PIPPIN: Generating variable length full events from partons

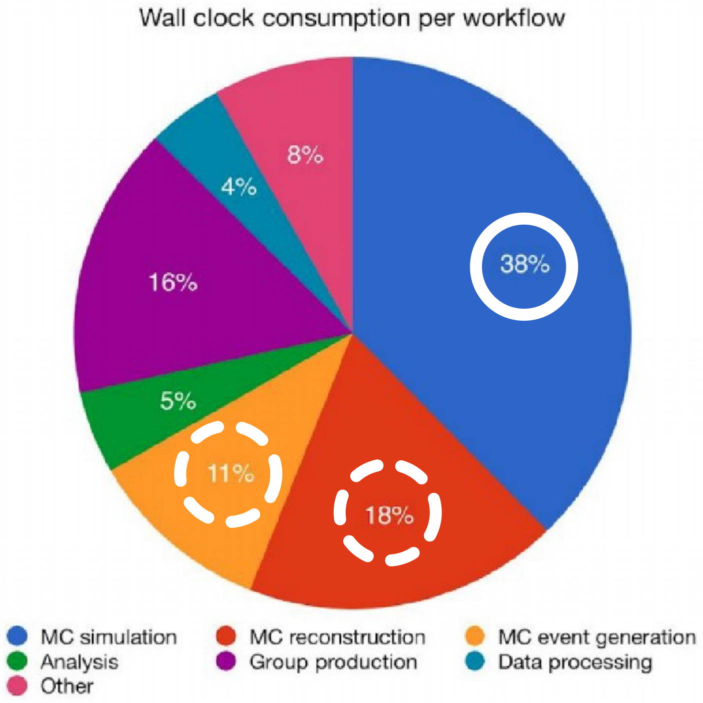
Guillaume Quétant, John Andrew Raine, Matthew Leigh, Debajyoti Sengupta, Tobias Golling

Presenter; Henry Day-Hall
13/09/2024

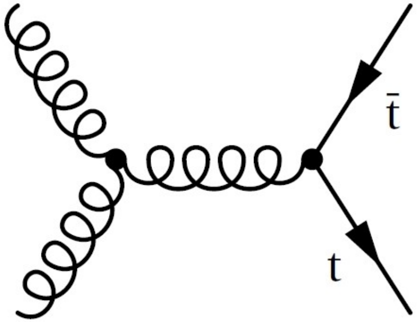
Background

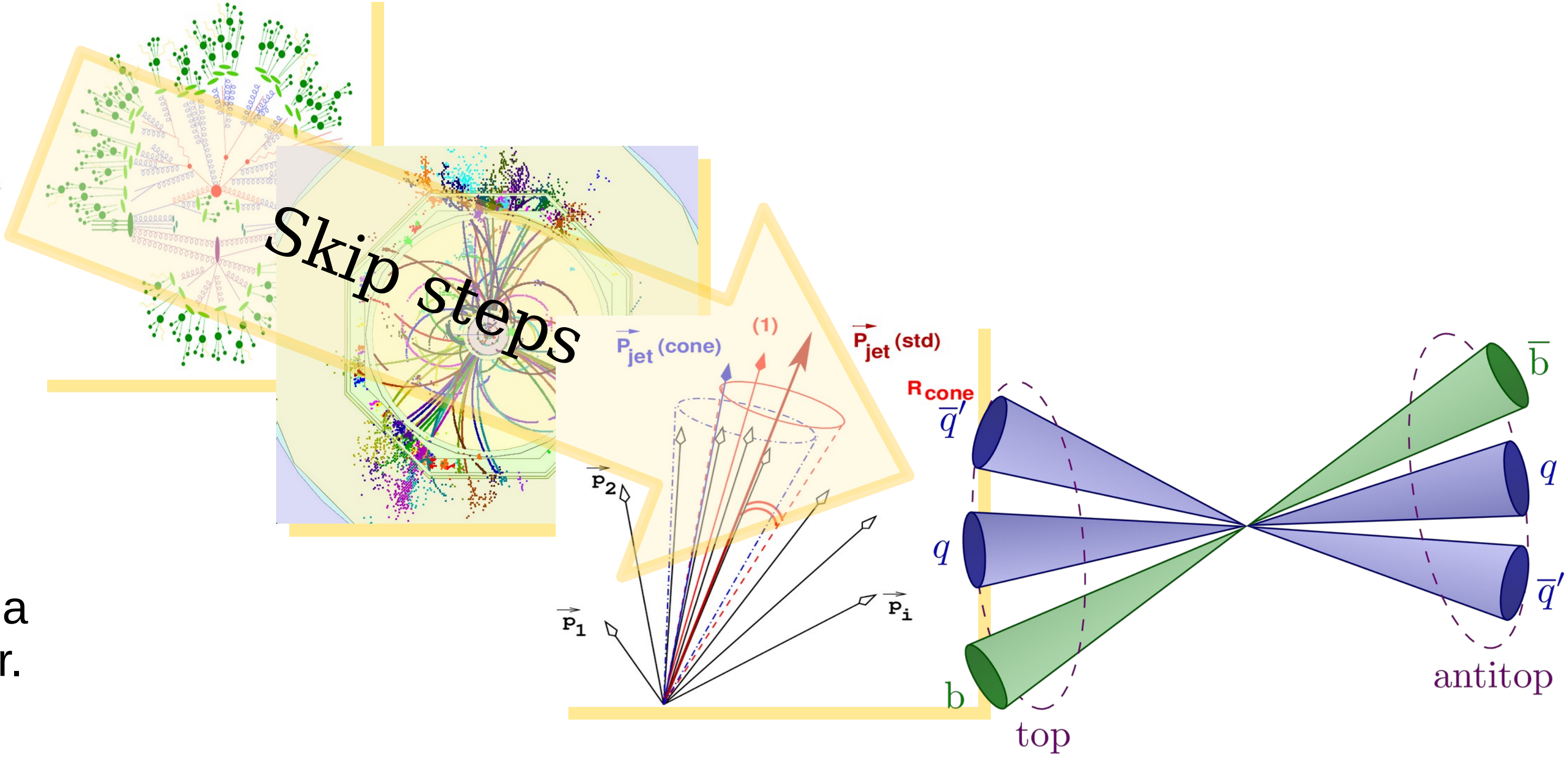
- Simulating collider events happens in aprox 4 steps;
  - Hard process (matrix element calculation)
  - Hadronisation + radiation
  - **Detector interactions** < computationally exspensive
  - Reconstruction
- The model in this paper generates partia reconstructed output with conditioning from the particles generated before hadronisation and radiation.

ATLAS compute 2018



In pictures;

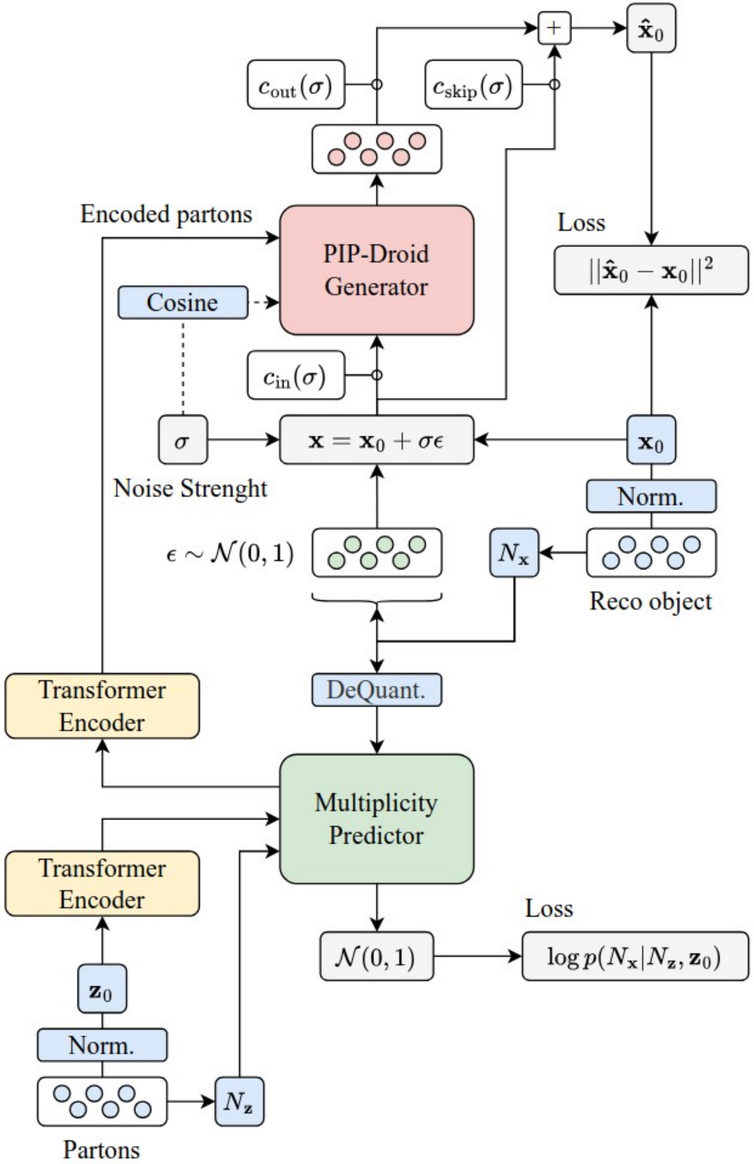




Skip steps

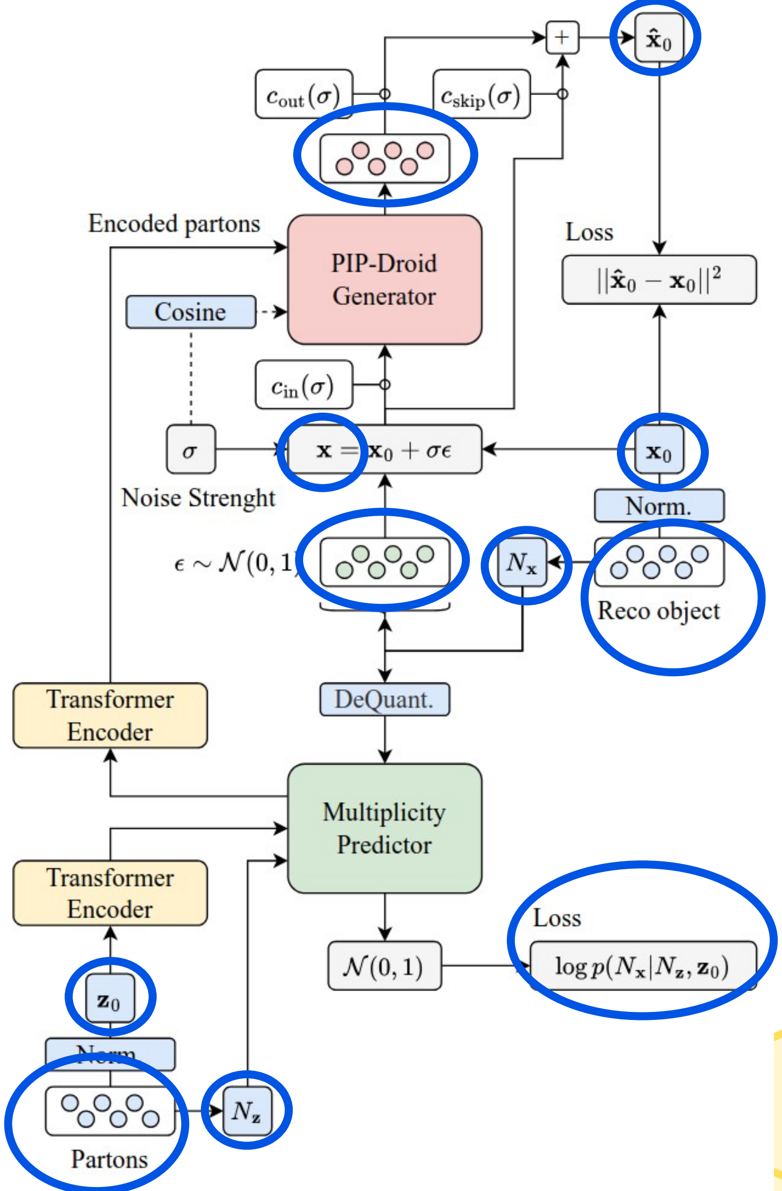
They are actually using Dephes as a detector simulator.

ML models used



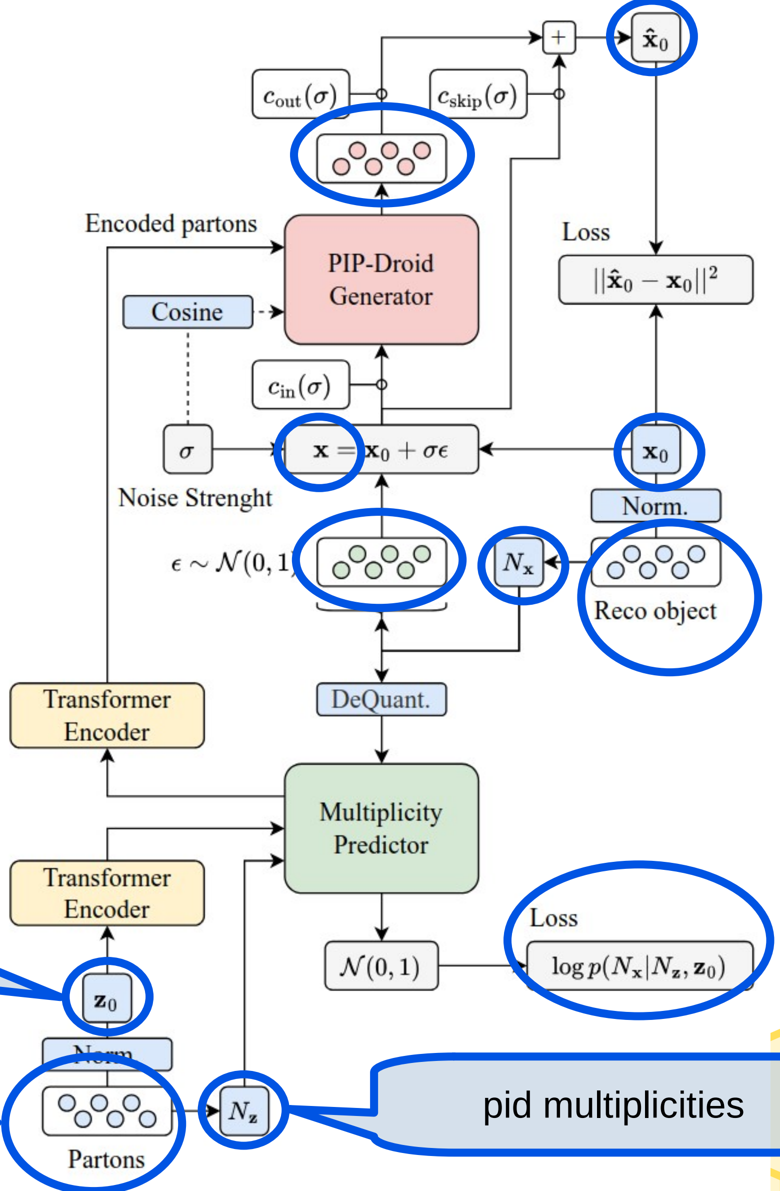
ML models used

pid = particle-id,
reco = reconstructed



ML models used

pid = particle-id,
reco = reconstructed



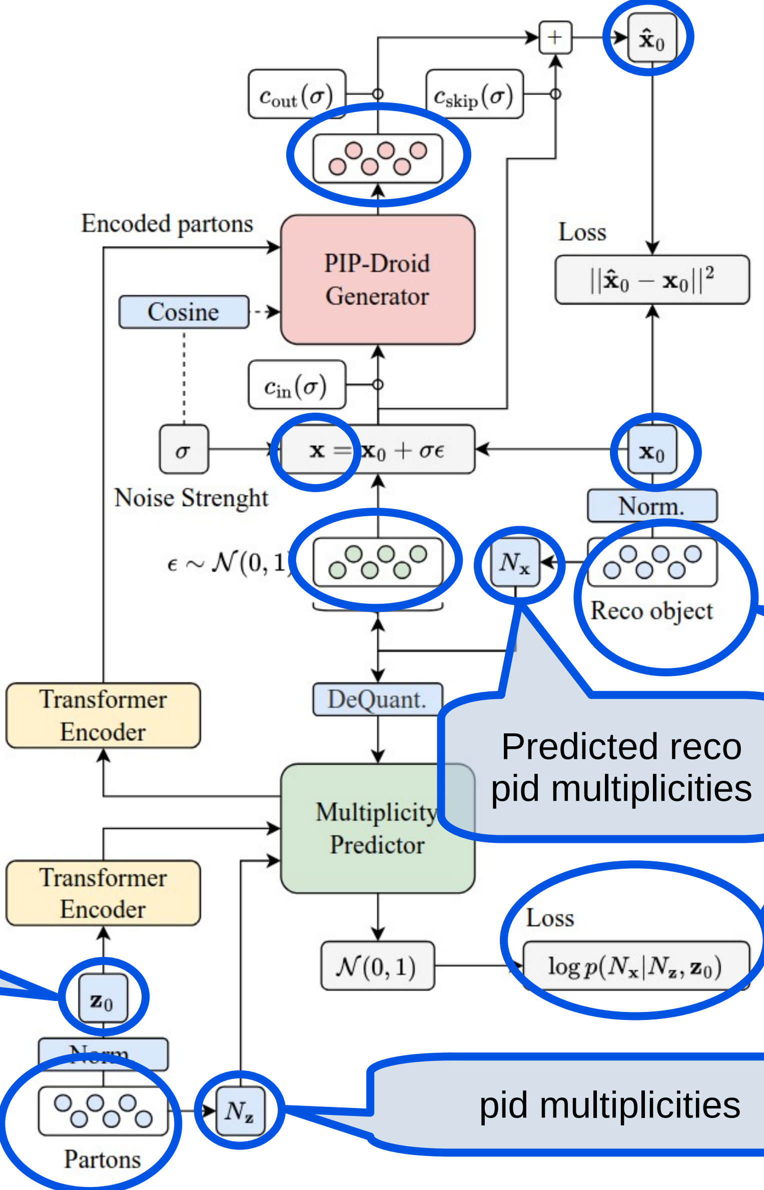
N × normalised
(px, py, pz, e, pid)
“Input token”

N × (px, py, pz, e, pid)

pid multiplicities

ML models used

pid = particle-id,
reco = reconstructed



N reco × (px, py, pz, e, pid) reco

Predicted reco pid multiplicities

Log prob of predicted reco pid multiplicities

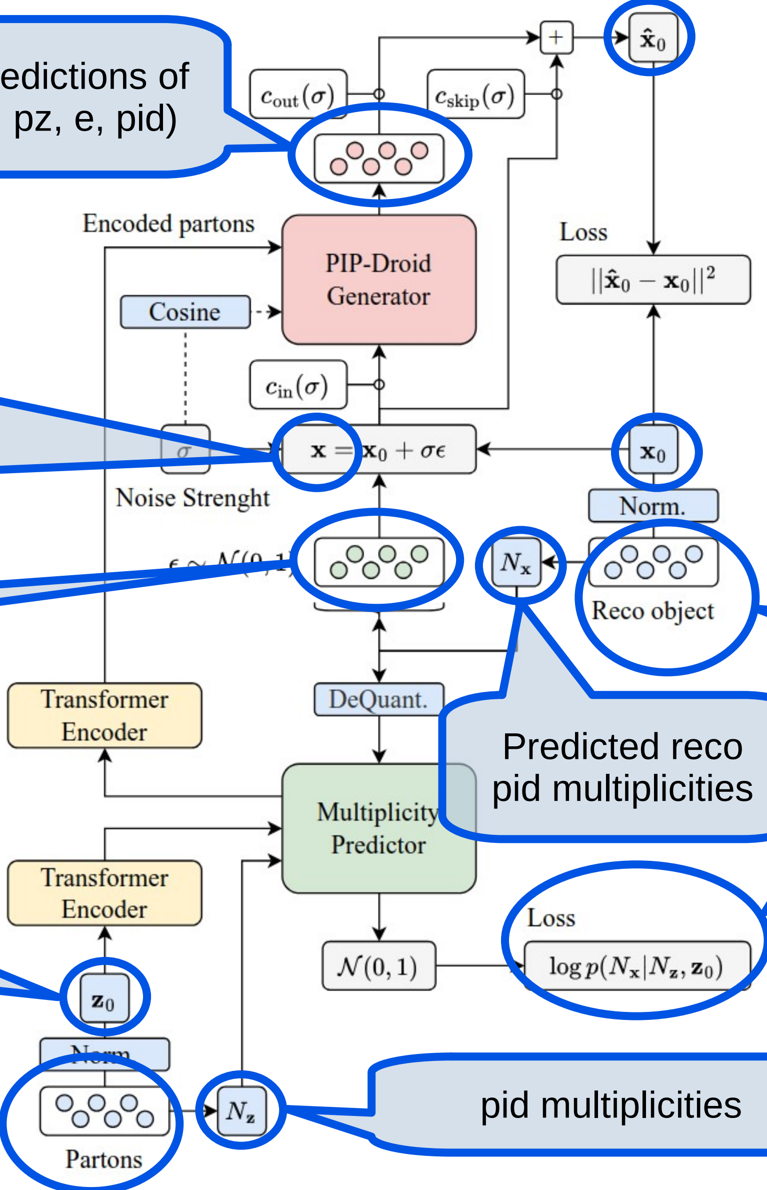
N × normalised (px, py, pz, e, pid) “Input token”

N × (px, py, pz, e, pid)

pid multiplicities

ML models used

pid = particle-id,
reco = reconstructed



Raw predictions of (px, py, pz, e, pid)

Noised reco (px, py, pz, e) with pid “learnable tokens” from the Multiplicity predictor.

Noise

N reco × (px, py, pz, e, pid) reco

Predicted reco pid multiplicities

Log prob of predicted reco pid multiplicities

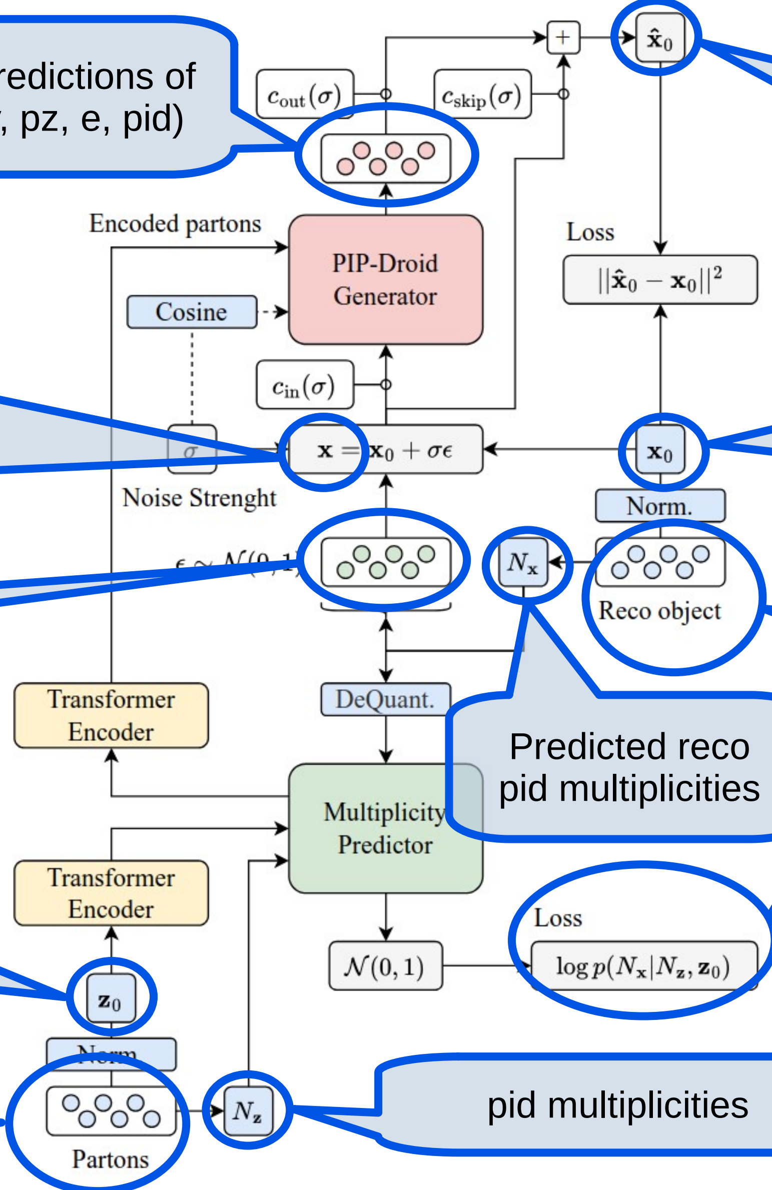
N × normalised (px, py, pz, e, pid) “Input token”

N × (px, py, pz, e, pid)

pid multiplicities

ML models used

pid = particle-id,
reco = reconstructed



Raw predictions of (px, py, pz, e, pid)

predicted
N reco × normalised
(px, py, pz, e, pid) reco

Noised reco
(px, py, pz, e)
with pid
“learnable tokens”
from the Multiplicity
predictor.

N reco × normalised
(px, py, pz, e, pid) reco

Noise

N reco × (px, py, pz, e, pid) reco

Predicted reco
pid multiplicities

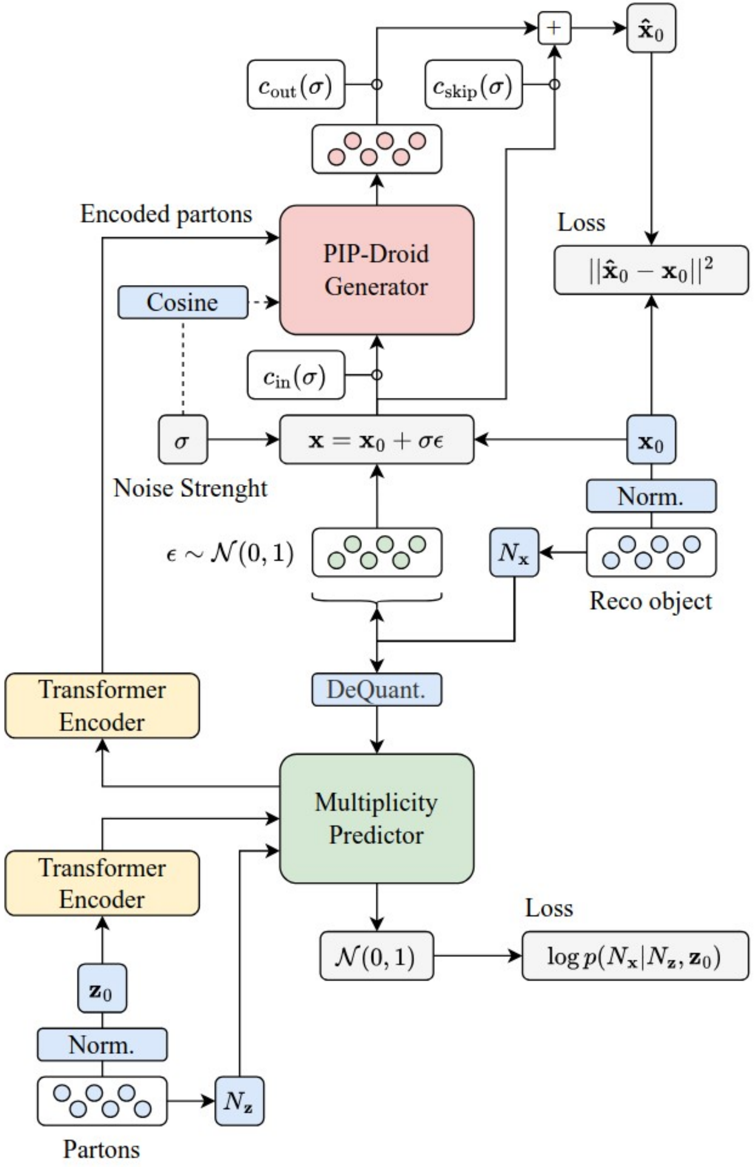
Log prob of predicted
reco pid multiplicities

N × normalised
(px, py, pz, e, pid)
“Input token”

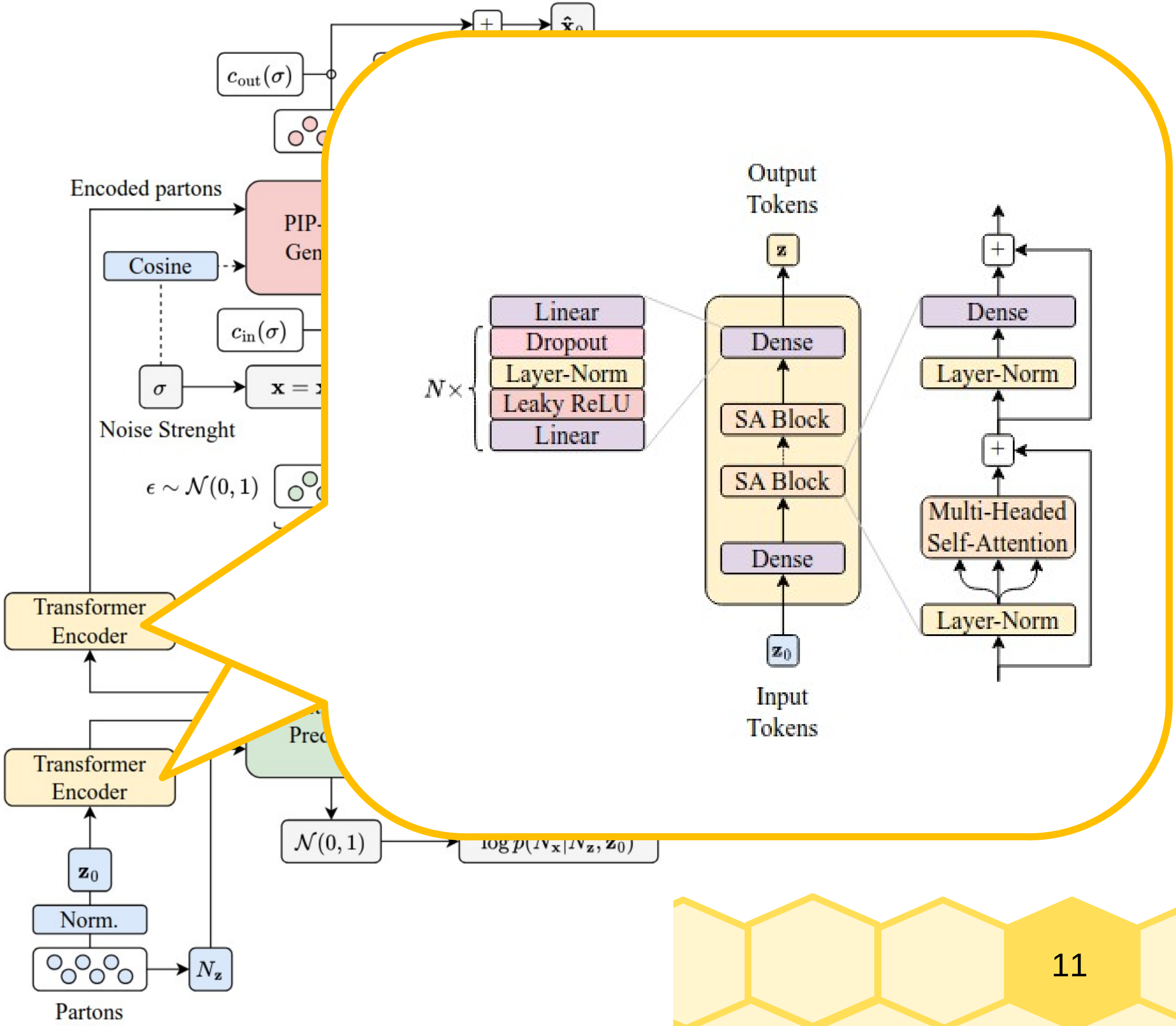
N × (px, py, pz, e, pid)

pid multiplicities

ML models used

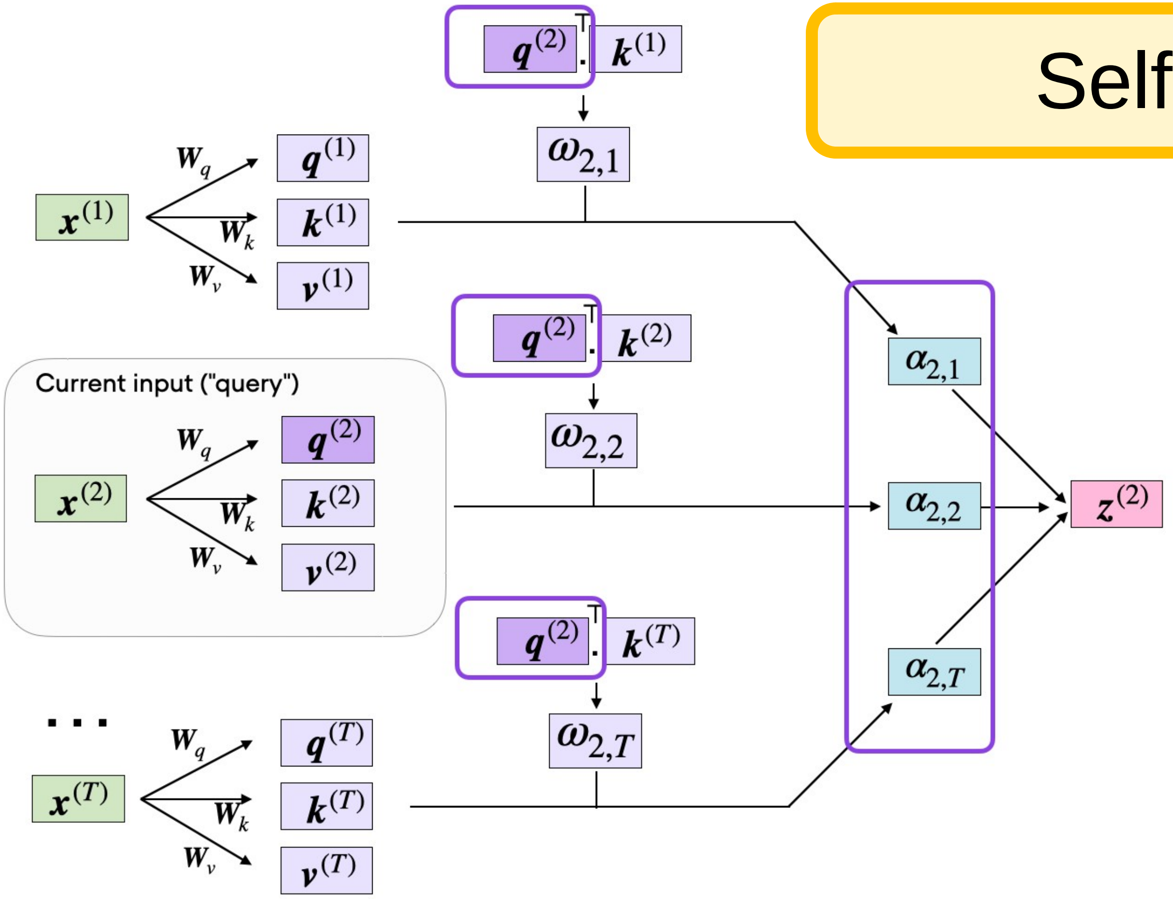


ML models used



ML models used

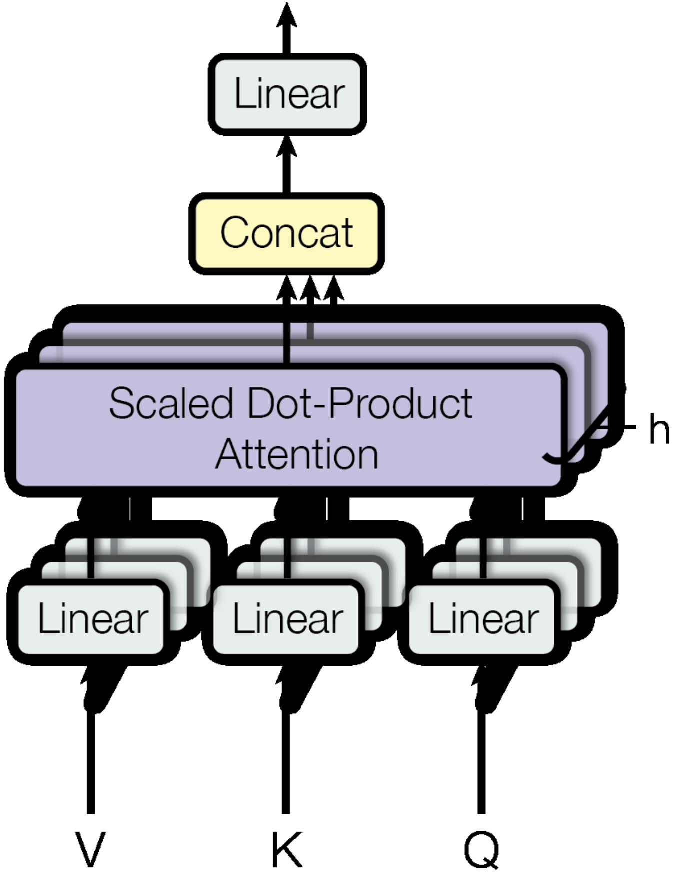
Self attention



where $z^{(2)} = \sum_{j=1}^{T} \boxed{\alpha_{2,j}} \, v^{(j)}$

https://sebastianraschka.com/blog/2023/self-attention-from-scratch.html

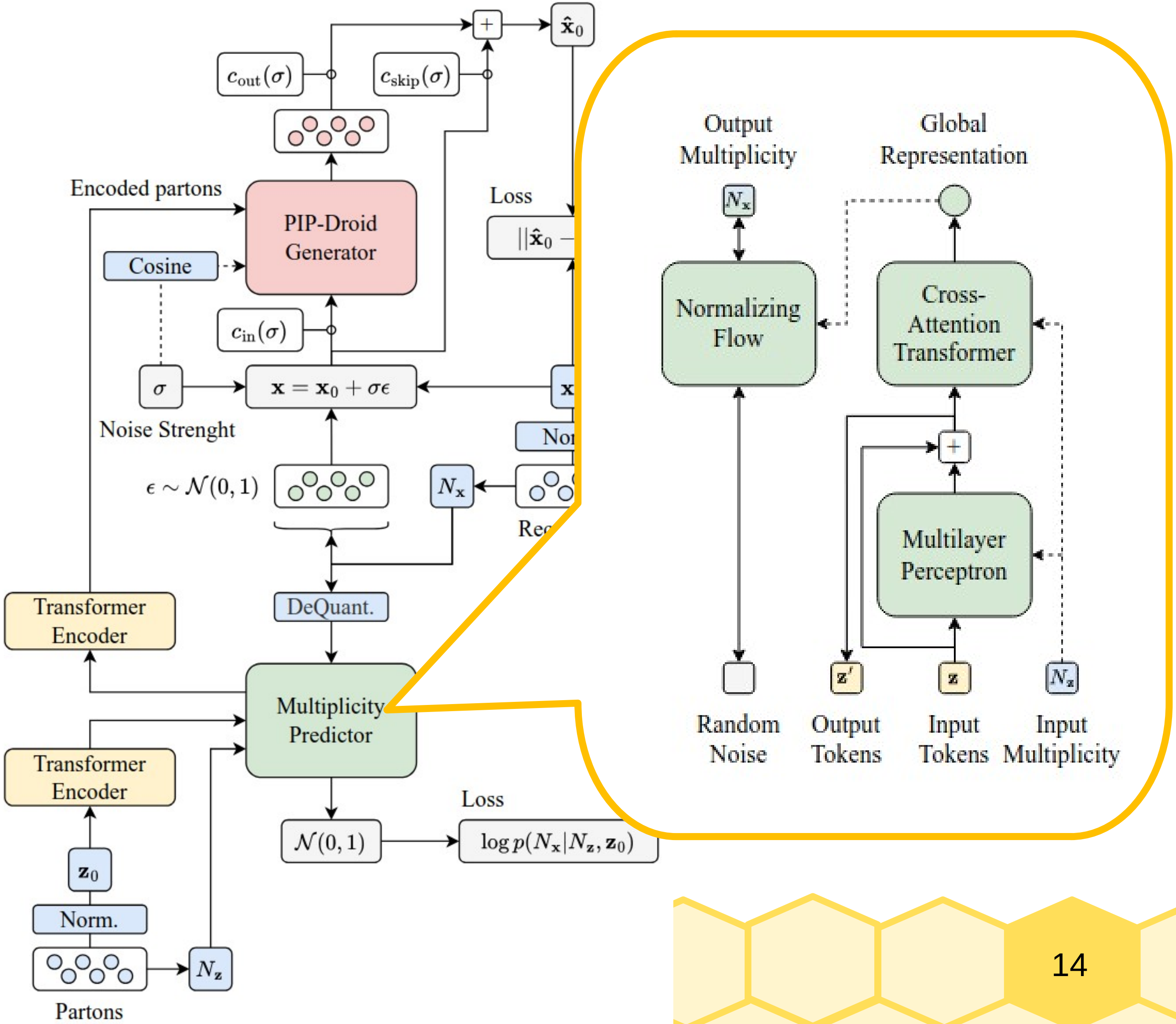
Self attention



Multiheaded self attention performs several independent self attention computations in parralel, then combines the results with a weighted sum.

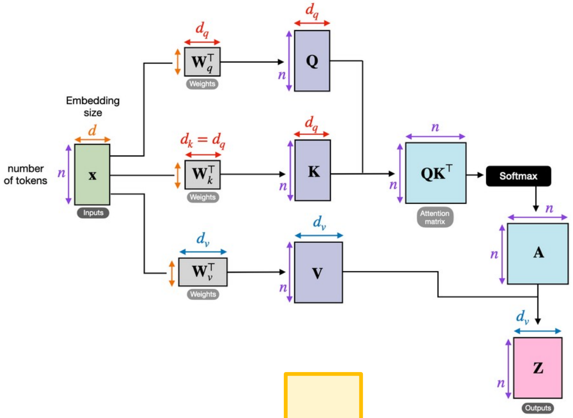
https://uvadlc-notebooks.readthedocs.io/en/latest/tutorial_notebooks/tutorial6/Transforr

ML models used

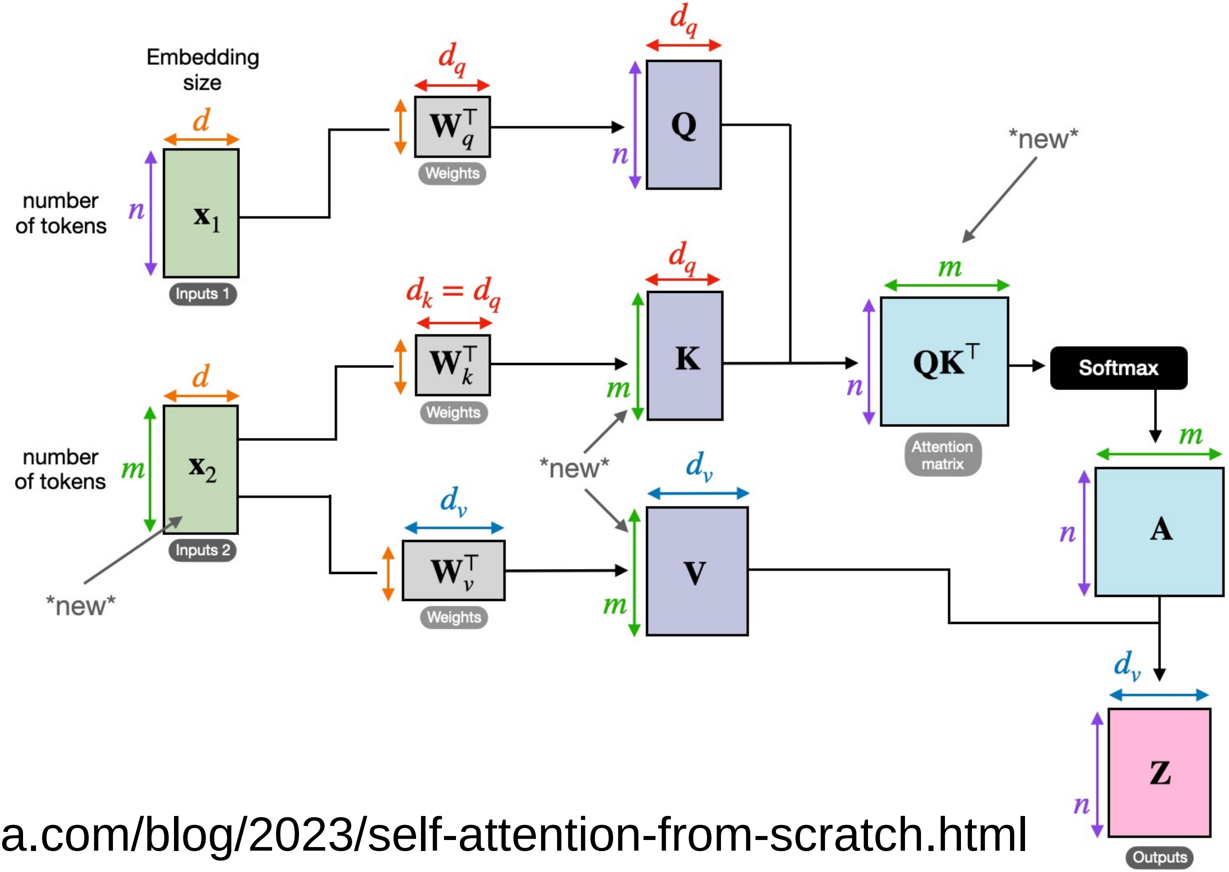


ML models used

Cross attention



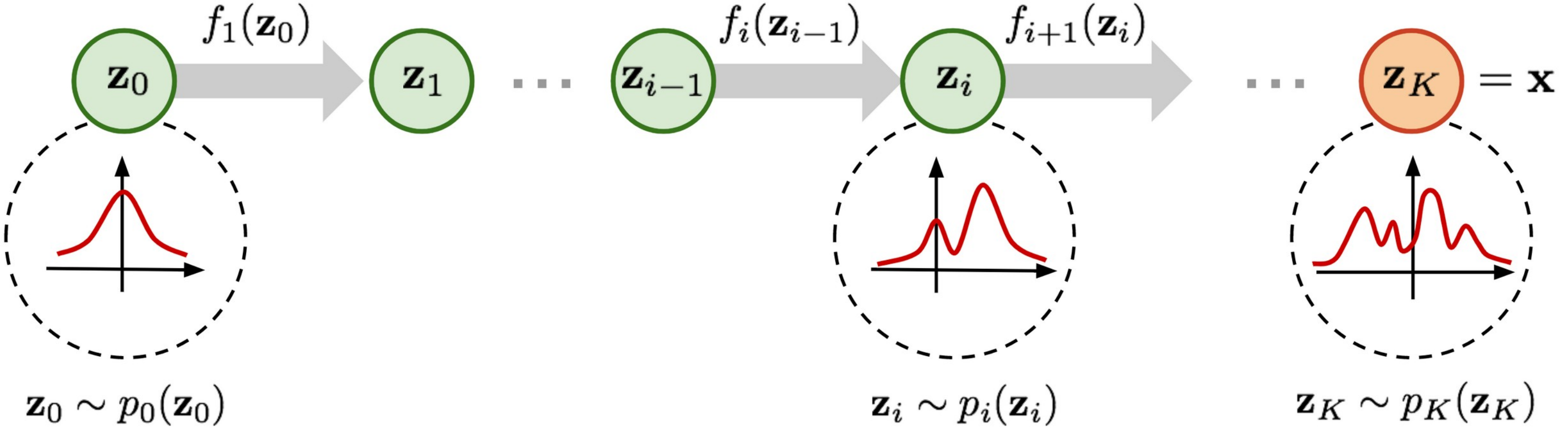
Evaluate sequence 2, using the queries of sequence 1



https://sebastianraschka.com/blog/2023/self-attention-from-scratch.html

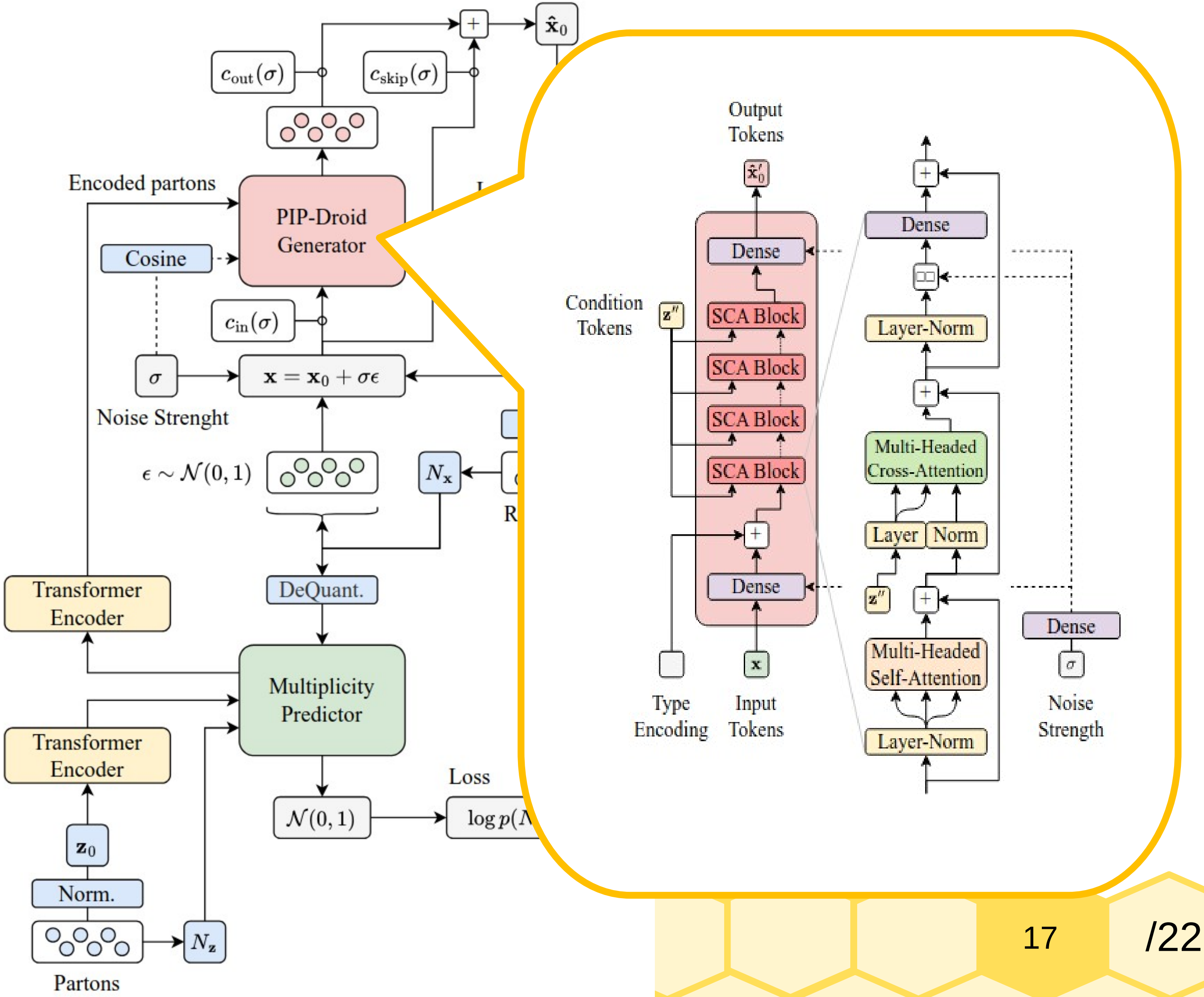
ML models used

Normallising flow


$f_1(\mathbf{z}_0)$
$f_i(\mathbf{z}_{i-1})$
$f_{i+1}(\mathbf{z}_i)$
$\mathbf{z}_0$
$\mathbf{z}_1$
$\mathbf{z}_{i-1}$
$\mathbf{z}_i$
$\mathbf{z}_K = \mathbf{x}$
$\mathbf{z}_0 \sim p_0(\mathbf{z}_0)$
$\mathbf{z}_i \sim p_i(\mathbf{z}_i)$
$\mathbf{z}_K \sim p_K(\mathbf{z}_K)$

https://lilianweng.github.io/posts/2018-10-13-flow-models/

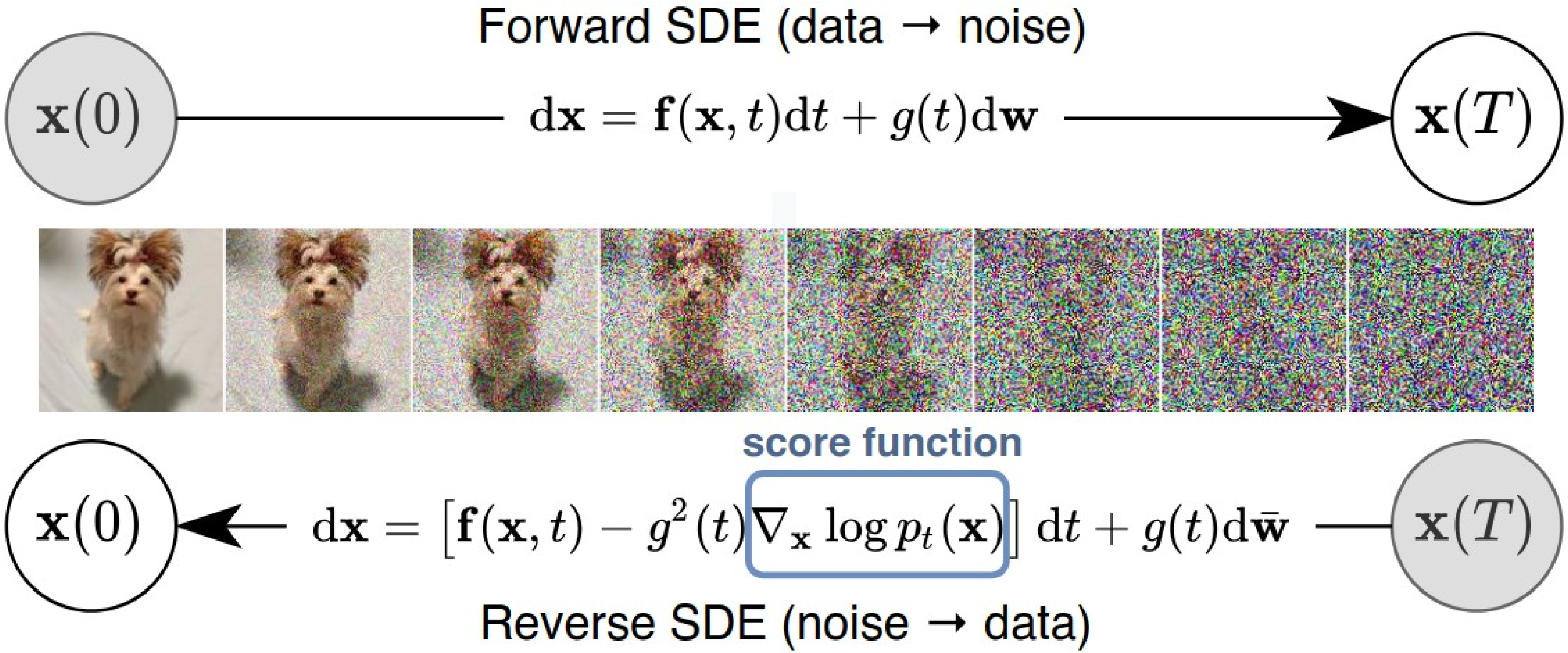
ML models used



ML models used

Score based model

It’s mathematically equivalent to a continuous time diffusion model.



Forward SDE (data → noise)

$$\mathbf{x}(0) \longrightarrow \mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w} \longrightarrow \mathbf{x}(T)$$

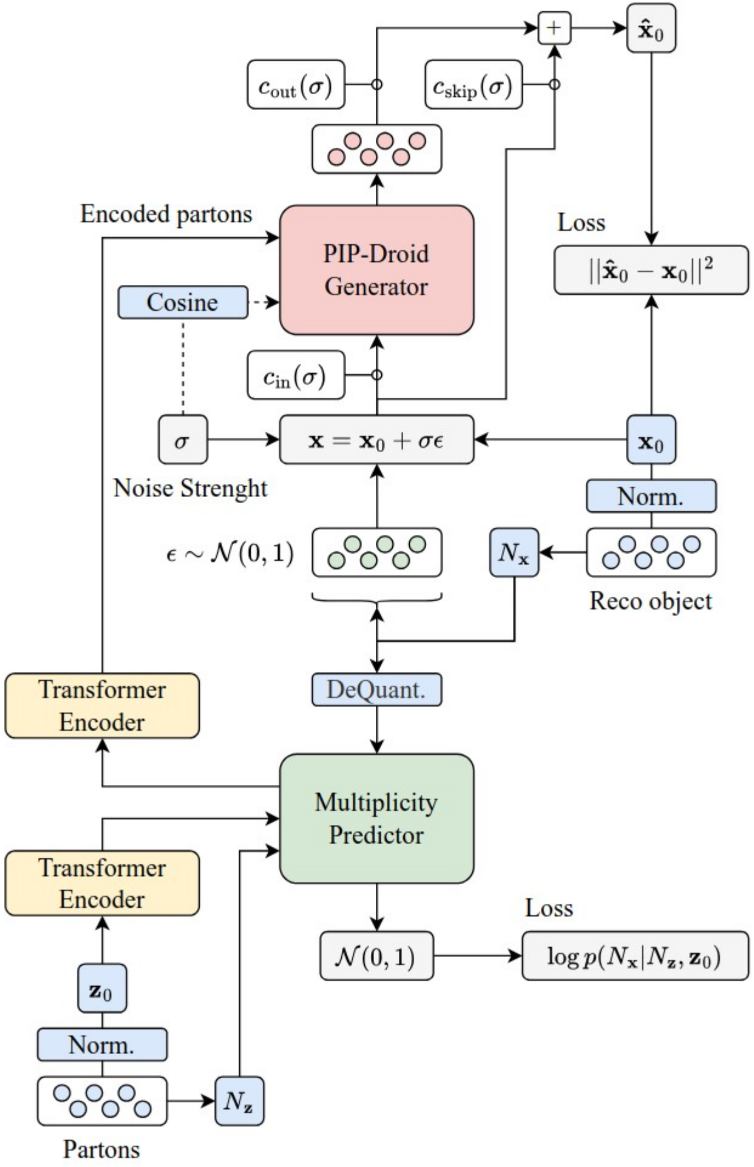
score function

$$\mathbf{x}(0) \longleftarrow \mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}} \longleftarrow \mathbf{x}(T)$$

Reverse SDE (noise → data)

https://arxiv.org/abs/2011.13456

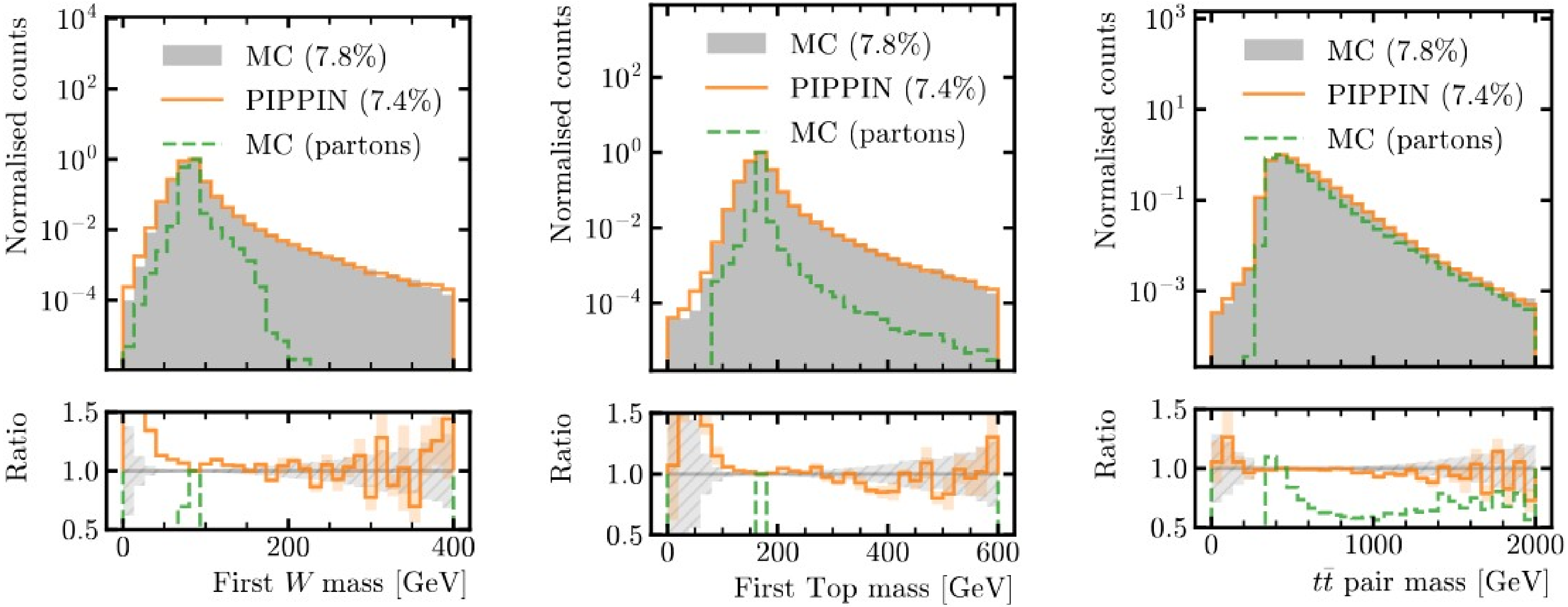
ML models used



Data and training

- Dataset is 50mil top quark pair production events.
- All decays (leptonic, semileptonic and hadronic) are included.
- Cuts are made on the dataset to exclude events that are hard to reconstruct. For example, jets are required to have $|\eta| < 2.5$.
- After cuts ~80% of the dataset remains.
- Model is trained end to end for 300 epoch
  - Except that in the first 2 epochs, the Multiplicty predictor is replaced with true multiplicites.

Results



Very good agreement overall.

Initial comments

- Skipping the detector stage entirely makes the model harder to validate.
- Without the detector stage, this model is specific to the reconstruction algorithms used (but then calibration generally is anyway).
- No pileup, no MPI, not sure about ISR and other noise?
- Could it have been simplified?

PIPPIN: Generating variable length full events from partons

Guillaume Quétant,[1, 2, *] John Andrew Raine,[1] Matthew Leigh,[1] Debajyoti Sengupta,[1] and Tobias Golling[1]

[1] *Département de Physique Nucléaire et Corpusculaire, University of Geneva, Switzerland*

[2] *Département d'Informatique, University of Geneva, Switzerland*

This paper presents a novel approach for directly generating full events at detector-level from parton-level information, leveraging cutting-edge machine learning techniques. To address the challenge of multiplicity variations between parton and reconstructed object spaces, we employ transformers, score-based models and normalizing flows. Our method tackles the inherent complexities of the stochastic transition between these two spaces and achieves remarkably accurate results. The combination of innovative techniques and the achieved accuracy demonstrates the potential of our approach in advancing the field and opens avenues for further exploration. This research contributes to the ongoing efforts in high-energy physics and generative modelling, providing a promising direction for enhanced precision in fast detector simulation.

I. INTRODUCTION

In the realm of high-energy physics, the simulation of particle collisions is a crucial tool for the downstream analysis of the huge amount of data produced by collider experiments. The classical generation of these simulated events is a complex task. By far the most computationally taxing subtask is the propagation of particles through the detectors, accounting for the interactions with the detector material and modelling the secondary radiation showers this produces. This process is usually performed using Monte Carlo (MC) full simulation tools such as Geant4 [1], or by faster, yet less accurate, parametrised simulation tools such as Delphes [2]. This is where machine learning comes into play, with the hope of bridging the gap between the speed of the fast simulators and the accuracy of the full ones.

The full simulation pipeline can be broken down into several subtasks. The first step is the event generation via matrix element calculation which simulates the hard process of the collision. This is typically done using MC event generators such as Pythia [3] or MadGraph5 [4], and the output of this step is a collection of partons. Next, the partons are used to simulate lower energy QCD processes such as radiation and hadronisation. This occurs before they hit the detector material and greatly increases the number of particles produced by the collision. Following this, all stable particles are propagated through the detector material, where they can interact with it and produce secondary particles. This showering process is typically the most computationally expensive part of the simulation. Finally, the deposited energy in the detector is digitised and reconstructed into objects such as jets, leptons and missing transverse energy (MET).

Several approaches have been proposed to replace parts of the pipeline with machine learning models, starting from the matrix element calculation and hard scattering event generation [5–11]. Other approaches focus on the time-consuming material interactions and detector response parts, both in terms of image [12–20] and point cloud [21–25] generation, with score-based generative models being applied with particular success to the latter [26–29]. While these are valid and well-motivated uses of machine learning, one can argue that a more ambitious goal would be to replace most of the simulation chain from hadronisation, showering and detector simulation to digitization and reconstruction [30–36]. The model would generate the reconstructed objects from random noise

[*] guillaume.quetant@unige.ch

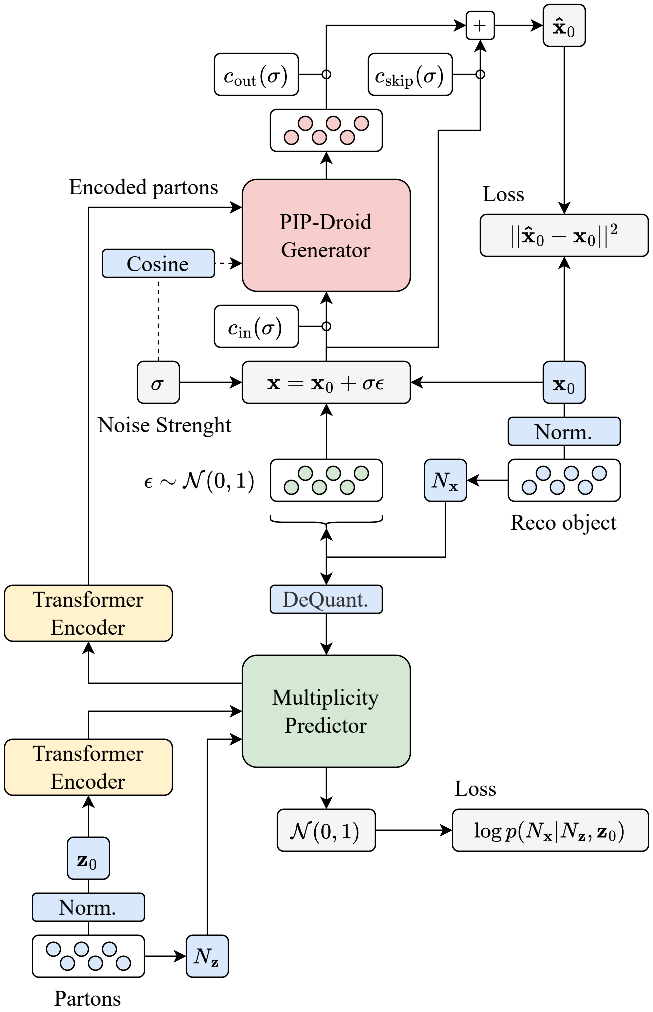


FIG. 1. Diagram of the global architecture and the training processes of the PIPPIN model. It is made of two Transformer Encoders, which encode the partons, a Multiplicity Predictor, which predicts the number of reconstructed objects, and a PIP-Droid Generator, which conditionally generate these reconstructed objects.

and may include conditioning by the partons from the event generator. We present a novel method called Particles Into Particles with Permutation Invariant Network (PIPPIN) to directly generate variable length full event point clouds at detector-level from parton-level information.

The PIPPIN model is based on the combination of transformers [37], score-based models [38], and normalizing flows [39]. A peculiarity of our model is its permutation invariance, an especially relevant feature to process unordered sets of particles. In addition, such sets of particles can have a variable number of elements, which is a challenge that PIPPIN takes up by predicting the number of outputs it has to generate. This means that we do not need to train a separate model for each multiplicity, truncate the output to a desired number of particles or capitalise on autoregressive generation. On the contrary, the model can directly generate any number of particles while preserving correlations based on the input. Furthermore, this property allows to generate events with a fixed number of particles of each type, which can be useful for filtering the simulation by restraining the phase space. Another important feature of the PIPPIN model is its conditional nature, which allows either to change the properties of the generated events depending on the chosen input particles or to fix them in order to stochastically generate multiple instances from the same input.

A related approach can be found in [34], with the main differences being that a) our inputs contain multiple partons of different natures instead of a single quark, b) our dataset is much larger but contains single pairs of input and output rather than several resimulated outputs per input, and c) our model uses score-based training and generation.

The case-study for our method is the simulation of top quark pair events in proton-proton collisions at the Large Hadron Collider (LHC), whose final state partons are used as input to the model to produce the full event at detector-level, also denoted as reconstructed objects. The main contributions of this paper are:[1]

- The introduction of a novel method to generate variable length full events at detector-level from parton-level information, leveraging cutting-edge transformers, score-based models and normalizing flows.

[1] Code is available at `https://github.com/rodem-hep/pippin`. Data is available at `https://zenodo.org/records/12117432`.

- A detailed study of the performance of the PIPPIN model in the context of top quark pair production at the LHC.
- The description and release of a new inclusive dataset of top quark pair events.

It should be noted that, despite the ambitions behind such an approach, a fundamental limitation arises whenever one wishes to generate full events rather than individual objects, namely the process-dependent nature of the simulation. We leave studies on the exploitation of the generalisation capabilities of the PIPPIN model, through its inclusive extension to a wider range of processes, to future work.

II. DATASET

In this work, we focus on the simulation of pairs of top quarks ($t\bar{t}$) produced in proton-proton collisions at a centre of mass energy $\sqrt{s} = 13$ TeV. The hard interactions as well as parton shower and hadronisation is performed using Pythia (v8.307) [3] with the Monash tuned set of parameters [40], using the NNPDF2.3LO PDF set [41] in the LHAPDF [42] framework, at leading order accuracy in both QCD and Electroweak interactions.

During event generation no constraints are placed on the decays of the top quarks or W bosons in the hard scatter, in order to capture the full range of final state particles. Top quarks decay to a W boson and a b-quark with a branching fraction over 99%. The W bosons then decay to a pair of quarks or to a charged lepton and a neutrino. Events can be categorised based on these decays into all-hadronic (0ℓ), semi-leptonic (1ℓ) and di-leptonic (2ℓ) events. In this categorisation we consider decays with tau leptons as semileptonic regardless of whether they decay hadronically or leptonically.

The detector response simulation is performed using Delphes [2] (v3.4.2) with a parametrisation consistent with the ATLAS detector [43].

Jets are clustered using the anti-k_t algorithm [44] with a radius parameter of $R = 0.4$ using the FastJet package [45]. They are required to fall within $|\eta| < 2.5$ and to have a minimum transverse momentum $p_{\mathrm{T}} > 25$ GeV. In addition to their four-momenta, jets are assigned a binary label corresponding to whether they pass a simulated b-jet identification algorithm, identifying jets which originate from b-hadrons (b-jets). A similar τ-jet identification algorithm is simulated, identifying jets which originate from hadronically decaying tau leptons.

Reconstructed electrons and muons are required to fall within $|\eta| < 2.5$ and to have a minimum transverse momentum $p_{\mathrm{T}} > 15$ GeV. They are represented by their associated charge in addition to their four-momenta. The reconstructed missing transverse momentum ($\vec{p}_{\mathrm{T}}^{\mathrm{miss}}$) of the event is calculated from the negative vector sum of all reconstructed visible particles.

All events with a minimum of 2 and a maximum of 16 jets are considered. No requirement is placed on the number of b-jets or leptons, though only up to the leading two leptons ordered in descending p_{T} are considered.[2]

In addition to the detector-level reconstructed objects, the four-momenta of the six parton-level final state particles (quarks, charged leptons and neutrinos) as well as for the two top quarks and W bosons are kept for each event. Truth association of reconstructed jets to the quarks from the $t\bar{t}$ decay (two quarks in the case of di-leptonic decays and up to six in the case of all-hadronic decays) is performed using ΔR matching.[3] Quarks are matched to jets within $\Delta R < 0.4$. In order to remove ambiguity in the reconstruction of W bosons and the top quarks from detector objects, an additional requirement is placed on events that no two jets are matched to the same parton, and no two partons are matched to the same jet. It

[2] This affects 0.002% of all events, corresponding to 0.1% of the events with at least two reconstructed leptons.

[3] From the pseudo-rapidity difference $\Delta\eta$ and the azimuthal angle difference $\Delta\phi$, one expresses the angular distance as $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$.

should be noted that this selection is not necessary for PIPPIN, and for a final model would be removed in order to preserve the event phase space from the original sample.[4]

In total there are 50 million events, of which 40.4 million pass the selection criteria after reconstruction. These correspond to approximately 18.6 million all-hadronic, 17.8 million semi-leptonic, and 4.1 million di-leptonic events based on the decays of the $t\bar{t}$ pair. We use 37 million events for training and 0.8 million for validation of the PIPPIN model; the remaining 2.4 million events are reserved for evaluation.

III. METHOD

PIPPIN is a conditional generative model that takes an unordered set of partons and outputs an unordered set of the reconstructed objects. Each element of both sets is described by their four-momenta. The model is made of four main components: two Transformer Encoders, the Multiplicity Predictor and the PIP-Droid Generator, as shown in Fig. 1.

The Transformer Encoders are two nearly identical[5] self-attention transformers [37], as shown in Fig. 2 (left), whose role is to encode the parton point cloud into latent representations further used by the other components of the model. The representation from the first encoder is used as input to the Multiplicity Predictor, while the final representation from the second encoder is used as conditional input to the PIP-Droid Generator.

The role of the Multiplicity Predictor is to estimate the number and type of reconstructed objects needed in the output point cloud. It is composed of three blocks, as shown in Fig. 3: a multilayer perceptron that predicts the probability that each of the input partons leads to a reconstructed object, a cross-attention transformer that extracts a global representation of the input point cloud, and a rational quadratic neural spline coupling normalizing flow [39] that predicts the multiplicity of each particle type in the output point cloud. Both the presence predictor and the global representation extractor are conditioned on the multiplicity of each particle type in the input point cloud, while the normalizing flow is conditioned on the extracted global representation. We use the CDF-Dequantization method introduced in Ref. [25] in order to preprocess the discrete multiplicity into continuous data to train the normalizing flow.

Finally, the PIP-Droid Generator is an adapted version of the PC-Droid score-based model [26, 27] that uses transformer decoder layers instead, as shown in Fig. 2 (right), in order to condition the denoising process using the encoded input point cloud. As in Ref. [27], we use the score-based framework of Ref. [38] including how we precondition our network using c_{in}, c_{out}, and c_{skip} functions in order to predict the original sample x_0 after perturbing it with scaled gaussian noise $x_t = x_0 + t \cdot \epsilon$, with $\epsilon \sim \mathcal{N}(0, I)$ and $t \in [0, 80]$.

As a conditional model, the PIP-Droid Generator takes as input the encoded parton point cloud, as well as the number of output tokens and their associated particle type, as given by the Multiplicity Predictor. During training, the true multiplicity is used for the first two epochs as the Multiplicity Predictor training is delayed for stability purpose. During generation, the predicted number of random tokens are sampled from a standard normal distribution and fed into the PIP-Droid Generator, which performs several denoising steps leading to the final output point cloud. In addition, a learnable token, specific to each particle type, is added to the input at each step as we observed that without this, the training was prone to collapse.

More details on the internal architecture and the training parameters can be found in Appendix A. Several im-

[4] This removes 16.5% of all events, with a higher fraction arising from the all-hadronic and semi-leptonic channels than dileptonic, due to the higher jet and quark multiplicities.

[5] Only the input dimension differs.

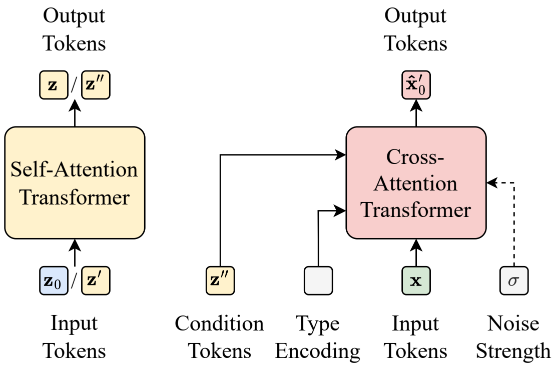


FIG. 2. The Transformer Encoder architecture (left), which encodes the parton point cloud and the PIP-Droid Generator architecture (right), which denoises random tokens with type encoding into reconstructed objects, conditioned on the parton point cloud.

portant features of the PIPPIN model are worth highlighting here.

Due to the permutation equivariant nature of transformers and the permutation invariant cross-attention operation used by the Multiplicity Predictor and the PIP-Droid Generator, the model as a whole is invariant to permutations of the input point cloud. This property is crucial when dealing with sets of particles, such as a couple of leptons and several jets, for which any imposed order is weakly motivated. In addition, since it is trained on multiple top quark decay channels at once, the model must handle this wide variety of inputs and outputs. The multiplicity distribution depends on the decay channel, which further increases the variability of the output point clouds. By learning to generate all these different types of point cloud, the PIPPIN model is expected to have a great capacity for generalisation.

One advantage of conditional generation is that the input point cloud can be modified before it is fed into the model, allowing the output point cloud to be generated accordingly. One can add partons to the input, or change their energy or direction, and generate new events based on these changes. Moreover, by removing the Multiplicity Predictor from the generation pipeline, one can use the model to generate point clouds with a fixed number of particles of each type, filtering the simulation and allowing for increased control over the generation process. The Multiplicity Predictor also predicts the probability for a parton to be present in the detector-level reconstructed point cloud. Partons can be lost by falling outside the acceptance region of the detector, or due to reconstruction inefficiency, because the parton is not energetic enough to be distinguished from noise, or is too close to another parton to be resolved. This additional task greatly assists the model in learning the correct number of particles to generate, as well as their kinematic properties, since partons provide a strong target for the reconstructed objects. The additional reconstructed objects are still conditionally generated, since they initially originate from parton radiations, but the model can learn that they are not the direct deposit of a parton.

The attention mechanism at the heart of transformers allows each token to attend to every other token, mean-

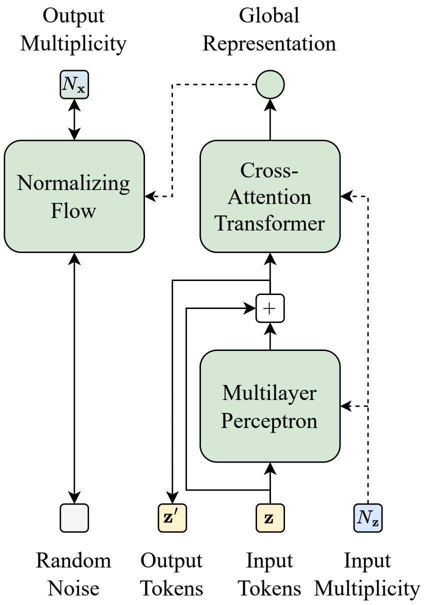


FIG. 3. The Multiplicity Predictor architecture. The partons presence is predicted and residually added to the encoded parton point cloud. Then, a global representation of the new encoded parton point cloud is learnt and conditionally used to sample the output multiplicity.

ing that the values of the output tokens are all correlated. This is relevant to generate unordered sets of particles, as it allows the model to learn correlations between them and thus to generate more realistic point clouds. In addition, compared to an autoregressive method, the PIPPIN model does not create a hierarchy between the particles, which is better suited to the unordered nature of the data. Moreover, since both the Multiplicity Predictor and the PIP-Droid Generator are conditional networks that take as input random noise, the model can be used to generate multiple outputs from the same input point cloud. Two main advantages come from this property. First, it may allow the model the properly learn the distinction between the content of the data, namely the physical nature of the point clouds such as the type of particles and features, and the style of the data, namely the statistical properties of the point clouds such as the distribution of the features. Second, it can be used to estimate part of the systematic uncertainty of the generation, by generating multiple instances of the same process and assessing its variability.

IV. RESULTS

The results are presented following the model pipeline. First, we present the estimations of the Multiplicity Predictor both in terms of parton presence in the reconstructed objects and the multiplicity of the outputs. Second, we show the outputs of the PIP-Droid Generator, i.e. the reconstructed object kinematics. Third, we perform an additional study of the kinematics of the underlying particles that initiated the partons. Note that for each plot in this section, the uncertainties for PIPPIN are estimated as the standard deviation over 5 generated sets initialised with different seeds, while for MC they are assumed to follow a Poisson distribution.

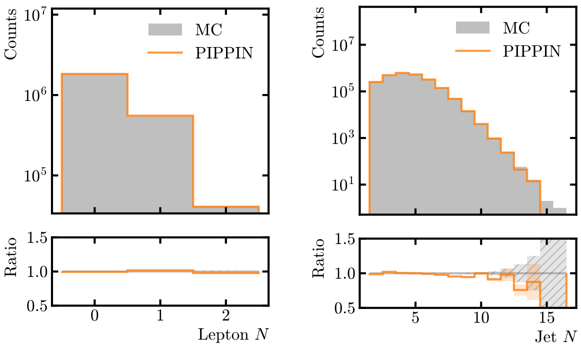


FIG. 4. Marginal distributions of the learnt reco-level multiplicities. **Left:** The multiplicity of the leptons in the reconstructed objects. **Right:** The multiplicity of the jets in the reconstructed objects. The grey area corresponds to the original MC simulation and the orange line to the output of the PIPPIN model. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas.

A. Multiplicity prediction

The first main step of the PIPPIN model is to predict the number of particles of each type that should be present in the reconstructed point cloud based on the input point cloud. Fig. 4 shows the marginal distributions of the predicted multiplicity N for the leptons and the jets. One can observe a perfect agreement between

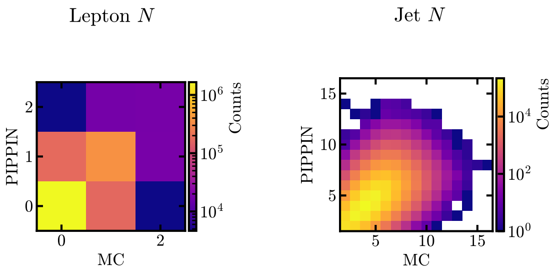


FIG. 5. 2D marginal distributions of the learnt reco-level multiplicities. **Left:** The multiplicity of the leptons in the reconstructed objects. **Right:** The multiplicity of the jets in the reconstructed objects. The x-axis corresponds to the original MC simulation and the y-axis to the associated output of the PIPPIN model.

the predicted and true lepton multiplicity. The jet multiplicity is also nearly perfectly predicted overall, but the model tends to slightly underestimate their number, having difficulties with events containing more than 14 jets. However, this tail of the distribution represents a tiny fraction of the events.

The corresponding 2D marginal distributions shown in Fig. 5 give more insight into how the input partons influence the predicted reconstructed object multiplicity. We recall here that the Multiplicity Predictor is conditioned on the input parton multiplicity and an encoded representation of the partons themselves, together with the estimated probability of presence of each of them in the output. A perfect agreement between the truth and the prediction would be a diagonal. We observe that indeed most events are close to it. However, due to the intrinsic stochasticity of the process, a spread around the diagonal is inevitable, as clearly seen for the jet multiplicity in Fig. 5 (right). The model is therefore not expected to predict the exact truth, but rather to learn the distribution of the reconstructed object multiplicity given the input partons. The 2D histograms confirm this expected behaviour.

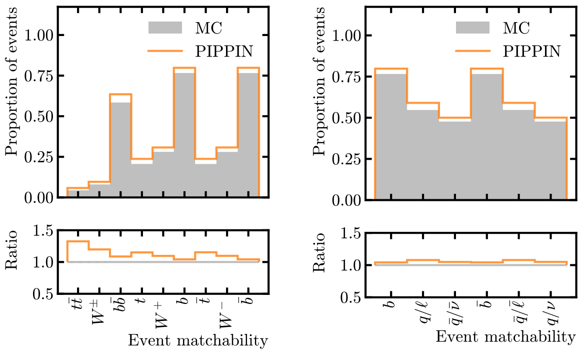


FIG. 6. Marginal distributions of the learnt event-level matchability. **Left:** The matchability for the intermediate underlying particles and the b-quarks. **Right:** The matchability for the final state particles, namely the partons. The grey area corresponds to the original MC simulation and the orange line to the output of the PIPPIN model. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded, but invisible, areas.

In addition, Fig. 6 shows a selected summary of the presence prediction, or matchability, gathered by underlying intermediate and final particles. A particle is considered present, or matched, if all its decay products, or partons, are present in the reconstructed objects of the event. As mentioned above, a parton may be missing due to acceptance and efficiency effects, namely detector blind regions, noise thresholds or resolution limitations. On the left plot, from left to right, the first bin shows the proportion of events for which all 6 partons are matched to form the $t\bar{t}$ pair, the second bin for which the 4 partons originating from the W bosons are matched, the third bin for which the 2 b-quarks are matched, etc. Intuitively, the b and $\bar{b}$-quark bins are expected to be the most populated, as they require a single parton to be matched to a reconstructed object per event. The $b\bar{b}$ bin contains the intersection of the two and is therefore expected to be slightly less populated. The W boson bins also require two partons to be matched, one of which being sometimes a neutrino. This adds a layer of complexity to the matching, which is reflected in the lower population of these bins. The t and $\bar{t}$ bins being the intersection of the W boson and b-quark bins, they are also slightly less populated. Finally, the $t\bar{t}$ bin requires all 6 partons to be matched and is therefore the least populated. On the right plot, the bins simply show the proportion of events for which each of the 6 partons are individually matched.

We observe a reasonably good agreement between the predicted and true presence, with a slight overestimation in all cases. This means that the model has a conservative tendency, trying to keep the partons in the reconstructed objects more often than the imperfect detector response would allow. The exact reasons behind this behaviour are not clear, but one has to keep in mind that the histograms show the matchability as a binary variable, while the model predicts a probability of presence for each parton. Furthermore, the presence prediction is not a critical quantity for the model to produce accurate output point clouds, but rather a helpful auxiliary task. Therefore, the observed discrepancy is not a major concern.

B. Kinematic properties

Each particle in the point clouds is represented by a set of four features: its transverse momentum, pseudorapidity, azimuthal angle, and mass (for the partons) or energy (for the reconstructed objects). The marginal distributions of the energy and transverse momentum are shown in Fig. 7 for the jets and in Appendix B for the other reconstructed objects. Apart from an underestimated number of high-energy and transverse momentum jets, and an overestimate of very high-energy jets, likely due to a low statistics in the tails of the distributions, and an overestimate of very low-energy and transverse momentum jets, likely due to the sharp cut at zero, one can observe a very good agreement between the predicted and true histograms. This good agreement is expected as the PIP-Droid Generator score-based model is trained on these features.

The 2D marginal distributions in Fig. 8 lead to the same conclusion. The diagonals are well populated, with an expected spread due to the stochastic nature of the process.

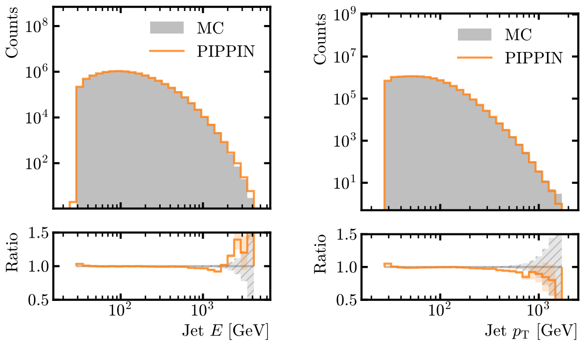


FIG. 7. Marginal distributions of the learnt features of the reco-level jets. **Left:** The energy of the jets in the reconstructed objects. **Right:** The p_T of the jets in the reconstructed objects. The grey area corresponds to the original MC simulation and the orange line to the output of the PIPPIN model. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas.

C. Underlying particles

In addition to the features of the reconstructed objects, we can compute several properties of the underlying decayed particles, namely the invariant masses and transverse momenta of the W bosons, the top and anti-top quarks, and the whole $t\bar{t}$ system. In order to identify the objects originating from each underlying particle, we perform a matching between the partons and the reconstructed objects based on their proximity in the (η, ϕ) plane. Similarly to how the dataset is produced, objects with $\Delta R < 0.4$ are matched to partons, keeping only fully matched events and discarding those with multiple objects matched to a parton, and vice versa. Furthermore, di-leptonic events originally contain two neutrinos, which have to be disentangled from the single MET object present at the detector level. To avoid the need of any complex algorithm, we create two fake detector-level neutrinos by taking the highest p_T parton-level neutrino and the remaining MET after subtraction. For both the semi-

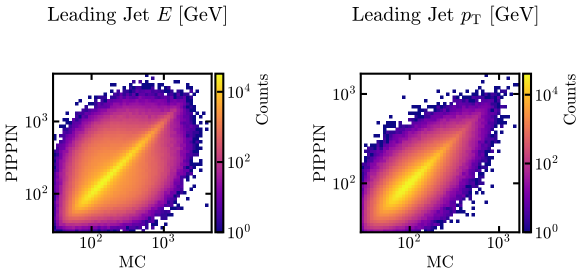


FIG. 8. 2D marginal distributions of the learnt features of the reco-level jets. **Left:** The energy of the jets in the reconstructed objects. **Right:** The p_T of the jets in the reconstructed objects. The x-axis corresponds to the original MC simulation and the y-axis to the associated output of the PIPPIN model.

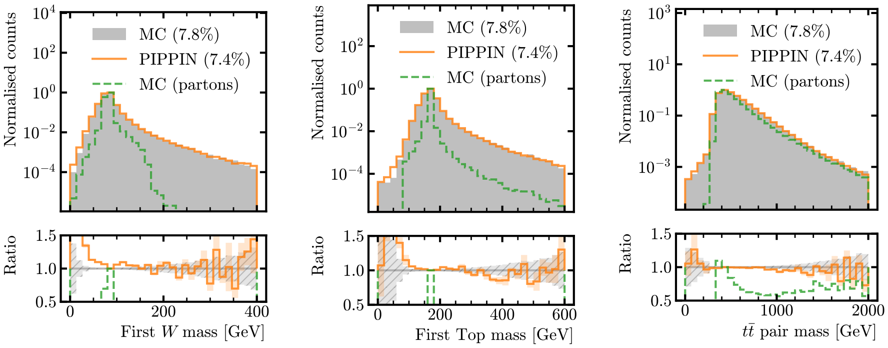


FIG. 9. Marginal distributions of the invariant masses of the underlying particles at reco-level. **Left:** The mass of the first reconstructed W boson. **Middle:** The mass of the first reconstructed top quark. **Right:** The mass of the whole reconstructed $t\bar{t}$ system. By first we mean the particle originating from the top quark, as opposed to the anti-top quark. The grey area corresponds to the original MC reco-level simulation, the orange line to the output of the PIPPIN model and the dashed green line to the MC parton-level simulation on which the model is conditioned. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas. The percentages indicate the proportion of events for which all partons are unambiguously matched and therefore present on the plots.

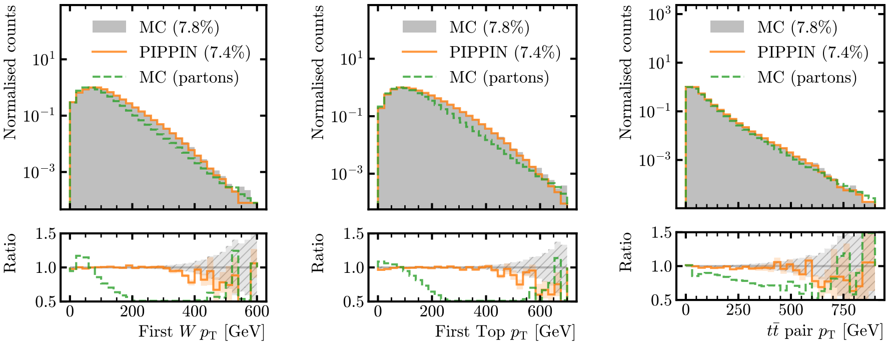


FIG. 10. Marginal distributions of the transverse momenta of the underlying particles at reco-level. **Left:** The transverse momentum of the first reconstructed W boson. **Middle:** The transverse momentum of the first reconstructed top quark. **Right:** The transverse momentum of the whole reconstructed $t\bar{t}$ system. By first we mean the particle originating from the top quark, as opposed to the anti-top quark. The grey area corresponds to the original MC reco-level simulation, the orange line to the output of the PIPPIN model and the dashed green line to the MC parton-level simulation on which the model is conditioned. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas. The percentages indicate the proportion of events for which all partons are unambiguously matched and therefore present on the plots.

leptonic and di-leptonic cases, we also use the parton-level neutrino system to extract the pseudo-rapidity associated to the MET, and then compute the corresponding energy as $E = p_T \cdot \cosh \eta$. The use of these approximate proxy variables is not a problem as we are only interested in the PIPPIN capabilities to reproduce them and not in their precise estimation.

The marginal distributions of the invariant masses and transverse momenta of the underlying particles are shown in Figs. 9 and 10, where we also overlay the parton-level distributions for comparison purpose. In essence, the PIPPIN model aims to transform the parton-level distributions into the reco-level ones, conditioned on these partons. The percentage of fully matched events, i.e., events for which we can recover all the underlying particles, is shown in the legend of each plot. It is expected to correspond to the height of the $t\bar{t}$ bin in Fig. 6 (left), as this represents the proportion of events for which the 6 partons are predicted to be present in the reconstructed objects. Note that the percentages are small, indicating that more than 92% of the time at least one parton is missing due to the aforementioned detector acceptance and efficiency limitations. It is interesting to notice that the truth events percentage is almost reached by the PIPPIN generated events. This is a good indicator that the model is able to generate different event topologies in the same proportion as the truth. The apparent contradiction between the lower percentage of fully matched events generated by PIPPIN shown in Figs. 9 and 10 and the higher proportion of events for which all partons are predicted to be present in Fig. 6 (left) with respect to MC should not be a concern. On the contrary, this is a good sign that the PIP-Droid Generator is able to compensate for a possible mismodelling of the presence predicted by the Multiplicity Predictor, justifying our claim that this is not a critical quantity to predict.

Although there is an tendency to overestimate the number of low mass particles at the expense of high mass ones, the distributions of the invariant masses in Fig. 9 and the transverse momenta in Fig. 10 show a good agreement between the truth and the prediction. It is worth highlighting this agreement with the detector-level distributions, bearing in mind that the model is conditioned on the partons whose corresponding distributions are significantly different. This is very encouraging, as these quantities are not used as training targets, while their modelling is crucial for the detector-level point cloud to be physically meaningful. Additional plots of invariant masses and transverse momenta can be found in Appendix B.

D. Comparison to other models

PIPPIN is the first model applied to the problem of simulating an inclusive set of top quark pair production events of variable length at the LHC. This means that for now the only way to compare our method to other models is to restrict the set of events to the specific topology considered by them. We can compare to the OTUS [31] and Turbo-Sim [32] related models, which are both designed to simulate the semi-leptonic decay of a top quark pair with exactly 1 lepton, MET and 4 jets present in the reconstructed objects. Plots comparing the features of the dataset originally used by these two models and the restricted version of the dataset used in this paper are provided in Appendix B.

In Table I, we show the Kolmogorov-Smirnov (KS) distances, which reflect the agreement between two cumulative distribution functions, between the original MC reco-level simulation and the output of the different models. The observables considered are divided in two categories, namely the kinematic features of the reconstructed point cloud and the features of the underlying intermediate particles. For the point cloud, we show the KS distances for the p_{T} leading jet momentum in the y and z directions as well as for its energy. For the underlying particles, we show the KS distances for the mass of the $t\bar{t}$ system, one of the W bosons, and the two top quarks. Fig. 11 shows a comparison of the distributions of the leading jet energy and the $t\bar{t}$ mass for the MC simulation as well as the output of PIPPIN and Turbo-Sim. We show both the results when the output events produced by PIPPIN are selected to match the topology of the other models, and when all events are considered regardless of topology. We also show the original results of the OTUS and Turbo-Sim models based on the dataset provided by the authors, as well as the results of the Turbo-Sim model when retrained on the restricted version of the dataset presented in this study. Note that the Turbo-Sim model was retrained using the same architecture and hyperpa-

	Reco. objects			Underlying particles			
Model	$p_y^{jet_1}$	$p_z^{jet_1}$	E^{jet_1}	$m^{t\bar{t}}$	m^{W_1}	m^{t_1}	m^{t_2}
OTUS	*3.78*	*2.39*	*5.75*	*15.8*	*11.7*	*14.1*	*24.9*
Turbo-Sim	*2.89*	*10.3*	*4.43*	*2.97*	*7.72*	*5.20*	*8.52*
Turbo-Sim (new)	8.63	12.6	6.32	7.90*	38.8*	38.5*	43.6*
PIPPIN	**0.32**	**0.33**	**0.34**	**4.00**	**3.66**	**3.27**	**2.44**
PIPPIN (inc)	*0.08*	*0.14*	*0.12*	*0.33*	*1.69*	*0.54*	*0.60*

TABLE I. The Kolmogorov-Smirnov distances [$\times 10^{-2}$] computed between the original MC reco-level simulation and the output of different models. Lower values mean better agreement. The observables considered are the momentum in the y and z directions and the energy of the leading reconstructed jet in p_T, as well as the mass of several underlying particles. For OTUS and Turbo-Sim, m^{W_1}, m^{t_1} and m^{t_2} are the masses of the hadronic W boson, the leptonic top quark and the hadronic top quark, respectively. For PIPPIN, they are the masses of the first W boson, the first top quark, and the second top quark, respectively, as defined in Figs. 9 and 10. The mass of the whole $t\bar{t}$ system is denoted $m^{t\bar{t}}$ for all models. The OTUS and Turbo-Sim lines show the original results of these models based on the dataset provided by the authors. The Turbo-Sim (new) and PIPPIN lines show the results of these models on the restricted version of the dataset presented in this study. The PIPPIN (inc) line shows the results for PIPPIN when not restricted to semi-leptonic events containing exactly 1 lepton, MET and 4 jets. We have marked with an asterisk results where the matching procedure was slightly relaxed in order to capture enough events for comparison. The matching radius was increased to $\Delta R < 0.8$ and events with multiple matching reconstructed objects per parton were allowed, with the closest being selected.

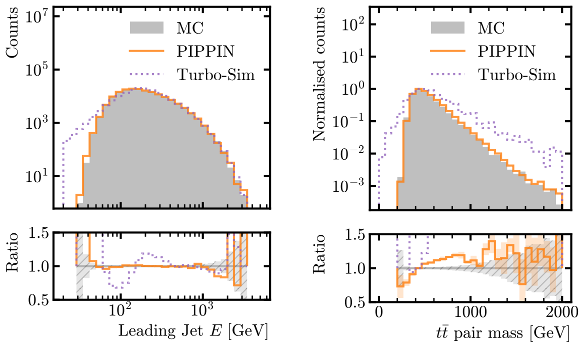


FIG. 11. Comparison of the marginal distributions for different models. **Left:** The energy of the leading jet in the reconstructed objects. **Right:** The mass of the whole reconstructed $t\bar{t}$ system. The grey area corresponds to the original MC simulation, the orange line to the output of the PIPPIN model and the purple dotted line to the output of the Turbo-Sim model. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas.

rameters as the original model, without further optimisation for the new dataset. The worse results compared to the original Turbo-Sim are therefore to be expected. We observe that PIPPIN significantly outperforms the two other models. It is also worth noting that, without the topological restriction, the results are even better, which has two meanings. On the one hand, it may indicate that the subset of events considered by the other models is more difficult to simulate. This would not be surprising as the majority of events in the dataset are all-hadronic and do not contain charged leptons and neutrinos, removing an important source of complexity. On the other hand, it means that PIPPIN is able to accurately simulate a much wider range of events than the other models, while still outperforming them on the specific subset. It would be interesting to see how PIPPIN compares to itself when trained directly on the restricted dataset, rather than trained inclusively and then restricted on the outputs, but this goes beyond the scope of this work and is left for future studies.

V. CONCLUSION

In this work, we have presented a new approach to the simulation of high-energy physics events, leveraging cutting-edge deep learning architectures and a shift of paradigm, namely the direct generation of reconstructed objects from final state partons instead of the traditional step-by-step decomposition of the pipeline. We have shown that the proposed PIPPIN model is able to handle the complexity of the top quark pair production problem and to correctly predict the multiplicity and kinematics of the outputs. It is worth emphasising that the model is able to learn the topologies of several decay channels simultaneously, without the need for channel-specific training. Moreover, the correlations between the different simulated reconstructed objects, tested by exploring the kinematics of the underlying decayed particles, are well captured by the model. This is a crucial feature for the accurate simulation of high-energy physics events, as the kinematics of the final state particles are not independent of each other. The control over the generated events is also a key feature of the model. It could allow to fix some properties of the outputs, as the number of jets and leptons or the presence of specific partons in the reconstructed objects.

It would be interesting to make use of this controllability to further investigate the generalisation capabilities of the model, by testing it on modified or unseen processes. For example, one could train the model on a given Beyond Standard Model process and test it on the same process parametrised with different masses or couplings. One could even test the model on a completely different process, such as the production of a new particle alongside the top quark pairs, and see how well it generalises.

Lastly, the method presented in this work is nicely suited for the reverse problem, namely the unfolding of the reconstructed objects to the initial partons. If several final states are allowed, it would be a natural extension of the PIPPIN model, as the tasks involved would be analogous. The model would have to predict the multiplicity of each parton and the correspondence between the reconstructed objects and the partons. Then, the conditional generation would follow the same procedure, with the partons as outputs and the reconstructed objects as conditional inputs. An analogous task, which however stops the unfolding before reversing the hadronisation and radiation processes, has been studied in [46] during the development of this work with a similar pipeline that uses a variational latent diffusion generator instead. We leave the complete study of applying our method to the unfolding problem for future work.

ACKNOWLEDGEMENTS

The authors would like to acknowledge funding through the SNSF Sinergia grant CRSII5_193716 “Robust Deep Density Models for High-Energy Particle Physics and Solar Flare Analysis (RODEM)”, the SNSF project grant 200020_212127 “At the two upgrade frontiers: machine learning and the ITk Pixel detector” and the ESKAS Scholarship for international students. The computations were performed at the University of Geneva on the “Baobab” and “Yggdrasil” HPC clusters.

[1] S. Agostinelli *et al.*, GEANT4—a simulation toolkit, Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **506**, 250 (2003).

[2] The DELPHES 3 collaboration, J. de Favereau, C. Delaere, P. Demin, A. Giammanco, V. Lemaître, A. Mertens, and M. Selvaggi, DELPHES 3: a modular framework for fast simulation of a generic collider experiment, Journal of High Energy Physics **2014**, 57 (2014).

[3] T. Sjöstrand, S. Mrenna, and P. Skands, A brief introduction to PYTHIA 8.1, Computer Physics Communications **178**, 852 (2008).

[4] J. Alwall, R. Frederix, S. Frixione, V. Hirschi, F. Maltoni, O. Mattelaer, H.-S. Shao, T. Stelzer, P. Torrielli, and M. Zaro, The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations, Journal of High Energy Physics **2014**, 79 (2014).

[5] A. Butter, T. Plehn, and R. Winterhalder, How to GAN LHC events, SciPost Physics **7**, 075 (2019).

[6] D. Maître and H. Truong, A factorisation-aware Matrix element emulator, Journal of High Energy Physics **2021**, 66 (2021).

[7] R. Winterhalder, V. Magerya, E. Villa, S. Jones, M. Kerner, A. Butter, G. Heinrich, and T. Plehn, Targeting multi-loop integrals with neural networks, SciPost Physics **12**, 129 (2022).

[8] F. Bishara and M. Montull, Machine learning amplitudes for faster event generation, Physical Review D **107**, L071901 (2023).

[9] T. Heimel, R. Winterhalder, A. Butter, J. Isaacson, C. Krause, F. Maltoni, O. Mattelaer, and T. Plehn, MadNIS - Neural multi-channel importance sampling, SciPost Physics **15**, 141 (2023).

[10] T. Heimel, N. Huetsch, F. Maltoni, O. Mattelaer, T. Plehn, and R. Winterhalder, The MadNIS Reloaded (2023), arXiv:2311.01548 [hep-ph].

[11] S. Badger, A. Butter, M. Luchmann, S. Pitz, and T. Plehn, Loop amplitudes from precision networks, SciPost Physics Core **6**, 034 (2023).

[12] L. de Oliveira, M. Paganini, and B. Nachman, Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis, Computing and Software for Big Science **1**, 4 (2017).

[13] The ATLAS Collaboration, Deep Generative Models for Fast Photon Shower Simulation in ATLAS, Computing and Software for Big Science **8**, 7 (2024).

[14] M. Paganini, L. de Oliveira, and B. Nachman, CaloGAN: Simulating 3D high energy particle showers in multilayer electromagnetic calorimeters with generative adversarial networks, Physical Review D **97**, 014021 (2018).

[15] C. Krause and D. Shih, CaloFlow II: Even Faster and Still Accurate Generation of Calorimeter Showers with Normalizing Flows (2021), arXiv:2110.11377 [physics.ins-det].

[16] J. C. Cresswell, B. L. Ross, G. Loaiza-Ganem, H. Reyes-Gonzalez, M. Letizia, and A. L. Caterini, CaloMan: Fast generation of calorimeter showers with density estimation on learned manifolds (2022), arXiv:2211.15380 [hep-ph].

[17] V. Mikuni and B. Nachman, Score-based generative models for calorimeter shower simulation, Physical Review D **106**, 092009 (2022).

[18] D. Kobylianskii, N. Soybelman, E. Dreyer, and E. Gross, CaloGraph: Graph-based diffusion model for fast shower generation in calorimeters with irregular geometry (2024), arXiv:2402.11575 [hep-ex].

[19] Q. Liu, C. Shimmin, X. Liu, E. Shlizerman, S. Li, and S.-C. Hsu, Calo-VQ: Vector-Quantized Two-Stage Generative Model in Calorimeter Simulation (2024), arXiv:2405.06605 [physics.ins-det].

[20] L. Favaro, A. Ore, S. P. Schweitzer, and T. Plehn, CaloDREAM – Detector Response Emulation via Attentive flow Matching (2024), arXiv:2405.09629 [hep-ph].

[21] R. Kansal, J. Duarte, H. Su, B. Orzari, T. Tomei, M. Pierini, M. Touranakou, J.-R. Vlimant, and D. Gunopulos, Particle Cloud Generation with Message Passing Generative Adversarial Networks (2021), arXiv:2106.11535 [cs.LG].

[22] B. Käch, D. Krücker, I. Melzer-Pellmann, M. Scham, S. Schnake, and A. Verney-Provatas, JetFlow: Generating Jets with Conditioned and Mass Constrained Nor-

malising Flows (2022), arXiv:2211.13630 [hep-ex].

[23] E. Buhmann, G. Kasieczka, and J. Thaler, EPiC-GAN: Equivariant point cloud generation for particle jets, SciPost Physics **15**, 130 (2023).

[24] B. Käch and I. Melzer-Pellmann, Attention to Mean-Fields for Particle Cloud Generation (2023), arXiv:2305.15254 [hep-ex].

[25] S. Schnake, D. Krücker, and K. Borras, CaloPointFlow II Generating Calorimeter Showers as Point Clouds (2024), arXiv:2403.15782 [physics.ins-det].

[26] M. Leigh, D. Sengupta, G. Quétant, J. A. Raine, K. Zoch, and T. Golling, PC-JeDi: Diffusion for particle cloud generation in high energy physics, SciPost Physics **16**, 018 (2024).

[27] M. Leigh, D. Sengupta, J. A. Raine, G. Quétant, and T. Golling, Faster diffusion model with improved quality for particle cloud generation, Physical Review D **109**, 012010 (2024).

[28] E. Buhmann, C. Ewen, D. A. Faroughy, T. Golling, G. Kasieczka, M. Leigh, G. Quétant, J. A. Raine, D. Sengupta, and D. Shih, EPiC-ly Fast Particle Cloud Generation with Flow-Matching and Diffusion (2023), arXiv:2310.00049 [hep-ph].

[29] V. Mikuni, B. Nachman, and M. Pettee, Fast point cloud generation with diffusion models in high energy physics, Physical Review D **108**, 036025 (2023).

[30] M. Bellagente, A. Butter, G. Kasieczka, T. Plehn, A. Rousselot, R. Winterhalder, L. Ardizzone, and U. Köthe, Invertible networks or partons to detector and back again, SciPost Physics **9**, 074 (2020).

[31] J. N. Howard, S. Mandt, D. Whiteson, and Y. Yang, Learning to simulate high energy particle collisions from unlabeled data, Scientific Reports **12**, 7567 (2022).

[32] G. Quétant, M. Drozdova, V. Kinakh, T. Golling, and S. Voloshynovskiy, Turbo-Sim: a generalised generative model with a physical latent space (2021), arXiv:2112.10629 [cs.LG].

[33] G. Quétant, Y. Belousov, V. Kinakh, and S. Voloshynovskiy, TURBO: The Swiss Knife of Auto-Encoders, Entropy **25**, 1471 (2023).

[34] N. Soybelman, N. Kakati, L. Heinrich, F. A. Di Bello, E. Dreyer, S. Ganguly, E. Gross, M. Kado, and J. Shlomi, Set-conditional set generation for particle physics, Machine Learning: Science and Technology **4**, 045036 (2023).

[35] A. Butter, T. Heimel, S. Hummerich, T. Krebs, T. Plehn, A. Rousselot, and S. Vent, Generative networks for precision enthusiasts, SciPost Physics **14**, 078 (2023).

[36] A. Butter, N. Huetsch, S. P. Schweitzer, T. Plehn, P. Sorrenson, and J. Spinner, Jet Diffusion versus JetGPT – Modern Networks for the LHC (2023), arXiv:2305.10475 [hep-ph].

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, Attention Is All You Need (2017), arXiv:1706.03762 [cs.CL].

[38] T. Karras, M. Aittala, T. Aila, and S. Laine, Elucidating the Design Space of Diffusion-Based Generative Models (2022), arXiv:2206.00364 [cs.CV].

[39] C. Durkan, A. Bekasov, I. Murray, and G. Papamakarios, Neural Spline Flows (2019), arXiv:1906.04032 [stat.ML].

[40] P. Skands, S. Carrazza, and J. Rojo, Tuning PYTHIA 8.1: the Monash 2013 Tune, The European Physical Journal C **74**, 3024 (2014).

[41] NNPDF Collaboration, R. D. Ball, *et al.*, Parton distributions with LHC data, Nuclear Physics B **867**, 244 (2013).

[42] A. Buckley, J. Ferrando, S. Lloyd, K. Nordström, B. Page, M. Rüfenacht, M. Schönherr, and G. Watt, LHAPDF6: parton density access in the LHC precision era, The European Physical Journal C **75**, 132 (2015).

[43] The ATLAS Collaboration, The ATLAS Experiment at the CERN Large Hadron Collider, Journal of Instrumentation **3**, S08003.

[44] M. Cacciari, G. P. Salam, and G. Soyez, The anti-kt jet clustering algorithm, Journal of High Energy Physics **2008**, 063 (2008).

[45] M. Cacciari, G. P. Salam, and G. Soyez, FastJet user manual, The European Physical Journal C **72**, 1896 (2012).

[46] A. Shmakov, K. Greif, M. J. Fenton, A. Ghosh, P. Baldi, and D. Whiteson, Full Event Particle-Level Unfolding with Variable-Length Latent Variational Diffusion (2024), arXiv:2404.14332 [hep-ex].

Appendix A: Detailed architecture

In this appendix we describe the architecture details of the three components of the PIPPIN model, as well as the main training parameters.[6]

1. Transformer Encoder

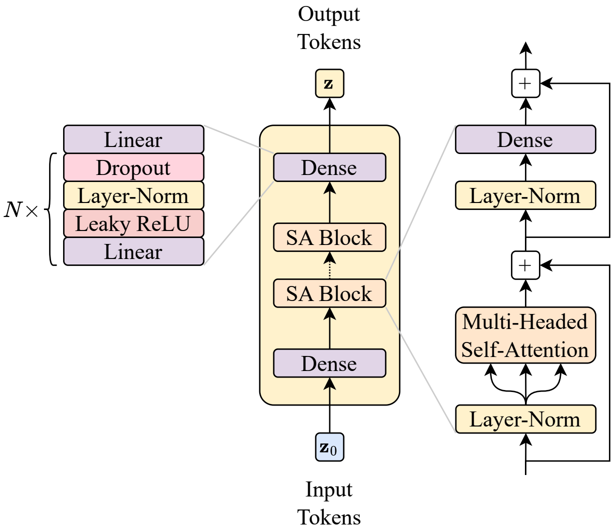


FIG. 12. The Transformer Encoder architecture details.

The Transformer Encoder is composed of a node embedding Dense network followed by two Self-Attention (SA) blocks and an output embedding Dense network as shown in Fig. 12. Each Dense network has a similar architecture, namely stacks of a linear layer, a leaky ReLU activation, a layer-normalisation and a dropout layer. The SA blocks are the core components of the Transformer Encoder, and are made of a residual Multi-Headed Self-Attention layer and a residual Dense network, both preceded by layer-normalisation.

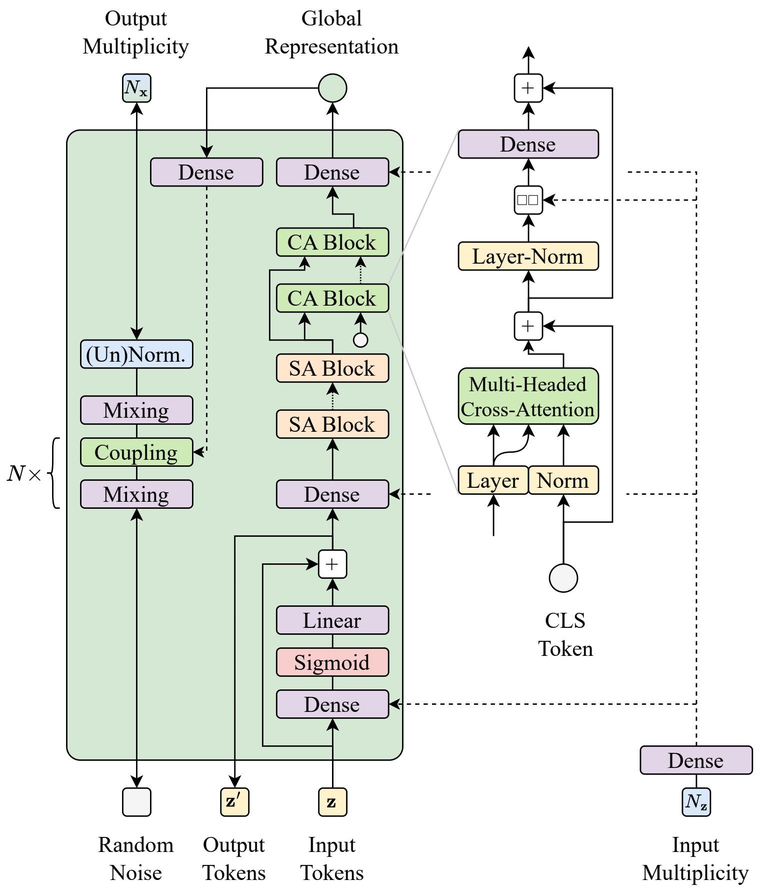


FIG. 13. The Multiplicity Predictor architecture details.

2. Multiplicity Predictor

As shown in Fig. 13, the first stage of the Multiplicity Predictor pipeline is a residual presence predictor, conditioned on the input multiplicity, made of a single Dense network whose output is scaled between 0 and 1 by a sigmoid activation and embedded to the input dimension by a linear layer. The role of this first component is to predict which input partons should be present in the output reconstructed object. The embedded prediction being residually added to the input, all following layer, including the PIP-Droid Generator, are aware of this presence.

The second stage of the Multiplicity Predictor pipeline is a global representation extractor, also conditioned on the input multiplicity, with an analogue architecture to the Transformer Encoders, but with two additional Cross-Attention (CA) blocks. These blocks swap the

[6] All default hyperparameters are available in the code.

residual Multi-Headed Self-Attention layer for a Cross-Attention version which distributes the input information on a learnable token (CLS) finally representing the extracted global information.

The third stage of the Multiplicity Predictor pipeline is a rational quadratic neural spline coupling normalizing flow. It is made of stacks of mixing and coupling layers, conditioned on the embedded global representation previously extracted. The role of this component is the actual prediction of the output reconstructed objects multiplicity. On top of the leptons and the jets multiplicity, and even though all events contain a single object representing the MET, we decided to let the model predict the MET multiplicity as well in order to better learn how to handle the cases of zero, one and two neutrinos.

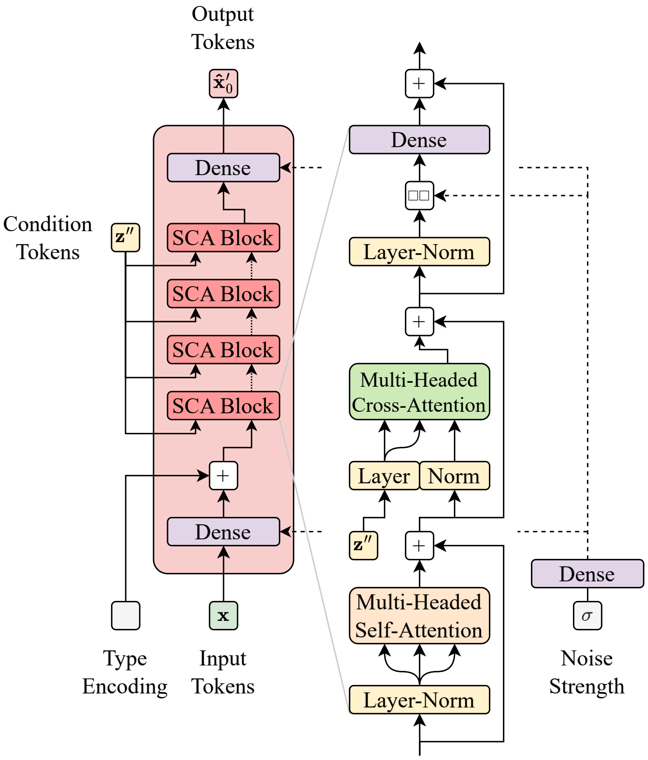


FIG. 14. The PIP-Droid Generator architecture details.

3. PIP-Droid Generator

The PIP-Droid Generator has a transformer decoder architecture, shown in Fig. 14, made of input and output embedding Dense networks, as well as four core Self and Cross-Attention (SCA) blocks, all of them conditioned on the noise strength corresponding to the diffusion step. Each SCA block contains a first residual Multi-Headed Self-Attention layer to encode the noisy tokens followed by a Multi-Headed Cross-Attention layer to distribute on them the conditional information coming from the parton tokens. A type encoding learnable token is also added before feeding the SCA blocks. These tokens are specific to each particle type, namely lepton, MET and jet, in order to avoid the collapse of the output. Indeed, we observed that the reconstructed objects have a tendency to converge to an average non-physical particle without encoding the desired output type.

4. Training Details

The whole model is trained end-to-end for 300 epochs with a single combined objective function, except for the first two epochs, for which the Multiplicity Predictor is not used and therefore not trained. Indeed, giving the true multiplicity at the beginning of the training helps the score-based model to stabilise. The objective function is composed of a binary cross-entropy loss on the presence prediction, a negative log-likelihood loss on the multiplicity prediction and a mean squared error loss on the output point cloud. The model is trained with the AdamW optimiser, a batch size of 8192 and a learning rate of 1×10^{-4} with a warming up schedule of 50'000 optimisation steps. We recall that the training dataset contains 37 million events.

Although the global representation extractor and the normalizing flow found in the Multiplicity Predictor form a separate branch of the pipeline, the other branch being the second Transformer Encoder and the PIP-Droid Generator, they are trained together with the rest of the model. This has two main advantages. On the one hand, the first few blocks of the PIPPIN model, namely the first Transformer Encoder and the presence prediction, receive gradient from both branches, since they participate in both. This helps the convergence of the model by maintaining the consistency of the encoded partons over the Multiplicity Predictor and the PIP-Droid Generator. On the other hand, it simplifies the training process by avoiding the need to train the two branches separately, which would require a more complex training loop.

Appendix B: Additional plots

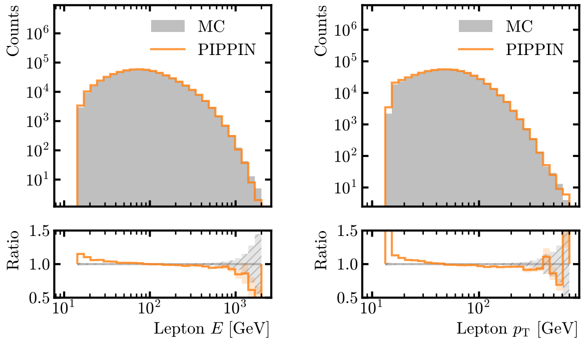


FIG. 15. Marginal distributions of the learnt features of the reco-level leptons. **Left:** The energy of the leptons in the reconstructed objects. **Right:** The p_T of the leptons in the reconstructed objects. The grey area corresponds to the original MC simulation and the orange line to the output of the PIPPIN model. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas.

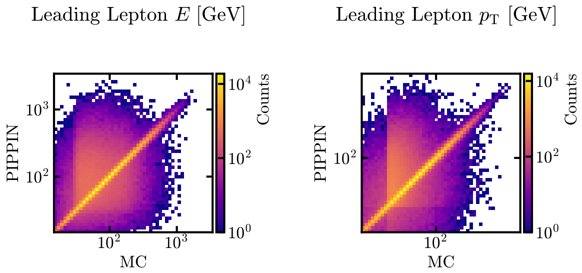


FIG. 16. 2D marginal distributions of the learnt features of the reco-level leptons. **Left:** The energy of the leptons in the reconstructed objects. **Right:** The p_T of the leptons in the reconstructed objects. The x-axis corresponds to the original MC simulation and the y-axis to the associated output of the PIPPIN model.

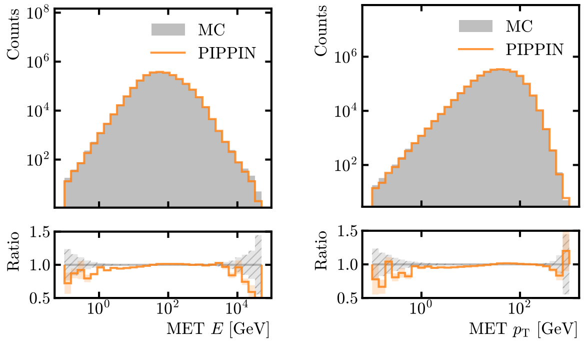


FIG. 17. Marginal distributions of the learnt features of the reco-level MET. **Left:** The energy of the MET in the reconstructed objects, as calculated in IV C. **Right:** The p_T of the MET in the reconstructed objects. The grey area corresponds to the original MC simulation and the orange line to the output of the PIPPIN model. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas.

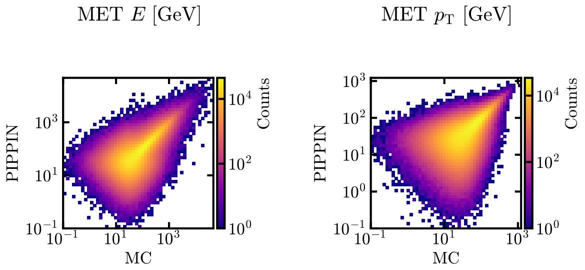


FIG. 18. 2D marginal distributions of the learnt features of the reco-level MET. **Left:** The energy of the MET in the reconstructed objects, as calculated in IV C. **Right:** The p_T of the MET in the reconstructed objects. The x-axis corresponds to the original MC simulation and the y-axis to the associated output of the PIPPIN model.

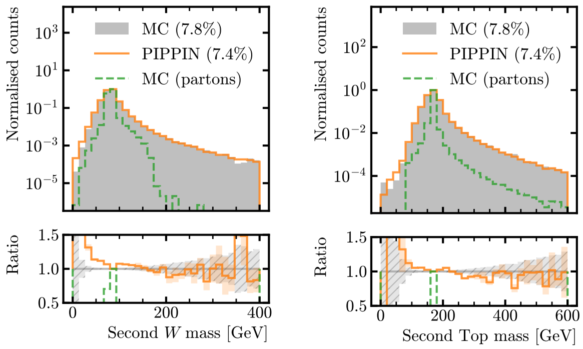


FIG. 19. Additional marginal distributions of the invariant masses of the underlying particles at reco-level. **Left:** The mass of the second reconstructed W boson. **Right:** The mass of the second reconstructed top quark. By second we mean the particle originating from the anti-top quark, as opposed to the top quark. The grey area corresponds to the original MC reco-level simulation, the orange line to the output of the PIPPIN model and the dashed green line to the MC parton-level simulation on which the model is conditioned. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas. The percentages indicate the proportion of events for which all partons are unambiguously matched and therefore present on the plots.

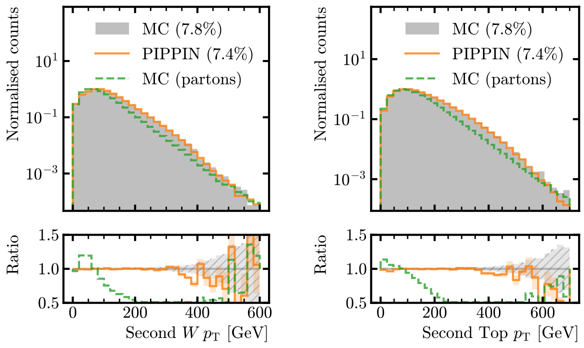


FIG. 20. Additional marginal distributions of the transverse momenta of the underlying particles at reco-level. **Left:** The transverse momentum of the second reconstructed W boson. **Right:** The transverse momentum of the second reconstructed top quark. By second we mean the particle originating from the anti-top quark, as opposed to the top quark. The grey area corresponds to the original MC reco-level simulation, the orange line to the output of the PIPPIN model and the dashed green line to the MC parton-level simulation on which the model is conditioned. The bottom plots show the ratios of the histograms with respect to MC and the uncertainties as shaded areas. The percentages indicate the proportion of events for which all partons are unambiguously matched and therefore present on the plots.

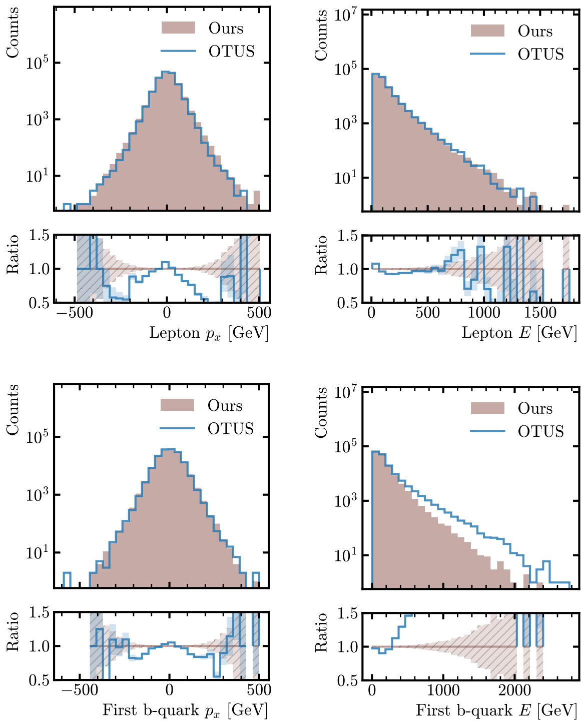


FIG. 21. Comparison of the parton-level distributions of the lepton and first b-quark energy and momentum in the x direction between the dataset presented in this work (brown area) and the one originally used by the OTUS and Turbo-Sim models (blue line). The bottom plots show the ratios of the histograms with respect to our dataset and the uncertainties as shaded areas.

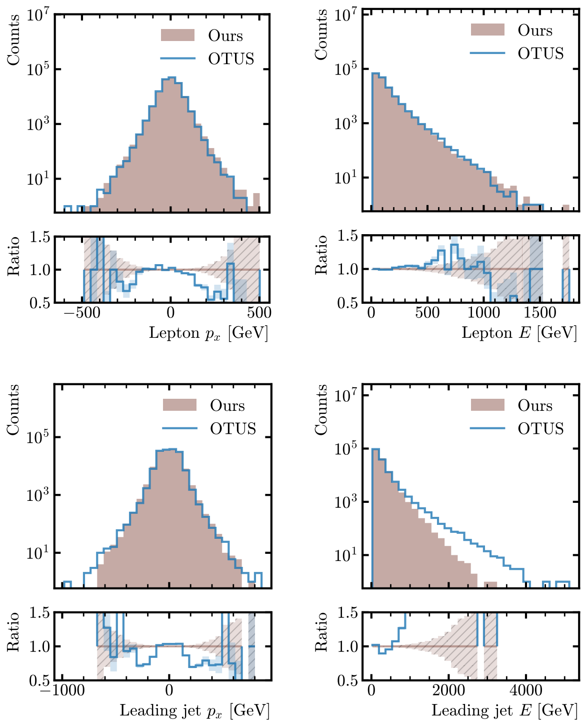


FIG. 22. Comparison of the reco-level distributions of the lepton and leading jet energy and momentum in the x direction between the dataset presented in this work (brown area) and the one originally used by the OTUS and Turbo-Sim models (blue line). The bottom plots show the ratios of the histograms with respect to our dataset and the uncertainties as shaded areas.