Set2Graph: Learning Graphs From Sets

Hadar Serviansky 1  Nimrod Segol 1  Jonathan Shlomi 1  Kyle Cranmer 2  Eilam Gross 1  Haggai Maron 3  Yaron Lipman 1

Abstract

Many problems in machine learning can be cast as learning functions from sets to graphs, or more generally to hypergraphs; in short, Set2Graph functions. Examples include clustering, learning vertex and edge features on graphs, and learning features on triplets in a collection.

A natural approach for building Set2Graph models is to characterize all linear equivariant set-to-hypergraph layers and stack them with non-linear activations. This poses two challenges: (i) the expressive power of these networks is not well understood; and (ii) these models would suffer from high, often intractable computational and memory complexity, as their dimension grows exponentially.

This paper advocates a family of neural network models for learning Set2Graph functions that is both practical and of maximal expressive power (universal), that is, can approximate arbitrary continuous Set2Graph functions over compact sets. Testing these models on an important particle physics problem, we find them favorable to existing baselines.1

1. Introduction

We consider the problem of learning functions taking sets of vectors in \( \mathbb{R}^d \) to graphs, or more generally hypergraphs; we name this problem Set2Graph, or set-to-graph. Set-to-graph functions appear in machine-learning applications such as clustering, predicting features on edges and nodes in graphs, and learning \( k \)-edge information in sets.

Mathematically, we represent each set-to-graph function as a collection of set-to-\( k \)-edge functions, where each set-to-

---

1Weizmann Institute of Science, Rehovot, Israel 2New York University, New-York, USA 3NVIDIA Research. Correspondence to: Hadar Serviansky <hadar.serviansky@weizmann.ac.il>.


1This is a workshop version of (Serviansky et al., 2020)
(as the graph-to-graph case), or low order tensors (as in the set-to-set case) are sufficient?

In this paper we: (i) show that low order tensors are sufficient for set-to-graph universality, and (ii) build an equivariant model for the set-to-graph problem that is both practical (i.e., small number of parameters and no-need to build high-order tensors in memory) and provably universal. We achieve that with a composition of three networks: \( F^k = \psi \circ \beta \circ \phi \), where \( \phi \) is a set-to-set model, \( \beta \) is a non-learnable broadcasting set-to-graph layer, and \( \psi \) is a simple graph-to-graph network using only a single Multi-Layer Perceptron (MLP) acting on each k-edge independently.

Our main motivation for this work comes from an important set-to-2-edges learning problem in particle physics: partitioning (clustering) of simulated particles generated in the Large Hadron Collider (LHC). We demonstrate our model produces state of the art results on this task compared to relevant baselines.

2. Learning hypergraphs from sets

We would like to learn functions of sets of \( n \) vectors in \( \mathbb{R}^{dn} \) to hypergraphs with \( n \) nodes (think of the nodes as corresponding to the set elements), and arbitrary k-edge feature vectors in \( \mathbb{R}^{dk} \), where a k-edge is defined as a k-tuple of set elements. A function mapping sets of vectors to k-edges is called set-to-k-edge function and denoted \( F^k : \mathbb{R}^{n \times d_n} \rightarrow \mathbb{R}^{n^k \times d_k} \). Consequently, a set-to-hypergraph function would be modeled as a sequence \( \{ F^1, F^2, \ldots, F^K \} \), for target hypergraphs with hyperedges of maximal size \( K \). For example, \( F^2 \) learns pairwise relations in a set; and \( \{ F^1, F^2 \} \) is a function from sets to graphs (outputs both node features and pairwise relations); see Figure 1.

Our goal is to devise equivariant neural network models for \( F^k \) that are as-efficient-as-possible in terms of number of parameters and memory usage, but on the same time with maximal expressive power, i.e., universal.

Representing sets and k-edges. A matrix \( X = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^{n \times d_n} \) represents a set of \( n \) vectors \( x_i \in \mathbb{R}^{d_n} \) and therefore should be considered up to reordering of its rows. We denote by \( S_n = \{ \sigma \} \) the symmetric group, that is the group of bijections (permutations) \( \sigma : [n] \rightarrow [n] \), where \( [n] = \{ 1, \ldots, n \} \). We denote by \( \sigma \cdot X \) the matrix resulting in reordering the rows of \( X \) by the permutation \( \sigma \), i.e., \((\sigma \cdot X)_{i,j} = X_{\sigma^{-1}(i),j}\). In this notation, \( X \) and \( \sigma \cdot X \) represent the same set, for all permutations \( \sigma \).

k-edges are represented as a tensor \( Y \in \mathbb{R}^{n^k \times d_n} \), where \( Y_{i,j} \in \mathbb{R}^{d_n} \) denotes the feature vector attached to the k-edge defined by the k-tuple \((x_{i_1}, x_{i_2}, \ldots, x_{i_k})\), where \( i = (i_1, i_2, \ldots, i_k) \in [n]^k \) is a multi-index with non-repeating indices. Similarly to the set case, k-edges are considered up to renumbering of the nodes by some permutation \( \sigma \in S_n \).

That is, if we define the action \( \sigma \cdot Y \) by \((\sigma \cdot Y)_{i,j} = Y_{\sigma^{-1}(i),j}\), where \( \sigma^{-1}(i) = (\sigma^{-1}(i_1), \ldots, \sigma^{-1}(i_k)) \), then \( Y \) and \( \sigma \cdot Y \) represent the same k-edge data, for all \( \sigma \in S_n \).

Equivariance. A sufficient condition for \( F^k \) to represent a well-defined map between sets \( X \in \mathbb{R}^{n \times d_n} \) and k-edge data \( Y \in \mathbb{R}^{n^k \times d_n} \) is equivariance to permutations, namely

\[
F^k(\sigma \cdot X) = \sigma \cdot F^k(X),
\]

for all sets \( X \in \mathbb{R}^{n \times d_n} \) and permutations \( \sigma \in S_n \). Equivariance guarantees, in particular, that the two equivalent sets \( X \) and \( \sigma \cdot X \) are mapped to equivalent k-edge data tensors \( F^k(X) \) and \( \sigma \cdot F^k(X) \).

Set-to-k-edge models. In this paper we explore the following neural network model family for approximating \( F^k \):

\[
F^k(X; \theta) = \psi \circ \beta \circ \phi(X),
\]

where \( \phi, \beta, \) and \( \psi \) will be defined soon. For \( F^k \) to be equivariant it is sufficient that its constituents, namely \( \phi, \beta, \psi \), are equivariant. That is, \( \phi, \beta, \psi \) all satisfy equation 1.

Set-to-graphs models. Given the model of set-to-k-edge functions, a model for a set-to-graph function can now be constructed from a pair of set-to-k-edge networks \( \{ F^1, F^2 \} \). Similarly, set-to-hypergraph function would require \( \{ F^1, \ldots, F^K \} \), where \( K \) is the maximal hyperedge size. Figure 1 shows an illustration of set-to-k-edge and set-to-graph functions.

\beta component. \( \beta : \mathbb{R}^{n \times d_1} \rightarrow \mathbb{R}^{n \times d_2} \) is a non-learnable linear broadcasting layer mapping sets to k-edges. In theory, as shown in (Maron et al., 2019b) the space of equivariant linear mappings \( \mathbb{R}^{n \times d_1} \rightarrow \mathbb{R}^{n \times d_2} \) is of dimension \( d_1 d_2 b(\ell(k + 1)) \) which can be very high since bell numbers have exponential growth. Interestingly, in the set-to-k-edge case one can achieve universality with only k linear operators. We define the broadcasting operator to be \( \beta(X)_{i,j} = [x_{i_1}, x_{i_2}, \ldots, x_{i_k}] \) where \( i = (i_1, \ldots, i_k) \) and brackets denote concatenation in the feature dimension, that is, for \( A \in \mathbb{R}^{n \times d_2}, B \in \mathbb{R}^{n \times d_2} \) their concatenation is \( [A, B] \in \mathbb{R}^{n \times (d_1 + d_2)} \). Therefore, the feature output dimension of \( \beta \) is \( d_2 = kd_1 \).

As an example, consider the graph case, where \( k = 2 \). In this case \( \beta(X)_{i_1,i_2} = [x_{i_1}, x_{i_2}] \). This function is illus-
Figure 2. The model architecture for the Set-to-graph and set-to-2-edge functions.

Figure 2 illustrates set-to-graph broadcasting data in $\mathbb{R}^{n \times d_1}$ to $\mathbb{R}^{n^2 \times d_2}$.

To see that the broadcasting layer is equivariant, it is enough to consider the rows of $X$ by a permutation $\sigma$ we get $\beta(X)_{i,j} = x_{\sigma^{-1}(i),j} = \beta(X)_{\sigma^{-1}(i),j} = (\sigma \cdot \beta(X))_{i,j}$.

ψ component. $\psi: \mathbb{R}^{n \times k \times d_2} \rightarrow \mathbb{R}^{n \times d_{out}}$ is a mapping of $k$-tensors to $k$-tensors. Here the theory of equivariant operators indicates that the space of linear equivariant maps is of dimension $d_2 d_{out} \text{bell}(2k)$ that suggests a huge number of model parameters even for a single linear layer. Surprisingly, universality can be achieved with much less, in fact a single linear operator (i.e., scaled identity) in each layer. In the multi-feature multi-layer case this boils to applying a Multi-Layer Perceptron $m: \mathbb{R}^{d_2} \rightarrow \mathbb{R}^{d_{out}}$ independently to each feature vector in the input tensor $X \in \mathbb{R}^{n \times k \times d_2}$. That is, we use $\psi(X)_{i,:} = m(X_{i,:})$. Figure 2 illustrates set-to-edges and set-to-graph models incorporating the three components $\phi, \beta, \psi$ discussed above.

3. Universality of set-to-graph models.

The model $F^k$ introduced above, is universal, in the sense it can approximate arbitrary continuous equivariant set-to-k-edge functions $G^k: \mathbb{R}^{n \times d_n} \rightarrow \mathbb{R}^{n^2 \times d_m}$ over compact domains $K \subset \mathbb{R}^{n \times d_n}$.

Theorem 1. The model $F^k$ is set-to-k-edge universal.

Due to the page limit, we provide a simpler universality proof (under some mild extra conditions) for the set-to-edge model, $F^2$, based on the Singular Value Decomposition (SVD). We refer the reader to the full version of the paper for a general proof (Serviansky et al., 2020).

3.1. A simple proof for universality of second-order tensors

It is enough to consider the $d_{out} = 1$ case; the general case is implied by applying the argument for each output feature dimension independently. Let $G^2$ be an arbitrary continuous equivariant set-to-2-edge function $G^2: K \subset \mathbb{R}^{n \times d_n} \rightarrow \mathbb{R}^{n \times n}$. We want to approximate $G^2$ with our model $F^2$.

First, note that without losing generality we can assume $G^2(X)$ has a simple spectrum (i.e., eigenvalues are all different) for all $X \in K$. Indeed, if this is not the case we can always choose $\lambda > 0$ sufficiently large and consider $G^2 + \lambda \text{diag}(1, 2, \ldots, n)$. This diagonal addition does not change the 2-edge values assigned by $G^2$, and it guarantees a simple spectrum using standard hermitian matrix eigenvalue perturbation theory (see (Stewart, 1990), Section IV.4).

Now let $G^2(X) = U(X) \Sigma(X) V(X)^T$ be the SVD of $G^2(X)$, where $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_n]$. Since $G^2(X)$ has a simple spectrum, $U, V, \Sigma$ are all continuous in $X$; $\Sigma$ is unique, and $U, V$ are unique up to a sign flip of the singular vectors (i.e., columns of $U, V$) (O’Neil, 2005). Let us first assume that the singular vectors can be chosen uniquely also up to a sign, later we show how we achieve this with some additional mild assumption.

Now, uniqueness of the SVD together with the equivariance of $G^2$ imply that $U, V$ are continuous set-to-set equivariant and $\Sigma$ is a continuous set invariant function:

$$
(\sigma \cdot U(X) \Sigma(X) (\sigma \cdot V(X))^T
= \sigma \cdot G(X) = G(\sigma \cdot X)
(3)
= U(\sigma \cdot X) \Sigma(\sigma \cdot X) V(\sigma \cdot X)^T.
$$

Lastly, since $\phi$ is set-to-set universal there is a choice of its parameters so that it approximates arbitrarily well the equivariant set-to-set function $Y = [U, V, 1^T \Sigma]$. The $\psi$ component can be chosen by noting that $G^2(X)_{i_1, i_2} = \sum_{j=1}^n \sigma_j U_{i_1,j} V_{i_2,j} = p(\beta(Y)_{i_1, i_2})$, where $\sigma_j$ are the singular values, and $p: \mathbb{R}^n \rightarrow \mathbb{R}$ is a cubic polynomial. To conclude pick $m$ to approximate $p$ sufficiently well so that $\psi \circ \beta \circ \phi$ approximates $G^2$ to the desired accuracy.

To achieve uniqueness of the singular vectors up to a sign we can add, e.g., the following assumption: $1^T u_i(X) \neq 0 \neq 1^T v_i(X)$ for all singular vectors and $X \in K$. Using this assumption we can always pick $u_i(X), v_i(X)$ in the SVD so that $1^T u_i(X) > 0, 1^T v_i(X) > 0$, for all $i \in [n]$. Lastly, note that equation 3 suggests that also outer-product can be used as a broadcasting layer.

4. Applications

4.1. Model variants and baselines

Variants of our model. We consider two variations of our model: S2G: This is Our basic model. We used the $F^2$ and $F^3$ (resp.) models for these learning tasks. For $F^i$, $\phi$ is implemented using DeepSets (Zaheer et al., 2017) with 5 layers and output dimension $d_1 \in \{5, 80\}$; $\psi$ is implemented with an MLP, $m$, with $\{2, 3\}$ layers with input dimension $d_2$ defined by $d_1$ and $\beta$. $\beta$ is implemented according to Section 2: for $k = 2$ it uses $d_2 = 2 \times d_1$ output features. S2G+:

For the $k = 2$ case we have also tested a more general (but not more expressive) broadcasting $\beta$ defined using the full equivariant basis $\mathbb{R}^n \rightarrow \mathbb{R}^n$ from (Maron et al., 2019b) that contains bell(3) = 5 basis operations. This broadcasting layer gives $d_2 = 5 \times d_1$. 
Observed Particles

We tackle learning set-to-set functions. Here, each training example is a pair \((X, Y)\) where \(X\) is a set \(X = (x_1, x_2, \ldots, x_n)^T \in \mathbb{R}^{n \times d_x}\) and \(Y \in \{0, 1\}^{n \times n}\) is an adjacency matrix (the diagonal of \(Y\) is ignored). Our main experiment tackles an important particle physics problem.

Problem statement. In particle physics experiments, such as the Large Hadron Collider (LHC), beams of incoming particles are collided at high energies. The results of the collision are outgoing particles, whose properties (such as the trajectory) are measured by detectors surrounding the collision point. A critical low-level task for analyzing this data is to associate the particle trajectories to their progenitors, which can be formalized as partitioning sets of particle trajectories into subsets according to their unobserved point of origin in space. This task is referred to as vertex reconstruction in particle physics and is illustrated in Figure 3.

We cast this problem as a set-to-set edge problem by treating the measured particle trajectories as elements in the input set and nodes in the output graph, where the parameters that characterize them serve as the node features. An edge between two nodes indicates that the two particles come from a common progenitor or vertex.

Data. We consider three different types (or flavors) of particle sets (called jets) corresponding to three different fundamental data generating processes labeled bottom-jets, charm-jets, and light-jets (B/C/L).

Evaluation metrics, loss and post processing. We consider multiple quantities to quantify the performance of the partitioning: the F1 score, the Rand Index (RI), and the Adjusted Rand Index (ARI = \((RI - \mathbb{E}[RI])/(1 - \mathbb{E}[RI]))\). All models are trained to minimize the F1 score. We make sure the adjacency matrix of the output graph encodes a valid partitioning of nodes to clusters by considering any connected components as a clique.

Results. We compare the results of all learning based methods and a typical baseline algorithm used in particle physics (AVR). We also add the results of a trivial baseline that predicts that all nodes have the same progenitor. All models have roughly the same number of parameters. We performed each experiment 11 times with different random initializations, and evaluated the model F1 score, RI and ARI on the test set. The results are shown in Table 1. For bottom and charm jets, which have secondary vertices, both of our models significantly outperform the baselines by 5%-10% in all performance metrics. In light-jets, without secondary decays, our models yield similar scores.

<table>
<thead>
<tr>
<th>Model</th>
<th>F1</th>
<th>RI</th>
<th>ARI</th>
</tr>
</thead>
<tbody>
<tr>
<td>S2G</td>
<td>0.646±0.005</td>
<td>0.756±0.004</td>
<td>0.491±0.006</td>
</tr>
<tr>
<td>SGN</td>
<td>0.655±0.004</td>
<td>0.747±0.006</td>
<td>0.508±0.007</td>
</tr>
<tr>
<td>GNN</td>
<td>0.586±0.003</td>
<td>0.661±0.004</td>
<td>0.381±0.005</td>
</tr>
<tr>
<td>B</td>
<td>0.606±0.002</td>
<td>0.675±0.005</td>
<td>0.411±0.004</td>
</tr>
<tr>
<td>SIAM</td>
<td>0.597±0.002</td>
<td>0.673±0.005</td>
<td>0.396±0.005</td>
</tr>
<tr>
<td>MLP</td>
<td>0.533±0.000</td>
<td>0.643±0.000</td>
<td>0.315±0.000</td>
</tr>
<tr>
<td>AVR</td>
<td>0.565±0.012</td>
<td>0.612±0.018</td>
<td>0.318±0.026</td>
</tr>
<tr>
<td>trivial</td>
<td>0.438±0.030</td>
<td>0.303±0.026</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. Results: partitioning for particle physics.

Baselines. We compare our results to the following baselines: MLP: A standard multilayer perceptron applied to the flattened set features. SIAM: A popular similarity learning model (see e.g., (Zagoruyko and Komodakis, 2015)) based on Siamese networks. This model has the same structure as in equation 2 where \(\phi\) is a Siamese MLP (a non-universal set-to-set function) that is applied independently to each element in the set. We use the same loss we use with our model (according to the task at hand). SIAM-3: The same architecture as SIAM but with a triplet loss (Weinberger et al., 2006) on the learned representations based on \(l/2\) distance, see e.g., (Schroff et al., 2015). Edge predictions are obtained by thresholding distances of pairs of learned representations. GNN: A Graph Neural Network (Morris et al., 2018) applied to the \(k\)-NN \((k \in \{0, 5, 10\})\) induced graph. Edge prediction is done via outer-product (Kipf and Welling, 2016). AVR: A non-learnable geometric-based baseline called Adaptive Vertex Reconstruction (Waltenberger, 2011) typically used for the particle physics problem we tackle.

More applications, architecture, implementation and hyperparameter details and number of parameters can be found in full paper (Serviansky et al., 2020).

4.2. Partitioning for particle physics

Figure 3. Illustration of a particle physics experiment. The task is to partition the set of observed particles based on their point of origin (in blue).
References


