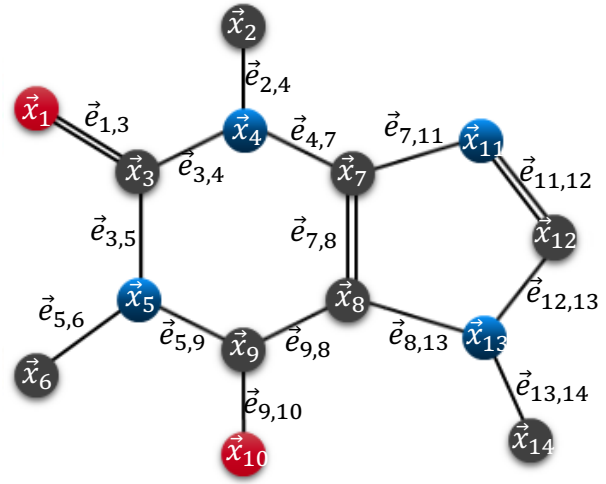
hhu
Heinrich Heine
Universität Düsseldorf

Graph neural Networks for small molecules

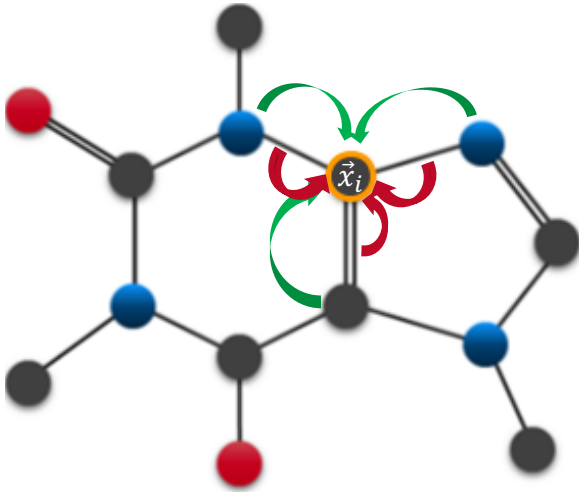
Representation of small molecules as graphs

- Atoms can be interpreted as the nodes of a graph and bonds as the edges of a graph
- All N atoms have to be numbered arbitrarily
- Node feature vectors $\vec{x}_i \in \mathbb{R}^{D_n}$ for node i are calculated and saved in a matrix $\mathrm{X} \in \mathbb{R}^{N\ x\ D_n}$
- Bond feature vectors $\vec{e}_{i,j} \in \mathbb{R}^{D_e}$ (bond between atom i and j) are calculated and saved in a tensor $\mathrm{E} \in \mathbb{R}^{N\ x\ N\ x\ D_n}$
- Adjacency matrix $A \in \mathbb{R}^{N\ x\ N}$ stores between which atoms a bond exists:
  - $A_{i,j} = 1 \ -> $ a bond exists between atom i and j
  - $A_{i,j} = 0 \ -> $ no bond exists between atom i and j
- X, E, and A are used as input for a graph neural network (GNN)



$$\vec{x}_i = \begin{pmatrix} atom\ type \\ mass \\ valence \\ charge \\ \dots \end{pmatrix} \qquad \vec{e}_{i,j} = \begin{pmatrix} bond\ type \\ part\ of\ ring \\ \dots \\ \dots \\ \dots \end{pmatrix}$$

Typical forward pipeline of a GNN



- First, the node feature vectors are iteratively updated T times by using information about neighboring atoms and edges:

$$\vec{x}_i = \vec{x}_i^{(0)} \rightarrow \vec{x}_i^{(1)} \rightarrow \ \dots \ \rightarrow \vec{x}_i^{(T)} \in \mathbb{R}^{D_n}$$

- Update for a single atom feature vector ($N(i)$ is the set of neighboring atoms):

$$\vec{x}_i^{(t+1)} = \sum_{j \in N(i)} U_t(\vec{x}_i^{(t)}, \ \vec{x}_j^{(t)}, \ \vec{e}_{i,j})$$

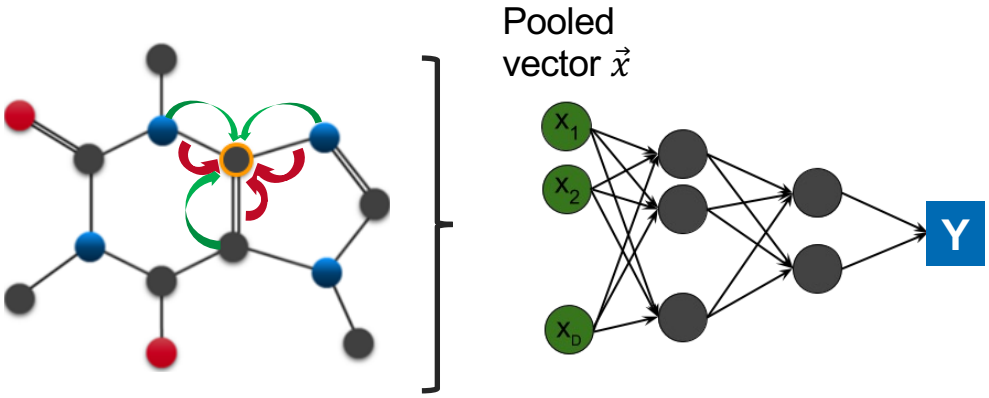
- After T update steps, we pool all feature vectors $\vec{x}_i$:

$$Pooling(\vec{x}_1^{(T)}, \vec{x}_2^{(T)}, \dots, \vec{x}_N^{(T)}) = \ \vec{x} \in \mathbb{R}^{D_n}$$

- Predicting a molecule/graph property:

$$FFN(\vec{x}) = y \in \mathbb{R}$$

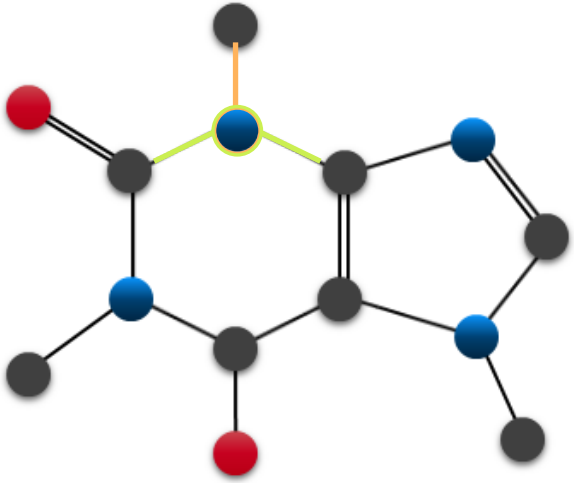
Training of GNNs



- Whole network is trained end-to-end
  - Parameters of the update functions and the FFN are adjusted simultaneously
  - Training is the same as for standard deep neural networks

- GNNs can also be pre-trained in a self-supervised way
  - For example: Masking nodes and prediction the type of node from the neighboring bond and node feature vectors

Directed Message Passing Neural Network (D-MPNN)



- Calculating atom feature vectors $\vec{x}_i \in \mathbb{R}^{D_n}$ and bond feature vectors $\vec{e}_{i,j} \in \mathbb{R}^{D_e}$
- Initial hidden representations of atom-bond-pairs:

$$\vec{h}_{i,j}^{(0)} = ReLU(W_h \cdot concat(\vec{x}_i, \vec{e}_{i,j}))$$

- Update hidden vectors T times
  - $t = 1$:

$$\vec{m}_{i,j}^{(1)} = \sum_{k \in N(i) \setminus \mathrm{j}} \vec{h}_{i,k}^{(0)} \qquad \vec{h}_{i,j}^{(1)} = ReLU(\vec{h}_{i,j}^{(0)} + W_m^1 \cdot \vec{m}_{i,j}^{(1)})$$

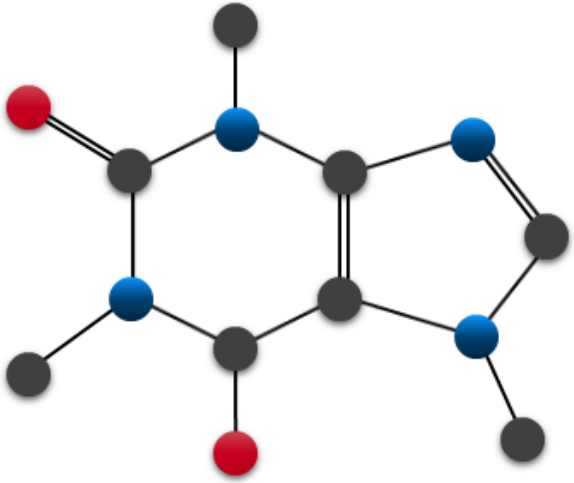
  - $t = 2$:

$$\vdots \qquad \cdots \qquad \cdots$$

  - $t = T$:

$$\vec{m}_{i,j}^{(T)} = \sum_{k \in N(i) \setminus \mathrm{j}} \vec{h}_{i,k}^{(T-1)} \qquad \vec{h}_{i,j}^{(T)} = ReLU(\vec{h}_{i,j}^{(0)} + W_m^T \cdot \vec{m}_{i,j}^{(T)})$$

Directed Message Passing Neural Network (D-MPNN)



- Initial hidden representations of node-bond-pairs:

$$\vec{h}_{i,j}^{(0)} = ReLU(W_h \cdot concat(\vec{x}_i, \vec{e}_{i,j}))$$

- Update hidden vectors T times
  - For $t = 1, \dots, T$:

$$\vec{m}_{i,j}^{(t)} = \sum_{k \in N(i) \setminus \mathrm{j}} \vec{h}_{i,k}^{(t-1)} \qquad \vec{h}_{i,j}^{(t)} = ReLU(\vec{h}_{i,j}^{(0)} + W_m^t \cdot \vec{m}_{i,j}^{(t)})$$

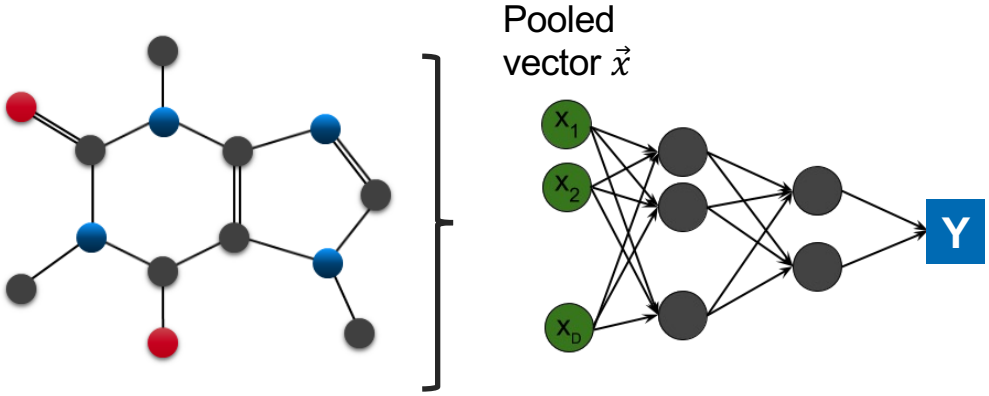
- Calculating atom representations for all atoms i:

$$\vec{m}_i = \sum_{k \in N(i)} \vec{h}_{i,k}^{(T)} \qquad \vec{h}_i = ReLU(W_a \cdot concat(\vec{x}_i, \vec{m}_i))$$

- Pooling all atom feature vectors $\vec{x} = \sum_{i=1,\dots,N} \vec{h}_i$
- Predicting a molecule/graph property:

$$FFN(\vec{x}) = y \in \mathbb{R}$$

After model training



- We can use this model as our final prediction model
- Alternatively, we can extract the vectors $\vec{x}$ for all molecules and train another ML model for the prediction task with these vectors as model input

Comparison to Transformer Networks

	BACE *RMSE*	Clearance *RMSE*	Delaney *RMSE*	Lipo *RMSE*	BACE *ROC*	BBBP *ROC*	ClinTox *ROC*	SR-p53 *ROC*
D-MPNN	2.253	49.754	1.105	1.212	0.812	0.697	**0.906**	0.719
ChemBERTa-2								
MTR-10M	1.417	48.934	**0.858**	**0.744**	0.783	0.733	0.601	0.827
MTR-77M	1.363	**48.515**	0.889	0.798	0.799	0.728	0.563	0.817

Dataset Tasks	BBBP 1	Tox21 12	ClinTox 2	HIV 1	BACE 1	SIDER 27
D-MPNN[54]	71.2	68.9	**90.5**	75.0	85.3	63.2
MOLFORMER-XL	**93.7**	84.7	**94.8**	82.2	88.21	**69.0**

- Potential reasons for superior performance of Transformer Networks:
  - Can encode stereochemistry of molecules
  - Pre-trained on millions of small molecules

Modified ChemBERTa-2 table: https://arxiv.org/pdf/2209.01712; Modified Molformer table: https://arxiv.org/pdf/2106.09553