Message-Passing Neural Networks (MPNNs) and their applications

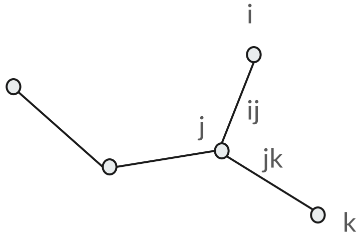
- Definitions
- Examples
- Architectures
- 3D Generation

Olivier Peltre
CRIL, CNRS

Datacraft
14/04/24

Definitions

Graphs



A graph $G = (G_0, G_1)$ may be defined as

- $G_0 = \{i, j, k...\}$ set of vertices
- $G_1 \rightarrow G_0 \times G_0$ set of edges, e.g. $G_1 = \{(i, j), (j, k), ... \}$ subset of $G_0 \times G_0$.

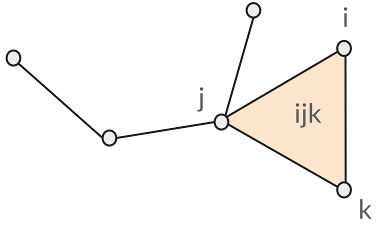
Features $x \in F(G)$ usually also map the elements of G to node- and edge-feature spaces:

$G_0 \rightarrow F_0$ $G_1 \rightarrow F_1$

$(x_i) \in F_0$ $(x_{ij}) \in F_1$

Definitions

Hypergraphs



In general, d-graphs $G = (G_0, G_1, \ldots, G_{d-1})$ also include higher-order relations:

- $G_0 = \{i, j, k\ldots\}$ set of vertices
- $G_1 \rightarrow G_0 \times G_0$ set of edges,
- …
- $\mathbf{G_d \rightarrow G_0 \times \ldots \times G_0}$ set of **d-faces** or hyperedges

For instance, including **2-faces** = triangles in a 3D structure is necessary to compute **angles.**

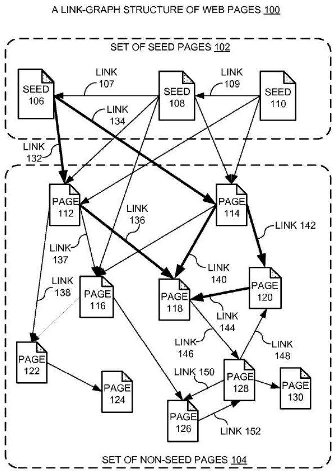
… **3-faces** = tetrahedra in a 3D chain … **torsion angles**

…

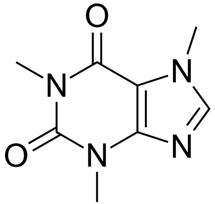
Examples

Examples

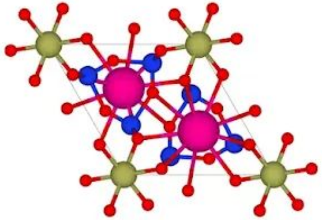
- Web : Pages and links,
- Networking : Hosts and routing,
- Telecoms : Bits and checksums,
- Advertising : Users and content,
- ...
- Chemistry : Atoms in organic molecules
- Material Sciences : Atoms in periodic crystals
- Biology : Amino Acid (AA) sequences in proteins
- ...



SEO

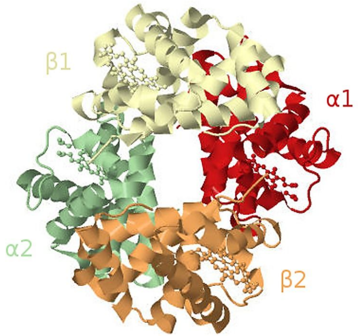


Caffeine



$Rb_2HfSi_3O_9$

Deepmind crystal



Hemoglobin

Example Tasks

Classification

$F(G) \rightarrow K = \{0, 1, \ldots\}$

Toxicity, binding …

Clustering

Regression

$F(G) \rightarrow R$ or R^d

Energy, band gaps, chemical properties…

Likelihood estimation (e.g. critic)

Feature prediction

$G \rightarrow F(G)$

Recommendation systems

Error correction

AA sequence to 3D structure

Generation

$N(0, 1) \rightarrow F(G)$

Drug discovery

Material discovery

Architectures

Graph Convolutional Networks (GCNs)

Graph gradient / **differential**

$d : F(G_0) \to F(G_1)$

$d(x)_{ij} = w_{ij} (x_i - x_j)$

+

Graph divergence / **codifferential**

$d^* : F(G_1) \to F(G_0)$

$d^*(v)_j = \Sigma_{i>j} v_{ij} - \Sigma_{j>i} v_{ji}$

→

Graph Laplacian and its polynomials

$L = d^*d : F(G_0) \to F(G_0)$

$L(x)_j = x_j$ - average over neighbours $(x)_j$

$P(L) = \Sigma_k p_k L^k$: local filter of range k

$P(L)(x) = \Sigma_k p_k L^k (x)$

Architectures

Message-Passing Neural Networks (MPNNs)

Message function

$M : F(G_0) \longrightarrow F(G_1)$

$M_{ij} = f_\theta(x_i, x_j)$

+

Update function

$U : F(G_1) \longrightarrow F(G_0)$

$U(M)_j = \Sigma_{i > j} M_{ij}$

→

Message-Passing layers and their iterations

$UM : F(G_0) \rightarrow F(G_0)$

Architectures

Message-Passing Neural Networks (MPNNs)

Message function

$M : F(G) \to F(G)$

$M_{ij} = f_\theta(x_i, x_j, x_{ij})$

+

Update function

$U : F(G) \to F(G)$

$U(x, M)_j = x_j + \Sigma_{i > j} M_{ij}$

Note that many different algorithms fall under the umbrella of MPNNs

Graph Convolution is a particular case of message-passing

Shouldn’t edge features be updated as well?

Architectures

Message-Passing on Hypergraphs

Nerve: ordered chains

$N_k G = \{a_0 > \dots > a_k\}$

a_0 contains a_1 contains …

$N_1 G = \{a > b\}$

Messages:

$M : F(G) \to F(N_1 G)$

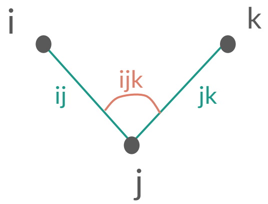
$M_{a > b} = f_\theta(x_a, x_b)$

Update function

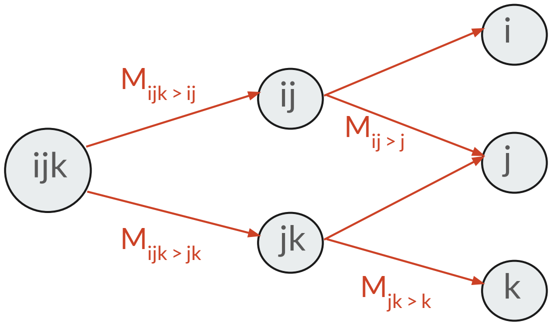
$U : F(N_1 G) \to F(G)$

$U(M)_b = \Sigma_{a > b} M_{a > b} - \Sigma_{b > c} M_{b > c}$

hypergraph



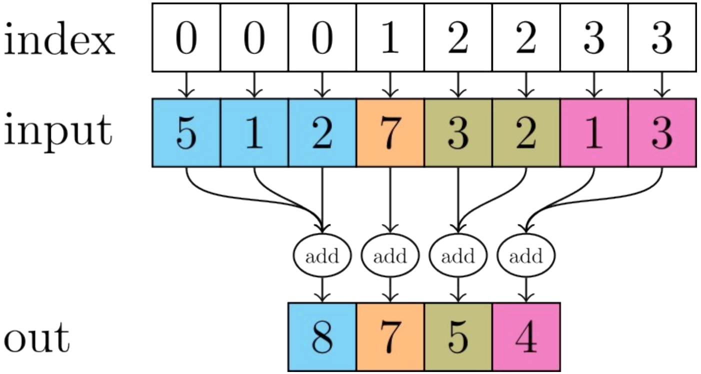
→



partial order

Architectures

Implementation Details



```
from torch_scatter import scatter

out = scatter(src, idx, dim=0)

>>> for x, i in zip(src, idx):
        out[i] += x
```

N.B: This is equivalent to a sparse matrix-vector product: sparse CSR matrices are optimal for this task.

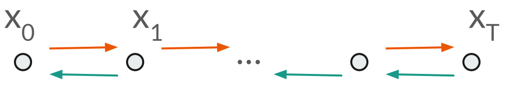
However, handling sparse (node) / dense (feature) dimensions and reshapes quickly become cumbersome.

Diffusion Models (aka Annealed Importance Sampling)

Forward process:

$x_{t+1} \sim P(x_{t+1} \mid x_t)$

$x_{oo} \sim N(0, 1)$



data samples

Gaussian noise

Backward process:

$x_{t-1} \sim g_\theta(x_{t-1} \mid x_t, t)$

...

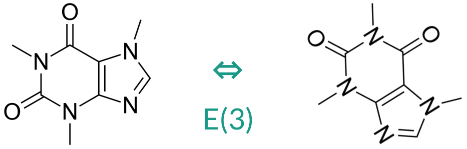
$x_0 \sim g_\theta(x_0 \mid x_1, t)$

Stochastic Differential Equations (SDEs) in the continuous limit:

$$\longrightarrow \qquad dx_t = -\frac{1}{2}\beta(t)x_t dt + \sqrt{\beta(t)}dW_t$$

$$\longleftarrow \qquad x_{t-dt} = x_t + \frac{1}{2}\beta(t)x_t dt + \beta(t) f_\theta(x_t, t)dt + \sqrt{\beta(t)}dW_t$$

3D Generation



Equivariance?

SO(3) and E(3) act on R^{3xN}

> For regression tasks, predictions should not depend on molecule presentation.

> For generation tasks, position updates should be E(3) equivariant.

Instead of working with atomic positions directly, we should work with local Euclid invariants :

0: none
1: lengths
2: angles, areas
3: torsion angles, volumes
...

$$\xrightarrow{\nabla}$$

0: none
1: edge vectors
2: rotations
3: torsions
...

see Klipfel, P, et al. 2023: Equivariant Message Passing Neural Network for Crystal Material Discovery

3D Generation

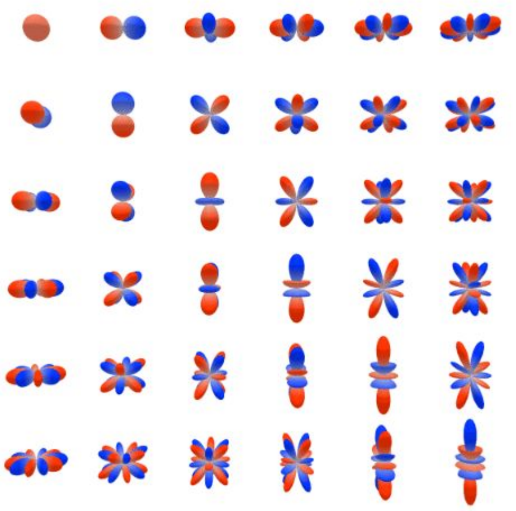
Equivariance?

Other approaches rely on **spherical harmonics** as angular activation functions:

$$r \to Y_{lm}(r) \quad \text{for } l < l_{max}, \ m = -l, \ \dots, +l$$

Steeper learning curve!
See:

- e3nn (torch + jax)
- e3x (jax / by Google)



Spherical Harmonics
(from e3nn)

Spherical harmonics $\mathbf{Y_{lm}}$ span the space $\mathbf{H_l}$ of degree-l **harmonic polynomials,** i.e. homogeneous polynomials with zero laplacian.

Warning: choosing a privileged basis of quantum numbers on $\mathbf{H_l}$ may break the equivariance…

References

Extensive review on 3D structures:
- A Hitchhiker’s Guide to Geometric GNNs for 3D Atomic Systems (Duval 2023)

Higher order MPNNs:
- MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields (Batatia et al 2022)

Deepmind nature paper :
- Scaling deep learning for materials discovery (Merchant et al 2023)

General review on GNNs:
- A Gentle Introduction to Graph Neural Networks

For more references and these slides: → github.com/opeltre/gnn

Technical case

→ https://github.com/opeltre/gnn

QM7 and QM9 datasets may be found here

- http://quantum-machine.org/datasets/

Respectively consist of ~7k and ~130k organic molecules with:

- atomic positions
- atomization energy (QM7) and more (QM9) obtained by quantum simulations

The small QM7 dataset may be used for simple regression tasks on atomization energy.

Bigger datasets such as the QM9 (~130k molecules) are probably more suited for generation tasks.