Exchangeable Generative Models with Flow Scans

Christopher M. Bender,[*1] **Kevin O'Connor,**[*2] **Yang Li,**[1] **Juan Jose Garcia,**[1]
Junier Oliva,[1] **Manzil Zaheer**[3]

[1]Department of Computer Science, UNC Chapel Hill
[2]Department of Statistics and Operations Research, UNC Chapel Hill
[3]Google Research
{bender, yangli95, jjgarcia, joliva}@cs.unc.edu,
koconn@live.unc.edu, manzilz@google.com

Abstract

In this work, we develop a new approach to generative density estimation for exchangeable, non-*i.i.d.* data. The proposed framework, FlowScan, combines invertible flow transformations with a sorted scan to flexibly model the data while preserving exchangeability. Unlike most existing methods, FlowScan exploits the intradependencies within sets to learn both global and local structure. FlowScan represents the first approach that is able to apply sequential methods to exchangeable density estimation without resorting to averaging over all possible permutations. We achieve new state-of-the-art performance on point cloud and image set modeling.

Introduction

Modeling unordered, non-*i.i.d.* data is an important problem in machine learning and data science. Collections of data objects with complicated intrinsic relationships are ubiquitous. These collections include sets of 3d points sampled from the surface of complicated shapes like human organs, sets of images shared within the same web page, or point cloud LiDAR data observed by driverless cars. In any of these cases, the collections of data objects do not possess any inherent ordering of their elements. Thus, any generative model which takes these data as input should not depend on the order in which the elements are presented *and* must be flexible enough to capture the dependencies between co-occurring elements.

The unorderedness of these kinds of collections is captured probabilistically by the notion of *exchangeability*. Formally, a set of points $\{x_j\}_{j=1}^n \subset \mathbb{R}^d$ with cardinality n, dimension d, and probability density $p(\cdot)$ is called exchangeable if

$$p(x_1, ..., x_n) = p(x_{\pi_1}, ..., x_{\pi_n}) \tag{1}$$

for every permutation π. In practice $\{x_j\}_{j=1}^n$ often represent 2d or 3d spatial points (see Fig. 1) in which case we refer to the set as a point cloud. In other settings, the points of interest may be more complex like images represented as very high-dimensional vectors.

As a simple example, one may trivially generate a set of exchangeable points by drawing them *i.i.d.* from some distribution. More commonly, elements within an exchangeable

---

Figure 1: A training dataset of sets. Each instance $\mathcal{X}_i$ is a set of points $\mathcal{X}_i = \{x_{i,j} \in \mathbb{R}^d\}_{j=1}^{n_i}$ ($d = 2$ shown). We estimate $p(\mathcal{X}_i)$, from which we can sample distinct sets.

set share information with one another, providing structure. Despite the abundance of such data, the bulk of existing approaches either ignore the relation between points (*i.i.d.* methods) or model dependencies in a manner that depends on inherent orderings (sequential methods) (Rezatofighi et al. 2017; You et al. 2018). In order to accurately learn the structure of a set whilst preserving the exchangeability of its likelihood, one cannot rely solely on either approach.

In this work, we focus on the task of tractable, non-*i.i.d.* density estimation for exchangeable sets. We explore both low cardinality sets of high dimension (10-20 points with many hundreds of dimensions each, e.g. collections of images) and high cardinality sets of low dimension (hundreds of points with 2-7 dimensions each, e.g. point clouds). We develop a generative model suitable for exchangeable sets in either regime, called FlowScan, which does not rely on *i.i.d.* assumptions and is provably exchangeable. Contrary to intuition, we show that one can preserve exchangeability while scanning over the data in a sorted manner. *FlowScan is the first method to achieve a tractable, non-*i.i.d.*, exchangeable likelihood by leveraging traditional (e.g. sequential), non-exchangeable density estimators.*

Main Contributions. 1) We show that transforming points with an equivariant change of variables allows for modeling sets in a different space. 2) We introduce a scanning-based technique for modeling exchangeable data, relating the underlying exchangeable likelihood to that of the sorted covariates. 3) We demonstrate how traditional density estimators may be used for the task of principled and feasible exchangeable density estimation via a scanning-based approach. 4) We show empirically that FlowScan achieves the state-of-the-art for density estimation tasks in both synthetic and real-world point cloud and image set datasets.

Motivation and Challenges

We motivate our problem with a simple, yet common, set generative process that requires a *non-i.i.d.*, exchangeable density estimator. Consider the following generative process for a set: 1) generate *latent* "parameters" $\phi \sim p_\Phi(\cdot)$ and then 2) generate a set $\mathcal{X} \sim p(\cdot \mid \phi)$. Here $p(\cdot \mid \phi)$ may be as simple as a Gaussian model (where ϕ is the mean and covariance parameters) or as complex as a nonparametric model (where ϕ may be infinite-dimensional).

This simple set generative process requires a *non-i.i.d.* approach, even for the case when the ground truth conditional set likelihood, $p(\mathcal{X} \mid \phi)$, is *conditionally i.i.d.*. We show this by first noting that with *conditionally i.i.d.* $p(\mathcal{X} \mid \phi) = \prod_{j=1}^n p(x_j \mid \phi)$, the complete set likelihood is:

$$p(\mathcal{X}) = \int p_\Phi(\phi) \prod_{j=1}^n p(x_j \mid \phi)\, \mathrm{d}\phi. \qquad (2)$$

(Note, that Eq. 2 is in the same vein as De Finetti's theorem (Bernardo and Smith 2009).) One can show dependency (*non-i.i.d.*) with the conditional likelihood of a single point x_k given a disjoint subset $S \subset \mathcal{X} \setminus \{x_k\}$: $p(x_k \mid S) = \int p_\Phi(\phi \mid S) p(x_k \mid S, \phi)\, \mathrm{d}\phi = \int p_\Phi(\phi \mid S) p(x_k \mid \phi)\, \mathrm{d}\phi \neq \int p_\Phi(\phi) p(x_k \mid \phi)\, \mathrm{d}\phi = p(x_k)$. That is, the conditional likelihood $p(x_k \mid S)$ depends on other points in $\mathcal{X}$ via the posterior $p_\Phi(\phi \mid S)$, which accounts for what ϕ was likely to have generated S. As a consequence, the complete generative process (2) is *not marginally i.i.d.*, notwithstanding the *conditional i.i.d.* $p(\mathcal{X} \mid \phi)$. Thus, any model built on an *i.i.d.* assumption may be severely biased.

The generative process in Eq. (2) is especially applicable for surface point cloud data. For such sets, $\mathcal{X}_i$, points are drawn *i.i.d.* from (conditioned on) the surface of a shape with (*unknown*) parameters ϕ_i (e.g. object class, length, orientation, noise, etc.), resulting in the dataset $\mathcal{D} = \{\mathcal{X}_i \sim p(\cdot \mid \phi_i)\}_{i=1}^N$ of N sets. As shown above, modeling such point cloud set data requires a *non-i.i.d.* approach even though points may be drawn independently given the surface parameters. FlowScan will not only yield an exchangeable, *non-i.i.d.* generative model, but will also directly model elements in sets without latent parameters. In effect, FlowScan will automatically marginalize out dependence on latent parameters of a given set, and is thus capable of handling complicated $p(\cdot \mid \phi)$.

Broadly, the primary challenge in direct exchangeable density estimation is designing a flexible, invariant architecture which yields a valid likelihood. As explained above, using an *i.i.d.* assumption to enforce this property will severely hamper the performance of a model. To avoid this simplification, techniques often shoehorn invariances to observed orderings by feeding randomly permuted data into sequential models (Rezatofighi et al. 2017; You et al. 2018). Such approaches attempt to average out the likelihood of the model over all permutations:

$$p(\mathcal{X}) = \frac{1}{n!} \sum_\pi p_s(x_{\pi_1}, \ldots, x_{\pi_n}), \qquad (3)$$

where p_s is some sequential model. Of course, the observation of all potential orderings for even a modest collection

of points is infeasible. Furthermore, there are often no guarantees that the sequential model p_{seq} will learn to ignore orderings, especially for unseen test data (Vinyals, Bengio, and Kudlur 2015).

Given that an *i.i.d.* assumption is not robust and averaging over all permutations is infeasible, what operation should be used to ensure permutation invariance of the architecture? Instead of attempting to wash out the effect of order in an architecture as in Eq. 3, we propose to enforce invariance by adopting a prespecified ordering and scanning over elements in this order. As will be discussed in the Methods section, the benefit of estimating a likelihood over sorted data is that it frees us from the restriction of exchangeability. Given the sorted data, we can apply any number of traditional density estimators. However, such an approach presents its own challenges:

- **Determining a suitable way to scan through an exchangeable sequence.** That is, one must map the set $\mathcal{X} = \{x_j\}_{j=1}^n$ to a sequence $\mathcal{X} \mapsto (x_{[1]}, \ldots, x_{[n]})$ where $x_{[j]}$ denotes the j'th point in the sorted order.

- **Relating the likelihood of the scanned sequence to likelihood of the exchangeable set.** Modeling the exchangeable likelihood through a scanned likelihood is not immediately obvious; *a simple equality of the two does not hold*, $p(\mathcal{X}) \neq p(x_{[1]}, \ldots, x_{[n]})$.

- **Scanning in a space that is beneficial for modeling.** The native input space may not be best suited for modeling or scanning, hence it would be constructive to transform the exchangeable input prior to the scan.

- **Developing an architecture that exploits the structure gained in the scan.** The scanning operation will introduce sequential correlations among elements which need to be modeled successfully.

Next, we develop the FlowScan model while addressing each of these challenges.

Methods

FlowScan consists of three components: 1) a sequence of equivariant flow transformations $(\hat{q}_e)$, *to map the data to a space that is easier to model*; 2) a sort with correction factor *to allow for the use of non-exchangeable density estimators*; 3) a density estimator $(\hat{p}_s)$ (e.g. an autoregressive model which may utilize sequential flow transformations, $\hat{q}_c$), *to estimate the likelihood while accounting for correlations induced by sorting* (see Fig. 2). In this section, we motivate each piece of the architecture and detail how they combine to yield a highly flexible, exchangeable density estimator.

Equivariant Flow Transformations

FlowScan first utilizes a sequence of equivariant flow transformations. So-called "flow models" rely on the change of variables formula to build highly effective models for traditional non-exchangeable generative tasks (like image modeling) (Kingma and Dhariwal 2018). Using the change of variables formula, flow models approximate the likelihood of a d-dimensional distribution over real-valued covariates
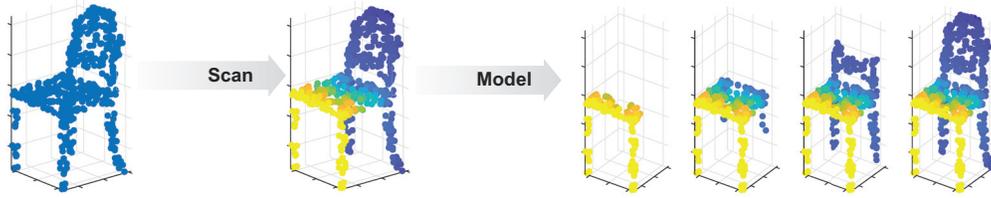
Figure 2: Illustration of our proposed method. First, input sets are scanned (in a possibly transformed space). After, the scanned covariates are modeled (possibly in a autoregressive fashion, as shown).

$x = (x^{(1)}, ..., x^{(d)}) \in \mathbb{R}^d$, by applying an invertible (flow) transformation $\hat{q}(x)$ to an estimated base distribution $\hat{f}$:

$$\hat{p}(x^{(1)}, ..., x^{(d)}) = \left| \det \frac{\mathrm{d}\hat{q}}{\mathrm{d}x} \right| \hat{f}(\hat{q}(x)), \qquad (4)$$

where $\left| \det \frac{\mathrm{d}\hat{q}}{\mathrm{d}x} \right|$ is the Jacobian of the transformation $\hat{q}$. Often, the base distribution is a standard Gaussian. However, (Oliva et al. 2018) recently showed that performance may be improved with a more flexible base distribution on transformed covariates such as an autoreggressive density (Germain et al. 2015; Gregor et al. 2014; Larochelle and Murray 2011; Uria et al. 2016; Uria, Murray, and Larochelle 2013).

There are a myriad of possible invertible transformations, $\hat{q}$, that one may apply to inputs $\mathbf{x} \in \mathbb{R}^{n \times d}$ in order to model elements in a more expressive space, where $\mathbf{x}$ is the set $\mathcal{X}$ represented as a matrix. However, in our case one must take care to preserve exchangeability of the inputs when transforming the data. For example, a simple affine change of variables will be sensitive to the order in which the elements of $\mathbf{x}$ were observed, resulting in a space which is no longer exchangeable. One can circumvent this problem by requiring that any transformation, $\hat{q}$, used is *equivariant*. That is, for all permutation operators, Γ, we have that $\hat{q}(\Gamma \mathbf{x}) = \Gamma \hat{q}(\mathbf{x})$. Proposition 1 states that equivariance of the transformations in conjunction with invariance of the base distribution is enough to ensure that exchangeability is preserved, *allowing one to model set data in a transformed space*. The proof is straightforward and relegated to the Appendix.

Proposition 1. *Let $\hat{q} : \mathbb{R}^{n \times d} \mapsto \mathbb{R}^{n \times d}$ be a permutation equivariant, invertible transformation and the base distribution, $\hat{f}$, be exchangeable. Then the likelihood, $\hat{p}(\mathbf{x}) = \left| \det \frac{\mathrm{d}\hat{q}}{\mathrm{d}\mathbf{x}} \right| \hat{f}(\hat{q}(\mathbf{x}))$, is exchangeable.*

Given an invertible transformation, $q : \mathbb{R}^d \to \mathbb{R}^d$, one may construct a simple permutation equivariant transformation by applying it to each point in a set independently: $(x_1, ..., x_n) \mapsto (q(x_1), ..., q(x_n))$. However, it is possible to engineer equivariant transformations which utilize information from other points in the set while still preserving equivariance. Proposition 1 shows that FlowScan is compatible with any combination of these transformations.

Set-Coupling Among others, we propose a novel set-level scaling and shifting coupling transformation (Dinh, Sohl-Dickstein, and Bengio 2016). For d-dimensional points, the coupling transformation scales and shifts one subset, $S \subset \{1, ..., d\}$ of the d covariates given then rest, S^c as (letting superscripts index point dimensions):

$$x^{(S)} \mapsto \exp\left(f\left(x^{(S^c)} \right) \right) \cdot x^{(S)} + g\left(x^{(S^c)} \right)$$
$$x^{(S^c)} \mapsto x^{(S^c)}, \qquad (5)$$

for learned functions $f, g : \mathbb{R}^{|S^c|} \mapsto \mathbb{R}^{|S|}$. We propose a set-coupling transformation as follows:

$$x_i^{(S)} \mapsto \exp\left(f\left(\varphi(\mathbf{x}^{(S^c)}), x_i^{(S^c)} \right) \right) \cdot x_i^{(S)} + g\left(\varphi(\mathbf{x}^{(S^c)}), x_i^{(S^c)} \right)$$
$$x_i^{(S^c)} \mapsto x_i^{(S^c)}, \qquad (6)$$

where $\mathbf{x}^{(S^c)} \in \mathbb{R}^{n \times |S^c|}$ is the set of unchanged covariates, $\varphi(\mathbf{x}^{(S^c)}) \in \mathbb{R}^r$ are general, learnable permutation invariant set embeddings. We utilize embeddings from DeepSet architectures (Zaheer et al. 2017); however, other embeddings are possible, e.g. prescribed statistics (Jebara, Kondor, and Howard 2004), and $f, g : \mathbb{R}^{r+|S^c|} \to \mathbb{R}^{|S|}$ are learned functions. The embedding φ is responsible for capturing set-level information from other covariates. This is combined with each point $x_i^{(S^c)}$ to yield shifts and scales with both point- and set-level dependence (see Fig. 3). The log-determinant and inverse are detailed in the Appendix along with several other examples of flexible, equivariant transformations.
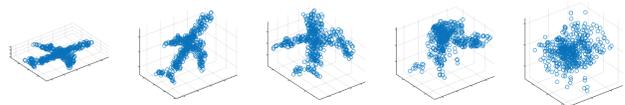

Figure 3: An illustration of how set-coupling transformations act on a set. The first plot shows the input data to be transformed. In the subsequent plots, the set is transformed in an invertible, equivariant fashion by stacking set-coupling transformations. Iteratively transforming dimensions of a set in this way yields a set with simpler structure that may be modeled more easily, as shown in the last plot.

Invariance Through Sorting

After applying a series of equivariant flow transformations, FlowScan performs a sort operation and corrects the likelihood with a factor of $1/n!$. Sorting in a prespecified fashion ensures that different permutations of the input map to the

same output. In this section, we prove that this yields an analytically correct likelihood and comment on the advantages of such an approach. Specifically, we show that the exchangeable (unordered) likelihood of a set of n points $p_e(x_1, \ldots x_n)$ (where $x_j \in \mathbb{R}^d$) can be written in terms of the non-exchangeable (ordered) likelihood of the points in a sorted order $p_s(x_{[1]}, \ldots, x_{[n]})$ as stated in Prop. 2 below.

Proposition 2. *Let p_e be an exchangeable likelihood which is continuous and non-degenerate (e.g. $\forall j \in \{1, \ldots, d\}$ $\Pr[x_1^{(j)} \neq x_2^{(j)} \neq \ldots \neq x_n^{(j)}] = 1$). Then,*

$$p_e(x_1, \ldots x_n) = \frac{1}{n!} p_s(x_{[1]}, \ldots, x_{[n]}), \qquad (7)$$

where $x_{[j]}$ is the jth point in the sorted order.

Proof. We derive Eq. 7 from a variant of the change of variables formula (Casella and Berger 2002). It states that if we have a partition of our input space, $\{\mathcal{A}_j\}_{j=1}^M$, such that a transformation of variables q is invertible in each partition $\mathcal{A}_j$ with inverse q_j^{-1}, then we may write the likelihood f of $z = q(u)$ in terms of the likelihood p of the input data u as:

$$f(z) = \sum_{j=1}^M \left| \det \frac{\mathrm{d}q_j^{-1}}{\mathrm{d}z} \right| p(q_j^{-1}(z)). \qquad (8)$$

For the moment, suppose that the points $\{x_j\}_{j=1}^n$ are sorted according to the first dimension. That is, $x_{[1]}, \ldots, x_{[n]}$ in Eq. 7 are such that $x_{[1]}^{(1)} < \ldots < x_{[n]}^{(1)}$. The act of sorting these points amounts to a transformation of variables $s : \mathbb{R}^{n \times d} \mapsto \mathbb{R}^{n \times d}$, $s(x_1, \ldots, x_n) = (x_{[1]}, \ldots, x_{[n]})$. The transformation s is one-to-one on the partitions of the input space $\mathbb{R}^{n \times d}$ defined by the relative order of points. In other words, we may partition the input space according to the permutation that would sort the data: $\mathcal{A}_\pi = \{\mathbf{x} \in \mathbb{R}^{n \times d} \mid x_{\pi_1}^{(1)} < x_{\pi_2}^{(1)} < \ldots < x_{\pi_n}^{(1)}\}$. We may invert s in $\mathcal{A}_\pi$ via the inverse permutation matrix of π, Γ_π^{-1}. Letting Π be the set of all permutations, Eq. 8 yields:

$$p_s(s(\mathbf{x})) \overset{*}{=} \sum_{\pi \in \Pi} \left| \Gamma_\pi^{-1} \right| p_e(\Gamma_\pi^{-1} s(\mathbf{x})) \overset{**}{=} n! \, p_e(\mathbf{x}), \qquad (9)$$

where (*) follows from Eq. 8 and (**) follows from the exchangeability of p_e. Thus, we may compute the exchangeable likelihood $p_e(\mathbf{x})$ using the likelihood of the sorted points, as in Eq. 7. Trivially, similar arguments also hold when sorting according to a dimension other than the first. Furthermore, it is possible to sort according any appropriately transformed space of x_j, rather than any native dimension itself (as this is equivalent to applying a transformation, sorting, and inverting said transformation). $\square$

Consequently, the exchangeable likelihood may be estimated via an approximation of the scanned covariates: $p_e(\mathbf{x}) \approx \frac{1}{n!} \hat{p}_s(s(\mathbf{x}))$. Since the density of sorted scan is not exchangeable, we may estimate $\hat{p}_s$ using traditional density estimation techniques. *This gives a principled approach to reduce the problem of exchangeable likelihood estimation to a flat vector (or sequence) likelihood estimation task.*

Autoregressive Scan Likelihood

After performing equivariant flow transformations and sorting, FlowScan applies a non-exchangeable density estimator to model the transformed and sorted data. Let $z = s(\hat{q}(\mathbf{x})) \in \mathbb{R}^{n \times d}$ be the sorted covariates. Since z is not exchangeable, one can apply any traditional likelihood estimator on its covariates, e.g. one may treat z as a vector and model $\hat{p}_s(\mathrm{vec}(z))$ using a flat density estimator. However, flattening in this way suffers from several disadvantages. First, it is inflexible to varying cardinalities. Furthermore, the total number of covariates, nd, may be large for sets with large cardinality or dimensionality. Finally, a general flat model loses the context that covariates are from multiple points in some shared set. To address these challenges, we use an autoregressive likelihood:

$$\hat{p}(z_k) = \prod_{k=1}^n \hat{p}(z_k \mid h_{<k}), \qquad (10)$$

where $\hat{p}(z_k \mid h_{<k})$ is itself a d-dimensional density estimator (such as Eq. 4) conditioned on a recurrent state $h_{<k} = h(z_1, \ldots, z_{k-1})$. This proposed approach is capable of sharing parameters across the n d-dimensional likelihoods and is more amenable to large, possibly varying, cardinalities.

Correspondence Flow Transformations

In much the same way that nearby pixels are correlated in image space, points with neighboring indices will be correlated in a scan space. Thus, we also propose a coupling (Dinh, Krueger, and Bengio 2014) invertible transformation to transform adjacent points, exploiting existing correlations among points as follows. We note that it is straightforward to use a sequential coupling transformation to shift and scale points z_i as in Eq. 5, but based on inputting a recurrent output $h_{<i}$ to f and g functions. In addition, it is also possible to split individual points for coupling as follows. First, split the scanned points $z = s(\hat{q}(\mathbf{x})) = (z_1, \ldots, z_n)$ into two groups depending on the parity (even/odd) of their respective index. Second, transform each even point, with a scale and shift based on the corresponding odd point. That is for pairs of points (z_{2j}, z_{2j+1}) we perform the following transformation: $(z_{2j}, z_{2j+1}) \mapsto (s(z_{2j+1})z_{2j} + m(z_{2j+1}), z_{2j+1})$, where $s : \mathbb{R}^d \mapsto \mathbb{R}^d, m : \mathbb{R}^d \mapsto \mathbb{R}^d$ are scale and shifting functions, respectively, parameterized by a learnable fully connected network. This correspondence coupling transformation $z \mapsto z'$ is easily invertible and has analytical Jacobian determinant $\left| \det \frac{\mathrm{d}z'}{\mathrm{d}z} \right| = \prod_{j=0}^{n/2-1} |s(z_{2j+1})|$. Several of these transformations may be stacked before the autoregressive likelihood by alternating between shifting and scaling even points based on odd and vice-versa odd points based on even. We shall also make use of a similar splitting scheme to split sets of images into 3d tensors that are fed into 3d convolution networks for shifting and scaling.

Complete FlowScan Architecture

Since the scanned likelihood in Eq. 7 yields an exchangeable likelihood, one may use as the base likelihood following a permutation equivariant transformation as in Prop. 1. This

enables us to apply the sorting step after performing any number of equivariant transformations and improve the flexibility of the model as a result. As no generality is lost, we choose to sort on the first dimension in our experiments detailed below. Combining the three components detailed above, we arrive at the complete FlowScan architecture: a sequence of equivariant flow transformations, a sort with correction factor, and an autoregressive scan likelihood. The estimated exchangeable likelihood that results is:

$$\hat{p}_{\text{fs}}(\mathbf{x}) = \frac{1}{n!} \left| \det \frac{d\hat{q}_e}{dx} \right| \hat{p}_s(s(\hat{q}_e(\mathbf{x}))), \quad (11)$$

where $\hat{q}_e$ and $\hat{p}_s$ are the estimated (via maximum likelihood) equivariant flow transformation and sorted flow scan covariate likelihood, respectively. When correspondence flow transformations are included after the sort operation, we obtain an estimated exchangeable likelihood:

$$\hat{p}_{\text{fs}}(\mathbf{x}) = \frac{1}{n!} \left| \det \frac{d\hat{q}_e}{dx} \right| \left| \det \frac{d\hat{q}_c}{dx} \right| \prod_{k=1}^{n} \hat{p}(\mathbf{z}_k \mid h(\mathbf{z}_{<k})), \quad (12)$$

where $\mathbf{z}$ is the resulting covariates from corresponding coupling transforming the flow scanned covariates. In both cases, FlowScan gives a valid, provably exchangeable density estimate relying neither on variational lower bounds of the likelihood nor averaging over all possible permutations of the inputs. Furthermore, FlowScan is easily adapted to input sets with varying cardinalities, as is commonly observed in practice. In the Experiments section, we demonstrate empirically that FlowScan is highly flexible and capable of modeling sets of both points clouds and images.

Related Work

Unlike the recent surge in flexible density estimation for flat vectors with deep architectures (Dinh, Krueger, and Bengio 2014; Dinh, Sohl-Dickstein, and Bengio 2016; Kingma and Dhariwal 2018; Larochelle and Murray 2011; Uria, Murray, and Larochelle 2013; Uria et al. 2016; Gregor et al. 2014; Germain et al. 2015; Oliva et al. 2018), exchangeable treatments of data in ML have been limited with some notable exceptions. Some recent work (Lee et al. 2018; Qi et al. 2017; Zaheer et al. 2017) has explored neural architectures for constructing a permutation invariant set embeddings. They featurize input sets exchangeably in a way that is useful for (typically supervised) downstream tasks; *but the embeddings themselves will not result in valid likelihoods*. In other work, Generative Adversarial Networks (GAN) have been explored as a means of sampling point clouds (Zaheer et al. 2018). However, none of these methods provide a valid exchangeable likelihood estimate as is our focus.

A recently proposed model, BRUNO (Korshunova et al. 2018), preserves exchangeability by performing independent point-wise changes of variables, a simple equivariant linear transformation, and an *i.i.d.* base exchangeable process in the latent space. The Neural Statistician (NS) (Edwards and Storkey 2017) estimates a permutation invariant code produced by an exchangeable VAE. That is, the Neural Statistician uses an encoder, called a statistics network, on the entire

exchangeable set to get an approximate posterior on the latent code. Given the success of a point cloud autoencoder with a DeepSet network as the statistics network in (Oliva et al. 2018), we consider this architecture for the variational Neural Statistician which is an especially strong baseline, representing the state-of-the-art likelihood method for point cloud data.

Experiments

In this section, we compare the performance of FlowScan to that of BRUNO and NS in a variety of exchangeable point cloud and image modeling tasks. In each experiment, our goal is to estimate an exchangeable likelihood $p(\mathbf{x})$ for $\mathbf{x} \in \mathbb{R}^{n \times d}$ which models the inputs well. As is standard in density estimation tasks, we measure the success of the model via the estimated likelihood of a held out test set for each experiment. For readability, we report the estimated log likelihood divided by the number of points (per point log likelihoods, PPLL): $\frac{1}{n} \log \hat{p}(\mathbf{x})$. The PPLL provides a set-level likelihood that eases comparison across dataset and cardinality. As NS does not yield a likelihood, we report its estimated variational lower bound on the PPLL. Results for each datasets can be found in Tab. 1.

As a qualitative assessment of each model's performance, we also include samples generated by each trained model. Those which are not reported in the main text can be found in the Appendix. Unless stated explicitly, the figures included are *not reconstructions*, but completely synthetic point clouds or images generated by each model. Further implementation details (including code and Appendices) can be found at https://github.com/lupalab/flowscan.

Shuffled Synthetic Sequential Data

We begin with a synthetic point cloud experiment to test FlowScan's ability to learn a known, ground truth likelihood. To allow for complex interactions between points, we study a common scenario that leads to exchangeable data: sequential data with time marginalized out. In other words, we suppose that all time-points $x_j \in \mathbb{R}^d$ of a sequence $(x_1, \ldots, x_n)$ are put into an unordered set $\{x_1, \ldots, x_n\}$. Effectively, this yields observations of sequences in matrices that are randomly shuffled from the sequential order. Hence, exchangeable instances are $\mathbf{x} = \Gamma_\pi \mathbf{x}_s$, for permutations

Dataset	BRUNO	NS	FlowScan
Synthetic	-2.28	-1.07	**0.14**
Airplanes	2.71	4.09	**4.81**
Chairs	0.75	2.02	**2.58**
ModelNet10	0.49	2.12	**3.01**
ModelNet10a	1.20	2.82	**3.58**
Caudate	1.29	4.49	**4.87**
Thalamus	-0.815	2.69	**3.12**
SpatialMNIST	-5.68	-5.37	**-5.26**

Table 1: Per-point log-likelihood (PPLL) of the test set for all point cloud experiments. Higher PPLL indicates better modeling of the test set.

$\Gamma_\pi \in \mathbb{R}^{n \times n}$ (drawn uniformly at random) and sequential data $\mathbf{x}_s = (x_1, \ldots, x_n) \in \mathbb{R}^{n \times d}$ (drawn via a sequential likelihood p_{seq}). Here we consider a synthetic ground truth sequential model p_{seq} where the likelihood of an instance is computed by marginalizing out the permutation: $p(\mathbf{x}) = \sum_{\pi'} \Pr(\pi = \pi') p_{\text{seq}}(\Gamma_{\pi'}^{-1} \mathbf{x}) = \frac{1}{n!} \sum_\pi p_{\text{seq}}(\Gamma_\pi \mathbf{x})$. To obtain interesting non-linear dependencies we consider a sinusoidal sequence (see Fig. 4 and Appendix for details). To allow for computing the ground truth likelihood in a timely manner, we consider $n = 8$, leading to a large number, $8! = 40320$, of summands in the likelihood of the data.
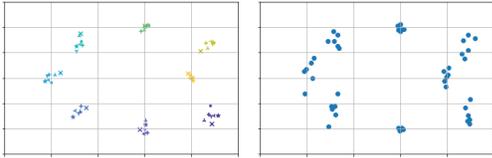

Figure 4: Left: true samples; markers and colors indicate instances and sequential order. Right: FlowScan samples.

Table 1 illustrates the per point log likelihood (PPLL) estimates across the synthetic sets using BRUNO, the NS, and FlowScan. The FlowScan model outperforms the other methods, achieving nearly the same PPLL as the ground truth (0.23) despite not averaging over all $n!$ permutations. For further comparison, we also trained a sequential model on the randomly permuted instances (and marginalizing out the permutation as in Eq. 3). However, randomly permuting the input sequence proved to be ineffective and resulted in low test PPLLs (with severe overfitting).

ModelNet

Next, we illustrate the efficacy of our model on real world point cloud data. We consider object classes from the Model-Net dataset (Wu et al. 2015), which contains CAD models of common real world objects. Point clouds were created by randomly sampling 512 points from the surface of each object. All point cloud sets are modeled in an unsupervised fashion. That is, we estimate $p(\mathbf{x})$, where $\mathbf{x} \in \mathbb{R}^{512 \times 3}$. Models are compared on the following datasets comprised of different subsets of point cloud classes: airplanes, chairs, ModelNet10, and ModelNet10a. ModelNet10 is the standard subset (Wu et al. 2015) consisting of *bathtub*, *bed*, *chair*, *desk*, *dresser*, *monitor*, *night stand*, *sofa*, *table*, and *toilet* classes. Since ModelNet10 is composed largely of furniture-like objects, we also select a more diverse, ten-class subset that we will refer to as ModelNet10a, containing *airplane*, *bed*, *car*, *chair*, *guitar*, *lamp*, *laptop*, *plant*, *stairs*, and *table* classes.

Results can be found in Tab. 1 and four samples from FlowScan are included in Fig. 6. For each of the four datasets tested, we find that FlowScan achieves the highest average test log-likelihood. Qualitatively, we also observe superior samples from the FlowScan model as can be seen in Fig. 5 and in the Appendix. In addition to training on these Model-Net datasets, we also performed an ablation study (see the Appendix) where we see that our full architecture yields the
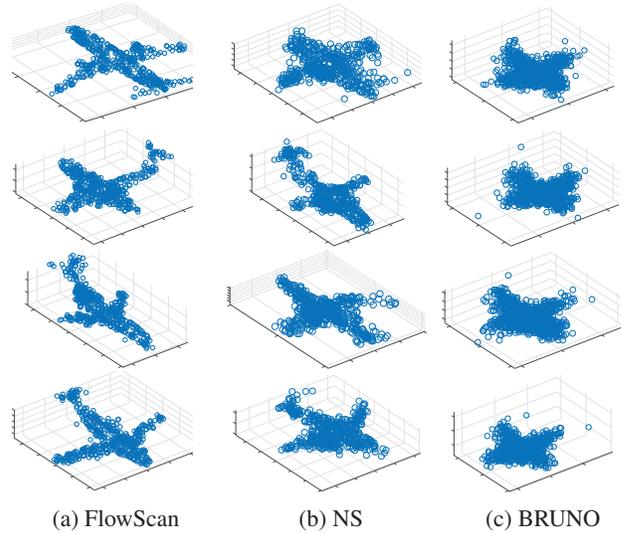

(a) FlowScan   (b) NS   (c) BRUNO

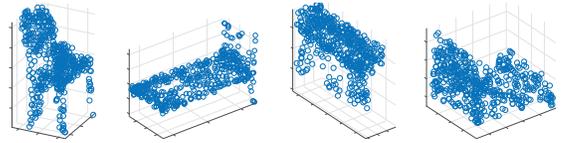Figure 5: Synthetic plane samples from trained models

Figure 6: FlowScan ModelNet10 samples

best performance over alternatives.

Brain Data

We test FlowScan's performance on a medical imaging task in a higher dimensional setting using samples of the Caudate and Thalamus (Cody et al. 2017). Each set contains 512 randomly sampled 7d points. The first three dimensions contain the Cartesian coordinates of the surface boundary (as in ModelNet). The next two dimensions represent the normal direction at the boundary in terms of angles. The final two dimensions represent the local curvature (expressed as shape index and curvedness (Koenderink 1990)). Table 1 enumerates the PPLL for both datasets across all three methods. Comparing samples from FlowScan (see Fig. 7) to that of NS and BRUNO (included in the Appendix) we see that FlowScan better captures the geometric features of the data than NS. Overall, superior PPLLs and samples suggest that FlowScan seamlessly incorporates the additional geometric information to model point clouds more accurately than baseline methods.
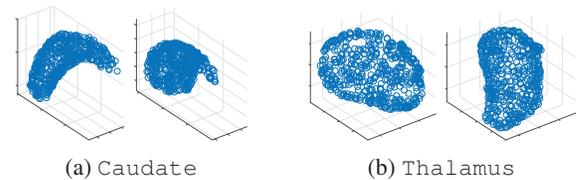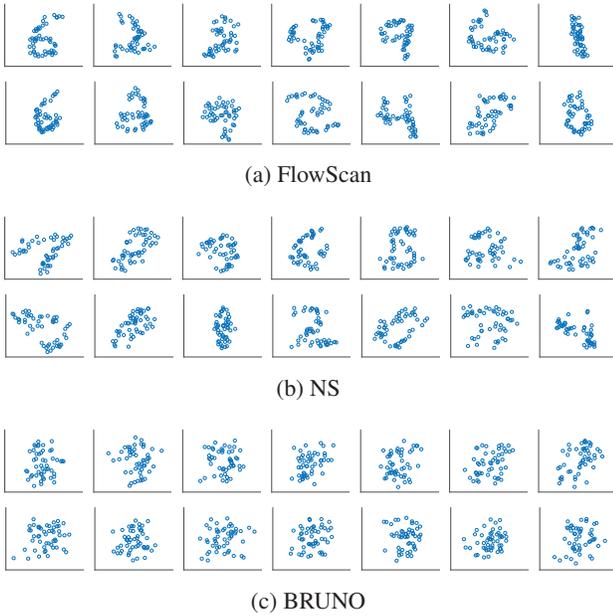

(a) Caudate   (b) Thalamus

Figure 7: FlowScan Caudate and Thalamus samples

(a) FlowScan

(b) NS

(c) BRUNO

Figure 8: `SpatialMNIST` samples from each model.

(a) FlowScan

(b) Set-Coupling

(c) BRUNO

Figure 9: Single digit set samples from FlowScan, Set-Coupling, and BRUNO trained on `MNIST`. Each row corresponds to a single set of 20 images generated by one model.

Spatial MNIST

For a direct comparison to NS, we also trained our model on the `SpatialMNIST` dataset, used by (Edwards and Storkey 2017). Each set consists of 50 points sampled uniformly at random from active pixels of a single `MNIST` (LeCun et al. 1998) image with uniform noise added to ensure non-degeneracy. The dataset that results consists of 2-dimensional point clouds each representing a digit from 0 to 9. PPLLs for each model can be found in Tab. 1 and a random selection of samples from each can be found in Fig. 8. Both the (unconditioned) likelihoods and the samples indicate that FlowScan gives superior performance in this task.

MNIST

Finally, we show that FlowScan exhibits superior likelihoodsand samples in a high-dimensional, low-cardinality setting. Following (Korshunova et al. 2018), sets are composed of 20 random images corresponding to the *same digit class* from the `MNIST` dataset. After training, PPLLs are evaluated on held out test sets constructed from unseen images. Our baseline is BRUNO, which achieves a PPLL of -643.6. BRUNO's unconditional samples (Fig. 9c) often contain elements from different digits, indicating a lack of intra-set dependency in the resulting model. We improve upon BRUNO by first adding convolution-based Set-Coupling transformations (but keeping the *i.i.d.* base likelihood), which achieves a PPLL of -634.8. Still, sample sets (Fig. 9b) show mixed digit classes. Finally, we consider a full FlowScan model that adds a sort, scan, and 3d convolution-based correspondence coupling transformations, which achieves the best PPLL of -621.7. Furthermore, FlowScan samples consistently contain the same digit class (Fig. 9a), showing that we are able to fully model the intra-set dependencies of elements.
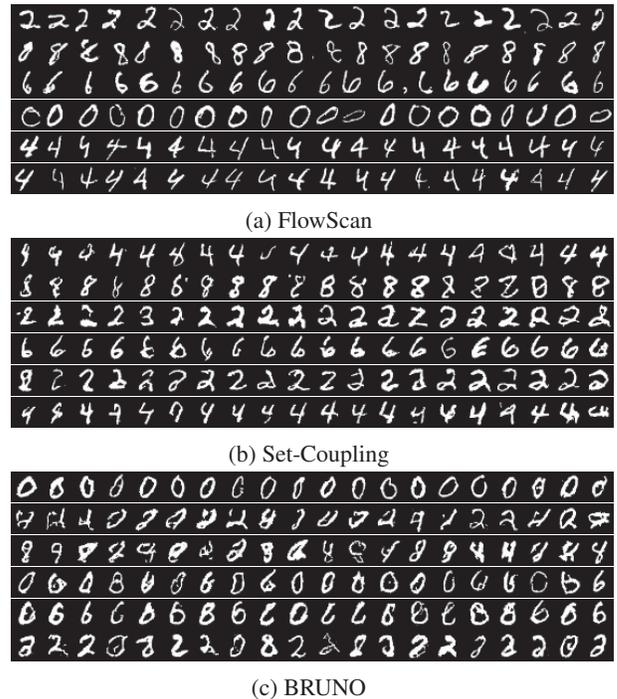
Conclusion

In this work, we introduced FlowScan for estimating exchangeable densities. This is a difficult task, where models were previously limited to either exchangeable base likelihoods (Korshunova et al. 2018), or conditionally *i.i.d.* restrictions with variational approximations of the likelihood (Edwards and Storkey 2017). We explored how to map inputs to a space that is easier to model whilst preserving exchangeability via equivariant flow transformations. Among others, we proposed the Set-Coupling transformation which extends existing pointwise coupling transformations (Dinh, Krueger, and Bengio 2014) to sets. Additionally, we demonstrated how to apply non-exchangeable density estimators to this task via sorting and scanning. This is the first tractable approach to achieve this, avoiding averaging over any permutations of the data while unlocking a much larger class of base likelihoods for exchangeable density estimation. Finally, we argued for the use of an autoregressive base likelihood with sequential transformations to exploit the sequential structure gained in the sort and scan. Combining equivariant flow transformations, sorting and scanning, and an autoregressive likelihood, we arrived at FlowScan. We showed empirically that FlowScan's ability to model intradependencies within sets surpassed that of other state-of-the-art methods in both high-cardinality, low-dimensionality and low-cardinality, high-dimensionality settings. Quantitatively FlowScan's likelihoods were a substantial improvement (see Tab. 1). Furthermore, there was a clear qualitative improvement in samples from FlowScan.

Acknowledgements

References

Bernardo, J. M., and Smith, A. F. 2009. *Bayesian theory*, volume 405. John Wiley & Sons.

Casella, G., and Berger, R. L. 2002. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA.

Cody, H.; Gu, H.; Munsell, B.; Kim, S.; Styner, M.; Wolff, J.; Elison, J.; Swanson, M.; Zhu, H.; Botteron, K.; Collins, L.; N. Constantino, J.; R. Dager, S.; M. Estes, A.; Evans, A.; Fonov, V.; Gerig, G.; Kostopoulos, P.; C. McKinstry, R.; and H. Gu, C. 2017. Early brain development in infants at high risk for autism spectrum disorder. *Nature* 542:348–351.

Dinh, L.; Krueger, D.; and Bengio, Y. 2014. Nice: Non-linear independent components estimation. *CoRR* abs/1410.8516.

Dinh, L.; Sohl-Dickstein, J.; and Bengio, S. 2016. Density estimation using real NVP. *CoRR* abs/1605.08803.

Edwards, H., and Storkey, A. 2017. Towards a neural statistician. In *5th International Conference on Learning Representations (ICLR 2017)*.

Germain, M.; Gregor, K.; Murray, I.; and Larochelle, H. 2015. MADE: Masked Autoencoder for Distribution Estimation. In Bach, F., and Blei, D., eds., *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, 881–889. Lille, France: PMLR.

Gregor, K.; Danihelka, I.; Mnih, A.; Blundell, C.; and Wierstra, D. 2014. Deep autoregressive networks. In Xing, E. P., and Jebara, T., eds., *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, 1242–1250. Bejing, China: PMLR.

Jebara, T.; Kondor, R.; and Howard, A. 2004. Probability product kernels. *J. Mach. Learn. Res.* 5:819–844.

Kingma, D. P., and Dhariwal, P. 2018. Glow: Generative flow with invertible 1x1 convolutions. In *Advances in Neural Information Processing Systems*, 10236–10245.

Koenderink, J. J. 1990. *Solid Shape*. Cambridge, MA, USA: MIT Press.

Korshunova, I.; Degrave, J.; Huszar, F.; Gal, Y.; Gretton, A.; and Dambre, J. 2018. Bruno: A deep recurrent model for exchangeable data. In *Advances in Neural Information Processing Systems*, 7190–7198.

Larochelle, H., and Murray, I. 2011. The neural autoregressive distribution estimator. In Gordon, G.; Dunson, D.; and Dudík, M., eds., *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, 29–37. Fort Lauderdale, FL, USA: PMLR.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11):2278–2324.

Lee, J.; Lee, Y.; Kim, J.; Kosiorek, A. R.; Choi, S.; and Teh, Y. W. 2018. Set transformer. *arXiv preprint arXiv:1810.00825*.

Oliva, J. B.; Dubey, A.; Póczos, B.; Schneider, J.; and Xing, E. P. 2018. Transformation autoregressive networks. *arXiv preprint arXiv:1801.09819*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* 1(2):4.

Rezatofighi, S. H.; Milan, A.; Abbasnejad, E.; Dick, A.; Reid, I.; et al. 2017. Deepsetnet: Predicting sets with deep neural networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 5257–5266. IEEE.

Uria, B.; Côté, M.-A.; Gregor, K.; Murray, I.; and Larochelle, H. 2016. Neural autoregressive distribution estimation. *Journal of Machine Learning Research* 17(205):1–37.

Uria, B.; Murray, I.; and Larochelle, H. 2013. Rnade: The real-valued neural autoregressive density-estimator. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 2175–2183.

Vinyals, O.; Bengio, S.; and Kudlur, M. 2015. Order matters: Sequence to sequence for sets. *arXiv preprint arXiv:1511.06391*.

Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; and Xiao, J. 2015. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 1912–1920.

You, J.; Ying, R.; Ren, X.; Hamilton, W.; and Leskovec, J. 2018. Graphrnn: Generating realistic graphs with deep autoregressive models. In *International Conference on Machine Learning*, 5694–5703.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; and Smola, A. J. 2017. Deep sets. In *Advances in Neural Information Processing Systems*, 3391–3401.

Zaheer, M.; Li, C.-L.; Zhang, Y.; Poczos, B.; and Salakhutdinov, R. 2018. Point cloud gan. *arXiv preprint arXiv:1810.05795*.