DTU Compute
Department of Applied Mathematics and Computer Science

Molecular Property Prediction using Graph Neural Networks

Clara Hollenbeck (s214722), Hans Harck Tønning (s214700), Jakob Cetti Hansen (s214695), Mathias Herløv Lund (s204696)

1 Introduction

- Molecular properties are crucial in materials science and drug development.
- Traditional methods are slow and computationally expensive.
- Graph Neural Networks can be a more efficient alternatives.
- Polarizable Atom Interactive Neural Network (PaiNN) is a graph neural network designed for molecular property predictions.

2 Graph Neural Networks (GNNs)

GNNs use neural message passing, in which *messages* are passed between neighboring nodes to share information. A GNN takes a graph $G = (V, E)$ with a set of node features $\boldsymbol{X} \in \mathbb{R}^{F \times |V|}$ as input.

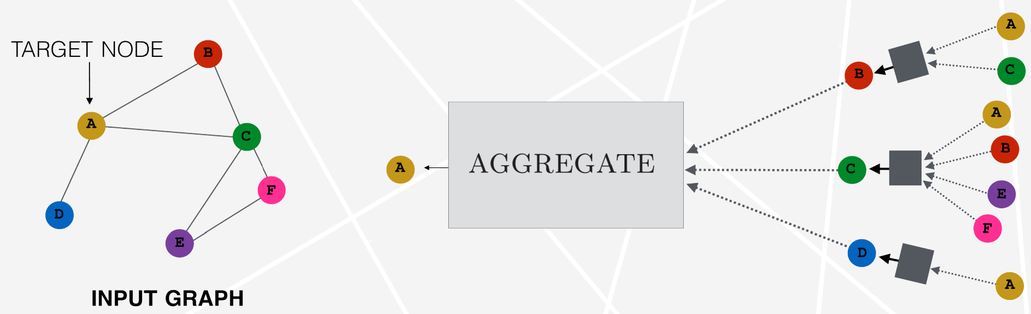

Figure 1: Illustration of message passing, figure from p. 49 in *Hamilton, William, Graph Representation Learning*.

For a node u, the initial node embedding are the node features $\mathbf{h}_u^{(0)} = \boldsymbol{x}_u \in \mathbb{R}^F$. For a iteration k, the new node embeddings $\mathbf{h}_u^{(k+1)}$ are computed using the message function M_k and the update function U_k:

$$\mathbf{m}_{N(u)}^{(k)} = M_k(\{\mathbf{h}_v^{(k)} : v \in N(u)\})$$
$$\mathbf{h}_u^{(k+1)} = U_k(\mathbf{h}_u^{(k)}, \mathbf{m}_{N(u)}^{(k)}).$$

Both the update function U_k and the message function M_k are learned functions which can be shared or be different for each message passing iteration k. After K iterations, the final node embedding is $\mathbf{z}_u = \mathbf{h}_u^{(K)} \in \mathbb{R}^F$, which is the output of the GNN.

3 GNNs of molecular data

Molecules can be represented as a graph, with each atom as a node, and each chemical bond between two atoms as an edge.

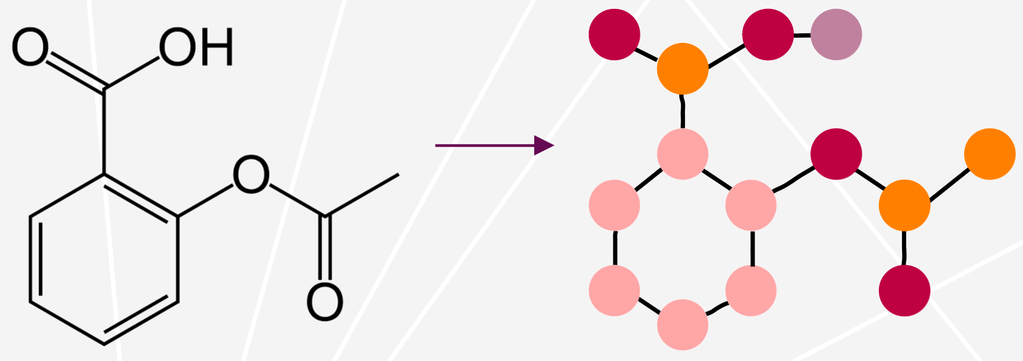

Figure 2: Illustration of a molecule as a graph.

Chemical interaction decreases with distance, therefore we can use the distance $\|r_{ij}\|$ between atom i and j with a cutoff distance of r_{cut} to determine if there should be an edge instead of looking at the chemical bonds.

Molecules can have different rotations, therefore rotational equivariant (or alternatively rotational invariant) is an important property of our GNN. To ensure rotational equivariance, the message function M_k and update function U_k needs to be restricted to rotationally equivariant operations like

- Any (nonlinear) function of scalars: $f(\boldsymbol{s})$
- Linear combinations of equivariant vectors: $\boldsymbol{W}\boldsymbol{v}$
- Scalar products: $\boldsymbol{s} = \|\boldsymbol{v}\|^2$, $s = \langle \boldsymbol{v}_1, \boldsymbol{v}_2 \rangle$

4 PaiNN

PaiNN uses rotationally equivariant message-passing layers to efficiently propagate directional information. This allows the network to capture spatial relationships that rotationally invariant layers cannot.

The structure of PaiNN alternates between message-passing and update blocks, followed by a multilayer perceptron before the post-processing stage.

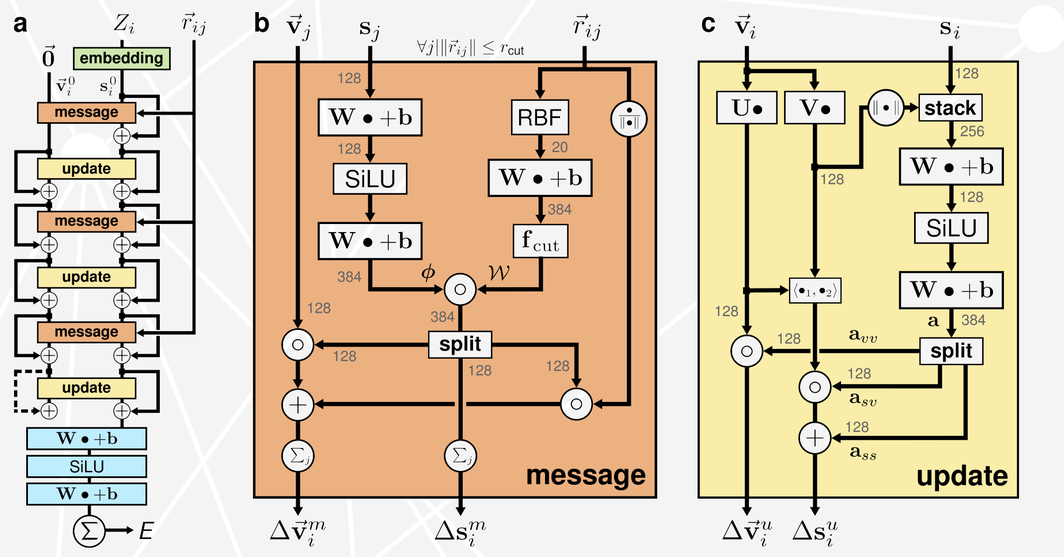

Figure 3: The PaiNN Architecture with our modifications to the update layer whose input indices were incorrect, and a connection was missing between $U\bullet$ and the inner-product.

We used early stopping and learning rate scheduling in our training of the PaiNN network to prevent over-fitting. PaiNN has high variance in the validation loss, therefore we exponential smooth the validation loss when used in early stopping and learning rate scheduling.

5 Layer optimization

The number of message passing layers is a key hyperparameter in the PaiNN architecture. Each layer involves complex operations (e.g., MLP and RBF updates), making the network deep and prone to vanishing gradients and over-smoothing when too many layers are used, while too few limit feature propagation. We experimented with training models with 1–10 layers.

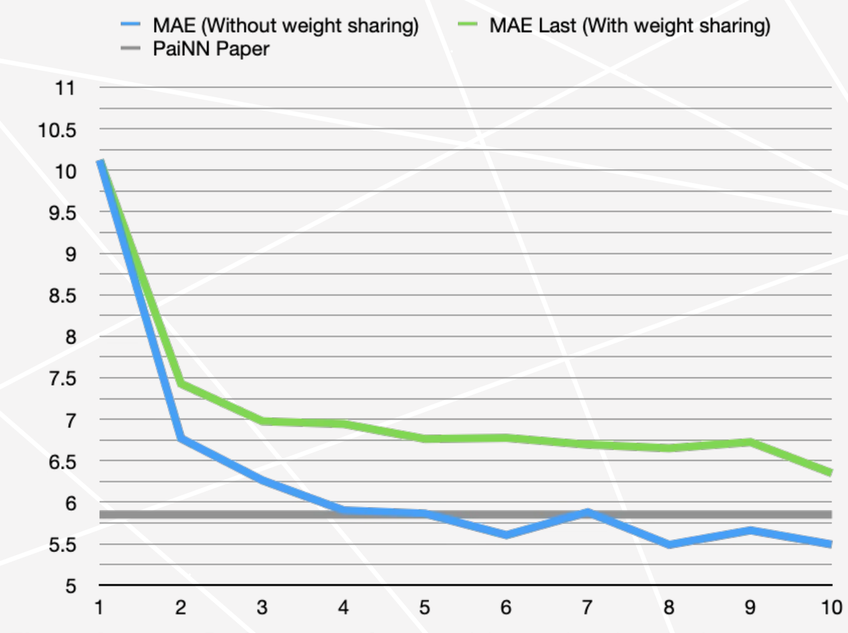

Figure 4: Performance of our model at estimating the internal energy at zero Kelvin with varying numbers of message parsing layers on the X-axis and MAE (Mean Absolute Error) on the Y-axis. The MAE from the PaiNN paper has been included as a baseline.

The Mean Absolute Errors (MAE) of our unseen test dataset can be seen in Figure 4. We see that we surpass the PaiNN article's MAE of 5.85 for U_0, achieved with 3 layers, when using 6, 8, 9, and 10 layers.

In general, we see that the MAE decreases, as the number of message passes increases, without any of the expected problems such as over smoothing occurring.

The same test was also conducted with weight sharing between all the message passing and update layers. In this version of the model, the MAE is worse, but not significantly when comparing it to the much smaller parameter space.