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 $\mathbb{R}^{d_{\text{in}}}$ 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-

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

Proceedings of the 37th International Conference on Machine Learning, Vienna, Austria, PMLR 108, 2020. Copyright 2020 by the author(s).

[1]This is a workshop version of (Serviansky et al., 2020)

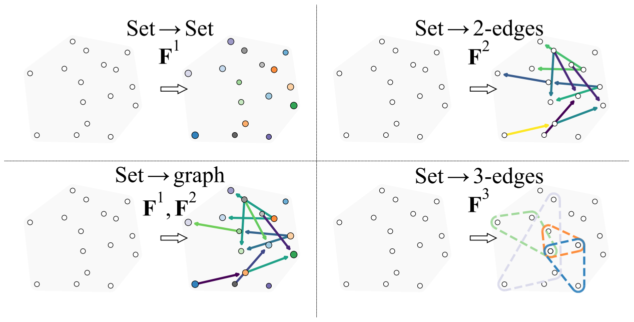

Figure 1. Set-to-graph functions are represented as collections of set-to-k-edge functions.

to-k-edge function learns features on k-edges. That is, given an input set $\mathcal{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\} \subset \mathbb{R}^{d_{\text{in}}}$ we consider functions $\mathbf{F}^k$ attaching feature vectors to k-edges: each k-tuple $(\boldsymbol{x}_{i_1}, \dots, \boldsymbol{x}_{i_k})$ is assigned with an output vector $\mathbf{F}^k(\mathcal{X})_{i_1,i_2,\dots,i_k,:} \in \mathbb{R}^{d_{\text{out}}}$. Now, functions mapping sets to hypergraphs with hyper-edges of size up-to k are modeled by $(\mathbf{F}^1, \mathbf{F}^2, \dots, \mathbf{F}^k)$. For example, functions mapping sets to standard graphs are represented by $(\mathbf{F}^1, \mathbf{F}^2)$, see Figure 1.

Set-to-graph functions are well-defined if they satisfy a property called *equivariance*, and therefore the set-to-graph problem is an instance of the bigger class of equivariant learning (Cohen and Welling, 2016; Ravanbakhsh et al., 2017; Kondor and Trivedi, 2018). A natural approach for learning equivariant set-to-graph model is using out-of-the-box full equivariant model as in (Maron et al., 2019b).

A central question is: Are equivariant models *universal* for set-to-graph functions? That is, can equivariant models approximate any continuous equivariant function? In equivariant learning literature set-to-set models (Zaheer et al., 2017; Qi et al., 2017) are proven equivariant universal (Keriven and Peyré, 2019; Segol and Lipman, 2020; Sannai et al., 2019; Maron et al., 2020). In contrast, the situation for graph-to-graph equivariant models is more intricate: some models, such as message passing (a.k.a. graph convolutional networks), are known to be non-universal (Xu et al., 2019; Morris et al., 2018; Maron et al., 2019a; Chen et al., 2019), while high-order equivariant models are known to be universal (Maron et al., 2019c) but require using high order tensors and therefore not practical. Universality of equivariant set-to-graph models is not known, as far as we are aware. In particular, are high order tensors required for universality (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: $\mathbf{F}^k = \boldsymbol{\psi} \circ \boldsymbol{\beta} \circ \boldsymbol{\phi}$, where $\boldsymbol{\phi}$ is a set-to-set model, $\boldsymbol{\beta}$ is a *non-learnable* broadcasting set-to-graph layer, and $\boldsymbol{\psi}$ 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 $\mathbb{R}^{d_{\text{in}}}$ to hypergraphs with n nodes (think of the nodes as corresponding to the set elements), and arbitrary k-edge feature vectors in $\mathbb{R}^{d_{\text{out}}}$, 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 $\mathbf{F}^k : \mathbb{R}^{n\times d_{\text{in}}} \to \mathbb{R}^{n^k \times d_{\text{out}}}$. Consequently, a set-to-hypergraph function would be modeled as a sequence $(\mathbf{F}^1, \mathbf{F}^2, \ldots, \mathbf{F}^K)$, for target hypergraphs with hyperedges of maximal size K. For example, $\mathbf{F}^2$ learns pairwise relations in a set; and $(\mathbf{F}^1, \mathbf{F}^2)$ is a function from sets to graphs (outputs both node features and pairwise relations); see Figure 1.

Our goal is to design equivariant neural network models for $\mathbf{F}^k$ 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 $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)^T \in \mathbb{R}^{n\times d_{\text{in}}}$ represents a set of n vectors $\boldsymbol{x}_i \in \mathbb{R}^{d_{\text{in}}}$ 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) $\sigma : [n] \to [n]$, where $[n] = \{1, \ldots, n\}$. We denote by $\sigma \cdot \boldsymbol{X}$ the matrix resulting in reordering the rows of $\boldsymbol{X}$ by the permutation σ, i.e., $(\sigma \cdot \boldsymbol{X})_{i,j} = \boldsymbol{X}_{\sigma^{-1}(i),j}$. In this notation, $\boldsymbol{X}$ and $\sigma \cdot \boldsymbol{X}$ represent the same set, for all permutations σ.

k-edges are represented as a tensor $\mathbf{Y} \in \mathbb{R}^{n^k \times d_{\text{out}}}$, where $\mathbf{Y}_{\boldsymbol{i},:} \in \mathbb{R}^{d_{\text{out}}}$ denotes the feature vector attached to the k-edge defined by the k-tuple $(\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_k})$, where $\boldsymbol{i} = (i_1, i_2, \ldots, i_k) \in [n]^k$ 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 $\sigma \cdot \mathbf{Y}$ by $(\sigma \cdot \mathbf{Y})_{\boldsymbol{i},j} = \mathbf{Y}_{\sigma^{-1}(\boldsymbol{i}),j}$, where $\sigma^{-1}(\boldsymbol{i}) = (\sigma^{-1}(i_1), \sigma^{-1}(i_2), \ldots, \sigma^{-1}(i_k))$, then $\mathbf{Y}$ and $\sigma \cdot \mathbf{Y}$ represent the same k-edge data, for all $\sigma \in S_n$.

Equivariance. A sufficient condition for $\mathbf{F}^k$ to represent a well-defined map between sets $\boldsymbol{X} \in \mathbb{R}^{n\times d_{\text{in}}}$ and k-edge data $\mathbf{Y} \in \mathbb{R}^{n^k\times d_{\text{out}}}$ is *equivariance* to permutations, namely

$$\mathbf{F}^k(\sigma \cdot \boldsymbol{X}) = \sigma \cdot \mathbf{F}^k(\boldsymbol{X}), \tag{1}$$

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

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

$$\mathbf{F}^k(\boldsymbol{X}; \theta) = \boldsymbol{\psi} \circ \boldsymbol{\beta} \circ \boldsymbol{\phi}(\boldsymbol{X}), \tag{2}$$

where $\boldsymbol{\phi}, \boldsymbol{\beta}$, and $\boldsymbol{\psi}$ will be defined soon. For $\mathbf{F}^k$ to be equivariant it is sufficient that its constituents, namely $\boldsymbol{\phi}, \boldsymbol{\beta}, \boldsymbol{\psi}$, are equivariant. That is, $\boldsymbol{\phi}, \boldsymbol{\beta}, \boldsymbol{\psi}$ 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 $(\mathbf{F}^1, \mathbf{F}^2)$. Similarly, set-to-hypergraph function would require $(\mathbf{F}^1, \ldots, \mathbf{F}^K)$, where K is the maximal hyperedge size. Figure 1 shows an illustration of set-to-k-edge and set-to-graph functions

$\boldsymbol{\phi}$ component. $\boldsymbol{\phi} : \mathbb{R}^{n\times d_{\text{in}}} \to \mathbb{R}^{n\times d_1}$ is a set-to-set equivariant model, that is $\boldsymbol{\phi}$ is mapping sets of vectors in $\mathbb{R}^{d_{\text{in}}}$ to sets of vectors in $\mathbb{R}^{d_1}$. To achieve the universality goal we will need $\boldsymbol{\phi}$ to be universal as set-to-set model; that is, $\boldsymbol{\phi}$ can approximate arbitrary continuous set-to-set functions. Several options exists (Keriven and Peyré, 2019; Sannai et al., 2019) although probably the simplest option is either DeepSets (Zaheer et al., 2017) or one of its variations; all were proven to be universal in (Segol and Lipman, 2020).

$\boldsymbol{\beta}$ component. $\boldsymbol{\beta} : \mathbb{R}^{n\times d_1} \to \mathbb{R}^{n^k\times d_2}$ 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 $\mathbb{R}^{n\times d_1} \to \mathbb{R}^{n^k\times d_2}$ is of dimension $d_1 d_2 \text{bell}(k+1)$ 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 $\boldsymbol{\beta}(\boldsymbol{X})_{\boldsymbol{i},:} = [\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}, \ldots, \boldsymbol{x}_{i_k}]$ where $\boldsymbol{i} = (i_1, \ldots, i_k)$ and brackets denote concatenation in the feature dimension, that is, for $\boldsymbol{A} \in \mathbb{R}^{n^k\times d_a}$, $\boldsymbol{B} \in \mathbb{R}^{n^k\times d_b}$ their concatenation is $[\boldsymbol{A}, \boldsymbol{B}] \in \mathbb{R}^{n^k\times(d_a+d_b)}$. Therefore, the feature output dimension of $\boldsymbol{\beta}$ is $d_2 = kd_1$.

As an example, consider the graph case, where $k = 2$. In this case $\boldsymbol{\beta}(\boldsymbol{X})_{i_1,i_2,:} = [\boldsymbol{x}_{i_1}, \boldsymbol{x}_{i_2}]$. This function is illus-

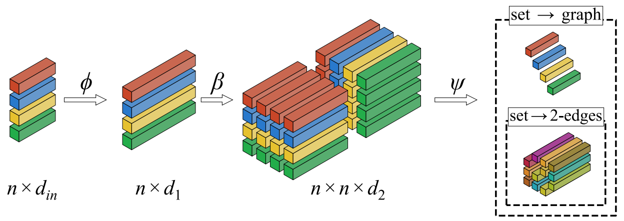

Figure 2. The model architecture for the Set-to-graph and set-to-2-edge functions.

trated in Figure 2 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 a single feature $\boldsymbol{\beta}(\boldsymbol{X})_{\boldsymbol{i}} = \boldsymbol{x}_{i_1}$. Permuting the rows of $\boldsymbol{X}$ by a permutation σ we get $\boldsymbol{\beta}(\sigma\cdot\boldsymbol{X})_{\boldsymbol{i},j} = \boldsymbol{x}_{\sigma^{-1}(i_1),j} = \boldsymbol{\beta}(\boldsymbol{X})_{\sigma^{-1}(\boldsymbol{i}),j} = (\sigma\cdot\boldsymbol{\beta}(\boldsymbol{X}))_{\boldsymbol{i},j}$.

$\boldsymbol{\psi}$ component. $\boldsymbol{\psi}:\mathbb{R}^{n^k\times d_2}\to\mathbb{R}^{n^k\times d_{\text{out}}}$ 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 $d_2 d_{\text{out}}\text{bell}(2k)$ 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 $\boldsymbol{m}:\mathbb{R}^{d_2}\to\mathbb{R}^{d_{\text{out}}}$ independently to each feature vector in the input tensor $\mathbf{X}\in\mathbb{R}^{n^k\times d_2}$. That is, we use $\boldsymbol{\psi}(\mathbf{X})_{\boldsymbol{i},:} = \boldsymbol{m}(\mathbf{X}_{\boldsymbol{i},:})$. Figure 2 illustrates set-to-2-edges and set-to-graph models incorporating the three components $\boldsymbol{\phi},\boldsymbol{\beta},\boldsymbol{\psi}$ discussed above.

3. Universality of set-to-graph models.

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

Theorem 1. *The model $\mathbf{F}^k$ 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-2-edge model, $\mathbf{F}^2$, 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 $d_{\text{out}}=1$ case; the general case is implied by applying the argument for each output feature dimension independently. Let $\mathbf{G}^2$ be an arbitrary continuous equivariant set-to-2-edge function $\mathbf{G}^2: K\subset\mathbb{R}^{n\times d_{\text{in}}}\to\mathbb{R}^{n\times n}$. We want to approximate $\mathbf{G}^2$ with our model $\mathbf{F}^2$. First, note that without losing generality we can assume $\mathbf{G}^2(\boldsymbol{X})$ has a simple spectrum (i.e., eigenvalues are all different) for all $\boldsymbol{X}\in K$. Indeed, if this is not the case we can always choose $\lambda>0$ sufficiently large and consider $\mathbf{G}^2+\lambda\text{diag}(1,2,\ldots,n)$. This diagonal addition does not change the 2-edge values assigned by $\mathbf{G}^2$, and it guarantees a simple specturm using standard hermitian matrix eigenvalue perturbation theory (see (Stewart, 1990), Section IV:4).

Now let $\mathbf{G}^2(\boldsymbol{X}) = \boldsymbol{U}(\boldsymbol{X})\boldsymbol{\Sigma}(\boldsymbol{X})\boldsymbol{V}(\boldsymbol{X})^T$ be the SVD of $\mathbf{G}^2(\boldsymbol{X})$, where $\boldsymbol{U}=[\boldsymbol{u}_1,\ldots,\boldsymbol{u}_n]$ and $\boldsymbol{V}=[\boldsymbol{v}_1,\ldots,\boldsymbol{v}_n]$. Since $\mathbf{G}^2(\boldsymbol{X})$ has a simple spectrum, $\boldsymbol{U},\boldsymbol{V},\boldsymbol{\Sigma}$ are all continuous in $\boldsymbol{X}$; $\boldsymbol{\Sigma}$ is unique, and $\boldsymbol{U},\boldsymbol{V}$ are unique up to a sign flip of the singular vectors (i.e., columns of $\boldsymbol{U},\boldsymbol{V}$) (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 $\mathbf{G}^2$ imply that $\boldsymbol{U},\boldsymbol{V}$ are continuous *set-to-set* equivariant and $\boldsymbol{\Sigma}$ is a continuous *set invariant* function:

$$\begin{aligned}&(\sigma\cdot\boldsymbol{U}(\boldsymbol{X}))\boldsymbol{\Sigma}(\boldsymbol{X})(\sigma\cdot\boldsymbol{V}(\boldsymbol{X}))^T\\ &=\sigma\cdot\boldsymbol{G}(\boldsymbol{X})=\boldsymbol{G}(\sigma\cdot\boldsymbol{X})\\ &=\boldsymbol{U}(\sigma\cdot\boldsymbol{X})\boldsymbol{\Sigma}(\sigma\cdot\boldsymbol{X})\boldsymbol{V}(\sigma\cdot\boldsymbol{X})^T.\end{aligned}\tag{3}$$

Lastly, since $\boldsymbol{\phi}$ is set-to-set universal there is a choice of its parameters so that it approximates arbitrarily well the equivariant set-to-set function $\boldsymbol{Y}=[\boldsymbol{U},\boldsymbol{V},\mathbf{1}\mathbf{1}^T\boldsymbol{\Sigma}]$. The $\boldsymbol{\psi}$ component can be chosen by noting that $\mathbf{G}^2(\boldsymbol{X})_{i_1,i_2} = \sum_{j=1}^n\sigma_j\boldsymbol{U}_{i_1,j}\boldsymbol{V}_{i_2,j} = \boldsymbol{p}(\boldsymbol{\beta}(\boldsymbol{Y})_{i_1,i_2,:})$, where σ_j are the singular values, and $\boldsymbol{p}:\mathbb{R}^{6n}\to\mathbb{R}$ is a cubic polynomial. To conclude pick $\boldsymbol{m}$ to approximate $\boldsymbol{p}$ sufficiently well so that $\boldsymbol{\psi}\circ\boldsymbol{\beta}\circ\boldsymbol{\phi}$ approximates $\mathbf{G}^2$ to the desired accuracy.

To achieve uniqueness of the singular vectors up-to a sign we can add, e.g., the following assumption: $\mathbf{1}^T\boldsymbol{u}_i(\boldsymbol{X})\neq 0\neq\mathbf{1}^T\boldsymbol{v}_i(\boldsymbol{X})$ for all singular vectors and $\boldsymbol{X}\in K$. Using this assumption we can always pick $\boldsymbol{u}_i(\boldsymbol{X})$, $\boldsymbol{v}_i(\boldsymbol{X})$ in the SVD so that $\mathbf{1}^T\boldsymbol{u}_i(\boldsymbol{X})>0$, $\mathbf{1}^T\boldsymbol{v}_i(\boldsymbol{X})>0$, 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 $\mathbf{F}^2$ and $\mathbf{F}^3$ (resp.) models for these learning tasks. for $\mathbf{F}^2$, $\boldsymbol{\phi}$ is implemented using DeepSets (Zaheer et al., 2017) with 5 layers and output dimension $d_1\in\{5,80\}$; $\boldsymbol{\psi}$ is implemented with an MLP, $\boldsymbol{m}$, with $\{2,3\}$ layers with input dimension d_2 defined by d_1 and $\boldsymbol{\beta}$. $\boldsymbol{\beta}$ is implemented according to Section 2: for $k=2$ it uses $d_2=2*d_1$ output features. **S2G+**: For the $k=2$ case we have also tested a more general (but not more expressive) broadcasting $\boldsymbol{\beta}$ defined using the full equivariant basis $\mathbb{R}^n\to\mathbb{R}^{n^2}$ from (Maron et al., 2019b) that contains $\text{bell}(3)=5$ basis operations. This broadcasting layer gives $d_2=5*d_1$.

	Model	F1	RI	ARI
B	S2G	0.646±0.003	0.736±0.004	0.491±0.006
	S2G+	**0.655±0.004**	**0.747±0.006**	**0.508±0.007**
	GNN	0.586±0.003	0.661±0.004	0.381±0.005
	SIAM	0.606±0.002	0.675±0.005	0.411±0.004
	SIAM-3	0.597±0.002	0.673±0.005	0.396±0.005
	MLP	0.533±0.000	0.643±0.000	0.315±0.000
	AVR	0.565	0.612	0.318
	trivial	0.438	0.303	0.026
C	S2G	0.747±0.001	0.727±0.003	0.457±0.004
	S2G+	**0.751±0.002**	**0.733±0.003**	**0.467±0.005**
	GNN	0.720±0.002	0.689±0.003	0.390±0.005
	SIAM	0.729±0.001	0.695±0.002	0.406±0.004
	SIAM-3	0.719±0.001	0.710±0.003	0.421±0.005
	MLP	0.686±0.000	0.658±0.000	0.319±0.000
	trivial	0.610	0.472	0.078
	AVR	0.695	0.650	0.326
L	S2G	0.972±0.001	**0.970±0.001**	**0.931±0.003**
	S2G+	0.971±0.002	0.969±0.002	0.929±0.003
	GNN	0.972±0.001	**0.970±0.001**	0.929±0.003
	SIAM	**0.973±0.001**	**0.970±0.001**	0.925±0.003
	SIAM-3	0.895±0.006	0.876±0.008	0.729±0.015
	MLP	0.960±0.000	0.957±0.000	0.894±0.000
	trivial	0.910	0.867	0.675
	AVR	0.970	0.965	0.922

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 ϕ 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 $l2$ 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

We tackle learning set-to-2-edge functions. Here, each training example is a pair $(\boldsymbol{X}, \boldsymbol{Y})$ where $\boldsymbol{X}$ is a set $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)^T \in \mathbb{R}^{n \times d_{\text{in}}}$ and $\boldsymbol{Y} \in \{0,1\}^{n \times n}$ is an adjacency matrix (the diagonal of $\boldsymbol{Y}$ 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 progenitor, 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-2-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 *j*ets) corresponding to three different fundamental data generating processes labeled bottom-jets, charm-jets, and light-jets (B/C/L). The important distinction between the flavors is the typical number of particles in each set. Since it is impossible to label real data collected in the detectors at the LHC, algorithms for particle physics are typically designed with high-fidelity simulators, which can provide labeled training data. These algorithms are then applied to and calibrated with real data collected by the LHC experiments. The generated sets are small, ranging from 2 to 14 elements each, with around 0.9M sets divided to train/val/test using the ratios 0.6/0.2/0.2. Each set element has 10 features (d_{in}). More information can be found in the supplementary material.

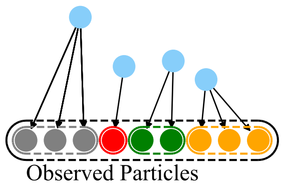

Figure 3. Illustration of a particle physics experiment. The task is to partition the set of observed particles based on their point of origin (in blue).

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 ($\text{ARI} = (\text{RI} - \mathbb{E}[\text{RI}])/(1 - \mathbb{E}[\text{RI}])$). 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.

References

Chen, Z., Villar, S., Chen, L., and Bruna, J. (2019). On the equivalence between graph isomorphism testing and function approximation with gnns. *arXiv preprint arXiv:1905.12560*.

Cohen, T. and Welling, M. (2016). Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999.

Keriven, N. and Peyré, G. (2019). Universal invariant and equivariant graph neural networks. *CoRR*, abs/1905.04943.

Kipf, T. N. and Welling, M. (2016). Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*.

Kondor, R. and Trivedi, S. (2018). On the generalization of equivariance and convolution in neural networks to the action of compact groups. *arXiv preprint arXiv:1802.03690*.

Maron, H., Ben-Hamu, H., Serviansky, H., and Lipman, Y. (2019a). Provably powerful graph networks. *arXiv preprint arXiv:1905.11136*.

Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. (2019b). Invariant and equivariant graph networks. In *International Conference on Learning Representations*.

Maron, H., Fetaya, E., Segol, N., and Lipman, Y. (2019c). On the universality of invariant networks. *arXiv preprint arXiv:1901.09342*.

Maron, H., Litany, O., Chechik, G., and Fetaya, E. (2020). On learning sets of symmetric elements. *arXiv preprint arXiv:2002.08599*.

Morris, C., Ritzert, M., Fey, M., Hamilton, W. L., Lenssen, J. E., Rattan, G., and Grohe, M. (2018). Weisfeiler and leman go neural: Higher-order graph neural networks. *arXiv preprint arXiv:1810.02244*.

O'Neil, K. A. (2005). Critical points of the singular value decomposition. *SIAM journal on matrix analysis and applications*, 27(2):459–473.

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

Ravanbakhsh, S., Schneider, J., and Poczos, B. (2017). Equivariance through parameter-sharing. *arXiv preprint arXiv:1702.08389*.

Sannai, A., Takai, Y., and Cordonnier, M. (2019). Universal approximations of permutation invariant/equivariant functions by deep neural networks. *arXiv preprint arXiv:1903.01939*.

Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.

Segol, N. and Lipman, Y. (2020). On universal equivariant set networks. In *International Conference on Learning Representations*.

Serviansky, H., Segol, N., Shlomi, J., Cranmer, K., Gross, E., Maron, H., and Lipman, Y. (2020). Set2graph: Learning graphs from sets. *arXiv preprint arXiv:2002.08772*.

Stewart, G. W. (1990). Matrix perturbation theory.

Waltenberger, W. (2011). RAVE: A detector-independent toolkit to reconstruct vertices. *IEEE Trans. Nucl. Sci.*, 58:434–444.

Weinberger, K. Q., Blitzer, J., and Saul, L. K. (2006). Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). How powerful are graph neural networks? In *International Conference on Learning Representations*.

Zagoruyko, S. and Komodakis, N. (2015). Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4353–4361.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. (2017). Deep sets. In *Advances in neural information processing systems*, pages 3391–3401.