683426

[17] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, 1412.6980

[18] H. Touvron, M. Cord, A. Sablayrolles, G. Synnaeve, H. Jégou, Going deeper with Image Transformers, 2103.17239, http://arxiv.org/abs/2103.17239