Diffusion Autoencoders for Sparse Graph Data Compression

Woody Hulse

Advisor: Richard Gaitskell
Reader: Daniel Ritchie

A thesis submitted in partial fulfillment of the requirements
for the Honors Bachelor of Science degree
in Computer Science

Department of Computer Science
Brown University
Providence, RI
April 2026

Abstract

Learned data compression is commonly approached using neural network autoencoders, which provide a standard framework for lossy encoding. Conventional autoencoders are often trained with mean-squared-error objectives, which bias reconstructions toward smooth representations and attenuate high-frequency structure that may carry meaningful signal.

To mitigate this effect in domains where data are not naturally structured on regular grids, we present a graph-native generative compression framework that conditions a graph denoising diffusion probabilistic model (DDPM) decoder on latent representations of continuous graph-structured signals. By replacing deterministic reconstruction with conditional diffusion sampling, the model learns distributions of plausible signals rather than smoothed averages. Importantly, our framework is built on sparse-only graph operations, allowing both compute and memory to scale linearly with graph size per diffusion step by avoiding the quadratic cost of dense graph methods.

We evaluate the model on calibration data from the LUX-ZEPLIN (LZ) dark matter detection experiment time projection chamber (TPC), a fixed-topology continuous-valued detector data stream, where it achieves a 494$\times$ reduction in representation size with minimal visual degradation in event reconstruction and outperforms the vanilla autoencoder baseline on reconstruction benchmarks. We show that the resulting encoded representations are higher-fidelity and more distributionally aligned than those of vanilla autoencoders. Further, we show that the learned latent space retains better separability of physics-relevant features, accelerates downstream tasks, reduces storage loads, and enables analyses such as event clustering and rare-event anomaly detection. Our results position graph-conditioned generative autoencoding as a scalable approach for reducing the storage and computational burden of high-volume, graph-structured continuous scientific measurement data while preserving essential information content.

Acknowledgments

I would like to thank my advisor, Richard Gaitskell, for his consistent support and guidance throughout the past two years. I'd also like to thank Chen Ding, Shawn Dubey, Chongwen Yu, Yinchen Zhou, Benjamin Almquist, Charles Kong, and the rest of the Brown Particle Astrophysics Group for their feedback, help, and kindness. Through weekly presentations and discussion of my work during group meetings, I developed significantly as a researcher. For that I am greatly appreciative.

I'd also like to thank my second reader, Daniel Ritchie, firstly for teaching possibly my favorite computer science course at Brown (CSCI1230: Computer Graphics), and also for taking the time to meet with me and ideate in the early stages of the project.

Contents

1 Introduction

1.1 Data Compression

Finding small and effective representations of data is a foundational problem in information science. Compression methods leverage patterns within data to form more efficient representations that consume fewer bits per canonical element. In practice, compression is useful in a variety of ways. Smaller data are easier to store, move, operate on, and analyze. Within these techniques is a dichotomy between "lossless" compression, which results in zero degradation of the underlying information content, and "lossy" compression, which sacrifices a small distortion function (or reduction of underlying signal in the data) in order to achieve greater theoretical data size reductions.

Algorithmic approaches, both lossless and lossy, are common in computing infrastructure. Where conventional algorithms fall short, however, in particular within domains where the nature of the data is ambiguous (i.e. there is no straightforward algorithm to efficiently encode information), modern methods invoke machine learning techniques to find patterns to further reduce data size.

The most common and longstanding data-agnostic deep learning approach to compression is the autoencoder, first formalized under the name "Autoassociative Neural Networks" by Kramer [1]. Autoencoders are a type of deep neural network that predicts an identity function on the input but with intermediate layers that are smaller than the input, forcing an information bottleneck which acts as a learned "latent" space.

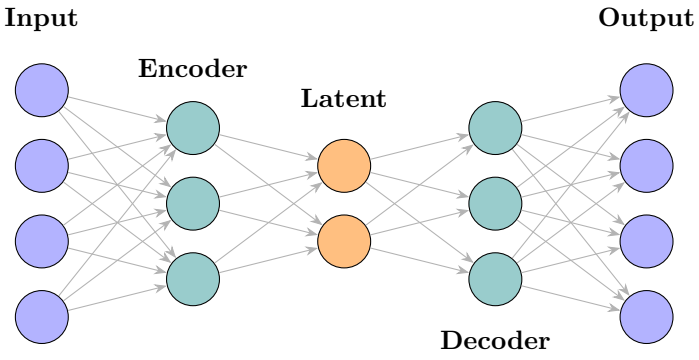

Figure 1: Autoencoder architecture. The encoder compresses the input into a low-dimensional latent representation z. The decoder reconstructs the original input from the latent.

Modern image and video autoencoding approaches have achieved strong compression ratios and have approached information-theoretic limits in practice [2].

However, autoencoders are a form of lossy compression, and a fundamental limitation is the nature of the distortion they impose on data. Conventional autoencoders typically attempt to minimize mean-squared error (MSE) between the original and reconstructed representations. For a weight vector θ, latent z, and decoder g_θ, autoencoders minimize

$$\arg\min_\theta \mathbb{E}\left[\|x - g_\theta(z)\|^2\right] \implies g_\theta^\star(z) = \mathbb{E}[x \mid z]. \tag{1}$$

At convergence, the optimum g effectively reconstructs the probability-weighted mean of all possible realizations of x given information from z. Because the encoding of z is lossy, in practice this is the average of a superposition of many possible representations, producing a "blurred" reconstruction (see Figure 2). This is especially prominent in high compression regimes, as with a smaller z there are more plausible reconstructions pigeonholed into similar latent representations.

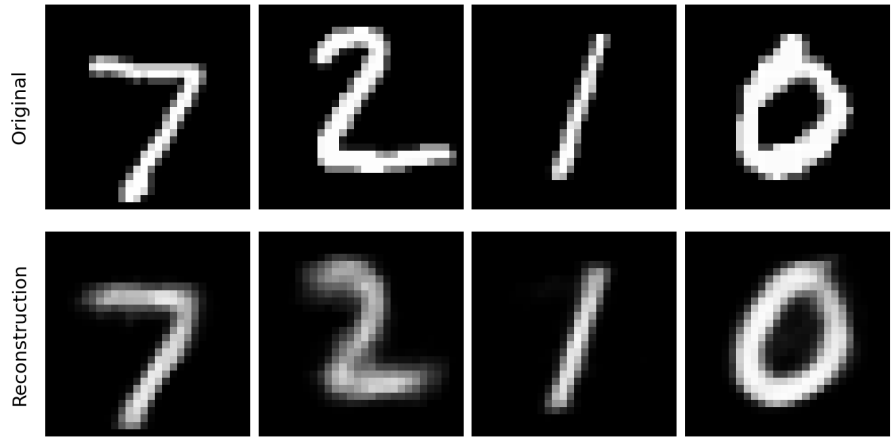

Figure 2: At high compression rates, autoencoders tend to "blur" reconstructions, effectively eliminating high frequency components from the representation.

Fine-grained detail, such as texture, sharp edges, or other high-frequency substructure vary in non-globally consistent ways. For instance, a single "texture" may have many valid phase realizations all consistent with the same global structure. Thus, a finite z minimizes these details. In scientific domains, it may not be permissible to forgo the representation of this entire class of signal, as that information may be important to model (distributionally, if not exactly) for the data to remain faithful.

The more recent adaptation to vanilla autoencoding is the variational autoencoder (VAE) [3]. VAEs model the latent representation as a normal distribution rather

than a single vector, but they fall to the same optimization objective trap for small latents as other autoencoders.

1.2 Generative Autoencoders

To address the smoothing problem in the general regressive framework, recent generative approaches have introduced different objectives better suited for modeling distributions of plausible events rather than mean-biased aggregates. Instead of reconstructing directly with respect to a latent z, generative decoders learn either an iterative denoising process [4] or a flow field [5] from a pure Gaussian distribution $\mathcal{N}(0,1)^d$ to the input distribution.[1] These models are extremely effective at creating within-distribution samples and are the backbone to almost all modern commercial image- and video- generative software.

Recently, stochastic generative approaches have been shown to be capable decoders [6]. Instead of generating from only a noise sample or a noise sample and a class or prompt embedding, diffusion and flow-matching autoencoders condition the diffusion process on the latent z vector. Where vanilla autoencoders reconstruct the average of all plausible samples given the finite information content in z (which in most domains lies out-of-distribution of true samples), diffusion and flow matching autoencoders can directly sample from the distribution of plausible samples given z through their choice in $\mathcal{N}(0,1)^d$. In practice, this enables high-fidelity reconstructions at far more extreme compression ratios [4, 9, 7].

1.3 Deterministic Reconstruction versus Conditional Diffusion

The advantage of diffusion decoders in this setting can be understood most directly by distinguishing point reconstruction from conditional density modeling. Consider first the deterministic decoder $g_\theta(z)$ trained with squared-error reconstruction from before. For a fixed joint distribution over data and latents, the population objective is

$$g_\theta^* = \arg\min_\theta \mathbb{E}_{p_{\text{data}}(x,z)} \left[\|x - g_\theta(z)\|^2 \right]. \tag{2}$$

The point-wise optimum is the conditional mean:

$$g_\theta^*(z) = \mathbb{E}_{p_{\text{data}}(x|z)}[x]. \tag{3}$$

Thus, if multiple distinct high-fidelity observations are compatible with the same latent code z, and if $p_{\text{data}}(x \mid z)$ is multimodal, a deterministic squared-error

[1] There are also adversarial generative models where a critic model evaluates the plausibility of generations; however, these have fallen out of favor in recent literature due to unwieldy competing objectives in theory and long optimization paths in practice [9]

decoder is encouraged to output an average of those possibilities.

We would like to solve a different problem. Instead of asking what the distance minimizing guess is given a conditional, we want to find the most likely plausible reconstruction given the information in z.

Consider a Denoising Diffusion Probabilistic Model (DDPM) decoder [4] conditioned on the latent z. The forward noising process gradually corrupts a clean event x_0 into Gaussian noise:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\right),$$

which admits the closed-form reparameterization

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I).$$

The model learns a conditional reverse process

$$p_\theta(x_{t-1} \mid x_t, z) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t, z), \Sigma_\theta(x_t, t, z)\right),$$

which defines the conditional reconstruction density $p_\theta(x_0 \mid z) = \int p_\theta(x_{0:T} \mid z)\, dx_{1:T}$.

The conditional log-likelihood $\log p_\theta(x_0 \mid z)$ is intractable since it marginalizes over the full denoising trajectory $x_{1:T}$. We instead optimize a variational lower bound, the conditional diffusion ELBO,

$$\mathcal{E}_{\text{DiffAE}}(x_0, z) = \mathbb{E}_{q(x_{1:T}|x_0)}\left[\log \frac{p_\theta(x_{0:T} \mid z)}{q(x_{1:T} \mid x_0)}\right] \leq \log p_\theta(x_0 \mid z), \tag{4}$$

which follows from Jensen's inequality applied to the marginalization integral. Standard manipulations (Appendix A) reduce the negative ELBO, up to timestep-dependent weights, to the denoising objective

$$\mathcal{L}_{\text{DiffAE}} = \mathbb{E}_{x_0,t,\epsilon}\left[w_t \left\|\epsilon - \epsilon_\theta(x_t, t, z)\right\|^2\right], \tag{5}$$

so minimizing $\mathcal{L}_{\text{DiffAE}}$ approximately maximizes a variational lower bound on $\log p_\theta(x_0 \mid z)$. Effectively, the diffusion autoencoder optimizes toward a maximum-likelihood objective in the generative space conditioned on the latent vector, which achieves our desired result.

This likelihood-based statement concerns the ideal density-modeling objective. Some generative decoders, such as normalizing flows, define exact likelihoods through a change-of-variables formula, while flow-matching models are often trained by a regression objective on a velocity field rather than maximizing an ELBO of $\log p_\theta(x \mid z)$. Thus, flow matching may still define a conditional density model in principle, but its standard training loss is an indirect surrogate

for density matching rather than the conditional maximum-likelihood objective written above. For scientific settings in which the residual distribution of high-fidelity events matters, the diffusion-decoder formulation gives a particularly clean likelihood-based justification for its choice in this work.

1.4 Time-Projection Chamber (TPC) Data

One of the most prolific experimental testbeds in modern physics is the time-projection chamber (TPC). TPCs observe subatomic interactions through the collection of scintillation light. In many liquid noble experiments, an incident particle deposits energy in a medium (e.g., liquid `Ar` or `Xe`), producing scintillation photons and freeing electrons. The electrons drift under an applied electric field and ultimately produce measured signals on sensor arrays (e.g., photomultiplier tubes, PMTs), which sample high-frequency spatiotemporal activity during an event [10].

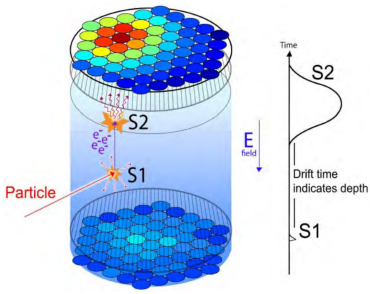

Figure 3: Model of a TPC chamber from the LUX-ZEPLIN conceptual design report [10] (p. 3-2).

Large-scale liquid TPC experiments produce high-volume streams of PMT data from arrays of hundreds or thousands of detectors sampling continuous waveform data at high frequency. For example, in the LUX-ZEPLIN (LZ) Weakly Interacting Massive Particle (WIMP) dark matter search, public data-management documents estimate total data volumes on the order of ∼1.1–1.2 PB per year during science operations [11]. Managing data at this scale poses significant challenges: storage, processing, and analysis costs all depend on the size and efficiency of data representations.

To accommodate circular detectors, many TPCs use irregular or six-fold symmetric PMT array patterns to maximize packing efficiency. The lack of grid symmetry prevents conventional image autoencoding methods which rely on regular fixed kernel operations applied in parallel across the data. Thus, TPC data demands more general graph methods when designing analysis models.

Many experiments already employ lossless compression tactics for storage, but

these methods offer limited compression and often have little utility for downstream analysis. For instance, LZ employs pulse-only digitization (POD) and zero suppression, which temporally splice active data regions for collection and ignore large spaces of near-zero data, which collectively reduce data size by $\sim 50\times$ [12] [13]. Lossy approaches have shown great promise in other domains, but have yet to see widespread adoption in LZ. A suitable lossy representation could reduce storage demands, enable faster transfer and retrieval, and make advanced analyses on broad corpora of data (e.g. machine learning-based event classification, comprehensive rare event searches past simple cuts, anomaly detection, etc.) computationally feasible.

1.5 The LUX-ZEPLIN (LZ) TPC

The LZ experiment is a next-generation dual-phase xenon time projection chamber containing 7 t of active liquid xenon, located underground at the Sanford Underground Research Facility [14]. The experiment searches for candidate WIMP dark matter particles. Dark matter constitutes approximately 83% of the total matter in the universe, yet has never been observed at the particulate level and currently does not reside in the Standard Model of particle physics. The ability to complete the objectives of the experiment is directly impacted by the quality of data representations.

Particle interactions within the liquid chamber produce an initial prompt scintillation (S1) and delayed electroluminescence (S2). Ionization electrons drift upward toward the liquid surface induced by a $\sim$97 V/cm field and transition into S2 photons collected by the PMT array 8 mm above the surface. Both signals are observed by 494 Hamamatsu R11410 PMTs (253 top, 241 bottom), digitized at 100 MHz with 14-bit precision via dual-gain amplifiers [15].

The S2 light is produced in the gas gap directly beneath the top PMT array, so the top array captures the spatially localized hit pattern that encodes the (x, y) position of the interaction. This 253-PMT top array is the focus of our analysis.

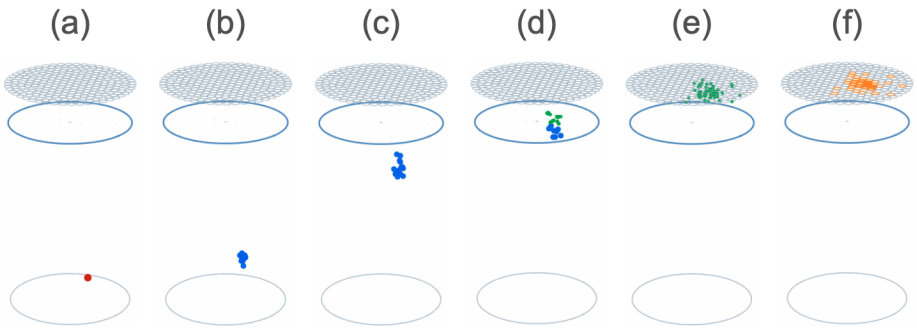

Figure 4: An S2 single-scatter event, simplified. An interaction occurs in (a) when an incident particle interacts with liquid xenon (red). A Poisson-distributed number of electrons (blue) are released and, induced by an upward electric field, take a mean-free stochastic path through the liquid toward the surface (b, c). At the surface, electrons scintillate into photons (green) (d), and are collected by PMTs (orange) sampled at 10ns (e).

The number of extracted electrons and the corresponding S2 size vary dramatically across LZ's physics channels. An S2 waveform block from the top array consists of 253 channels, each sampled at 10 ns intervals. An S2 pulse from a low-energy event originating from an interaction near the top of the detector occupies roughly 1–2 µs (100–200 samples). Including a surrounding window for pulse context, a typical extraction might span $\sim$10 µs, or 1,000 samples per channel. The raw data volume for a single S2 event in the top array is then

$$253 \text{ channels} \times 1000 \text{ samples} \times 2 \text{ bytes/sample} \approx 500\,\text{kB}. \tag{6}$$

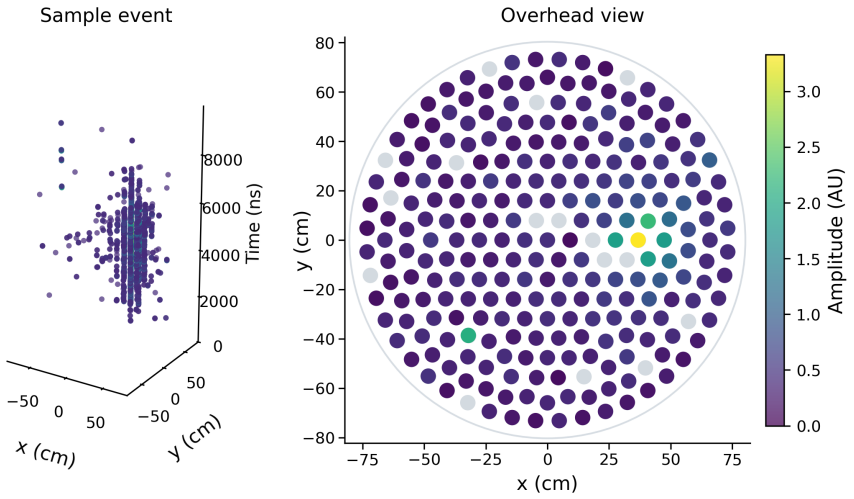

Figure 5: Example single-scatter tritium beta decay event, $1\mu s$ PMT trace (left) and cumulative PMT signal (right). One observable pathology in real TPC data are dead/malfunctioning PMTs (grey).

For a WIMP-like S2 (~ 300 phd), approximately 300 detected photons are distributed across the 253 top PMTs. The S2 hit pattern is sharply peaked around the (x, y) position of the event. Typically only 20–50 PMTs register any photons at all, and most of those record 1–5 photoelectrons. Each single-PE pulse occupies $\sim$50–100 ns (5–10 samples) within the 1,000-sample window. At higher energies, a 5 keV ER event illuminates more PMTs with more photons, and MeV-scale events produce dense, extended pulses across the full array. But even at 2,000 phd, the pulse still occupies only $\sim$200 of 1,000 samples on any given channel.

The S2 pulse shape is the convolution of the electron cloud's approximately Gaussian longitudinal diffusion profile with the single-electron electroluminescence response of the gas gap (~ 1 µs transit). For a given drift time (known from the S1–S2 delay), the envelope is largely predictable. The stochastic content is limited to the discrete Poisson arrival of individual electrons within that envelope and the binomial distribution of their photons across PMTs. The distribution of S2 light across the top array is determined primarily by the (x, y) position of the electron extraction point, the geometry of the gas gap, and the PMT positions. This is essentially a two-parameter family of light response functions, modulated by the total number of photons. The hit pattern for any event is well approximated by a scaled, shifted copy of a detector-specific template plus Poisson noise, though other characteristics of the TPC and particle interactions make direct modeling difficult. One such event type is the multi-scatter (MS) event, where there are interactions with two or more liquid xenon nuclei in the

chamber, thus inducing two separate scattering events. We treat MS events as a reasonable bound on the complexity of individual events; for that reason, we choose these as the compression targets rather than their much simpler single-scatter counterparts.

We can estimate the intrinsic information content of a top-array S2 waveform block by counting the underlying degrees of freedom. For a WIMP-like event with $n_e \approx 7$ extracted electrons:

- (x, y) **position:** sub-cm resolution over a 1.46 m diameter $\rightarrow$ $2 \times 7 =$ 14 bits.
- **Drift time** (known from the S1–S2 delay; sets the diffusion width): $\sim$ 14 bits (950 µs range at 10 ns resolution).
- **Number of electrons:** $\sim$4 bits (covering 1–15).
- **Arrival time of each electron** within the S2 envelope: $n_e \times \sim 7$ bits (1 µs window at 10 ns resolution) $\approx$ 49 bits.

Total: $\sim$81 bits $\approx$ 10 bytes. This is the information needed to fully specify the physics, from which the expected waveform on every PMT could be regenerated given a detector response model. The diffusion autoencoding framework admits this extreme compression case at high-fidelity, and some of our later analyses will approach that theoretical maximum bound.

However, this extreme bound ignores the stochastic photon-level detail or multi-scatter event characteristics. A more conservative estimate preserves the identity and timing of every detected photon. Each of the $\sim$300 detected photons carries: which PMT ($\log_2(253) \approx 8$ bits), arrival time within the S2 window ($\sim$7 bits), and pileup multiplicity ($\sim$1–2 bits), giving roughly

$$300 \times 17 \;\approx\; 5{,}100 \text{ bits} \;\approx\; 640 \text{ bytes}. \tag{7}$$

Compared to the $\sim$500 kB raw waveform (6), these bounds imply theoretical compression ratios of:

$$\text{Physics-parameter level:} \quad \sim 50{,}000 : 1, \tag{8}$$

$$\text{Photon level:} \quad \sim 800 : 1. \tag{9}$$

Conventional lossless methods like entropy coding achieve ratios of roughly 3:1 to 10:1. The gap between $\sim$ 10:1 and the photon-level bound of $\sim$ 800:1 represents redundancy in the event shape and characteristics that a physics-aware learned compressor can in principle exploit. Where on this spectrum a practical compressor lands depends on how much waveform-level fidelity must be preserved beyond the extracted physics quantities. Our work attempts to

approach this bound with modern methods that naturally accept stochasticity. We target a $494:1$ dimensionality reduction for the primary model.

Although the basic physics content underlying an event may be representable in a few floating-point numbers (x, y, z, e) as shown in the aggressive data estimate, modeling of the high-frequency structure of PMT waveforms in the LZ two-phase xenon TPC is essential for creating faithful event-level compressions. Low-frequency templates used to describe bulk S1 and S2 envelopes miss physics that directly impacts sensitivity. The digitized signal is fundamentally a superposition of ~ 10 ns single photoelectron (SPE) pulses with a non-Gaussian charge response, and in the few-photon S1 regime, smooth shape fits systematically bias photon counting and energy reconstruction downward. Double photoelectron emission ($\sim 20\%$ probability for 175 nm VUV photons on the R11410 bialkali photocathode [14]) further distorts the SPE charge spectrum, while PMT afterpulsing and dark counts inject correlated and uncorrelated SPE-scale features that masquerade as small signals if not explicitly simulated. In the S2 channel, isolated single-electron emissions from impurity photoionization, delayed extraction, and grid emission produce narrow few-PE pulses whose sub-microsecond structure is set by extraction and gas-gap physics. Capturing these phenomena requires waveform-level modeling at the SPE scale.

1.6 Graph Diffusion Autoencoders for TPC Data

Relevant literature within the physics space, including prior work on TPC data, has shown promise in creating compressed latent representations that preserve spatial information [16], while other autoencoder-based methods have been successful in compressing simulated spatiotemporal TPC data [18]. More recently in physics, researchers in the Kamioka Liquid Scintillator Anti-Neutrino Detector (KamLAND) experiment have shown that generative methods specifically within particle physics can appropriately model detector events modeled as point clouds [17]. Their work uses Generative Adversarial Networks (GANs) [8]. For the reasons cited earlier, we found this approach was difficult to scale to the size and greater high-frequency signal behavior of LZ data.

While diffusion and flow matching models have been studied primarily for generation (not compression), diffusion autoencoders have been shown in practice to create latent spaces that better preserve semantic separability than conventional autoencoders [6]. These models have found impact in several niche areas, primarily within the image and video compression space.[2]

Within graph-generative modeling, much recent work has focused on domains such as molecular generation, protein design, and crystalline materials discovery, where the graph topology, categorical attributes, and often continuous geometric

[2]Images and video frames are especially pragmatic applications because they exist on regular grids with trivially parallelizable operations on those grids.

coordinates are generated jointly [19, 20, 21, 22, 23, 24]. These settings motivate graph diffusion architectures, but differ from TPC compression, as they focus on discrete graph generation. In our formulation, the detector graph is fixed, and the object of interest is a continuous stochastic signal defined on that graph. Recent work on graph-signal diffusion addresses closely related known-graph signal-generation problems [25, 26], and diffusion-based graph autoencoding has also been studied for representation learning [27]. However, these methods are not designed as compression models for detector data, nor do they study conditional reconstruction from a compact event-level latent code. The closest detector-side comparator is recent variable-rate neural compression for sparse TPC-like data [28], which provides an important learned-compression baseline but does not use a conditional graph diffusion decoder. Thus, the present work is best positioned as a detector-scale conditional diffusion autoencoder for sparse, fixed-topology, continuous waveform compression.

The viability of unsupervised bulk training for representation learning on large-scale particle event data has been explored, most prominently by OmniLearned [29]. The authors demonstrated a transformer-based foundation model for jet physics pretrained on over one billion simulated and real jets which produces representations applicable to downstream tasks including jet tagging and anomaly detection. While the data modality (high-Q^2 collider jets) and downstream targets differ substantially from the TPC waveform setting, the underlying motivation is shared. Representations learned without explicit task supervision can expose physics-relevant structure that supports diverse analyses on the same compressed substrate. We pursue a similar objective in the low-energy detector regime with continuous-valued graph signals, in which training signal is attained by reconstruction rather than classification or tagging.

We claim that experimental particle physics data is a modality where diffusion autoencoders in particular are an appropriate choice for feature-rich data compression, storage, and analysis. Stochasticity in the event generation process lends itself to the inductive conditioned diffusion path taken by a random vector sample resolving to an event reconstruction in DiffAEs. By effectively modeling event distributions, we hypothesize that diffusion autoencoders will represent fine-grained SPE-scale data better than other methods at high compression rates. In this paper, we use synthetic multi-scatter tritium calibration data as an experimental testbed to evaluate the viability, strengths, and weaknesses of diffusion autoencoders on experiment-scale data. We show that, like image diffusion models have demonstrated improved separability of relevant image features, a diffusion model trained on TPC-recorded physics events may be able to learn to separate underlying physics from random particle movements while providing empirically less biased estimators of key analysis quantities and high-fidelity final reconstructions of representations nearly $500\times$ smaller than the original data.

To our knowledge, this is the first application of diffusion autoencoding to detector-scale, fixed-topology sparse waveform graphs, and the first such study for LZ-like TPC data compression.

2 Methods

[3]

Our autoencoder framework consists of three components: 1) a residually-connected graph convolutional network (GCN) encoder, 2) a GCN regressive decoder (for early training), and 3) a graph-based denoising diffusion probabilistic model (DDPM). The primary design consideration of our model is scalability; to optimize for large data with hundreds of thousands of features, any dense operations at the input- or output-level would become intractable due to memory constraints, even on modern hardware. This imposes a significant constraint on the types of modern GNN techniques that we can use.

A further constraint is the expressivity of graph operations. Because adjacency isn't regular, operations like kernels, which learn "weights" between adjacent nodes, aren't possible. Worse, common remedies to this limitation, such as graph attention, aren't available on sparse graphs without careful engineering.[4] As a result, we must engineer the network to be maximally expressive. Some general strategies we use are:

Graph Isomorphism: A fundamental theoretical limitation of message-passing GNNs is their expressive power with respect to the "graph isomorphism problem:" determining whether two graphs are structurally identical. Xu et al. [32] showed that standard message-passing GNNs are at most as expressive as the 1-dimensional Weisfeiler-Leman (1-WL) graph isomorphism test, because common neighborhood aggregation functions such as mean or max pooling are not injective over multisets and therefore collapse distinct graph structures to the same representation. The same work proposes the Graph Isomorphism Network (GIN), which uses a sum-based aggregator with an MLP to achieve a provably injective update rule, making it maximally expressive within the message-passing class. However, GIN matches the 1-WL bound rather than surpassing it, so non-isomorphic graphs that are indistinguishable by 1-WL remain indistinguishable by GIN.

[3] Code available at `https://github.com/woody-hulse/sgda`.

[4] Some modern formulations of graph attention reduce the quadratic cost of full pairwise attention by restricting interactions to sparse subsets of node pairs, either following the graph's edge structure or via learned sparsity patterns [31]; we found these approaches unwieldy and still far too expensive in compute and memory, though it's possible an engineering solution exists.

PNA Aggregation: Instead of only aggregating the raw messages, we aggregate distribution-level features across the hidden vectors of neighboring nodes. This has shown to be strictly more expressive, though also more expensive [33]. Corso et al. prove that in the continuous feature setting no single aggregation function is sufficient, and that multiple aggregators are theoretically necessary to distinguish between distinct node neighborhoods. PNA combines mean, standard deviation, minimum, and maximum aggregators alongside degree-based scalars, and demonstrates empirically superior performance over single-aggregator baselines such as GIN and GCN on graph-theoretic benchmarks.

Multilevel Residuals: To preserve low-level information that may be progressively removed from iterative convolution, we pass vector projections between layers close to the input space and layers close to the latent space. This is especially important for high-frequency signal modeling.

Adaptive Pooling: Self-Attention Graph Pooling (SAGPool) [34] is a learned graph pooling method that selects a subset of nodes to retain at each pooling step based on an "attention score" derived from node features. This approach is more powerful than regular top-k pooling [35], but not as expressive as soft pooling methods like DiffPool [36] which require the dense graph in memory.[5]

Stronger Conditioning: To engineer a graph network that observes conditioning information, we supply conditioning globally through Feature-wise Linear Modulation (FiLM) [38] and locally at the node-level. An MLP supplies each node with a synthesis of conditioning information, spatiotemporal position, and Laplacian graph embedding.

[5]We also implement a lightweight model that uses Graclus, a geometry-based edge-contractive pooling operation [37]. This approach may be more pragmatic than more expressive options, and in fact demonstrates comparable event reconstruction performance.

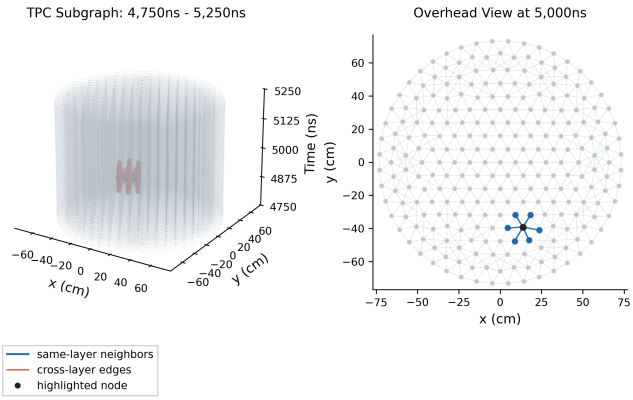

Figure 6: Graph generated from TPC spatiotemporal structure. Nodes are connected if they are within 12 cm in (x, y) and within 40 ns. The resulting graph is sparse, with a per-node degree ~ 78.6. Thus, the graph scales linearly with the number of nodes.

2.1 Data

Our data include 30,000 single-scatter S2 waveforms of LZ tritium calibration data, split with an 80/10/10 train/validation/testing ratio. Our framework should work for a diverse set of events, so to compensate for the regularity of single-scatter S2s we co-add two events at randomized temporal offsets (with no mixing between splits) to simulate a multi-scatter (MS) event to synthesize our data. We also augment a small amount of random Gaussian noise on events to prevent memorization of particular waveforms.

Each event is represented as a sparse graph whose nodes correspond to PMT channel-time tuples (x, y, z). For the 253-channel, 1,000-bin waveform data, this gives $N = 253 \times 1{,}000 = 253{,}000$ scalar-valued nodes per event. Edges are built from detector geometry and local temporal neighborhoods, using radius-based spatiotemporal connectivity. All graph operations are performed on this distance-normalized adjacency pattern. Since the sample rate and x, y position of all PMT reads are fixed, we operate off of the helpful assumption that the reconstructed graph must look structurally identical to the original. This is in contrast to other work on point cloud data [17] where the network must learn the position and number of nodes in the final representation as well as node values.

2.2 Encoder and Regressive Decoder

The encoder is a multiscale graph network. A linear input projection maps each scalar waveform value to a hidden node representation. At each scale, node states are updated by residual graph blocks that aggregate local neighborhood node feature statistics through sparse matrix addition/multiplication. Positional

features from normalized (x, y, z) coordinates and Laplacian embeddings are injected at each node. The graph after each layer is pooled with self-attention graph pooling [34]. After each graph message-passing stage, the learned scoring function assigns an importance score to every node. For each graph in the batch, the top $\lceil rN \rceil$ nodes are retained, with $r = 0.5$ pooling ratio. The retained node features are multiplied by a sigmoid gate of their scores, and the next encoder stage operates on the induced subgraph formed by the selected nodes.

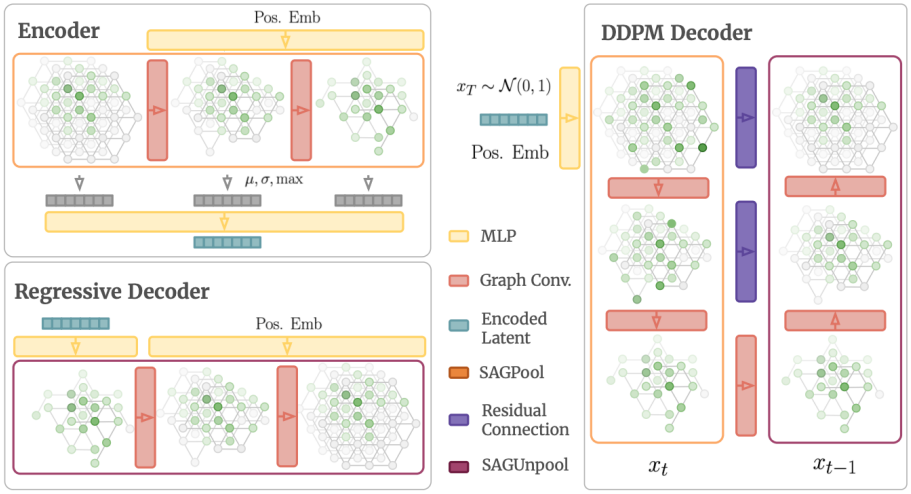

Figure 7: Overview of the graph diffusion autoencoder approach. Notable architectural deviations from conventional diffusion models are (1) the replacement of convolution/pooling layers with graph convolution/SAGPool, (2) FILM per-node conditioning of node features and the latent, and (3) the use of a regression head to steer the utility of the latent representation.

The latent vector is produced from a global multiscale readout. For each encoder pooling block, the model concatenates the MLP readout of the event at that scale. This pooled representation is augmented with anchor-pooling features, which summarize node activations around learned geometric anchors. A normalized MLP then maps the concatenated readout to the latent code $z \in \mathbb{R}^{d_z}$.

In addition to the diffusion decoder, the model includes a regressive decoder head to stabilize early training, encouraging useful latents. This head maps the latent code directly back to waveform space using a conditioned graph decoder over the same graph pooling pyramid. Unlike the diffusion decoder, the regressive head does not receive a noisy input or diffusion timestep and instead reconstructs x_0 directly from z.

For consistency, the encoder and regressive decoder architectures are directly mirrored in the baseline autoencoder model. The results then only characterize

the difference in power of the diffusion-based decoder model and objective.

2.3 Denoising Diffusion Probabilistic Decoder

The main decoder is a conditional graph diffusion model. Given a clean normalized event x_0, a diffusion timestep t is sampled uniformly and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ is added according to a cosine noise schedule:

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon.$$

The decoder receives x_t and a conditioning vector formed by concatenating the encoder latent z with a sinusoidal timestep embedding [4]:

$$c_t = [z, \phi(t)].$$

Architecturally, the diffusion decoder is a graph U-Net over the static temporal pyramid. It first projects the noisy scalar node input into the hidden dimension, adds positional embeddings, and applies conditioned residual graph blocks while downsampling through the temporal hierarchy. A bottleneck block operates at the coarsest scale, after which the model upsamples back to the original graph resolution. Skip connections transfer information from matching downsampling stages to the upsampling path.

Conditioning enters the decoder in two complementary ways. First, the global vector c_t produces scale-level FiLM-style shifts, scales, and gates for the residual blocks. Second, node-local conditioning is built by concatenating c_t with each node's normalized spatial-temporal position and projecting this through an MLP. This lets the decoder use both global event information and local detector geometry when predicting the denoising target.

The default training objective uses the velocity parameterization,

$$v_t = \sqrt{\bar{\alpha}_t}\epsilon - \sqrt{1 - \bar{\alpha}_t} x_0,$$

rather than directly predicting ϵ, which has been shown in recent diffusion work to be a more principled and practically efficient objective [30]. At sampling time, the model converts the predicted velocity back to an estimate of x_0 and applies the standard DDPM posterior update.

Stage	Module	Default shape	Details
Input representation	Spatiotemporal graph	$x_0 \in \mathbb{R}^{N\times 1}, \quad N = C \times T$	One node per *(PMT channel, time bin)* sample.
Encoder	`GraphEncoder`	hidden width 64, depth 4, 2 blocks/stage, pool ratio 0.5, hidden dim 1024, latent dim 512	The encoder maps the normalized event graph to a global latent code z. Residual connections map each stage to the latent; these are synthesized by a final MLP.
Time conditioning	Sinusoidal timestep embedding	$e_t \in \mathbb{R}^{64}$	The diffusion timestep t is embedded with a sinusoidal positional encoding. The decoder conditioning vector is the concatenation $c_t = [z; e_t] \in \mathbb{R}^{1024+64}$. This condition is used in two ways: (i) global FiLM-style modulation, and (ii) node-wise modulation obtained by repeating c_t across nodes, concatenating with node position, and passing through a node-conditioning MLP.
Forward diffusion	Cosine noise schedule	$T = 250$ steps	During training, a timestep t is sampled uniformly and noise is added as $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon$, $\epsilon \sim \mathcal{N}(0, I)$. with cosine scheduling.
Conditional denoiser	`GraphDDPMUNet`	input dim 1, hidden width 64, depth 4, 2 blocks/stage, pool ratio 0.5	The diffusion decoder is a graph U-Net operating on the noisy graph x_t, conditioned on $c_t = [z; e_t]$ per-node. It predicts a per-node diffusion target.
Prediction target	Velocity parameterization	output velocity $\hat{v}$	v-parameterization, with training target $v = \sqrt{\bar{\alpha}_t}\, \epsilon - \sqrt{1-\bar{\alpha}_t}\, x_0$.
Primary objective	Diffusion MSE	scalar loss	The main loss is mean-squared error between the decoder prediction and the chosen diffusion target at $t \in T$.
Auxiliary reconstruction head	`GraphDecoder`	hidden width 64, depth 4, 2 blocks/stage	For early training. A second decoder reconstructs x_0 directly from z and adds an auxiliary reconstruction loss.
Inference	Reverse diffusion	$x_T \sim \mathcal{N}(0, I)$	At reconstruction time, the input event is first encoded to z, then reverse diffusion is run from noise using the conditional decoder. A separate routine also supports sampling directly from a provided latent vector z.

Table 1: Summary of diffusion autoencoder components and hyperparameters under the primary configuration.

2.4 Training

Training uses multi-scatter event data, generated online. For each minibatch, the encoder first maps the clean event x_0 to a latent code z. A timestep t and Gaussian noise ϵ are sampled, producing the noised event x_t. The diffusion

decoder predicts v_t,[6] and diffusion loss is computed by the mean squared error over all graph nodes:

$$\mathcal{L}_{\text{diff}} = \mathbb{E}_{x_0,t,\epsilon}\left[\|f_\theta(x_t, z, t) - v_t\|_2^2\right].$$

Early in training, a second regressive loss is added on the regressive decoder:

$$\mathcal{L}_{\text{reg}} = \|g_\psi(z) - x_0\|_2^2.$$

forming the full objective

$$\mathcal{L} = \mathcal{L}_{\text{diff}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}},$$

with λ_{reg} starting at 1.0 but annealing towards zero throughout training.

Optimization uses AdamW with learning rate 5×10^{-4}, batch size 8, gradient clipping at norm 5, cosine learning-rate decay, and mixed precision on CUDA. We performed an ad-hoc grid search for parameter families that were computationally efficient and performed well in the diffusion reconstruction benchmark.

We train models at varying compression ratios, ranging from 250× to 63, 250×. To manage training costs when training a sequence of compression rates models, we scale the number of feature nodes inversely with the compression ratio. For example, we use a single summed data channel at $63k\times$ compression up to the full 253-channel data at 250× to 500× compression. All cross-comparisons between models are made using a common input form so as to eliminate any biases that this may introduce at higher rates.

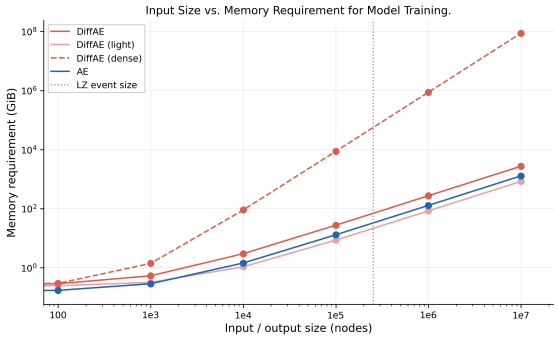

Figure 8: The memory scaling behavior of each autoencoder, compared. The non-sparse counterpart to our model exceeds consumer hardware limits past an input size of ∼ 10,000. Our architecture (and in particular the light version) can plausibly be trained on event sizes 1, 000× greater.

[6]We use a more stable v-parametrization as opposed to direct prediction of x_{t-1}. The linear combination maintains the properties in Section 1.3 [30].

At our primary latent target, we encode 512 `fp16` numbers. From the original event stream of $253,000$ `fp16` numbers per event, we achieve an effective compression of $494.14\times$. When both the input and latent are stored in fp16, this corresponds to a reduction from 253,000 values (about 506 kB per event) to 512 values (about 1.02 kB per event). Under this matched-rate setting, a vanilla autoencoder and a diffusion autoencoder have essentially the same nominal bitrate if both serialize their latent codes directly in fp16. Accordingly, the relevant comparison is not additional bitrate savings between the two models, but rather the rate-distortion performance. Any advantage of the diffusion autoencoder at this operating point should therefore be described as improved reconstruction fidelity and better preservation of physics-relevant waveform structure at the same $494\times$ compression ratio, reflecting the stronger generative prior of the diffusion decoder rather than a smaller stored representation.

2.5 Experiments

The diffusion autoencoder should reconstruct realistic, in-distribution events at high compression rates. The encoded representations of the DiffAE should also be more feature-rich and useful than their vanilla autoencoded counterparts. We evaluate both claims.

2.5.1 Event Reconstruction

Reconstruction Fidelity. Any generated event sample from a real encoded latent should closely align qualitatively with the original event, even at high compression rates.

Distributional Analysis. Accurate reconstruction should match not only individual events, but also the ensemble statistics of the dataset. A core motivation for choosing the diffusion approach was for precise modeling of pixel-level variance. To study this, we examine both event-level statistics and nodewise means, distributions, residual dispersions, and biases to assess whether the model reproduces realistic event-level and detector-level variation.

Reduced Quantities (RQ) Analysis. We use reduced quantities to evaluate whether reconstructions preserve the physically meaningful pulse-shape summaries that are commonly used in downstream analysis. For each held-out event and its reconstruction, we compute the channel-summed z-profile and a set of standard RQs including peak amplitude, peak time, total integral, rise time, fall time, full-width half-maximum (FWHM), 10-90 width, and temporal standard deviation. We then compare raw and reconstructed values for both the vanilla autoencoder and diffusion autoencoder with per-RQ correlation plots, residual histograms, and aggregate error summaries, which reveal both systematic bias and event-by-event fidelity in RQ space.

2.5.2 Latent Representation

The capacity to produce more faithful reconstructions would demonstrate the expected behavior of the diffusion autoencoder producing more distributionally-aligned reconstructions than the autoencoder. However, these claims don't explicitly support the idea that the *latent space* is more powerful in the DiffAE. This is the more subtle claim, and for any implementation of the encoder in practice, is the arguably more important argument. To evaluate these claims, we develop the following tests to examine the utility of the latent representation.

Latent Separability. We probe whether the latent space cleanly separates event classes that differ in underlying scattering structure. For the experiment, we consider two important feature characteristics, one of which ($\Delta\mu$ separation — see section on MS $\Delta\mu$ prediction) being a continuous-valued time measure of the event vertex separation for MS events, and the other a discrete-valued classification of different types of events. For the latter, we synthetically generate a new class of scatters called "lopsided" events based on event archetypes observed in real LZ data. These events have a low-pass filter ($\sigma = 30$ns Gaussian filter) applied temporally on one side of the pulse, resulting in a lopsided variance pattern.

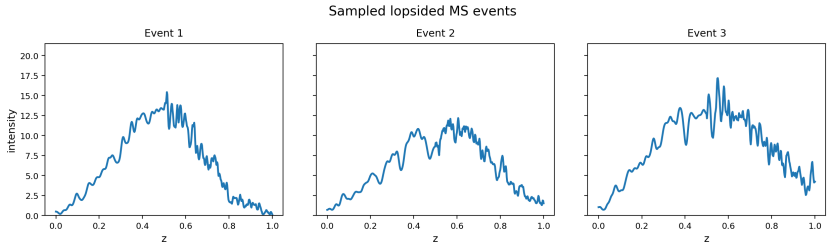

Figure 9: Three sampled lopsided events with a low-pass Gaussian filter ($\sigma = 30$ns) applied on the left half of the pulse.

We fine-tune each model on each class of events and then examine the projected latent spaces, using a small MLP to determine if the event information is preserved.

Anomaly and Rare Event Detection. A useful latent representation should make unusual or pathological events easier to identify than hand-crafted summaries or the full-feature event space. Several works on anomaly detection in data streams [39, 40] have shown that autoencoders are a useful mechanism in unsupervised event detection. To test this, we construct a set of synthetic anomaly prototypes from real single-scatter events, including spatial anomalies that preserve the temporal profile and temporal anomalies that preserve the

spatial weighting while altering the waveform shape. We embed both real events and anomaly prototypes in RQ space and in the learned latent spaces and score each anomaly type by its Mahalanobis distance percentile relative to the in-distribution cloud.

Auxiliary Task: Multi-scatter Prediction. A primary objective of data compression is accelerating analysis tasks. We estimate that we can achieve minimal performance degradation in arbitrary downstream analysis with significant speedups using the learned latent representations. We first encode a large set of online-generated multi-scatter events (see Section 2.1) and save each latent vector together with its ground-truth $\Delta\mu$ time separation between the constituent scatters. A lightweight MLP regressor is then trained on frozen latents to predict $\Delta\mu$ on held-out data, comparing the performance of the autoencoded representations to those of the diffusion-autoencoder with respect to latent size.

Diffusion as an Event Generator. Because our DiffAE includes a generative decoder, we can construct an empirical latent distribution

$$\{\mathcal{N}(\mu_i, \sigma_i)\}_{i=0}^{d}. \tag{10}$$

Sampling from this distribution, we get brand-new within-distribution latents to pass through the decoder.[7] This gives us a simulation-free, fast, and prior-free event generator that can reproduce any pathologies/irregularities in the detector inexplicable by the more basic physics processes. We fit a simple latent prior, draw random latent vectors, and run the reverse diffusion process conditioned on those latents to synthesize new events. The resulting samples are inspected through charge maps, summed waveforms, and summary statistics to assess whether the model generates plausible detector responses with realistic spatial and temporal structure.

3 Results

3.1 Event Reconstruction

3.1.1 Generation Fidelity.

We first examine the ability of our model to compress and reconstruct high-dimensional TPC data to evaluate the claim that our method can better model event characteristics, in particular high-frequency noise.

[7] A more principled approach would be to either a) constrain the latent space to be an isotropic gaussian by adding a KL-divergence loss or b) build a separate diffusion model for latent sampling. For the proof-of-concept, we take the empirical approach to be sufficient.

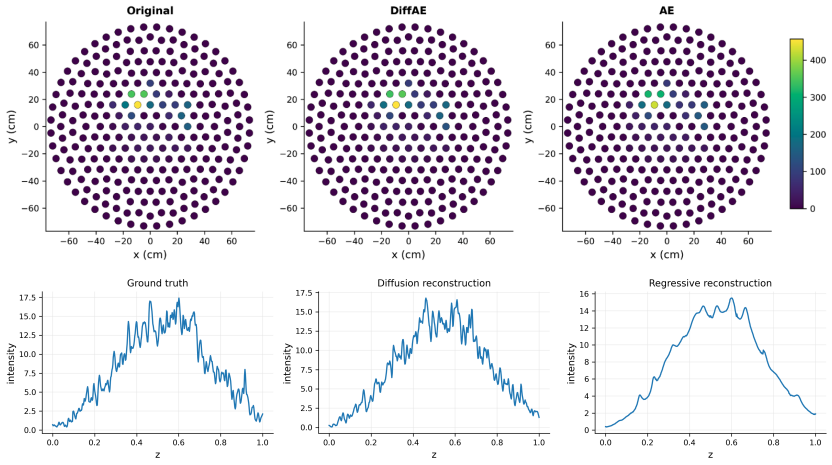

Figure 10: Cross sections over (x, y) and z (time, normalized) in the middle 500 samples at 500× compression. The diffusion-based autoencoder reconstruction is able to model higher-frequency signal details that the regression-based autoencoder reconstruction can't represent.

Figure 11: Channel-level reconstructions (middle 500 samples). The diffusion autoencoder (red) preserves stochasticity in low-amplitude tail areas where the regressive autoencoder is smooth.

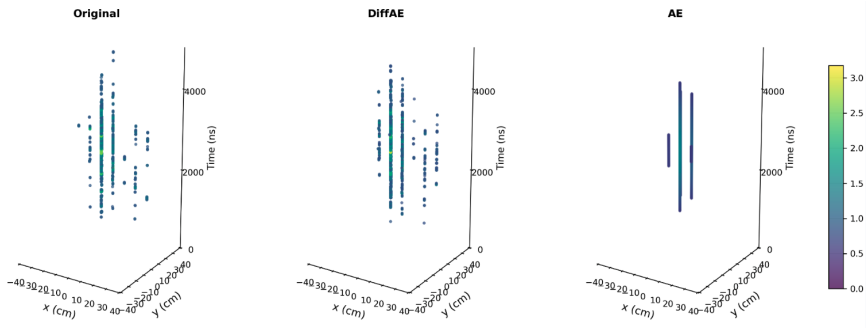

Figure 12: 3D reconstructions of event at 500× compression.

We find that, even at high compression rates, the diffusion autoencoder produces qualitatively believable events, whereas the autoencoder baseline produces notably smoothed reconstructions. We find that, as expected, the iterative diffusion process can in principle produce a believable event at any compression rate.[8] Without conditioning, the DDPM samples over the distribution defined by the entire event population. By increasing latent size, we are able to achieve reconstructions more resolved to the individual event's distributional subspace.

3.1.2 Distributional Analysis

To substantiate our claim that the graph diffusion autoencoder approach produces better results, we investigate further the sample-level and event-level distributional characteristics. We find that, as expected, the vanilla autoencoder significantly underestimates the marginal dispersion of samples in the time domain (where stochastic behavior is the most prevalent) as a result of mean-biasing, while the diffusion approach produces a near-exact trace of the ground truth event pixel-level dispersion.

[8]This is likely also the case with no latent conditioning whatsoever.

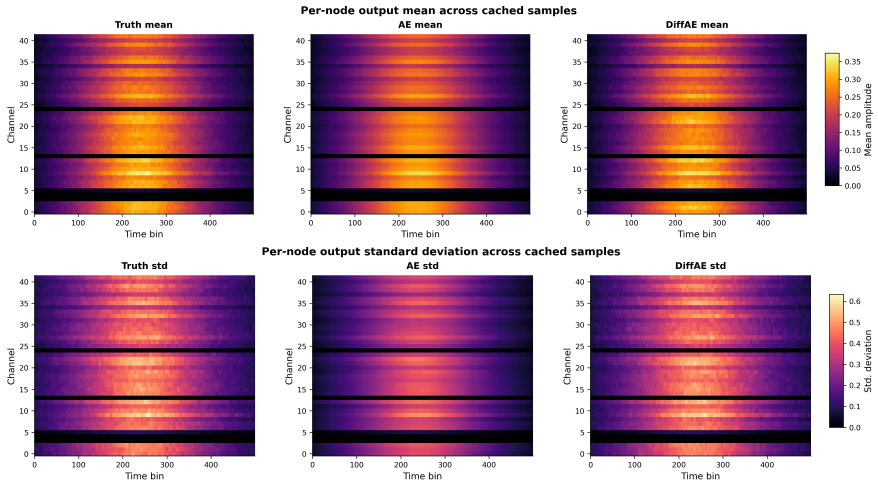

Figure 13: Aggregate node-level mean and standard deviation for a set of 1024 test events. Characteristics of the diffusion autoencoder reconstruction distributions much more faithfully represent the ground truth.

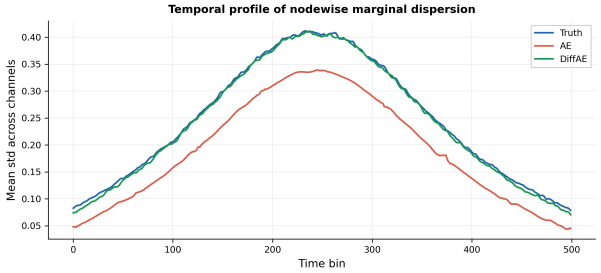

Figure 14: Aggregate node-level standard deviation in the time domain for a set of 1024 test events. The trend for the diffusion autoencoder matches the ground truth: autoencoder reconstructions underpredict variance.

We also evaluate event-scale distributional characteristics. We find similar results: the DiffAE is able to better fit the distribution of total PMT energy deposition and event centroid in x, y, and z.

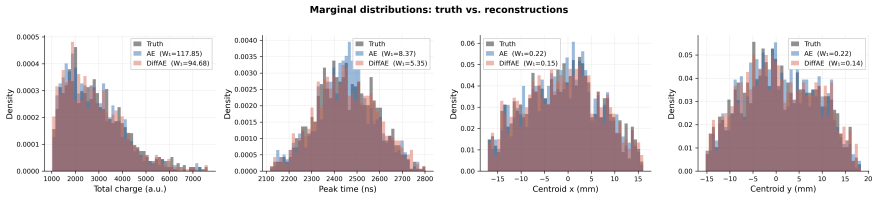

Figure 15: Aggregate event-level event positioning and size statistics representing the core 14-bit event characteristics. The Wasserstein distance to the ground truth distribution is minimized by the DiffAE.

3.1.3 Reduced Quantities (RQ) Analysis

We find that in all eight measured RQs, the diffusion autoencoder maintains little to no distributional bias while reconstructing the quantity with high accuracy. Whereas the smooth vanilla autoencoder reconstructions consistently skew RQs, the deviation of the distributional means in each RQ space of DiffAE generated events is near-zero (Figure 16). Importantly, the error width of DiffAE generated RQs are consistent with the autoencoder, which practically bounds the performance without explicit supervision of the RQ values on the latent encoding.

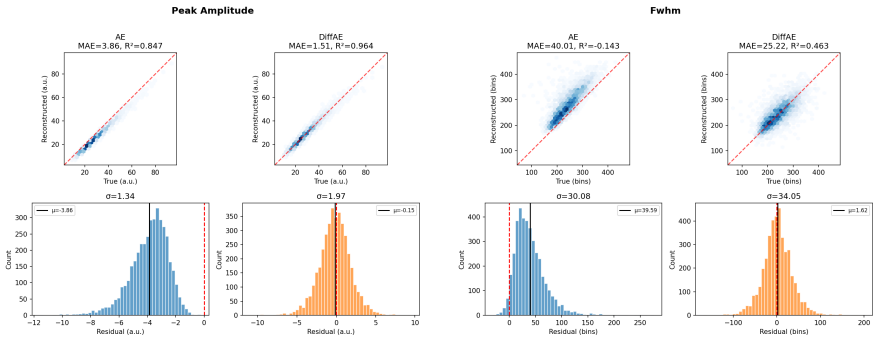

Figure 16: Full-width half maximum (FWHM) and peak amplitude reduced quantities for a vanilla autoencoder (AE) and diffusion autoencoder (DiffAE) at 500× compression. The AE events show strong systematic biases across RQs, which are not evident in DDPM-generated events. See Appendix B for all RQs.

As a result of this bias reduction, we also find significantly better distribution-level alignment and lower error rates in DiffAE reconstructions (Figure 17).

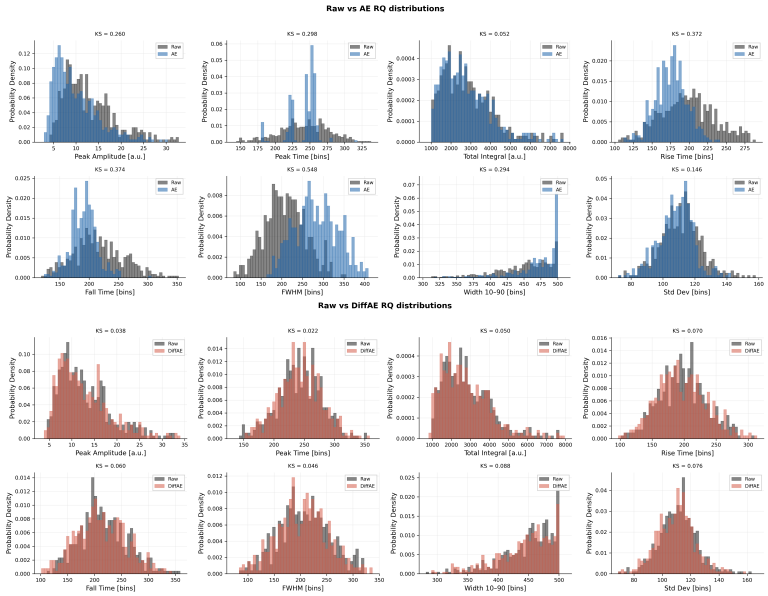

Figure 17: Distributions of eight RQs from autoencoder compared to GT (top) and diffusion autoencoder compared to GT (bottom).

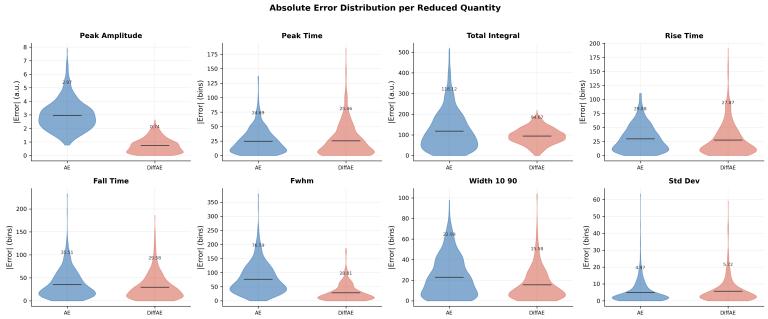

Figure 18: Error distribution on the suite of eight RQs. Due to its biases, the autoencoder has significantly greater error in seven of eight.

3.2 Latent Representations

We then investigate the utility of the latent representations, which we find not only useful for a variety of tasks, but differentially better under the diffusion autoencoder compression scheme than the conventional autoencoder.

3.2.1 Latent Separability

We investigate the ability for the model to discriminate between different event types within the latent space. Figure 19 shows the latent representation of the $\Delta\mu$ multi-scatter event property. $\Delta\mu$ is a measure almost entirely explainable by the low-frequency shape of events. Thus, both autoencoding approaches separate well in latent space. Conversely, despite event noise lopsidedness being a visually straightforward event class to distinguish, the difference is encoded entirely in high-frequency signal. Figure 20 shows that the vanilla autoencoder approach, where reconstructions lose this signal regardless, are unable to encode lopsidedness information, evidenced by non-separability in the projected latent space. The diffusion autoencoder on the right pane maintains high separability of these events, demonstrating that the DiffAE is able to encode this information.

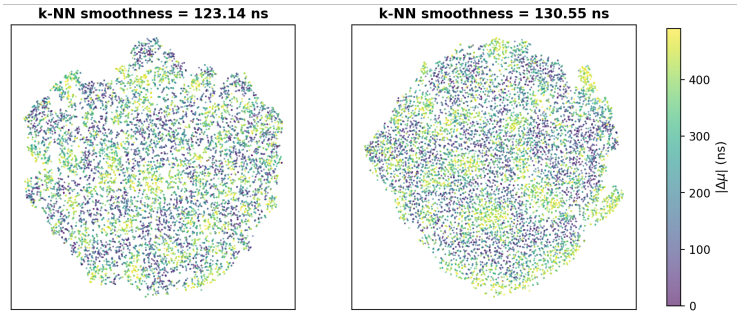

Figure 19: TSNE projection of compressed latent vectors from the autoencoder (left) and diffusion autoencoder (right) at 2,000× compression. k-NN smoothness is a measure of the mean variance of the $k = 10$ nearest latent vector $\Delta\mu$. Both methods achieve comparable separability.

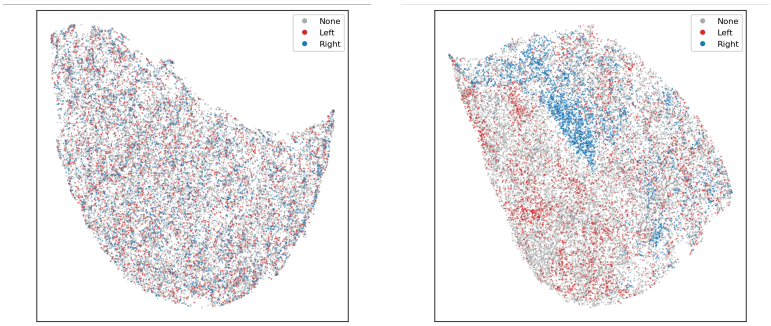

Figure 20: UMAP projection of compressed latent vectors from the autoencoder (left) and diffusion autoencoder (right) at 2,000× compression. Red nodes indicate the lopsidedness smoothing applied on the left side of the event, blue nodes indicate smoothing applied on the right, and gray nodes have no lopsidedness. The diffusion autoencoder is able to maintain high separability of these events, while the vanilla autoencoder is unable to encode this information.

We can see this discrepancy in Figure 20 numerically when training a lightweight MLP on each latent representation to predict $\Delta\mu$ and classify the lopsidedness application.

	Full Event	**AE Latent**	**DiffAE Latent**
MLP $\Delta\mu$ MAE (ns)	58.2	**58.9**	59.2
MLP Accuracy (%)	100.0	54.0	**99.7**

Table 2: Accuracy of MLP on latent representations: MS $\Delta\mu$ MAE and event lopsidedness classification.

This improved expressivity of DiffAE-encoded lopsided events is perhaps more visible when investigating the resulting reconstructions. When examining AE reconstructions next to those of the DiffAE, the failure mode becomes apparent. Even with heavy lopsidedness, the autoencoder presents no visible difference in the left and right half of lopsided events–they present as uniformly smoothed. The diffusion autoencoder, on the other hand, is able to capture the difference in variance (Figure 21).

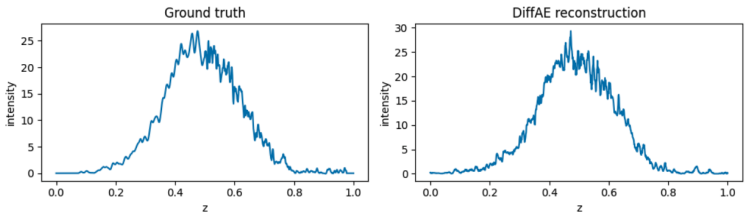

Figure 21: Example left-lopsided event reconstruction of autoencoder (top) and diffusion autoencoder (bottom). By smoothing the reconstruction, the autoencoder is unable to represent variance characteristics whatsoever in the output or, more importantly, in latent space.

The practical implications of this experiment are significant. There are entire classes of deviations, event types, and detector pathologies that are functionally invisible to a vanilla autoencoder at high compression. Through modeling the event distribution itself, the DiffAE is more agnostic.

3.2.2 Anomaly and Rare Event Detection

We find that, for the ten types of synthetic anomalous events the DiffAE latent space scores the highest in seven in terms of Mahalanobis distinguishability. For all but one event, the DiffAE latents score the same or better than the AE latents. For four of the event types, the event is in the $> 99^{\text{th}}$ percentile in distributional deviance. This suggests that, without any supervision on the explicit pathologies or the event characteristics at all, the DiffAE is already a reliable means of separation.

Anomaly type	RQ metrics	AE	DiffAE
Spatial anomalies			
Uniform spatial	51%	89%	**99%**
Peripheral PMTs	51%	**99%**	**99%**
Single PMT	51%	**99%**	**99%**
Checkerboard	51%	92%	**94%**
Temporal / waveform anomalies			
Square wave	**99%**	98%	**99%**
Smooth Gaussian	**82%**	72%	79%
PMT saturation	42%	**58%**	56%
Diffusion smear	**91%**	65%	81%
Delayed echo	64%	61%	**67%**
Stretched tail	55%	58%	**72%**

Table 3: Mahalanobis percentile of the encoding of ten anomalous event types.

3.2.3 Auxiliary Task: Multi-scatter Prediction

Ultimately, the latent representation should be useful for downstream analyses. The representation should be agnostic to the type of analysis—thus, we do not perform any explicit supervision on these tasks in training (otherwise, we risk biasing toward a subset of possible use-cases). We also expect that these tasks will be much faster, both in training time and training iterations, with a greatly compressed representation. In the $\Delta\mu$ prediction task, we find that both representations perform at parity with the raw event at extremely high compression rates (2,000×):[9]

[9]The error measure in Figure 22 does not capture all modes of variance. A better measure, for example, would be to train multiple autoencoders to form multiple latent representation sets and use the convergence of MLPs for those different latent data.

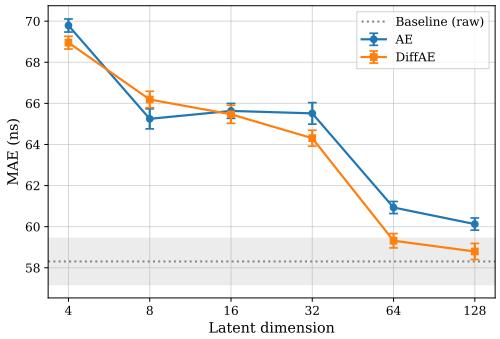

Figure 22: Comparison of MAE accuracy in $\Delta\mu$ prediction at different latent sizes, compared to a model trained on raw data. Both representations achieve comparable performance in the downstream $\Delta\mu$ prediction task, even at very high compression ($\sim 100,000\times$ to $\sim 2{,}000\times$), after which point they are entirely at parity. Error bars indicate the difference in converged validation MAE for MLPs trained on the same latent encoding set with K-fold cross-validation.

We also confirm the training speed differential. While training speed has no general relationship with data size (as the scaling nature is model- and task-specific), we find that in the $\Delta\mu$ prediction task convergence speed is $\sim 50\times$ better at the encoded scale than the raw data, with convergence uniformly in fewer epochs in the former case.

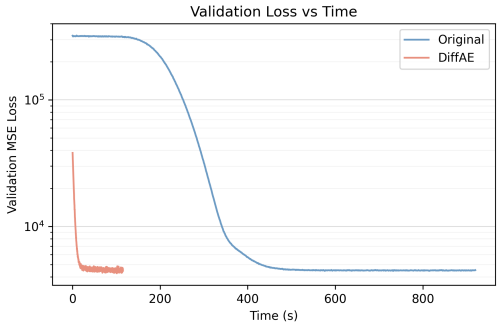

Figure 23: Training time vs. validation accuracy. As expected, the $2,000\times$ compressed representation trains much faster than the full data model.

3.2.4 Diffusion as an Event Generator

We find that, even using a hand-constructed empirical prior, latent samples drawn from a distribution produce qualitatively realistic new event samples. While the diffusion process is likely more compute-intensive than an algorithmic simulation of novel events, it's likely that this latent distribution and diffusion

reconstruction is able to represent various TPC- and PMT-level characteristics and simulation that aren't visible in the prototypical event physics.

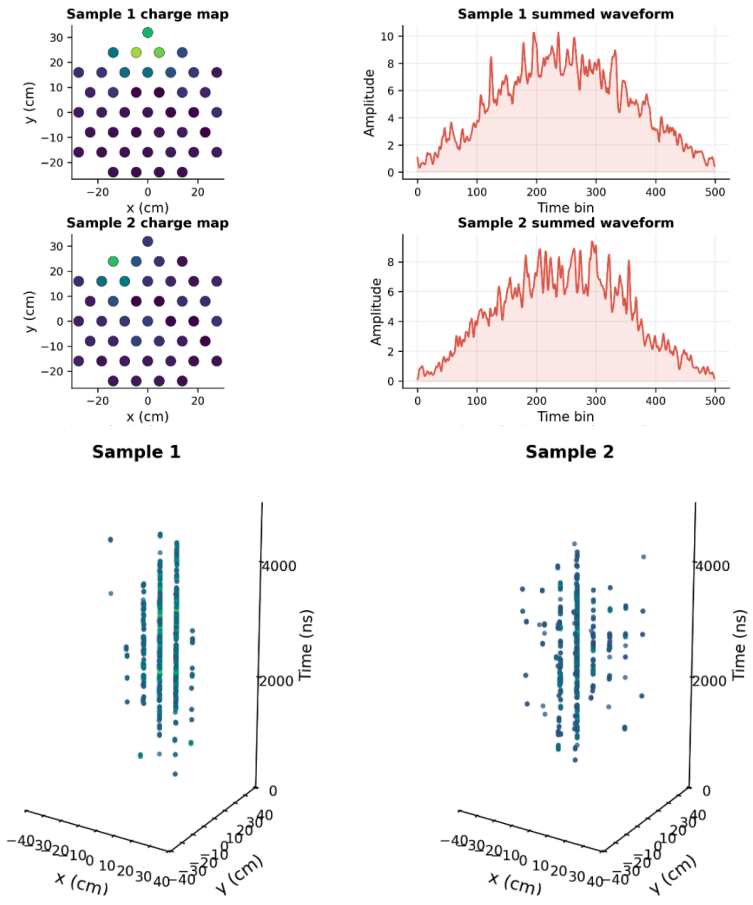

Figure 24: Samples drawn from the empirical latent distribution described in 10 and reconstructed through the decoder.

More work is required to investigate the physics viability of these events. The power to select a part of a feature-rich distribution and pick new, prototypical events could be an interesting avenue for future study for event classification.

4 Discussion

4.1 Summary of Findings

We've demonstrated the viability of diffusion autoencoders on large-scale sparse graph datasets. We show that in scientific domains like the LZ experiment where modeling of high-frequency signal is important, the diffusion-based approach exceeds baseline autoencoding approaches in reconstruction fidelity, distributional alignment, and RQ preservation. We also find that the resulting latent representations are generally more useful and agnostic to the original event in the DiffAE case. Our model is memory- and compute-efficient, trainable on large experimental datasets with high-grade consumer hardware.

Our results collectively support two hypotheses. Most obviously, at high compression rates the mean-biasing optimization objective of conventional autoencoders results in smoothed events that distort visual clarity (Figures 10 and 12), distributional event characteristics (Figures 13, 14, 15), and physics-relevant quantities (Figures 17 and 18) in practice.

A plausible counterargument is that this is expected behavior for autoencoders and irrelevant to the encoding task. Because the nature of the distortion in distribution and RQ space is generally a systemic bias rather than correlational, a linear correction alone on recovered autoencoder RQs would likely recover a significant amount of the error discrepancy between the AE and DiffAE.

The second, more subtle result, is that the biases in the autoencoder reconstruction directly impact the types of events that the encoding itself is able to represent, as well as the viability of the latent space to perform downstream tasks. The latent separability task reveals a symptom of this effect. For a generic synthetic event class ("lopsided" events), the vanilla autoencoder was unable to separate events in that class from other events in latent space. This is almost certainly emblematic of an issue where any event type that contains meaningful signal above the effective frequency capacity of the encoder can not be encoded by an autoencoder. By modeling the node-wise distributions as a diffusion process, the DiffAE properly reconstructs these events in the output. This representational ability confers directly to separability in the latent space (Figure 20). In essence, instead of suppressing signal below a pixel-space frequency upper bound in the autoencoder case, the log-probability objective bound of the diffusion autoencoder chooses the most condensable underlying signal that characterizes the conditional distribution.

Based on our results, the DiffAE therefore presents a more visually defensible decoding and agnostic encoding regime compared to the autoencoder. To further confirm these results (particularly the second), future work should evaluate more event classes to identify a more concrete effective tradeoff. Our experiments only cover abstract event prototypes, so evaluating multiple-class encoding and

separability on real LZ event types is a logical next step toward proving the practical utility of the diffusion autoencoder approach.

4.2 Future Work

Our work has several implications. Reducing the effective size of LZ or similar experiments by $\sim 500\times$ unlocks more sophisticated analysis that currently is computationally impermissible on consumer hardware. It is an open question in experiments similar to LZ how to best identify event types and detect potential anomalies. Both of these studies are slow and error-prone on full-scale data, where the subset of feasible, mostly algorithmic methods perform rudimentary cuts on the data to determine event class or to pick out rare or anomalous events. A representation that can compress signal at very high data rates with minimal inductive bias has the potential to form extremely feature-rich data manifolds.[10] Compared to VAEs, we need not impose constraints on the latent representation, giving rise to the emergent separability and anomaly detection that we observe in our experimental results. We provide very minimal study into the practical uses of autoencoded particle data, and investigating the breadths of analyses, machine-learning or otherwise, possible on a data which are two-to-five orders of magnitude smaller.

The sparse graph node-space diffusion model is a characteristically slow choice for the decoder. The current configuration requires 250 denoising passes through the model per sample generation. Combined with overhead introduced by sparse operations, on a standard consumer GPU single event generation can take up to a minute. During inference time this isn't typically an issue; if we only care about constructing the encoded representation, we dodge this cost altogether. While this choice was made to conserve the information objective toward log probability maximization of the data distribution, it's entirely possible that more pragmatic approaches exist. While we claim that flow matching models have a less direct optimization and have a malformed inductive bias toward linear interpolation between distributions (particularly for stochastic physics data), we have not evaluated the two models side-by-side to determine if this is the case. If not, reconstruction could see up to a $10\times$ speedup due to more efficient iterative sampling [5].

A middle ground likely exists between node-distributional modeling fidelity in data space and compute efficiency. The current modeling regime only lacks bias in the information-theoretic limit. In practice, the limitations of the sparse GNN introduce unavoidable biases. That is, performing diffusion or flow matching in *some* latent space likely wouldn't massively change the behavior of the model. For instance, one could envision disentangling the latent scale, the diffusion scale,

[10] A concession of not having explicit regularization on the latent space distribution is the correctness of the naïve empirical sampling process during synthetic event generation. See that section on better constraint-free approaches.

and the data scale (apart from their natural is-greater-than relationship). We fix the input representation, choose a latent scale according to the data needs, then choose a diffusion space that is close enough to the data space so as to not introduce excess distortion in the decoding, but small enough to minimize compute costs. In the LZ case, we could set the latent representation on the expected upper bound on physics information content, and the diffusion space size to roughly map to the expected upper bound of stochastic information content, such that each has sufficient information to guide the conditioning and stochastic generation processes.

There are a variety of engineering choices made in the diffusion autoencoder, either to improve efficiency or improve expressivity, that warrant future consideration. One prominent choice was the one to model the TPC time domain as separate node layers, as opposed to feature vectors of TPC-correspondent nodes. It is entirely possible that this was the wrong choice—we made it to avoid building additional machinery to represent the time-correlation behavior of events. With a more refined graph approach, further study into whether this was correct is possible. If so, better dense graph operations become newly permissible (as the memory-limiting adjacency matrix is now six orders of magnitude smaller). In either case, study into sparse graph compression in particular is likely worthy, and a legitimate gap in current generative autoencoder research. Regardless, testing a much broader suite of models is required for certainty on which approach is optimal.

An interesting continuation of this work would be on the generalization and transfer learning capabilities of the DiffAE model. In experiments like LZ, which operate on data streams, designing practical learned compression solutions for data which may drift distributionally takes thought. With the current structure, learned compression schemes are constrained to the set of events already generated, with new data requiring new model versions, but if inference-time distribution matching techniques work for this approach the model could be implemented online.

Appendices

A ELBO Derivation

Here, we provide the full derivation of the evidence lower bound (ELBO) for the DDPM decoder. The reverse denoising process is

$$p_\theta(x_{0:T} \mid z) = p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1} \mid x_t, z),$$

where

$$p_\theta(x_{t-1} \mid x_t, z) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t, z), \Sigma_\theta(x_t, t, z)\right).$$

Thus the conditional reconstruction density is

$$p_\theta(x_0 \mid z) = \int p_\theta(x_{0:T} \mid z)\, dx_{1:T}.$$

As in standard diffusion models, the forward noising process is fixed:

$$q(x_{1:T} \mid x_0) = \prod_{t=1}^{T} q(x_t \mid x_{t-1}) \quad \text{s.t.} \quad q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\right).$$

The conditional log-likelihood is generally intractable because it marginalizes over the denoising trajectory $x_{1:T}$:

$$\begin{aligned}
\log p_\theta(x_0 \mid z) &= \log \int p_\theta(x_{0:T} \mid z)\, dx_{1:T} \\
&= \log \int q(x_{1:T} \mid x_0) \frac{p_\theta(x_{0:T} \mid z)}{q(x_{1:T} \mid x_0)}\, dx_{1:T} \\
&= \log \mathbb{E}_{q(x_{1:T}|x_0)} \left[\frac{p_\theta(x_{0:T} \mid z)}{q(x_{1:T} \mid x_0)}\right] \\
&\geq \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p_\theta(x_{0:T} \mid z)}{q(x_{1:T} \mid x_0)}\right],
\end{aligned}$$

where the final inequality follows from Jensen's inequality. We can define the conditional diffusion ELBO as this bound

$$\mathcal{E}_{\text{DiffAE}}(x_0, z) = \mathbb{E}_{q(x_{1:T}|x_0)} \left[\log \frac{p_\theta(x_{0:T} \mid z)}{q(x_{1:T} \mid x_0)}\right].$$

so

$$\mathcal{E}_{\text{DiffAE}}(x_0, z) \leq \log p_\theta(x_0 \mid z).$$

Substituting the Markov factorizations gives

$$\mathcal{E}_{\text{DiffAE}}(x_0, z) = \mathbb{E}_q \left[\log p(x_T) + \sum_{t=1}^{T} \log p_\theta(x_{t-1} \mid x_t, z) - \sum_{t=1}^{T} \log q(x_t \mid x_{t-1})\right]. \tag{11}$$

Rearranging this expression yields the usual diffusion decomposition, now condi-

tioned on z:

$$\begin{aligned}-\mathcal{E}_{\mathrm{DiffAE}}(x_0, z) =& D_{\mathrm{KL}}\left(q(x_T \mid x_0) \,\|\, p(x_T)\right) \\ &+ \sum_{t=2}^{T} \mathbb{E}_{q(x_t|x_0)}\left[D_{\mathrm{KL}}\left(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t, z)\right)\right] \\ &- \mathbb{E}_{q(x_1|x_0)}\left[\log p_\theta(x_0 \mid x_1, z)\right]. \end{aligned} \tag{12}$$

The first term is fixed by the noising process and terminal prior. The middle terms are the learned reverse-process matching terms. The final term is the decoder likelihood at the last denoising step.

The forward posterior has the closed form

$$q(x_{t-1} \mid x_t, x_0) = \mathcal{N}\left(x_{t-1}; \tilde{\mu}_t(x_t, x_0), \tilde{\beta}_t I\right),$$

while the learned conditional reverse transition is

$$p_\theta(x_{t-1} \mid x_t, z) = \mathcal{N}\left(x_{t-1}; \mu_\theta(x_t, t, z), \sigma_t^2 I\right).$$

Therefore, each transition-level KL is a Gaussian-to-Gaussian KL:

$$D_{\mathrm{KL}}\left(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t, z)\right).$$

When the variances are fixed or otherwise independent of the mean prediction, the θ-dependent part reduces to a weighted squared error between reverse-process means:

$$D_{\mathrm{KL}}\left(q(x_{t-1} \mid x_t, x_0) \,\|\, p_\theta(x_{t-1} \mid x_t, z)\right) = c_t \left\|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t, z)\right\|^2 + \text{const},$$

for a timestep-dependent weight c_t.

Using the standard noising reparameterization,

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \qquad \epsilon \sim \mathcal{N}(0, I),$$

the learned reverse mean can be parameterized through a noise predictor

$$\epsilon_\theta(x_t, t, z).$$

Under this parameterization, the mean-matching term is equivalent, up to timestep-dependent weighting, to noise prediction:

$$\left\|\tilde{\mu}_t(x_t, x_0) - \mu_\theta(x_t, t, z)\right\|^2 = k_t \left\|\epsilon - \epsilon_\theta(x_t, t, z)\right\|^2,$$

for a timestep-dependent weight k_t.

Thus, up to constants and timestep weights, the negative conditional diffusion ELBO is implemented by the denoising objective

$$\begin{aligned} \mathcal{L}_{\text{DiffAE}} &= \mathbb{E}_{x_0,t,\epsilon}\left[w_t \left\|\epsilon - \epsilon_\theta(x_t, t, z)\right\|^2\right] \\ \implies \min_{\theta,\phi} \mathcal{L}_{\text{DiffAE}} &\approx \max_{\theta,\phi} \mathbb{E}_{p_{\text{data}}(x_0)}\left[\mathcal{E}_{\text{DiffAE}}\big(x_0, z\big)\right]. \end{aligned}$$

B Additional Figures

Below is the correlation of each RQ in the original and reconstructed events for the autoencoder (AE, left) and diffusion autoencoder (DiffAE, right):

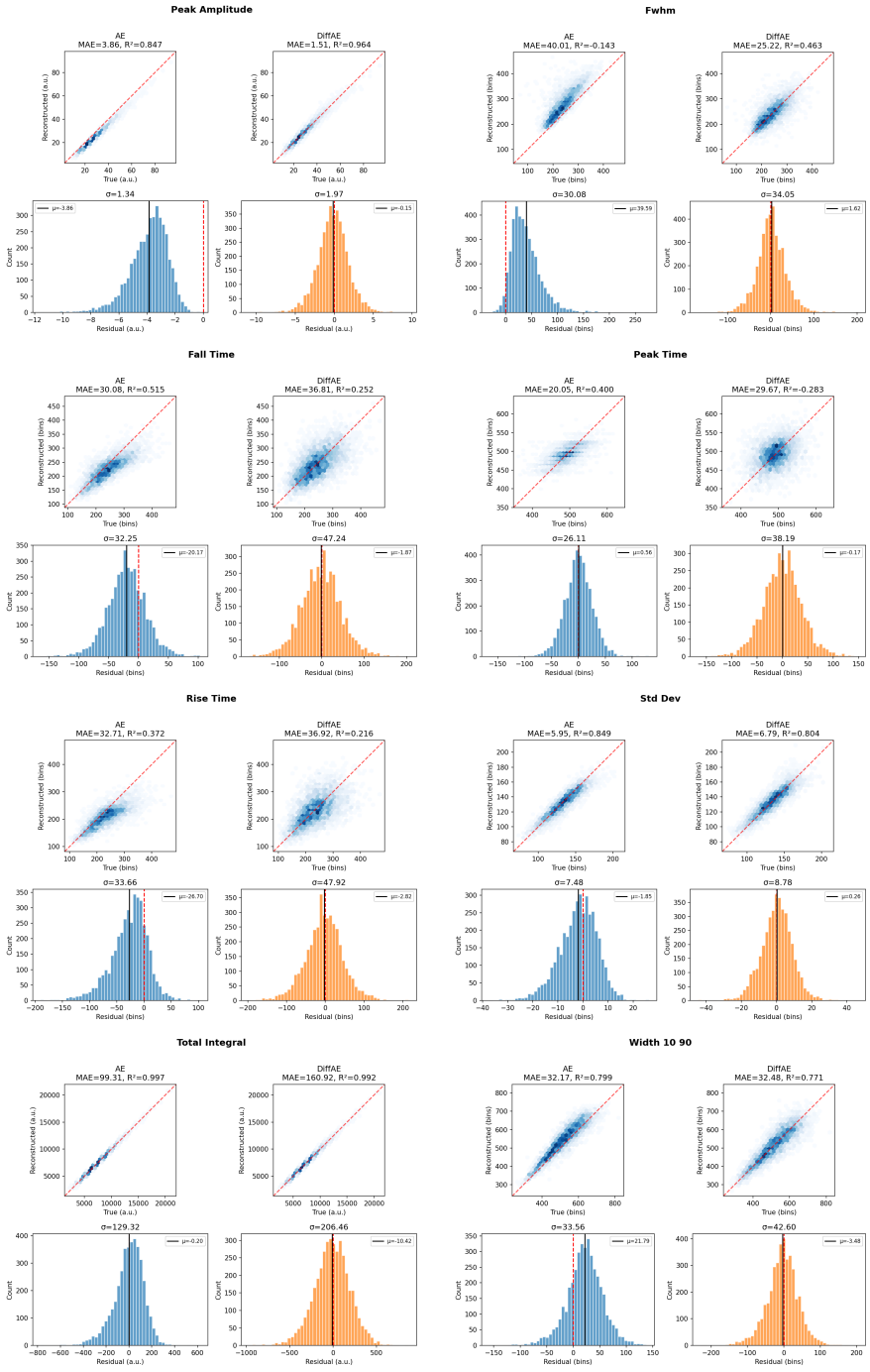

Figure 25: Correlation and error distribution of all RQs. The AE events show strong systematic biases across RQs, which are not evident in DDPM-generated events.

References

[1] M. A. Kramer, *Autoassociative neural networks*, *Computers & Chemical Engineering*, vol. 16, no. 4, pp. 313–328, 1992.

[2] J. Ballé, V. Laparra, and E. P. Simoncelli, *End-to-end optimized image compression*, *International Conference on Learning Representations (ICLR)*, 2017. https://arxiv.org/abs/1611.01704

[3] D. P. Kingma and M. Welling, *Auto-encoding variational Bayes*, *International Conference on Learning Representations (ICLR)*, 2014. https://arxiv.org/abs/1312.6114

[4] J. Ho, A. Jain, and P. Abbeel, *Denoising diffusion probabilistic models*, *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, pp. 6840–6851, 2020. https://arxiv.org/abs/2006.11239

[5] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, *Flow matching for generative modeling*, *International Conference on Learning Representations (ICLR)*, 2023. https://arxiv.org/abs/2210.02747

[6] K. Preechakul, N. Chatthee, S. Wizadwongsa, and S. Suwajanakorn, *Diffusion autoencoders: Toward a meaningful and decodable representation*, *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10619–10629, 2022. https://arxiv.org/abs/2111.15640

[7] A. Nichol and P. Dhariwal, *Improved denoising diffusion probabilistic models*, *International Conference on Machine Learning (ICML)*, 2021. https://arxiv.org/abs/2102.09672

[8] M. Arjovsky, S. Chintala, and L. Bottou, *Wasserstein generative adversarial networks*, *International Conference on Machine Learning (ICML)*, pp. 214–223, 2017. https://arxiv.org/abs/1701.07875

[9] P. Dhariwal and A. Nichol, *Diffusion models beat GANs on image synthesis*, *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, pp. 8780–8794, 2021. https://arxiv.org/abs/2105.05233

[10] LZ Collaboration, *LUX-ZEPLIN conceptual design report*, Fermilab-TM-2621-AE-E-PPD, 2015. https://lss.fnal.gov/archive/test-tm/2000/fermilab-tm-2621-ae-e-ppd.pdf

[11] LZ Collaboration, *Summary of data management principles: LZ experiment*, 2021. https://lz.lbl.gov/wp-content/uploads/sites/6/2021/09/LZ-DMP-Operations.pdf

[12] LZ Collaboration, *The data acquisition system of the LZ dark matter detector: FADR*, *Nuclear Instruments and Methods in Physics Research A*, 2024. https://arxiv.org/abs/2405.14732

[13] Y. Qie et al. (LZ Collaboration), *Application of a novel method to characterize the performance of zero suppression in the LUX-ZEPLIN experiment*, *APS Meeting Abstracts*, 2023.

[14] LZ Collaboration, *First dark matter search results from the LUX-ZEPLIN (LZ) experiment*, *Physical Review Letters*, vol. 131, p. 041002, 2023.

[15] D. S. Akerib et al. (LZ Collaboration), *The LUX-ZEPLIN (LZ) experiment*, *Nuclear Instruments and Methods in Physics Research A*, vol. 953, p. 163047, 2020.

[16] I. Li, A. Higuera, S. Liang, J. Qin, and C. Tunnell, *Energy reconstruction with semi-supervised autoencoders for dual-phase time projection chambers*, *EPJ Web of Conferences*, vol. 295, p. 09022, 2024. https://doi.org/10.1051/epjconf/202429509022

[17] Z. Fu et al., *Generative models for simulation of KamLAND-Zen*, *European Physical Journal C*, vol. 84, 2024. https://arxiv.org/abs/2312.14372

[18] Y. Huang, Y. Ren, S. Yoo, and J. Huang, *Efficient data compression for 3D sparse TPC via bicephalous convolutional autoencoder*, arXiv:2111.05423, 2021. https://arxiv.org/abs/2111.05423

[19] M. Zhang et al., *A survey on graph diffusion models: Generative AI in science for molecule, protein and material*, arXiv:2304.01565, 2023. https://arxiv.org/abs/2304.01565

[20] C. Vignac et al., *DiGress: Discrete denoising diffusion for graph generation*, *International Conference on Learning Representations (ICLR)*, 2023. https://arxiv.org/abs/2209.14734

[21] X. Hou et al., *Improving molecular graph generation with flow matching and optimal transport*, arXiv:2411.05676, 2024. https://arxiv.org/abs/2411.05676

[22] K. Yi et al., *Graph denoising diffusion for inverse protein folding*, *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 36, 2024. https://arxiv.org/abs/2306.16819

[23] K. E. Wu et al., *Protein structure generation via folding diffusion*, *Nature Communications*, vol. 15, p. 1059, 2024. https://arxiv.org/abs/2209.15611

[24] C. Zeni et al., *MatterGen: A generative model for inorganic materials design*, *Nature*, 2025. https://arxiv.org/abs/2312.03687

[25] Y. B. Uslu et al., *Graph Signal Generative Diffusion Models*, *arXiv preprint arXiv:2509.17250*, 2025. https://arxiv.org/abs/2509.17250

[26] S. Rozada et al., *Graph-Aware Diffusion for Signal Generation*, *arXiv preprint arXiv:2510.05036*, 2025. https://arxiv.org/abs/2510.05036

[27] D. Wesego, *Graph Representation Learning with Diffusion Generative Models*, *arXiv preprint arXiv:2501.13133*, 2025. https://arxiv.org/abs/2501.13133

[28] Y. Huang et al., *Variable Rate Neural Compression for Sparse Detector Data*, *arXiv preprint arXiv:2411.11942*, 2024. https://arxiv.org/abs/2411.11942

[29] W. Bhimji, C. Harris, V. Mikuni, and B. Nachman, *Foundation model framework for all tasks involving jet physics*, *Physical Review D*, vol. 113, no. 3, 2026. https://arxiv.org/abs/2510.24066

[30] T. Salimans and J. Ho, *Progressive distillation for fast sampling of diffusion models*, *International Conference on Learning Representations (ICLR)*, 2022. https://arxiv.org/abs/2202.00512

[31] H. Shirzad et al., *Exphormer: Sparse transformers for graphs*, *International Conference on Machine Learning (ICML)*, 2023. https://arxiv.org/abs/2303.06147

[32] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, *How powerful are graph neural networks?*, *International Conference on Learning Representations (ICLR)*, 2019. https://arxiv.org/abs/1810.00826

[33] G. Corso et al., *Principal neighbourhood aggregation for graph nets*, *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020. https://arxiv.org/abs/2004.05718

[34] J. Lee, I. Lee, and J. Kang, *Self-attention graph pooling*, *International Conference on Machine Learning (ICML)*, 2019. https://arxiv.org/abs/1904.08082

[35] T. N. Kipf and M. Welling, *Semi-supervised classification with graph convolutional networks*, *International Conference on Learning Representations (ICLR)*, 2017. https://arxiv.org/abs/1609.02907

[36] Z. Ying et al., *Hierarchical graph representation learning with differentiable pooling*, *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 31, 2018. https://arxiv.org/abs/1806.08804

[37] I. S. Dhillon, Y. Guan, and B. Kulis, *Weighted graph cuts without eigenvectors: A multilevel approach*, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 11, pp. 1944–1957, 2007.

[38] M. Brockschmidt, *GNN-FiLM: Graph neural networks with feature-wise linear modulation*, *International Conference on Machine Learning (ICML)*, 2020. https://arxiv.org/abs/1906.12192

[39] V. Chandola, A. Banerjee, and V. Kumar, *Anomaly detection: A survey*, *ACM Computing Surveys*, vol. 41, no. 3, pp. 1–58, 2009.

[40] V. Belis, P. Odagiu, and T. K. Aarrestad, *Machine learning for anomaly detection in particle physics*, *Reviews in Physics*, vol. 12, p. 100091, 2024. https://arxiv.org/abs/2312.14190