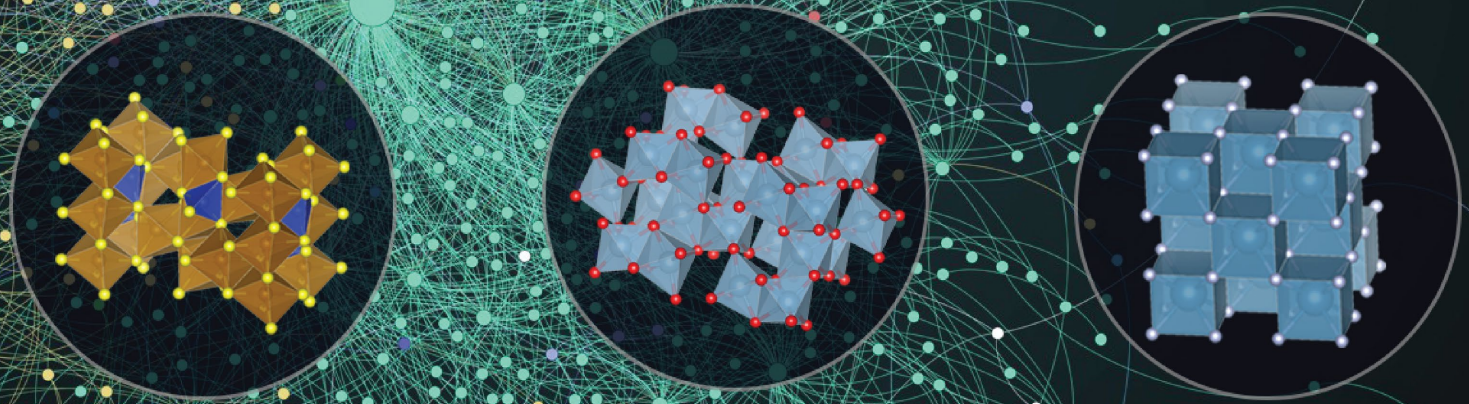
Crystal Structure graphs



Outline

Components, directionality, tasks
Molecules and crystals as graphs
graph connectivity and centralities
Matrix representation
message passing and attention
Architecture design options

What should we look for in future graph network architectures?

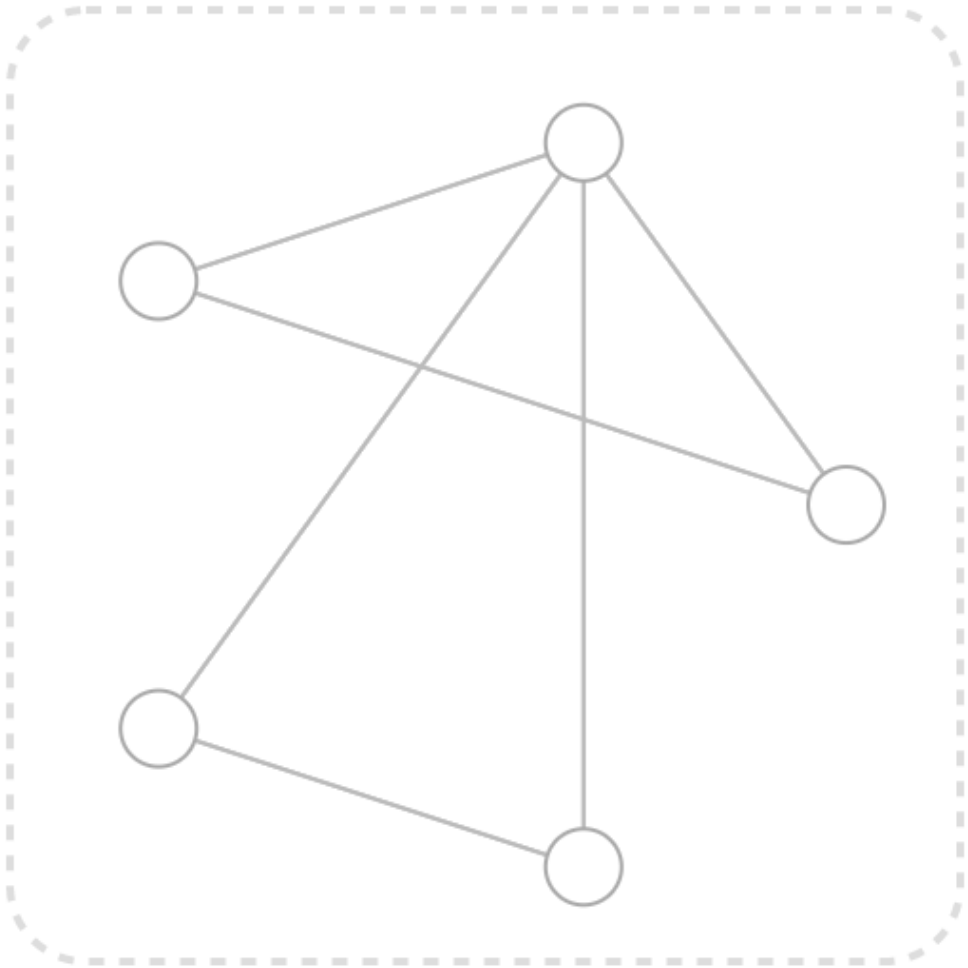
Graphs are a convenient and powerful way to represent information

Nodes: people
Edges: relationships

Nodes: email users
Edges: email message

Nodes: papers
Edges: citations

Graphs contain key components



V Vertex (or node) attributes

e.g., node identity, number of neighbors

E Edge (or link) attributes and directions

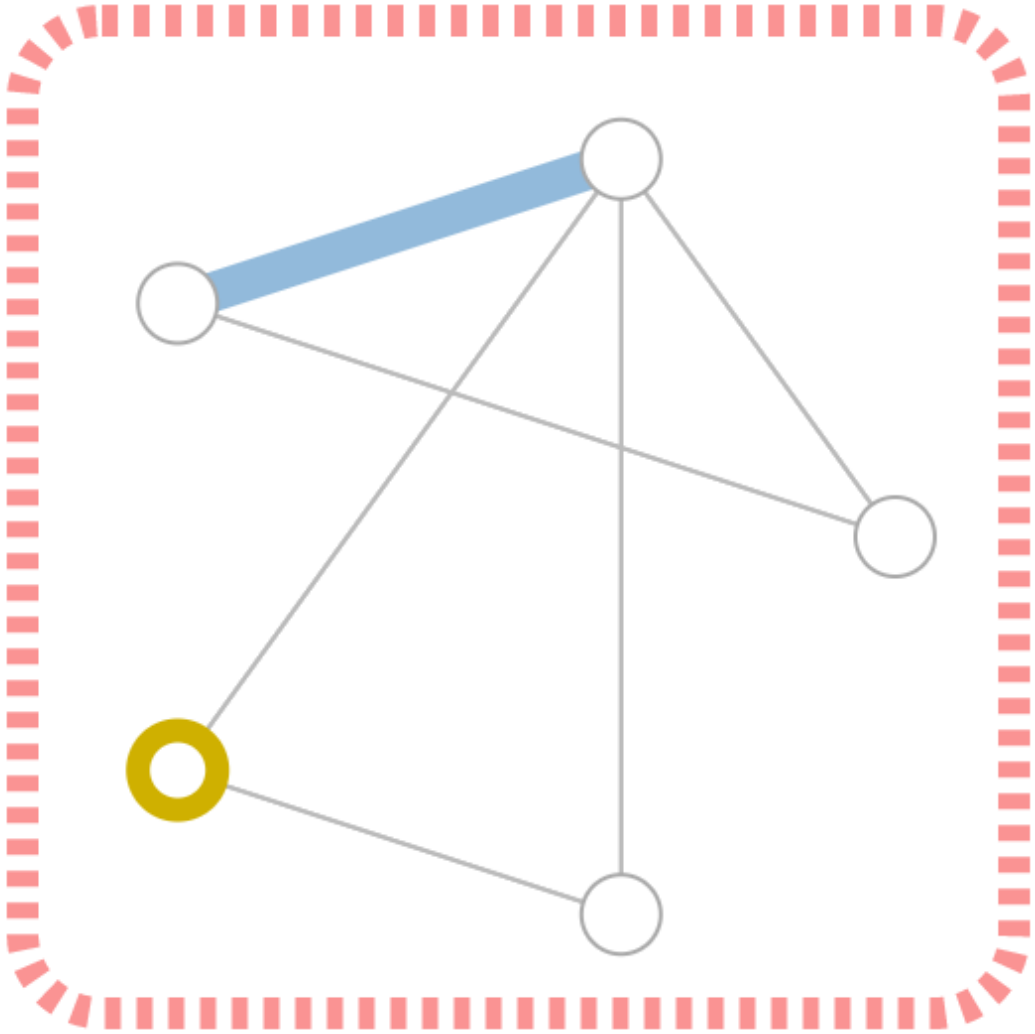
e.g., edge identity, edge weight

U Global (or master node) attributes

e.g., number of nodes, longest path

Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

We can encode information at all of these components



Vertex (or node) embedding

Edge (or link) attributes and embedding

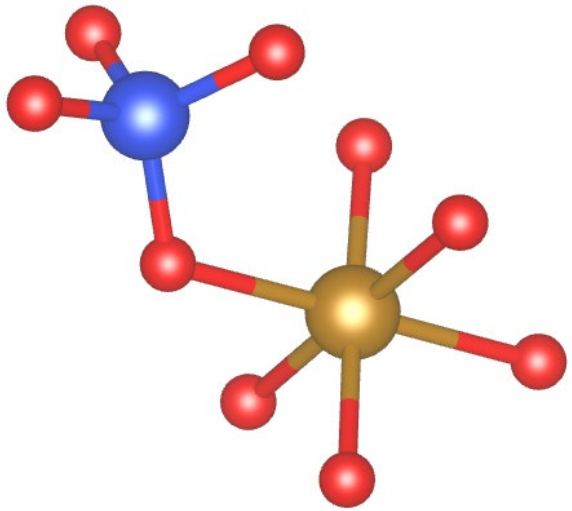
Global (or master node) embedding

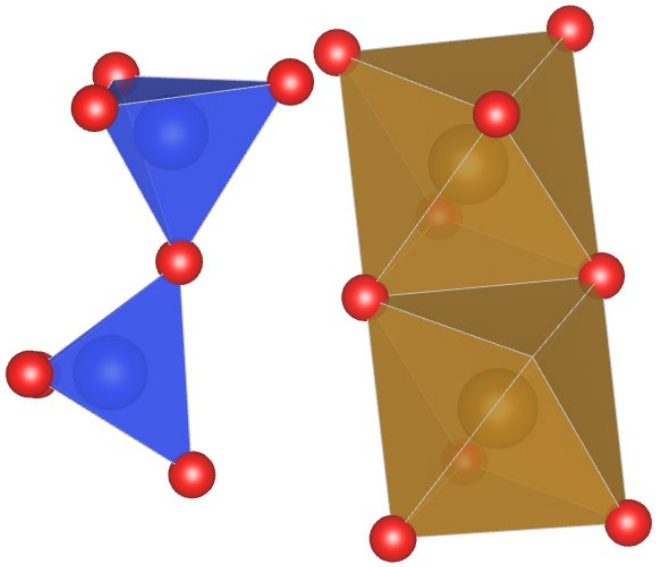
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Another way of encoding information is in **direction** of edges

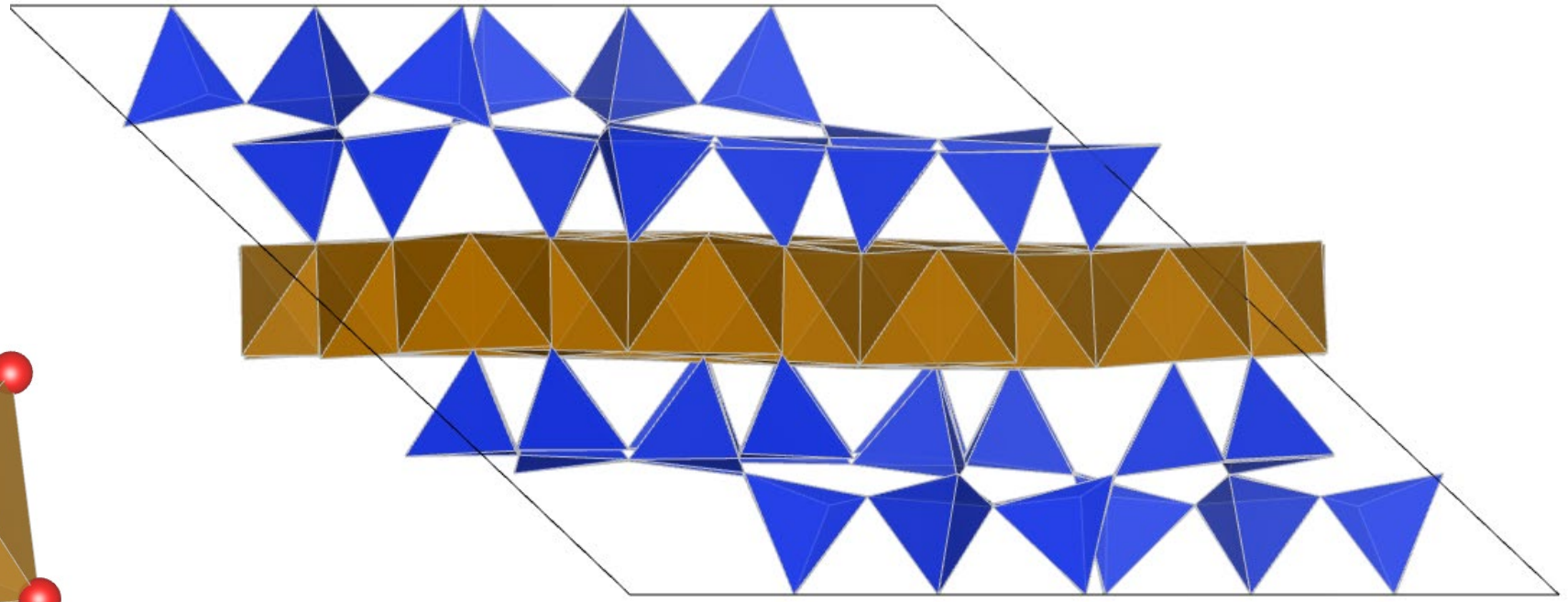
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Molecules and crystals naturally lend themselves to graph networks

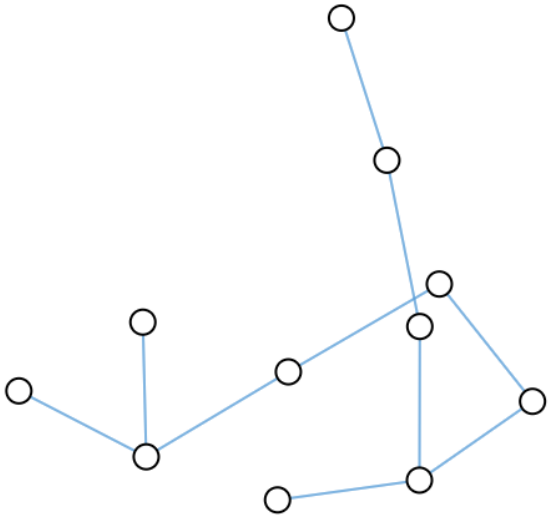


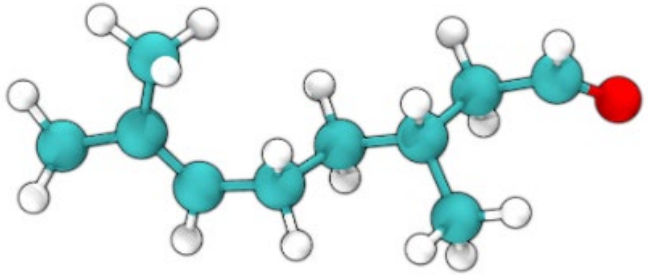


Nodes: atoms
Edges: bonds

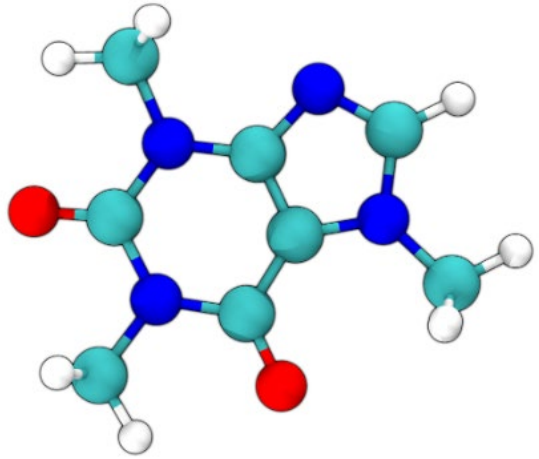


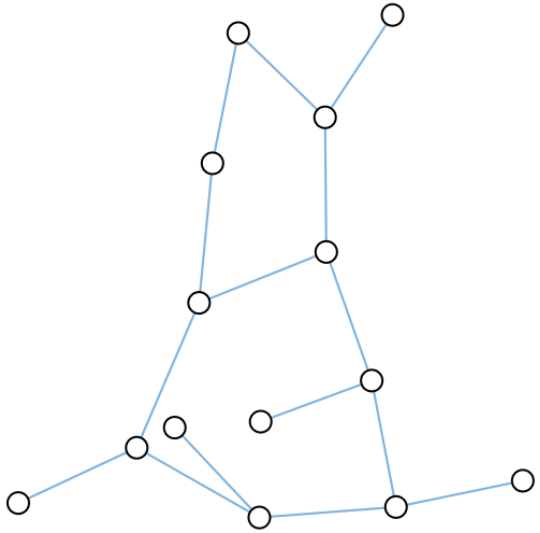
Molecules look a bit different as graphs.





Why graphs belong to which?





Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

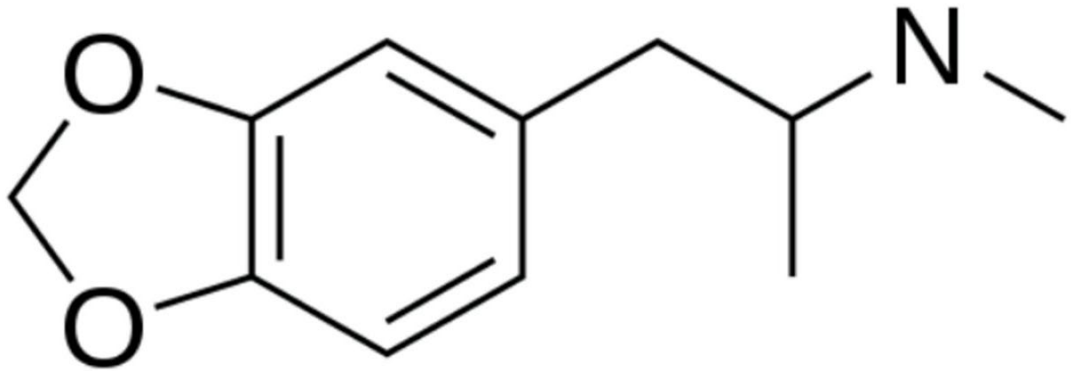
Strings are a common alternative representation

String-based Representation

SMILES: `CC(CC1=CC2=C(C=C1)OCO2)NC`

DeepSMILES: `CCCC=CC=CC=C6))OCO5)))))))NC`

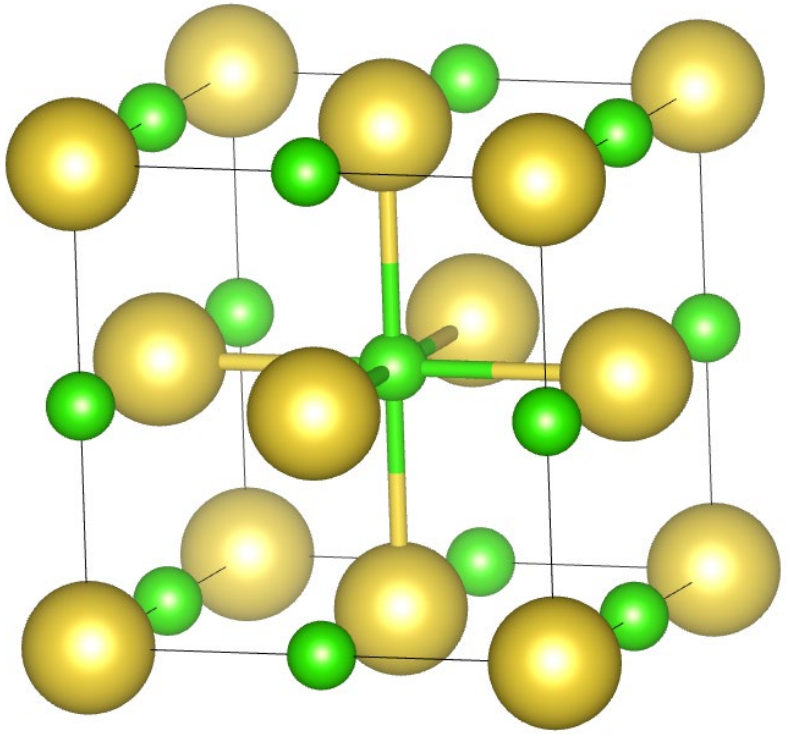
SELFIES: `[C][C][B1_1][P][C][C][=C][C][=C][B1_1][B1_1][C][=C][R1][B1_2][O][C][O][R1][B1_3][N][C]`



https://twitter.com/MarioKrenn6240/status/1348814674415001600?s=20

Graphs for periodic lattices are more challenging!

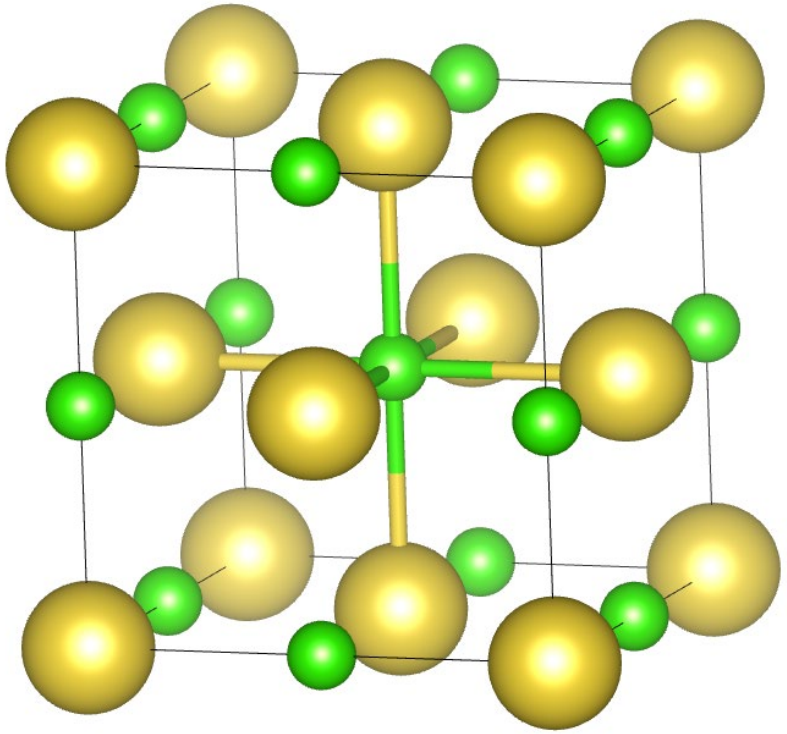
$2\text{Å} < x < 3\text{Å}$



6 neighbors

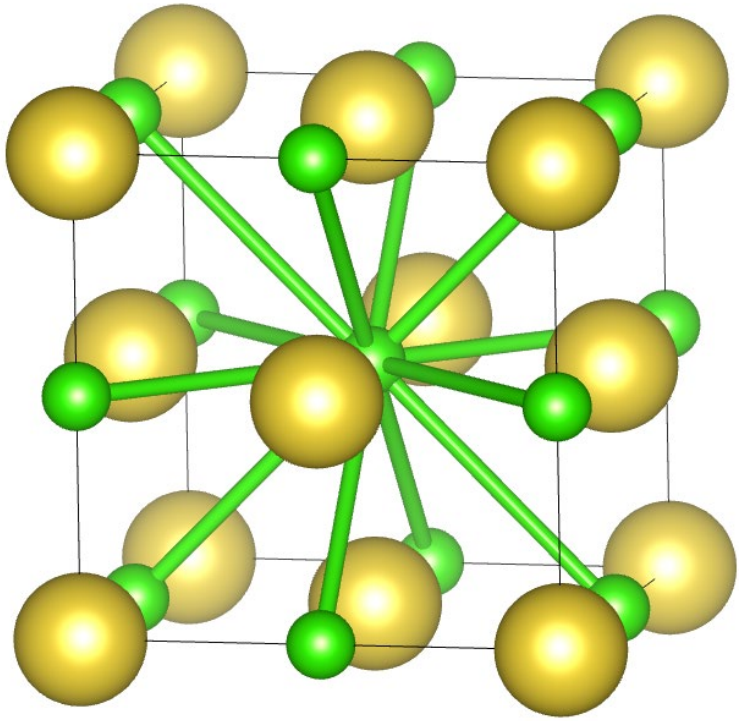
Graphs for periodic lattices are more challenging!

$2\text{Å} < x < 3\text{Å}$



6 neighbors

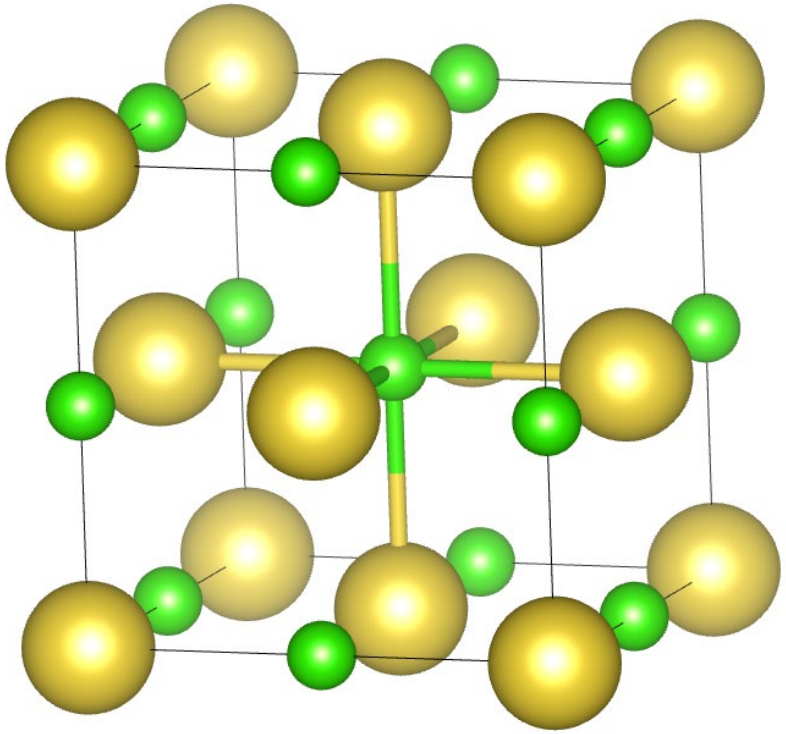
$3\text{Å} < x < 4\text{Å}$



12 neighbors

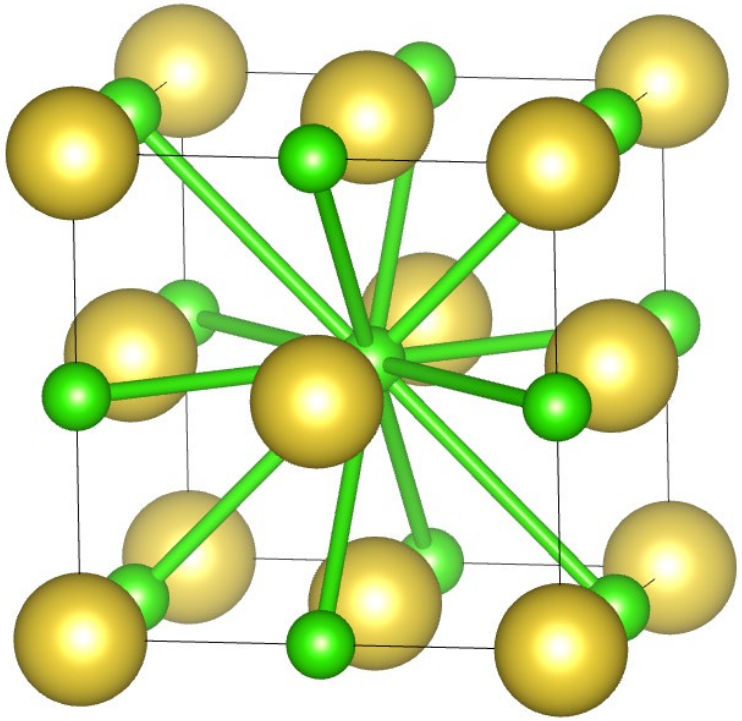
Graphs for periodic lattices are more challenging!

$2\text{Å} < x < 3\text{Å}$



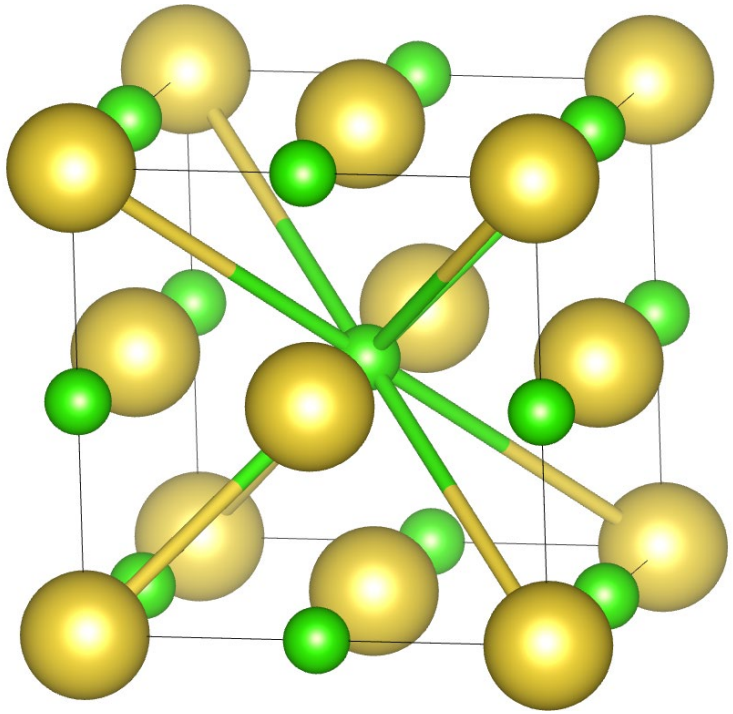
6 neighbors

$3\text{Å} < x < 4\text{Å}$



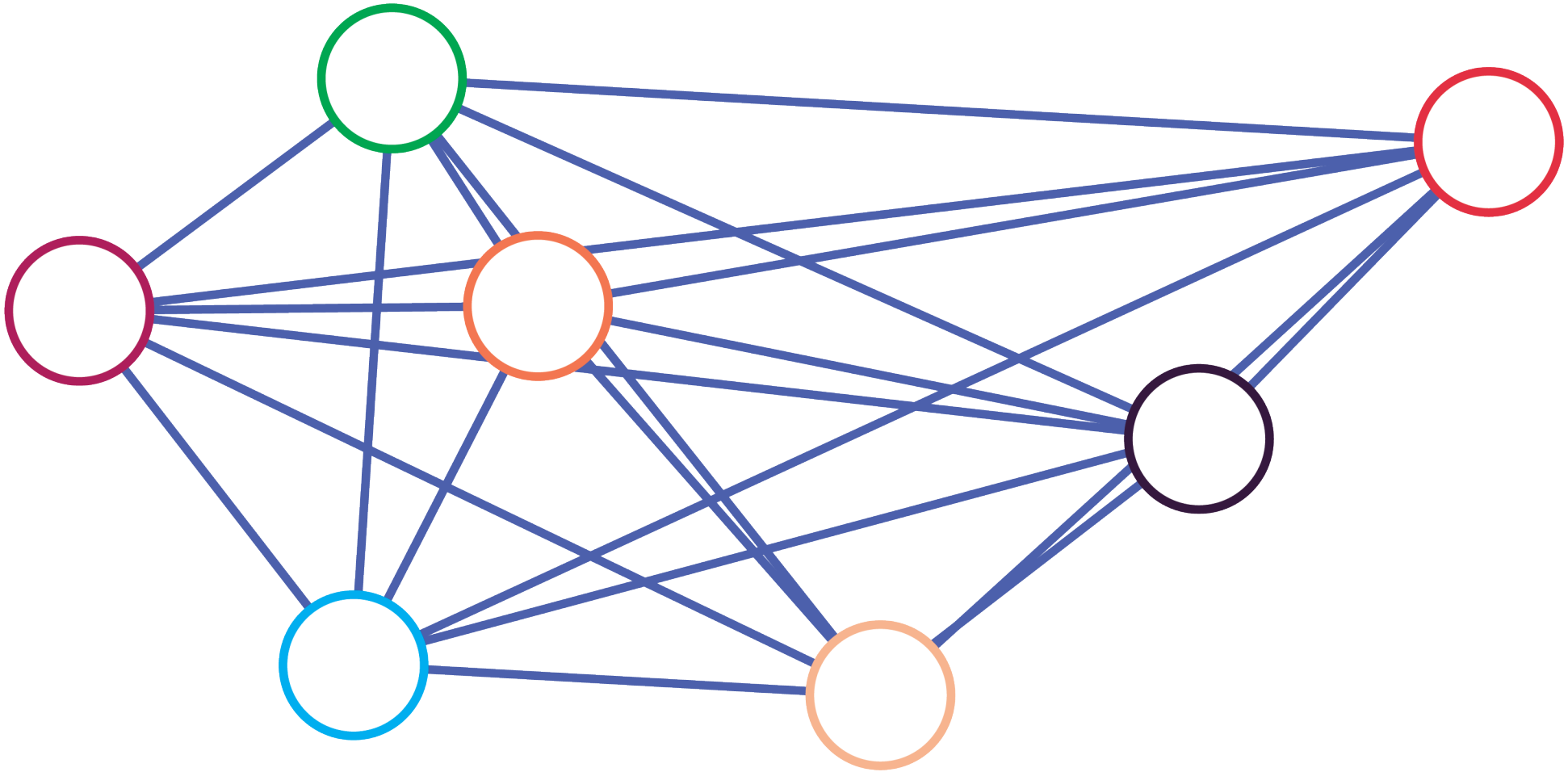
12 neighbors

$4\text{Å} < x < 5\text{Å}$

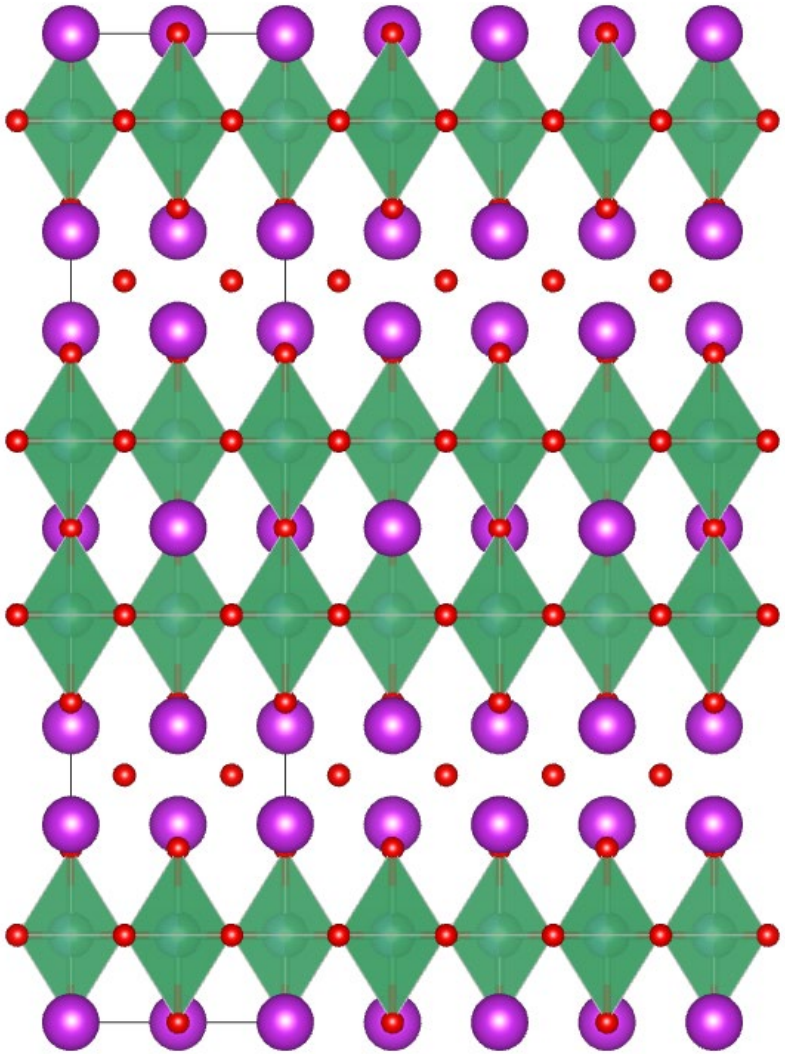


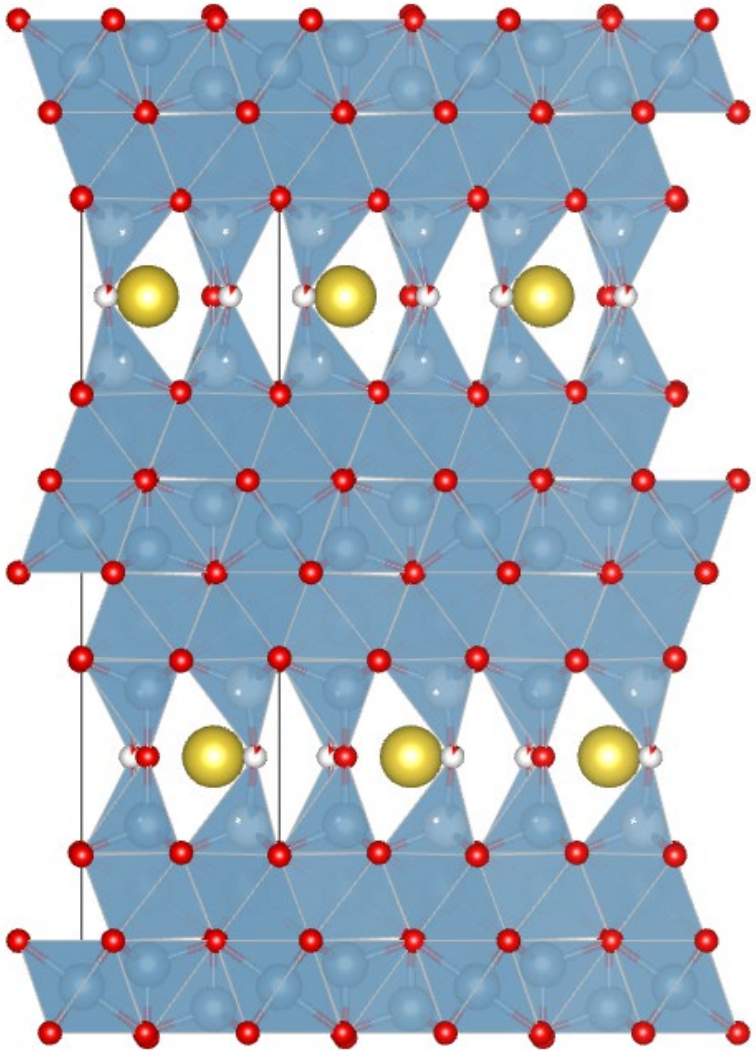
8 neighbors

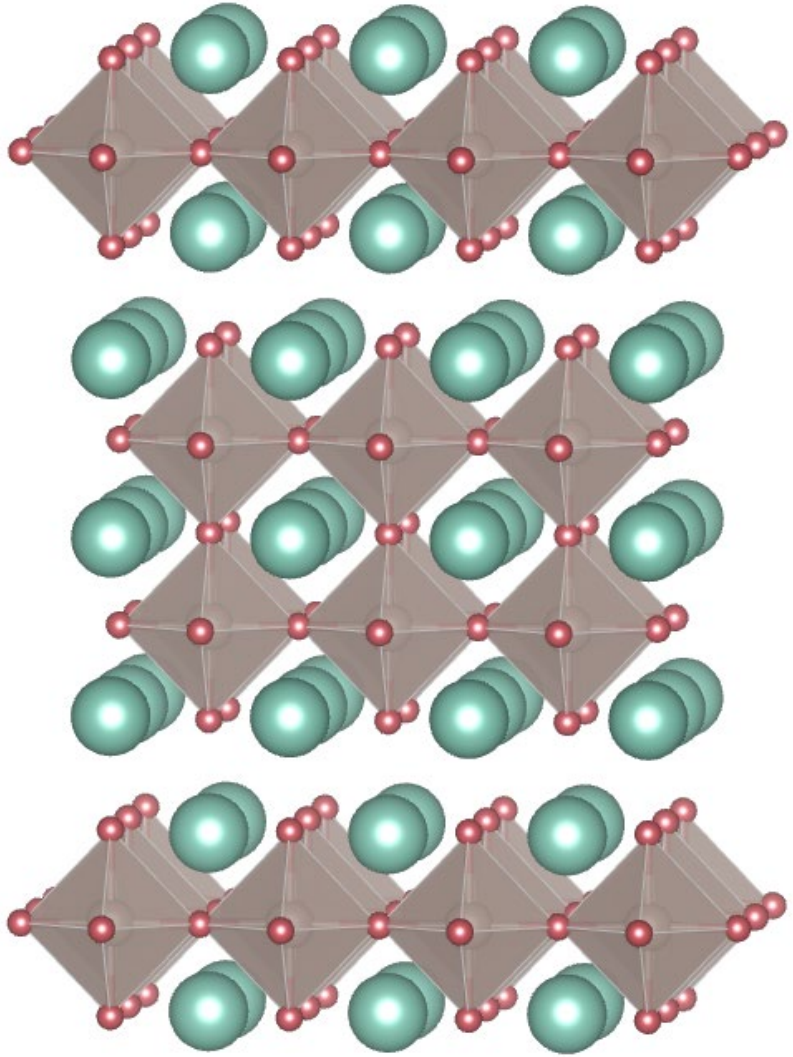
Current graph networks are fully connected



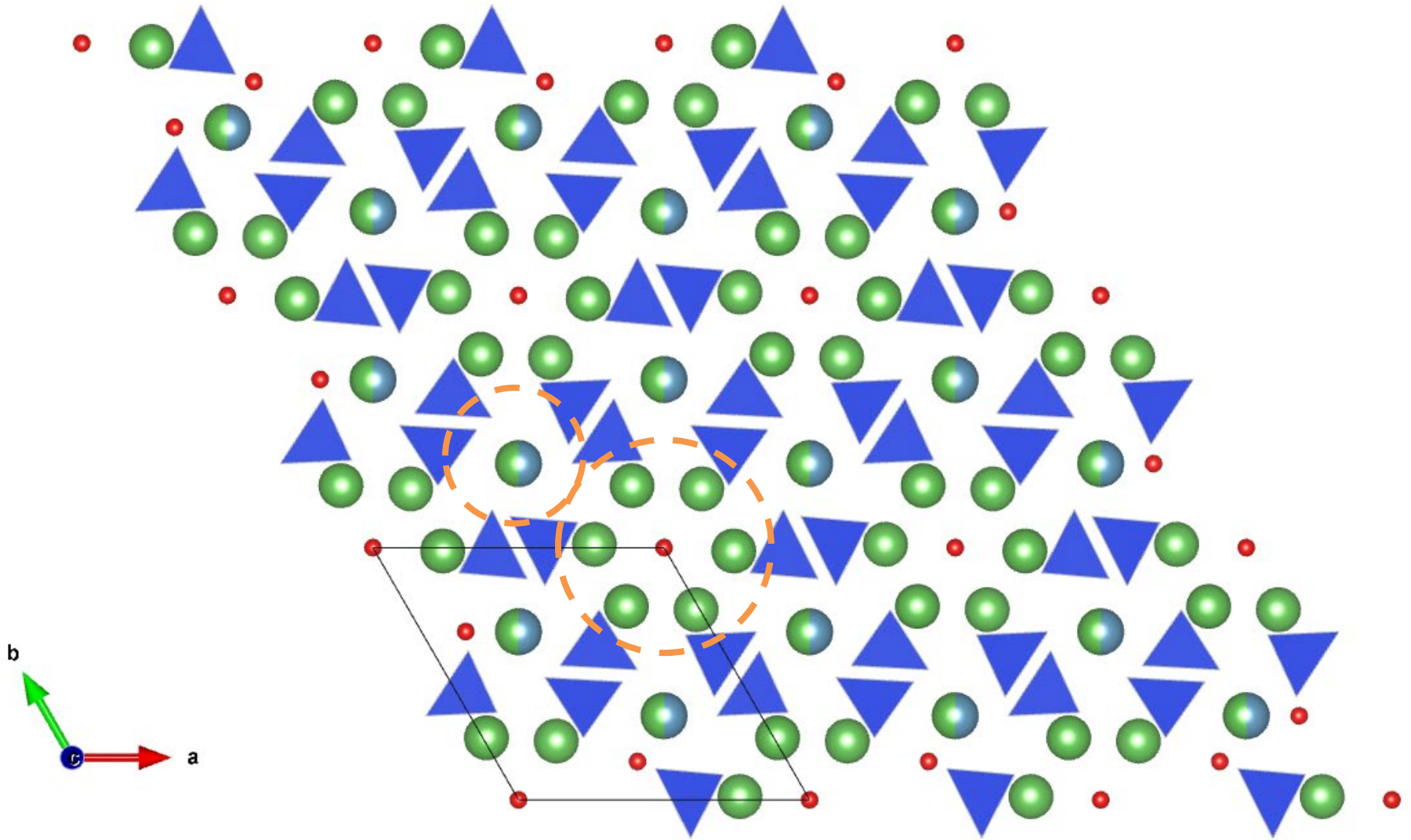
Many properties rely on specific interactions between atoms



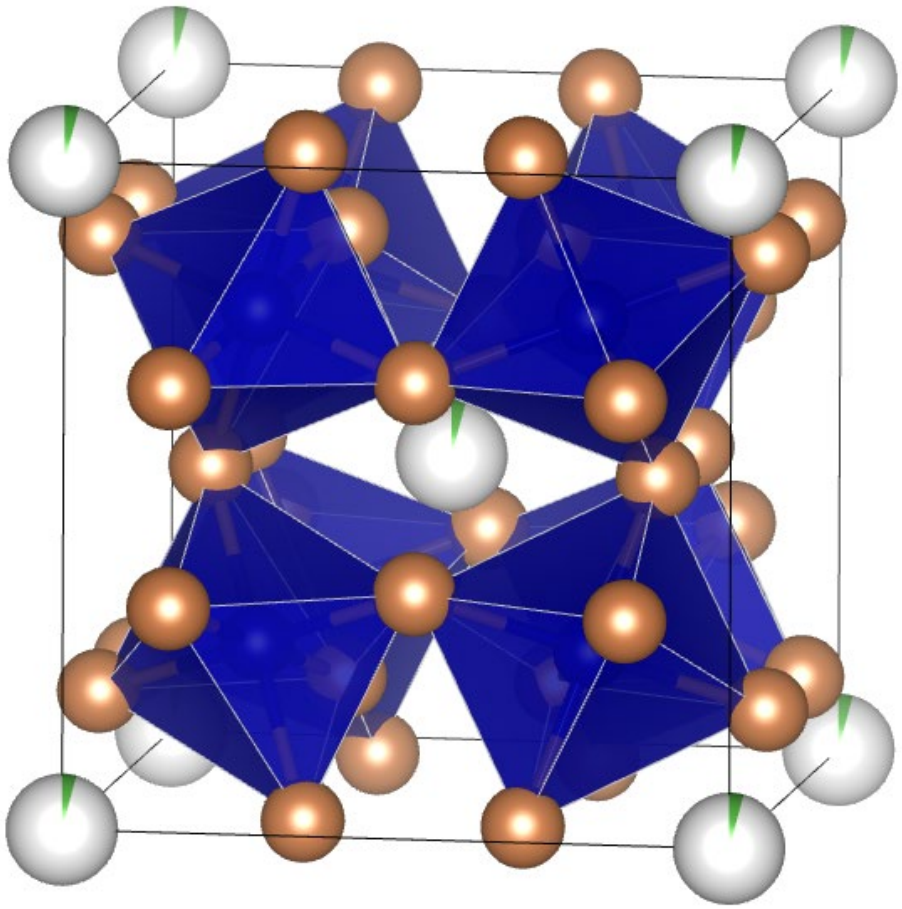


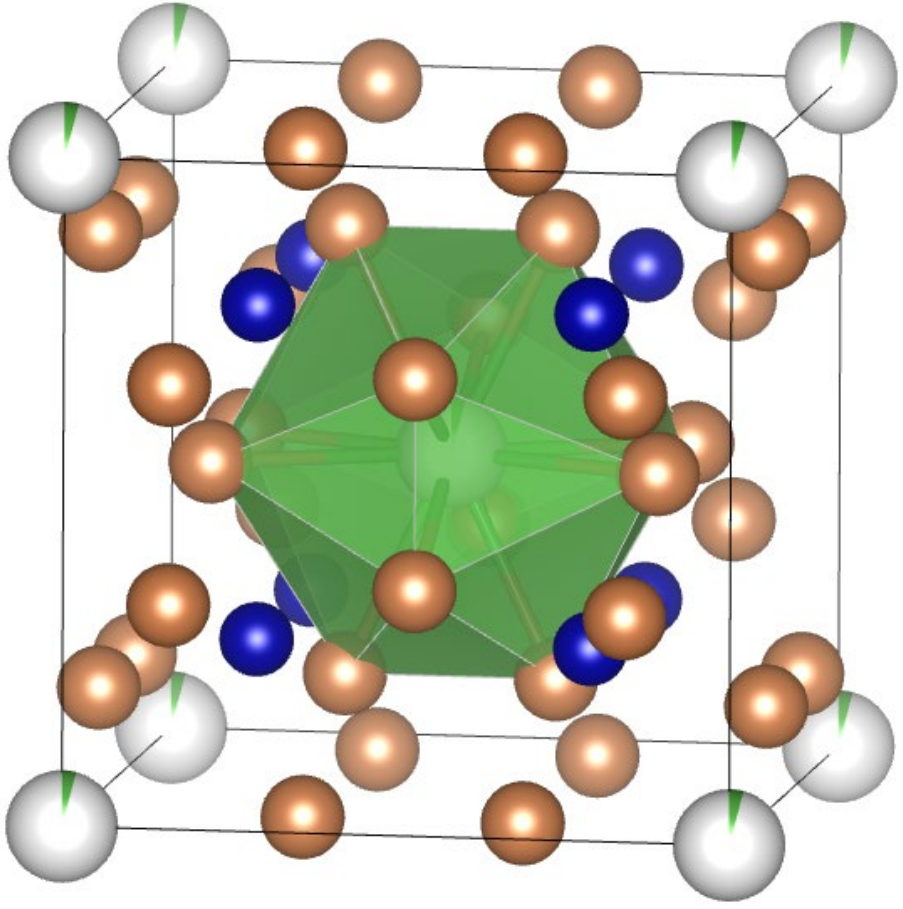


Many properties rely on specific interactions between atoms

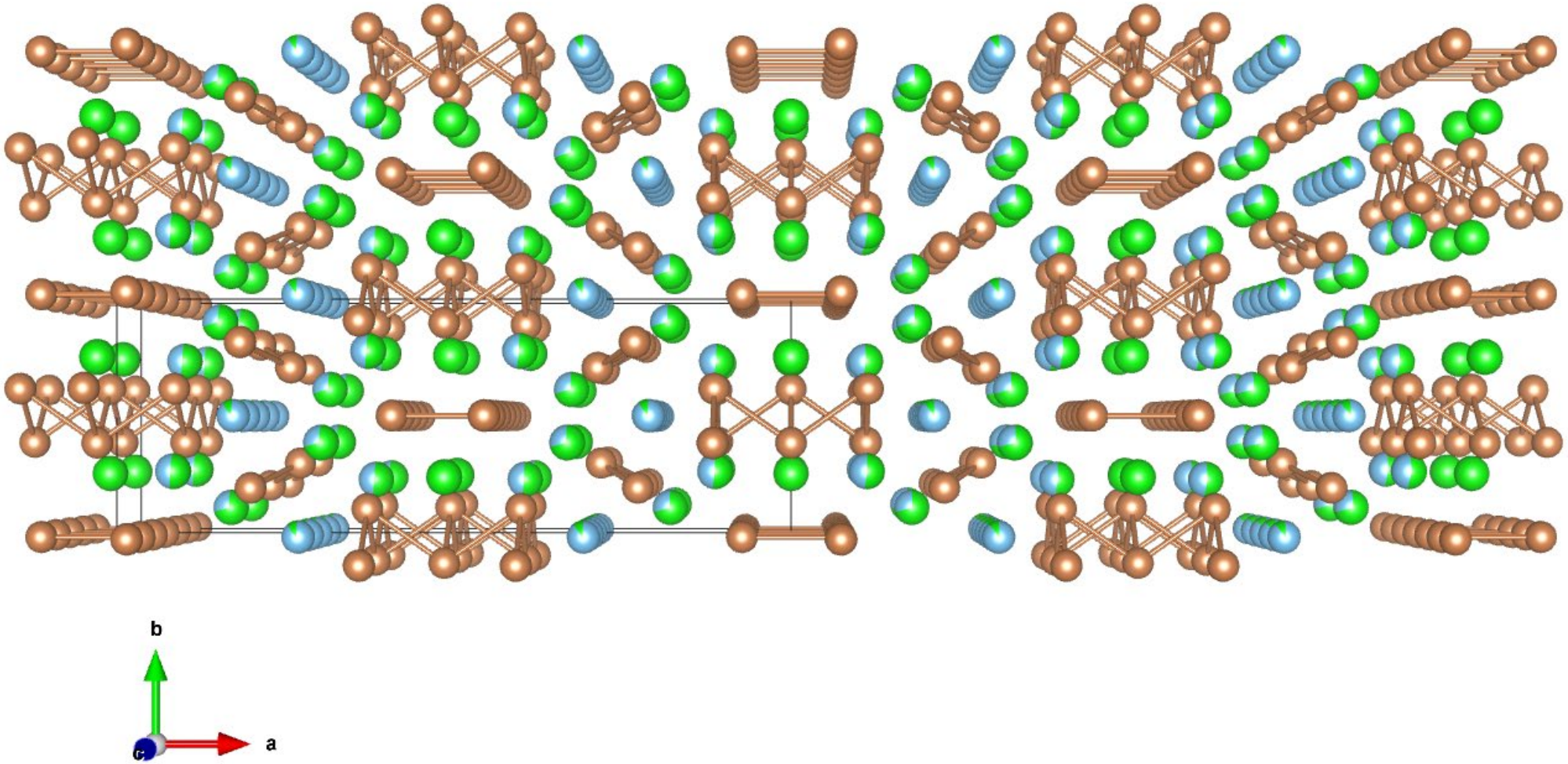


Many properties rely on specific interactions between atoms

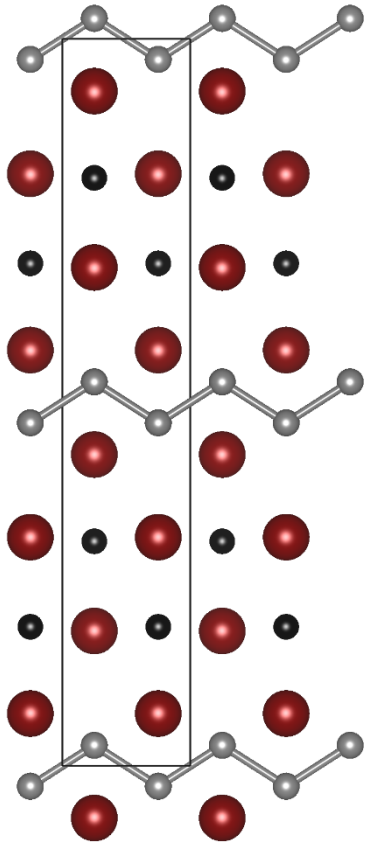


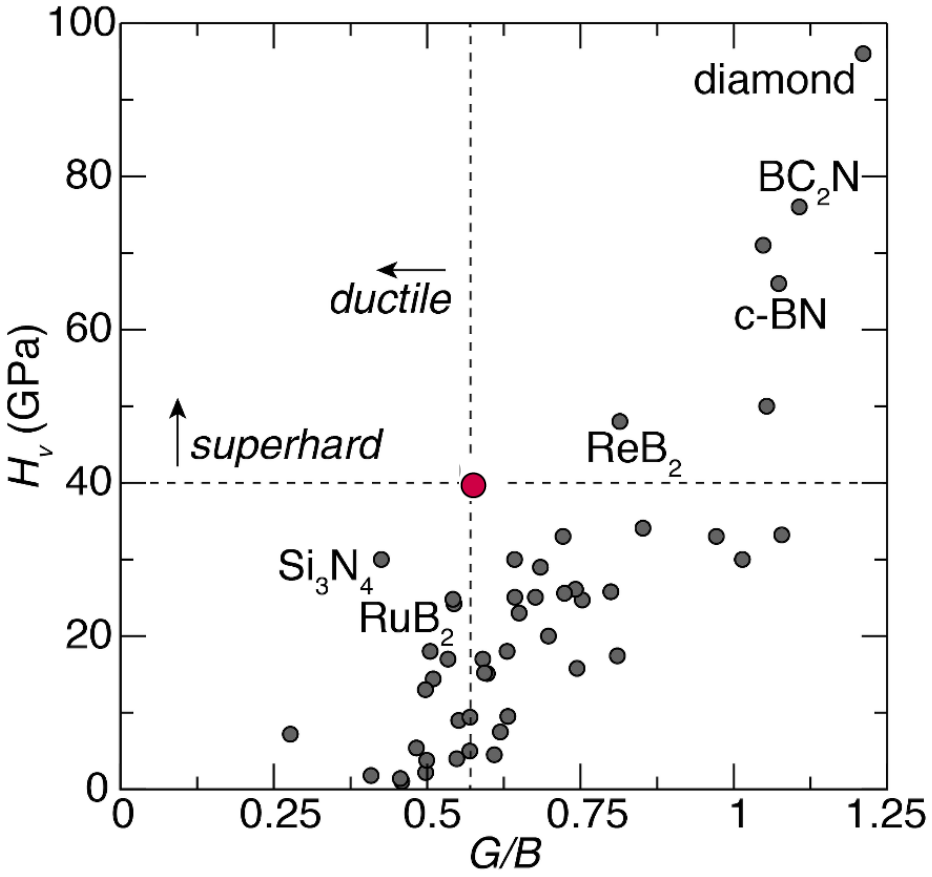


Many properties rely on specific interactions between atoms



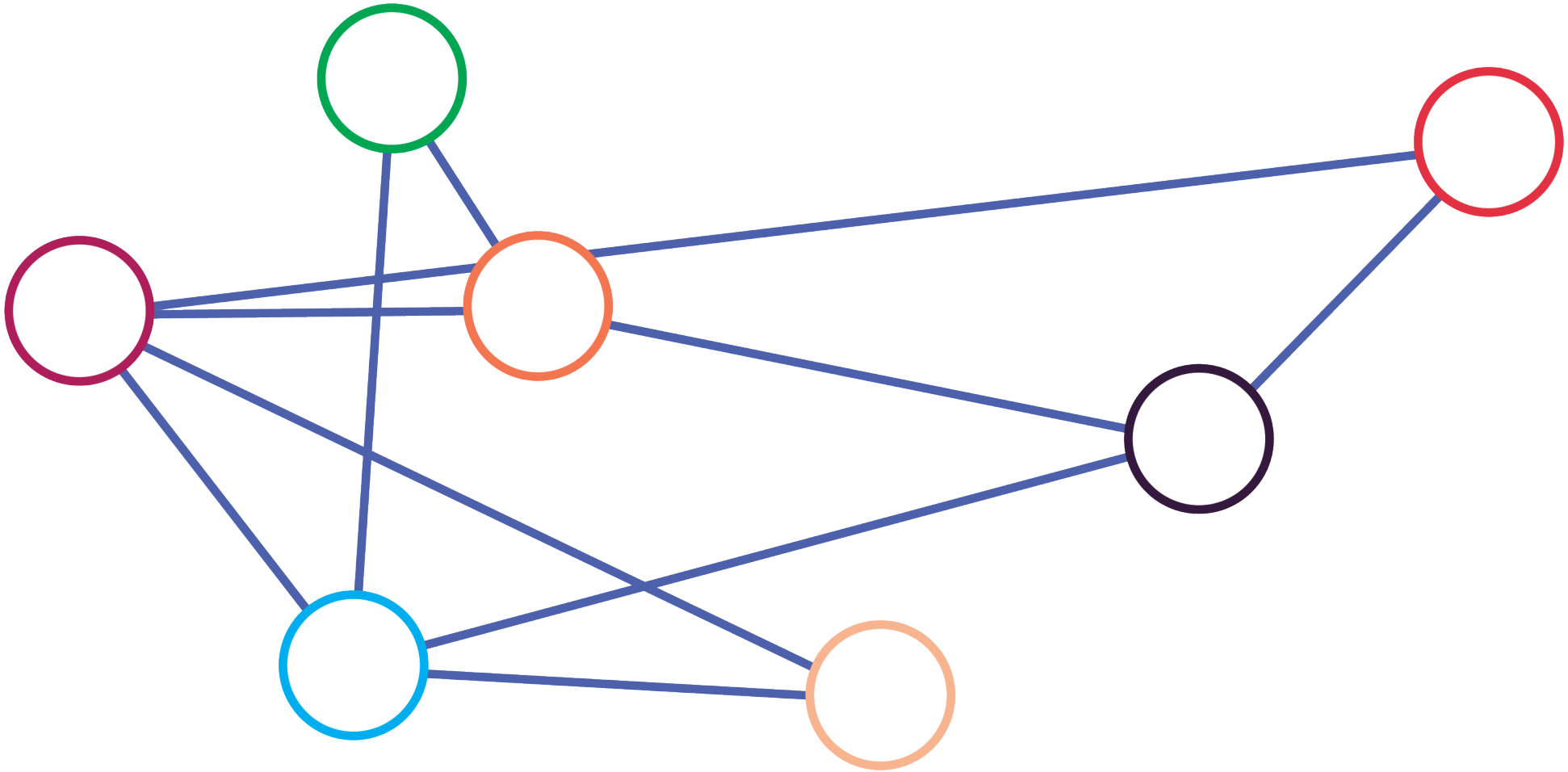
Many properties rely on specific interactions between atoms



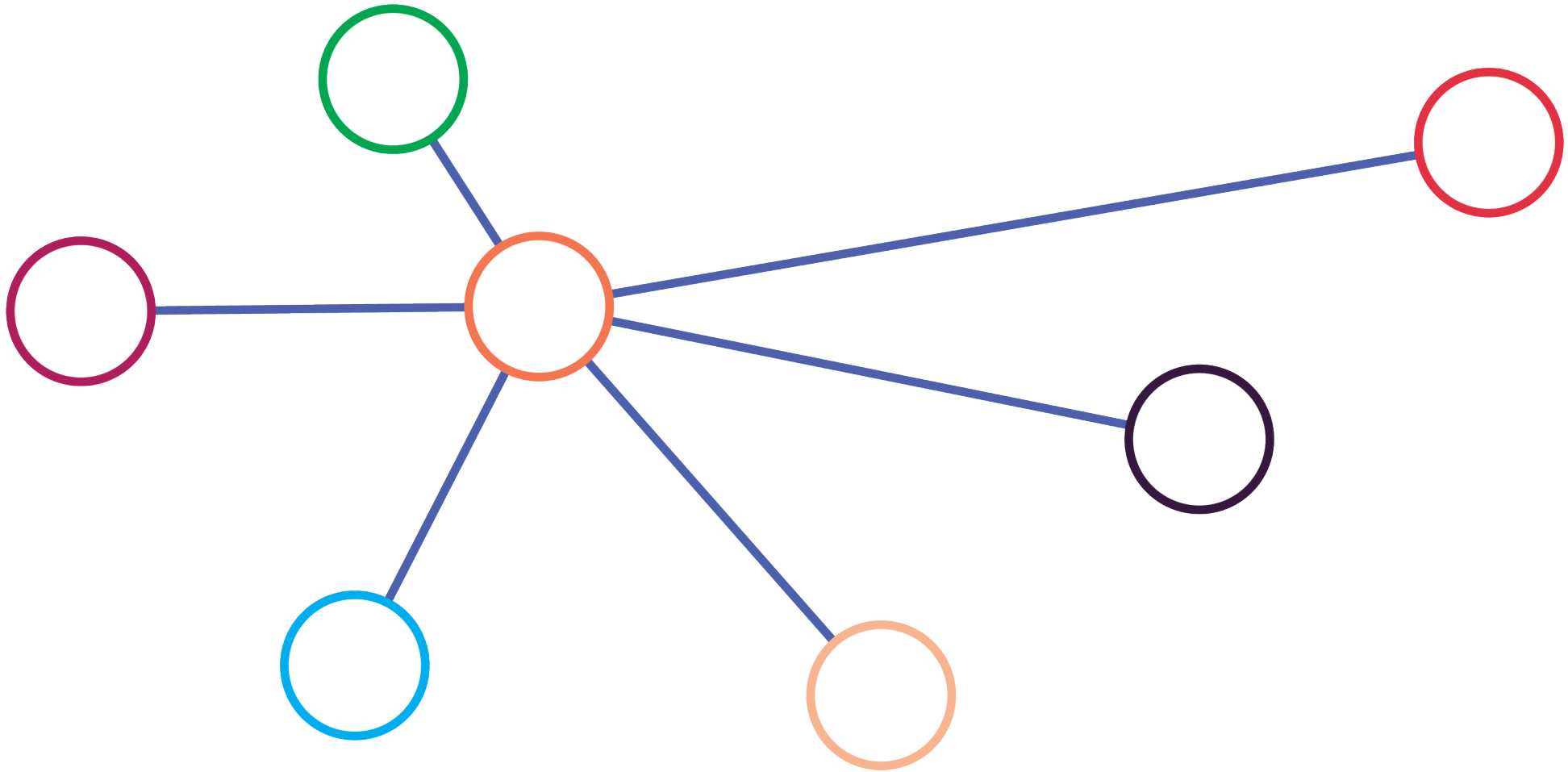


Mansouri et al. (2019) Chem. Mater.
Mansouri et al. (2018) JACS, Parry et al. (2019) J Mat Chem A

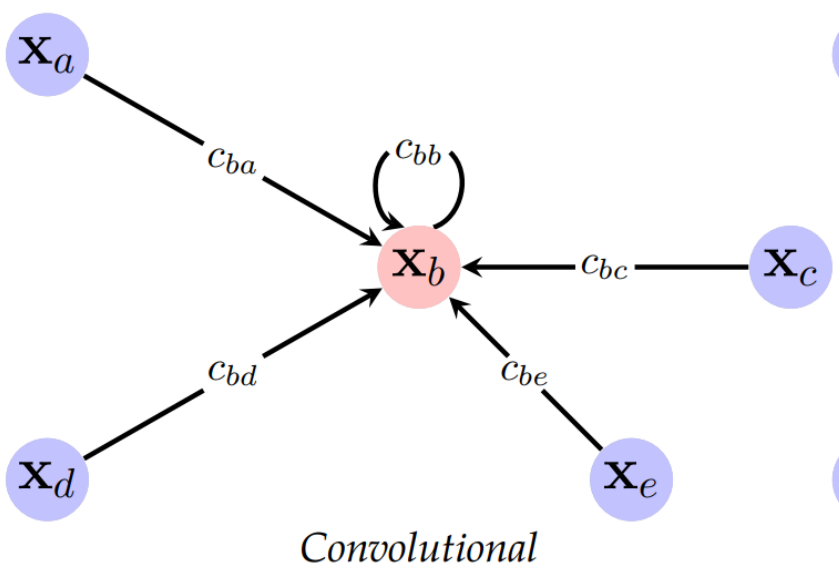
Sparse graphs offer unique nodal information and faster speeds



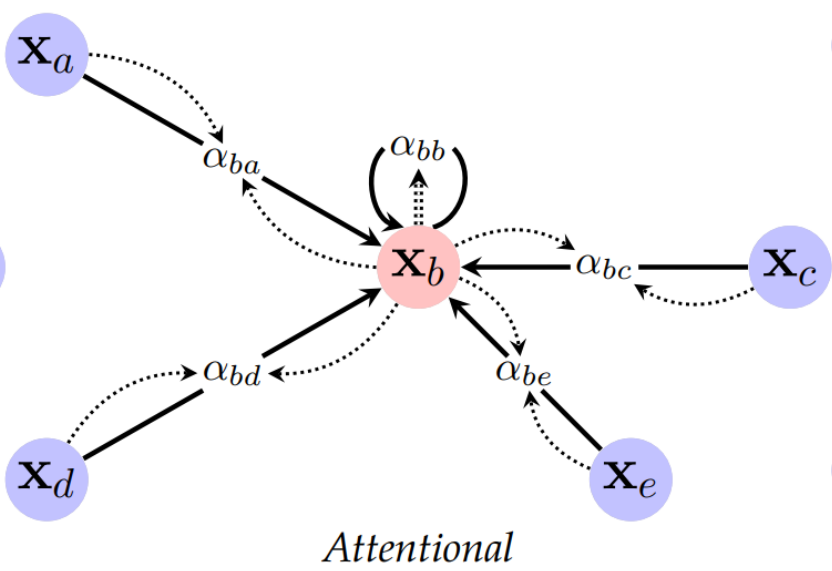
Connectivity impacts aggregation, attention, and message passing!



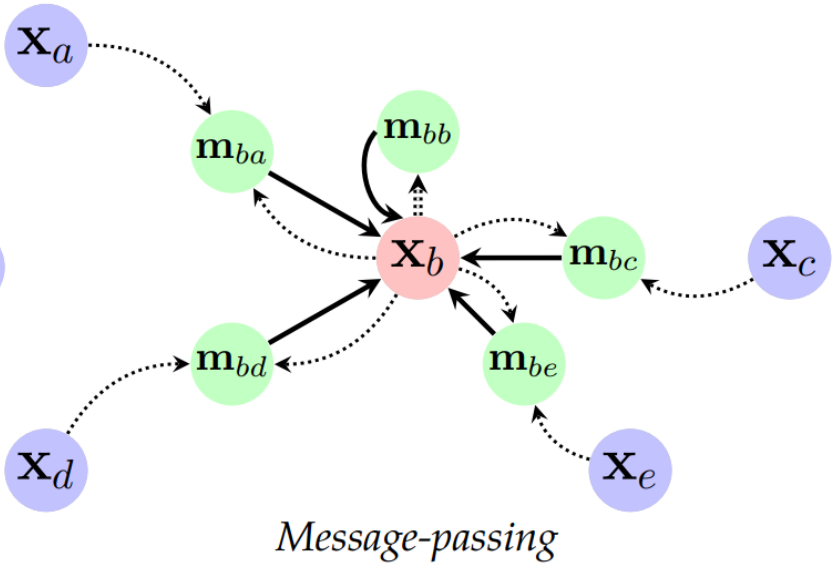
Connectivity impacts aggregation, attention, and message passing!



Convolutional

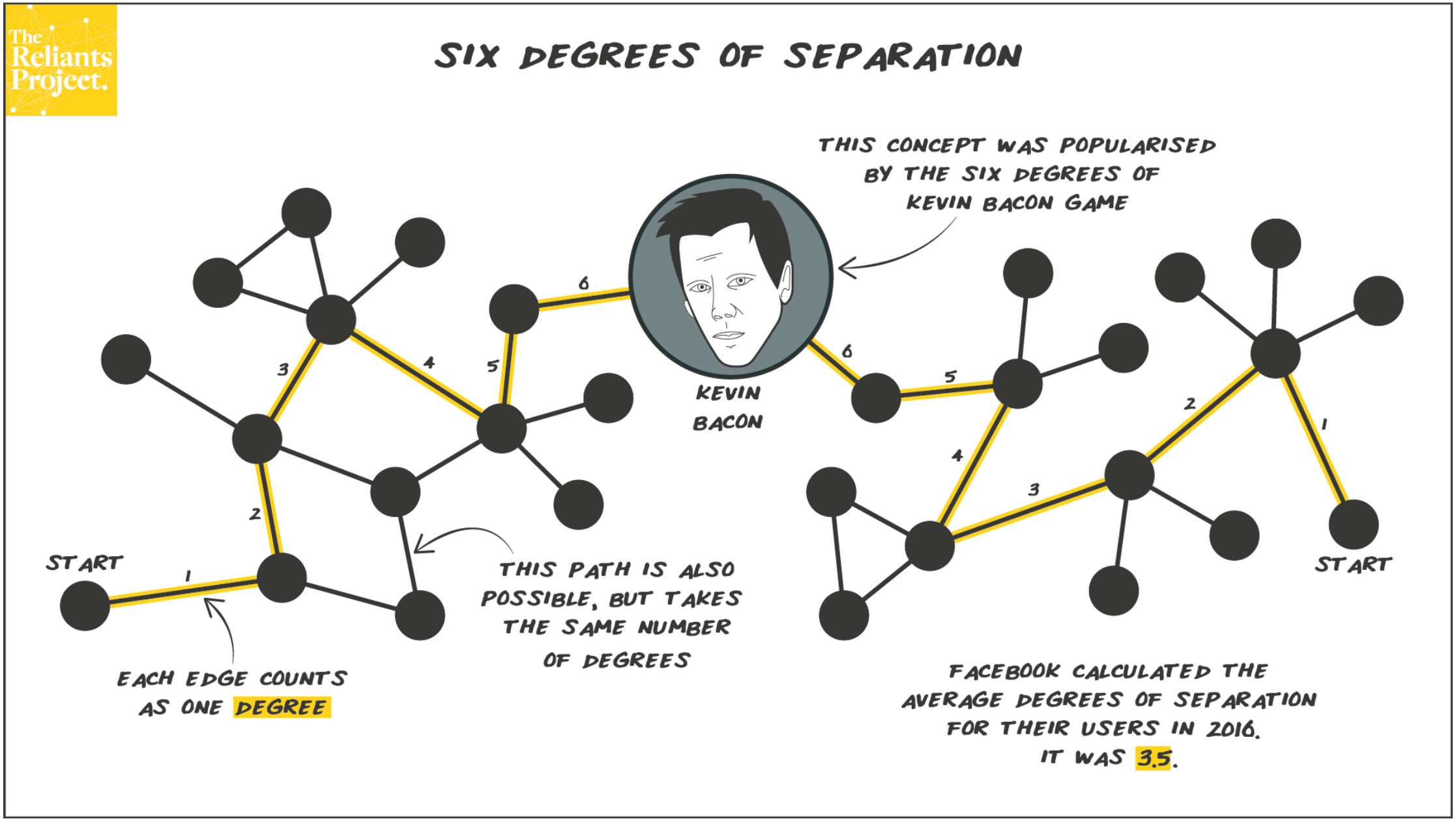


Attentional

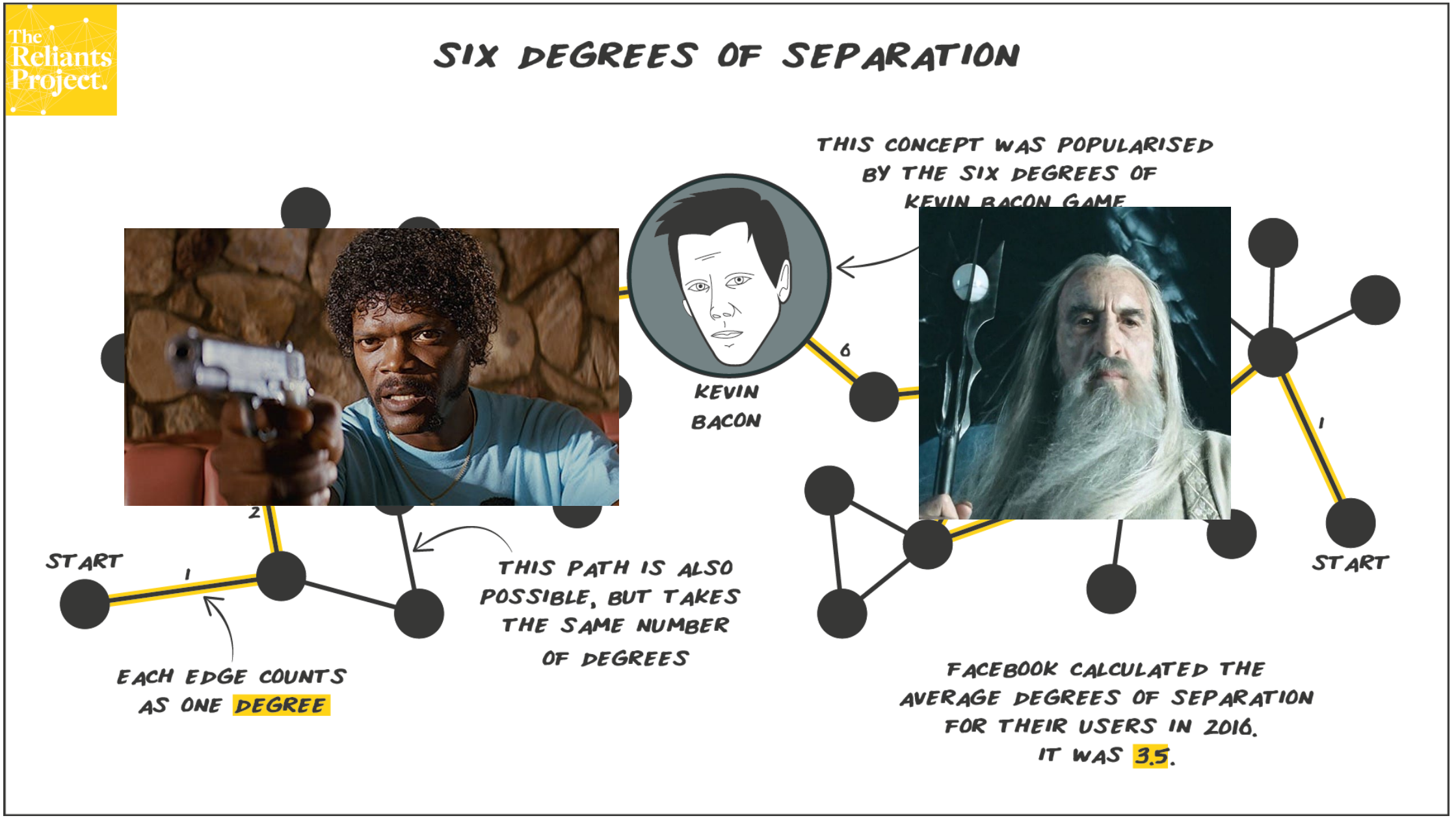


Message-passing

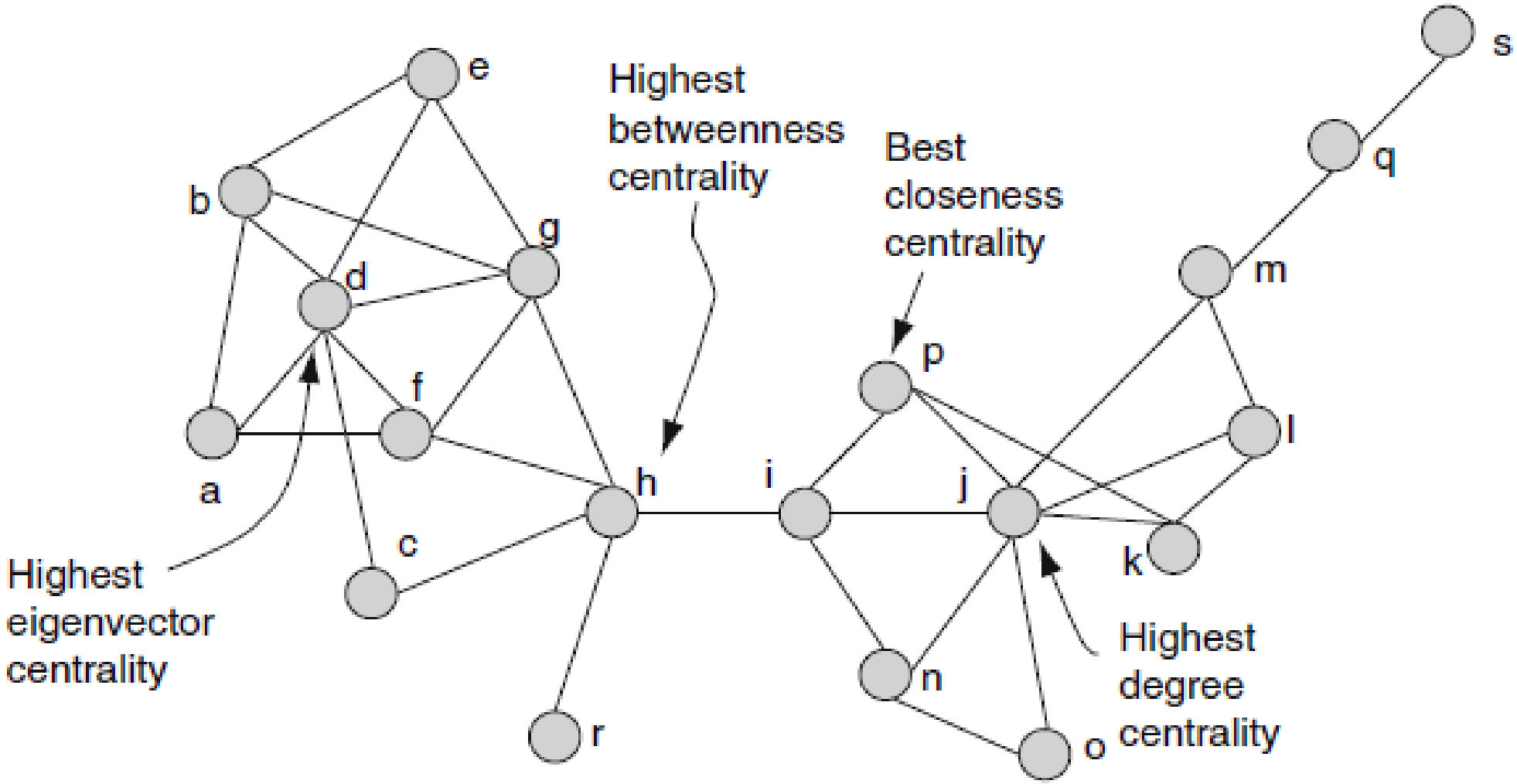
Many centralities can be considered for a given network



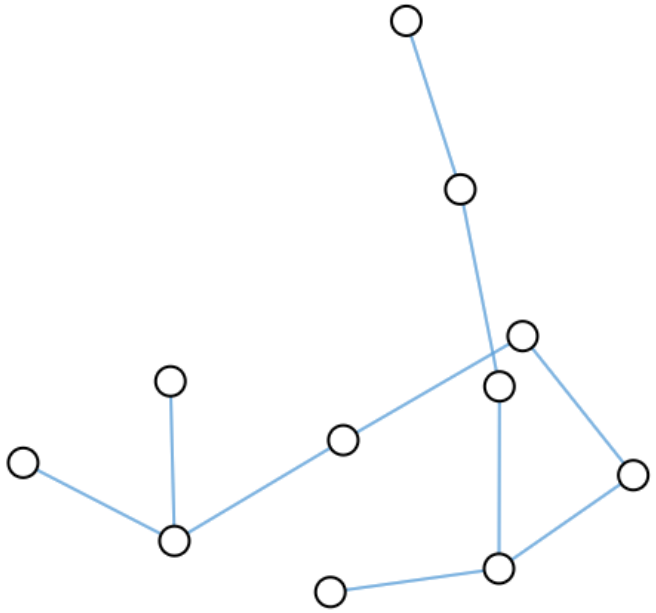
Many centralities can be considered for a given network

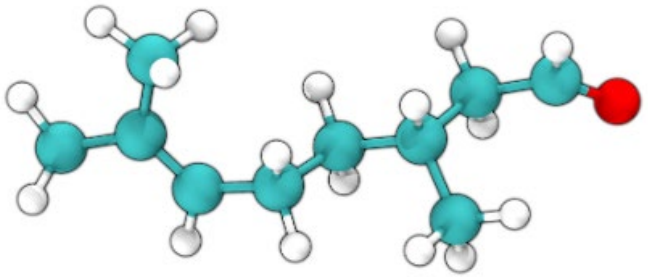


Many centralities can be considered for a given network



Tasks can be broken down to node, edge, and graph level



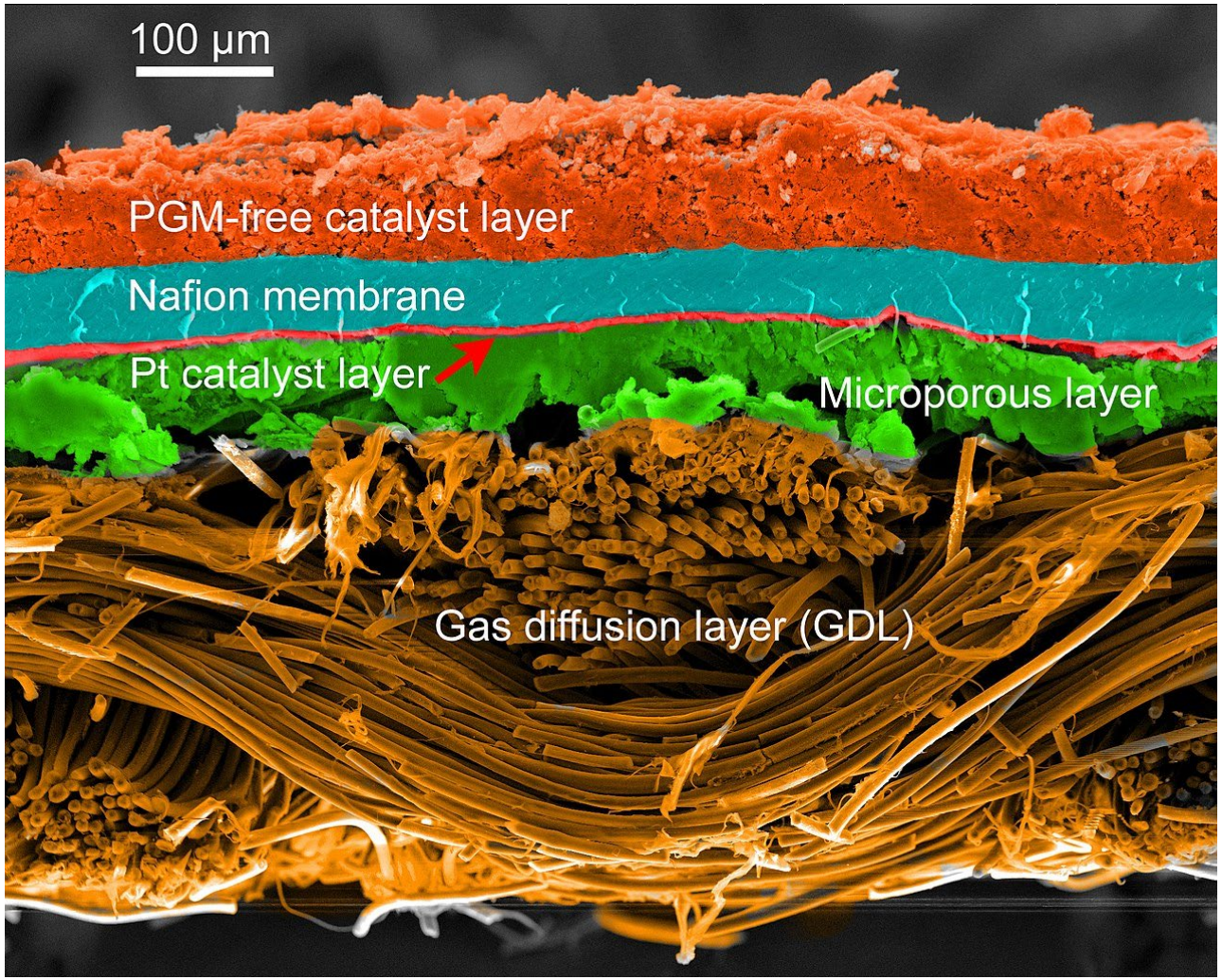


Graph level?

Node level?

Edge level?

Images can be thought of as graphs too!



https://commons.wikimedia.org/wiki/File:SEM_micrograph_of_an_MEA_cross_section.jpg

Neural networks require rectangular or grid like arrays of data inputs

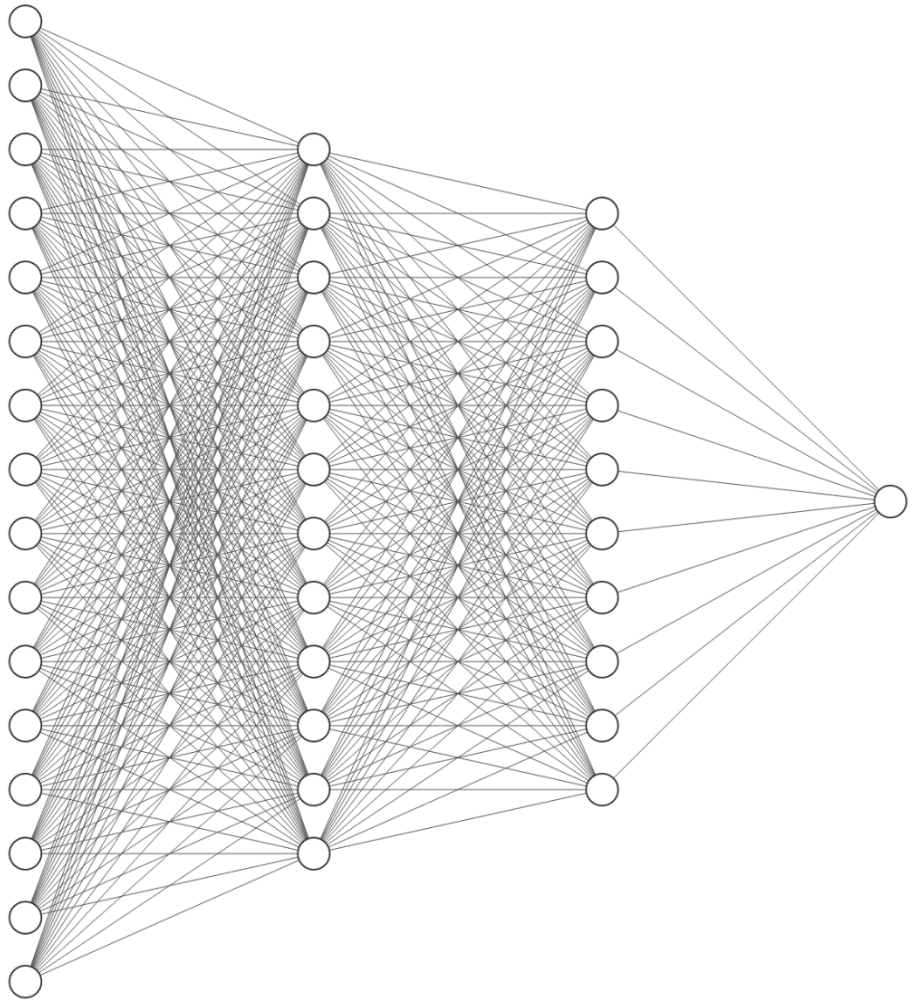
Nodes store numbers

- “activated” with a value between 0 and 1
- Functions that pass information

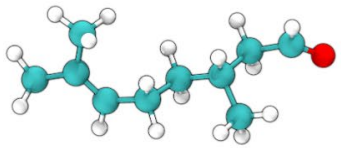
Result of network is a prediction (regression or classification)

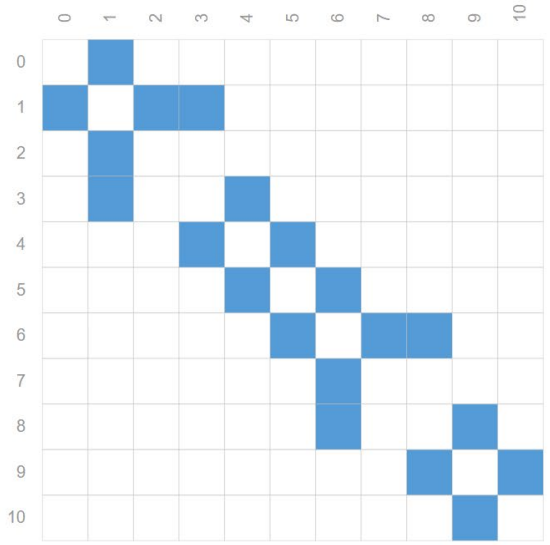
Intermediate layers allow us to build up complex non-linearity

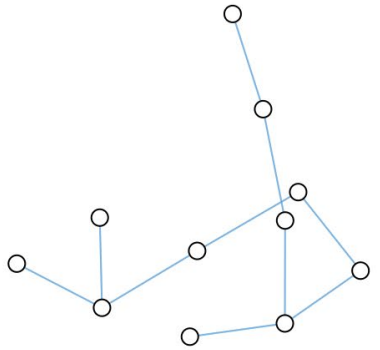
How do we make a graph compatible with NN?

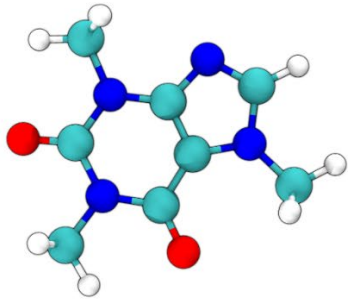


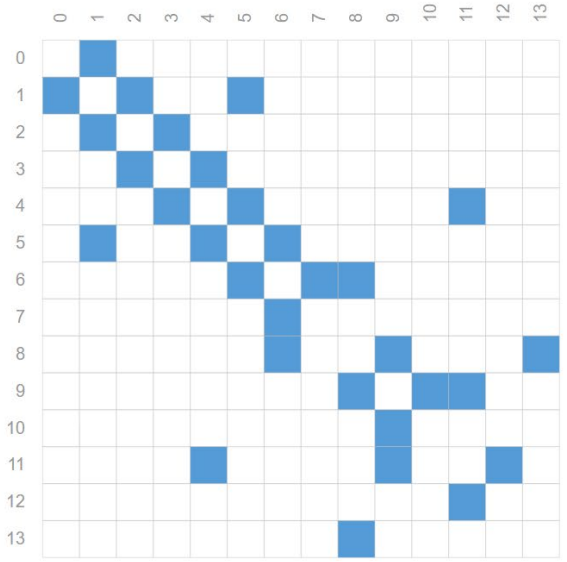
Adjacency matrices are one way to make graphs NN compatible

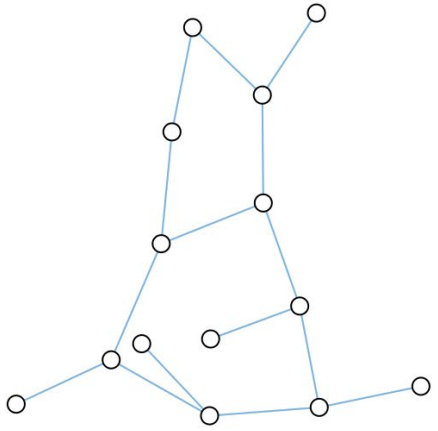










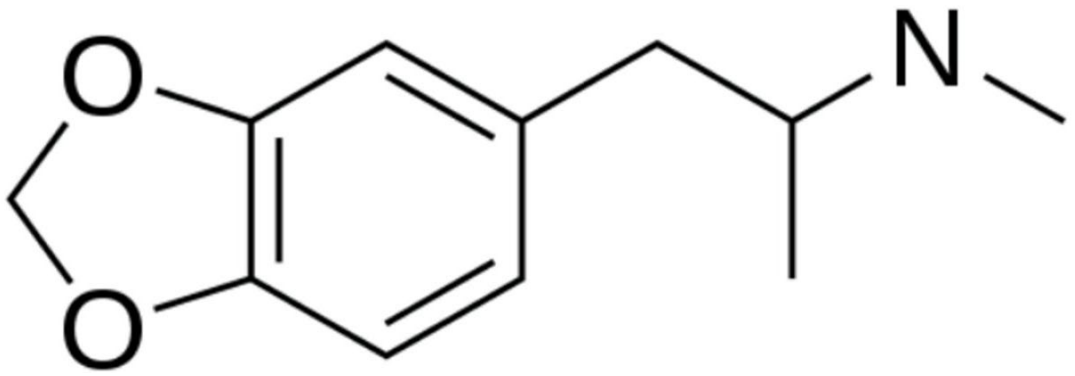


Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Matrices can contain different types of information

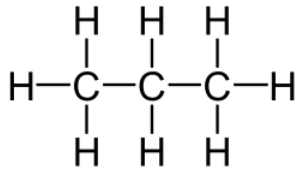
Adjacency-Matrix-based Representation

C	0	1	0	1	0	0	0	0	0	0	0	0	0
O	1	0	1	0	0	0	0	0	0	0	0	0	0
C	0	1	0	0	2	1	0	0	0	0	0	0	0
O	1	0	0	0	1	0	0	0	0	0	0	0	0
C	0	0	2	1	0	0	1	0	0	0	0	0	0
C	0	0	1	0	0	0	0	0	2	0	0	0	0
C	0	0	0	0	1	0	0	2	0	0	0	0	0
C	0	0	0	0	0	0	2	0	1	0	0	0	0
C	0	0	0	0	0	2	0	1	0	1	0	0	0
C	0	0	0	0	0	0	0	0	1	0	1	0	0
C	0	0	0	0	0	0	0	0	0	1	0	1	0
N	0	0	0	0	0	0	0	0	0	0	1	0	1
C	0	0	0	0	0	0	0	0	0	0	0	1	0
C	0	0	0	0	0	0	0	0	0	0	1	0	0



https://twitter.com/MarioKrenn6240/status/1348814674415001600?s=20

Adjacency matrices have some limitations



Permutation invariance

- If you ordered points differently for the same structure, you should get the same representation
- Adjacency lists help here!

	C1	C2	C3
C1	0	1	0
C2	1	0	1
C3	0	1	0

	C1	C2	C3
C1	0	0	1
C2	0	0	1
C3	1	1	0

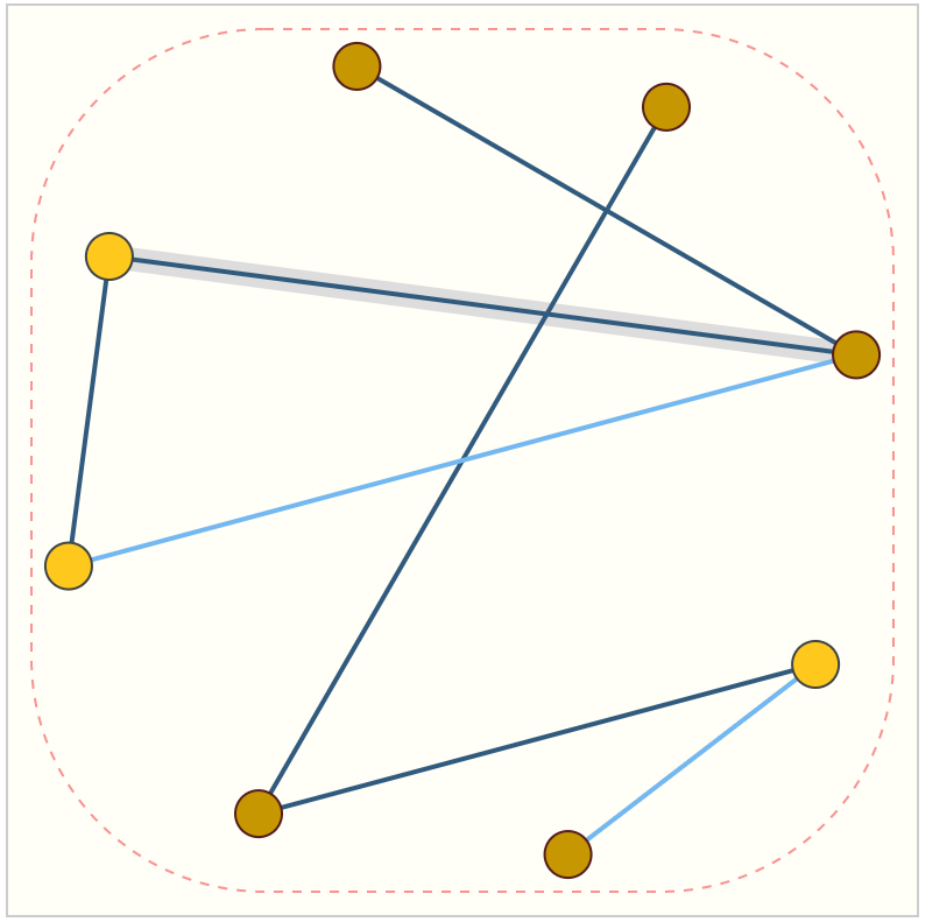
Adjacency-Matrix-based Representation

Representation should be space efficient

- Adjacency matrices can end up being very sparse if most nodes are only connected to a few others of many

$$\begin{pmatrix} C \\ O \\ C \\ O \\ C \\ C \\ C \\ C \\ C \\ C \\ C \\ N \\ C \\ C \end{pmatrix} \quad \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Remember, graphs have 4 types of info!



Nodes

```
[0, 1, 1, 0, 0, 1, 1, 1]
```

Edges

```
[2, 1, 1, 1, 2, 1, 1]
```

Adjacency List

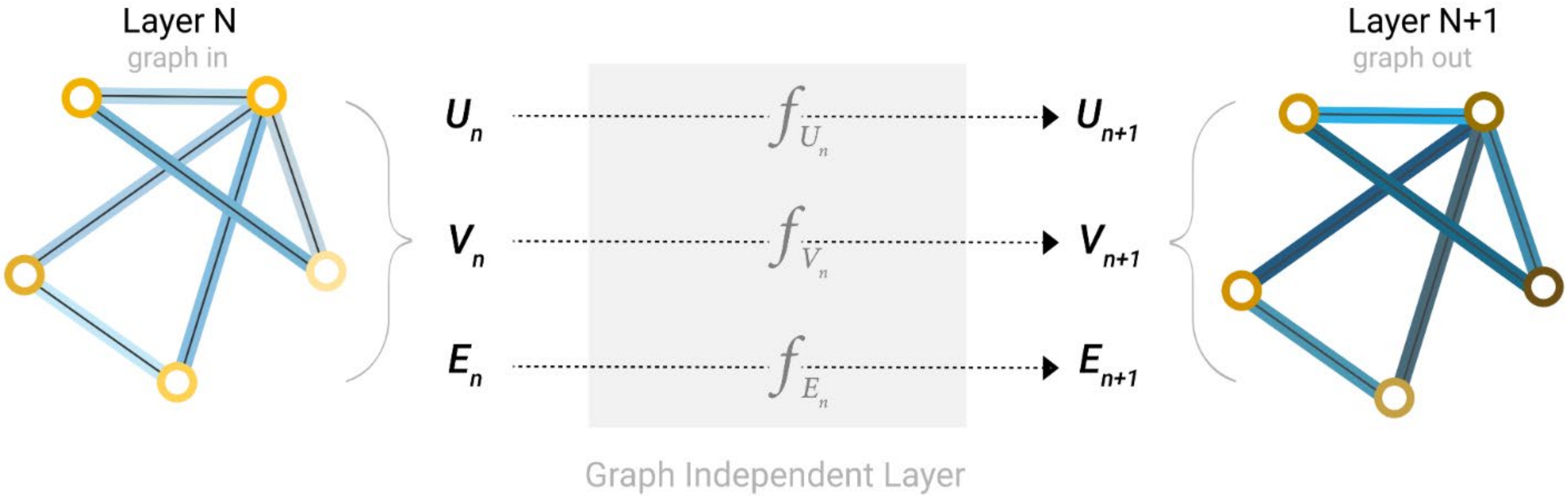
```
[[1, 0], [2, 0], [4, 3], [6, 2],
 [7, 3], [7, 4], [7, 5]]
```

Global

```
0
```

Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Most GNNs use graph in – graph out approach

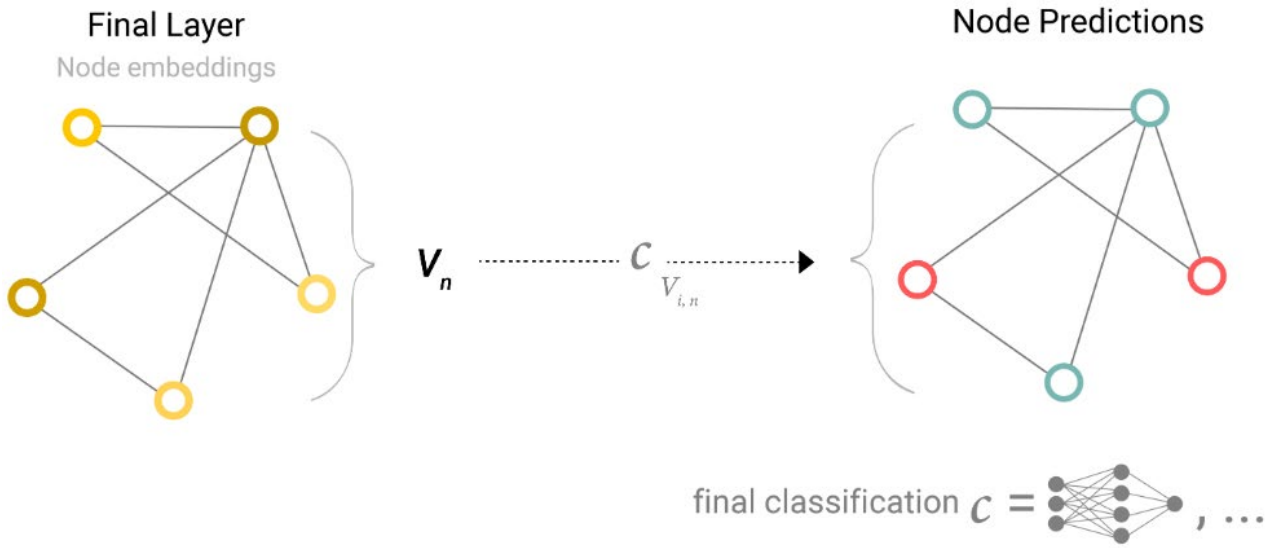


Simplest would be to use MLP for each node, edge, and graph attribute.

update function $f =$  , ...

Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

We have various options when it comes to making predictions



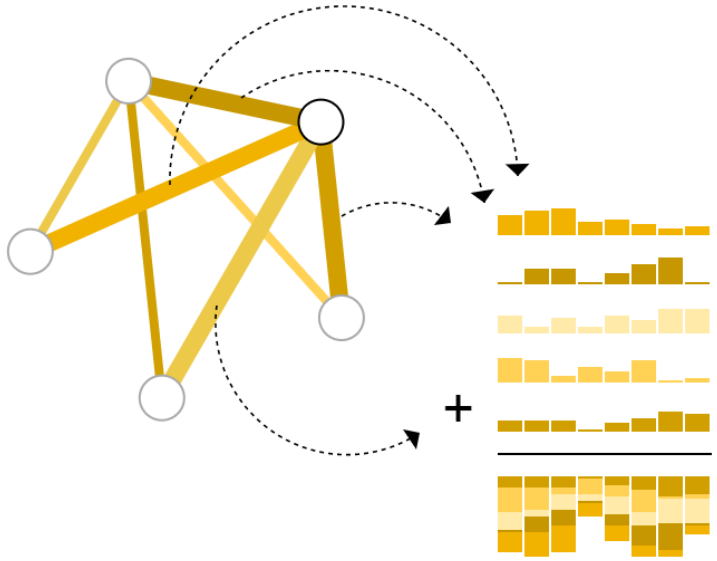
If you have information for each node and you need to make a prediction for each node

- Simply apply an activation function!

What if node prediction needed with only edge info?

Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

We can learn neighbor interactions through pooling!



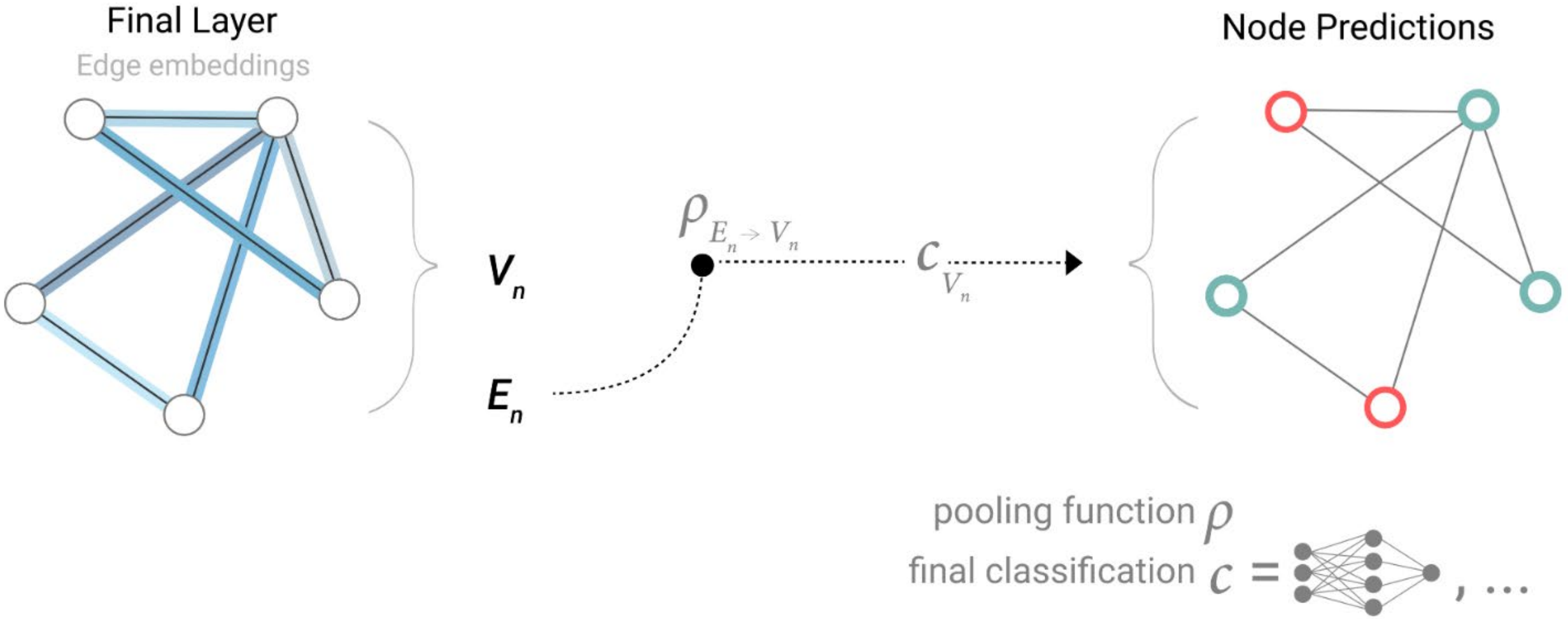
Aggregate information
from adjacent edges

A matrix of edge vectors is created and these are aggregated via sum operation to update node value

$$\rho_{E_n \to V_n}$$

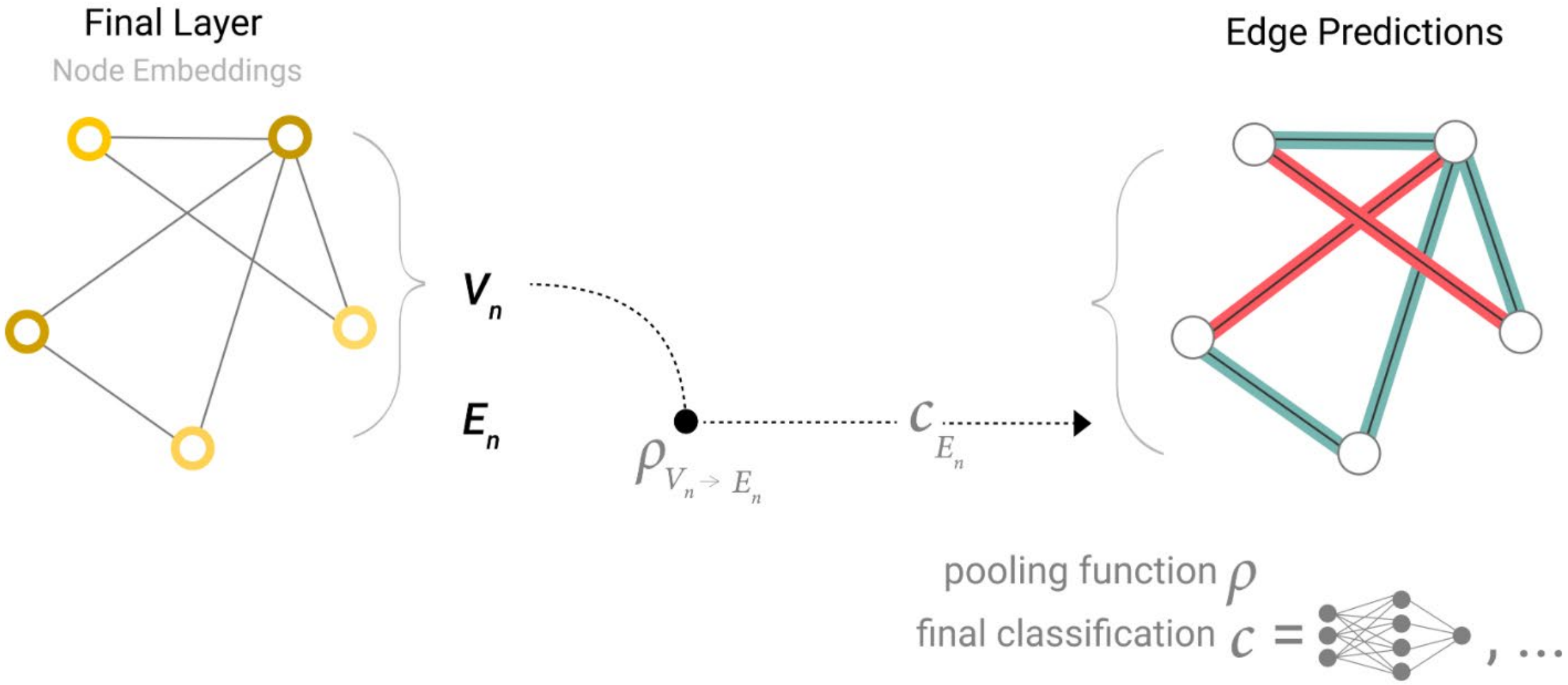
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Using pooling we can predict, node, edge, or global attributes



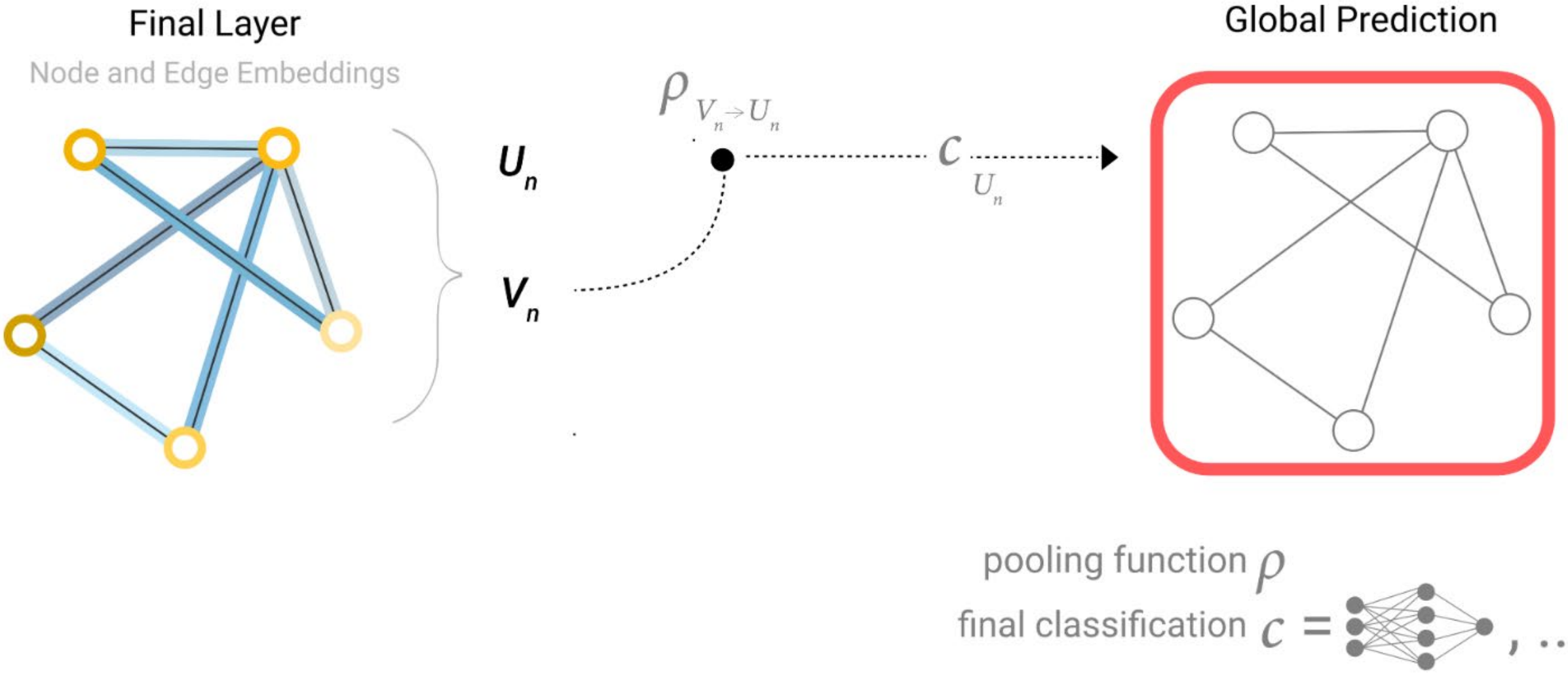
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Using pooling we can predict, node, edge, or global attributes



Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

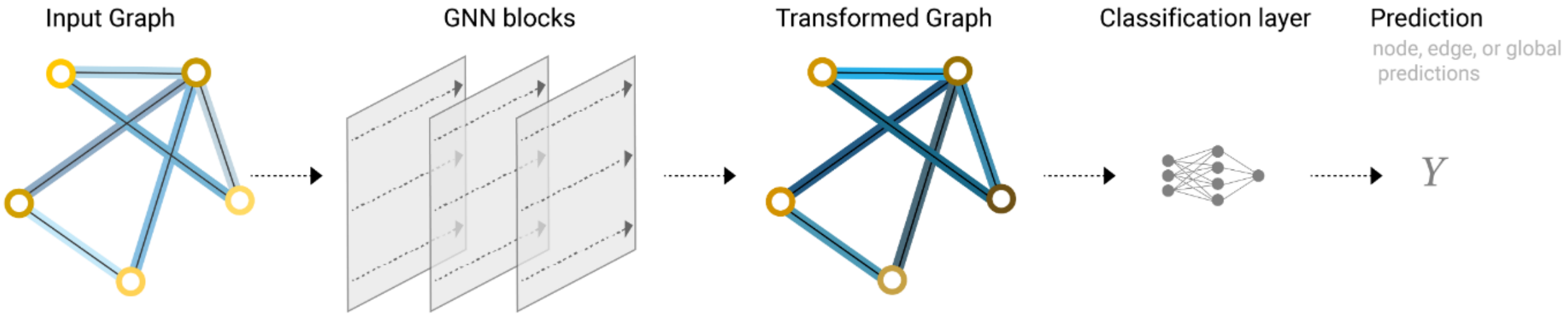
Using pooling we can predict, node, edge, or global attributes



Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Simple GNNs do not use connectivity in calculations

Each node , edge, and graph attribute is processed independently and connectivity is only used during pooling



An end-to-end prediction task with a GNN model.

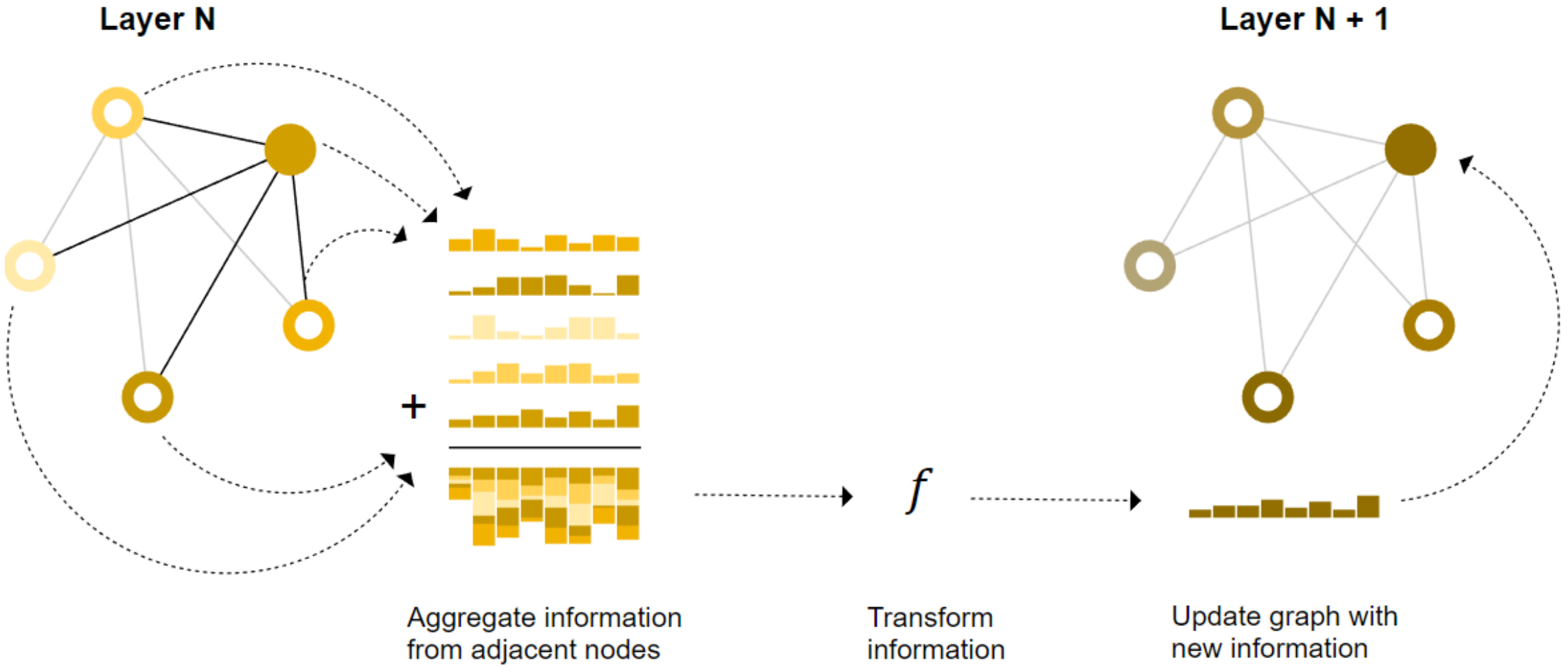
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Message passing allows connectivity to influence embeddings

1. For each node in the graph, gather all the neighboring node embeddings (or messages)
2. Aggregate all messages via an aggregate function (like sum)
3. All pooled messages are passed through an update function, usually a learned neural network

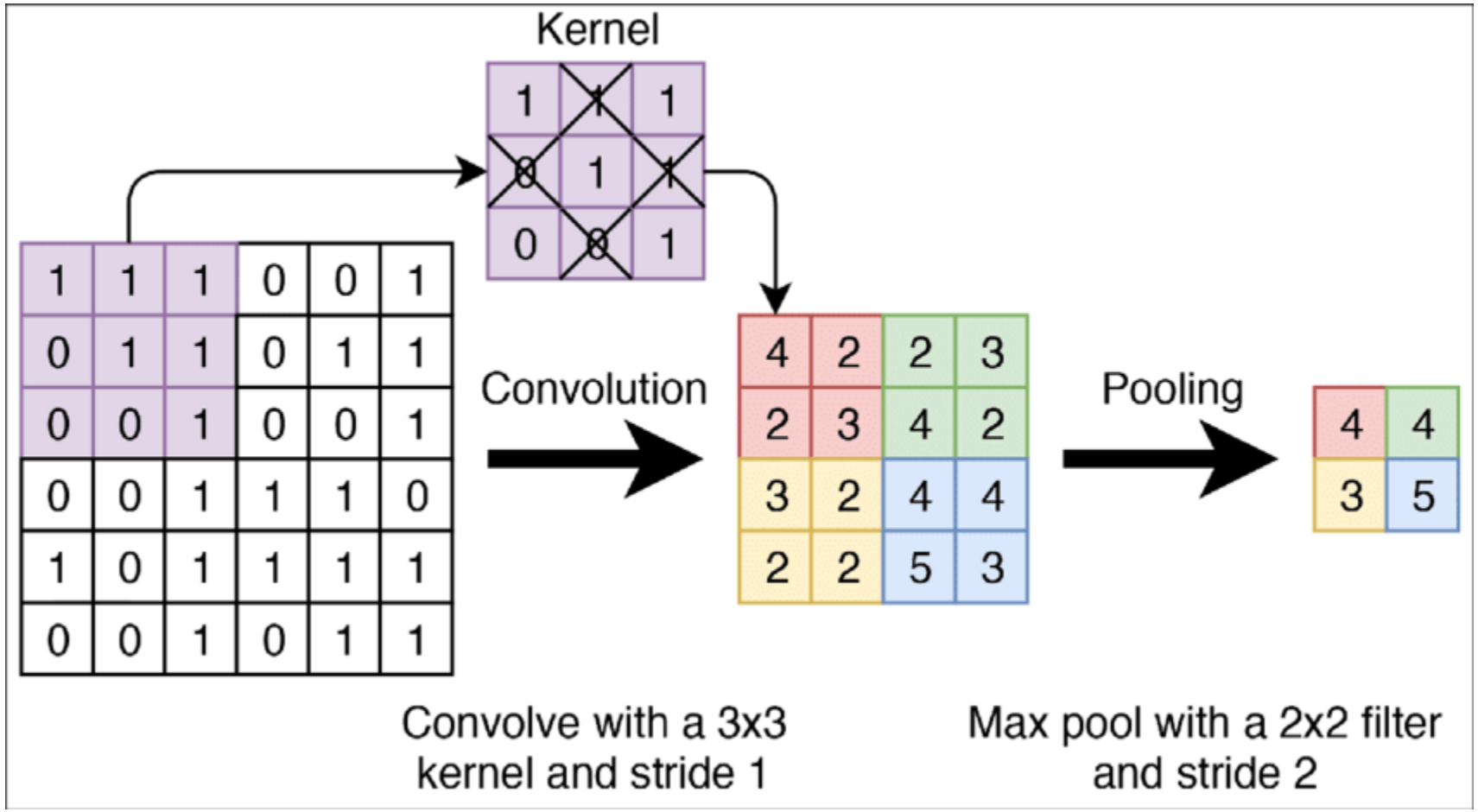
Gilmer et al “Neural Message Passing for Quantum Chemistry” Proc 34th International Conference on Machine Learning, 2017.

Message passing allows connectivity to influence embeddings



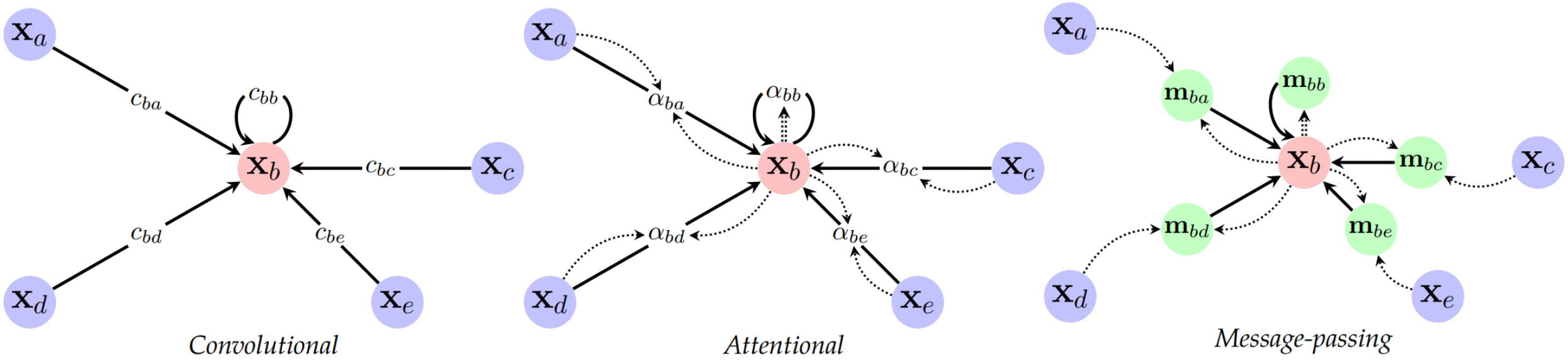
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Message passing is similar to convolutional neural networks

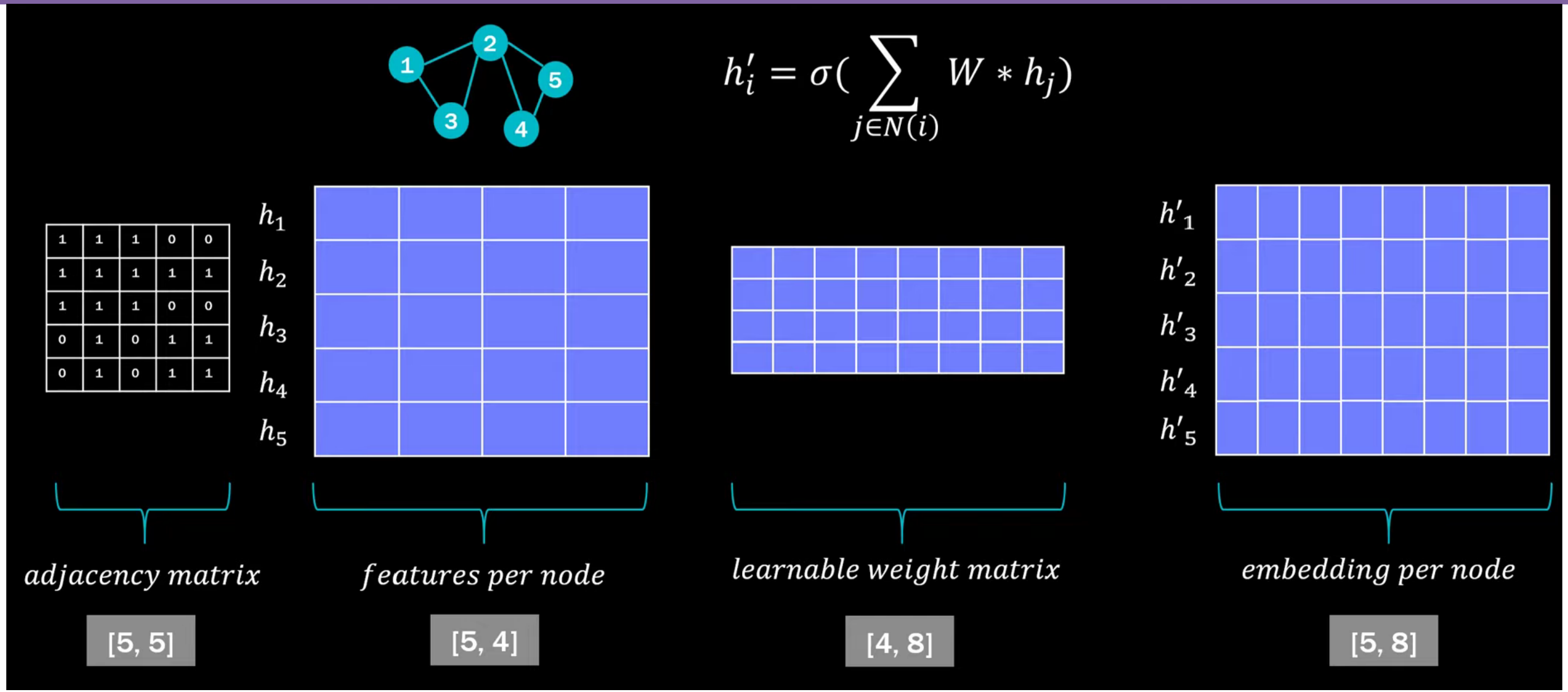


Verschoof-van der Vaart and Lambers, J. Comp. App. in Arch., (2019)

We can use connectivity in different ways



Message passing learns from all connected nodes



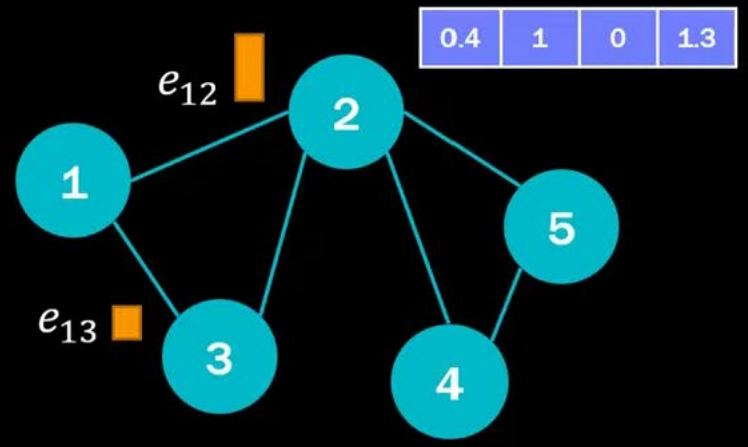
Velickovik, “Graph attention networks” ICLR 2018

DeepFindR “Understanding graph attention networks”, YouTube 2021

We know that some connections are more important than others

$$e_{ij} = a(Wh_i, Wh_j)$$

attention coefficient



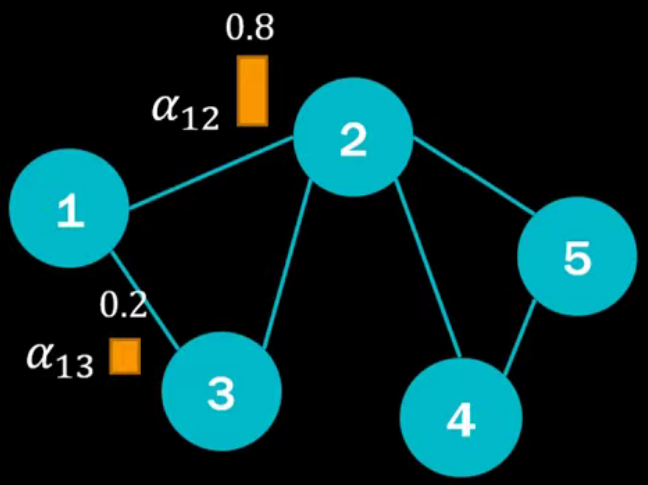
Velickovik, “Graph attention networks” ICLR 2018

DeepFindR “Understanding graph attention networks”, YouTube 2021

We can learn which nodes to pay attention to over others

$$e_{ij} = a(Wh_i, Wh_j)$$

attention coefficient



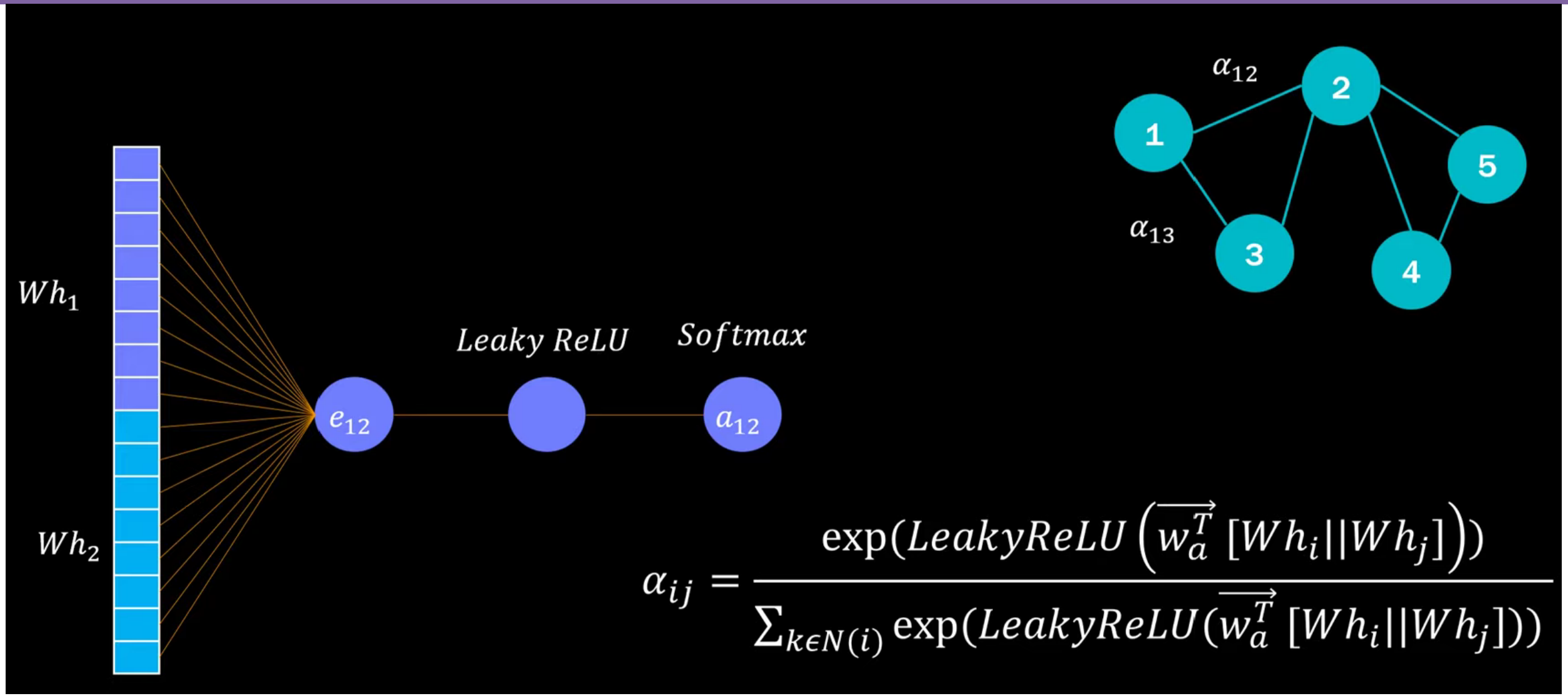
$$\alpha_{ij} = softmax_j(e_{ij}) = \frac{\exp(e_{ij})}{\sum_{k \epsilon N(i)} \exp(e_{ik})} = \frac{\exp(a(Wh_i, Wh_j))}{\sum_{k \epsilon N(i)} \exp(a(Wh_i, Wh_j))}$$

normalized attention coefficient

Velickovik, “Graph attention networks” ICLR 2018

DeepFindR “Understanding graph attention networks”, YouTube 2021

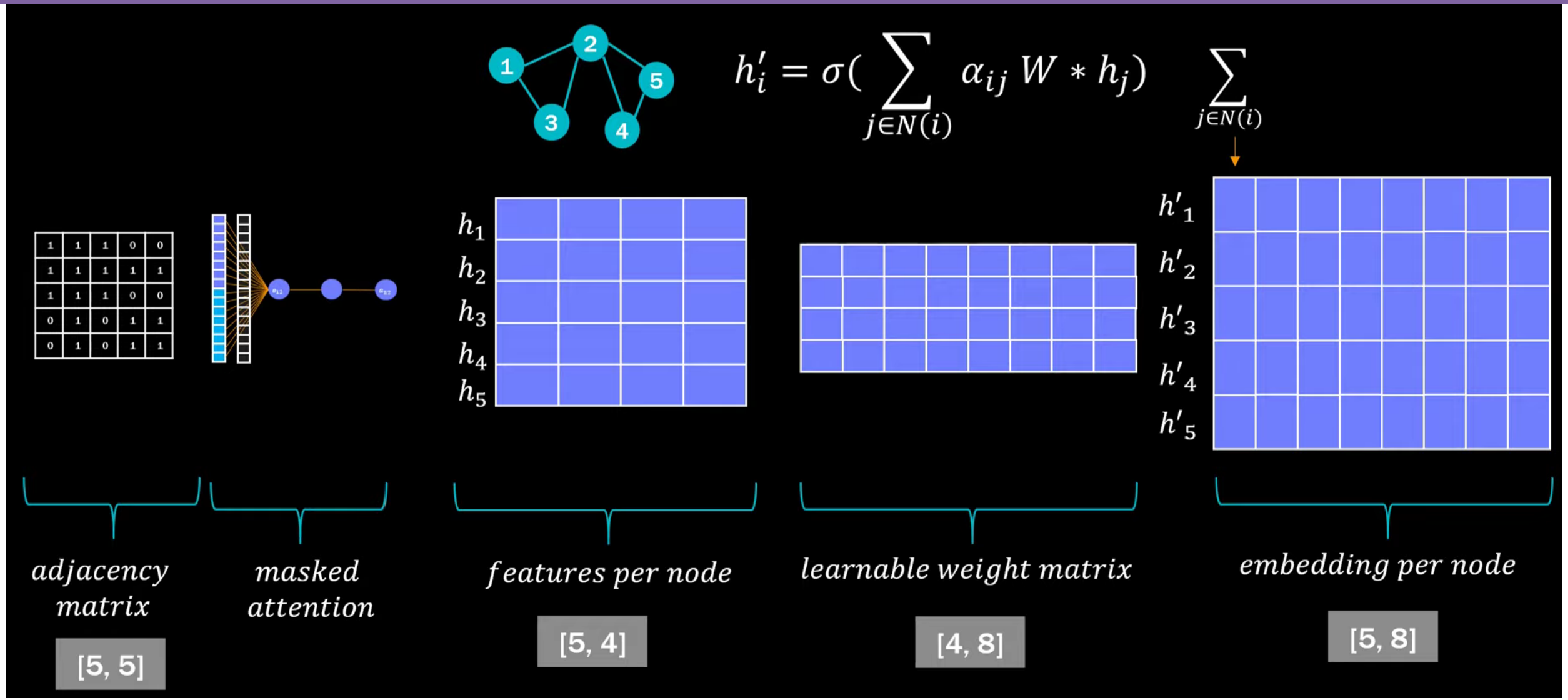
Attention coefficients are learned for all connections



Velickovik, “Graph attention networks” ICLR 2018

DeepFindR “Understanding graph attention networks”, YouTube 2021

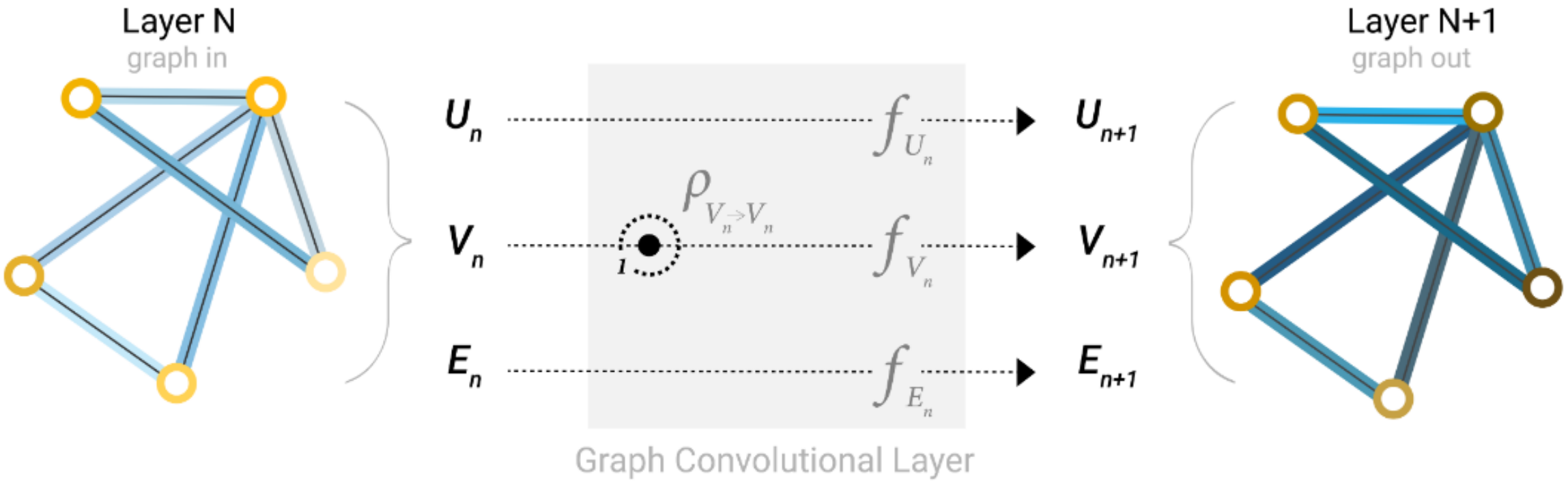
Overall, it looks similar to message passing, but with more learning



Velickovik, “Graph attention networks” ICLR 2018

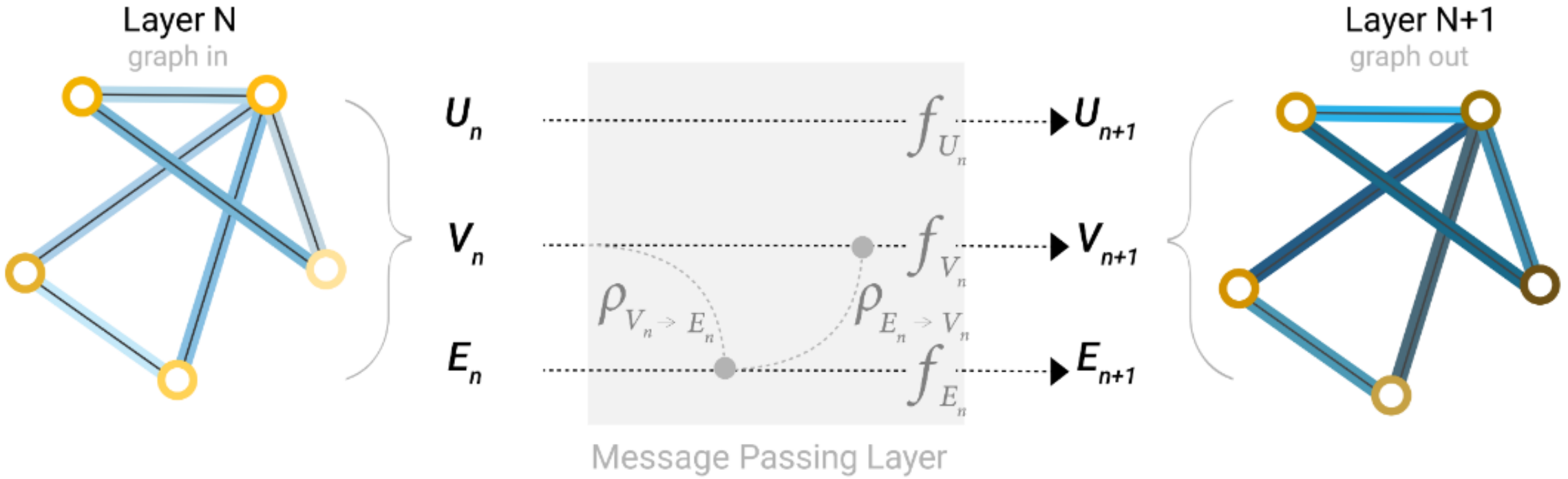
DeepFindR “Understanding graph attention networks”, YouTube 2021

Repeated layers expands information from across graph



Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

We can also go between components with message passing

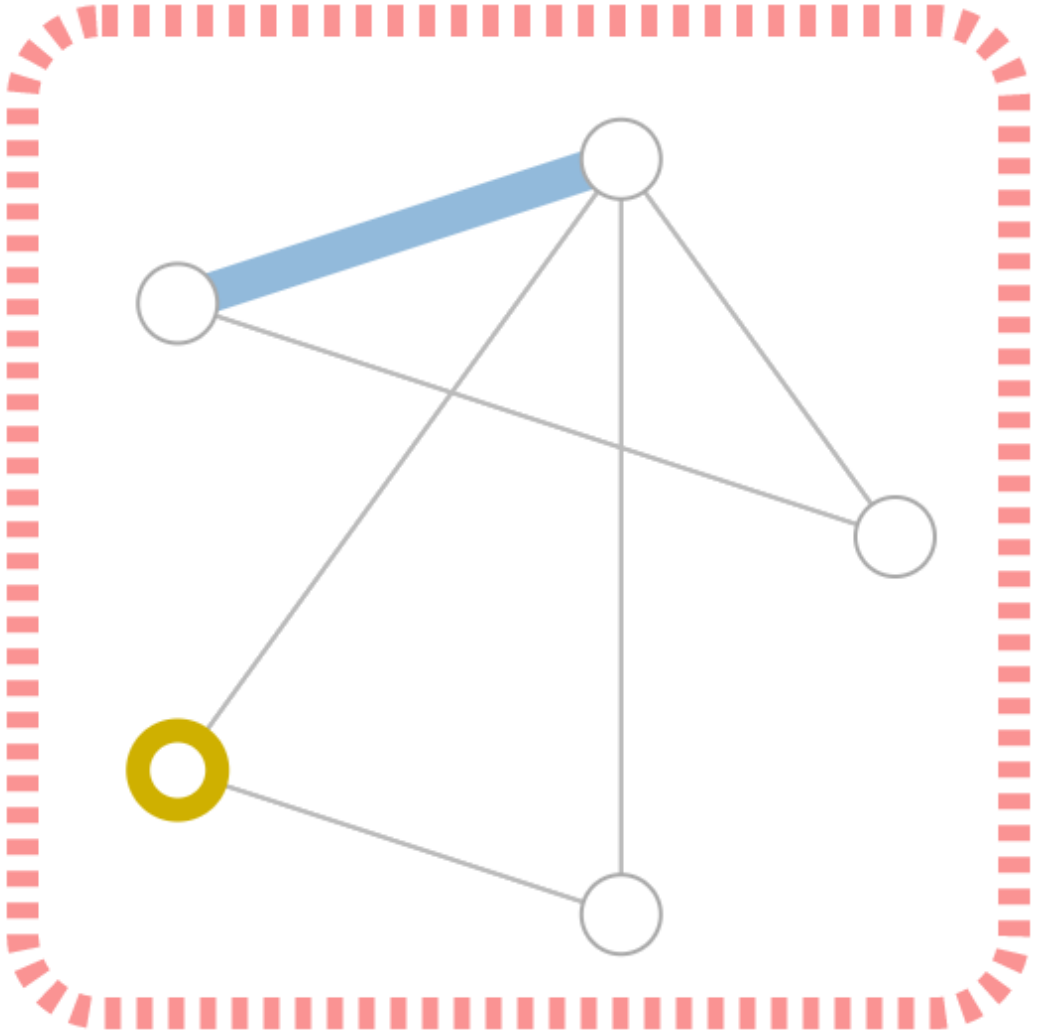


update function $f =$, ...

pooling function ρ

Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

What to we do if our edge/node embeddings are different shape?



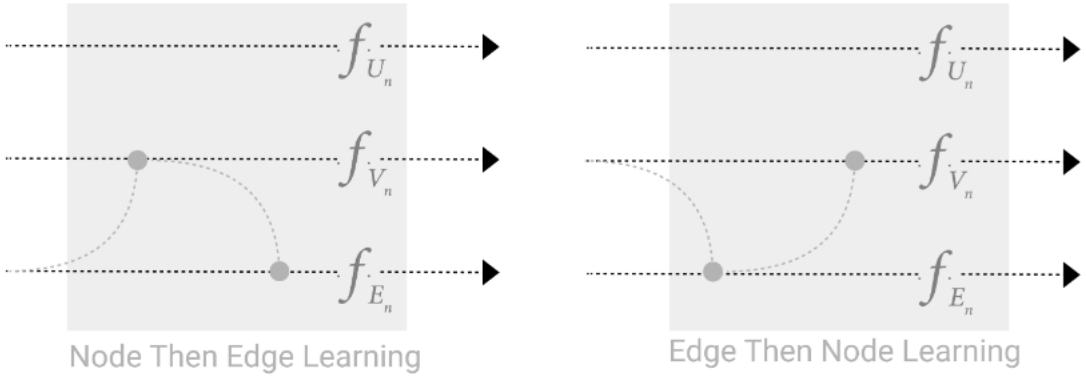
Vertex (or node) embedding

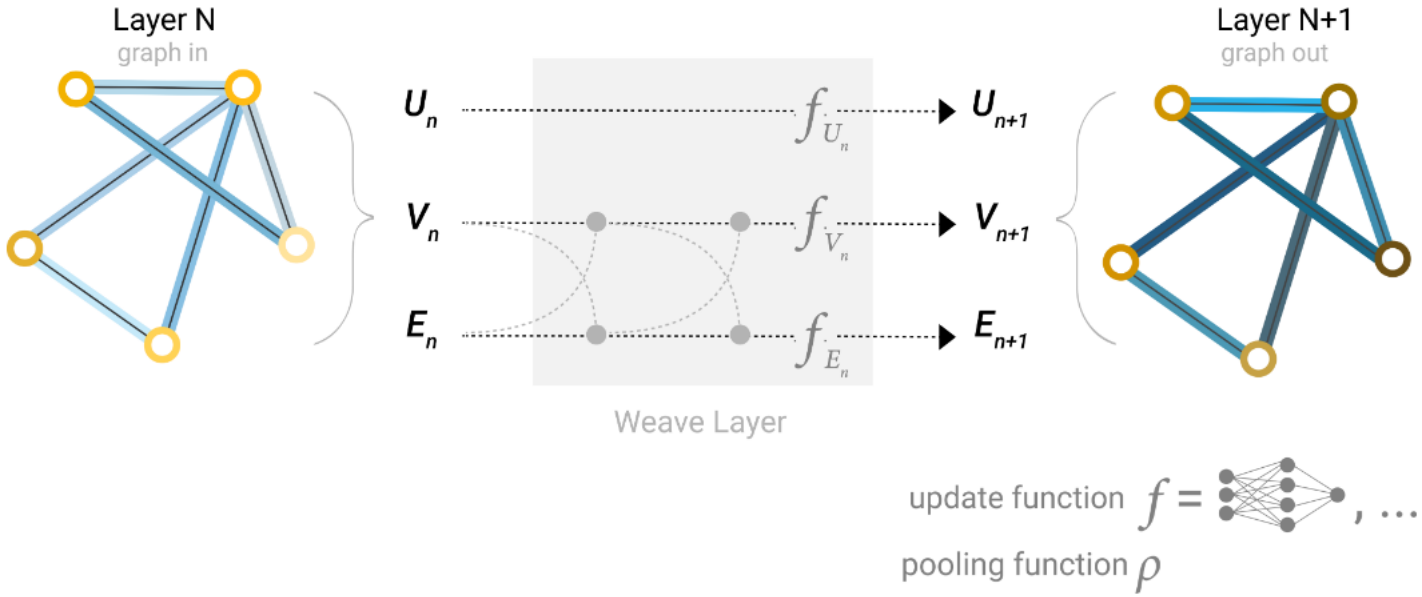
Edge (or link) attributes and embedding

Global (or master node) embedding

Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

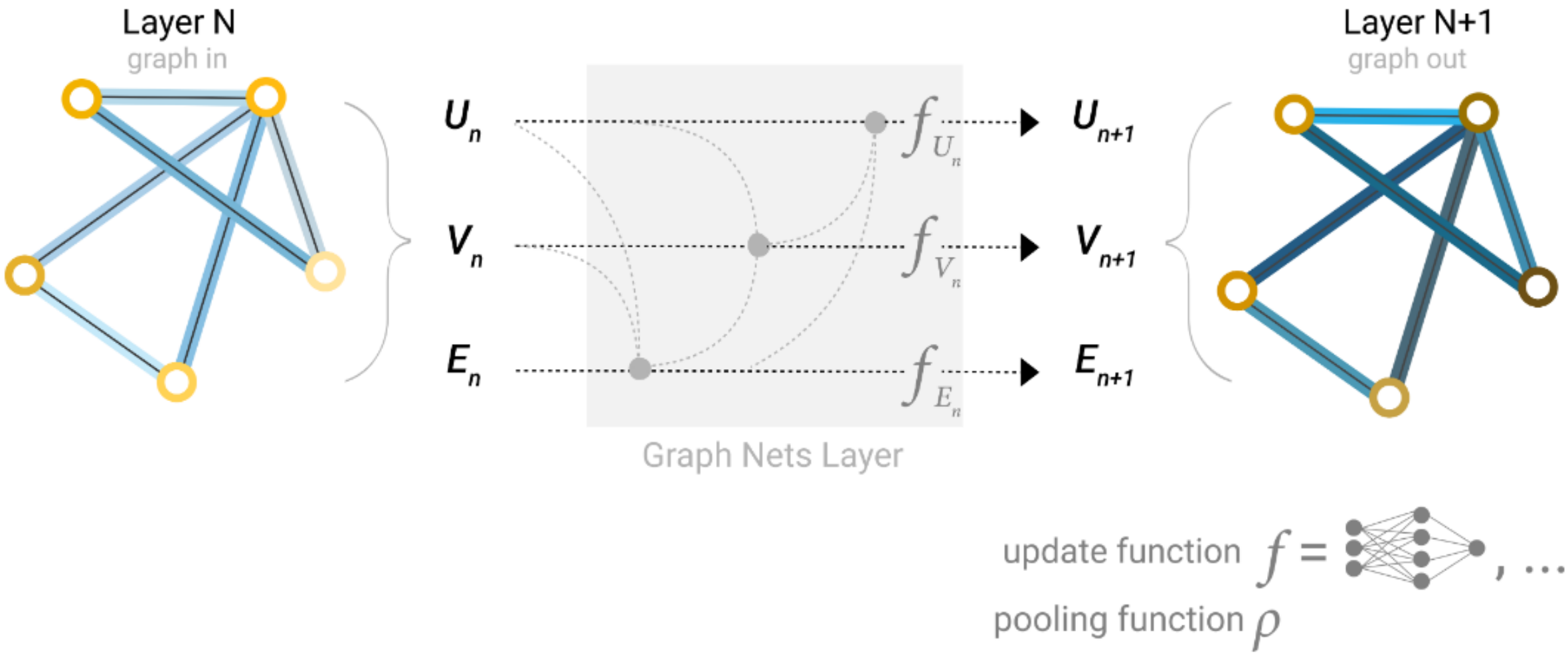
We have many options and choices when it comes to order





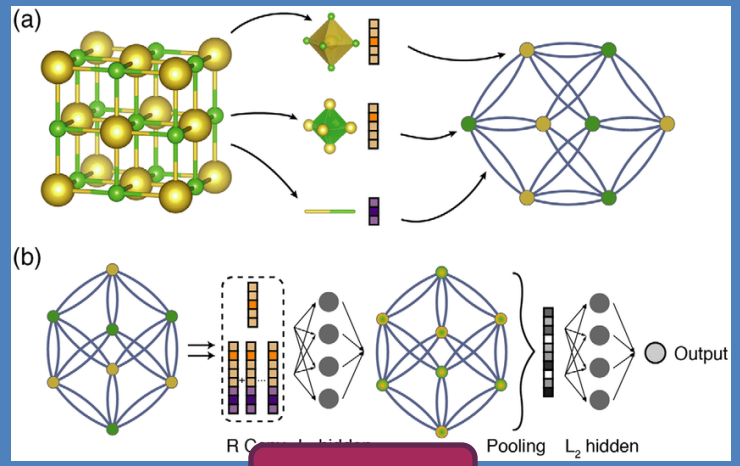
Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

Global attribute acts as a “master node” to reduce graph distances

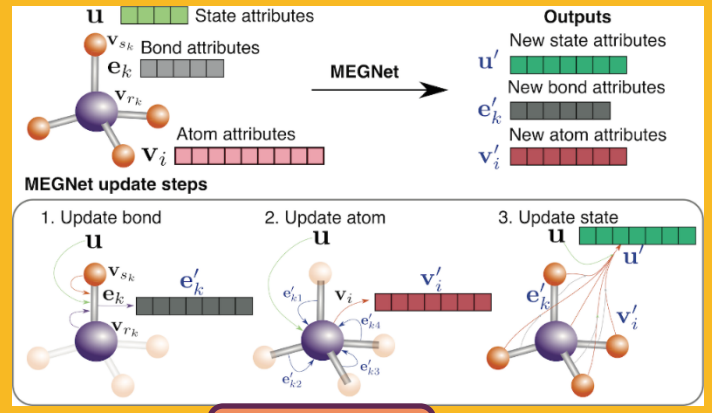


Sanchez-Lengeling et al. "A Gentle Introduction to Graph Neural Networks", Distill, 2021

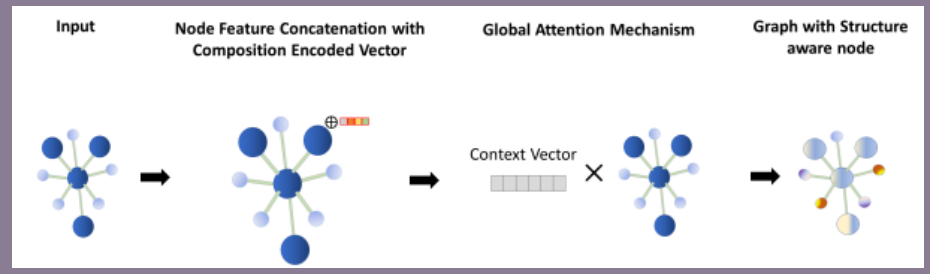
Some of the best materials informatics tools are graph networks



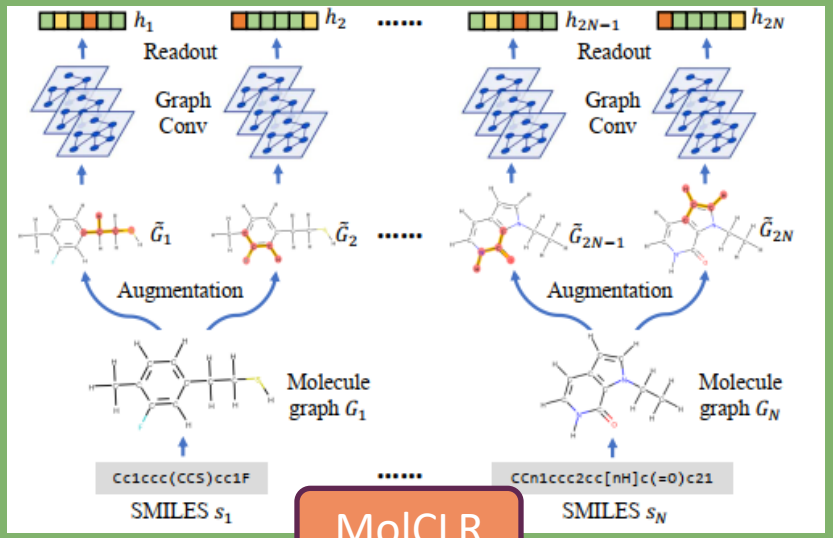
CGCNN



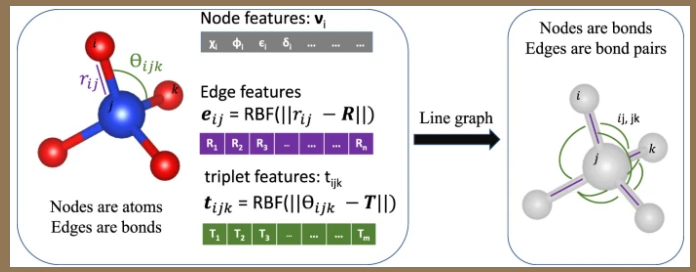
MEGNet



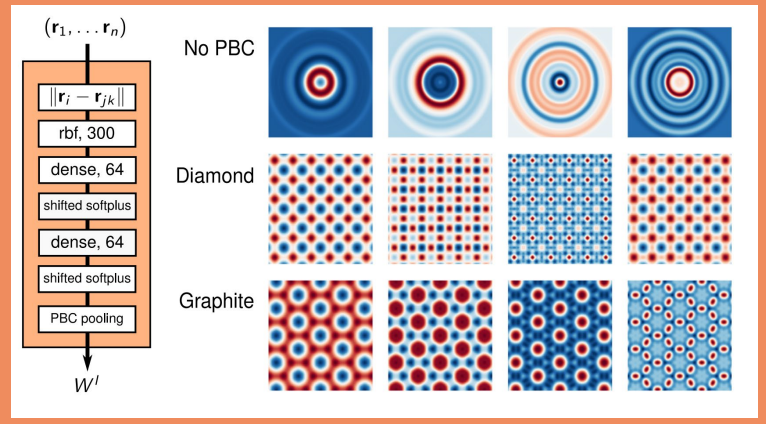
GATGNN



MolCLR



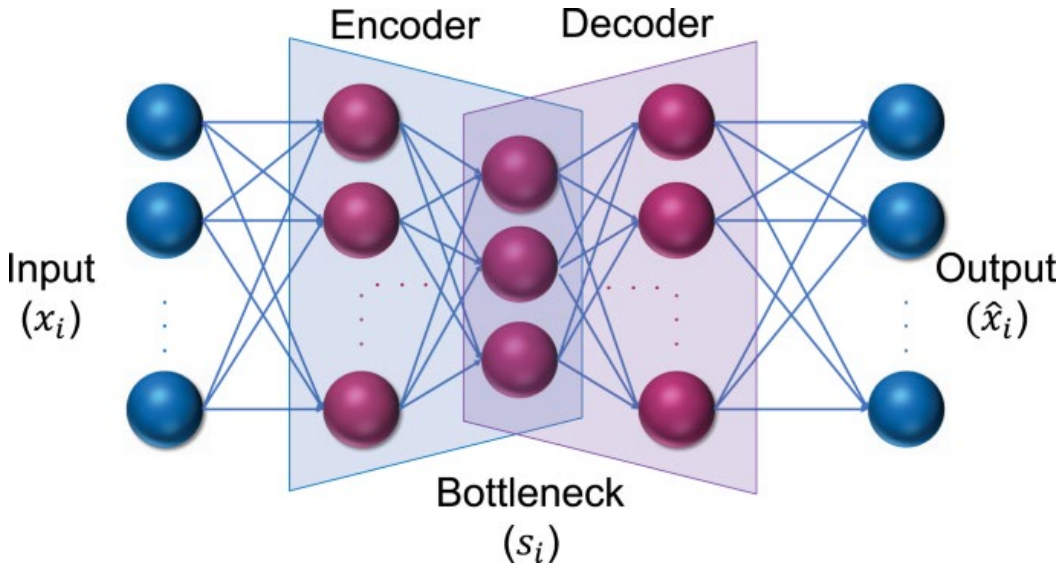
ALIGNN



SchNet

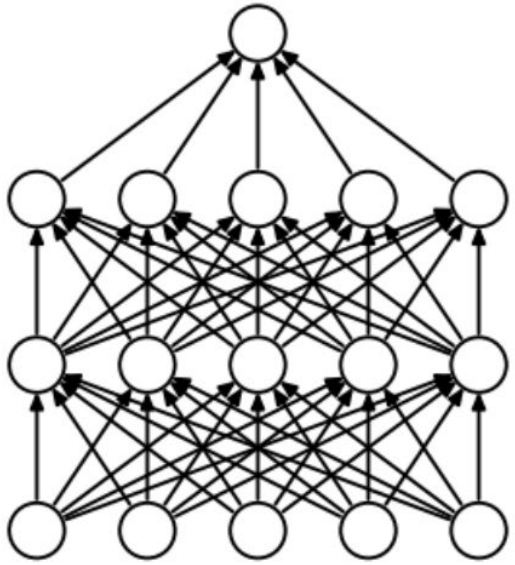
What do new model architectures give the community?

Is feature reduction sufficient? Or is something else needed?

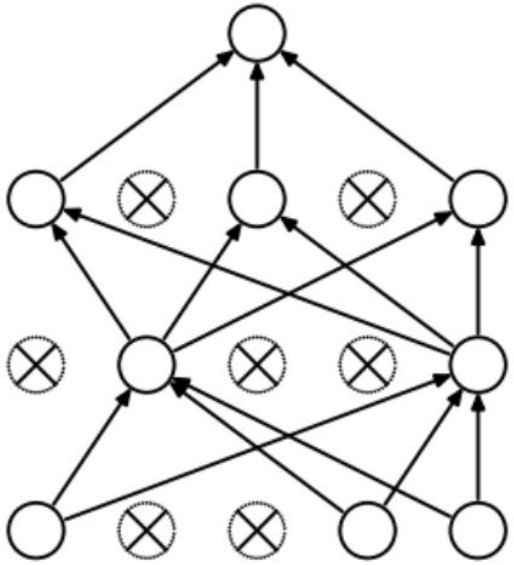


Feature reduction

Dilution (dropout) regularization



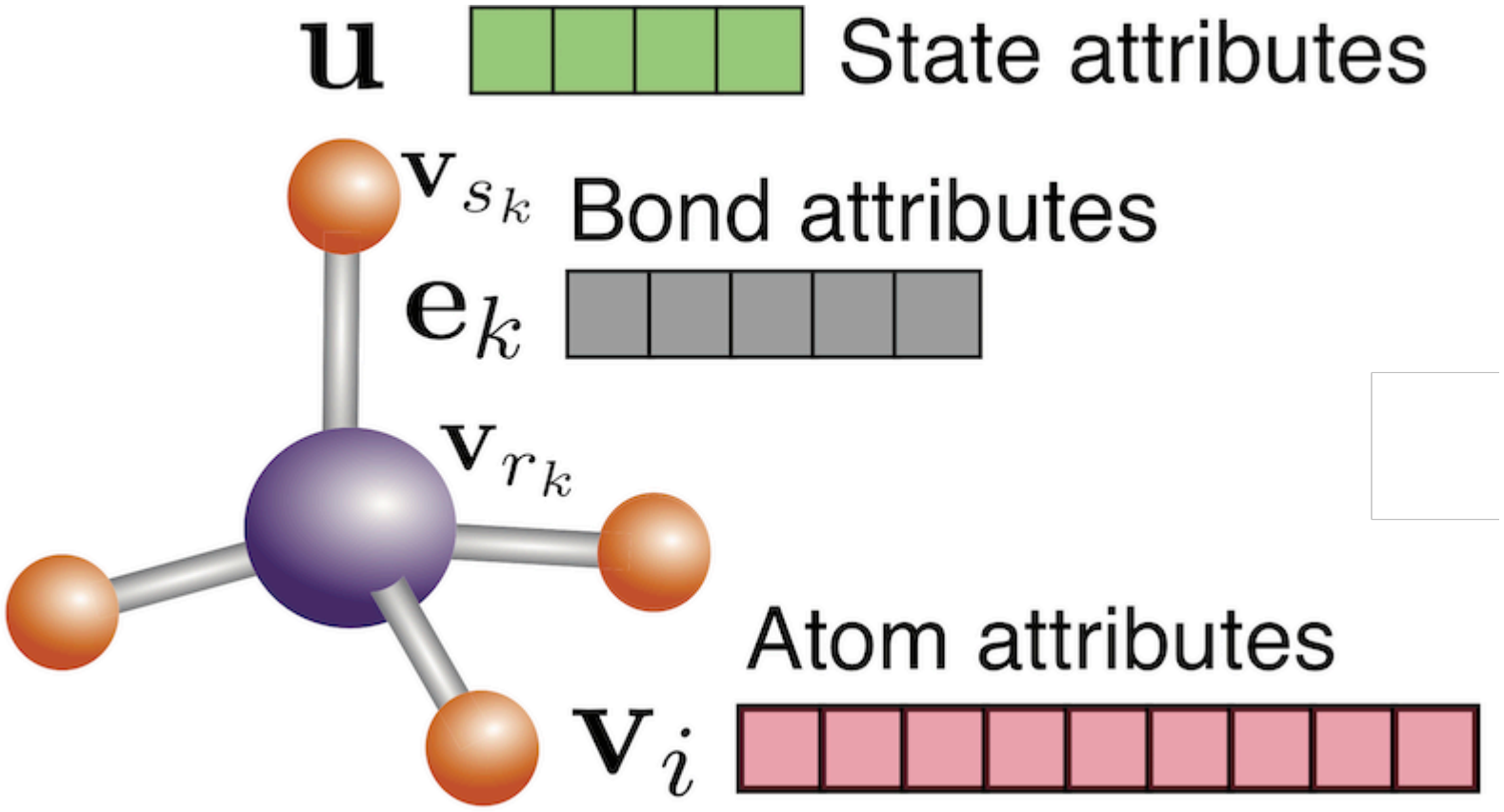
(a) Standard Neural Net



(b) After applying dropout.

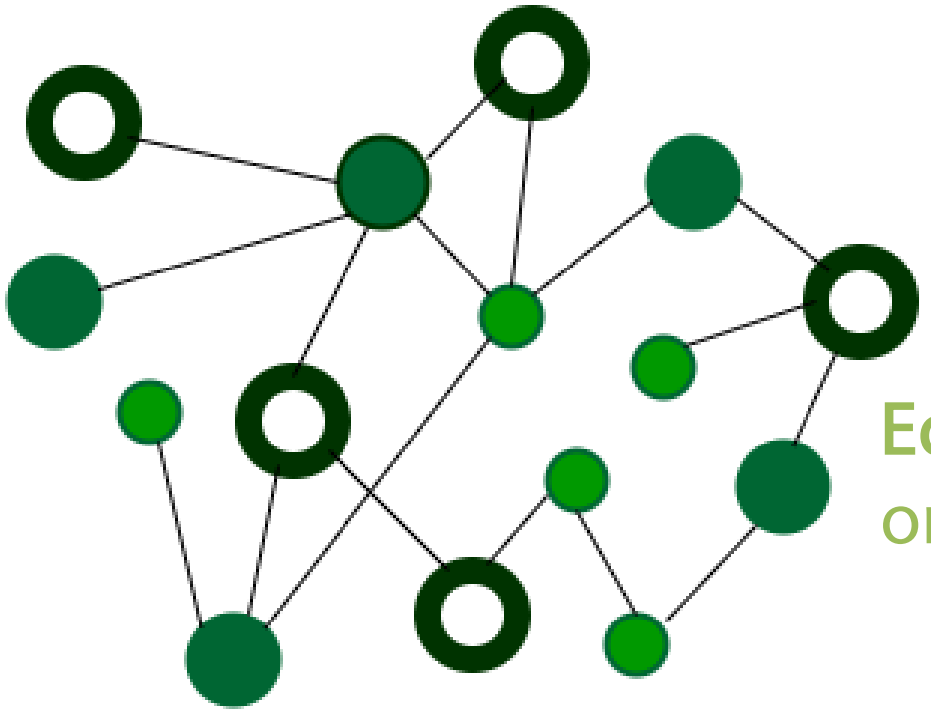
Features are becoming quite large for many GNNs

Features are becoming quite large for many GNNs



Is there a better, simpler graph representation?

Nodes: atoms with limited features
(quantum numbers, ground state energy)



Edges: bonds, limited connectivity, based on physically meaningful parameters

Graph: centrality, clustering coefficient, connectivity, motifs, symmetry

2 pt statistics