Conditional Permutation Invariant Flows

Berend Zwartsenberg[1*] *berend.zwartsenberg@inverted.ai*
Adam Ścibior[1,2] *adam.scibior@inverted.ai*
Matthew Niedoba[1,2] *matthew.niedoba@inverted.ai*
Vasileios Lioutas[1,2] *vasileios.lioutas@inverted.ai*
Justice Sefas[1,2] *justice.sefas@inverted.ai*
Yunpeng Liu[1,2] *larry.liu@inverted.ai*
Setareh Dabiri[1] *setareh.dabiri@inverted.ai*
Jonathan Wilder Lavington[1,2] *jonathan.lavington@inverted.ai*
Trevor Campbell[3] *trevor@stat.ubc.ca*
Frank Wood[1,2,3,4] *fwood@cs.ubc.ca*

[1] *Inverted AI*
[2] *Department of Computer Science, University of British Columbia*
[3] *Department of Statistics, University of British Columbia*
[4] *Mila*

Reviewed on OpenReview: *https://openreview.net/forum?id=DUsgPi3oCC*

Abstract

We present a conditional generative probabilistic model of set-valued data with a tractable log density. This model is a continuous normalizing flow governed by permutation equivariant dynamics. These dynamics are driven by a learnable per-set-element term and pairwise interactions, both parametrized by deep neural networks. We illustrate the utility of this model via applications including (1) complex traffic scene generation conditioned on visually specified map information, and (2) object bounding box generation conditioned directly on images. We train our model by maximizing the expected likelihood of labeled conditional data under our flow, with the aid of a penalty that ensures the dynamics are smooth and hence efficiently solvable. Our method significantly outperforms non-permutation invariant baselines in terms of log likelihood and domain-specific metrics (offroad, collision, and combined infractions), yielding realistic samples that are difficult to distinguish from data.

1 Introduction

Invariances built into neural network architectures can exploit symmetries to create more data efficient models. While these principles have long been known in discriminative modelling (Lecun et al., 1998; Cohen & Welling, 2015; 2016; Finzi et al., 2021), in particular permutation invariance has only recently become a topic of interest in generative models (Greff et al., 2019; Locatello et al., 2020). When learning a density that should be invariant to permutations we can either incorporate permutation invariance into the architecture of our deep generative model or we can factorially augment our observations and hope that the generative model architecture is sufficiently flexible to at least approximately learn a distribution that assigns the same mass to known equivalents. The former is vastly more data efficient but places restrictions on the kinds of architectures that can be utilized, which might lead one to worry about performance limitations. While the latter does allow unrestricted architectures it is often so data-inefficient that, despite the advantage of fewer limitations, achieving good performance is extremely challenging, to the point of being impossible.

In this work we describe a new approach to permutation invariant *conditional* density estimation that, while architecturally restricted to achieve invariance, is demonstrably flexible enough to achieve high performance on a number of non-trivial density estimation tasks.

*Work done in part while at UBC

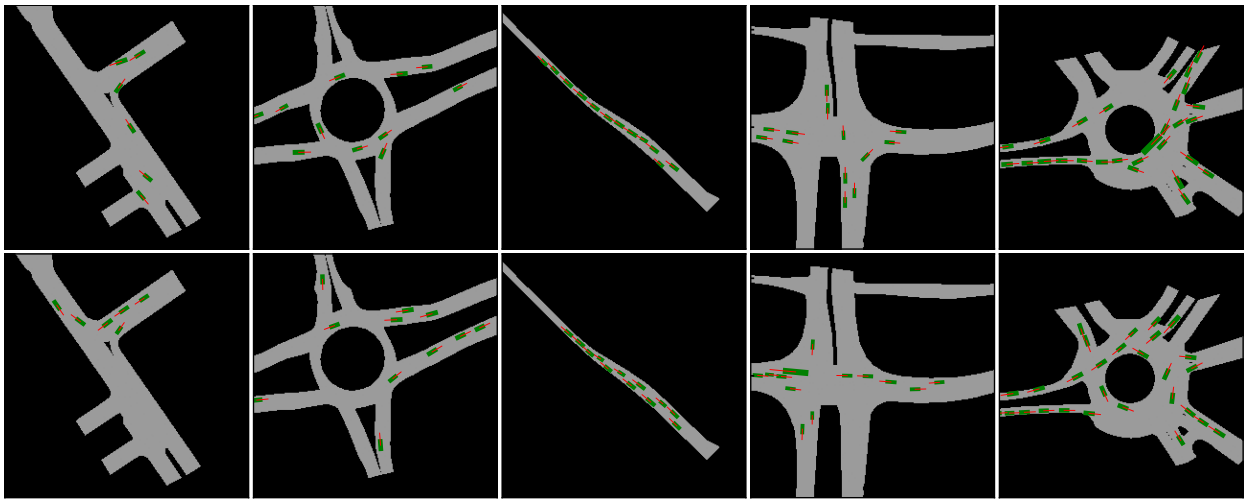

Figure 1: Realistic vehicle placement as a permutation invariant modeling problem. At every moment in time vehicles in the real world exhibit a characteristic spatial distribution of position, orientation, and size; notably vehicles (green rectangles) do not overlap, usually are correctly oriented (red lines indicate forward direction), and almost exclusively are conditionally distributed so as to be present only in driving lanes (shown in grey). The likelihood of each such arrangement does not depend on the ordering of the vehicles (permutation invariance). Each column shows a particular map with vehicle positions from real training data and from infraction free samples drawn from our permutation invariant flow conditioned on the map image. Note that because the image indicates lanes, not drivable area, the training data includes examples of vehicles that hang over into the black. We invite the reader to guess which image in each column is real and which is generated by our model. The answer appears in a footnote at the end of the paper.[2]

Permutation invariant distributions, where the likelihood of a collection of objects does not change if they are re-ordered, appear widely. The joint distribution of independent and identically distributed observations is for example permutation invariant, while in more complex cases the observations are no longer independent, but still exchangeable. Practical examples include the distribution of non-overlapping physical object locations in a scene, the set of potentially overlapping object bounding boxes given an image, and so forth (see Fig. 1). In all of these we know that the probability assigned to a set of such objects (i.e. locations, bounding boxes) should be invariant to the order of the objects in the joint distribution function argument list.

Recent work has addressed this problem by introducing equivariant normalizing flows (Köhler et al., 2020; Satorras et al., 2021; Biloš & Günnemann, 2021). Our work builds on theirs but differs in subtle but key ways that increase the flexibility of our models. More substantially this body of prior art focuses on *non-conditional* density estimation. The work of Satorras et al. (2021) does consider a form of implicit conditioning, where the flow is evaluated for different graph sizes. In this work we go beyond that by making the dynamics that constitute our flow dependent on a conditional input. To this end, we believe we are the first to develop *conditional* permutation invariant flows, that are *explicitly* dependent on external input.

We demonstrate our conditional permutation invariant flow on two difficult conditional density estimation tasks: realistic traffic scene generation (Fig. 1) given a map and bounding box prediction given an image. In both the set of permutation invariant objects is a set of oriented bounding boxes with additional associated semantic information such as heading. For traffic scene generation, we show that our method significantly outperforms baselines and meaningful ablations of our model on relevant domain metrics. For the object detection task, we are the first to present a method with a tractable likelihood. Since there are to the best of our knowledge no existing benchmark tasks for conditional set generation, as a part of our contributions we have released code to recreate the datasets and metrics in our work.[1]

[1] https://github.com/inverted-ai/conditional-permutation-invariant-flows-datasets

1.1 Background

1.1.1 Normalizing flows

Normalizing Flows (Tabak & Vanden-Eijnden, 2010; Tabak & Turner, 2013; Rezende & Mohamed, 2015) are probability distributions that are constructed by combining a simple base distribution $p_{\mathbf{z}}(\mathbf{z})$ (e.g., a standard normal) and a differentiable transformation T with differentiable inverse, that maps $\mathbf{z}$ to a variable $\mathbf{x}$

$$\mathbf{x} = T^{-1}(\mathbf{z}). \tag{1}$$

We can then express the density using the change of variables formula

$$p_{\mathbf{x}}(\mathbf{x}) = p_{\mathbf{z}}(T(\mathbf{x})) \left| \det \frac{\partial T^{-1}(\mathbf{z})}{\partial \mathbf{z}} \Big|_{\mathbf{z}=T(\mathbf{x})} \right|^{-1}, \tag{2}$$

where p denotes the respective densities over variables $\mathbf{x}$ and $\mathbf{z}$ connected by transformation T with inverse T^{-1}. The transformation T can be parametrized and used to approximate some distribution over data $\mathbf{x} \sim \pi$ by maximizing the likelihood of this data under the approximate distribution using gradient descent. An important feature distinguishing normalizing flows from other models is that in addition to a method to generate samples they provide a tractable log density, enabling maximum likelihood training and outlier detection among others. This formulation, while powerful, has two noteworthy design challenges: the right hand side of Eq. (2) has to be efficiently evaluable and the aforementioned requirement that T be invertible. The approach in the field generally is to define a chain of transformations $T_0 \circ \cdots \circ T_n$, each of which satisfy both conditions. In this manner, they can be comparatively simple, yet when joined together provide a flexible approximating family.

1.1.2 Continuous normalizing flows

Continuous normalizing flows were first introduced in Chen et al. (2018), and then further developed in Grathwohl et al. (2019). The concept is to use a continuous transformation of variables, described by dynamics function $\mathbf{v}$ parametrized by t in the form of an ordinary differential equation (ODE)

$$\mathbf{x}(t_1) = \mathbf{x}(t_0) + \int_{t_0}^{t_1} \mathbf{v}_\theta(\mathbf{x}(t), t) dt. \tag{3}$$

We set $\mathbf{x}(t_0) = \mathbf{z}$ and $\mathbf{x}(t_1) = \mathbf{x}$, so that Eq. (3) provides our definition of T^{-1} as defined in Eq. (1). Similarly, if we integrate backward in time from t_1 to t_0 we obtain T. The dynamics $\mathbf{v}_\theta(\mathbf{x}(\mathbf{t}), t)$ can be represented by a flexible function. As long as the dynamics function is uniformly Lipschitz continuous in $\mathbf{x}$ and uniformly continuous in t, the solution to the ODE is unique, and the transformation is invertible (Coddington & Levinson, 1955). In this case, we can write the probability density as another ODE (Grathwohl et al., 2019)

$$\frac{\mathrm{d} \log p_t(\mathbf{x}(t))}{\mathrm{d}t} = -\nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta(\mathbf{x}(t), t). \tag{4}$$

The term on the right hand side is the *divergence* (not gradient) of the dynamics (sometimes equivalently written as the trace of the Jacobian, note that $\mathbf{v}_\theta$ is vector valued in Eq. (4)). Integrating this ODE from the probability density at t_0 gives the density at t_1

$$\log p_{t_1}(\mathbf{x}(t_1)) = \log p_{t_0}(\mathbf{x}(t_0)) - \int_{t_0}^{t_1} \nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta(\mathbf{x}(t), t) dt. \tag{5}$$

Eq. (5) is the equivalent of Eq. (2) for continuous normalizing flows. Together with a suitable base distribution (e.g. a standard normal), the above transformation constitutes a distribution with a tractable likelihood and generative mechanism, which we will exploit to construct our flows.

1.1.3 Invariance and equivariance

We seek to construct distributions that have a permutation invariant density via permutation equivariant transformations. We state here the definition of permutation invariance and equivariance we adopt.

Definition 1. Let $\mathbf{x} = (\mathbf{x}_1 \dots \mathbf{x}_N)$ where each $\mathbf{x}_n \in \mathbb{R}^D$, and let permutations σ act on $\mathbf{x}$ via

$$\sigma \mathbf{x} = (\mathbf{x}_{\sigma_1} \dots \mathbf{x}_{\sigma_N}) \,. \tag{6}$$

A function $G : \mathbb{R}^{N \times D} \to \mathbb{R}$ is permutation *invariant* if for any permutation σ,

$$\forall \, \mathbf{x} \in \mathbb{R}^{N \times D}, \quad G(\sigma \mathbf{x}) = G(\mathbf{x}). \tag{7}$$

A function $F : \mathbb{R}^{N \times D} \to \mathbb{R}^{N \times D}$ is permutation *equivariant* if for any permutation σ,

$$\forall \, \mathbf{x} \in \mathbb{R}^{N \times D}, \quad F(\sigma \mathbf{x}) = \sigma F(\mathbf{x}). \tag{8}$$

In this work we aim to present a distribution, with a density $p(x)$ that is *permutation invariant*, and a score $\nabla_x p(x)$ that is *permutation equivariant*.

1.2 Related work

Permutation invariant models have been studied in the literature for some time, although most research has been focused on supervised learning. Examples include models of sets (Zaheer et al., 2017; Ilse et al., 2018; Lee et al., 2019), and graphs (Duvenaud et al., 2015; Kipf & Welling, 2017; Kipf et al., 2018). A particular application of permutation invariant architectures is modeling properties of datasets, or "bags" of objects (Ilse et al., 2018). Recently, also generative models for sets have made an appearance, (Zhang et al., 2019; Burgess et al., 2019; Greff et al., 2019; Locatello et al., 2020; Zhang et al., 2020). Our conditional permutation invariant flows belong to the larger class of generative models, such as variational autoencoders (Kingma & Welling, 2014), generative adversarial networks (Goodfellow et al., 2014), and normalizing flows (Rezende & Mohamed, 2015), the latter being the only class of models that enables exact likelihood evaluation. There has been prior work that focuses on coditioning normalizing flows (Winkler et al., 2019), although their focus was on images instead of the set valued data we study here.

Other work belonging to the equivariant generative category is "Equivariant Hamiltonian flows" (Rezende et al., 2019), which relates to our work since it models interactions elements of a set using Hamiltonian dynamics. The choice of these dynamics allows the use of a symplectic integrator, and the transformation is volume conserving, eliminating the need to integrate a divergence term. However, this requires the introduction of a set of momentum variables that preclude the exact calculation of a density. In Liu et al. (2019), the authors present a flow for graphs with tractable density. Although the target domain is similar, their flow differs from ours as it is based on the mechanism developed in RealNVP (Dinh et al., 2017), whereas our flow uses continuous normalizing flows. Recently, score based methods have also been studied to generate graphs (Niu et al., 2020), including edge features.

Our work is strongly related to, and draws inspiration from recent work that uses continuous normalizing flows with permutation invariant dynamics (Köhler et al., 2020; Satorras et al., 2021; Biloš & Günnemann, 2021). However Köhler et al. (2020) and Satorras et al. (2021) focus also on rotation and translation invariance, in order to model molecular graphs. Our work is also related to PointFlow, a continuous normalizing flow for point clouds (Li et al., 2021). We focus on sets like in Biloš & Günnemann (2021) and Li et al. (2021), however these studies focus on reducing evaluation cost for large set sizes. Our dynamics function on the other hand focuses on strong interactions between set elements, more akin to Satorras et al. (2021) and Köhler et al. (2020). Importantly, none of the existing work on graphs and sets considers the viability of *conditioning* on external inputs and learning a distribution that is able to deal with a varying conditional input distribution.

2 Conditional permutation invariant flows

2.1 Permutation invariant flows

In this work, we will construct normalizing flows that are characterized by a permutation equivariant transformation $T(\sigma\mathbf{x}) = \sigma T(\mathbf{x})$; we will demonstrate these flows produce a permutation invariant density $p(\mathbf{x}) = p(\sigma\mathbf{x})$. We construct our permutation invariant flows using a dynamics function that is based on a global force term and pairwise interaction terms

$$\mathbf{v}_{\theta,i}(\mathbf{x}) = \sum_{j,j\neq i} f_\theta(\mathbf{x}_i, \mathbf{x}_j) + g_\theta(\mathbf{x}_i). \tag{9}$$

Here, $\mathbf{v}_{\theta,i}(\mathbf{x}) \in \mathbb{R}^D$ denotes the ith element of $\mathbf{v}_\theta$, $\mathbf{x} \in \mathbb{R}^{N\times D}$, $g_\theta : \mathbb{R}^D \to \mathbb{R}^D$, and $f_\theta : \mathbb{R}^{2D} \to \mathbb{R}^D$. This construction can be interpreted as objects $\mathbf{x}_i$ moving in a global potential with corresponding force field $g_\theta(\mathbf{x}_i)$, and interacting with other objects through the pairwise interaction $f_\theta(\mathbf{x}_i, \mathbf{x}_j)$. We proceed to construct a continuous normalizing flow using the function described in Eq. (9) as the dynamics. If we use a permutation invariant base distribution $p(\mathbf{x}(t_0)) = p(\mathbf{z})$ we obtain the following:

Theorem 1. *If the transformation $\mathbf{z} = T(\mathbf{x})$ defined in Eq. (3) has dynamics $\mathbf{v}_\theta(\mathbf{x})$ defined in Eq. (9), then T is permutation equivariant. If in addition $p(\mathbf{z})$ is permutation invariant, then the density $p(\mathbf{x})$ is permutation invariant.*

The proof of Theorem 1 is given in Appendix A.1. We note that this theorem can be seen as a special case of theorems presented in previous work (Köhler et al., 2020; Zaheer et al., 2017), but add it here for a complete exposition. The dynamical system this theorem represents is similar to an interacting set of particles in a global potential. The dynamics as presented in Eq. (9) are independent of time; in a few cases, however, we have found it useful to make the dynamics time-dependent, i.e. $\mathbf{v}_\theta(\mathbf{x}, t)$, by passing time to both g_θ and f_θ as an input. A complete overview of when time dependence is used is given in Appendix A.2.3. In practice we represent f_θ and g_θ by neural networks, which satisfy the criterion of uniform Lipschitz continuity if activation functions are chosen appropriately, thereby guaranteeing invertibility. Implementation details can be found in Appendix A.2.1. Throughout this work we will choose $p(\mathbf{z})$ to be a product of standard normal Gaussians over all dimensions unless specifically stated otherwise. A diagonal Gaussian satisfies the permutation invariant requirement in Theorem 1.

An important detail to point out is that $\mathbf{v}_\theta(\mathbf{x})$ is full rank, and therefore the distribution described by it is not simply of dimension D, but rather a distribution over $D \times N$ dimensions. This is an important distinction, as it highlights the ability of our flows to model interactions between set elements, which precludes a description in terms of a D dimensional distribution.

2.2 Divergence

To compute the density at time t in Eq. (4), we need the divergence of the dynamics in Eq. (9). We can express this divergence as

$$\nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta(\mathbf{x}) = \sum_{i,j,j\neq i} \nabla_{\mathbf{x}_i} \cdot f_\theta(\mathbf{x}_i, \mathbf{x}_j) + \sum_i \nabla_{\mathbf{x}_i} \cdot g_\theta(\mathbf{x}_i). \tag{10}$$

A naïve computation of the divergence in Eq. (4) using automatic differentiation is expensive, as computing the Jacobian requires ND evaluations, one for each of the ND terms in $\mathbf{v}_\theta$ (Chen et al., 2018; Grathwohl et al., 2019). Since the cost of evaluating Eq. (9) is quadratic in N, this would result in an asymptotic computational cost of N^3D^2 for the forward and backward pass. Earlier work has suggested the use of the Hutchinson's trace estimator (Chen et al., 2018) for the divergence, which reduces the cost of the divergence to that of $\mathbf{v}$, but suffers from high variance (Chen & Duvenaud, 2019). Instead we opt to re-express the divergence of $\mathbf{v}_\theta$ in terms of derivatives of $f(\mathbf{x}_i, \mathbf{x_j})$ and $g(\mathbf{x}_i)$, resulting in Eq. (10). The form in Eq. (10) is quadratic in N, and therefore same cost in N as the evaluation of $\mathbf{v}_\theta$, both in the forward and backward pass.

2.3 Regularization

Continuous normalizing flows have no inherent mechanism that penalizes very complex dynamics. While in theory there is no reason to prefer simple dynamics, in practice, the numerical integration of complex dynamics can result in long computation when using an adaptive scheme. This effect has been previously observed in the literature, and suggestions to regularize the dynamics have been proposed in previous work (Finlay et al., 2020; Kelly et al., 2020). While the work of Kelly et al. (2020) is more comprehensive, we find that an adaptation of the solution proposed in Finlay et al. (2020) works well for our purposes. The proposed solution in Finlay et al. (2020) is to add a term proportional to the squared Frobenius norm of the Jacobian, and the ℓ^2-norm of the dynamics. We use the ℓ^2-norm for the dynamics; however, since we do not estimate the full Jacobian we calculate

$$\ell^2_{div} = \sum_{i \neq j, d, d'} \left(\frac{\partial f_d(x_{id}, x_{jd})}{\partial x_{id'}} \right)^2 + \sum_{i, d, d'} \left(\frac{\partial g_d(x_{id})}{\partial x_{id'}} \right)^2. \tag{11}$$

We find that this penalty significantly reduces the number of evaluations of our trained flows. We visualize the effect this penalty has on the dynamics in some examples in Appendix A.3.5.

2.4 Conditional permutation invariant flows

When performing amortized inference, in which a family of posterior distributions is learned, a requirement is a flexible variational family that can be made to depend (i.e. *conditioned*) on external input. An example would be to produce a valid distribution of a set of bounding box locations and sizes $\mathbf{x}$ for objects in an image $\mathbf{y}$ selected from a distribution of images. We will denote the conditioning input as $\mathbf{y}$, coming from some data distribution $\pi(\mathbf{y})$. To model such cases we construct a dynamics function that depends on $\mathbf{y}$ by modifying the pair forces, and the global force to $f_\theta(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y})$ and $g_\theta(\mathbf{x}_i, \mathbf{y})$. The dynamics then become:

$$\mathbf{v}_{\theta,i}(\mathbf{x}, \mathbf{y}) = \sum_{j \setminus i} f_\theta(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y}) + g_\theta(\mathbf{x}_i, \mathbf{y}). \tag{12}$$

Note that here too, the dynamics can be made time dependent by passing time t as an argument. In practice, we condition the dynamics $f_\theta(\mathbf{x}_i, \mathbf{x}_j, \mathbf{y})$ and $g_\theta(\mathbf{x}_i, \mathbf{y})$ on $\mathbf{y}$ by passing $\mathbf{y}$ through a convolutional neural net, and concatenating this to the second layer of the neural net representing f_θ and g_θ. While other ways of conditioning are possible, we empirically found this to give the best results for the tasks considered here.

We will train our flows by minimizing the Kullback-Leibler divergence to the joint distribution $\pi(\mathbf{x}, \mathbf{y}) = \pi(\mathbf{x}|\mathbf{y})\pi(\mathbf{y})$ over data $\mathbf{x}$ and condition $\mathbf{y}$:

$$\arg\min_\theta \mathrm{D}_{\mathrm{KL}} \left(\pi(\mathbf{x}, \mathbf{y}) || p_\theta(\mathbf{x}|\mathbf{y})\pi(\mathbf{y}) \right) = \arg\min_\theta \mathop{\mathbb{E}}_{\mathbf{y} \sim \pi(\mathbf{y})} \mathrm{D}_{\mathrm{KL}} \left(\pi(\mathbf{x}|\mathbf{y}) || p_\theta(\mathbf{x}|\mathbf{y}) \right) \right). \tag{13}$$

In other words, we optimize our flow to match the distribution of $\mathbf{x}$ in *expectation* over $\pi(\mathbf{y})$.

3 Experiments

3.1 Synthetic examples

We start our experiments with two pedagogical examples that demonstrate the capabilities and mechanisms of our flows. The first example task is to model the spatial distribution of five *non-overlapping* squares of width $w = 1$, that furthermore do not overlap with the prohibited regions shown in blue. This example is representative of placing assets into a physically realizable configuration in accordance with constraints imposed by an environment. We fit our conditional flow to a dataset generated by first sampling a prohibited region—three boxes of width $w = 1.5$ from a standard normal prior—and then sampling box locations independently from a standard normal prior, with rejection for overlap with previous boxes or the prohibited region, until a total of five boxes are sampled. The prohibited region is input to our conditional flow as an image tensor. Since the dataset was generated via rejection sampling, we can compare our sample efficiency

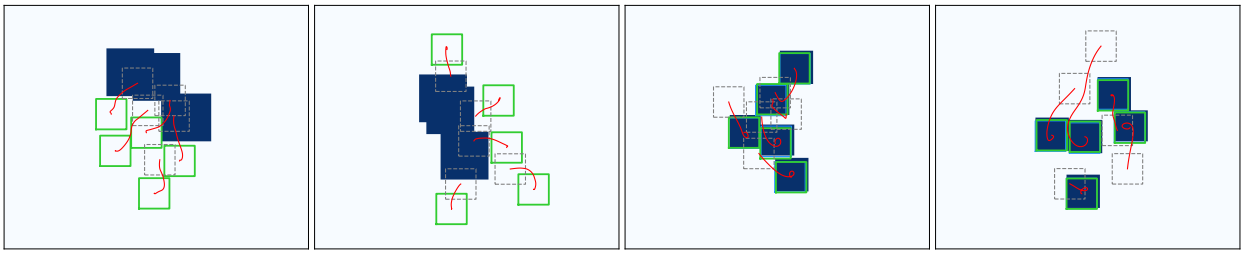

Figure 2: Two pedagogical permutation invariant modeling tasks. The left two panels illustrate the first task; *conditionally* modeling non-overlapping squares (green), which also do not overlap with the blue boxes whose arrangement varies between data points. The right two panels illustrate the second task; modeling boxes that are conditionally distributed so as to bound the underlying blue boxes. Samples from the base $p(\mathbf{z})$ and final distribution $p_\theta(\mathbf{x}|\mathbf{y})$ are plotted in dashed grey and green lines respectively. The conditional input is plotted as a blue on white image. Red lines indicate the trajectories the objects follow by integrating the dynamics function $\mathbf{v}(\mathbf{x}(t), \mathbf{y})$.

against it. The conditional flow provides a substantial sampling efficiency improvement (77% valid) over rejection sampling from the prior (0.02% valid), in addition to a providing a tractable density.

The second example task is bounding box prediction, or conditionally generating object bounding boxes $\mathbf{x}$ directly from an image $\mathbf{y}$. Here the objects are monochrome blue squares. Data is generated in a similar manner as in the first experiment: squares are sampled independently from a standard normal prior, and rejected if they overlap. The conditional input is an image of the generated boxes. Sampled bounding boxes from our trained flow achieve an average intersection over union (IOU) of 0.85 with the ground truth bounding boxes.

We note that in both examples the dense interaction in Eq. (9) is a requirement for reasonable results. Interactions between squares need to be modeled avoid overlaps in the first example, and to assure coverage of all objects in the second. This produces a ten-dimensional distribution in both cases, rather than a two-dimensional one.

We display representative samples and their trajectories through time in Fig. 2 for both experiments. In the left two panels it can be seen that initial samples are transported around the space to avoid one another; in the right two, the boxes coordinate through the pairwise interactions to each surround exactly one of the objects in the scene. Additional experiments on different set cardinalities, and further details for the ones provided here appear in Appendix A.3.

3.2 Traffic scenes

Modeling and being able to sample realistic traffic scenes is an essential task related to autonomous driving simulation and control. Referring back to Fig. 1, the problem—similar to the first pedagogical task above—is one of modeling the physical configuration a collection of agents conditioned on a representation of the environment. Until recently, the predominant methods for generating realistic vehicle configurations were rule-based (Yang & Koutsopoulos, 1996; Lopez et al., 2018). Rule-based systems can be tailored to have desirable properties such as avoiding occurrences of offroad and vehicle overlap, but they produce vehicle arrangements that are distributionally dissimilar to real data.

Recent work addressing this problem uses a non-rule-based autoregressive model (Tan et al., 2021) that enables sequential generation of vehicle and agent positions conditioned on a visual representation of a map. While this model-based approach closes the gap between simulation and reality, modeling sets autoregressively introduces a factorial data augmentation requirement, as there is no intrinsic ordering of actors. The authors of Tan et al. (2021) avoid this by imposing an arbitrary order, sampling agents from left to right. Our experiments indicate that, at least for this specific task, directly addressing permutation invariance is preferred, and avoids the need to arbitrarily fix the order of elements.

Table 1: Quantitative results for traffic scene generation. NLL indicates negative log-likelihood in nats, while the other metrics indicate the fraction of samples exhibiting offroad, collision, or either (lower is better).

Method	NLL	Offroad	Collision	Infraction
Gaussian	46.3 ± 0.0	0.99 ± 0.00	0.27 ± 0.00	0.99 ± 0.00
RealNVP	30.4 ± 1.4	0.96 ± 0.01	0.26 ± 0.00	0.98 ± 0.00
CNF	20.8 ± 0.9	0.86 ± 0.01	0.33 ± 0.00	0.92 ± 0.01
Autoregr.	11.0 ± 1.7	0.72 ± 0.02	0.14 ± 0.01	0.76 ± 0.02
PIF Single	7.3 ± 0.2	**0.09 ± 0.01**	0.57 ± 0.00	0.61 ± 0.01
PIF Pair	**6.3 ± 0.3**	0.17 ± 0.01	0.18 ± 0.02	0.32 ± 0.02
PIF Pair MF	7.0 ± 0.1	0.11 ± 0.02	0.56 ± 0.01	0.61 ± 0.00
PIF (ours)	**6.5 ± 0.3**	0.12 ± 0.01	0.09 ± 0.01	**0.20 ± 0.01**
Cond. single	7.1 ± 0.3	0.11 ± 0.01	0.12 ± 0.00	0.22 ± 0.01
Cond. pair	7.2 ± 0.1	0.18 ± 0.02	0.17 ± 0.03	0.32 ± 0.03
Cond. base	16.5 ± 2.2	0.91 ± 0.01	**0.05 ± 0.01**	0.92 ± 0.01

To test the performance of our flows on this task, we train them to generate a scene of cars in the INTERACTION dataset (Zhan et al., 2019), conditioned on a rendered image of the drivable area $\mathbf{y}$. The properties that our flows predict are two-dimensional position, size, aspect ratio and orientation for each of N agents, i.e. $\mathbf{x} \in \mathbb{R}^{N \times 5}$. We remind the reader that the distribution modeled here is not D dimensional, but rather $D \times N$ dimensional. This is an important distinction, since it allows us to model interactions between set elements, a crucial ingredient to avoid overlap between agents. A direct advantage of the formulation of the dynamics in Eq. (12) is that they can be applied with the same f and g regardless of the number of agents N. We make use of this property, and train a single model on a varying amount of agents N. At test time, generating N agents is accomplished simply by initializing the flow with a base distribution sample $\mathbf{z}$ of appropriate size.

We train our flows until the likelihood of the data stops increasing, or the likelihood of a held out validation sets starts decreasing, whichever comes fist. Examples of the data we train on, and representative samples from our trained flow are shown in Fig. 1.

3.2.1 Baselines

We compare our conditional flows against several baselines. Three are non-permutation invariant flows: a unimodal Gaussian model, a RealNVP based model (Dinh et al., 2017), and a "vanilla" continuous normalizing flow ("CNF"). We also implement an autoregressive model consisting of a convolutional neural net paired with a recurrent neural network and a 10 component Gaussian mixture for every prediction component, and adopt the canonical ordering discussed in Tan et al. (2021). We also test two ablations of our model: one where the dynamics are restricted only to single particle terms g_θ ("PIF Single"), and one where the dynamics only include the pair terms f_θ ("PIF Pair"). The flow where the dynamics are restricted to single particle terms only corresponds to a conditional version of PointFlow (Li et al., 2021). We also compare against a conditional version of the model presented in Biloš & Günnemann (2021) ("PF Pair MF"), in which interactions are addressed, but only in terms of a mean field (being cheaper to calculate). Both these methods result in a significantly worse collision and resulting infraction rate.

To compare to non-permutation invariant methods, we have to fix the number of agents, as these architectures cannot straightforwardly be provisioned to generate and score sets that differ in cardinality. We exclude all data with less than seven agents, and cases with more than seven agents are pruned to retain only the agents closest to the center. The cardinality of seven was chosen to retain as much data as possible while not making each individual scene too small. Furthermore, we restricted the INTERACTION dataset to the roundabout scenes in order to better match the seven agent target while still maintaining a semantically similar set of possible $\mathbf{y}$. We compare the negative log likelihood (NLL) of the various models on a held-out test dataset. For these traffic scenarios there are two other useful metrics we can compare: the fraction of

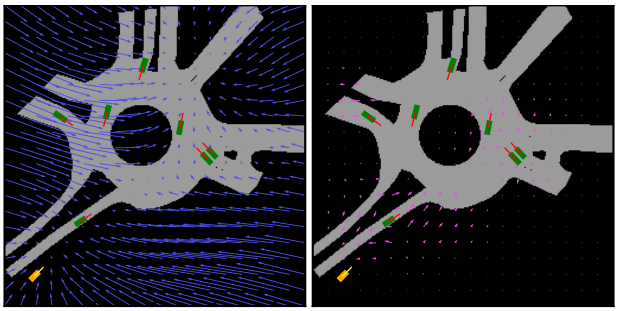

Figure 3: Vector field describing the dynamics for the spatial dimensions of a hypothetical agent of fixed size and orientation (in orange, bottom left), as a function of position. Contributions of g_θ (left panel) and f_θ (right panel) are separated.

offroad (i.e. an agent is on the undrivable area), collision (i.e., an agent overlaps with another agent) and total infractions (offroad or collision) the model makes. We note that these metrics are sensitive indicators of model fitness, in the sense that the training data contains no offroad or colliding data examples. Samples that exhibit these infractions are evidence of model error, and additionally inform whether modeling mistakes are made globally (i.e. offroad) or through interactions (i.e. collisions). We report our results in Table 1.

Comparing the likelihood and infraction metrics demonstrates the clear advantage of using a permutation invariant model. The non-permutation invariant varieties of flows, do not converge to a competitive likelihood, and struggle to generate infraction-free examples. Although the canonically ordered autoregressive model is much more capable than the non-permutation invariant flows, it still underperforms compared to ours. The two ablations of our model provide an insightful result: the function of g_θ is to define the collective behavior of the agents (all of them need to stay on the road, independently of one another), whereas f_θ provides the necessary interaction between them (agents should not collide with one-another). These functions are evident from the respective infraction metrics. Furthermore, there appears to be a certain amount of competition between the pair and single terms: as the agents are steered onto the road, they have a higher density and thus a higher chance to collide. The opposite is equally true, as repelling agents can push each other off-road. As such it is not terribly surprising that the "single-only" (g_θ-based) flow performs better when only considering offroad infractions. Nevertheless, the combined flow has the best performance overall, both in overall infraction rate and negative log likelihood. To gain an even better understanding of the role of the two functions f_θ and g_θ, in Fig. 3 we present the generated (time independent) vector field for the positional coordinates, as a function of position for a hypothetical agent with fixed orientation and typical size (this orientation and size are displayed in orange in the bottom left). The contribution of g_θ shown in the left panel clearly substantiates the earlier claims that this part of the dynamics represents the collective behavior: it pushes the agent to sensible locations on the map, given its orientation. The agent in the example is pointed north-east, and the flow points only toward north-east facing roads (see Appendix A.3.4 for plots at different orientations). The contribution of f_θ on the other hand is largest around other agents, flowing away from their positions. Note that the flow direction is dependent on the map location, and whether space is available to flow to. Note also that the vector field for this hypothetical agent is not two- but five-dimensional; we simply drop the last three dimensions (size, aspect ratio and orientation) for ease of presentation. Moreover, the vector field is *different* by virtue of f_θ for each of the N agents in Fig. 3, reflecting the overall $N \times D$ dimensionality. These presented vector fields provide further support for what the collision and offroad metrics already hinted at: both f_θ and g_θ are necessary for optimal performance.

To better understand the effects of our conditioning inputs, we report the performance of two ablations, one where the conditional input is only used in g_θ ("Cond. Single"), and one where it is only used in f_θ ("Cond. Pair"). In the former, the interactions between actors are independent of the scene, which one may expect to be a reasonable approximation. However, the results in Table 1 indicate that this input is in fact important, which may be understood from the fact that different locations lead to different traffic configurations. In the case where conditioning on g_θ is ignored, we also do worse than conditioning both, while surprisingly maintaining a fairly competitive result. Finally, we show a variant of our model where

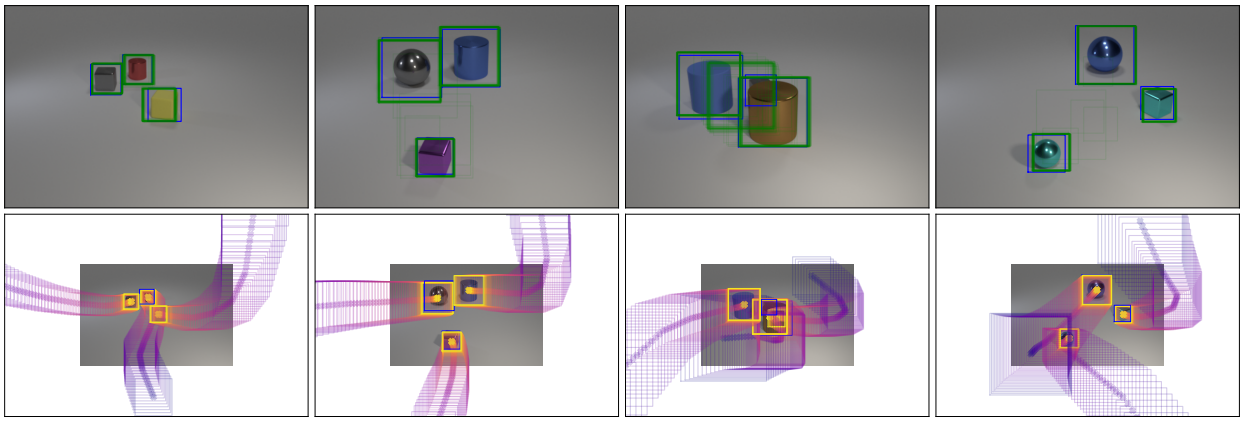

Figure 4: Bounding box prediction on CLEVR3 images. Each image shows 50 samples from our conditional flow (green, ground truth in blue), conditioned on the background image. The bottom row shows the trajectories of the boxes with time along the trajectory encoded by the color of the box.

Table 2: Quantitative results for bounding box prediction. IOU refers to the intersection over union of object area covered by the samples (higher is better).

	Dataset IOU					
Method	CLEVR3	CLEVR6-3	CLEVR6-3	CLEVR6-3	CLEVR6-6	CLEVR6
PIF (ours)	0.732	0.759	0.711	0.644	0.596	0.679
DSPN		0.89	0.88	0.84	0.83	0.86

we condition the base distribution (i.e. $p_\theta(\mathbf{z}|\mathbf{y})$ vs. $p(\mathbf{z})$), but otherwise remove the conditional dependence from f_θ and g_θ. In this ablation, our base distribution takes the form $p(\mathbf{z}) = \prod_i^N p(\mathbf{z}_i)$, with $\mathbf{z}_i \in \mathbb{R}^D$ and $p(\mathbf{z}_i) = \mathcal{N}(\mathbf{z}_i; \mu_\theta(\mathbf{y}), \mathbf{I}\sigma_\theta(\mathbf{y}))$. This satisfies the permutation invariance requirement for $p(\mathbf{z})$ described in Theorem 1. Although in this case the collision rate is lowest, this can be attributed to the poor performance with respect to offroad infractions. Overall, this variation fails to provide a competitive result.

3.2.2 Variable set size

Since our model is trained on a variety of different set sizes, and performs well for each of the different set sizes, we can investigate whether the it generalizes beyond the number of agents it has seen during training. We therefore generate samples in our roundabouts model with a previously unseen number of agents. Such samples are presented in the last column of Fig. 1, which have 28 agents, while the maximum number of agents present in the training data is 22. These results indicate the the inductive bias of the representation in Eq. (12) not only performs well in sample with respect to set size, but generalizes to larger set sizes too.

3.3 Bounding box prediction

Our final experiment considers bounding box prediction. The gold standard in this sub-problem of object detection remains so-called "non-max-suppression," in which a large number of putative bounding boxes are scene-conditionally generated and then "pruned" by a greedy selection algorithm (Girshick et al., 2014; Ren et al., 2015). Some very recent studies (Hu et al., 2018; Zhang et al., 2019; Carion et al., 2020) have proposed the use of conditional set generators for object detection. We build on this idea by illustrating how our flow can be used to conditionally generate and score sets of object bounding boxes. Being able to compute the log density of a configuration of bounding boxes eliminates the need for a differentiable matching algorithm (Hu et al., 2018; Carion et al., 2020). Having a tractable density also opens up uncertainty-preserving approaches to downstream tasks such as outlier detection and inferring object count through the use of Bayes factors.

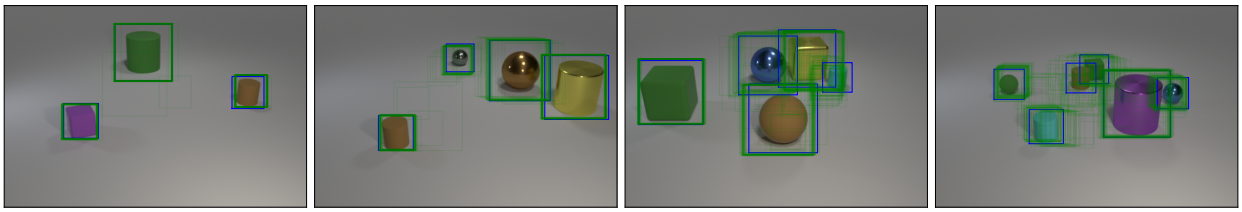

Figure 5: Bounding box prediction on CLEVR6 images. Each image shows 50 samples (green, ground truth blue) from our flow conditioned on the corresponding background image.

We assess the viability of our approach through the CLEVR (Johnson et al., 2017) dataset, which is a standard benchmark for set-generation models (Greff et al., 2019; Zhang et al., 2019; Locatello et al., 2020). While this former work combines localization with classification (i.e., predicting object *position* and *type*), we focus on the task of bounding box prediction (i.e predicting object *position* and *size*). The CLEVR dataset does not provide bounding boxes, so we generate ground truth bounding boxes from object metadata. The full details on how to create bounding boxes for the CLEVR dataset are described in Appendix A.4.1.

We begin with a subset of the CLEVR dataset only containing three objects. We find that our flows perform well on this task, and show example predictions in Fig. 4. For each conditional image (displayed in the background), 50 samples from the conditional distribution are shown, graphically illustrating the variance of the conditional density. For each column, the bottom row displays the trajectory taken to generate a single one of these samples. The trajectories show the interactions between the bounding boxes over time, as they coordinate through the use of repulsive forces. Note that by sorting out which of the base distribution samples goes to which of the objects, the flow solves the assignment problem along the way. It is worth pointing out that the third sample has an occluded object, and the variance of the object position is clearly higher than that for objects where there is no occlusion, which we take to be evidence that these flows for bounding box prediction should be useful in uncertainty aware downstream applications. Importantly, the variance of the size is not significantly increased, which is correct behavior for this example. Additional samples are presented in the appendix. The averaged IOU with the ground truth is reported in Table 2, and corresponds approximately to an average mismatch of 15% in each spatial dimension.

We continue our exploration with a larger subset of the CLEVR dataset, including images that have between three and six objects. This subset has also been used in Locatello et al. (2020) for object detection. We assume that the number of objects is given, and only predict the bounding box locations and sizes given the number of bounding boxes and the image. Some example samples are presented in Fig. 5, with green boxes respresenting samples from the distribution that is conditioned on the image in the background. More samples are presented in Appendix A.3.6. The flow generalizes well over set cardinalities, hinting that some generalizing principles are learned by the flow about these bounding boxes interact, even with different set size. We moreover see that the more crowded the image becomes, the more spread there is in the predicted bounding boxes, representing increased uncertainty about object sizes and positions, also representing more occlusion. The overall average IOU for both tasks ("3" and "6"), as well as the IOU's separated by test set cardinality ("6-3", etc.) are given in Table 2. These results show that the flow trained on data with variable set size performs marginally better on the CLEVR3 subset than a flow trained only on that data, which we speculate is due to the larger amount of available data. A modest decrease in IOU can be observed as the sets become larger, resulting in an overall performance that is slightly lower. Table 2 also provides baseline results of Deep Set Prediction Networks (DSPN, see Appendix A.4.2) (Zhang et al., 2019). Their method outperforms ours in terms of intersection over union, but their method does not provide a tractable log density, hence limiting the comparability of the two methods.

[2]By column from left to right, the flow samples are: top, top, top, bottom, both. For the last column in particular, our flow is able to produce more vehicles than ever appeared on that map in the training data.

4 Discussion and conclusions

This work introduced conditional permutation invariant flows, a framework built on continuous normalizing flows that enables conditional set generation with a tractable density. We have applied our flows to two problems: realistic traffic scene generation, and bounding box prediction. For traffic scene generation, we significantly outperform baselines, and are the first to present a permutation invariant solution. Ablations to our flows highlight their intuitive mechanism, that can be understood as objects moving jointly in a field, augmented with pairwise interaction potentials. We have moreover shown that bounding box prediction can be enhanced with a tractable log density, opening an avenue to develop downstream vision algorithms that deal with uncertainty in a more principled way.

References

Marin Biloš and Stephan Günnemann. Scalable normalizing flows for permutation invariant densities. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 957–967. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/bilos21a.html.

Christopher P. Burgess, Loïc Matthey, Nicholas Watters, Rishabh Kabra, Irina Higgins, Matthew Botvinick, and Alexander Lerchner. Monet: Unsupervised scene decomposition and representation. *CoRR*, abs/1901.11390, 2019. URL http://arxiv.org/abs/1901.11390.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (eds.), *Computer Vision – ECCV 2020*, pp. 213–229, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58452-8.

Ricky T. Q. Chen and David K Duvenaud. Neural networks with cheap differential operators. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/770f8e448d07586afbf77bb59f698587-Paper.pdf.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/69386f6bb1dfed68692a24c8686939b9-Paper.pdf.

Earl A. Coddington and Norman Levinson. *Theory of Ordinary Differential Equations*. McGraw-Hill, 1955. ISBN 978-0-07-099256-6.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2990–2999, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/cohenc16.html.

Taco S. Cohen and Max Welling. Transformation properties of learned visual representations. In Yoshua Bengio and Yann LeCun (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.7659.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=HkpbnH9lx.

J.R. Dormand and P.J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980. ISSN 0377-0427. doi: https://doi.org/10.1016/0771-050X(80)90013-3. URL https://www.sciencedirect.com/science/article/pii/0771050X80900133.

David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/f9be311e65d81a9ad8150a60844bb94c-Paper.pdf.

Chris Finlay, Jörn-Henrik Jacobsen, Levon Nurbekyan, and Adam M. Oberman. How to train your neural ode: the world of jacobian and kinetic regularization. In *ICML*, pp. 3154–3164, 2020. URL http://proceedings.mlr.press/v119/finlay20a.html.

Marc Finzi, Max Welling, and Andrew Gordon Gordon Wilson. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 3318–3328. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/finzi21a.html.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, and David Duvenaud. Scalable reversible generative models with free-form continuous dynamics. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJxgknCcK7.

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2424–2433. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/greff19a.html.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Towards non-autoregressive language models. *CoRR*, abs/2102.05379, 2021.

Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Maximilian Ilse, Jakub Tomczak, and Max Welling. Attention-based deep multiple instance learning. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2127–2136. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/ilse18a.html.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Jacob Kelly, Jesse Bettencourt, Matthew James Johnson, and David Duvenaud. Learning differential equations that are easy to solve. In *Neural Information Processing Systems*, 2020. URL https://arxiv.org/abs/2007.04504.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL http://arxiv.org/abs/1312.6114.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural relational inference for interacting systems. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2688–2697. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/kipf18a.html.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Jonas Köhler, Leon Klein, and Frank Noe. Equivariant flows: Exact likelihood generative learning for symmetric densities. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5361–5370. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/kohler20a.html.

Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Xiangtai Li, Hao He, Xia Li, Duo Li, Guangliang Cheng, Jianping Shi, Lubin Weng, Yunhai Tong, and Zhouchen Lin. Pointflow: Flowing semantics through points for aerial image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4217–4226, June 2021.

Jenny Liu, Aviral Kumar, Jimmy Ba, Jamie Kiros, and Kevin Swersky. Graph normalizing flows. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/1e44fdf9c44d7328fecc02d677ed704d-Paper.pdf.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 11525–11538. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/8511df98c02ab60aea1b2356c013bc0f-Paper.pdf.

Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wiessner. Microscopic traffic simulation using sumo. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pp. 2575–2582, 2018. doi: 10.1109/ITSC.2018.8569938.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, and Stefano Ermon. Permutation invariant graph generation via score-based generative modeling. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 4474–4484. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/niu20a.html.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1530–1538, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/rezende15.html.

Danilo Jimenez Rezende, Sébastien Racanière, Irina Higgins, and Peter Toth. Equivariant hamiltonian flows, 2019.

Victor Garcia Satorras, Emiel Hoogeboom, Fabian Bernd Fuchs, Ingmar Posner, and Max Welling. E(n) equivariant normalizing flows. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=N5hQI_RowVA.

E. G. Tabak and Cristina V. Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013. doi: https://doi.org/10.1002/cpa.21423. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.21423.

Esteban G. Tabak and Eric Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217 – 233, 2010. doi: cms/1266935020. URL https://doi.org/.

Shuhan Tan, Kelvin Wong, Shenlong Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. Scenegen: Learning to generate realistic traffic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 892–901, June 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity, 2020.

Christina Winkler, Daniel Worrall, Emiel Hoogeboom, and Max Welling. Learning likelihoods with conditional normalizing flows, 2019. URL https://arxiv.org/abs/1912.00042.

QI Yang and Haris N. Koutsopoulos. A microscopic traffic simulator for evaluation of dynamic traffic management systems. *Transportation Research Part C: Emerging Technologies*, 4(3):113–129, 1996. ISSN 0968-090X. doi: https://doi.org/10.1016/S0968-090X(96)00006-X. URL https://www.sciencedirect.com/science/article/pii/S0968090X9600006X.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/f22e4747da1aa27e363d86d40ff442fe-Paper.pdf.

Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kümmerle, Hendrik Königshof, Christoph Stiller, Arnaud de La Fortelle, and Masayoshi Tomizuka. INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv:1910.03088 [cs, eess]*, 2019.

Yan Zhang, Jonathon Hare, and Adam Prugel-Bennett. Deep set prediction networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/6e79ed05baec2754e25b4eac73a332d2-Paper.pdf.

Yan Zhang, Jonathon Hare, and Adam Prügel-Bennett. Fspool: Learning set representations with featurewise sort pooling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJgBA2VYwH.

A Appendix

A.1 Proof

We provide here the proof for Theorem 1.

Proof. We will consider equivariance of T to a transposition of elements i and j, denoted $\sigma_{i,j}$. A transposition $\sigma_{i,j}$ is a permutation for which $\sigma_i = j$, $\sigma_j = i$, and $\sigma_k = k$ for all $k \in \{1 \dots N\} \setminus \{i, j\}$. Since any permutation can be constructed from a series of transpositions, proving that T is equivariant to a transposition, trivially extends to equivariance to all permutations. In the following we have dropped the dependence on $\mathbf{y}$ and t for notational clarity. We have

$$T(\sigma_{i,j}\mathbf{x}) = \sigma_{i,j}\mathbf{x} + \int_{t_0}^{t_1} \mathbf{v}_\theta(\sigma_{i,j}\mathbf{x})dt. \tag{14}$$

The first term $\sigma_{i,j}\mathbf{x}$ trivially satisfies the equivariance condition. Focusing on the ith term of the dynamics function $\mathbf{v}_{\theta,i}$

$$\begin{aligned}
\mathbf{v}_{\theta,i}(\sigma_{i,j}\mathbf{x}) &= \sum_{k\setminus\{i\}} f_\theta\left((\sigma_{i,j}\mathbf{x})_i, (\sigma_{i,j}\mathbf{x})_k\right) + g_\theta\left((\sigma_{i,j}\mathbf{x})_i\right) && (15)\\
&= \sum_{k\setminus\{i,j\}} f_\theta\left((\sigma_{i,j}\mathbf{x})_i, (\sigma_{i,j}\mathbf{x})_k\right) + f_\theta\left((\sigma_{i,j}\mathbf{x})_{iv}, (\sigma_{i,j}\mathbf{x})_j\right) + g_\theta\left(\mathbf{x}_j\right) && (16)\\
&= \sum_{k\setminus\{i,j\}} f_\theta\left(\mathbf{x}_j, \mathbf{x}_k\right) + f_\theta\left(\mathbf{x}_j, \mathbf{x}_i\right) + g_\theta\left(\mathbf{x}_j\right) && (17)\\
&= \sum_{k\setminus\{j\}} f_\theta\left(\mathbf{x}_j, \mathbf{x}_k\right) + g_\theta\left(\mathbf{x}_j\right) && (18)\\
&= \mathbf{v}_{\theta,j}\left(\mathbf{x}\right) = \left(\sigma_{i,j}\mathbf{v}_\theta(\mathbf{x})\right)_i, && (19)
\end{aligned}$$

thus demonstrating the dynamics are equivariant and

$$T(\sigma_{i,j}\mathbf{x}) = \sigma_{i,j}\mathbf{x} + \int_{t_0}^{t_1} \sigma_{i,j}\mathbf{v}_\theta(\mathbf{x})dt = \sigma_{i,j}T(\mathbf{x}). \tag{20}$$

For continuous normalizing flows, the inverse transformation $T^{-1}(\mathbf{x})$ is obtained by reversing the integration limits, and an identical derivation can be made to show $T^{-1}(\mathbf{x})$ is also equivariant. For the density, we have

$$\log p_{t_1}(\sigma_{i,j}\mathbf{x}(t_1)) = \log p_{t_0}\left(T^{-1}\left(\sigma_{i,j}\mathbf{x}(t_1)\right)\right) - \int_{t_0}^{t_1} \nabla_{\mathbf{x}} \cdot \mathbf{v}(\sigma_{i,j}\mathbf{x})dt. \tag{21}$$

Here, the divergence $\nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta\left(\sigma_{i,j}\mathbf{x}\right)$ denotes the divergence with respect to the argument of $\mathbf{v}$, evaluated at $\sigma_{i,j}\mathbf{x}$. Since the base distribution p_{t_0} is permutation invariant, and T^{-1} is equivariant

$$\begin{aligned}
\log p_{t_0}\left(T^{-1}\left(\sigma_{i,j}\mathbf{x}(t_1)\right)\right) &= \log p_{t_0}\left(\sigma_{i,j}T^{-1}\left(\mathbf{x}(t_1)\right)\right) && (22)\\
&= \log p_{t_0}\left(T^{-1}\left(\mathbf{x}(t_1)\right)\right). && (23)
\end{aligned}$$

The divergence in the second term $\nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta\left(\sigma_{i,j}\mathbf{x}\right)$ is a sum over derivatives with respect to all its arguments, so is invariant to $\sigma_{i,j}$

$$\nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta\left(\sigma_{i,j}\mathbf{x}\right) = \nabla_{\mathbf{x}} \cdot \mathbf{v}_\theta\left(\mathbf{x}\right). \tag{24}$$

We therefore have

$$\log p_{t_1}(\sigma_{i,j}\mathbf{x}(t_1)) = \log p_{t_1}(\mathbf{x}(t_1)), \tag{25}$$

thus showing that the dynamics are *equivariant*, and the density is *invariant*. □

Table 3: Hyperparameters used for experiments. Abbreviations are defined in the appendix text.

Experiment	g_θ		f_θ		$\mathbf{y}_{emb}$				
	n	h	n	h	n	c	h	t	batch
Example conditional	5	200	5	200	3	16	200	yes	100
Example bounding box	5	200	5	200	3	16	200	yes	100
Traffic baseline	5	200	4	100	3	32	500	no	100
CLEVR3	4	188	5	196	5	18	409	no	100
CLEVR6	5	100	5	200	5	28	478	no	100

A.2 Experimental details

A.2.1 Force Function Implementation

We model the force functions $f_\theta(\mathbf{x}_i, \mathbf{x}_j)$ and $g_\theta(\mathbf{x}_i)$ by feed forward neural networks. For the pair force $f_\theta(\mathbf{x}_i, \mathbf{x}_j)$, we concatenate the inputs $\mathbf{x}_i$ and $\mathbf{x}_j$. In the case a time variable (t) is used, it is also concatenated. For the conditional input, we construct an embedding vector $\mathbf{y}_{\text{emb}}$, which we concatenate to the second layer inputs of g_θ and f_θ. The embedding vector $\mathbf{y}_{\text{emb}}$ is generated using a separate neural network, here chosen to be a convolutional network, since all our conditional distributions have images as inputs.

A.2.2 Solving the ODE

We use the adaptive solver of Dormand and Prince of order 4 to solve the ODE (Dormand & Prince, 1980). To calculate the gradients of the ODE with respect to its parameters we use the adjoint method (Coddington & Levinson, 1955). This enables calculation of the gradients without back propagating through the computational graph. This functionality is all available in the `torchdiffeq` package (Chen et al., 2018), which is the implementation we use in our experiments.

A.2.3 Table of experimental hyperparameters

In all experiments g_θ, f_θ are implemented as neural networks of n layers and h neurons per layer. The convolutional embedding network has n layers of c channels, followed by a single feed forward layer of h neurons. We use sigmoid-linear units in all our dynamics functions, which satisfy the requirement of Lipschitz continuity, provided the networks are evaluated on a finite domain. The use of a time variable in the dynamics is indicated with $t \in \{\text{yes}, \text{no}\}$. For each experiment, these parameters are presented in Table 3.

A.2.4 Computational Cost

Summing over all pairs of interactions is necessary to make the transformation permutation equivariant, but it comes at a quadratic cost in N. While not problematic for the set sizes in this study, this is clearly a limited approach for large numbers of objects. In these cases, it would be possible to set a boundary on the interaction range, or use a fixed set of $M < N$ inducing points, for a total cost of MN. Such approximations have been studied for example in transformers (which are also quadratic in the sequence length) (Vaswani et al., 2017; Wang et al., 2020). Furthermore, the divergences with respect to $\mathbf{x}_i$ are still quadratic with respect to D. This has been addressed in recent work by using functions that have divergences that can be easily evaluated using automatic differentiation (Chen & Duvenaud, 2019; Biloš & Günnemann, 2021). Although these types of functions are compatible with our framework, the current work only considers cases where $D \leq 5$, and therefore we do not implement it. Our overall algorithm therefore is of cost N^2D^2. If it is necessary to construct distributions with larger D (in for example object detection, rather than bounding box prediction), it is possible to use the methodologies from the aforementioned work to end up with a total computational cost of MND.

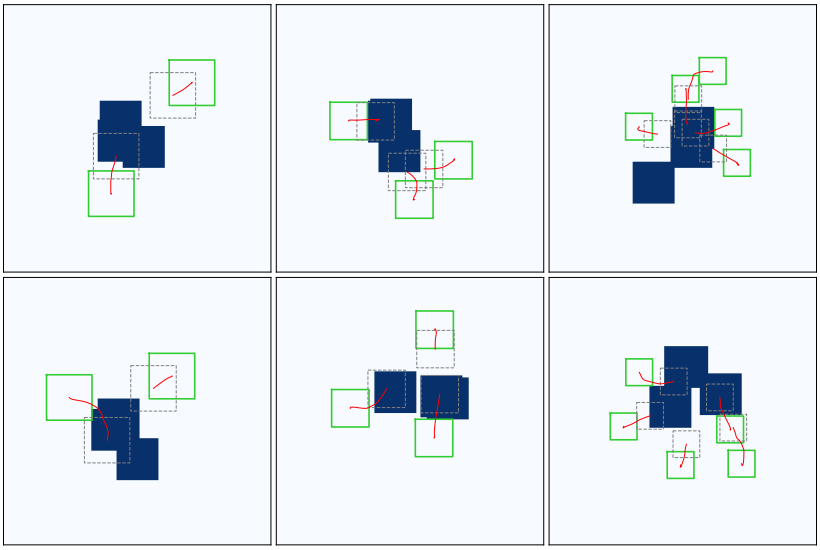

Figure 6: Conditional samples. The condition is an image, which is plotted as a blue on white background. The distribution is trained on samples that do not overlap with the blue regions, or with oneanother. The grey boxes are samples from the base distribution, the green boxes are samples from the flow. The red curves indicate the traveled trajectory for each box.

Table 4: Acceptance rates for conditional sampling. Results presented are the acceptance rate (AR) for Prior samples and the Conditional Permutation Invariant Flows (PIF)

Set Size	Prior AR	PIF AR
2	0.01	0.83
3	$2.01 \cdot 10^{-3}$	0.79
5	$1.76 \cdot 10^{-4}$	0.77

A.2.5 Computational resources

All our experiments were performed on a single GPU, all permutation invariant models were trained between 2 and 7 days of wall-clock time. The "vanilla" continuous normalizing flow, realNVP, and autoregressive model were trained over 14 days of wall-clock time.

A.2.6 Rejection sampling

All samples presenting traffic scenarios in the main text of this manuscript have been checked by an automated procedure, and in a small amount of cases were rejected if an infraction occurred based on actors being offroad or vehicle overlap.

A.2.7 Datasets

The datasets used in this study are the INTERACTION datset (Zhan et al., 2019) (available for research purposes), and the CLEVR dataset (Johnson et al., 2017) (available under the Creative Commons CC BY 4.0 license).

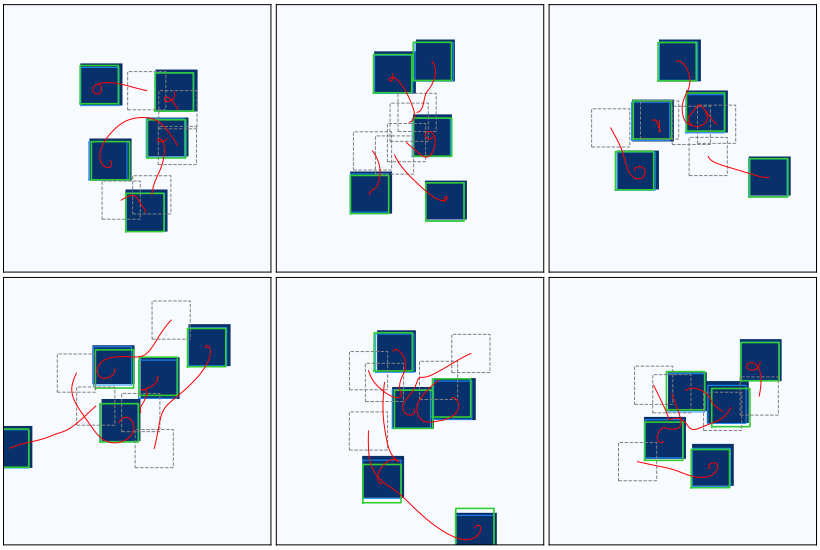

Figure 7: Additional samples of the example bounding box prediction problem.

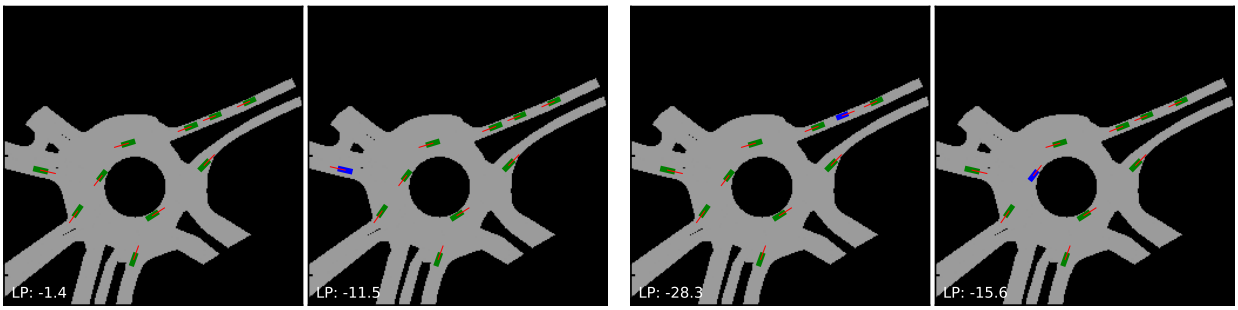

Figure 8: Sample and its log probability (left panel) and three corrupted variations where actors in blue have been turned around (last three panels).

A.3 Additional Experimental results

A.3.1 Conditional non-overlapping boxes

Additional samples of of the conditional generation of non-overlapping boxes (Section 3.1) are presented in Fig. 6, for cardinalities 2, 3 and 5. Performance in terms of acceptance rate against an independent prior are reported in Table 4.

A.3.2 Bounding box prediction

Additional samples for the example bounding box prediction task (Section 3.1) are presented in Fig. 7.

A.3.3 Outlier detection

Since our model has a tractable density, we can use it for outlier detection. In the traffic scene task, we study the case of mislabelled examples, which we artificially generate by rotating one of the actors in the scene by π. The original and corrupted scenes and corresponding log probabilities are shown in the last three panels of Fig. 8. It is clear that reversing one of the actors substantially decreases the probability. Moreover the model correctly captures the severity of the resulting infraction, which is less when an actor is going the wrong way on a two-way road, without the presence of other surrounding actors.

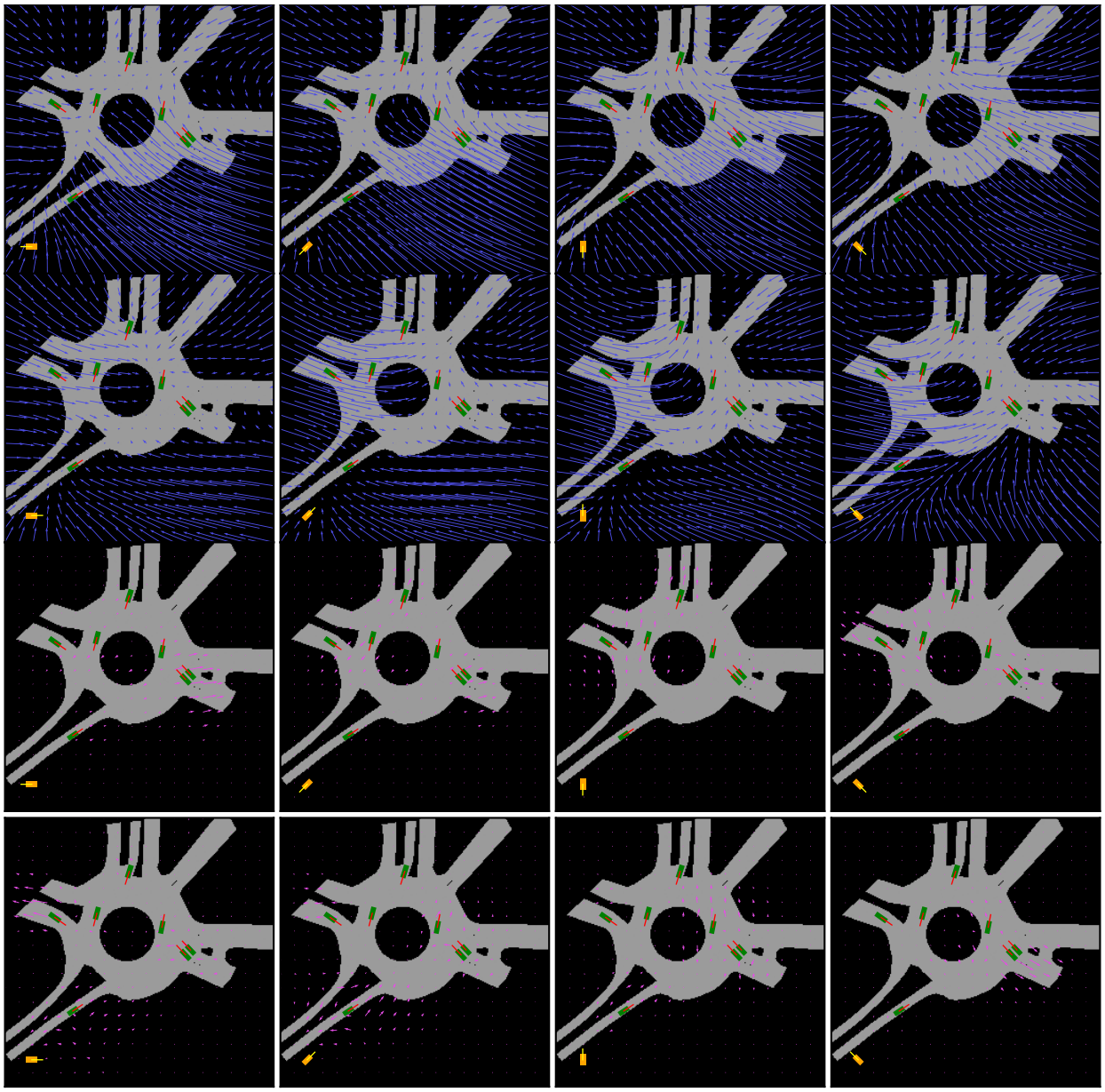

Figure 9: Vector field describing the dynamics for the spatial dimensions of a hypothetical agent of fixed size and orientation (in orange, bottom left), as a function of position. Contributions of g_θ (top two rows) for various orientations of the hypothetical agent are shown as well as f_θ (bottom two rows) for the same orientations.

A.3.4 Flow vector fields

Additional flow fields like the ones presented in the main text, for additional orientations of the hypothetical agent are presented in Fig. 9.

A.3.5 Regularization

We present the effect of our regularization term on the bounding box prediction task, in which the effect is most pronounced. The proportionality constants of the ℓ^2 and ℓ^2_{div} penalty terms are denoted as λ and λ_{div} respectively. Results for various λ and λ_{div} are presented in Fig. 10. The increase of the parameters λ and

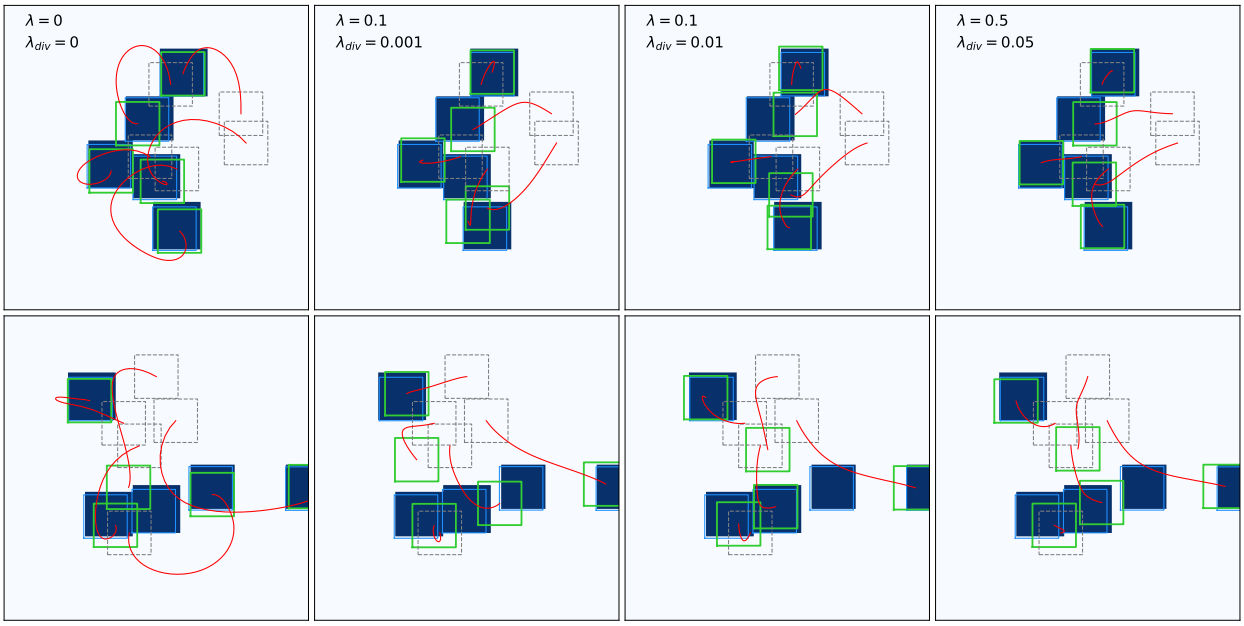

Figure 10: The effect of regularization on the dynamics. The penalty proportionality constants are reported per column in the top panel.

λ_{div} evidently creates more direct trajectories. We further find empirically that it drastically reduces the number of calls to the dynamics made by the ODE solver.

A.3.6 CLEVR

Additional samples for the CLEVR3 dataset are presented in Fig. 11.

Additional samples for the CLEVR6 dataset are presented in Fig. 12.

A.4 Bounding box prediction

A.4.1 CLEVR size

The CLEVR dataset contains the pixel positions of objects, but not the bounding box sizes. The dataset does however provide the center locations of the objects in the global frame $\{x_i, y_i, z_i\}$. Since the objects are sitting on a flat plane at $z = 0$, its z coordinate is equal to half its size. Furthermore, the dataset provides the distance to the camera along the viewpoint axis, d_z. Using these quantities, we approximate the bounding box size Δ as:

$$\Delta \approx \frac{z_i}{\sqrt{d_z}}. \tag{26}$$

We find empirically that this results in reasonable bounding boxes.

A.4.2 CLEVR DSPN

We use Deep Set Prediction Networks (Zhang et al., 2019) to provide some experimental context for the CLEVR results. We note that this method does not provide a tractable log density, and therefore comparison between this method and ours is limited. The network was trained for 100 epochs on the CLEVR6 dataset (including all examples up to a set size of 6).

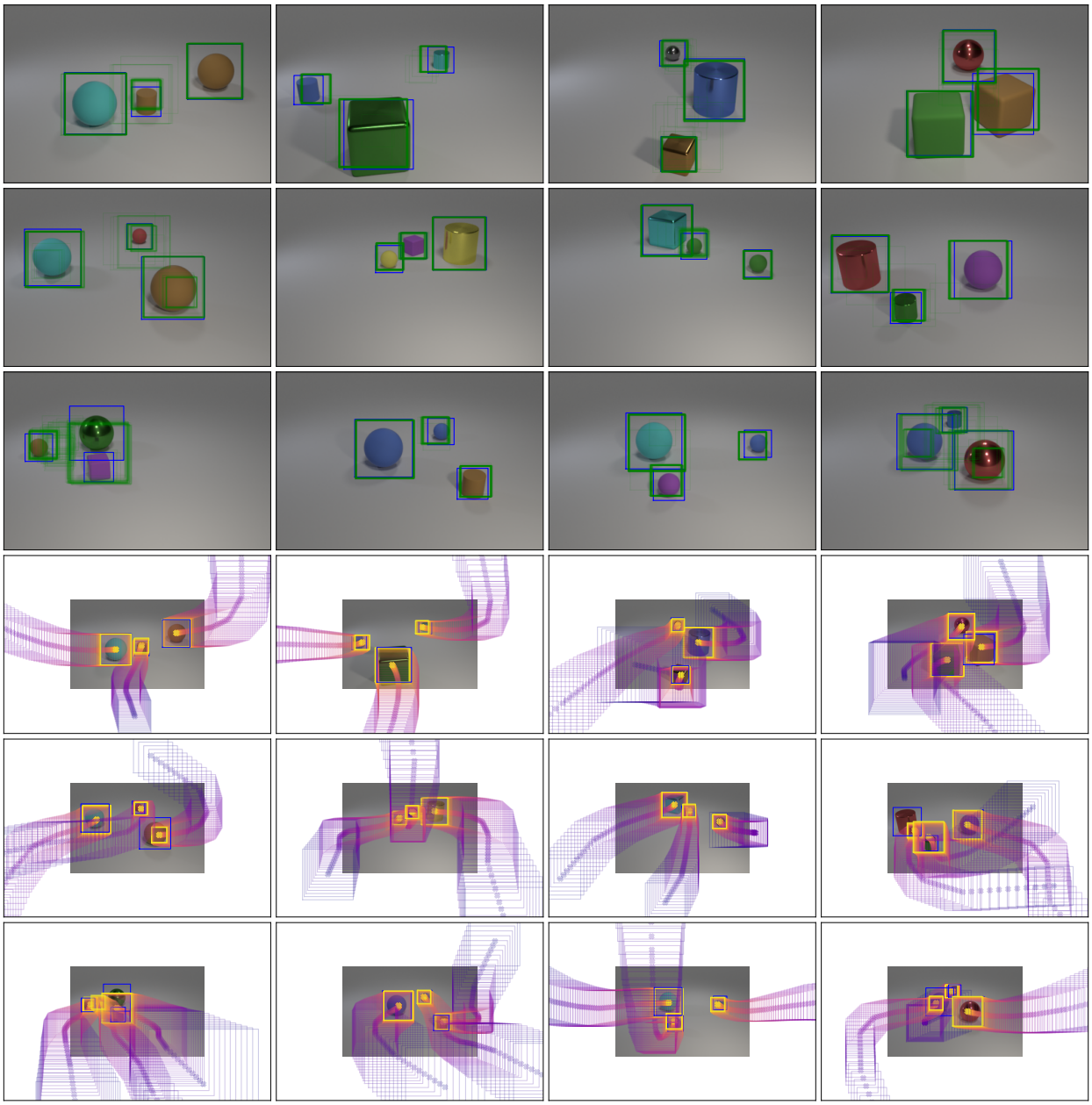

Figure 11: Additional examples for the CLEVR3 dataset. The blue boxes show ground truth bounding boxes, while the green boxes are all samples from the learned conditional distribution.

A.4.3 Detection

In order to turn our bounding box detector in an object detection system, the objects need to be classified as well, which requires the use of categorical variables. While a full treatment of these is beyond the scope of this work, we provide some pointers to ways in which these could be included. Categorical variables are not trivially reparameterizable, but there has been a significant body of work that presents techniques to do this (Maddison et al., 2017; Hoogeboom et al., 2021), none of which provides an exact likelihood. Alternatively, the object classes could be treated as conditionally independent given the image and bounding box, in which case the likelihood is available, but this would neglect correlations between object classes other than those through the image and their bounding boxes.

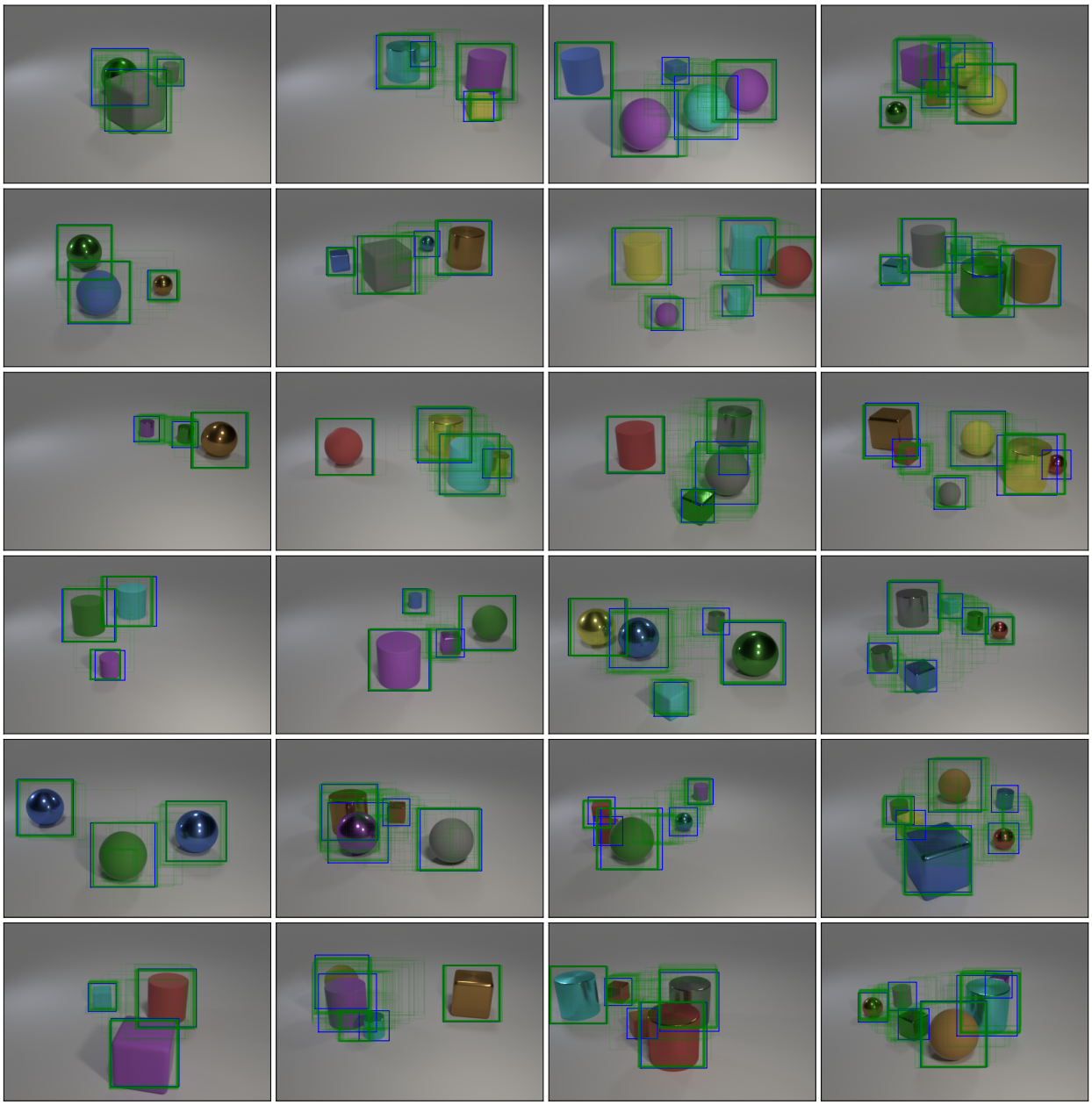

Figure 12: Additional examples for the CLEVR6 dataset. The blue boxes show ground truth bounding boxes, while the green boxes are all samples from the learned conditional distribution.