Generative Adversarial Set Transformers

Karl Stelzner [1] **Kristian Kersting** [1] **Adam R. Kosiorek** [2]

Abstract

Groups of entities are naturally represented as sets, but generative models usually treat them as independent from each other or as sequences. This either over-simplifies the problem, or imposes an order to the otherwise unordered collections, which has to be accounted for in loss computation. We therefore introduce *generative adversarial set transformer* (GAST)—a GAN for sets capable of generating variable-sized sets in a permutation-equivariant manner, while accounting for dependencies between set elements. It avoids the problem of formulating a distance metric between sets by using a permutation-invariant discriminator. When evaluated on a dataset of regular polygons and on MNIST point clouds, GAST outperforms graph-convolution-based GANs in sample fidelity, while showing good generalization to novel set sizes.

1. Introduction

Many important problems in machine learning involve datasets which cannot easily be represented using fixed-length vectors, but rather consist of variable-sized collections of potentially unordered items. For instance, in object detection, the set of objects present in a visual scene can vary in size and has no natural ordering. With the advent of attention-based architectures and transformers (Vaswani et al., 2017), there is now a widely used architecture capable of handling this type of data, but which has been applied predominantly to sequences (Devlin et al., 2019). When it was used on sets, it was typically in the context of supervised, discriminative problems with sets as inputs (Zaheer et al., 2017; Lee et al., 2019), or outputs (Carion et al., 2020). Unsupervised generative modelling of sets, however, has received considerably less attention. Perhaps as a consequence, many current generative models reasoning about objects are sidestepping this issue by assuming independence between objects, introducing expensive recurrent dependencies between them, or relying on iterative refinement methods (Eslami et al., 2016; Greff et al., 2019; Engelcke et al., 2020; Nguyen-Phuoc et al., 2020).

Adapting standard generative architectures to sets brings about two key challenges. First, the represented distribution should be order-invariant, i. e., all permutations of a generated sample should be equiprobable. It has been shown that naïvely applying non-order-invariant architectures, such as multilayer perceptrons (MLPs), to sets results in degraded performance due to symmetries of the loss landscape (Zaheer et al., 2017; Zhang et al., 2020). Second, many generative models require computing the likelihood of the input during training, which, for sets, often amounts to comparing the input set with a generated one. This requires considering all possible permutations of the two sets, giving rise to a bipartite graph matching problem. Existing works in this direction have either solved it using the cubic-time Hungarian algorithm, or approximated it via e. g. the Chamfer loss (Zhang et al., 2019b), which can introduce perilous local optima, especially for sets of different sizes. Both aspects show that generating sets remains technically challenging.

In this paper, we propose a new architecture for set generation, the *generative adversarial set transformer* (GAST). It addresses the two problems described above by employing set transformers (Lee et al., 2019) to generate sets in an equivariant manner, and by formulating an adversarial objective with an order-invariant discriminator, making an explicit distance metric between sets unnecessary. Previous work on adversarial set generation (Achlioptas et al., 2018; Valsesia et al., 2019; Shu et al., 2019; Li et al., 2018) has focused on the generation of 3D point clouds, with the assumption that input sets are all of equal size, and that their elements are independently and identically distributed (IID). These assumptions are ill-suited for reasoning about objects: scenes can contain different numbers of objects, and there may be complicated dependencies between objects, which we would like to model. GAST avoids these restrictions and allows the generation of variable-sized sets, taking into account dependencies between elements. GAST also differs from prior art in that it uses self-attention instead of graph-convolutional network (GCN) variants (Kipf & Welling, 2017), resulting in a fully differentiable architecture that is not restricted to a particular graph structure.

[1] TU Darmstadt, Germany [2] DeepMind, United Kingdom. Correspondence to: Karl Stelzner <stelzner@cs.tu-darmstadt.de>.

Workshop on Object-Oriented Learning at ICML 2020. Copyright 2020 by the author(s).

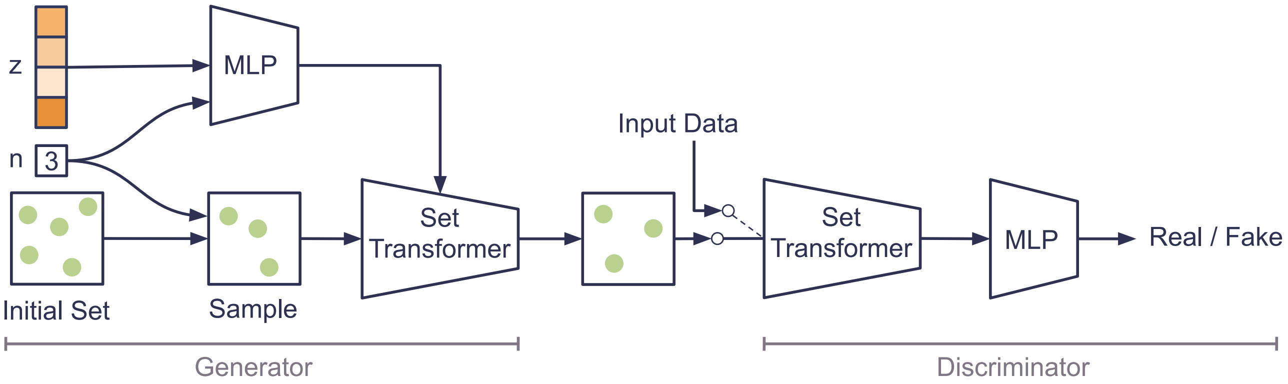


Figure 1. Overview of the generative adversarial set transformer (GAST).

2. Further Related Work

GANs are widely used as generative models for images (Goodfellow et al., 2014). While some apply self-attention to facilitate long-distance interactions between pixels (Zhang et al., 2019a; Brock et al., 2019), few adversarial models have been proposed for the set domain. We are aware of four exceptions, all of which focus on the generation of IID point clouds—sets of points independently sampled from the surface of 3D shapes, such as meshes from the ShapeNet dataset (Chang et al., 2015).

The r-GAN (Achlioptas et al., 2018) uses an order-invariant discriminator, but employs a standard MLP as its generator. The following models improved on this by implementing order equivariant generators. Valsesia et al. (2019) use a GCN, whereby the graph structure is dynamically generated using the k-nearest neighbor heuristic. Shu et al. (2019) present the less computationally-expensive TreeGAN, in which the GCN operates on a tree structure obtained by repeatedly upsampling points. Finally, Li et al. (2018) propose point-cloud GANs (PCGANs), a hierarchical model which first generates a latent shape-variable, and then samples points independently conditioned on that shape variable. This allows sampling arbitrarily many points, at the price of neglecting any interactions between individual points.

All of these models operate on input data where each set has the same number of points, and where points are IID. For PCGANs, the IID assumption is also encoded in the architecture of the generator. The other models have, in principle, the capability of generating non-IID data, but this has not been confirmed empirically.

3. Background

GCNs and attention based-architectures such as transformers share the goal of processing collections of entities by aggregating pairwise interactions between them (Battaglia et al., 2018). They exhibit important technical differences, however: GCNs only allow interactions between nodes connected via an edge, requiring the specification of a graph structure. Attention-based architectures, on the other hand, typically account for all pairwise interactions. In this paper, we build on set transformers (Lee et al., 2019), a family of architectures explicitly designed for modelling operations on sets which are equivariant/invariant to the order of their elements. Their basic operation is the multi-head dot-product attention block $\text{MAB}(Q, K)$, which computes interactions between a query set $Q \in \mathbb{R}^{n \times d_k}$ and a key set $K \in \mathbb{R}^{m \times d_k}$, followed by aggregating the results for each query (Vaswani et al., 2017). We use several modules constructed from this building block, starting with the self-attention block (SAB). SAB simply computes the multi-head attention of a set of entities with itself, i. e., $\text{SAB}(X) = \text{MAB}(X, X)$. Repeatedly applying this block lets us transform a simple initial set into samples resembling the training data.

Self-attention on a set requires computing interactions between all pairs of entities — a computationally prohibitive operation if the number of entities is large. We follow Lee et al. (2019) and replace the SABs with *induced* self-attention blocks (ISABs); here, interactions are funnelled through a smaller set of anchor entities A, such that $\text{ISAB}(X) = \text{MAB}(X, \text{MAB}(A, X))$. As a result, the total number of interactions to compute is reduced from $|X|^2$ to $2|A||X|$, while still allowing all X to interact with each other. The anchor points A can be initialized to a fixed value, learned during training, or conditioned on other data. Finally, in order to map from sets to a fixed vector in the discriminator, we make use of *induced set encoders* (ISEs). They also use a set of anchor points A, but summarize the results of the attention block via summing: $\text{ISE}(X) = \sum_i MAB(a_i, X)$. ISEs represent a more expressive alternative to simpler set-pooling operations.

4. Generative Adversarial Set Transformers

We now present GAST in detail, following the illustration given by Fig 1. Our goal is to represent distributions over sets of variable size. We choose to view them factorized as $p(n)p(\mathbf{x} \mid n)$, where n is the number of elements,

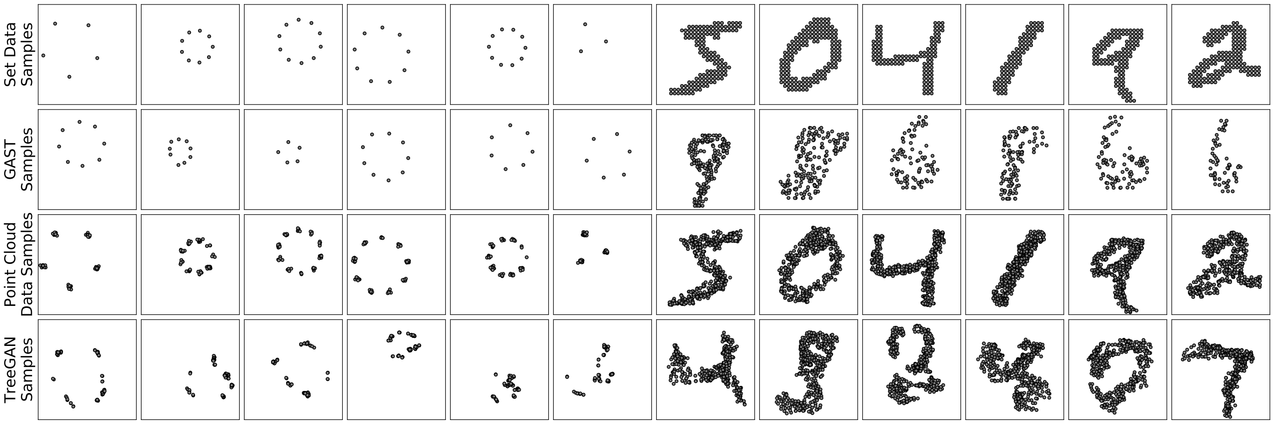


Figure 2. Training data and samples obtained from GAST (top) and TreeGAN (bottom) for the polygon dataset (left) and MNIST (right).

and $\mathbf{x} := \{\mathbf{x}_i\}_{i=1}^n$ is the set of elements, with elements $\mathbf{x}_i \in \mathbb{R}^d$. This allows us to start the generative process by drawing a random initial set $\mathbf{x}^0$ of the desired size n from some $p(\mathbf{x}^0 \mid n)$. We can then parameterize $p(\mathbf{x} \mid n)$ using a set-to-set function f, which transforms the initial set according to some noise variables $\mathbf{z} \sim p(\mathbf{z})$ such that $p(\mathbf{x} \mid n) = \int \delta(f(\mathbf{x}^0, \mathbf{z}))p(\mathbf{x}^0 \mid n)p(\mathbf{z})d\mathbf{x}^0 d\mathbf{z}$. Such functions f can be readily formulated using the set-transformer blocks outlined above. The benefit of this scheme is that it does not require dynamically adding or removing elements over the course of the generation process, thereby avoiding any discrete operations which would complicate gradient-based training.

To represent variable-sized sets in the context of batched computations, we operate on tensors of maximum set size $N \geq n$, typically the size of the largest set in the training data. Specifically, a batch of b sets containing d-dimensional elements is represented by a tensor of shape $[b, N, d]$, combined with a binary presence indicator matrix of shape $[b, N]$. Whenever interactions between entities are computed by the set-transformer blocks, we take care to set the interactions involving non-existing elements to zero. For simplicity, we ignore this technical detail in the following and instead only focus on individual sets.

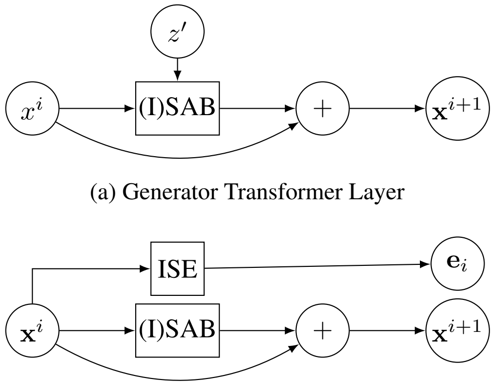
The generative process begins by drawing the number of points n from a categorical distribution whose parameters are equal to the relative set-size frequencies in the training set. We then draw a global noise vector $\mathbf{z}$ from $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. Both $\mathbf{z}$ and n are then processed by an MLP, allowing the model to compute global features $\mathbf{z}'$ of the set such as position, orientation, and shape. Finally, we sample an initial set $\mathbf{x}^0$ of size n. To do so, we first instantiate N Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2)$ with learnable parameters $\boldsymbol{\mu}, \boldsymbol{\sigma} \in \mathbb{R}^d$. We then choose n such Gaussians uniformly at random without replacement, and sample one $\mathbf{x}_i^0$ from each Gaussian.

To process this initial set, we use a set-transformer conditioned on $\mathbf{z}'$, implementing the function $f(\mathbf{x}^0, \mathbf{z}')$ introduced above. Specifically, we stack multiple self-attention blocks (SAB), each featuring a residual connection (Fig. 3(a)). Conditioning on $\mathbf{z}'$ is done by concatenating each element $\mathbf{x}_i^0$ with $\mathbf{z}'$ before applying self-attention. When N is large enough so that the quadratic cost of SABs is prohibitive, we instead use induced self-attention blocks (ISABs). In this case, we predict the anchor points A from the conditioning vector $\mathbf{z}'$ using a small two-layer MLP. Finally, after k_{gen} such layers, we use a linear mapping to project each element in the set of the desired dimensionality. The discriminator is structured very similarly. It begins with an element-wise linear mapping of the inputs to a higher dimensionality. Again, the resulting set is then processed using a sequence of k_{disc} (induced) self-attention blocks with residual connections, this time without external conditioning (Fig. 3b). Each intermediate result, as well as the final set, is also processed using an induced set encoder (ISE), yielding an encoding vector $\mathbf{e}^i$. At the end, the vectors $\mathbf{e}^0, \ldots, \mathbf{e}^{k_{\text{disc}}}$ are concatenated and fed into an MLP to obtain the final score.

To train the model, we follow the techniques employed by self-attention GANs (Zhang et al., 2019a). We minimize the adversarial hinge loss using the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 10^{-4} for the generator and $4 \cdot 10^{-4}$ for the discriminator. To avoid mode collapse, we apply spectral normalization (Miyato et al., 2018) to all linear layers, including the ones within the attention blocks.

5. Experiments

To evaluate our model's ability to model non-IID sets of variable size, we considered two datasets. First, we built a toy dataset in which the sets contain the vertices of regular polygons, represented via 2D Cartesian coordinates. We varied the number of vertices between 3 and 10, and randomized the polygons' position, rotation, and scale. As a



(a) Generator Transformer Layer

(b) Discriminator Transformer Layer

Figure 3. Layers used within our set transformers.

second dataset, we followed Zaheer et al. (2017) and generated a version of MNIST in which the digits are represented by sets of non-zero pixels, again represented by their 2D coordinates. Here, the set size ranges from 32 to 342. For both datasets, we used $k_{\text{gen}} = 4$ layers for the generator and $k_{\text{disc}} = 2$ layers for the discriminator, and four heads for each attention block. Within the set transformers, we represented points as 64-dimensional vectors. For the MNIST dataset, we utilized induced attention blocks to reduce the computational load, using $|A| = 24$ anchor points per block.

As a baseline, we examined TreeGANs (Shu et al., 2019). Since that model is designed for point clouds of constant size, we created variants of our datasets, which fit this setting. To do so, we sampled N' points with replacement from each set in the training data, and then added Gaussian noise with a small standard deviation (0.02) to the resulting points. To provide for good coverage of the original set, we choosed N' somewhat higher than N, namely $N' = 64$ for the polygons and $N' = 384$ for MNIST. We adjusted the TreeGAN's branching factors to facilitate these set sizes, but otherwise left the architecture and hyperparameters as recommended by the authors.

5.1. Sample Quality

Fig. 2 depicts samples from the training sets, and samples from the corresponding models. We find that GAST learns to generate almost perfectly arranged polygons, whereas TreeGAN struggles to allocate its samples in a regular manner. It also almost exclusively generates between 4 and 6 recognizable clusters. We suspect that this is a result of the fact that its points tend to cluster by subtree, and the rigid tree structure makes it difficult to dynamically change the number of clusters. GAST, in contrast, generates convincing samples for all set sizes. On MNIST, the visual fidelity of both models is closer, but there are still interesting qualitative differences. For GAST, the distribution of points is more even,likely a result of the fact that the training data was obtained in a structured way instead of being sampled IID.

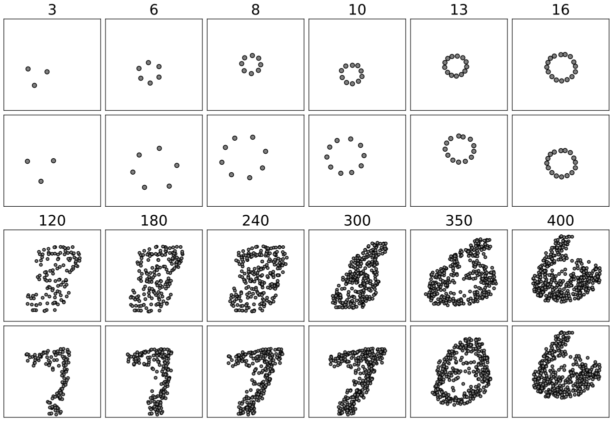


Figure 4. Samples from GAST for different set sizes. All other random variables are left constant across each row, but differ between the rows. The last two columns show set sizes not present in the training dataset.

5.2. Generalization and Disentanglement

In order to investigate the influence that different set sizes have on our model, we generated samples by varying n while leaving all other random variables constant. We also tried values for n, which were not present in the training dataset. For the polygon dataset, we find that the model generalizes to higher numbers of points quite well, as the sets still resemble regular polygons. However, the variation between sets decreases as n approaches N, likely a result of the fact that the sampling from the initial set exhibits lower variance as n increases.

The size of the polygons appears to be mostly disentangled from the number of points, whereas positions are influenced by n. For MNIST, we find that the shape of the generated digit often changes with n, reflecting the fact that the average number of white pixels differs greatly between digit classes. While this lack of disentanglement might be undesirable in some cases, if the goal is to fit the distribution of the training data as faithfully as possible, then this behavior is correct.

6. Conclusion

We have presented *generative adversarial set transformer* (GAST) — a generative model for variable-sized collections of unordered non-IID elements. We found that it yields higher-quality samples compared to previous approaches, while also showing good generalization to novel set sizes. We hope that it will help future generative models to accurately and efficiently model dependencies between set elements, with object-aware scene modelling as one of the most promising applications.

Acknowledgements

We would like to thank Thu Nguyen-Phuoc for helpful discussions on GAN training.

References

Achlioptas, P., Diamanti, O., Mitliagkas, I., and Guibas, L. J. Learning representations and generative models for 3d point clouds. In *International Conference on Machine Learning*, 2018.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V. F., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Çaglar Gülçehre, Song, H. F., Ballard, A. J., Gilmer, J., Dahl, G. E., Vaswani, A., Allen, K. R., Nash, C., Langston, V., Dyer, C., Heess, N. M. O., Wierstra, D., Kohli, P., Botvinick, M. M., Vinyals, O., Li, Y., and Pascanu, R. Relational inductive biases, deep learning, and graph networks. *arXiv*, 1806.01261, 2018.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A. M., and Zagoruyko, S. End-to-end object detection with transformers. *arXiv*, 2005.12872, 2020.

Chang, A. X., Funkhouser, T. A., Guibas, L. J., Hanrahan, P., Huang, Q.-X., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., and Yu, F. Shapenet: An information-rich 3d model repository. 2015, arXiv:1512.03012.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Conference of the North American Chapter of the Association for Computational Linguistics*, 2019.

Engelcke, M., Kosiorek, A. R., Jones, O. P., and Posner, I. Genesis: Generative scene inference and sampling with object-centric latent representations. In *International Conference on Learning Representations*, 2020.

Eslami, S. M. A., Heess, N. M. O., Weber, T., Tassa, Y., Szepesvari, D., Kavukcuoglu, K., and Hinton, G. E. Attend, infer, repeat: Fast scene understanding with generative models. In *Neural Information Processing Systems*, 2016.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Neural Information Processing Systems*, 2014.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., and Teh, Y. W. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning*, 2019.

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

Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

Nguyen-Phuoc, T., Richardt, C., Mai, L., Yang, Y.-L., and Mitra, N. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *arXiv*, 2002.08988, 2020.

Shu, D. W., Park, S. W., and Kwon, J. 3d point cloud generative adversarial network based on tree structured graph convolutions. In *International Conference on Computer Vision*, 2019.

Valsesia, D., Fracastoro, G., and Magli, E. Learning localized generative models for 3d point clouds via graph convolution. In *International Conference on Learning Representations*, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Neural Information Processing Systems*, 2017.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Póczos, B., Salakhutdinov, R., and Smola, A. J. Deep sets. In *Neural Information Processing Systems*, 2017.

Zhang, H., Goodfellow, I., Metaxas, D., and Odena, A. Self-attention generative adversarial networks. In *International Conference on Machine Learning*, 2019a.

Zhang, Y., Hare, J. S., and Prügel-Bennett, A. Deep set prediction networks. In *Neural Information Processing Systems*, 2019b.

Zhang, Y., Hare, J. S., and Prügel-Bennett, A. Fspool: Learning set representations with featurewise sort pooling. In *International Conference on Learning Representations*, 2020.