Generative Distribution Embeddings

Nic Fishman[1,†], **Gokul Gowri**[2,†], **Peng Yin**[3], **Jonathan Gootenberg**[4,5,6,*], **and Omar Abudayyeh**[4,5,6,*]

[1]Department of Statistics, Harvard University; [2]Laboratory for Information and Decision Systems, MIT;
[3]Wyss Institute for Biologically Inspired Engineering and Department of Systems Biology, Harvard University;
[4]Center for Virology and Vaccine Research, Beth Israel Deaconess Medical Center, Harvard Medical School;
[5]Dept. of Medicine, Div. of Engineering in Medicine, Brigham and Women's Hospital, Harvard Medical School;
[6]Gene and Cell Therapy Institute, Mass General Brigham
[†]*Equal contribution*: njwfish@gmail.com, gokulg@mit.edu.
[*]*Senior authors jointly supervised this work*: jgootenb@bidmc.harvard.edu, omar@abudayyeh.science.

Abstract

Many real-world problems require reasoning across multiple scales, demanding models which operate not on single data points, but on entire distributions. We introduce generative distribution embeddings (GDE), a framework that lifts autoencoders to the space of distributions. In GDEs, an encoder acts on *sets* of samples, and the decoder is replaced by a generator which aims to match the input distribution. This framework enables learning representations of distributions by coupling conditional generative models with encoder networks which satisfy a criterion we call distributional invariance. We show that GDEs learn predictive sufficient statistics embedded in the Wasserstein space, such that latent GDE distances approximately recover the W_2 distance, and latent interpolation approximately recovers optimal transport trajectories for Gaussian and Gaussian mixture distributions. We systematically benchmark GDEs against existing approaches on synthetic datasets, demonstrating consistently stronger performance. We then apply GDEs to six key problems in computational biology: learning donor-level representations from single-nuclei RNA sequencing data (6M cells), capturing clonal dynamics in lineage-traced RNA sequencing data (150K cells), predicting perturbation effects on transcriptomes (1M cells), predicting perturbation effects on cellular phenotypes (20M single-cell images), designing synthetic yeast promoters (34M sequences), and spatiotemporal modeling of viral protein sequences (1M sequences).

1 Introduction

Advancements in science and engineering increasingly depend on our ability to reason across multiple scales: modeling not just individual data points, but entire *populations* those datapoints are drawn from. In applications ranging from single-cell genomics to DNA sequence design, the relevant unit of analysis is not an individual sample (e.g., a single cell), but the distribution from which it is drawn (e.g. the cell state or the patient they were sampled from). These settings are fundamentally *hierarchical*: we observe sets of samples from latent distributions, which themselves are drawn from a meta-distribution. Without directly modeling these distributions, population-level signals can be lost underneath unit-level noise. The fundamental challenge we consider is how to learn representations at the level of distributions, not just individual data points.

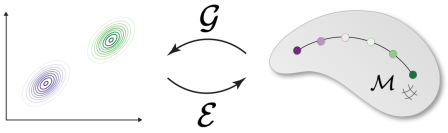

Figure 1: GDEs leverage distribution-invariant encoders ($\mathcal{E}$) and conditional generative models ($\mathcal{G}$) to lift autoencoders to statistical manifolds where points correspond to distributions ($\mathcal{M}$).

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

We introduce *generative distribution embeddings* (GDEs) (Fig. 1), a framework that lifts autoencoders to the distribution space. In GDEs, the encoder maps a finite set of samples – an empirical distribution – to a latent space, while the decoder is replaced by a generative model that reconstructs the distribution by sampling conditional on this latent representation. Our central observation is that strong distributional representations can be learned by coupling conditional generative models with encoders that satisfy a minimal *distributional invariance* property, which we motivate through a statistical decision-theoretic lens.

Our framework synthesizes ideas from conditional generation, empirical process theory, and information geometry. We show empirically that GDEs behave as approximate *predictive sufficient statistics* [1, 2], capturing distribution-level structure while marginalizing over sampling noise. Moreover, the learned latent spaces exhibit geometric regularity: latent L_2 distances correlate with Wasserstein distances (W_2) between underlying distributions, and linear interpolation in latent space approximates optimal transport geodesics [3], such that one can generate synthetic data which smoothly interpolate between observed distributions.

We benchmark GDEs on synthetic datasets with known parametric structure, demonstrating improved generative fidelity and structure preservation relative to baselines. We then scale our approach to multiple domains in computational biology, showcasing GDEs' versatility in modeling distributions defined across diverse organizing principles such as distinct populations, varying experimental conditions, spatial arrangements, and temporal dynamics. We demonstrate six applications: learning donor-level representations from single-nuclei RNA sequencing data (6M cells), capturing clonal dynamics in lineage-traced RNA sequencing data (150K cells), predicting perturbation effects on transcriptomes (1M cells), predicting perturbation effects on cellular phenotypes (20M single-cell images), designing synthetic yeast promoters (34M sequences), and spatiotemporal modeling of viral protein sequences (1M sequences). Across these domains, GDEs offer a flexible and scalable framework for distribution-level inference. Code for all experiments is available here.

2 Setting and methods

2.1 A motivating example

Many modern datasets comprise large groups of *exchangeable* samples: for example, single cells or DNA sequences collected from individual patients [4, 5]. We observe n such groups, each written as $S_{i,m} = \{x_{ij}\}_{j=1}^{m}$, and collect them as $\mathcal{D} = \{S_{i,m}\}_{i=1}^{n}$ with $x_{ij} \in \mathcal{X}$. Each group's samples are i.i.d. draws from a latent, group-specific distribution $P_i \in \mathcal{P}(\mathcal{X})$, and the P_i themselves are i.i.d. draws from a meta-distribution Q; that is, we first sample $P_i \sim Q$ and then, conditional on P_i, draw $x_{ij} \sim P_i$. In this framework, a "patient" (or more generally, any group) is implicitly represented by its probability measure P_i.

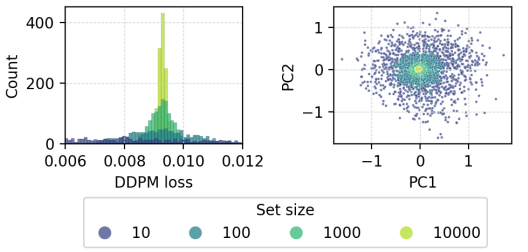

Figure 2: *Concentration of distribution embeddings and plug-in loss. (Left)* Distribution of plug-in GDE loss (diffusion generator) for sets of different sizes sampled from the same distribution P_i over MNIST digits. *(Right)* First two principal components of embeddings of sample sets of different sizes generated by the same P_i.

This is a classical hierarchical data generating process, which gives rise to a multiscale problem. The subject of interest here is not only individual cells or DNA sequences x_{ij}, but the probability measures P_i. As we explore in Sec. 3 and concretely demonstrate in Sec. 6, this setting is broadly applicable beyond the particular case of representing patients as a collection of samples.

In practice, there are often two major challenges in modeling this kind of data. First, unit-level data is often inherently noisy. For example, single-cell data suffers from noise due to molecular undersampling [6]. Our goal is to learn patient embeddings which capture distribution-level signal, rather than the sample-level noise. The second challenge is that groups can contain millions of samples (in the case of DNA sequencing reads per patient, m can be $\sim 10^8$). It is computationally infeasible to train a model on all samples simultaneously, but given the inherent noise at the unit level we would benefit from embedding all available samples at inference time.

In the remainder of Sec. 2, we will show that both of these practical challenges can be overcome by learning distribution embeddings rather than simply encoding sets of samples. The distributional

perspective enables models which distill distribution-level signal, and are able to massively scale at inference time to make use of all available data for precise embeddings.

2.2 Learning generative distribution embeddings

We address this problem with GDEs, which consist of an encoder $\mathcal{E}$ that maps a finite set of samples $S_{i,m}$ to a latent representation, and a conditional generator $\mathcal{G}$ that maps the latent representation back to the sample space. Formally, we aim to learn $\mathcal{E}, \mathcal{G}$ such that

$$\mathcal{G}\big(\mathcal{E}(S_{i,m})\big) \xrightarrow{D} P_i \text{ as } m \to \infty \tag{1}$$

Algorithm 1 Training GDEs

for each set $S_{i,m}$ **do**
  $z_i \leftarrow \mathcal{E}(S_{i,m})$
  loss $\leftarrow \ell(S_{i,m}, \mathcal{G}(z_i))$
  Backpropagate through $\mathcal{E}$ and $\mathcal{G}$
end for

The loss ℓ is the standard training objective for the conditional generator (for example, an evidence lower bound for a VAE or a denoising score-matching objective for a diffusion model); we do not need to backpropagate through the sampling process of $\mathcal{G}$.

We show that achieving Eq. (1) is only possible if the encoder satisfies the following two constraints:

1. Permutation invariance: reordering the samples in $S_{i,m}$ does not change the embedding.
2. Proportional invariance: duplicating every sample K times does not change the embedding.

We refer to an encoder with these properties as *distributionally invariant*: the encoder must depend only on the empirical distribution. So for some function ϕ we can write:

$$P_{i,m} = \frac{1}{m}\sum_{j=1}^{m} \delta_{x_{ij}}, \qquad \mathcal{E}(S_{i,m}) = \phi(P_{i,m}).$$

We show formally that distributionally invariant encoders can capture any distributional property and furthermore that any non-distributionally invariant architecture can spuriously encode noise features irrelevant to the distribution (formal proofs in App. D.2):

Proposition 1. ***(Informal statement of Corollary 1)*** *Any class of encoder-generator architectures that consistently recovers the data-generating distribution from i.i.d. samples must encode exactly the information in the empirical distribution—no more, no less.*

Beyond separating signal and noise, distributional invariance, coupled with Hadamard differentiability of the pooling operator, has a second crucial consequence: it enables a *central limit theorem for the embeddings*. As the set size grows, $\mathcal{E}(S_{i,m})$ concentrates around its population value with Gaussian fluctuations:

$$\sqrt{m}, \big(\mathcal{E}(S_{i,m}) - \phi(P_{i,m})\big) \xrightarrow{d} \mathcal{N}(0, \Sigma)$$

This result, illustrated empirically in Fig. 2, is what makes encoding massive sets possible. The CLT composes through the plug-in loss so that $\ell(S_{i,m}, \mathcal{G}(\mathcal{E}(S_{i,m})))$ is an unbiased, asymptotically normal estimator of the population loss. This provides theoretical justification for training GDEs on subsets of larger sample sets: the gradient of the loss computed on small sets matches (in expectation) the gradient computed using all samples per set (see App. D.2):

Proposition 2. ***(Informal statement of Theorem 2)*** *Fixing P, under mild regularity conditions: (i) a distributionally invariant encoder will have asymptotically normal distribution embeddings; (ii) for a suitable divergence, the plug-in loss, $\widehat{\ell}_m := \mathfrak{d}(P, \mathcal{G}(\mathcal{E}(P_m)))$, is asymptotically unbiased and normally distributed around the population loss; (iii) a global minimizer will recover the true data distribution as $m \to \infty$: $\mathcal{G}(\mathcal{E}(P_m)) \Rightarrow P$. See Fig. 2.*

Violating distributional invariance (for example, by using sum pooling) causes the embedding to depend on set size and breaks this limit theory causing Eq. (1) to fail. In contrast, mean pooling and M/Z-estimators satisfy these properties (see App. D.2).

3 From labels to distributions

In the previous section, we focused on a simple motivating example where we have patients and their associated single-cell data. In that setting, the multiscale nature is clear and it is straightforward to define a metadistribution (in other words, how to group samples into sets). In many datasets, a natural hierarchy is not as clear: for example, MNIST digits and labels are not multiscale in the same sense as our first motivating example.

Here we illustrate how to set up the distribution learning problem in a more general framework based on a dataset of unit-level outcomes associated with labels $D = \{(x_k, y_k)\}_{k=1}^N$ rather than sets drawn from Q. We will show how to group data points into sets $\{x_{ij}\}_{j=1}^m$ whose empirical distributions $P_{i,m}$ approximate draws from Q. The grouping reflects the structure of the label space $\mathcal{Y}$ and enables us to shape the GDE latent space for downstream applications (see Sec. 5).

When $\mathcal{Y}$ is discrete, we can form sets by grouping datapoints with the same label (e.g., our motivating patient example further explored in Sec. 6.2, cells by clone identity in Sec. 6.3, cell transcriptomes by perturbation in Sec. 6.4, or epigenetic samples by tissue in Sec. B.1). If there is some semantic similarity between discrete labels we can define sets proportional to those similarity metrics, paralleling contrastive learning, where labels define semantic neighborhoods.

When $\mathcal{Y}$ is continuous or structured (e.g. spatial coordinates for the x_{ij} as in Sec. 6.5 or temporal in the viral protein sequences in Sec. 6.7), we can use a similarity kernel to sample points near a target y_i^*: $w_{ik} = \exp(-d(y_k, y_i^*)^2/(2\sigma^2))$, defining a probabilistic neighborhood in the label space, enforcing the consideration of the local structure.

When labels are noisy measurements (e.g. expression associated with DNA sequences as in Sec. 6.6), we can invert the noise model $y_k = y_k^* + \epsilon_k$ by sampling a latent target y_i^* and computing likelihood weights $w_{ik} = p(y_k \mid y_i^*)$, yielding samples that reflect the uncertainty of the data.

All these constructions can be unified as instances of a general framework: let $Q^{(\mathcal{Y})}$ be a prior over label distributions. For each set i, we draw $P_i^{(Y)} \sim Q^{(\mathcal{Y})}$ and compute weights

$$w_{ik} = \frac{dP_i^{(Y)}}{d\hat{P}_{\text{emp}}}(y_k), \quad \hat{P}_{\text{emp}}(y) = \frac{1}{N}\sum_{k=1}^{N} \delta_{y_k}(y),$$

and sample x_{ij} from D accordingly. This framework subsumes the above examples and giving us a general set of tools for shaping the GDE latent space, as we will illustrate in Sec. 6.

4 Related work

Several lines of literature have tried to learn distribution embeddings or summary statistics. Kernel methods, such as kernel mean embedding (KME) and set kernels, provide nonparametric approaches to represent probability measures as points in a reproducing kernel Hilbert space, enabling tasks like distributional regression and classification [7, 8, 9, 10, 11]. GDEs naturally nest these methods as particular choices of distributionally invariant encoders. GDEs also generalize the approach in [12], where they develop a particular encoder and VAE-based generator.

Distribution embeddings have also been studied from a geometric perspective. Building on theoretical foundations from Amari [13], several works model distributions as points on a manifold imbued with the Fisher-Rao metric [14, 15, 16, 17]. These methods are either not generative or restricted to categorical distributions. Building on the work of Otto [3], others have considered learning flows over Wasserstein spaces [18, 19] (see Appendix C.2 for background on Wasserstein spaces), primarily focused on leveraging distribution encodings for transport problems as opposed to GDEs which aim to auto-encode distributions. GDEs are complementary to these works, and can be plugged in to many of these frameworks. One recent method closely related to GDEs, Wasserstein Wormhole [20], aims to represent distributions as points in a space where Euclidean distances match Sinkhorn divergences in the sample space. Wasserstein Wormhole is a particular instantiation of a GDE, using an attention-based encoder and generator that only samples a fixed number of points.

A related body of work aims to learn informative summary statistics [21, 22, 23, 24, 25]. These methods typically consider a supervised setting with a particular inferential target. For example, in the context of likelihood-free inference, one aims to learn summary statistics which are maximally informative about the parameters of a generative model [23, 24].

GDEs are distinct from these approaches along several dimensions: first, we generalize these methods under a common framework with a central objective of re-sampling the encoded distribution (1); second, we develop theory to guide the design and analysis of GDEs toward this objective; third, we show that distribution embedding is deeply related to generative modeling, enabling domain-specific generative models to be bootstrapped into high-quality GDEs to tackle multiscale problems.

On the architectural side, the encoder in the GDE framework requires a distributionally invariant model. While distributional invariance is a concept introduced in this work, it requires permutation invariance, which has been well-studied [26, 27, 28]. Some permutation invariant approaches, such as deep sets [26], are not distributionally invariant due to proportional sensitivity, while others, such as mean-pooled attention layers, are also distributionally invariant (as shown in Appendix D.2).

A key contribution of our work is the observation that any conditional generative model can be repurposed to learn distributional representations. Recent work in the vision domain has found that conditional diffusion models can induce strong image representations [29]. Our work formalizes and generalizes this finding. We demonstrate in practice that a number of modern techniques, including variational autoencoders [30], Sinkhorn-based generative models [31], sliced Wasserstein models [32], denoising diffusion models [33], and autoregressive sequence models [34, 35], can be leveraged to learn GDEs. This is by no means exhaustive: any other conditional generative modelling approach [36, 37], including those which will emerge in the future, can be used in the GDE framework.

5 Statistical and geometric properties GDEs

GDEs aim to learn representations that separate the structure of the data-generating distribution from finite-sample noise, and to synthesize new data consistent with that structure. We formalize this dual role through two complementary perspectives. First, as *predictive sufficient statistics*, GDEs act like learned Rao–Blackwellizations that denoise sampling variability. Second, as *statistical manifold embeddings*, they interpolate between distributions along smooth geometric paths.

5.1 Learning an approximate predictive sufficient statistic

The core objective of GDEs is to recover the true data-generating distribution P from finite samples by learning an aggregate representation that distills the structure of P from sampling noise. This objective is captured by the notion of *asymptotic predictive sufficiency* [1, 2], which reformulates sufficiency in terms of conditional independence:

$$\mathbb{P}(x_{\text{new}} \in A \mid T(S_m)) \approx \mathbb{P}(x_{\text{new}} \in A \mid S_m), \quad m \to \infty,$$

for all measurable $A \subseteq \mathcal{X}$. In our setting, the encoder $\mathcal{E}(S_m)$ serves as such a statistic, asymptotically determining P and marginalizing over sampling variability in S_m.

Predictive sufficiency implies a *Rao–Blackwell* improvement principle: conditioning any predictor on a predictively sufficient statistic cannot increase predictive risk under convex loss [2]. To illustrate this empirically, we consider $X_i \sim \text{Pois}(\lambda)$ and predict $\mathbb{P}(X_{n+1} = 0)$. The baseline uses the observed frequency of $X_i = 0$, the GDE estimator conditions on the embedding and draws 10^6 synthetic samples upon which the baseline estimator is applied, and the Rao–Blackwellized (RB) estimator conditions on the sufficient statistic $T = \sum_i X_i$. In Tab. 1 we can see that the GDE estimator outperforms the RB bound on the data manifold at low sample numbers. This suggests GDEs can act as data-driven analogues of Rao–Blackwellization, using synthetic sampling to magnify signal.

n	Naive	RB	GDE
10	4.72e-03	3.79e-03	3.12e-03
100	3.41e-04	2.76e-04	2.71e-04
1000	3.03e-05	2.81e-05	2.67e-05
10000	3.23e-06	2.64e-06	3.32e-06

Table 1: MSE of Naive, RB, and GDE estimators for $P(X = 0)$, $X \sim \text{Poi}(\lambda)$.

A predictive sufficient statistic distills the structural properties of the meta-distribution while marginalizing over sampling variability in the observed data. Generative distribution embeddings achieve this in practice: they recover consistent representations of underlying distributions, even across diverse domains and observational sample spaces.

We demonstrate this using the multinomial distribution. We learn GDEs of 3-dimensional multinomial distributions using a mean-pooled deep sets encoder and a diffusion generator. The model's latent space is able to recover the structure of the multinomial simplex (Fig. 3). Next, we use two real-world

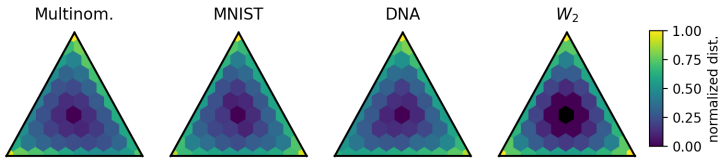

Figure 3: *L_2 in GDE latent space compared to W_2 distance.* Normalized distances from the center, $p = (1/3, 1/3, 1/3)$. The plots to the left show GDE L_2 learned from empirical distributions. MNIST and DNA distributions are constructed by sampling conditional on class label according to a multinomial, for MNIST subsetted to images of (0, 1, 2) and a synthetic DNA dataset with 3 patterns respectively. Rightmost plot shows the Gaussian approximation for the W_2 between multinomials.

datasets with discrete class labels and conditionally sample observations according to label identities, which are drawn from the same family of 3-dimensional multinomial distributions. For both a three-digit subset of MNIST and a set of three synthetic DNA sequence patterns, GDEs (using 2D and 1D convolutional encoders and diffusion and HyenaDNA generators, respectively) recover the same structure of the underlying multinomial simplex in the latent space. Despite coming from three different domains and using three vastly different architectures, the latent geometry learned between these experiments is nearly identical demonstrating GDEs capacity to learn signal from noise.

In fact, the learned geometry is rather particular: the L_2 distance in GDE latent spaces in all three cases closely resemble W_2 distances between multinomials (computed under a Gaussian approximation). This points to a geometric interpretation of GDEs, bringing us to our second theoretical perspective.

5.2 Learning a statistical manifold

From the geometric perspective, GDEs go beyond summarizing distributions: they can *generate synthetic distributions* that interpolate smoothly between observed populations. Empirically, latent trajectories in GDE space correspond to families of synthetic data that move coherently through probability space, providing a controllable mechanism for exploring or augmenting realistic generative scenarios.

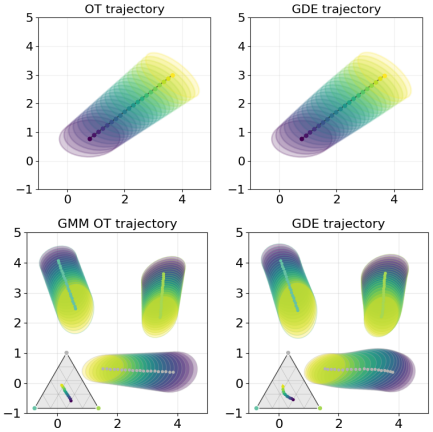

Figure 4: *Top row:* Trajectories between pairs of Gaussians under optimal transport (left) and GDE (right). *Bottom row:* Similar comparison for Gaussian mixture models, we compute the "OT" by finding the optimal pairing between Gaussians and computing the OT. Inset ternary plots show mixture weights during interpolation.

Formally, let $\mathcal{X}$ denote the sample space and $\mathcal{P}_2(\mathcal{X})$ the set of Borel probability measures with finite second moment. The 2-Wasserstein distance W_2 defines the minimal transport cost between distributions and endows $\mathcal{P}_2(\mathcal{X})$ with a Riemannian structure: straight-line transport maps correspond to geodesic interpolants that trace optimal mass transfers between distributions [3]. For a meta-distribution Q over $\mathcal{P}_2(\mathcal{X})$, we define the statistical manifold $\mathcal{M} := \mathrm{supp}(Q)$, a submanifold of $\mathcal{P}_2(\mathcal{X})$ whose geometry is inherited from the Wasserstein metric. Geodesics within $\mathcal{M}$ then correspond to constrained optimal transport paths that remain within the family of data-generating distributions defined by Q.

The encoder $\mathcal{E}$ can be viewed as an (approximate) smooth embedding $\psi : \mathcal{M} \to \mathbb{R}^d$, while the generator parameterizes the inverse map that decodes latent trajectories into synthetic distributions along $\mathcal{M}$. Ideally, ψ would preserve the manifold's intrinsic geometry, as an *approximate isometry* that maps Wasserstein distances between distributions to Euclidean distances in latent space. In practice, such preservation is not guaranteed, yet GDEs empirically behave as near-isometric embeddings: linear latent interpolations correspond to smooth, geometrically consistent synthetic distributions. When trained on samples from 2D Gaussian distributions, latent-space L_2 distances correlate with true W_2 distances at $\rho = 0.96$. For 3-component Gaussian mixtures and W_2 distances restricted to the mixture family [38], the correlation remains high ($\rho = 0.76$). In both cases, linear latent

interpolants decode into realistic synthetic distributions along the optimal transport path (Fig. 4). These results indicate that GDEs learn embeddings that are approximately both *predictively sufficient* in the sense of Section 5.1. and *geometrically meaningful*.

Proposition 3. ***(Informal statement of Theorem 3)*** *A smooth map $\psi : \mathcal{M} \to \mathbb{R}^d$ is an* asymptotically predictively sufficient statistic *if and only if it is a* smooth embedding.

Classically, sufficient statistics and statistical manifolds are fixed once the model family is specified, and the geometry is independent of the likelihood of observing a particular distribution. In contrast, our heirarchical model involes a metadistributional prior, endowing the setting with a Bayesian flavor. GDEs predictive sufficiency is therefore evaluated *with respect to* Q: favoring statistics that preserve predictive information for distributions more common under Q. As a result, the learned representation and the synthetic data generated from it become Q*-weighted*, allocating resolution to regions of $\mathcal{M}$ according to their probability. This adaptive weighting explains why GDEs can outperform the RB estimator in Tab. 1.

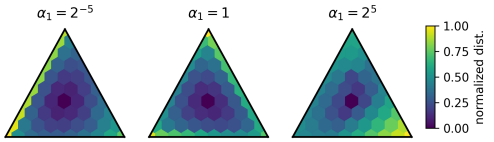

Figure 5: Similar to Fig. 3 we show the GDE distances of multinomials from $p = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. We shift the prior asymmetrically by changing α_1 while fixing $\alpha_2 = \alpha_3 = 1$. This shifts the focus of the model, leading to a different learned geometry.

In earlier examples Q was approximately uniform over a region of $\mathcal{M}$, yielding a learned geometry close to $\mathcal{P}_2(\mathcal{X})$. When Q is non-uniform, the geometry warps: high-density regions expand to preserve finer distinctions, while low-density regions contract. We demonstrate this in our synthetic multinomial setting by training GDEs on empirical distributions sampled from skewed Dirichlet task distributions, $\alpha = (2^{-5}, 1, 1)$ and $\alpha = (2^5, 1, 1)$, compared to the uniform $\alpha = (1, 1, 1)$ (Fig. 5). As expected, distances stretch precisely where Q concentrates.

The takeaway is operational: the embedding is not only geometrically faithful but also prior-weighted. By choosing Q strategically (e.g., via the task-informed sampling in Sec. 3), we can bias the model toward capturing distributional properties most relevant to downstream objectives.

6 Applications

We first benchmark our approach and then demonstrate the generality of GDEs on tasks across the biological sciences, spanning several data domains: DNA sequences, protein sequences, gene expression data, and microscopy data. Throughout, we explore different combinations of encoder-generator pairs, see App. A for a detailed discussion of architectures and training dynamics.

6.1 Benchmarking GDEs on synthetic distribution datasets

The design space of GDEs is large: any distributionally invariant encoder can be coupled to any conditional generative model. To guide our implementation choices, we systematically benchmark architectures using synthetic datasets. Included in the benchmarked models are two existing methods that GDEs generalize, kernel mean embeddings and Wasserstein Wormhole [11, 20].

Table 2: Wasserstein reconstruction error across synthetic distributional datasets. Computed as W_2 for normal and GMM, and as Sinkhorn divergence for MNIST and FMNIST.

Model	Normal	GMM	MNIST	FMNIST
KME + DDPM	0.04	2.17	80.46	111.01
W_2 Wormhole	0.20	2.88	263.29	320.18
GDE	**0.02**	**1.82**	**63.79**	**102.21**

We benchmark 30 combinations of encoders and generators on multivariate normal distributions in 5 dimensions. For evaluation we compute the Wasserstein reconstruction error from ground truth distribution by estimating means and covariance matrices from generated samples and using the closed-form for W_2 between Gaussians. We find that mean-pooled deep sets with skip-connections coupled with DDPM generators provide the highest quality generations, outperforming existing techniques. For synthetic distributions we present results for this architecture (see App. B.2).

In Table 2 we additionally benchmark this GDE architecture on three more sophisticated datasets: (1) 3-component Gaussian mixtures in 5 dimensions, (2) mixtures of MNIST [39] images according to categorical distributions of 3 classes, and (3) an analogous dataset using Fashion-MNIST [40]. For image datasets, where W_2 distances are not tractable, we instead compute the Sinkhorn divergences between pretrained Resnet18 [41] representations of generated and ground truth samples. In all cases, our chosen GDE architecture outperforms existing techniques, including kernel mean embeddings (with DDPM for generation) and Wasserstein Wormhole.

6.2 GDEs enable semi-supervised distribution-level representation learning

We next explore GDEs in our motivating example: for learning patient representations from a single-nucleus RNA-seq atlas of the human prefrontal cortex [4], which profiled over 6.3 million nuclei from 1,494 donors across neurological and psychiatric conditions. We consider each donor's nuclei as samples from an empirical distribution, and each condition as a label we wish to predict. As a baseline, we first train a supervised model to predict patient labels from nuclei sets using a mean-pooled deep sets architecture using 10% of the available labelled data. We compare this with a semisupervised model implemented using GDEs. To construct a GDE model, we combine a mean-pooled deep sets encoder with a CVAE generator, and train it using the same 10% labelled data available to the supervised model, along with the remaining 90% with labels withheld. Semi-supervised GDEs outperform supervised baselines across all evaluation metrics (Tab. 3).

Table 3: Patient label prediction from single-cell data. Semi-supervised GDEs improve performance.

Metric	Supervised	Semi-supervised
Accuracy	0.8791	**0.8887**
ROC AUC	0.4872	**0.5131**
F1 Score	0.1293	**0.1479**

6.3 Modeling clonal populations in lineage-traced scRNA-seq data

While many methods have been developed for learning representations of single cells from scRNA-seq data [42], methods for learning representations of cell populations remain relatively underexplored. This task is relevant to the analysis of lineage tracing data, where the unit of interest is a *clone*, or a population of cells that arise from the same progenitor.

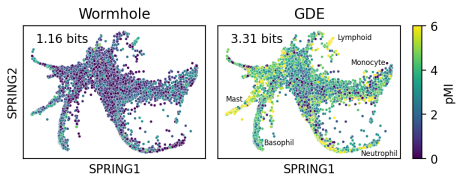

Figure 6: 2D embeddings of lineage-traced scRNA-seq data, hued by pointwise mutual information between clonal representation at early timepoint and clonal fate.

Using lineage-traced scRNA-seq data from mouse hematopoietic stem cells [43], we apply GDEs to learn clone-level representations by treating the set of cells within a clone as samples from an empirical distribution. Following prior frameworks [44], we evaluate the ability of representations to predict future clonal gene expression based on the mutual information (MI) between a clone's representation at an early timepoint during differentiation and its representation at a late timepoint. We find that GDEs with a CVAE generator outperform Wasserstein Wormhole embeddings by over 2 bits (Fig. 6). We next ask if this increase in predictive power is due to improved representations within certain cell types (e.g., neutrophils or monocytes). Decomposing MI estimates into their pointwise contributions [45], reveals contributions across the entire cell state space rather than any particular cell subtype (Fig. 6).

6.4 Predicting transcriptional responses to genetic perturbations

A central goal in genomics is to predict the transcriptional effects of genetic perturbations[46, 47, 48]. We evaluate GDE for genetic perturbation prediction, using the Perturb-seq data of Replogle et al. [49] that profiled gene expression responses to CRISPRi knockdown of thousands of genes.

We consider the following task: given the identity of a perturbation, predict the full distribution of transcriptional responses. We compare two approaches. In the first case, we train a linear model to predict the mean expression profile directly. In the second case, we predict the GDE embedding (trained on sets of cells subject to the same perturbation, via a Resnet Deep Sets encoder and CVAE generator as in Sec. 6.3) of the perturbation-induced expression distribution and then recover the mean

via a learned linear projection from the embedding space. In both cases, we use a ridge regression on top of GenePT embeddings [50] to enable zero-shot generalization across perturbation conditions, demonstrating that GDE improves both R^2 and MSE in Tab. 4. See Appendix B.5 for full details.

6.5 Learning morphological cellular responses to genetic perturbations

We apply GDEs to pooled image-based CRISPR screening data from Funk et al. [51], which profiles the phenotypic effects of perturbing 5,072 essential human genes in HeLa cells. The dataset includes over 20 million single-cell microscopy images with four stains, capturing diverse phenotypic variation.

Table 4: GenePT predicting held-out perturbations in mean expression space and GDE latent space.

	R^2	MSE
Mean	0.378290	1.854997
scVI	0.421491	1.551414
GDE	0.457941	1.500731

Each perturbation induces a distribution over cell morphologies based on perturbation groupings. We treat these as empirical distributions and train a GDE model to reconstruct them. To explore the role of inductive biases, we instantiate GDEs with two different priors: a spatial prior that models positional image structure (see App. B.6), and a perturbation prior that captures latent variation across perturbation conditions. These approaches capture spatial and perturbation sets, respectively.

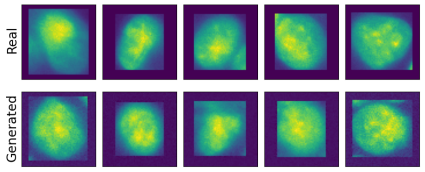

Figure 7: Real/generated DAPI images for the heldout RACGAP1 knockout.

Qualitatively, the model learns to reproduce phenotypic features, including nuclear shape, cytoplasmic texture, and boundary sharpness across perturbations (Fig. 7). Quantitatively, similar to Sec. 6.4 we hold out 30% of the most perturbative perturbations and use ridge regression with GenePT to enable zero-shot generalization across perturbations by predicting the GDE embedding. We then sample conditional on the predicted embedding and compute the nuclear signal intensity. The predictions on these held-out perturbations achieved an $R^2 = 0.7055$ and an MSE of 0.00068, indicating a strong zero-shot generalization of phenotypic outcomes.

6.6 Decoding yeast promoter sequence activity with GDEs

We next consider a large-scale dataset from a massively parallel reporter assay measuring transcriptional activity across 34 million randomly generated yeast promoter sequences [52]. Each promoter consists of a random 80 nucleotide DNA sequence embedded in a fixed DNA scaffold and assayed for expression in yeast cells. Because the sequences are randomly sampled, there is no shared structure across examples so unconditional generative models cannot learn anything meaningful. Instead, the signal lies entirely in how distributions over sequences give rise to distributions over expression levels, due to the presence of transcription factor binding sites (TFBS): short, position-specific DNA motifs that interact with transcription factors and control gene expression [52].

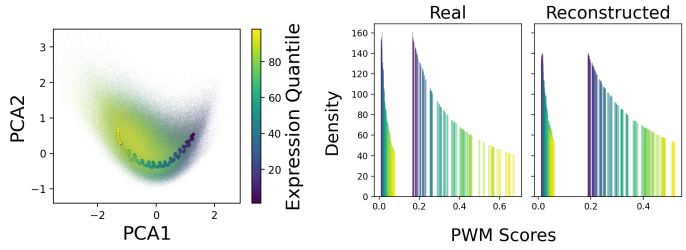

Figure 8: The PCA (left) of the GDE latent space of quantile embeddings with underlying 34 million promoter sequences and the recovered distribution of TFBS (right) as measured by motif counts in both the real and reconstructed data.

We construct a distributional learning task where each training example is a set of sequences sampled from a narrow expression quantile; we hold out the top 5 quantiles. We train a GDE with a 1D convolutional network over the one-hot encoded sequences as the encoder and HyenaDNA [35] as the decoder. As shown in Fig. 8, the learned GDE embeddings reflect a smooth gradient across expression quantiles. Using the set of all known yeast TFBS [53] we can identify the motifs present in each of the real and generated sequences. Reconstructed motif distributions closely match those of the input, indicating that the model learns to represent biologically meaningful variation across promoter sets. Further details are available in App. B.8.

6.7 Modeling spatiotemporal distributions of viral lineages

Powerful modeling approaches have been developed to represent individual protein sequences [54, 55, 56, 57, 34]. Here, we show that the GDE framework can naturally lift these modeling approaches to learn representations of distributions of sequences. In particular, we model distributions of SARS-CoV2 spike protein sequences over time and location. Using a dataset from the Global Initiative on Sharing All Influenza Data (GISAID) [58], we group sequences by sampling month and site location and treat each group as an empirical distribution over protein sequences. We embed these distributions using a GDE which couples the ESM architecture [56] to a mean-pooled deep sets as the encoder and a conditional ProGen2 architecture [34] as the generator.

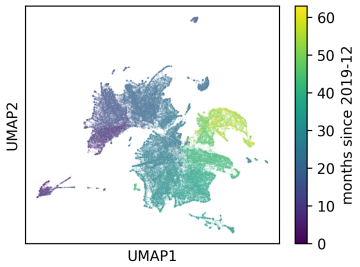

Figure 9: GDE representations of protein sequence distributions. Each point corresponds to a set of SARS-CoV2 spike sequences obtained from one lab in one month.

As shown in Fig. 9, the learned latent space organizes samples chronologically, suggesting that GDEs capture time-varying signal about sequence distributions. And indeed, this is observed quantitatively: ridge regression on GDE representations predicts the month of held out sequence distributions with mean absolute error (MAE) of 1.83 ± 0.01 months, an improvement over the baseline of mean-pooled ESM embeddings with MAE of 2.24 ± 0.01 months (errors reported as mean $\pm$ s.e.m. over 10 random train/test splits). See App B.9 for further details.

Similarly, we also observe a spatial signal, albeit much weaker. An SVM trained to classify distributions by country achieves 0.28 ± 0.001 accuracy from GDE representations, compared to 0.25 ± 0.003 from mean-pooled ESM embeddings. Both approaches slightly outperform the baseline of predicting the most common dataset label ('USA' with accuracy 0.21).

7 Discussion

We introduce *generative distribution embeddings*, a framework that couples distribution-invariant encoders with conditional generators to learn structured representations of distributions. Finite sample sets are mapped by smooth embeddings that asymptotically identify the underlying distribution, enabling consistent reconstruction in the large-sample limit. We formalized these properties via connections to predictive sufficiency and statistical manifold embeddings, and proved that a broad class of encoder architectures is asymptotically normal and unbiased when trained via a plug-in loss.

We demonstrated GDEs across a diverse set of large-scale biological problems. These applications highlight the generality of GDEs and their ability to operate directly on measurement data while modeling population-level structure. Crucially, GDEs support flexible distributional constructions (e.g. spatial neighborhoods, time windows, expression quantiles), showing that a wide range of problems can be cast as population-level modeling tasks. Code for model training and dataset preprocessing is available at this Github repository.

Limitations GDEs rely on sensible choices of meta-distributional priors (i.e. construction of sets, Sec. 3), often requiring careful, domain-specific design. GDEs also pose practical engineering challenges (propagate gradients to the encoder through the generator, scaling to large set sizes) discussed in App. A. On the theoretical side, the current formalism assumes exchangeable samples, and does not admit non-i.i.d. samples within a distribution. Regarding geometry, we provide empirical but not mechanistic evidence that GDEs learn isometries across domains.

Extensions GDEs can serve as a tool for generalization (akin to meta flow matching [19]), can be expanded to settings where the i.i.d. assumption within sets of samples does not hold, and extended to semi-supervised settings. More broadly, GDEs point toward questions at the intersection of empirical process theory, information geometry, and generative modeling; we hope this connection can be explored more deeply in future work.

Acknowledgements Funding: J.S.G. and O.O.A. are supported by NIH grants R01-EB031957, R01-AG074932, and R01-GM148745; G. Harold & Leila Y. Mathers Charitable Foundation; Rett

Syndrome Research Trust; The Gordon and Betty Moore Foundation; Impetus Grants; Cystic Fibrosis Foundation Pioneer Grant; Google Deepmind; Sanofi; Yosemite; Michelson Foundation; Hevolution Foundation; American Federation for Aging Research; Pivotal Life Sciences; and the MGB Gene and Cell Therapy Institute. G.G. is supported by a Tayebati Fellowship. N.F. is supported by the NSF GRFP.

References

[1] Kei Takeuchi and Masafumi Akahira. Characterizations of prediction sufficiency (adequacy) in terms of risk functions. *The Annals of Statistics*, 3(4):1018–1024, 1975.

[2] Erik N Torgersen. Prediction sufficiency when the loss function does not depend on the unknown parameter. *The Annals of Statistics*, 5(1):155–163, 1977.

[3] Felix Otto. The geometry of dissipative evolution equations: the porous medium equation. 2001.

[4] John F Fullard, Prashant Nm, Donghoon Lee, Deepika Mathur, Karen Therrien, Aram Hong, Clara Casey, Zhiping Shao, Marcela Alvia, Stathis Argyriou, et al. Population-scale cross-disorder atlas of the human prefrontal cortex at single-cell resolution. *Scientific Data*, 12(1): 954, 2025.

[5] Netanel Loyfer, Judith Magenheim, Ayelet Peretz, Gordon Cann, Joerg Bredno, Agnes Klochendler, Ilana Fox-Fisher, Sapir Shabi-Porat, Merav Hecht, Tsuria Pelet, Joshua Moss, Zeina Drawshy, Hamed Amini, Patriss Moradi, Sudharani Nagaraju, Dvora Bauman, David Shveiky, Shay Porat, Uri Dior, Gurion Rivkin, Omer Or, Nir Hirshoren, Einat Carmon, Alon Pikarsky, Abed Khalaileh, Gideon Zamir, Ronit Grinbaum, Machmud Abu Gazala, Ido Mizrahi, Noam Shussman, Amit Korach, Ori Wald, Uzi Izhar, Eldad Erez, Vladimir Yutkin, Yaacov Samet, Devorah Rotnemer Golinkin, Kirsty L Spalding, Henrik Druid, Peter Arner, A M James Shapiro, Markus Grompe, Alex Aravanis, Oliver Venn, Arash Jamshidi, Ruth Shemer, Yuval Dor, Benjamin Glaser, and Tommy Kaplan. A DNA methylation atlas of normal human cell types. *Nature*, 613(7943):355–364, 4 January 2023. ISSN 0028-0836,1476-4687. doi: 10.1038/s41586-022-05580-6.

[6] Gokul Gowri, Peng Yin, and Allon M. Klein. Measurement noise scaling laws for cellular representation learning, 2025. URL `https://arxiv.org/abs/2503.02726`.

[7] Alex Smola, Arthur Gretton, Le Song, and Bernhard Schölkopf. A Hilbert space embedding for distributions. In *Lecture Notes in Computer Science*, Lecture notes in computer science, pages 13–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007. ISBN 9783540752240,9783540752257. doi: 10.1007/978-3-540-75225-7_5.

[8] Krikamol Muandet, K Fukumizu, Francesco Dinuzzo, and B Scholkopf. Learning from distributions via support measure machines. *Neural Information Processing Systems*, 25:10–18, 29 February 2012.

[9] Junier B Oliva, B Póczos, and J Schneider. Distribution to Distribution Regression. *International Conference on Machine Learning*, 28(3):1049–1057, 16 June 2013.

[10] Zoltan Szabo, Arthur Gretton, Barnabas Poczos, and Bharath Sriperumbudur. Two-stage sampled learning theory on distributions. In *Artificial Intelligence and Statistics*, pages 948–957. PMLR, 21 February 2015.

[11] Krikamol Muandet, Kenji Fukumizu, Bharath Sriperumbudur, and Bernhard Schölkopf. Kernel mean embedding of distributions: A review and beyond. *Foundations and Trends® in Machine Learning*, 10(1-2):1–141, 2017. ISSN 1935-8237,1935-8245. doi: 10.1561/2200000060.

[12] Harrison Edwards and Amos Storkey. Towards a Neural Statistician. In *International Conference on Learning Representations*, 6 February 2017.

[13] S I Amari and H Nagaoka. Methods of information geometry. 191, 2000.

[14] K Carter, R Raich, W Finn, and A Hero. FINE: Fisher Information Nonparametric Embedding. *IEEE transactions on pattern analysis and machine intelligence*, 31(11):2093–2098, 14 February 2008. ISSN 0162-8828,1939-3539. doi: 10.1109/TPAMI.2009.67.

[15] Yonghyeon Lee, Seungyeon Kim, Jinwon Choi, and F Park. A statistical manifold framework for point cloud data. *International Conference on Machine Learning*, pages 12378–12402, 2022.

[16] Hannes Stärk, Bowen Jing, Chenyu Wang, Gabriele Corso, Bonnie Berger, R Barzilay, and T Jaakkola. Dirichlet flow matching with applications to DNA sequence design. *International Conference on Machine Learning*, 235:46495–46513, 8 February 2024. doi: 10.48550/arXiv.2402.05841.

[17] Oscar Davis, Samuel Kessler, Mircea Petrache, I Ceylan, Michael M Bronstein, and A Bose. Fisher flow matching for generative modeling over discrete data. *Neural Information Processing Systems*, abs/2405.14664:139054–139084, 23 May 2024. doi: 10.48550/arXiv.2405.14664.

[18] Doron Haviv, Aram-Alexandre Pooladian, Dana Pe'er, and Brandon Amos. Wasserstein flow matching: Generative modeling over families of distributions, 2024. URL `https://arxiv.org/abs/2411.00698`.

[19] Lazar Atanackovic, Xi Zhang, Brandon Amos, Mathieu Blanchette, Leo J Lee, Yoshua Bengio, Alexander Tong, and Kirill Neklyudov. Meta Flow Matching: Integrating Vector Fields on the Wasserstein Manifold. In *The Thirteenth International Conference on Learning Representations*, 4 October 2024.

[20] D Haviv, Russell Z Kunes, Thomas Dougherty, Cassandra Burdziak, T Nawy, Anna Gilbert, and D Pe'er. Wasserstein Wormhole: Scalable optimal transport distance with transformers. *International Conference on Machine Learning*, 235:17697–17718, 15 April 2024. doi: 10.48550/arXiv.2404.09411.

[21] John W Fisher, III, A Ihler, and Paul A Viola. Learning informative statistics: A nonparametnic approach. *Neural Information Processing Systems*, pages 900–906, 29 November 1999. doi: 10.5555/3009657.3009784.

[22] Paul Joyce and Paul Marjoram. Approximately sufficient statistics and bayesian computation. *Statistical applications in genetics and molecular biology*, 7(1):Article26, 30 August 2008. ISSN 1544-6115,2194-6302. doi: 10.2202/1544-6115.1389.

[23] Justin Alsing, Benjamin Wandelt, and Stephen Feeney. Massive optimal data compression and density estimation for scalable, likelihood-free inference in cosmology. *Monthly notices of the Royal Astronomical Society*, 477(3):2874–2885, 1 July 2018. ISSN 0035-8711,1365-2966. doi: 10.1093/mnras/sty819.

[24] Yanzhi Chen, Dinghuai Zhang, Michael U Gutmann, Aaron Courville, and Zhanxing Zhu. Neural Approximate Sufficient Statistics for Implicit Models. In *International Conference on Learning Representations*, 2 October 2020.

[25] Maxime Peyrard and Kyunghyun Cho. Meta-Statistical Learning: Supervised Learning of Statistical Inference, 2025.

[26] M Zaheer, Satwik Kottur, Siamak Ravanbakhsh, B Póczos, R Salakhutdinov, and Alex Smola. Deep Sets. *Advances in neural information processing systems*, 30, 10 March 2017.

[27] E Wagstaff, F Fuchs, Martin Engelcke, Michael A Osborne, and I Posner. Universal approximation of functions on sets. *Journal of machine learning research: JMLR*, 23(151):151:1–151:56, 5 July 2021. ISSN 1532-4435,1533-7928.

[28] Lily H Zhang, Veronica Tozzo, J Higgins, and R Ranganath. Set norm and equivariant skip connections: Putting the deep in Deep Sets. *International Conference on Machine Learning*, 162:26559–26574, 23 June 2022. doi: 10.48550/arXiv.2206.11925.

[29] Drew A Hudson, Daniel Zoran, Mateusz Malinowski, Andrew K Lampinen, Andrew Jaegle, James L McClelland, Loic Matthey, Felix Hill, and Alexander Lerchner. SODA: Bottleneck diffusion models for representation learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23115–23127. IEEE, 16 June 2024. doi: 10.1109/cvpr52733.2024.02181.

[30] Diederik P Kingma and M Welling. An Introduction to Variational Autoencoders. *Found. Trends Mach. Learn.*, 12(4):307–392, 6 June 2019. doi: 10.1561/2200000056.

[31] Aude Genevay, Gabriel Peyre, and Marco Cuturi. Learning Generative Models with Sinkhorn Divergences. In *International Conference on Artificial Intelligence and Statistics*, pages 1608–1617. PMLR, 31 March 2018.

[32] Soheil Kolouri, Kimia Nadjahi, Umut Simsekli, R Badeau, and G Rohde. Generalized Sliced Wasserstein Distances. *Neural Information Processing Systems*, 32:261–272, 1 February 2019.

[33] Jonathan Ho, Ajay Jain, and P Abbeel. Denoising Diffusion Probabilistic Models. *Neural Information Processing Systems*, abs/2006.11239:6840–6851, 19 June 2020.

[34] Erik Nijkamp, Jeffrey A Ruffolo, Eli N Weinstein, Nikhil Naik, and Ali Madani. ProGen2: Exploring the boundaries of protein language models. *Cell systems*, 14(11):968–978.e3, 15 November 2023. ISSN 2405-4712,2405-4720. doi: 10.1016/j.cels.2023.10.002.

[35] Eric D Nguyen, Michael Poli, Marjan Faizi, A Thomas, Callum Birch-Sykes, Michael Wornow, Aman Patel, Clayton M Rabideau, Stefano Massaroli, Y Bengio, Stefano Ermon, S Baccus, and Christopher Ré. HyenaDNA: Long-range genomic sequence modeling at single nucleotide resolution. *Neural Information Processing Systems*, 36:43177–43201, 27 June 2023. doi: 10.48550/arXiv.2306.15794.

[36] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 22 October 2020. ISSN 0001-0782,1557-7317. doi: 10.1145/3422622.

[37] Y Lipman, Ricky T Q Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow Matching for generative modeling. *International Conference on Learning Representations*, abs/2210.02747, 6 October 2022.

[38] Julie Delon and Agnes Desolneux. A wasserstein-type distance in the space of gaussian mixture models. *SIAM Journal on Imaging Sciences*, 13(2):936–970, 2020.

[39] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2, 2010.

[40] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *CoRR*, abs/1708.07747, 2017. URL `http://arxiv.org/abs/1708.07747`.

[41] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778. IEEE, June 2016. ISBN 9781467388511,9781467388528. doi: 10.1109/cvpr.2016.90.

[42] Romain Lopez, Jeffrey Regier, Michael B Cole, Michael I Jordan, and Nir Yosef. Deep generative modeling for single-cell transcriptomics. *Nature methods*, 15(12):1053–1058, December 2018. ISSN 1548-7091,1548-7105. doi: 10.1038/s41592-018-0229-2.

[43] Caleb Weinreb, Alejo Rodriguez-Fraticelli, Fernando D Camargo, and Allon M Klein. Lineage tracing on transcriptional landscapes links state to fate during differentiation. *Science*, 367 (6479), 14 February 2020. ISSN 0036-8075,1095-9203. doi: 10.1126/science.aaw3381.

[44] Gokul Gowri, Xiao-Kang Lun, Allon M Klein, and Peng Yin. Approximating mutual information of high-dimensional variables using learned representations. In A Globerson, L Mackey, D Belgrave, A Fan, U Paquet, J Tomczak, and C Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 132843–132875. Curran Associates, Inc., 2024.

[45] Xianghao Kong, Ollie Liu, Han Li, Dani Yogatama, and Greg Ver Steeg. Interpretable Diffusion via Information Decomposition. In *The Twelfth International Conference on Learning Representations*, 13 October 2023.

[46] Omar O Abudayyeh and Jonathan S Gootenberg. Programmable biology through artificial intelligence: from nucleic acids to proteins to cells. *Nature Methods*, 21(8):1384–1386, 2024.

[47] Charlotte Bunne, Yusuf Roohani, Yanay Rosen, Ankit Gupta, Xikun Zhang, Marcel Roed, Theo Alexandrov, Mohammed AlQuraishi, Patricia Brennan, Daniel B Burkhardt, et al. How to build the virtual cell with artificial intelligence: Priorities and opportunities. *Cell*, 187(25): 7045–7063, 2024.

[48] Julia Joung, Sai Ma, Tristan Tay, Kathryn R Geiger-Schuller, Paul C Kirchgatterer, Vanessa K Verdine, Baolin Guo, Mario A Arias-Garcia, William E Allen, Ankita Singh, et al. A transcription factor atlas of directed differentiation. *Cell*, 186(1):209–229, 2023.

[49] Joseph M Replogle, Reuben A Saunders, Angela N Pogson, Jeffrey A Hussmann, Alexander Lenail, Alina Guna, Lauren Mascibroda, Eric J Wagner, Karen Adelman, Gila Lithwick-Yanai, et al. Mapping information-rich genotype-phenotype landscapes with genome-scale perturb-seq. *Cell*, 185(14):2559–2575, 2022.

[50] Yiqun Chen and James Zou. Genept: a simple but effective foundation model for genes and cells built from chatgpt. *bioRxiv*, pages 2023–10, 2024.

[51] Luke Funk, Kuan-Chung Su, Jimmy Ly, David Feldman, Avtar Singh, Brittania Moodie, Paul C Blainey, and Iain M Cheeseman. The phenotypic landscape of essential human genes. *Cell*, 185 (24):4634–4653, 2022.

[52] Carl G de Boer, Eeshit Dhaval Vaishnav, Ronen Sadeh, Esteban Luis Abeyta, Nir Friedman, and Aviv Regev. Deciphering eukaryotic gene-regulatory logic with 100 million random promoters. *Nature biotechnology*, 38(1):56–65, 2020.

[53] Carl G De Boer and Timothy R Hughes. Yetfasco: a database of evaluated yeast transcription factor sequence specificities. *Nucleic acids research*, 40(D1):D169–D179, 2012.

[54] Adam J Riesselman, John B Ingraham, and Debora S Marks. Deep generative models of genetic variation capture the effects of mutations. *Nature methods*, 15(10):816–822, October 2018. ISSN 1548-7091,1548-7105. doi: 10.1038/s41592-018-0138-4.

[55] Tristan Bepler and Bonnie Berger. Learning the protein language: Evolution, structure, and function. *Cell systems*, 12(6):654–669.e3, 16 June 2021. ISSN 2405-4712,2405-4720. doi: 10.1016/j.cels.2021.05.017.

[56] Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences of the United States of America*, 118(15), 13 April 2021. ISSN 0027-8424,1091-6490. doi: 10.1073/pnas.2016239118.

[57] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, Allan Dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Salvatore Candido, and Alexander Rives. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 17 March 2023. ISSN 0036-8075,1095-9203. doi: 10.1126/science.ade2574.

[58] Yuelong Shu and John McCauley. GISAID: Global initiative on sharing all influenza data - from vision to reality. *Euro surveillance : bulletin Europeen sur les maladies transmissibles [Euro surveillance : European communicable disease bulletin]*, 22(13), 30 March 2017. ISSN 1025-496X,1560-7917. doi: 10.2807/1560-7917.ES.2017.22.13.30494.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[60] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[61] Jacob Devlin. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[62] Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Nikita Smetanin, Robert Verkuil, Ori Kabeli, Yaniv Shmueli, et al. Evolutionary-scale prediction of atomic-level protein structure with a language model. *Science*, 379(6637):1123–1130, 2023.

[63] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*, 2019.

[64] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[65] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 69(6 Pt 2):066138, June 2004. ISSN 1539-3755. doi: 10.1103/PhysRevE.69.066138.

[66] Netanel Loyfer, Jonathan Rosenski, and Tommy Kaplan. wgbstools: A computational suite for DNA methylation sequencing data representation, visualization, and analysis. *bioRxiv*, page 2024.05.08.593132, 10 May 2024. doi: 10.1101/2024.05.08.593132.

[67] Jean-David Benamou and Yann Brenier. A computational fluid mechanics solution to the monge-kantorovich mass transfer problem. *Numerische Mathematik*, 84(3):375–393, 2000.

[68] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge University Press, 2019.

[69] David Blackwell. An analog of the minimax theorem for vector payoffs. 1956.

[70] Aad W Van Der Vaart and Jon A Wellner. *Weak convergence*. Springer, 1996.

A Architectures and Training Dynamics

In this section we outline some general details about the architectures and training dynamics for GDEs. In the following section we will give more detailed explanations about each specific experiment, in addition to full details available in the codebase. All of these findings are somewhat provisional, and there is significant scope for future work to further explore these design choices, but we hope this is a useful complement to our codebase for researchers trying to train their own GDEs.

A.1 Encoder Architectures

Our framework utilizes permutation-invariant encoders to map input sets $S_m = \{x_1, ..., x_m\}$, where each $x_i \in \mathbb{R}^d$, to a fixed-dimensional latent representation $z \in \mathbb{R}^l$. We primarily employ several types of set encoders, including variants based on self-attention, Graph Neural Network (GNN)-style pooling, and residual connections. All encoders typically conclude by applying a final pooling operation (e.g., mean pooling) across the element representations, followed by a linear projection and a non-linearity (e.g., SELU) to produce the final latent vector z.

A.1.1 Simple Self-Attention Encoder

This encoder provides a baseline transformer-based approach. It first applies a linear layer followed by a SELU activation to project input elements x_i into a hidden dimension H. It then processes these representations through a series of multi-head Self-Attention blocks [59]. This architecture directly models pairwise interactions within the set.

A.1.2 Simple GNN Encoder

The simple GNN-style encoder offers an alternative based on iterative pooling and non-linear transformations, distinct from the standard DeepSets [26] sum-decomposition. It starts with an MLP projection into the hidden dimension H. Subsequently, it applies a sequence of layers, each performing a pooling operation across the set followed by an MLP. This structure iteratively refines element representations based on aggregated set information.

Pooling Operations: Our theoretical framework (see Appendix D.2) justifies the use of pooling operations that correspond to M/Z-estimators. We focus on mean pooling but additionally implement median pooling as an illustrative example. Notably, max pooling is generally not suitable in this context as its non-differentiability breaks the convergence guarantees we are interested in for Eq. (1), see the remarks in App D.2 for details. Future work might thoroughly explore which pooling operations lead to the greatest flexibility and stability for distribution embedding.

A.1.3 ResNet-GNN Encoder

To improve gradient flow and enable deeper architectures, we enhance the GNN-style encoder with residual connections. This encoder first projects each input element x_i into H using an MLP. It then processes the set through a series of blocks where each block k computes an intermediate representation $h_i^{(k)}$ for each element i. The core operation within a block uses mean pooling (or median pooling). Inspired by ResNet [60, 28], we incorporate skip connections. The input to block k includes the output from the previous block $h^{(k-1)}$, a linear projection of the original input x, and the output of the initial MLP projection. Formally:

$$h^{(k)} = \text{LayerNorm}(\text{PooledFC}(h^{(k-1)}) + h^{(k-1)} + \text{Linear}_k(x))$$

where $h^{(0)}$ is the output of the initial input projection combined with a projection of x, followed by Layer Normalization. This structure ensures the original input signal is preserved.

A.1.4 ResNet-Transformer Encoder

This variant follows the same residual structure as the ResNet-MLP encoder but replaces the layers with standard multi-head Self-Attention blocks [59]. This potentially allows the model to learn more complex interactions while benefiting from the improved training dynamics of residual connections. The skip connection mechanism remains identical to the ResNet-MLP version.

A.1.5 Encoder Comparison

Transformer-based encoders (Simple Self-Attention and ResNet-Transformer) often leverage pre-trained weights effectively and can converge in fewer epochs compared to GNN-style approaches. However, this typically comes at a higher computational cost per epoch and during inference due to the quadratic complexity of self-attention with respect to set size m. With sufficient training, we find that the GNN-based architectures, particularly the ResNet-GNN, achieve strong performance, often rivaling the transformer variants while being more computationally efficient for large sets.

Alternative Generative Strategies and Sampling The Wasserstein Wormhole [20] uses a self-attention decoder with fixed positional embeddings that can map the latent z back to samples. One potential method replaces fixed positional embeddings with samples drawn from a simple distribution (e.g., Gaussian) transforming this into a true generator. But this incurs substantial computational costs (e.g., quadratic cost in the number of generated samples for attention-based sampling decoders), and it is not clear this would lead to significant improvements in performance.

It also becomes less obvious how to adapt existing generator architectures using this approach. One option is to use self-attention to construct sample-specific condtional signals from the latent z and the noise vector, and then condition the generator on this signal. This is significantly more complex, and is not clear that this would lead to significant improvements in performance.

A.2 Adapting Pre-trained Models

Our framework is designed to flexibly incorporate pre-trained models, leveraging their learned representations and generative capabilities. We adapt pre-trained models for both the encoder and the generator components.

A.2.1 Encoder Adaptation

For tasks involving complex input modalities like natural language or protein sequences, we can utilize pre-trained transformer-based encoders such as BERT [61] or ESM [62] as powerful feature extractors. These pre-trained models can serve as the initial feature extraction layer, whose outputs $\{h_1, ..., h_N\}$ are then fed into the subsequent aggregation layers of our set encoders (e.g., ResNet-GNN or ResNet-Transformer, see subsection A.1).

The adaptation process typically involves:

1. **Loading Pre-trained Weights:** We load the desired pre-trained encoder model using standard libraries like Hugging Faces `transformers` [63].
2. **Feature Extraction:** For each element x_i in the input set $X = \{x_1, ..., x_N\}$, we pass it through the pre-trained transformer to obtain a contextualized representation h_i. Often, the output embedding corresponding to a special token (like `[CLS]` in BERT) or the mean/max-pooled output of the final hidden states is used.
3. **Set Aggregation:** These element-wise feature vectors $\{h_1, ..., h_N\}$ are then fed into the subsequent layers of our chosen set encoder (e.g., ResNet-MLP or ResNet-Transformer layers) which perform the permutation-invariant aggregation to produce the final latent representation z.
4. **Fine-tuning (Optional):** Depending on the task and dataset size, the pre-trained encoder's weights might be kept frozen initially or fine-tuned jointly with the rest of the model during end-to-end training.

A.2.2 Generator Adaptation and Conditioning

A core strength of our approach is the ability to use large pre-trained causal language models (LMs), such as GPT-2 [64], ProGen2 [34], or specialized models like HyenaDNA [35], as the conditional generator $p_\theta(x|z)$.

The adaptation involves:

1. **Loading Pre-trained Weights:** We load the chosen pre-trained causal LM and its associated tokenizer using 'transformers' [63].

2. **Prefix Conditioning:** The primary challenge is to effectively condition the generatorś output on the latent set representation z produced by the encoder. In practice, we find prefix tuning to be an effective and widely applicable method. The latent vector $z \in \mathbb{R}^L$ is projected, typically via a small MLP W_p, into one or more vectors $p = W_p(z)$ that have the same hidden dimension as the LM. These projected vectors p are then treated as continuous "prefix" embeddings prepended to the actual input sequence embeddings $E(x_{<T})$ before they are processed by the transformer layers. The model learns to interpret this prefix as the conditioning signal specifying the target distribution. Mathematically, the input embedding sequence to the transformer becomes $[p; E(x_{<T})]$. The attention mask is adjusted accordingly to allow all sequence tokens $x_{<T}$ to attend to the prefix p.
3. **Fine-tuning:** The pre-trained generator weights can be either frozen or fine-tuned. Fine-tuning the entire model allows the LM to adapt its generation process based on the conditioning prefix p. Freezing the LM backbone and only training the conditioning projection W_p (and potentially adapter layers) can be more parameter-efficient.

A.3 Training Details and Considerations

A.3.1 Learning Rate Schedule

For simpler models we use a fixed learning rate, but for more complex models we typically employ a cosine annealing learning rate schedule during training. This involves starting with an initial learning rate and gradually decreasing it towards zero following a cosine curve over the course of training epochs. This schedule is often effective in achieving stable convergence and good final performance. In general we have found that whatever the current state of the art for training the (unconditional) generator is, that will generally give good results when learning the encoder-generator jointly.

A.3.2 Performance and Convergence

Our experiments generally indicate that this training setup, combined with the described architectures and adaptation strategies, leads to strong performance across various tasks and datasets presented in the main paper. As noted in subsection A.1.5, the choice of encoder can impact convergence speed and computational cost.

A.3.3 Set Size and Batching Trade-offs

We observe that achieving optimal performance sometimes necessitates using large input set sizes (N). However, processing large sets can significantly increase the computational and memory requirements per batch, particularly for the attention mechanisms in transformer-based encoders or generators. This often forces a reduction in the overall batch size to fit within hardware constraints. Smaller batch sizes can, in turn, lead to increased variance in the loss gradients, potentially slowing down or destabilizing training. Careful tuning of the set size N, batch size, and learning rate parameters is often required to balance performance and training efficiency for a given task and hardware setup.

A.3.4 Gradient Propogation Challenges

A potential challenge arises, particularly with deeper encoder and generator architectures. The encoder only receives a learning signal indirectly through the generator via the shared latent variable z. If the generator itself struggles to utilize the latent information effectively, or if the dimensionality L of z creates an information bottleneck, the gradients flowing back to the encoder can become weak or noisy. This can make training deep encoders difficult. Addressing this might require more sophisticated generator architectures capable of integrating the latent information more effectively or alternative training schemes with auxiliary losses directly on the encoder. We found these issues in the simple encoder architectures, but they seemed to be alleviated in the ResNet-based architectures.

A.4 Implementation recipe for GDEs

The GDE framework is instantiated by pairing a distributionally invariant encoder with a conditional generative model. The following steps outline a general recipe for building GDEs across diverse data domains:

1. **Sample from the metadistribution (construct sets)** Group raw data into sets $S_i = \{x_{ij}\}_{j=1}^{m}$, where each set reflects a draw from an unknown latent distribution P_i. Groupings can be based on discrete metadata (e.g., text by author, reviews by rating, images by label, cell clones, gene perturbations) or continuous metadata (e.g., time, location, expression quantiles). Sets need not be mutually exclusive, meaning a single data point can belong to multiple sets.
2. **Choose a distributionally invariant encoder** Select or construct a distributionally invariant encoder $\mathcal{E}$. This selection generally involves (1) using an architecture for element-wise embeddings and (2) pooling across element-wise embeddings with a sample mean (or other M-estimate). We found that architectures with multiple pooling layers, where each layer's pooled output is concatenated with the element-wise embeddings, were particularly effective. This contrasts with pure DeepSets-style architectures that only pool once at the final layer. For deeper architectures, we have found that including skip-connections improves performance, especially if the generator is also a relatively deep network.
3. **Build a conditional generator** The generator $\mathcal{G}$ "decodes" from latent space back to the sample space. It should be conditionable on $z = \mathcal{E}(S)$.
4. **Train via plug-in loss** Optimize the generator to minimize the generator loss function $\ell(P_m, \mathcal{G}(\mathcal{E}(S_m)))$. This loss should be the standard training objective for the conditional generator. This plug-in loss encourages reconstruction of the true distribution.

A.5 Encoder and Generator Architectures by Experiment

The encoder-generator pairs used for each application in the paper are shown in the table below.

Sec.	Task	Set Construction	Encoder Arch.	Generator Arch.	Notes
6.1	MNIST, FMNIST	Same image class	see Table 1	see Table 1	Synthetic data benchmark
6.2	Lineage-traced scRNA-seq	Same cell clones	ResNet-GNN	CVAE	
6.3	Genetic perturbation (scRNA-seq)	Same perturbation	ResNet-GNN	CVAE	
6.4	Morphological responses (cell images)	Same perturbation	2D Conv-GNN	DDPM (U-Net)	
6.5	Tissue-specific methylation	Same patient; Same tissue type	1D Conv-GNN	HyenaDNA	Uses prefix conditioning
6.6	Yeast promoter quantile decoding	Expression quantile (continuous)	1D Conv-GNN	HyenaDNA	Uses prefix conditioning
6.7	Viral protein spatiotemporal modeling	Same sampling month and location	ESM + mean pooling	ProGen2	Uses prefix conditioning

B Experiments

B.1 Determining tissue-specific methylation signatures from bisulfite sequencing reads

Analyzing sequencing data typically extensive preprocessing, including alignment to a reference genome. GDEs present an alternative, where sequencing reads can be modeled directly – without alignment or other preprocessing steps. To demonstrate this capability, we show that GDEs can detect tissue-specific DNA methylation patterns directly from bisulfite sequencing (BS-seq) reads. BS-seq measures methylation indirectly through substitution errors: methylated cytosines remain unchanged, while unmethylated cytosines are substituted as thymines. Using publicly available methylation data from diverse tissues [5], we simulate sample-specific BS-seq read distributions by imposing corresponding base substitutions to the reference genome (see Appendix B.7 for details).

Critically, we do not provide the GDE model with any explicit information about methylation signals, the structure of the experimental assay, or a reference genome. The model has access only to sets of sequencing reads grouped by both patient and tissue type. For the GDE model architecture, we choose a 1D convolutional network encoder, and the decoder is a HyenaDNA model [35]. To support large-scale inference over tens of millions of reads per patient, we process 200,000 reads at a time through the encoder and aggregate the resulting embeddings using a simple mean, justified by Theorem 2. This design allows the model to scale efficiently while preserving distributional fidelity.

Our approach enables end-to-end learning of methylation signatures from tissue-specific read distributions. There are two levels of tissue classification, a coarse level with 37 categories and a fine-grained level classification with 83 tissues. Training a linear classifier on top of the GDE latent space, we achieve a test accuracy of 60% on the coarse task and 35% on the fine-grained classification.

B.2 Additional semi-synthetic experimental results

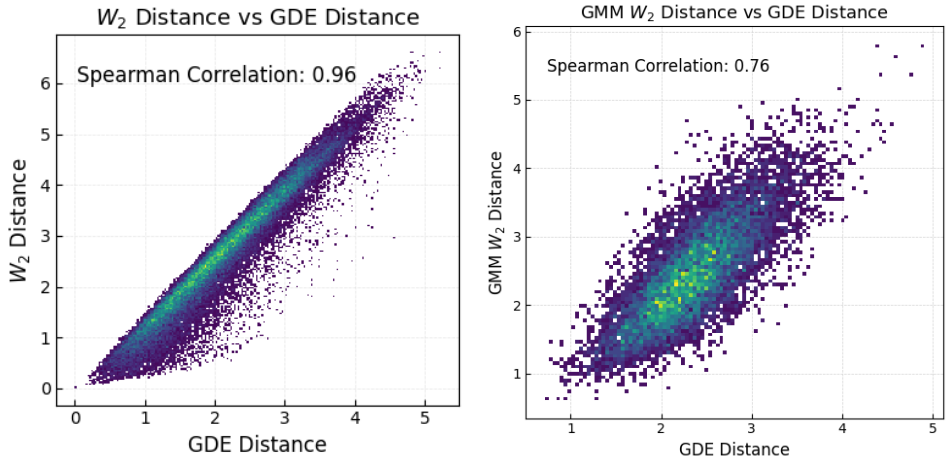

Figure 10: Left: Distance correlation showing high alignment between latent GDE distances and analytical W_2 distances (Spearman $\rho = 0.96$). Left: Distance correlation showing high alignment between latent GDE distances and the OT-GMM distance [38], which is a W_2 metric restricted to the subspace of GMMs (Spearman $\rho = 0.76$).

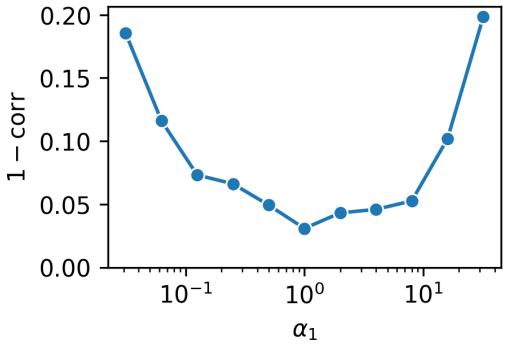

Figure 11: Expanding on Fig. 5 we show that the Pearson correlation between the W_2 (computed via normal approximation) and the latent GDE distances decreases as α_1 deviates from 1, while keeping fixed $\alpha_2 = \alpha_3 = 1$.

Table 5: W_2 reconstruction error of 30 possible GDE implementations (including two existing methods generalized by GDE, Wasserstein Wormhole and kernel mean embeddings) on 5-dimensional multivariate Gaussians. Covariance matrices sampled from Wishart distribution with scale of 1, and means sampled uniformly from $[0, 5]$. Further results included in Table 6.

Gen. ↓ \\ Enc. →	**Mean**	**Kernel mean**	**GNN**	**Med.-GNN**	**ResNet-GNN**	**SelfAttn.**
Sinkhorn	0.05	0.14	0.09	0.10	0.05	0.06
Sliced W_2	0.03	0.04	0.07	0.07	0.03	0.04
CVAE	0.16	0.16	0.19	0.20	0.15	0.17
DDPM	0.03	0.04	0.06	0.05	0.02	0.07
Wormhole	0.14	0.15	0.72	0.49	0.14	0.20

Table 6: W_2 reconstruction error (mean ± s.e.m. over 5 trials) for 30 possible GDE implementations (including two existing methods generalized by GDE, Wasserstein Wormhole and kernel mean embeddings) on 5-dimensional multivariate Gaussians. Covariance matrices sampled from Wishart distribution with scale of 0.1, and means sampled uniformly from $[0, 5]$.

Gen. ↓ \\ Enc. →	**Kernel mean**	**GNN**	**ResNet-GNN**	**Self-Attn.**
CVAE	0.15 ± 0.011	0.12 ± 0.006	0.12 ± 0.009	0.11 ± 0.007
DDPM	0.15 ± 0.008	0.13 ± 0.020	**0.09 ± 0.003**	0.10 ± 0.005
Direct SW	0.15 ± 0.008	0.13 ± 0.007	0.13 ± 0.009	0.15 ± 0.001
Direct Sinkhorn	0.29 ± 0.008	0.22 ± 0.010	0.17 ± 0.005	0.19 ± 0.010
Wormhole	0.23 ± 0.021	0.72 ± 0.090	0.24 ± 0.011	0.34 ± 0.021

B.3 Donor-level representation learning experiments

B.3.1 Data preprocessing

We use single-nucleus RNA-seq data from the Population-scale cross-disorder atlas of the human prefrontal cortex [4], which profiles over 6.3 million nuclei from 1,494 donors across 33 neurological and psychiatric conditions. The dataset consists of multiple sub-datasets, so to avoid integration issues we subset to the largest sub-dataset which contains 4 million cells. For each donor, raw count matrices were normalized to 10^4 counts per nucleus, log-transformed, and restricted to the top $2,000$ highly variable genes. We treat each donor's collection of nuclei as an empirical distribution over transcriptional states. Donor-level diagnostic metadata were obtained from the accompanying PsychAD clinical annotations, and we restrict prediction targets to the six major disease categories which have at least one positive and negative example in the dataset (Alzheimer's, Parkinson's, diffuse Lewy body, bipolar, schizophrenia, and vascular dementia).

B.3.2 Model architecture and training

Both the supervised and semi-supervised models use the same ResNet deep sets encoder to aggregate single-cell features into donor-level embeddings. For the supervised variant, the encoder is trained end-to-end with a classification head. For the semi-supervised GDE, the same encoder is coupled to a conditional variational autoencoder (CVAE) generator, trained jointly to reconstruct cell distributions while predicting disease labels for the 10% of donors with labeled diagnoses. In both cases, we use a 64-dimensional latent space and two hidden layers of size 128. After training, a logistic regression classifier is fit on the donor embeddings to predict multi-label disease status across the six categories.

B.3.3 Evaluation

We report donor-level predictive performance using accuracy, balanced accuracy, ROC-AUC, and F1 score on 10% of completely heldout data. Semi-supervised GDEs outperform purely supervised models across all metrics (Table 3), demonstrating that unlabeled donor distributions improve representation quality and generalization in large heterogeneous cohorts.

B.4 Lineage-traced scRNA-seq experiments

B.4.1 Data preprocessing details

We use lineage tracing data from Weinreb et al. [43]. The single-cell RNA sequencing (scRNA-seq) count matrices were preprocessed following standard procedures. Specifically, counts for each cell were normalized by rescaling to 10^4 counts per cell, followed by log transformation. Finally, the top 10^4 highly variable genes (HVGs) were selected. Cell-type annotations and two-dimensional SPRING embeddings were obtained directly from the annotations provided in Weinreb et al.

B.4.2 Mutual information estimation

We compute mutual information as a sample mean of pointwise mutual information estimates. To estimate pointwise mutual information in the representation space, we use the nonparametric nearest-neighbor estimator introduced by Kraskov et al. [65] with $k = 3$. This estimator has been shown to be effective in this setting: model latent spaces with tens of dimesions [44].

B.4.3 GDE modelling architecture

We use a Resnet-GNN architecture as the encoder and a CVAE as the generator. We use 64 latent dimensions, with 2 hidden layers of size 128.

B.5 Perturbation Prediction

B.5.1 Data preprocessing details

We use the pre-processed h5ad file from [49] including 10^4 genes. We compute the 10% most perturbative perturbations by examining the differentially expressed genes and then randomly select 20 of those perturbations to hold out. We hold these out across all cell types.

B.5.2 GDE modelling architecture

We use a Resnet-GNN architecture as the encoder and a CVAE as the generator, similar to the architecture in the lineage-tracing experiment, except we use a larger hidden state (1024) and a larger latent space (256). We include a perturbation prediction loss during training which trains a linear model with pairwise interactions between the control cell distribution embedding and the gene embedding to predict the difference in mean expression through a linear head. This structures the latent space for our downstream perturbation prediction task.

B.5.3 Perturbation Prediction

We fit a ridge regression to predict (1) the difference in mean expression and (2) the difference between the perturbed embedding and the control for each perturbation using GenePT gene embeddings [50] with cross-validation to perform grid search over λ. We then compute the predictions on the held-out perturbations and use a linear head to predict the mean expression from the latent difference. Finally we compute the R^2 score and the MSE.

B.6 Optical pooled screening dataset

B.6.1 Data preprocessing details

We use phenotyping images with assigned perturbation barcodes from Funk et al. [51]. We analyze only two of the measured channels: DAPI and GFP. Each image is a 64x64 bounding box surrounding a single cell (center-padded or center-cropped from the original bounding box as necessary). Image intensities are normalized to a minimum of -1 and a maximum of 1. Using the set of perturbative perturbations computed in [51] we randomly select 30% to holdout during training for evaluation.

B.6.2 GDE modelling architecture

For the encoder architecture, we extend our GNN approach to 2D convolutional layers, standard for image processing. For the generator we use a U-net architecture standard in diffusion for images, but upscaled in expressivity relative to our MNIST and Fashion-MNIST examples.

B.6.3 Perturbation Prediction

We find that empirically, our diffusion approach struggles to model the padded border of the cells. So, at inference time we condition on the border to generate our predictions. Using GenePT, we train a ridge regression with grid search (similar to App. B.5) to predict the perturbation distribution embeddings. We also construct a nearest neighbor model using the GenePT embeddings to sample the padding. We then condition on the padding and the predicted latent to sample a set of 1,000 cells from each heldout perturbation. We then compute the DAPI intensity and compare with the ground truth, computing the R^2 and the MSE.

B.7 Methylation atlas of human tissues

B.7.1 Simulating raw bisulfite-sequencing reads from methylation patterns

While sample-specific methylation patterns are published in [5], the raw sequencing reads are not public due to patient privacy considerations. Here, we instead use the published methylation patterns (in the form of `.pat` files) to simulate bisulfite sequencing reads. For each methylation site entry of the `.pat` file, we use `wgbstools`[66] to find the 100 preceding bases of the HG38 genome reference, and append to the CpG sequence. We omit all CpG sites with unknown methylation status. We subsample 10^7 sequencing reads per sample.

B.7.2 GDE modelling architecture

We use a 1D convolutional neural network as our encoder, with mean pooling at each layer (analogous to the fully connected GNN with an MLP, but using convolutional layers). For the generator, we use HyenaDNA [35]. We additionally include a linear classification head on top of the distribution embedding, co-trained with a cross-entropy loss.

B.8 GPRA

B.8.1 Data processing details

We collect all sequences in the Gal and Gly conditions from [52] and process them into 100 quantiles by measured expression, totaling 34 million sequences. We one-hot encode these sequences for ACTGN, and tokenize them using the HyenaDNA tokenizer. We break these sequences into 100 quantiles and hold out the top 5 quantiles during training. During training, we construct sets by selecting a "center" quantile and then randomly sampling from that quantile and the two adjacent quantiles.

B.8.2 GDE modelling architecture

We use the same architecture as in the methylation experiment (App. B.7).

B.8.3 Details for Fig. 8

We encode a random subsample of 130K sequences from each quantile in the Gal condition to construct the set embeddings (the larger dots). We then compute the PCA of these embeddings. We embed all the DNA sequences as sets of size one and project them to the PCA. For the histograms of the TFBS motifs we leverage the PWMs from [53]. We wrote a simple unidirectional motif scanning procedure in Torch to facilitate efficient scanning, and used a threshold of 5 to determine hits. We then sum over the motifs to derive the motif count per sequence, and then compute the histogram by plotting the distribution of these counts by quantile.

B.9 Spatiotemporal distribution of viral lineages

B.9.1 Data preprocessing details

We obtain all SARS-CoV2 spike sequences deposited up to April 2025 in GISAID [58]. We group sequences by submission month and lab of collection. We discard sequences with improperly formatted date fields. During tokenization, we truncate sequences to 1000 amino acids.

B.9.2 GDE modelling architecture

The encoder couples the ESM-50M [56] architecture coupled to a mean-pooled GNN, while the generator uses the Progen2-150M architecture [34] with prefix conditioning. We initialize (but do not freeze) the protein language models with their pretrained weights. We use a 128 dimensional latent space.

C Background

C.1 Frequentist, Bayesian, and Predictive Sufficiency

Sufficiency is a classical notion in statistics that formalizes when a statistic retains all information about a parameter or distribution. In this appendix, we distinguish three forms of sufficiency relevant to modern generative modeling and provide canonical examples.

C.1.1 Frequentist Sufficiency

Let $\{P_\theta : \theta \in \Theta\}$ be a parametric family of probability distributions on a sample space $\mathcal{X}$. A statistic $T(X_1, \ldots, X_n)$ is *frequentist sufficient* for θ if the conditional distribution of the data given T does not depend on θ:

$$P_\theta(X_1, \ldots, X_n \mid T(X_1, \ldots, X_n)) = \text{(independent of } \theta\text{)}.$$

Intuitively, the likelihood depends on the data only through T.

C.1.2 Bayesian Sufficiency

Given a prior $\pi(\theta)$ over the parameter space, a statistic T is *Bayesian sufficient* for θ if the posterior depends on the data only through T:

$$\pi(\theta \mid X_1, \ldots, X_n) = \pi(\theta \mid T(X_1, \ldots, X_n)).$$

Bayesian sufficiency holds if and only if T is a sufficient statistic in the sense that the posterior is conditionally independent of the data given T.

C.1.3 Predictive Sufficiency

A weaker notion, often relevant in nonparametric and distributional settings, is *predictive sufficiency*. A statistic T is predictive sufficient if the distribution of a new sample X_{new} given T is the same as given the full data:

$$\mathbb{P}(X_{\text{new}} \in B \mid T(X_1, \ldots, X_n)) = \mathbb{P}(X_{\text{new}} \in B \mid X_1, \ldots, X_n), \quad \forall B \in \mathcal{B}(\mathcal{X}).$$

This requires only that T contains enough information to match the predictive distribution of future data.

C.1.4 Implications and Comparisons

There is a strict hierarchy among these definitions:

$$\text{Frequentist sufficiency} \Rightarrow \text{Bayesian sufficiency} \Rightarrow \text{Predictive sufficiency}.$$

The first implication follows from the factorization of the likelihood, and the second follows because the posterior predictive is a marginal of the posterior. However, the reverse implications do not hold in general, especially in infinite-dimensional or nonparametric models. In particular, predictive sufficiency may hold in settings where no finite-dimensional parameter exists.

C.1.5 Examples

Example 1 (Gaussian Mean)**.** Let $X_1, \ldots, X_n \sim \mathcal{N}(\mu, \sigma^2)$ with known σ^2. Then the sample mean $\bar{X}_n$ is sufficient for μ in all three senses: frequentist, Bayesian, and predictive. The likelihood, posterior, and predictive distributions all depend on the data only through $\bar{X}_n$.

Example 2 (Gaussian Mixture Model)**.** Let $X_1, \ldots, X_n \sim P$ where P is a finite mixture of Gaussians:

$$P = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(\mu_k, \Sigma_k).$$

The sufficient statistics for this model (under known K) are:

- the soft assignment (responsibility) weights for each component,
- the empirical means and covariances of points assigned to each component,
- the mixture proportions.

These are sufficient in both the frequentist and Bayesian senses. In many applications, they are approximated via the Expectation-Maximization algorithm or variational inference.

Example 3 (Uniform$(0, \theta)$)**.** Let $X_1, \ldots, X_n \sim \text{Unif}(0, \theta)$. Then the sample maximum

$$T_n = \max\{X_1, \ldots, X_n\}$$

is the minimal sufficient statistic for θ in both the frequentist and Bayesian senses. It also suffices for prediction of future samples, since the predictive distribution under θ is uniform on $[0, \theta]$, and T_n provides all information about θ.

C.1.6 Nonparametric Extensions

In the nonparametric regime where P is not indexed by a finite-dimensional parameter, predictive sufficiency remains well-defined. For instance, the empirical measure $P_n = \frac{1}{n}\sum_{i=1}^{n} \delta_{X_i}$ is always predictive sufficient under exchangeable models. In this setting, stronger forms of sufficiency may not exist, but predictive sufficiency still supports meaningful generative modeling.

C.2 Otto's Geometry and Statistical Submanifolds

This appendix summarizes the formal Riemannian structure of the Wasserstein space $\mathcal{P}_2(\mathcal{X})$ introduced by Otto [3], and defines statistical manifolds as submanifolds equipped with a geometry induced from this structure. This provides the mathematical foundation for interpreting generative distributional encoders (GDEs) as learning smooth geometric embeddings of constrained distributional families.

C.2.1 Wasserstein Space and the Benamou–Brenier Formulation

Let $\mathcal{X} \subseteq \mathbb{R}^d$ be a domain, and let $\mathcal{P}_2(\mathcal{X})$ denote the space of Borel probability measures on $\mathcal{X}$ with finite second moment. The 2-Wasserstein distance between two measures $\mu_0, \mu_1 \in \mathcal{P}_2(\mathcal{X})$ is defined by the optimal transport problem

$$W_2^2(\mu_0, \mu_1) := \inf_{\gamma \in \Pi(\mu_0, \mu_1)} \int_{\mathcal{X}\times\mathcal{X}} \|x - y\|^2 \, d\gamma(x, y),$$

where $\Pi(\mu_0, \mu_1)$ denotes the set of couplings with marginals μ_0 and μ_1. An equivalent dynamic formulation, due to Benamou and Brenier [67], expresses the Wasserstein distance as a variational problem over time-dependent flows:

$$W_2^2(\mu_0, \mu_1) = \inf_{\substack{(\mu_t, v_t) \\ \partial_t \mu_t + \nabla\cdot(\mu_t v_t) = 0}} \int_0^1 \int_{\mathcal{X}} \|v_t(x)\|^2 \, d\mu_t(x)\, dt,$$

subject to boundary conditions μ_0, μ_1 and the continuity equation, which ensures mass conservation.

C.2.2 Otto's Riemannian Structure

Otto observed that the Benamou–Brenier problem defines a formal Riemannian structure on $\mathcal{P}_2(\mathcal{X})$, where the tangent space at a measure μ consists of velocity fields v such that the continuity equation describes admissible perturbations. The inner product between two such velocity fields $v_1, v_2 \in T_\mu \mathcal{P}_2$ is defined as

$$\langle v_1, v_2 \rangle_{T_\mu \mathcal{P}_2} := \int_{\mathcal{X}} v_1(x) \cdot v_2(x) \, d\mu(x).$$

This makes $\mathcal{P}_2(\mathcal{X})$ a formal infinite-dimensional Riemannian manifold, and Wasserstein geodesics become curves of minimal kinetic energy under this metric.

For absolutely continuous μ_0, the optimal transport map $T : \mathcal{X} \to \mathcal{X}$ from μ_0 to μ_1 induces a geodesic $(\mu_t)_{t\in[0,1]}$ by pushing μ_0 along linear interpolations:

$$\mu_t = ((1 - t)\mathrm{id} + tT)_{\#}\mu_0.$$

These displacement interpolants travel at constant speed under the W_2 metric and solve the geodesic equation associated with the Otto metric.

C.2.3 Statistical Manifolds as Submanifolds of Wasserstein Space

Let Q be a probability distribution over $\mathcal{P}_2(\mathcal{X})$, and define the *statistical manifold* as the support of Q:

$$\mathcal{M} := \mathrm{supp}(Q) \subset \mathcal{P}_2(\mathcal{X}).$$

We endow $\mathcal{M}$ with the *induced Riemannian structure* from $\mathcal{P}_2(\mathcal{X})$, by restricting the Otto metric to velocity fields that remain tangent to $\mathcal{M}$. That is, $T_\mu\mathcal{M} \subset T_\mu\mathcal{P}_2$ is a subspace of velocity fields preserving membership in $\mathcal{M}$, and the inner product is

$$\langle v_1, v_2 \rangle_{T_\mu\mathcal{M}} := \int_{\mathcal{X}} v_1(x) \cdot v_2(x) \, d\mu(x), \quad \text{for } v_1, v_2 \in T_\mu\mathcal{M}.$$

This leads to a constrained transport problem defining geodesics within $\mathcal{M}$:

$$W_{2,\mathcal{M}}^2(\mu_0, \mu_1) := \inf_{\substack{(\mu_t, v_t) \\ \mu_t \in \mathcal{M},\, \partial_t \mu_t + \nabla\cdot(\mu_t v_t) = 0}} \int_0^1 \int_{\mathcal{X}} \|v_t(x)\|^2 \, d\mu_t(x)\, dt.$$

This is simply the Wasserstein variational problem, but restricted to paths that lie within the submanifold $\mathcal{M}$. It defines the geometry relevant to learning distributions drawn from Q.

C.2.4 Examples and Application to GDEs

Typical examples of such submanifolds include:

- Gaussian families $\mathcal{N}(\mu, \Sigma)$, where geodesics can be computed in closed form;
- Mixture models with a fixed number of components, see [38];
- general parametric families.

In this work, we treat the statistical manifold $\mathcal{M} = \mathrm{supp}(Q)$ as the set of data-generating distributions and interpret GDEs as learning a smooth embedding of this submanifold into Euclidean space. While GDEs do not explicitly minimize Wasserstein distances, we observe empirically that their learned latent geometries often approximate the structure of $W_{2,\mathcal{M}}$, suggesting that they act as approximate isometric embeddings of this constrained transport geometry.

D Theory

Throughout, let $(\mathcal{X}, \mathcal{B})$ be a Polish space. Let $P \in \mathcal{P}(\mathcal{X})$ denote a probability law on $\mathcal{X}$. Given $m \in \mathbb{N}$, let $S_m = (X_1, \ldots, X_m)$ be an i.i.d. sample from P, and let $P_m = \frac{1}{m}\sum_{i=1}^m \delta_{X_i}$ denote the empirical measure.

We use P_1, P_2 to denote two (possibly distinct) probability laws on $\mathcal{X}$, and S_1, S_2 for independent samples from P_1, P_2 respectively.

For signed measures ν, μ on $(\mathcal{X}, \mathcal{B})$ define

$$d_{\mathrm{BL}}(\nu, \mu) \;:=\; \sup_{\substack{f:\mathcal{X}\to[-1,1]\\ \mathrm{Lip}(f)\le 1}} \Big|\int f\, d(\nu - \mu)\Big|.$$

We use $\|\cdot\|_{\mathrm{BL}}$ for the corresponding norm $\|\nu\|_{\mathrm{BL}} := d_{\mathrm{BL}}(\nu, 0)$ and recall that $d_{\mathrm{TV}}(\nu, \mu) \le d_{\mathrm{BL}}(\nu, \mu)$.

All random variables are defined on a common probability space unless otherwise specified.

D.1 Necessity of Distributional Invariance

Motivation Our goal is to design encoder architectures that flexibly model unknown data distributions while guaranteeing consistent generation of the underlying law as sample size grows. Since the true distribution P is not known in advance, the encoder must be constructed to generalize across all possible P, without leaking spurious information tied to the specific realization or sample size. If the encoder depends on sample-level artifacts—such as ordering, multiplicity, or the raw sample size—it may encode features that a generator can exploit, breaking the guarantee that

$$\mathcal{G}(\mathcal{E}(S_m)) \xrightarrow{d} P \quad \text{as } m \to \infty, \quad S_m \sim P^{\otimes m}.$$

This risk arises even under either permutation or proportional invariance on their own: both permit dependencies that vanish only in expectation and are insufficient to ensure correct extrapolation with increasing m. For example, encoders based on unnormalized sum aggregations (e.g., DeepSets) will vary with m even when the empirical distribution is unchanged, leading to divergence at inference time.

To formalize this constraint, we draw on Blackwell's theory of experiments, which provides a general framework for comparing the informativeness of statistical summaries. We adopt his game-theoretic perspective—viewing the encoder as a player that chooses an experiment, and the generator as an adversary that exploits the information it receives—and use this to characterize the minimal structural conditions an encoder must satisfy to guarantee asymptotic consistency. In particular, we show that dependence on the empirical distribution is necessary and sufficient: it is the least informative summary that still retains all information required to identify the law, so the generator cannot learn any spurious information that will fail to extrapolate at inference time.

Setting We consider the following general setting: Let $(\mathcal{X}, \mathcal{B})$ be a Polish space. We are interested in measurable summaries of infinite i.i.d. sequences $S \sim P^\infty$, where $P \in \mathcal{P}(\mathcal{X})$ is an unknown probability law. The goal is to characterize the minimal invariance properties required for encoders to guarantee consistent recovery of P from finite samples.

Definition 1 (Distributional Invariance)**.** A function $\mathcal{E} : \mathcal{X}^m \to \mathcal{Z}$ is *distributionally invariant* if for any $S_m \in \mathcal{X}^m$, $\mathcal{E}(S_m)$ depends only on the empirical measure P_m of S_m; that is, for any permutation π of $\{1, \ldots, m\}$ and any S_m, $\mathcal{E}(S_m) = \mathcal{E}(S_m^\pi)$, and ϕ is invariant to proportional duplications of the sample.

Definition 2 (Asymptotic Distributional Invariance)**.** A sequence of functions $\mathcal{E}_m : \mathcal{X}^m \to \mathcal{Z}$ is *asymptotically distributionally invariant* if for every $P \in \mathcal{P}(\mathcal{X})$, there exists a sequence of measurable functions $\phi_m : \mathcal{P}_m(\mathcal{X}) \to \mathcal{Z}$ such that

$$\mathbb{P}_{S_m \sim P^{\otimes m}}\big(\mathcal{E}_m(S_m) = \phi_m(P_m)\big) \to 1 \quad \text{as } m \to \infty,$$

where P_m is the empirical measure of S_m.

Lemma 1 (Strong Law of Large Numbers for Empirical Measures)**.** *Let $P \in \mathcal{P}(\mathcal{X})$ and $S_m = (X_1, \ldots, X_m)$ be i.i.d. samples from P. Then the empirical measure $P_m = \frac{1}{m}\sum_{i=1}^m \delta_{X_i}$ converges almost surely to P in the weak topology as $m \to \infty$.*

Lemma 2 (Wainwright's Rademacher–tail bound [68, Thm. 4.10])**.** *Let $\mathcal{F}$ be a class of measurable functions $f : \mathcal{X} \to \mathbb{R}$ that is b-*uniformly bounded*, i.e. $\|f\|_\infty \le b$ for all $f \in \mathcal{F}$. For any integer $n \ge 1$ and any $\delta > 0$,*

$$\big\|P_n - P\big\|_{\mathcal{F}} \;\le\; 2\,\mathcal{R}_n(\mathcal{F}) + \delta \quad \textit{with probability at least} \quad 1 - \exp\Big(-\tfrac{n\delta^2}{2b^2}\Big),$$

where $\mathcal{R}_n(\mathcal{F})$ is the (deterministic) Rademacher complexity defined in [68, Eq. 4.13].

To formalize what can go wrong, we introduce an adversarial two-player game, adapted from the decision-theoretic framework introduced by [69]. This game reveals the informational limits of summary statistics for distinguishing distributions.

1. *Player 1* selects a measurable summary rule $T : \mathcal{X}^{\mathbb{N}} \to \mathcal{T}$ before seeing any data.
2. *Player 2* observes T and chooses two probability laws $P_1, P_2 \in \mathcal{P}(\mathcal{X})$.
3. *Nature* draws two independent infinite i.i.d. sequences $S_1 \sim P_1^\infty$, $S_2 \sim P_2^\infty$.

The induced decision problem is whether the summary T is consistent with $P_1 = P_2$ or not. The payoff structure is:

	$T(S_1) = T(S_2)$	$T(S_1) \neq T(S_2)$
$P_1 = P_2$	$(1, 0)$	$(0, 1)$
$P_1 \neq P_2$	$(0, 1)$	$(1, 0)$

Player 1 aims to minimize both types of errors: introducing spurious distinctions when $P_1 = P_2$, and failing to distinguish when $P_1 \neq P_2$. In Blackwell's terms, the goal is to find a summary that is as informative as possible for this class of binary decision problems.

Definition 3 (Asymptotically Blackwell–optimal summary)**.** Let $(\mathcal{T}, d_{\mathcal{T}})$ be a separable metric space and let $T_n : \mathcal{X}^n \to \mathcal{T}$ be measurable. Fix any deterministic sequence $\varepsilon_n \downarrow 0$. We say that (T_n) is *asymptotically Blackwell–optimal* if, for every pair of probability laws $P_1, P_2 \in \mathcal{P}(\mathcal{X})$,

$$\text{(i)} \quad P_1 = P_2 \Longrightarrow \mathbb{P}_{P_1^\infty}\Big[d_{\mathcal{T}}\big(T_n(S_1), T_n(S_2)\big) > \varepsilon_n\Big] \xrightarrow[n\to\infty]{} 0, \tag{2}$$

$$\text{(ii)} \quad P_1 \neq P_2 \Longrightarrow \mathbb{P}_{P_1^\infty \times P_2^\infty}\Big[d_{\mathcal{T}}\big(T_n(S_1), T_n(S_2)\big) \leq \varepsilon_n\Big] \xrightarrow[n\to\infty]{} 0. \tag{3}$$

Main Result

Theorem 1 (Empirical distribution characterises optimal asymptotic summaries)**.** *Let $P_n(S) = \frac{1}{n}\sum_{i=1}^n \delta_{S_i}$ be the empirical distribution of the first n samples of $S \sim P^\infty$. For measurable summaries $T_n : \mathcal{X}^n \to \mathcal{T}$:*

*(i) (**Blackwell optimality**) the empirical distribution $T_n^{\text{emp}}(S) = P_n(S)$ is asymptotically optimal in the game of Definition 3.*

*(ii) (**Asymptotic information equivalence**) If (T_n) is asymptotically optimal, there exist measurable maps $f_n : \mathcal{P}(\mathcal{X}) \to \mathcal{T}$ and $g_n : \mathcal{T} \to \mathcal{P}(\mathcal{X})$ such that for every $P \in \mathcal{P}(\mathcal{X})$*

$$\mathbb{P}_{P^{\otimes n}}\big(T_n(S) = f_n(P_n(S))\big) \xrightarrow[n\to\infty]{} 1, \qquad \mathbb{P}_{P^{\otimes n}}\big(P_n(S) = g_n(T_n(S))\big) \xrightarrow[n\to\infty]{} 1.$$

In particular, T_n and P_n are asymptotically Blackwell-equivalent in probability.

We can immediately conclude the implications for encoders here:

Corollary 1 (Necessity of asymptotic distributional sufficiency)**.** Suppose we design an encoder $\mathcal{E}$ and decoder $\mathcal{G}$ with the goal that, for any unknown distribution $P \in \mathcal{P}(\mathcal{X})$,

$$\mathcal{G}(\mathcal{E}(S_m)) \xrightarrow{d} P \quad \text{as } m \to \infty, \quad S_m \sim P^{\otimes m}.$$

Then, in order for this convergence to hold for any P, the encoder architecture must satisfy two conditions:

(i) **Asymptotic distributional invariance**: $\mathcal{E}_m(S_m)$ must (eventually) depend only on the empirical distribution $P_m = \frac{1}{m}\sum_{j=1}^m \delta_{x_j}$—that is, for every P, $\mathbb{P}_{P^{\otimes m}}\big(\mathcal{E}_m(S_m) = \phi_m(P_m)\big) \to 1$.

(ii) **Distributional expressivity**: the class of encoder functions must be rich enough to approximate any measurable function of P_m.

These are constraints on the encoder *architecture*, not on the learned function after training. If $\mathcal{E}_m$ encodes any features not measurable with respect to P_m—such as sample order, indexing artifacts, or features sensitive to repeated observations—the generator can exploit these to fit P incorrectly, breaking consistency. Either permutation or proportional invariance alone are not sufficient: only sufficiency with respect to the empirical distribution rules out such failure modes.

Proof

Proof. **Step 1: Soundness.** The class $\mathcal{F}_{\text{BL}}$ is 1-uniformly bounded, and its empirical Rademacher complexity satisfies $\mathfrak{R}_n(\mathcal{F}_{\text{BL}}) = O(n^{-1/2})$. Equip $\mathcal{T}$ with d_{BL} and set $\varepsilon_n = n^{-1/2}$.

By the triangle inequality,

$$d_{\text{BL}}\big(P_n(S_1), P_n(S_2)\big) \leq d_{\text{BL}}\big(P_n(S_1), P\big) + d_{\text{BL}}\big(P_n(S_2), P\big).$$

Applying Lemma 2 [68] to each term with tolerance $\varepsilon_n/2$ and union-bounding yields

$$\mathbb{P}\Big[d_{\mathrm{BL}}\big(P_n(S_1), P_n(S_2)\big) > \varepsilon_n\Big] \le 2\,\exp\!\big(-\tfrac{n\varepsilon_n^2}{8}\big).$$

Choosing $\varepsilon_n = n^{-1/4}$ therefore fulfils (2).

If $P_1 \neq P_2$, the strong law gives $d_{\mathrm{BL}}\big(P_n(S_i), P_i\big) \xrightarrow{a.s.} 0$, hence for large n the event $d_{\mathrm{BL}}\big(P_n(S_1), P_n(S_2)\big) \le \varepsilon_n$ is impossible, establishing (3).

Step 2: T_n is a function of P_n *with high probability*. Let $\eta_n \downarrow 0$ be an arbitrary deterministic sequence. We will show

$$\mathbb{P}_{P^{\otimes n}}\big[T_n(S) \;=\; f_n(P_n(S))\big] \;\ge\; 1-\eta_n \quad \text{for all large } n,$$

for some measurable $f_n\colon \mathcal{P}(\mathcal{X}) \to \mathcal{T}$. That is, T_n differs from a measurable function of the empirical distribution with probability $o(1)$, which suffices for the asymptotic game.

Now fix $\eta > 0$ and define

$$\mathcal{I}_\eta := \Big\{\, n \ge 1 :\; \exists\, P \in \mathcal{P}(\mathcal{X}) \text{ s.t. } \mathbb{P}_{P^{\otimes n}}\big[T_n(S) \text{ is not } \sigma(P_n)\text{-measurable}\big] \;>\; \eta \Big\}.$$

If $\mathcal{I}_\eta$ were infinite for some positive η, we would construct a single distribution $P_\dagger$ to be any accumulation point of the sequence of counter-example measures forcing

$$\mathbb{P}\big[T_n(S_1) \neq T_n(S_2)\big] \;\ge\; \tfrac{\eta^2}{2} \quad \text{for all } n \in \mathcal{I}_\eta,$$

contradicting optimality condition (2).

Since asymptotic optimality holds, every $\eta > 0$ gives a finite $\mathcal{I}_\eta$, so we can choose $N(\eta)$ such that

$$\mathbb{P}_{P^{\otimes n}}\big[T_n(S) \text{ is } \sigma(P_n)\text{-measurable}\big] \;\ge\; 1-\eta \quad \text{for all } n \ge N(\eta) \text{ and every } P.$$

Taking $\eta = \eta_n$ and letting f_n be any measurable selector on the high-probability event (where T_n *is* $\sigma(P_n)$-measurable) proves the claim that T_n is *eventually* a function of P_n with probability $1-\eta_n$.

Step 3: Recoverability of P_n from T_n. If f_n collapses two distinct empirical measures $m \neq m'$, take $P_1 = m$, $P_2 = m'$. Then $d_{\mathrm{BL}}(m, m') > 0$ while $T_n(S_1) = T_n(S_2)$ a.s., violating (3). Hence there exists a measurable $g_n : \mathcal{T} \to \mathcal{P}(\mathcal{X})$ with $P_n = g_n\big(T_n\big)$ a.s. □

Remark. *The argument above shows that any asymptotically optimal summary T_n is both a measurable function of P_n and sufficient to recover P_n almost surely. In Blackwell's terminology, this means T_n and P_n are asymptotically equivalent experiments: they contain exactly the same information for distinguishing distributions from i.i.d. samples. In future work it would be interesting to consider how to design architectures with similar properties under different forms of dependence.*

D.2 A Complete Large-m Analysis of the Plug-in Loss

Motivation We analyze the statistical properties of the plug-in loss used to train distributional encoders and generators. Our goal is to understand the asymptotic behavior of this loss as the sample size grows, and to establish conditions under which the learned generator recovers the true data distribution. This analysis provides a principled foundation for the training objectives used in our framework.

Setting First we establish some notation and definitions.

Definition 4 (Hadamard differentiability)**.** A map $T : \mathcal{D} \to \mathcal{Y}$ between normed spaces is *Hadamard differentiable* at $x \in \mathcal{D}$ if there exists a continuous linear operator DT_x such that for every sequence $h_t \to h$ in $\mathcal{D}$ and $t \downarrow 0$, $\frac{T(x+th_t)-T(x)}{t} \longrightarrow DT_x[h]$.

Definition 5 (Fréchet differentiability)**.** Let $T : \mathcal{D} \to \mathcal{Y}$ be a map between normed vector spaces. T is *Fréchet differentiable* at $x \in \mathcal{D}$ if there exists a bounded linear operator $A : \mathcal{D} \to \mathcal{E}$ such that

$$\lim_{\|h\|_D \to 0} \frac{\|T(x+h) - T(x) - A(h)\|_{\mathbb{E}}}{\|h\|_{\mathcal{D}}} = 0.$$

The operator A is called the Fréchet derivative of T at x.

We work in the following general setting:

Assumption 1 (Data and Empirical Measure)**.** *$(\mathcal{X}, \mathcal{B})$ is a Polish space; $P \in \mathcal{P}(\mathcal{X})$ is the true data law. Observations $S_m = (X_1, \ldots, X_m)$ are i.i.d. P. The empirical measure is $P_m = \frac{1}{m}\sum_{i=1}^m \delta_{X_i}$.*

Assumption 2 (Encoder regularity)**.** *For each probability law $P \in \mathcal{P}(\mathcal{X})$ the encoder $\phi : \mathcal{P}(\mathcal{X}) \to \mathbb{R}^d$ satisfies*

(i) ***Distributional invariance:*** *$\mathcal{E}_m(S_m) = \phi(P_m)$ depends on the sample only via its empirical measure.*

(ii) ***Pathwise (Hadamard) differentiability:*** *ϕ is pathwise differentiable at P and its canonical gradient*[1] *$\psi_P : \mathcal{X} \to \mathbb{R}^d$ belongs to $L^2(P)$.*

(iii) ***Asymptotic linearity (AL):*** *the estimator obeys*

$$\sqrt{m}\left\{\phi(P_m) - \phi(P)\right\} \;=\; \frac{1}{\sqrt{m}} \sum_{i=1}^{m} \psi_P(X_i) + o_p(1).$$

where $\mathbb{E}_{X\sim P}[\psi_P(X)] = 0$, so ψ_P.

Under these conditions

$$\sqrt{m}\left\{\phi(P_m) - \phi(P)\right\} \;\xRightarrow{d}\; \mathcal{N}\big(0, \Sigma_\phi\big), \quad \Sigma_\phi := \mathrm{Var}_{X\sim P}\big[\psi_P(X)\big].$$

Assumption 3 (Generator)**.** *$\mathcal{G} : \mathbb{R}^d \to \mathcal{P}(\mathcal{X})$ is Fréchet differentiable on a neighbourhood of $\mu := \phi(P)$ and its derivative factors through $L^2(P)$, i.e.*

$$D_\mu \mathcal{G} = \; T \circ A, \quad \textit{where} \;\; A : \mathbb{R}^d \to L_0^2(P),\; T : L_0^2(P) \to \mathcal{M}_0(\mathcal{X})$$

are bounded linear maps and $L_0^2(P)$ denotes zero-mean square-integrable functions.

Assumption 4 (Divergence)**.** *The discrepancy $\mathfrak{d} : \mathcal{P}(\mathcal{X})^2 \to \mathbb{R}_+$ satisfies*

(i) *(**Hadamard differentiability**) the map $Q \mapsto \mathfrak{d}(P, Q)$ is Hadamard differentiable at $Q_0 = \mathcal{G}(\mu)$ tangentially to $\mathcal{M}_0(\mathcal{X})$, with continuous linear derivative $D_2\mathfrak{d}(P, Q_0) : \mathcal{M}_0(\mathcal{X}) \to \mathbb{R}$;*

(ii) *(**Separating property**) $\mathfrak{d}(P, Q) = 0 \implies P = Q$;*

(iii) *(**Weak-continuity**) if $\mathfrak{d}(Q_n, Q) \to 0$ then $Q_n \Rightarrow Q$.*

We define a general loss function $\ell(P, Q)$, where P is the true distribution and Q is a model output (e.g., a divergence such as KL or Wasserstein).

The *plug-in loss* is

$$\widehat{\ell}_m := \ell(P, \mathcal{G}(\phi(P_m)))$$

and the *population loss* is

$$\ell^* := \ell(P, \mathcal{G}(\phi(P)))$$

where P_m is the empirical measure of the sample, ϕ is the encoder, and $\mathcal{G}$ is the generator.

Lemma 3 (Donsker's Theorem for Empirical Measures)**.** *Let $(\mathcal{X}, \mathcal{B})$ be a Polish space, and let $P \in \mathcal{P}(\mathcal{X})$. Let $\{X_i\}_{i=1}^m$ be i.i.d. samples from P, and let P_m be the empirical measure:*

$$P_m = \frac{1}{m} \sum_{i=1}^{m} \delta_{X_i}.$$

Define the empirical process:

$$\sqrt{m}(P_m - P).$$

Then, viewed as an element of the Banach space $\ell^\infty(\mathcal{F})$ of bounded real-valued functions on $\mathcal{F}$, where $\mathcal{F}$ is any P-Donsker class of measurable functions, we have:

$$\sqrt{m}(P_m - P) \xrightarrow{d} \mathbb{G}_P,$$

where $\mathbb{G}_P$ is a P-Brownian bridge, a mean-zero tight Gaussian process indexed by $\mathcal{F}$ with covariance function

$$\mathrm{Cov}\big(\mathbb{G}_P(f), \mathbb{G}_P(g)\big) = \mathrm{Cov}_{X\sim P}\big(f(X), g(X)\big).$$

Lemma 4 (Functional Delta Method, [70, Thm. 3.9.4])**.** *Let $(\mathbb{D}, \|\cdot\|_{\mathbb{D}})$ and $(\mathbb{E}, \|\cdot\|_{\mathbb{E}})$ be normed vector spaces. Let $T : \mathbb{D} \to \mathbb{E}$ be a map that is Hadamard differentiable at a point $z \in \mathbb{D}$ tangentially to a subset $\mathbb{D}_0 \subseteq \mathbb{D}$, with continuous linear derivative denoted $DT_z : \mathbb{D}_0 \to \mathbb{E}$.*

Suppose:

(a) There exist random elements Z_m taking values in $\mathbb{D}$ such that:

$$\sqrt{m}(Z_m - z) \xrightarrow{d} Z$$

for some tight limit Z taking values in $\mathbb{D}_0$.

(b) Z is tight and Borel measurable.

Then:

$$\sqrt{m}\big(T(Z_m) - T(z)\big) \xrightarrow{d} DT_z(Z),$$

where $DT_z(Z)$ is a random element of $\mathbb{E}$.

In particular, if Z is Gaussian in $\mathbb{D}_0$ and DT_z is continuous and linear, then $DT_z(Z)$ is Gaussian in $\mathbb{E}$.

[1] In the semiparametric sense of 70, i.e. the unique influence function representing the functional derivative along $\mathcal{M}_0(\mathcal{X})$.

Main Result

Theorem 2 (Large-m behaviour of the plug-in loss)**.** *Assume 1, 2, 3, and 4. Let* $\mu := \phi(P)$ *and* $\widehat{\ell}_m := \mathfrak{d}\big(P,\, \mathcal{G}(\phi(P_m))\big)$.

We now combine the regularity assumptions with empirical-process theory to quantify the estimation error of the plug-in loss

(a) ***Asymptotic normality of the Encoder.***

$$\sqrt{m}\big\{\phi(P_m) - \phi(P)\big\} \;\overset{d}{\Rightarrow}\; \mathcal{N}\big(0, \Sigma_\phi\big), \qquad \Sigma_\phi := \mathrm{Var}_{X\sim P}\big[\psi_P(X)\big].$$

(b) ***Unbiasedness of the loss.*** $\mathbb{E}[\widehat{\ell}_m] = \ell^* + O(m^{-1}), \quad \ell^* := \mathfrak{d}\big(P, \mathcal{G}(\mu)\big)$.

(c) ***Asymptotic normality of the loss.***

$$\sqrt{m}\,(\widehat{\ell}_m - \ell^*) \;\overset{d}{\Rightarrow}\; \mathcal{N}\big(0, \sigma^2\big), \quad \sigma^2 = \nabla_\mu \ell^\top \Sigma_\phi \nabla_\mu \ell, \qquad \ell(\theta) := \mathfrak{d}\big(P, \mathcal{G}(\theta)\big).$$

(d) ***Sufficiency of the loss.*** *If* $(\phi^\star, \mathcal{G}^\star)$ *minimises* $P \mapsto \mathfrak{d}\big(P, \mathcal{G}(\phi(P))\big)$*, then* $\mathcal{G}^\star\big(\phi^\star(P_m)\big) \Rightarrow P$ *in probability as* $m \to \infty$.

Proof. **Step 1: Asymptotic Normality of the encoder (a).** Assumption 2(iii) (asymptotic linearity) gives

$$\sqrt{m}\big\{\phi(P_m) - \phi(P)\big\} = \frac{1}{\sqrt{m}} \sum_{i=1}^{m} \psi_P(X_i) + o_p(1),$$

and the classical multivariate CLT yields the stated convergence.

Let $\Delta_m := \phi(P_m) - \mu$ so that, by (a), $\sqrt{m}\,\Delta_m \overset{d}{\Rightarrow} \mathcal{N}(0, \Sigma_\phi)$.

Step 2 (unbiasedness). Because $\mathbb{E}[\Delta_m] = 0$ by Assumption 2(iii) and ℓ is twice continuously differentiable in a neighbourhood of μ, a Taylor expansion gives $\mathbb{E}[\widehat{\ell}_m - \ell^*] = \frac{1}{2}\,\mathrm{tr}\big\{\nabla^2_\mu \ell\ \mathrm{Var}\,\Delta_m\big\} + O(m^{-1}) = O(m^{-1})$.

Step 3: Asymptotic Normality of the loss (c). Apply the functional delta method twice:

1. to the generator $\mathcal{G} : \mathbb{R}^d \to \mathcal{P}(\mathcal{X})$, using Fréchet differentiability and the fact that $\sqrt{m}\,\Delta_m$ is tight, obtaining

$$\mathcal{G}(\mu + \Delta_m) = \mathcal{G}(\mu) + D_\mu \mathcal{G}[\Delta_m] + o_p(m^{-1/2});$$

2. to the divergence $Q \mapsto \mathfrak{d}(P, Q)$ at $Q_0 := \mathcal{G}(\mu)$, with linear derivative $\partial_2 \mathfrak{d}(P, Q_0)[\,\cdot\,]$.

Combining the two expansions produces the linear functional of $\sqrt{m}\,\Delta_m$ displayed in (c) and hence the Gaussian limit with variance $\sigma^2 = \nabla_\mu \ell^\top \Sigma_\phi \nabla_\mu \ell$.

Step 4: Sufficiency of the loss (d). If $(\phi^\star, \mathcal{G}^\star)$ is optimal, then $\mathfrak{d}\big(P, \mathcal{G}^\star(\phi^\star(P))\big) = 0$, so $\mathcal{G}^\star(\phi^\star(P)) = P$ by Assumption 4(ii). Repeating the expansion from (c) with $(\phi^\star, \mathcal{G}^\star)$ shows $\mathfrak{d}\big(P, \mathcal{G}^\star(\phi^\star(P_m))\big) = O_p(m^{-1/2})$, and consistency for weak convergence then implies $\mathcal{G}^\star(\phi^\star(P_m)) \Rightarrow P$. □

Encoders: examples, counter-examples, and CLTs The only encoder requirement entering Theorem 2 is Assumption 2. We now show that it is satisfied by a large family of permutation-invariant architectures built from *asymptotically-linear* (M/Z) *poolers*.

Generic K-layer pool–concat encoder Fix $K \in \mathbb{N}$. Given a set of samples $S_m = \{x_1, \ldots, x_m\}$ define recursively

$$h_i^{(0)} \;=\; \psi(x_i), \quad \bar{h}^{(\ell)} \;=\; T^{(\ell)}\big(h_{1:m}^{(\ell-1)}\big), \quad h_i^{(\ell)} \;=\; \mathrm{MLP}_\ell\big(h_i^{(\ell-1)}, \bar{h}^{(\ell)}\big), \qquad \ell = 1, \ldots, K,$$

and set the encoder output to be another pooler $\phi(P_m) = T^{(K+1)}\big(h_{1:m}^{(K)}\big)$.

We call a permutation-invariant functional an *asymptotically linear (AL) pooler* if it is root-m consistent and admits an influence-function expansion; precise details follow.

Definition 6 (Asymptotically-linear pooler)**.** A symmetric map $T : \mathcal{X}^m \to \mathbb{R}^d$ is an *AL pooler* at law P if there exists $\psi_P \in L^2(P)$ such that

$$\sqrt{m}\,\big\{T(X_{1:m}) - \phi(P)\big\} \;=\; \frac{1}{\sqrt{m}} \sum_{i=1}^{m} \psi_P(X_i) \;+\; o_p(1).$$

Examples: mean, median, trimmed mean, Huber M-estimator, M-quantiles, studentised Z-estimators with finite variance.

Proposition 4 (CLT for K-layer AL pool–concat encoders)**.** *Assume*

(i) each $T^{(\ell)}$ ($\ell = 1, \ldots, K+1$) is an AL pooler at P;

(ii) each MLP_ℓ and the base feature map $\psi : \mathcal{X} \to \mathbb{R}^p$ are C^2 with bounded derivatives, and weights are frozen as $m \to \infty$.

Then the encoder ϕ is distributionally invariant, pathwise differentiable, and satisfies the CLT of Assumption 2 with

$$\sqrt{m}\{\phi(P_m) - \phi(P)\} \overset{d}{\Rightarrow} \mathcal{N}\big(0, \Sigma_\phi\big),$$

for some finite covariance matrix Σ_ϕ.

Sketch. The composition of Lipschitz maps (MLP_ℓ) with AL poolers is Hadamard differentiable by repeated application of the delta method (iterating Lemma 4, [70]). Plugging each AL expansion into the chain yields an overall AL expansion whose leading empirical-process term is $m^{-1/2} \sum_{i=1}^m \psi_P^\star(X_i)$ for some $L^2(P)$ function $\psi_P^\star$, giving the CLT. □

Instantiation to common architectures

Corollary 2 (DeepSets, Transformers without positional enc.)**.** Encoder architectures of either type below satisfy Assumption 2 and Proposition 4:

(a) *DeepSets / fully-connected GNN with global mean:* $T^{(\ell)}$ and $T^{(K+1)}$ are sample means;

(b) *Self-attention block with mean head:* $T^{(\ell)}$ are sample means; MLP_ℓ includes the softmax-attention update.

Why max-pooling fails The max functional $T_{\max}(x_{1:m}) = \max_i x_i$ is *not* Hadamard differentiable at continuous laws: Its influence function is identically 0 whenever the maximum is attained at a unique point and undefined when it is not. Consequently, the centered statistic $m^{1/2}\{T_{\max}(P_m) - T_{\max}(P)\}$ has a *non-Gaussian* limit—the Gumbel extreme-value law—so Assumption 2(iii) fails. Using max-pooling inside a deep encoder, therefore breaks the loss-CLT of Theorem 2. (Softmax pooling with temperature $\tau > 0$, on the other hand, is smooth and becomes a valid AL pooler.)

The table below summarises the status of common poolers.

Pooler	AL / CLT?	Influence fcn. ψ_P in $L^2(P)$?
Sample mean	✓	✓
Huber M-estimator (δ fixed)	✓	✓
Sample median	✓	✓
Top-k or max	×	×
Softmax ($\tau > 0$ fixed)	✓	✓

Generators All neural generators considered in the experiments— MLPs, Transformer decoders, and diffusion-score networks with fixed weights—are compositions of C^2 maps on finite-dimensional spaces and therefore satisfy Assumption 3.

Smooth Approximation of Non-Regular Statistics The theory developed here establishes that Hadamard differentiability of the encoder ensure asymptotic normality and consistency and in subsection 5.1 we develop the idea that our encoders learn sufficient statistics. But what if the sufficient statistic of interest is not Hadamard differentiable? The sample maximum is a classic example: it is the minimal sufficient statistic for the endpoint of a uniform distribution (see Example 3), yet it is not asymptotically normal.

Let $X_1, \ldots, X_n \sim \mathrm{Uniform}(0, \theta)$. The sample maximum

$$X_{(n)} := \max\{X_1, \ldots, X_n\}$$

satisfies

$$n(\theta - X_{(n)}) \xrightarrow{d} \mathrm{Exp}(1/\theta),$$

so it converges to θ but its asymptotic distribution is exponential, not Gaussian. This occurs because the maximum is not a smooth functional of the empirical distribution: it fails Hadamard differentiability, so the functional delta method does not apply.

A natural remedy is to approximate the max by a smooth function. A standard choice is the *log-sum-exp*:

$$\mathrm{LSE}_\lambda(X_1, \ldots, X_n) = \frac{1}{\lambda} \log \left(\sum_{i=1}^{n} e^{\lambda X_i} \right).$$

For fixed λ, this is Hadamard differentiable and thus amenable to the theory developed above. As $\lambda \to \infty$, $\mathrm{LSE}_\lambda \to \max_i X_i$, so we recover the max in the limit.

Corollary 3 (Smooth approximation suffices for asymptotic normality)**.** Let $T(P_m)$ be a non-smooth statistic (e.g., the maximum), and let $T^{(\lambda)}(P_m)$ be a family of smooth approximations (e.g., LSE_λ) such that $T^{(\lambda)}(P_m) \to T(P_m)$ pointwise. Then for any fixed λ, $T^{(\lambda)}(P_m)$ is Hadamard differentiable and admits asymptotically normal plug-in estimators. Moreover, if $\lambda_n \to \infty$ slowly as $n \to \infty$, this family can approximate $T(P_m)$ arbitrarily closely while retaining asymptotic normality.

Thus, even when the true sufficient statistic is not regular, a Hadamard differentiable encoder can still be learned to approximate it. This ensures that the asymptotic guarantees from Theorem 2 continue to hold. This also highlight why we cannot use e.g. max-pooling in the encoder, since that would break our CLT.

D.3 Embeddings and Predictive Sufficiency

Setting. Let $\mathcal{M} \subset \mathcal{P}(\mathcal{X})$ be the statistical manifold introduced in Section D.2.

Here we assume the statistical manifold $\mathcal{M}$ is d–dimensional (in the usual differential-geometric sense), so $\dim T_P\mathcal{M} = d$ for every $P \in \mathcal{M}$.

For $P \in \mathcal{M}$ observe $S_m = (X_1, \ldots, X_m) \overset{\text{i.i.d.}}{\sim} P$ and write the empirical measure $P_m = m^{-1} \sum_{i=1}^{m} \delta_{X_i}$.

Throughout we use the *plug-in predictor* P_m. Given a statistic $T_m = \phi(P_m)$ with $\phi : \mathcal{M} \to \mathbb{R}^d$, define a measurable *reconstruction* map $R : \phi(\mathcal{M}) \to \mathcal{M}$ and set

$$P_m^\phi := R\big(T_m\big).$$

Definition 7 (Predictive sufficiency)**.** The statistic $T_m = \phi(P_m)$ is *asymptotically predictive sufficient* if there exists a reconstruction R such that, for every $P \in \mathcal{M}$,

$$\big\| P_m - P_m^\phi \big\|_{\mathrm{TV}} \xrightarrow[m\to\infty]{P^{\otimes m}} 0.$$

This notion of sufficiency coincides with the one used in Section D.2: both ask that the predictor available to the decoder (here P_m^ϕ) converges in total variation to the full plug-in predictor P_m.

Theorem 3 (Embedding $\iff$ Predictive sufficiency)**.** *Assume ϕ is C^1 and satisfies the encoder regularity conditions of Assumption 2. Then the following are equivalent.*

- *(i)* Smooth embedding: *ϕ is injective and its differential $d\phi_P : T_P\mathcal{M} \to \mathbb{R}^d$ is bijective for every $P \in \mathcal{M}$.*
- *(ii)* Predictive sufficiency: *$T_m = \phi(P_m)$ is asymptotically plug-in sufficient in the sense of Definition 7.*

Proof. Throughout, $\|\cdot\|_{\mathrm{BL}}$ denotes the bounded–Lipschitz norm on signed measures, and $\|\cdot\|_{\mathrm{TV}} \le \|\cdot\|_{\mathrm{BL}}$.

Step 1: (i) $\implies$ (ii).

Step 1(a): global inverse and Lipschitz constant. Because ϕ is a C^1 diffeomorphism onto its image, the inverse–function theorem supplies, for every $P \in \mathcal{M}$, an open neighbourhood $U_P \subset \mathcal{M}$ on which $R \equiv \phi^{-1}$ is also C^1. Shrink U_P so that the operator norm of dR_Q is bounded by some $L_P < \infty$ for all $Q \in U_P$; then R is L_P-Lipschitz on U_P under $\|\cdot\|_{\mathrm{BL}}$.

Step 1(b): stochastic linearisation of $\phi(P_m)$. Encoder regularity (Assumption 2) gives

$$\sqrt{m}\,\big\{\phi(P_m) - \phi(P)\big\} \;=\; \frac{1}{\sqrt{m}} \sum_{i=1}^{m} \psi_P(X_i) + o_P(1) \quad \text{in } \mathbb{R}^d,$$

so $\|\phi(P_m) - \phi(P)\| = O_P(m^{-1/2})$.

Step 1(c): reconstructing P_m. For m large enough $P_m \in U_P$ with probability one, whence

$$\big\| P_m - R\big(\phi(P_m)\big) \big\|_{\mathrm{BL}} \;\le\; L_P \, \big\| \phi(P_m) - \phi(P) \big\| \;=\; O_P(m^{-1/2}).$$

Dividing by $\|\cdot\|_{\mathrm{TV}}$ concludes plug-in sufficiency.

Step 2: (ii) $\implies$ (i).

Step 2(a): Continuity of ϕ. Suppose $P_n \to P$ in $\mathcal{M}$ but $\phi(P_n) \not\to \phi(P)$. Choose $\varepsilon > 0$ and a subsequence (still indexed by n) with $\|\phi(P_n) - \phi(P)\| \geq \varepsilon$. For each n draw $S_m^{(n)} \sim P_n^{\otimes m_n}$ with $m_n \uparrow \infty$ slowly enough that $\|P_{m_n}^{(n)} - P_n\|_{\mathrm{TV}} \leq \varepsilon/4$ w.p. $\geq 1 - \varepsilon$. By sufficiency, $\|R(\phi(P_{m_n}^{(n)})) - P_{m_n}^{(n)}\|_{\mathrm{TV}} \leq \varepsilon/4$ with the same probability. The triangle inequality then forces $\|R(\phi(P_{m_n}^{(n)})) - P\|_{\mathrm{TV}} \geq \varepsilon/2$, contradicting $R(\phi(P_{m_n}^{(n)})) \xrightarrow{d} P$. Hence ϕ is continuous at every point.

Step 2(b): Injectivity of ϕ. Assume $\phi(P_1) = \phi(P_2)$ with $P_1 \neq P_2$. Choose a measurable set B for which $P_1(B) \neq P_2(B)$. Under $P_1^{\otimes m}$ we have $P_m(B) \to P_1(B)$ almost surely, while sufficiency yields $R(T_m)(B) \to P_1(B)$. Repeating under $P_2^{\otimes m}$ forces $P_2(B) = P_1(B)$, contradiction. Hence ϕ is injective.

Step 2(c): Injectivity of $d\phi_P$. Suppose there is $v \in T_P\mathcal{M} \setminus \{0\}$ with $d\phi_P[v] = 0$. Pick a C^1 path $t \mapsto P_t$ in $\mathcal{M}$ with $P_0 = P$ and $\partial_t P_t|_0 = v$. Taylor expansion of $\phi(P_t)$ yields $\|\phi(P_t) - \phi(P)\| = o(t)$, whereas $\|P_t - P\|_{\mathrm{BL}} = \Theta(t)$. Setting $t = m^{-1/2}$ violates sufficiency exactly as in the previous step. Hence $d\phi_P$ is injective; because the tangent and target spaces share the same (finite) dimension, it is bijective.

Step 2(d): Smooth embedding. Injectivity, continuity, and bijective differentials for all $P \in \mathcal{M}$ imply that ϕ is a smooth embedding. □

Remark (Identifiability is automatic)**.** *Because each $P \in \mathcal{M}$ already defines a unique predictive distribution, any statistic that is plug-in sufficient must be injective; no separate identifiability condition is required.*

E Extensions

E.1 Extension to Multiscale Settings

In many applications, data is naturally organized across multiple scales. For example, we may observe distributions of samples at a fine scale (e.g., single cells), grouped into entities at a coarser scale (e.g., patients), which themselves may belong to larger groups (e.g., hospitals). More generally, we may observe hierarchical data in which each level exhibits internal distributional structure.

Our framework naturally extends to such multiscale settings. At each scale s, we observe a set of units indexed by $i = 1, \dots, n^{(s)}$. Each unit i at scale s is associated with: a set of samples $S_{i,m}^{(s)} = \{x_{ij}^{(s)}\}_{j=1}^{m}$, drawn i.i.d. from a distribution $P_i^{(s)}$ and a higher-scale sample $x_i^{(s+1)} \in \mathcal{X}^{(s+1)}$, representing the corresponding entity at scale $s+1$.

The lower-scale distributions $P_i^{(s)}$ are drawn i.i.d. from a meta-distribution $Q^{(s)}$ over $\mathcal{P}(\mathcal{X}^{(s)})$, while the higher-scale samples $x_i^{(s+1)}$ are drawn from $P_i^{(s+1)}$, where $P_i^{(s+1)} \sim Q^{(s+1)}$.

Each lower-scale set $S_{i,m}^{(s)}$ defines an empirical measure

$$P_{i,m}^{(s)} = \frac{1}{m} \sum_{j=1}^{m} \delta_{x_{ij}^{(s)}} \in \mathcal{P}_m(\mathcal{X}^{(s)}).$$

At each scale we learn: an encoder $\mathcal{E}^{(s)} : \mathcal{P}_m(\mathcal{X}^{(s)}) \to \mathbb{R}^{d_s}$ mapping lower-scale empirical distributions into latent space, an encoder $\mathcal{E}^{(s+1)} : \mathcal{X}^{(s+1)} \to \mathbb{R}^{d_{s+1}}$ mapping higher-scale samples into the corresponding latent space, and generators $\mathcal{G}^{(s)} : \mathbb{R}^{d_s} \to \mathcal{P}(\mathcal{X}^{(s)})$ and $\mathcal{G}^{(s+1)} : \mathbb{R}^{d_{s+1}} \to \mathcal{P}(\mathcal{X}^{(s+1)})$ at each scale.

To link adjacent scales, we introduce deterministic maps

$$f^{(s)} : \mathbb{R}^{d_s} \to \mathbb{R}^{d_{s+1}} \quad \text{and} \quad g^{(s)} : \mathbb{R}^{d_{s+1}} \to \mathbb{R}^{d_s},$$

which project embeddings upward and downward between latent spaces.

We jointly train to enforce: *Approximate identity* at each scale:

$$\mathcal{G}^{(s)}(\mathcal{E}^{(s)}(S_{i,m}^{(s)})) \approx P_i^{(s)}, \quad \mathcal{G}^{(s+1)}(\mathcal{E}^{(s+1)}(x_i^{(s+1)})) \approx P_i^{(s+1)},$$

and *co-embedding consistency*: the mapped lower-scale embedding $f^{(s)}(\mathcal{E}^{(s)}(S_{i,m}^{(s)}))$ should align with the higher-scale embedding $\mathcal{E}^{(s+1)}(x_i^{(s+1)})$ and vice versa via $g^{(s)}$.

Formally, we optimize objectives of the form:

$$L = \mathfrak{d}\big(P_i^{(s)}, \mathcal{G}^{(s)}(\mathcal{E}^{(s)}(S_{i,m}^{(s)}))\big) \tag{4}$$

$$+ \quad \mathfrak{d}\big(P_i^{(s+1)}, \mathcal{G}^{(s+1)}(\mathcal{E}^{(s+1)}(x_i^{(s+1)}))\big) \tag{5}$$

$$+ \quad \|f^{(s)}(\mathcal{E}^{(s)}(S_{i,m}^{(s)})) - \mathcal{E}^{(s+1)}(x_i^{(s+1)})\|^2 \tag{6}$$

$$+ \quad \|g^{(s)}(\mathcal{E}^{(s+1)}(S_{i,m}^{(s+1)})) - \mathcal{E}^{(s)}(x_i^{(s)})\|^2 \tag{7}$$

where $\mathfrak{d}$ is a divergence or distance (e.g., KL divergence, Wasserstein distance) defined by the generative model. One natural approach would be to let $f^{(s)}, g^{(s)}$ both be the identity, forcing the model to learn a co-embedding across scales. But this may be too rigid and we might prefer more flexilbity in practice.

This bi-directional coupling ensures that embeddings at adjacent scales are mutually predictive and geometrically aligned, while each scale individually satisfies distributional invariance and approximate identity. The framework naturally generalizes to hierarchies involving more than two scales by recursively composing the maps $f^{(s)}$ and $g^{(s)}$ across levels.

F Broader impacts

Generative distribution embeddings provide a general framework for modeling data across scales. They are broadly applicable to a wide variety of problems, including those with direct societal consequences, for example in healthcare. In these settings, it will be critical to consider any potential inequities induced by GDEs, as is the case for any modelling approach. Lastly, we acknowledge the environmental impact of this paper, which used nontrivial amounts of computational resources, estimated to be about 54kg CO_2.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction concretely state the main theoretical and empirical results of the paper, and enumerate the demonstrated applications of our method.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We have a separate limitations subheading under the Discussion section. We clearly state key limitations of our method (assumption of exchangability, etc).

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: Theoretical results are stated with complete proof and assumptions in the Appendix of the paper. Informal versions of theoretical results are provided in the main text.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.

- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All experimental details are provided in the Appendix. Moreover, all experimental results can be reproduced by running the code in the provided (anonymized Github repository). Models can be trained using the appropriate experiment configs in the `config/experiment/` directory, and figures from the paper can be reproduced by running notebooks in the `notebooks/` directory.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code and datasets are made publicly available, and code necessary to reproduce results are provided with documentation.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.

- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All experimental details (including train/test splits and model implementation choices) are provided in the Appendix, and can be found in the accompanying (anonymized) Github repository.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Standard errors are reported where relevant and feasible.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

Justification: The appendix includes details of the compute resources used for this work. Full internal cluster details will be released after the double-blind period ends.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We adhere to the code of ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss potential societal impacts in the broader impacts section in Appendix of the paper.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We do not release datasets or models with high risk for misuse.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite the datasets, code, and models used in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code provided in the accompanying repository are well-documented.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We do not perform crowdsourcing experiments or research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.

- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: LLMs are not an important component of this work.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.