E(n) Equivariant Graph Neural Networks and Normalizing Flows

Shih-Hsin Wang

Bwang-Team
Scientific Computing and Imaging Institute
University of Utah

Bi-weekly Meeting, Oct. 14, 2022

Outline

Outline

Definition of a Group

Definition

A group consists of a set G and a binary operation $\cdot : G \times G \to G$, called the group product that satisfies the following axioms:

- Identity: there exists an identity element $e \in G$ s.t.

$$e \cdot g = g = g \cdot e \text{ for any } g \in G$$

- Inverse: for any $g \in G$, there exists an inverse element $g^{-1} \in G$ s.t.

$$g \cdot g^{-1} = e = g^{-1} \cdot g$$

- Associativity: for any $g, h, i \in G$ we have

$$(g \cdot h) \cdot i = g \cdot (h \cdot i)$$

Examples of Groups

- For any vector space V, the general linear group GL(V) is the group of all bijective linear transformations from V to itself.
- If $V \cong \mathbb{R}^n$, we may write GL(n) $=$ GL(V) and it is clear that GL(n)$=$ $\{$all invertible $n \times n$ matrices$\}$
- The orthogonal group O(n) $= \{Q \in GL(n) | Q^T Q = QQ^T = I\}$ is a subgroup of $GL(n)$ containing all the rotations ($\det Q = 1$) and reflections ($\det Q = -1$)
- The Euclidean group E(n) is a group consisting of all isometries of the Euclidean space $\mathbb{R}^n$ (e.g. the transformations of $\mathbb{R}^n$ that **preserve the Euclidean distance**)
  Clearly, O(n) is also a subgroup of E(n)

Examples of Groups

- Indeed, E(n) is parametrized by (Q, g) where $Q \in O(n), g \in \mathbb{R}^n$ and the group product and inverse are defined by

$$
\begin{aligned}
(Q, g) \cdot (Q', g') &:= (QQ', Qg' + g) \\
(Q, g)^{-1} &:= (Q^{-1}, Q^{-1}g)
\end{aligned}
$$

- In this way, we see that each element (Q, g) in E(n) induces a bijective transformation $T(Q, g)$ of $\mathbb{R}^n$

$$
\begin{aligned}
T(Q, g) : \mathbb{R}^n &\to \mathbb{R}^n \\
x &\mapsto Qx + g
\end{aligned}
$$

Group Representation

Definition

A representation of a group G on a vector space V is a map $\rho : G \to GL(V)$ such that

$$\rho(g \cdot h) = \rho(g)\rho(h) \text{ for any } g, h \in G$$

In particular, we say that ρ is a trivial representation if ρ sends all the elements of G to the identity mapping of V.

- For example, $\rho : E(n) \to GL(n+1)$ defined by

$$(Q, g = (g_1, g_2, \cdots, g_n)) \mapsto \begin{bmatrix} 1 & 0 & \cdots & 0 & g_1 \\ 0 & 1 & \cdots & 0 & g_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & g_n \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} Q & 0 \\ 0 & 1 \end{bmatrix}$$

is a representation of E(n) on $\mathbb{R}^n$

Group Action

Definition

A morphism (or a function) from a set X to itself is call an endomorphism of X. We denote the set of all endomorphisms of X by $\mathrm{End}(X)$.
Let G be a group with identity e. A group action α of G on X, which will be written as $G \circlearrowright X$ is a function $\alpha : G \to \mathrm{End}(X)$ such that

$$\alpha(e) = I_X, \alpha(gh) = \alpha(g) \circ \alpha(h) \text{ for any } g, h \in G$$

Without ambiguity, we may say "G acts on X". Moreover, we say that α is trivial if α sends all the elements of G to the identity mapping I_X of X.

- For example, $\alpha_{E(n)} : E(n) \to \mathrm{End}(\mathbb{R}^n)$ defined by sending (Q, g) to $T(Q, g)$ is a group action of E(n) on $\mathbb{R}^n$
- Also, it is clear that a representation of a group G on a vector space V is also a group action of G on V

Equivariance and Invariance

Definition

Let $\alpha_V : G \to GL(V)$ and $\alpha_W : G \to GL(W)$ be the group actions of G on two sets V and W, respectively.
A (nonlinear) function $\phi : V \to W$ is said to be **equivariant** if

$$\phi(\alpha_V(g)(x)) = \alpha_W(g)(\phi(x)) \text{ for any } g \in G, x \in X,$$

that is, we have the following commutative diagram for any $g \in G$

$$\begin{array}{ccc} V & \xrightarrow{\alpha_V(g)} & V \\ \downarrow{\scriptstyle \phi} & & \downarrow{\scriptstyle \phi} \\ W & \xrightarrow{\alpha_W(g)} & W \end{array}$$

In particular, we say that ϕ is **invariant** when α_W is a trivial group action.

Equivariance and Invariance

- Consider the group action $\alpha_{E(n)}$ of E(n) on $\mathbb{R}^n$. We see that the identity map $I : \mathbb{R}^n \to \mathbb{R}^n$ is equivariant.
- Consider the group action $\tilde{\rho} = \rho \oplus \rho$ of E(n) on $\mathbb{R}^n \times \mathbb{R}^n$ which acts on the copies of $\mathbb{R}^n$ separately. Then the function $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ defined by the distance $d(x, y) = ||x - y||^2$ is invariant.
- On the other hand, the function $f : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ defined by $f(x, y) = x - y$ is neither equivariant nor invariant.
- Indeed, f is invariant under the translations and is equivariant under the rotations and reflections.

Remark

Indeed, $\alpha_{E(n)}$ is a map from E(n) to bijective endomorphisms of $\mathbb{R}^n$. So when ϕ is an equivariant function, the transformation of the output is predictable with the understanding of the transformation on the input - no information gets lost when the input is transformed.

Outline

1. Background
2. E(n) Equivariant Graph Neural Networks
3. E(n) Equivariant Normalizing flows

Graph Neural Networks

- Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $v_i \in \mathcal{V}$ and edges $e_{ij} \in \mathcal{E}$
- Let $M = |\mathcal{V}|$ be the number of nodes

Graph Convolutional Layer [Gilmer et al., 2017]

$$\begin{aligned} \boldsymbol{m}_{ij} &= \phi_e(\boldsymbol{h}_i^l, \boldsymbol{h}_j^l, a_{ij}) \\ \boldsymbol{m}_i &= \sum_{j \in \mathcal{N}(i)} \boldsymbol{m}_{ij} \\ \boldsymbol{h}_i^{l+1} &= \phi_h(\boldsymbol{h}_i^l, \boldsymbol{m}_i) \end{aligned} \tag{1}$$

- $\boldsymbol{h}_i^l \in \mathbb{R}^{\mathrm{nf}}$ is the feature embedding of node v_i at layer l
- a_{ij} are the edge attributes
- $\mathcal{N}(i)$ represents the set of neighbors of node v_i
- ϕ_e,ϕ_h are the edge and node operations (approximated by MLPs)

Equivariant Graph Neural Networks

Equivariant Graph Convolutional Layer (EGCL) [Satorras et al., 2021b]

$$
\begin{aligned}
\boldsymbol{m}_{ij} &= \phi_e(\boldsymbol{h}_i^l, \boldsymbol{h}_j^l, \|\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\|^2, a_{ij}) \\
\boldsymbol{x}_i^{l+1} &= \boldsymbol{x}_i^l + C \sum_{j \neq i} (\boldsymbol{x}_i^l - \boldsymbol{x}_j^l) \phi_x(\boldsymbol{m}_{ij}) \\
\boldsymbol{m}_i &= \sum_{j \neq i} \boldsymbol{m}_{ij} \\
\boldsymbol{h}_i^{l+1} &= \phi_h(\boldsymbol{h}_i^l, \boldsymbol{m}_i)
\end{aligned} \tag{2}
$$

- $\boldsymbol{x}_i^l \in \mathbb{R}^{\mathrm{n}}$ is the coordinate embedding of node v_i at layer l
- C is chosen to be $1/(M-1)$ that computes the average of the sum
- $\phi_x : \mathbb{R}^{nf} \to \mathbb{R}$ is a learnable function (approximated by MLPs)

We may simply write

$$\boldsymbol{x}^{l+1}, \boldsymbol{h}^{l+1} = \mathrm{EGCL}(\boldsymbol{x}^l, \boldsymbol{h}^l) \tag{3}$$

Equivariant Graph Neural Networks

EGCL including momentum [Satorras et al., 2021b]

$$\begin{aligned}\boldsymbol{v}_i^{l+1} &= \phi_v(\boldsymbol{h}_i^l)\boldsymbol{v}_i^{\text{init}} + C\sum_{j\neq i}(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l)\phi_x(\boldsymbol{m}_{ij}) \\ \boldsymbol{x}_i^{l+1} &= \boldsymbol{x}_i^l + \boldsymbol{v}_i^{l+1}\end{aligned} \tag{4}$$

- Note that if $\boldsymbol{v}^{\text{init}} = 0$ then this is exactly Equation 3.
- $\phi_v : \mathbb{R}^{nf} \to \mathbb{R}$ is a learnable function (approximated by MLPs)

Inferring the edges

In order to deal with the scalability, we can rewrite the aggregation in the following way:

$$\boldsymbol{m}_i = \sum_{j\in\mathcal{N}(i)} \boldsymbol{m}_{ij} = \sum_{j\neq i} e_{ij}\boldsymbol{m}_{ij}. \tag{5}$$

- e_{ij} is approximated by a soft embedding $\phi_{inf}(\boldsymbol{m}_{ij})$ ($\phi_{inf} : \mathbb{R}^n \to [0,1]$)

EGCL Equivariance

- To show that EGCL is equivariant, it suffices to prove that for any $Q \in$O(n)$, g \in \mathbb{R}^n$, we have

$$Q\boldsymbol{x}^{l+1} + g, \boldsymbol{h}^{l+1} = \text{EGCL}(Q\boldsymbol{x}^l + g, \boldsymbol{h}^l)$$

  where E(n) acts on the feature $\boldsymbol{h}^l$ trivially and acts on the coordinate $\boldsymbol{x}^l$ of each nodes separately.
- Recall that the EGCL is given by

$$\begin{aligned}
\boldsymbol{m}_{ij} &= \phi_e(\boldsymbol{h}_i^l, \boldsymbol{h}_j^l, \|\boldsymbol{x}_i^l - \boldsymbol{x}_j^l\|^2, a_{ij}) \\
\boldsymbol{x}_i^{l+1} &= \boldsymbol{x}_i^l + C\sum_{j\neq i}(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l)\phi_x(\boldsymbol{m}_{ij}) \\
\boldsymbol{m}_i &= \sum_{j\neq i}\boldsymbol{m}_{ij} \\
\boldsymbol{h}_i^{l+1} &= \phi_h(\boldsymbol{h}_i^l, \boldsymbol{m}_i)
\end{aligned}$$

EGCL Equivariance

Proof

Clearly, the first equation is invariant (since all the inputs are invariant) which implies the last two equations are also invariant.
Moreover, we can show that the second equation that updates the position is equivariant.

$$
\begin{aligned}
&(Q\boldsymbol{x}_i^l + g) + C\sum_{j\neq i}[(Q\boldsymbol{x}_i^l + g) - (Q\boldsymbol{x}_j^l + g)]\phi_x(\boldsymbol{m}_{ij}) \\
=&Q\boldsymbol{x}_i^l + g + QC\sum_{j\neq i}(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l)\phi_x(\boldsymbol{m}_{ij}) \\
=&Q(\boldsymbol{x}_i^l + C\sum_{j\neq i}(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l)\phi_x(\boldsymbol{m}_{ij})) + g \\
=&Q\boldsymbol{x}_i^{l+1} + g
\end{aligned}
$$

Outline

Normalizing flows

We are going to introduce a generative model for E(n) Equivariant data. Before that, let's see the central idea behind the model, called **the normalizing flows**. Given the following settings:

- $p_X(x)$: unknown underlying distribution of datapoints where X is the corresponding random variable
- $p_Z(z)$: a simple base distribution (such as a normal distribution) where Z is the corresponding random variable

Goal

Find an invertible transformation g_θ s.t. $X \approx g_\theta(Z)$

Then we can generate a sample from p_Z and then map the sample via the invertible transformation g_θ to a new datapoint

Normalizing flows

- We restrict g_θ to be invertible, i.e. $f_\theta = g_\theta^{-1}$ exists
- Then the inverse function f_θ **flows** in the **normalizing** direction: from a complicated data distribution towards the simpler and more "normal" distribution p_Z
- And we have the change of variables formula:

$$p_X(x) \approx p_Z(z) |\det J_{f_\theta}(x)| \tag{6}$$

  where J_{f_θ} is the Jacobian matrix of f_θ
- Also, we have the change of variables formula for the log density:

$$\log p_X(x) \approx \log p_Z(z) + \log |\det J_{f_\theta}(x)|$$

- However, computing the log determinant has a time cost of $\mathcal{O}(D^3)$ where $f : \mathbb{R}^D \to \mathbb{R}^D$
- Refer to [Kobyzev et al., 2020] for more details

Continuous Normalizing flows

[Chen et al., 2018] defines a generative model similar to those based on 6 which replaces the warping function with an integral of continuous-time dynamics (approximated by a neural network)

$$z = x + \int_0^1 \phi(x(t))dt \tag{7}$$

where $x(0) = x$ and $x(1) = z$

- $x(t)$ is redefined to be a function on time that joins x to z
- the first derivatives of x is predicted by a neural network

$$\frac{d}{dt}x(t) \approx \phi_\theta(x(t))$$

where only require ϕ_θ to be Lipschitz and continuously differentiable

Continuous Normalizing flows

Then we have the **instantaneous change of variables formula** for the change in log-density

$$\frac{d}{dt} \log p_X(x(t)) \approx -\operatorname{Tr} J_{\phi_\theta}(x(t)) \tag{8}$$

That is,

$$\log p_X(x) \approx \log p_Z(z) + \int_0^1 \operatorname{Tr} J_{\phi_\theta}(x(t)) dt \tag{9}$$

Remark

Continuous-time normalizing flows are desirable because the constraints that need to be enforced on ϕ_θ are relatively mild: ϕ_θ only needs to be high order differentiable and Lipschitz continuous, with a possibly large Lipschitz constant.

E(n) Equivariant Normalizing flows

Let (x, h) be a datapoint from the distribution $p_X(x, h)$ where x is the coordinate of a node and h is the feature of a node.
Consider the functions $x(t), h(t)$ on time s.t. $x(0) = x, h(0) = h$
The algorithm of Equivariant Normalizing flows is as follows [Satorras et al., 2021a]:

- Fix a latent distribution $p_Z(z_x, z_h)$ where z_x, z_h are the latent representations of position and feature, respectively.
- Approximate the first derivative by a $L-$layers EGNN with initial parameter θ_0

$$\frac{d}{dt}x(t), \frac{d}{dt}h(t) \approx x^L(t) - x(t), h^L(t)$$

  where

$$x^L(t), h^L(t) = EGNN[x(t), h(t)]$$

- Solve the ODE with the initial condition $x(0) = x, h(0) = h$

E(n) Equivariant Normalizing flows

- Compute $z_x = x(1), z_h = h(1)$
- Approximate $\log p_X(x, h)$ by

$$\log p_X(x, h) \approx \log p_Z(z_x, z_h) + \int_0^1 \mathrm{Tr}\, J_{\phi_\theta}(x(t), h(t))dt \tag{10}$$

  where the trace of J_{ϕ_θ} has been approximated with the Hutchinson's trace estimator
- if there is a set of datapoints $\{(x_i, h_i)\}_{i=1}^m$, the objective is set up to maximize the log-likelihood $\sum_{i=1}^{m} \log p_X(x_i, h_i)$

E(n) Equivariant Normalizing flows

Some remarks

- switch from discrete-time dynamics to continuous-time dynamics reduces the computation cost from $\mathcal{O}(D^3)$ to $\mathcal{O}(D)$ (refer to [Grathwohl et al., 2018])
- a modification of EGNN has been done to make it stable when applied in ODE

$$\boldsymbol{x}_i^{l+1} = \boldsymbol{x}_i^l + \sum_{j \neq i} \frac{(\boldsymbol{x}_i^l - \boldsymbol{x}_j^l)}{||\boldsymbol{x}_i^l - \boldsymbol{x}_j^l|| + C} \phi_x(\boldsymbol{m}_{ij})$$

  where they set C to be 1 (to ensure the differentiability)
- the main contribution in [Satorras et al., 2021a] is preserving the equivariance while using an EGNN in continuous normalizing flows

Reference

Chen, R. T., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018).
Neural ordinary differential equations.
Advances in neural information processing systems, 31.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017).
Neural message passing for quantum chemistry.
In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 1263–1272. JMLR.org.

Grathwohl, W., Chen, R. T., Bettencourt, J., Sutskever, I., and Duvenaud, D. (2018).
Ffjord: Free-form continuous dynamics for scalable reversible generative models.
arXiv preprint arXiv:1810.01367.

Kobyzev, I., Prince, S. J., and Brubaker, M. A. (2020).
Normalizing flows: An introduction and review of current methods.
IEEE transactions on pattern analysis and machine intelligence, 43(11):3964–3979.

Reference

Satorras, V. G., Hoogeboom, E., Fuchs, F. B., Posner, I., and Welling, M. (2021a).
E (n) equivariant normalizing flows.
arXiv preprint arXiv:2105.09016.

Satorras, V. G., Hoogeboom, E., and Welling, M. (2021b).
E (n) equivariant graph neural networks.
In *International conference on machine learning*, pages 9323–9332. PMLR.