Equivariant Graph Neural Networks

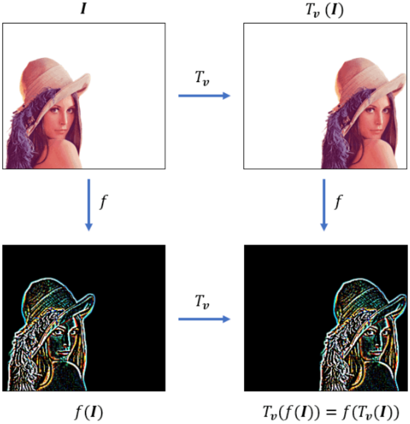
Kfir Eliyahu Ben Eliav Jonathan Kouchly

December 28, 2024

Outline

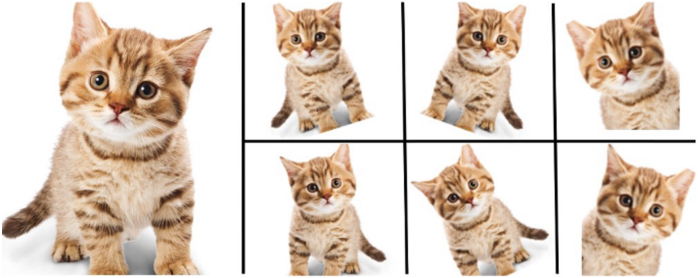
Motivation

- Our neural networks can operate on data of many types.
- We often work with images, text, audio, graphs and more.
- These data types have different structures and qualities, and we would like to build architectures that best suit them.



Motivation

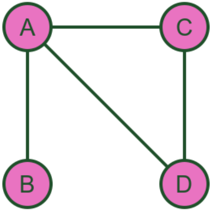
- A cat is a cat no matter how you look at it.



- It is acceptable to assume that being invariant to the rotation of the cat is a good property for a classification network.

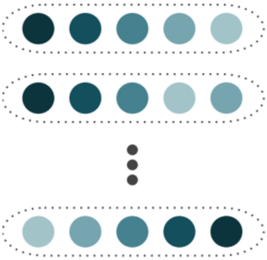
Motivation

- Our focus today is on sets and graph data.



Simple graph

	A	B	C	D
A	0	1	1	1
B	1	0	0	0
C	1	0	0	1
D	1	0	1	0



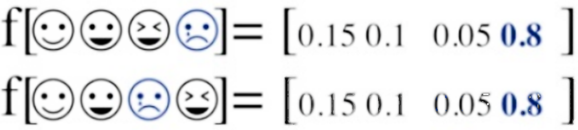
Construction of an Equivariant Neural Network

- When contructing an equivariant neural network, two things should always be considered:
  1. The symmetries of the data:
     What inherent structure should our model be oblivious to?
  2. The space of functions learnable by the network:
     Are we fully utilizing the space of functions that are equivariant

- The permutation group S_n is the group of all permutations of n elements.
- It has $n!$ elements, representing the $n!$ ways to order n elements.
- Given a set $X = \{x_1, x_2, \ldots, x_n\}$, a permutation $\pi \in S_n$ is a bijection $\pi : X \to X$
- e.g. $x = (x_1, x_2, x_3)$, and $\pi = (1, 2, 3) \in S_3$ is the permutation that maps $1 \to 2$, $2 \to 3$ and $3 \to 1$.
- We denote the **action** of π on x as $\pi x = (x_3, x_1, x_2)$.

Permutation Invariance

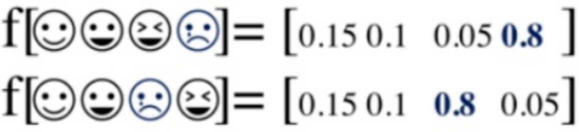
- Let $H \leq S_n$ be a subgroup of the symmetric group.
- $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is *permutation invariant* if $f(x) = f(\pi x)$ for all $\pi \in H$.



Permutation Equivariance

- Let $H \leq S_n$ be a subgroup of the symmetric group.

- $f : \mathbb{R}^n \to \mathbb{R}^n$ is *permutation equivariant* if $\pi f(x) = f(\pi x)$ for all $\pi \in H$.



Permutation of a Set

- Assume our set is $X = \{x_1, x_2, \ldots, x_n\}$.
- We can represent X as a matrix $X \in \mathbb{R}^{n \times d}$.
- Any permutation $g \in S_n$ can be represented as a permutation matrix $\boldsymbol{P} \in \mathbb{R}^{n \times n}$,
- The action of g on X is then $\boldsymbol{P}X$.
- An invariant neural network is a function $f : \mathbb{R}^{n \times d} \to \mathbb{R}^{d'}$ such that $f(X) = f(\boldsymbol{P}X)$.
- An equivariant neural network is a function $f : \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times d'}$ such that $\boldsymbol{P}f(X) = f(\boldsymbol{P}X)$.

Permutation of a Graph

- Our data is now a graph adjacency matric $A \in \mathbb{R}^{n \times n}$.
- A permutation matrix $\boldsymbol{P} \in \mathbb{R}^{n \times n}$ acts on the adjacency matrix A.
- The action of $\boldsymbol{P}$ on A is:

$$\boldsymbol{P}^{\boldsymbol{T}} A \boldsymbol{P}$$

Equivariant Network Construction

Theorem

Let L be a linear equivariant layer, and let f be a neural network constructed be stacking L and non-linearities σ. Then f is permutation equivariant.

Proof.

Let x be a set of n elements, and let $g \in S_n$ be a permutation.

$$f(gx) = L(\sigma(L(\sigma(\dots L(gx)\dots)))) = L(\sigma(L(\dots g\sigma(L(x))\dots))) = \dots$$

$$gL(\sigma(L(\sigma(\dots L(x)\dots)))) = gf(x)$$

□

Invariant Network Construction

Theorem

Let f be an equvariant neural network and let ϕ be a permutation invariant function. Then $h = \phi(f(x))$ is a permutation invariant neural network.

Proof.

Let x be a set of n elements, and let $g \in S_n$ be a permutation.

$$h(gx) = \phi(f(gx)) = \phi(gf(x)) = \phi(f(x)) = h(x)$$

$\square$

Deep Sets

- A seminal work in the field of equivariant neural networks.
- Recall the two properties we mentioned earlier (symmetries of the data and the space of functions learnable by the network).
- *DeepSets* is an architecture that is equivariant to set permutations and is maximally expressive in the space of permutation equivariant functions.
- We are going to see the construction and prove it satisfies equivariance and expressiveness.

DeepSets Layer

- We saw a general structure of an invariant and equivariant network.
- To fill in the details, we need to define the equivariant layer L and the invariant function ϕ.

Definition

Consider a set $x = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}$. Define its representation $\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$.
A *DeepSets* layer is defined as

$$L(x) = \lambda \mathrm{I}\mathbf{x} + \mathbf{x}\mathbf{1}$$

.

DeepSets Layer

Definition

Consider a set $x = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \mathbb{R}$.
A *DeepSets* layer is defined as

$$L(x) = \lambda \mathrm{I}\mathbf{x} + \mathbf{x}\mathbf{1}$$

.

Theorem

A DeepSets layer is permutation equivariant.

Proof.

Let x be a set of n elements, and let $g \in S_n$ be a permutation.

$$L(gx) = \lambda \mathrm{I}(gx) + (gx)\mathbf{1} = g(\lambda \mathrm{I}x) + x\mathbf{1} = gL(x)$$

$\square$

DeepSets Layer

Theorem

Any linear equivariant layer is of the shape $L(x) = \lambda \mathrm{I}\mathbf{x} + \mathbf{x1}$.

Proof (through example).

Let $\mathbf{x} = x_1, x_2, x_3$ be a set, $W \in \mathbb{R}^{3\times 3}$, and $L(x) = W\mathbf{x}$ such that L is equivariant, W $= \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}$ Consider permutation $g = (2,3) \in S_3$.

Note that $g\left(W\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right)_1 = (Wx)_1 = (Wgx)_1 = \left(W\begin{pmatrix} x_1 \\ x_3 \\ x_2 \end{pmatrix}\right)_1$ from equivariance.

Proof (through example) - continued.

The previous equation can be written as:

$$ax_1 + bx_2 + cx_3 = ax_1 + bx_3 + cx_2$$

This should hold for any choice of $\mathbf{x} \Rightarrow b = c,\ d = f, g = h$.
For the permutation $\sigma = (1, 3, 2)$:

$$W(\sigma x) = W \begin{pmatrix} x_2 \\ x_3 \\ x_1 \end{pmatrix} = \begin{pmatrix} ax_2 + bx_3 + bx_1 \\ dx_2 + ex_3 + dx_1 \\ fx_2 + fx_3 + ix_1 \end{pmatrix}$$

$$\sigma W \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = (1, 3, 2) \begin{pmatrix} ax_1 + bx_2 + bx_3 \\ dx_1 + ex_2 + dx_3 \\ fx_1 + fx_2 + ix_3 \end{pmatrix} = \begin{pmatrix} dx_1 + ex_2 + dx_3 \\ fx_1 + fx_2 + ix_3 \\ ax_1 + bx_2 + bx_3 \end{pmatrix} \overset{!}{=} W \begin{pmatrix} x_2 \\ x_3 \\ x_1 \end{pmatrix}$$

DeepSets Layer

Proof (through example) - continued.

We can use the same logic to show that $a = e = i$,
$b = c = d = f = g = h = i$
We get that all values in the diagonal must be equal and all values in the off-diagonal must be equal, and W must have the form:

$$W = \begin{pmatrix} a & b & b \\ b & a & b \\ b & b & a \end{pmatrix}$$

. $\square$

DeepSets Network

- We have an initial layer L, which we proved is equivariant.
- A deep sets invariant network is now constructed as:

$$f(x) = \phi(L\sigma(L\sigma(\dots L\sigma(L(x))\dots))) \quad \text{where} \quad \phi(x) = \sum_{i=1}^{n} x_i$$

- It is easy to see that ϕ is permutation invariant, and thus f is permutation invariant.
- For a classification network, take some classification module ρ (e.g. an MLP), and define the final network as:

$$h(x) = \rho(f(x))$$

DeepSets Network

- Notice that the network is only defined for sets with elements in $\mathbb{R}$.

- We can extend this to a set $X = \{x_1, x_2, \dots x_n\} \subset \mathbb{R}^d$ with representation $\mathbf{X} \in \mathbb{R}^{n \times d}$ by defining L as:

$$L(x) = \mathbf{X} W_1 + \mathbf{1}\mathbf{1}^T \mathbf{X} W_2$$

- This keeps the general structure of the layer: a linear transformation of the distinct elements of the set summed with the mean of the set.

DeepSets Expressivity

- We have shown that the *DeepSets* network is permutation invariant.
- We now want to show that it is maximally expressive in the space of permutation invariant functions.
- We show outline for the proof, showing *DeepSets* can separate any two sets that are not equal.

Theorem

Let $x, y \in \mathbb{R}^{n \times d} \quad s.t \quad x \neq gy \quad \forall g \in S_n$. *Then there exists a DeepSets network that separates* x *and* y*, namely* $F(x;\theta) \neq F(y;\theta)$.

DeepSets Expressivity

Proof outline:

1. Consider distinct rows of x, y.
2. Construct input output pairs using standard basis elements (one hot encodings of the elements in the set).
3. In each layer $L(x) = \mathbf{X}W_1 + \mathbf{1}\mathbf{1}^T\mathbf{X}W_2$, set $W_2 = 0$, collapsing our network to a standard MLP.
4. According to a result of the universal approximation theorem, we can learn a one to one mapping between the input and output pairs.
5. Output for each set the sum of the output pairs.
6. For each set, we computed the hitogram of its elements, which is a unique representation of the set.

Invariant and Equivariant Graph Networks

- We saw a simple, maximally expressive equivariant network for sets.

- We now want to extend this to graphs.

Invariant and Equivariant Graph Networks

- Recall the action of the permutation matrix $\boldsymbol{P}$ on the adjacency matrix A is:

$$\boldsymbol{P}^{\boldsymbol{T}} A \boldsymbol{P}$$

- Lets use this to define a Linear invariant layer $L \in \mathbb{R}^{1 \times n^2}$. We want to satisfy the following:

$$L \boldsymbol{P}^{\boldsymbol{T}} A \boldsymbol{P} = L(A)$$

- But working with bilinear forms is hard. We can define an equivalent condition by column stacking the adjacency matrix to get a vector, and then define a linear layer $L \in \mathbb{R}^{1 \times n^2}$.

$$L\text{vec}(\boldsymbol{P}^{T} A \boldsymbol{P}) = L\text{vec}(A)$$

- To be clear, L is a layer which takes a vector of size n^2 and outputs a scalar.

Invariant and Equivariant Graph Networks

- We now introduce a crucial property of the Kronecker product:

$$\text{vec}(XAY) = Y^T \otimes X\text{vec}(A)$$

- Using this, the previous condition can be written as:

$$L(P^T \otimes P^T)\text{vec}(A) = L\text{vec}(A)$$

- For this condition to hold for all A, we need L to satisfy:

$$L(P^T \otimes P^T) = L$$

- transposing the equation, and with severe abuse of notation ($L^T = \text{vec(L)}$), we finally get:

$$(P \otimes P)\text{vec}(L) = \text{vec}(L)$$

- After some work and moderate logic jumps, we got a condition for a permutation invariant linear layer L.
- Developing the condition for an equivariant layer $L \in \mathbb{R}^{n^2 \times n^2}$ is very similar:

$$L\text{vec}(\boldsymbol{P}^T A \boldsymbol{P}) = \boldsymbol{P}^T L\text{vec}(A)\boldsymbol{P}$$

- Using the Kronecker product property, we get:

$$L(P^T \otimes P^T)\text{vec}(A) = (P^T \otimes P^T)L\text{vec}(A)$$

- This should hold for every A, so we get:

$$L(P^T \otimes P^T) = (P^T \otimes P^T)L$$

Invariant and Equivariant Graph Networks

- $(P^T \otimes P^T)$ is an $n^2 \times n^2$ premutation matrix, and its inverse is $(P \otimes P)$.

$$(P \otimes P)L(P^T \otimes P^T) = L$$

- Once again using the Kronecker product property, we get:

$$\text{vec}(L) = \text{vec}(P \otimes P)L(P^T \otimes P^T) = (P \otimes P) \otimes (P \otimes P)\text{vec}(L)$$
$$= (P \otimes P \otimes P \otimes P)\text{vec}(L)$$

- The previous conditions are developed for a matrix $A \in \mathbb{R}^{n^2}$ expressing the relations between pairs of nodes.
- We can extend this to a matrix $A \in \mathbb{R}^{n^k}$ expressing the relations between every k tuple of nodes.
- for invariant layers $L \in \mathbb{R}^{1 \times n^k}$:

$$P^{\otimes k}\text{vec}(L) = \text{vec}(L)$$

- for equivariant layers $L \in \mathbb{R}^{n^k \times n^k}$:

$$P^{\otimes 2k}L = \text{vec}(L)$$

Invariant and Equivariant Graph Networks

- We developed conditions for a permutation invariant and equivariant linear layer L. Call these the Fixed Point Equations.
- Unfortunately, compared to the simple structure of the *DeepSets* layer, these equations are less intepretable. Let's try and make some sense of them.
- The condition we got is one that determines the values of the entries of L. We want to understand what entries of L should be equal.
- For an invaratiant layer $L \in \mathbb{R}^{1 \times n^k}$, $L_{i_1, i_2, \ldots, i_k}$ is the weight which multiplies entry $A_{i_1, i_2, \ldots, i_k}$.

Invariant and Equivariant Graph Networks

- The Fixed Point Equation for an invariant layer is:

$$P^{\otimes k}\text{vec}(L) = \text{vec}(L)$$

- The k-th permutation matrix premutes the k-th dimension of the input.

$$P^{\otimes k}\text{vec}(L)_{i_1,\dots i_k} = \text{vec}(L)_{\sigma(i)_1,\dots\sigma(i)_k}$$

- This partitions the entries of L into equivalence classes, where all entries in the same class are equal.

- For $k = 2$ we get 2 equivalence classes:

$$L_{i,j} \quad \text{where} \quad i \neq j \quad \text{and} \quad L_{i,i}$$

Invariant and Equivariant Graph Networks

- For an equivariant layer $L \in \mathbb{R}^{1 \times n^k}$, $L_{i_1, i_2, \ldots, i_k, j_1, j_2, \ldots, j_k}$ is the weight which multiplies entry $A_{i_1, i_2, \ldots, i_k}$ and sends it to entry $j_1, j_2, \ldots, j_k$ of the output.
- The Fixed Point Equation for an equivariant layer is:

$$P^{\otimes 2k} \text{vec}(L) = \text{vec}(L)$$

- The k-th permutation matrix premutes the k-th dimension of the input.

$$P^{\otimes 2k} \text{vec}(L)_{i_1, i_2, \ldots, i_k, j_1, j_2, \ldots, j_k} = \text{vec}(L)_{\sigma(i)_1, \ldots \sigma(i)_k, \sigma(j)_1, \ldots \sigma(j)_k}$$

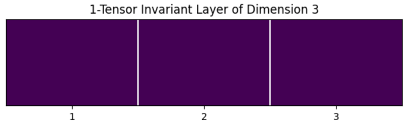
- Once again, this partitions the entries of L into equivalence classes.
- For $k = 2$ we get 15 equivalence classes:

$$\{\{1\}\{2\}\{3\}\{4\}\}, \{\{1,2\}\{3\}\{4\}\}, \{\{1\}\{2,3\}\{4\}\} \ldots \{\{1,2,3,4\}\}$$

In general, the number of equivalence classes is $Bell((2)k)$ for an (equivariant) invariant layer of order k

- Did you notice somthing similar to the *DeepSets* layer?
- The construction of the *DeepSets* layer is identicle to the construction of an equivariant layer of order 1.
- We managed to generalize the equivariant layer to higher orders of node connectivity.
- We also managed to generalize the invariant layer to higher orders of node connectivity.

Invariant and Equivariant Graph Networks



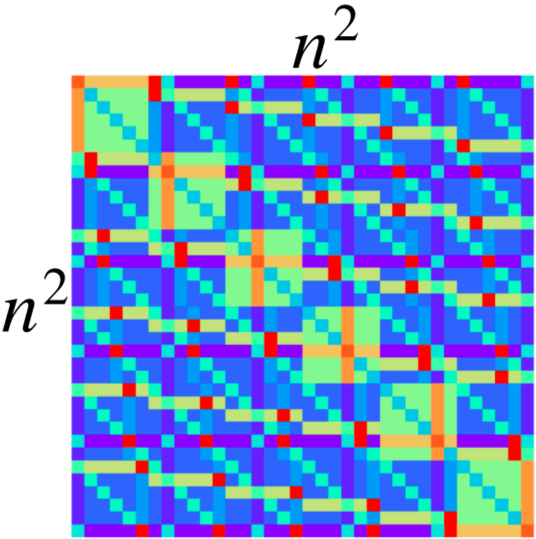


Figure taken from S. Ravanbakhsh (follow-up work)

Invariant and Equivariant Graph Networks

-

Thank You!