# Graph Nets for Learning Molecular Physics

{yuanqing.wang, josh.fass, chaya.stern, john.chodera*}
@choderalab.org

oct 14, 2019

# one-minute version in case you are really busy

Graph Net—a network that operates on the topological space of molecules and consists of three update and three aggregation functions —is capable of:

predicting:

- per-molecule attributes: energy, solubility, biophysical properties, etc;
- per-atom attributes: charges;
- per-bond attributes: Wiberg bond order;
- forcefield parameters

# part 0
# what is Graph and what is Graph Net?

# Graph

proteins and molecules could be modeled as

- undirected,
- node-, edge-, and graph-attributed,
- unlabeled,

graphs.

$$\mathcal{G} = \{\mathcal{E}, \mathcal{V}, \mathcal{U}\}$$

# Graph Nets

$$\mathcal{G} = \{\mathcal{E}, \mathcal{V}, \mathcal{U}\}$$

$$\mathbf{e}'_k = \phi^e(\mathbf{e}_k, \mathbf{v}_{rk}, \mathbf{v}_{sk}, \mathbf{u}) \qquad \bar{e}'_i = \rho^{e\to v}(E'_i)$$

$$\mathbf{v}'_i = \phi^v(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}) \qquad \bar{e}' = \rho^{e\to u}(E')$$

$$\mathbf{u}' = \phi^u(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}) \qquad \bar{\mathrm{v}}' = \rho^{v\to u}(V')$$

Battaglia et al.(2018) arXiv:1806.012

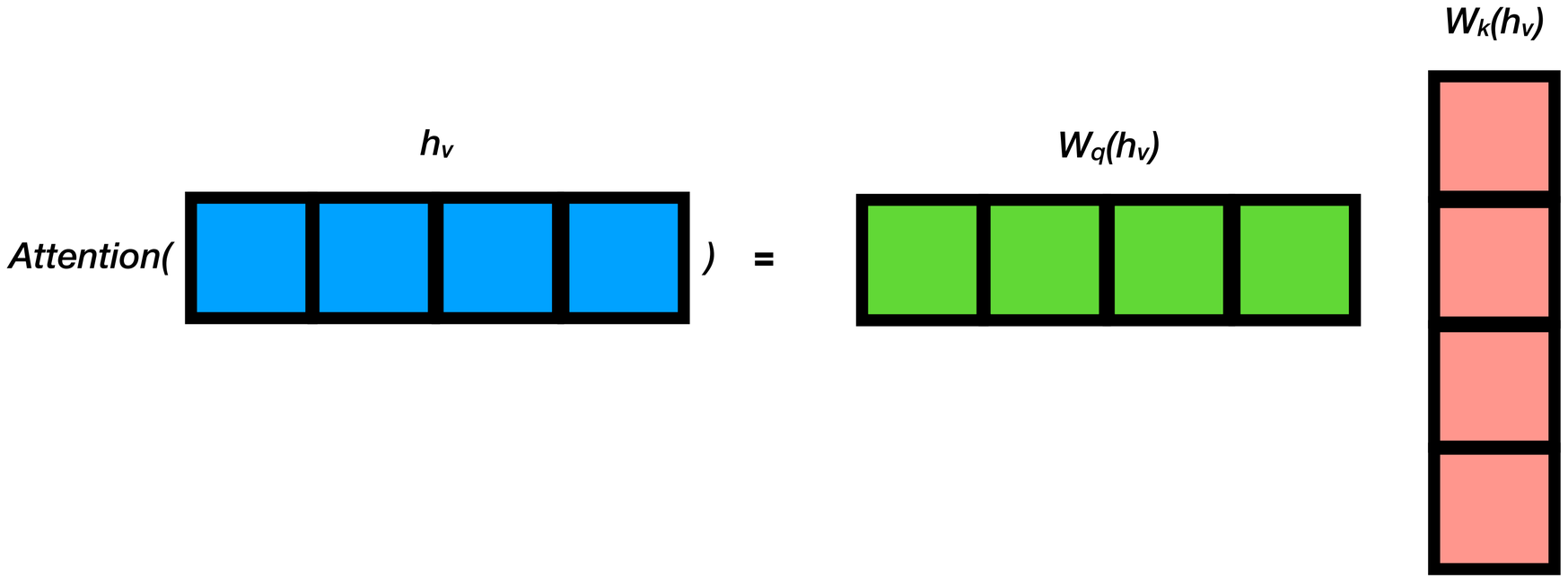
# hyperedges

$$
\begin{aligned}
\mathbf{a}_i^{(t+1)} &= \phi^a(\mathbf{e_{01}}^{(t)}, \mathbf{e_{02}}^{(t)}, \mathbf{v_0}^{(t)}, \mathbf{v_1}^{(t)}, \mathbf{v_2}^{(t)}, \mathbf{u}^{(t)});\\
\mathbf{d}_i^{(t+1)} &= \phi^d(\mathbf{e_{23}}^{(t)}, \mathbf{e_{12}}^{(t)}, \mathbf{e_{34}}^{(t)}, \mathbf{v_1}^{(t)}, \mathbf{v_2}^{(t)}, \mathbf{v_3}^{(t)}, \mathbf{v_4}^{(t)}, \mathbf{u}^{(t)});\\
\bar{\mathbf{a}}^{(t+1)} &= \rho^{a\to u}(A^{(t)});\\
\bar{\mathbf{d}}^{(t)} &= \rho^{d\to u}(D^{(t)}),
\end{aligned}
$$

# pairwise readout

$$\bar{h}_{\text{pairwise}} = f_r(\text{Attention}(h_v, h_v))$$

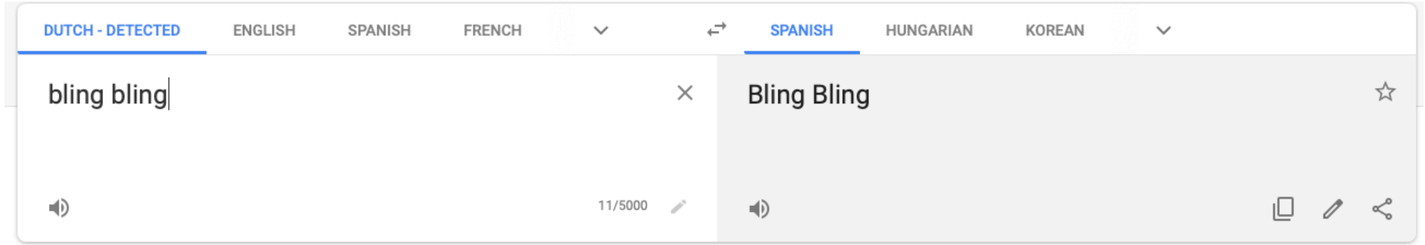
$$\text{Attention}(h_v, h_v) = W_k(h_v) W_q(h_v)^T$$

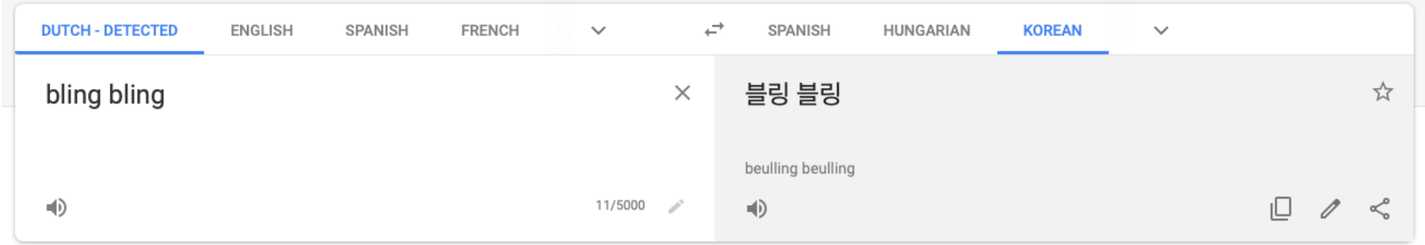


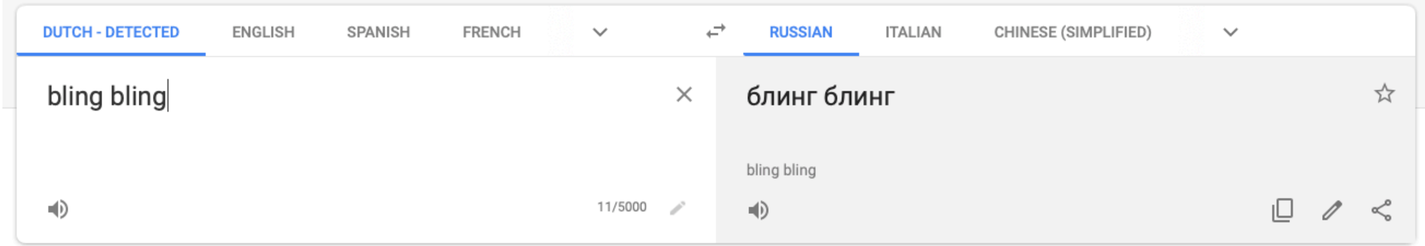
***hypothesis*:**
**since Google claims ‘attention is all you need’, they use only attention in their translation app.**

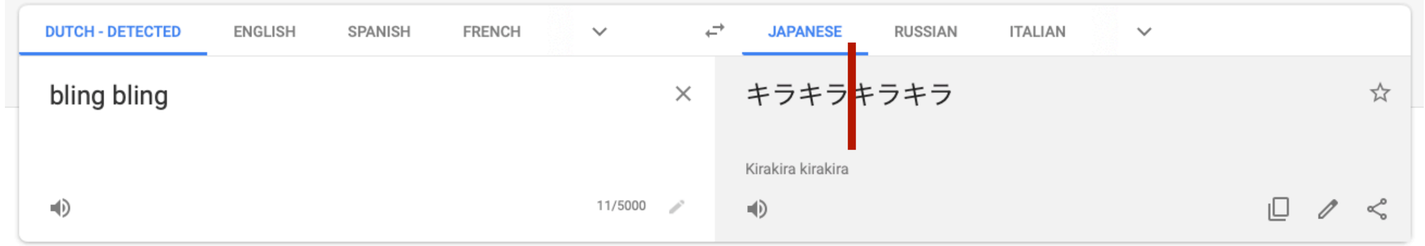
***reasoning*:**
**attention is permutation equivariant, hence no symmetric phrase in any given language should be translated into an asymmetric phrase in another language.**

***experiment:***





**symmetry broken!!!**

**hypothesis proven to be incorrect**

# hypergraph

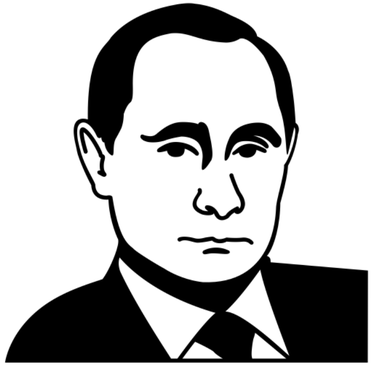
**inspired by forcefields:**

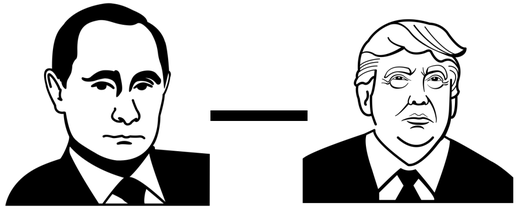
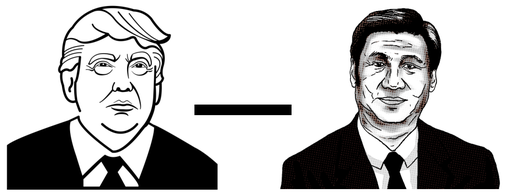
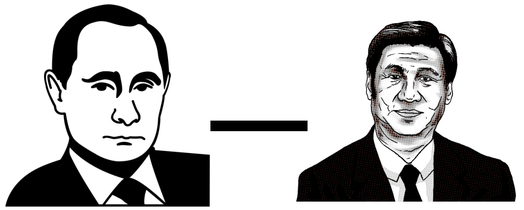
$$E_{\text{tot}} = E_{\text{bond}} + E_{\text{angle}} + E_{\text{torsion}} + E_{\text{pairwise}}$$

$$h_u = h_v + h_e + h_a + h_d + h_{\text{pairwise}}$$

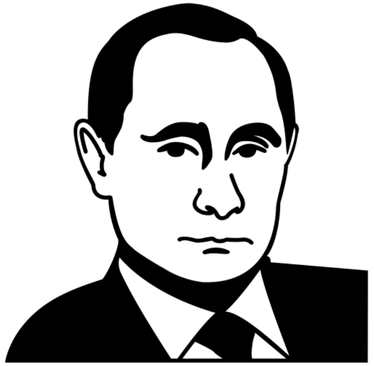
“a graph plays a double role:
it is both the input of the system and captures the network topology of the distributed system that solves the problem”

–Andreas Loukas (2019) arXiv:1907.03199

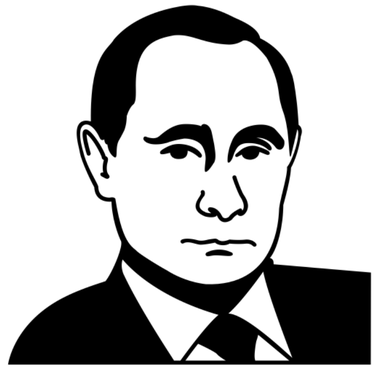


$$\mathcal{G} = \{\{\{\text{  }, \text{  }, \text{  }\}, \mathcal{V}\}, \{\{\text{  }, \text{  }, \text{  }\}, \mathcal{E}\}, \text{  }, \mathcal{U}\}$$

$$\mathbf{e}'_k = \phi^e(\mathbf{e}_k, \mathbf{v}_{rk}, \mathbf{v}_{sk}, \mathbf{u})$$

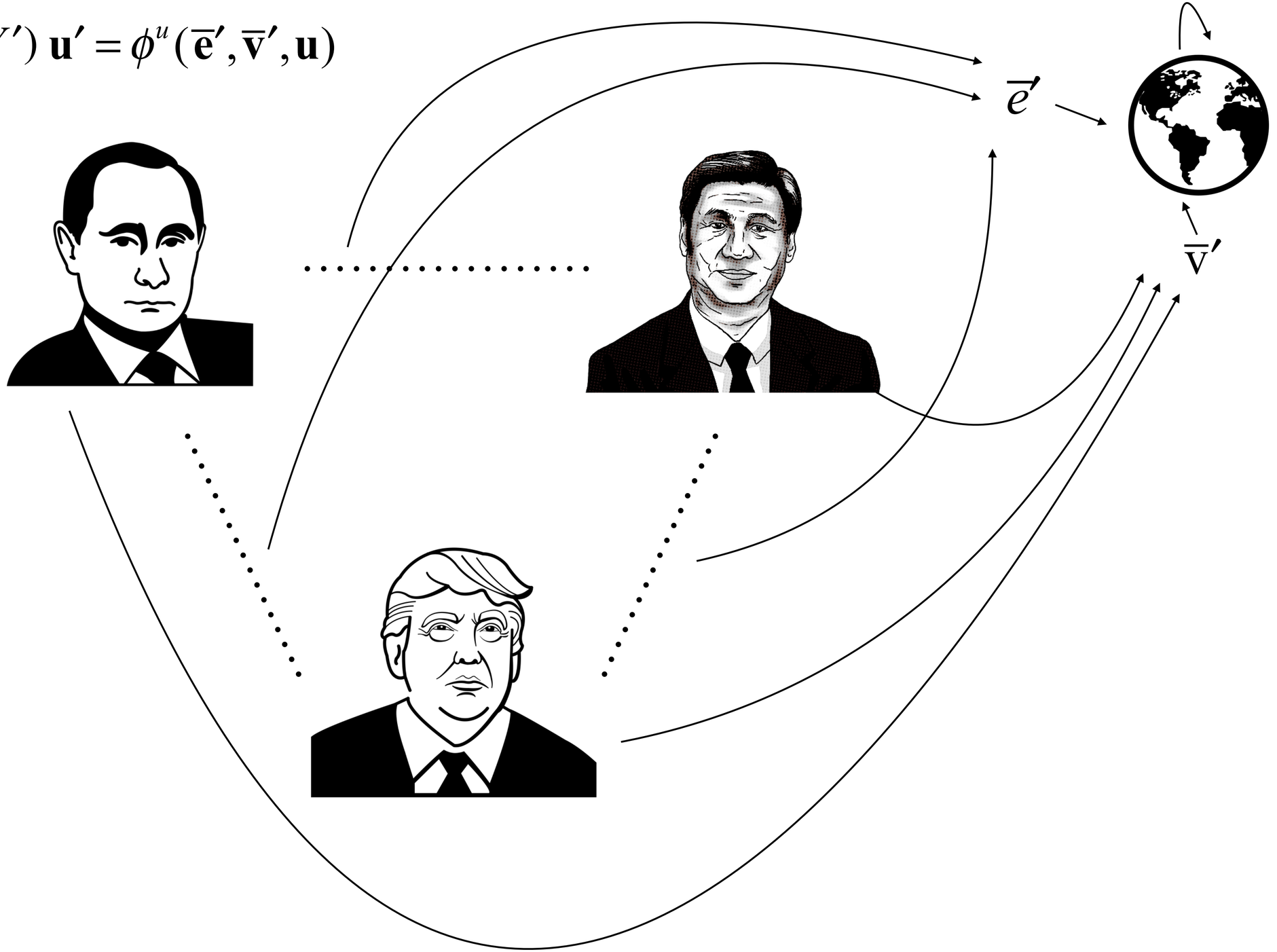


$$\vec{e}'_i = \rho^{e \to v}(E'_i) \qquad \mathbf{v}'_i = \phi^{v}(\vec{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u})$$



$\vec{e}'_i$

$$\overline{e}' = \rho^{e\to u}(E')\ \overline{\mathrm{v}}' = \rho^{v\to u}(V')\ \mathbf{u}' = \phi^{u}(\overline{\mathbf{e}}', \overline{\mathbf{v}}', \mathbf{u})$$



**Algorithm 1** Steps of computation in a full GN block.

**function** GRAPHNETWORK($E$, $V$, $\mathbf{u}$)

**for** $k \in \{1 \dots N^e\}$ **do**

$\mathbf{e}'_k \leftarrow \phi^e\left(\mathbf{e}_k, \mathbf{v}_{r_k}, \mathbf{v}_{s_k}, \mathbf{u}\right)$ ▷ 1. Compute updated edge attributes

**end for**

**for** $i \in \{1 \dots N^n\}$ **do**

**let** $E'_i = \{(\mathbf{e}'_k, r_k, s_k)\}_{r_k=i,\ k=1:N^e}$

$\bar{\mathbf{e}}'_i \leftarrow \rho^{e\to v}\left(E'_i\right)$ ▷ 2. Aggregate edge attributes per node

$\mathbf{v}'_i \leftarrow \phi^v\left(\bar{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}\right)$ ▷ 3. Compute updated node attributes

**end for**

**let** $V' = \{\mathbf{v}'\}_{i=1:N^v}$

**let** $E' = \{(\mathbf{e}'_k, r_k, s_k)\}_{k=1:N^e}$

$\bar{\mathbf{e}}' \leftarrow \rho^{e\to u}\left(E'\right)$ ▷ 4. Aggregate edge attributes globally

$\bar{\mathbf{v}}' \leftarrow \rho^{v\to u}\left(V'\right)$ ▷ 5. Aggregate node attributes globally

$\mathbf{u}' \leftarrow \phi^u\left(\bar{\mathbf{e}}', \bar{\mathbf{v}}', \mathbf{u}\right)$ ▷ 6. Compute updated global attribute

**return** $(E', V', \mathbf{u}')$

**end function**

https://github.com/choderalab/gimlet/blob/master/gin/probabilistic/gcn.py

Battaglia et al.(2018) arXiv:1806.012


Graph Inference on MoLEcular Topology github.com/choderalab/gimlet

- `gin/` the core (and fun) part of the package.
  - `i_o/` reading and writing popular molecule embedding/representing structures.
  - `deterministic/` property predictions, conformer and charge generations.
  - `probabilistic/` molecular machine learning through graph networks.
- `lime/` auxiliary scripts.
  - `for_biologists/` ready-to-use modules and scripts.
  - `architectures/` off-the-shelf model architectures developed elsewhere.
  - `scripts/` fun scripts we used to generate data and hypothesis.
  - `trained_models/` *Nomen est omen*.

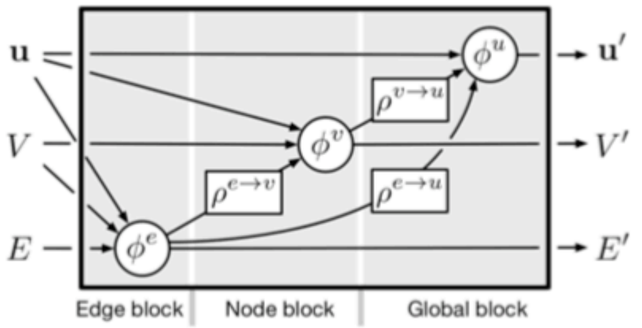
# popular choice of functions: trainable Neural Networks

$$\phi^e = \mathrm{NN}_e$$

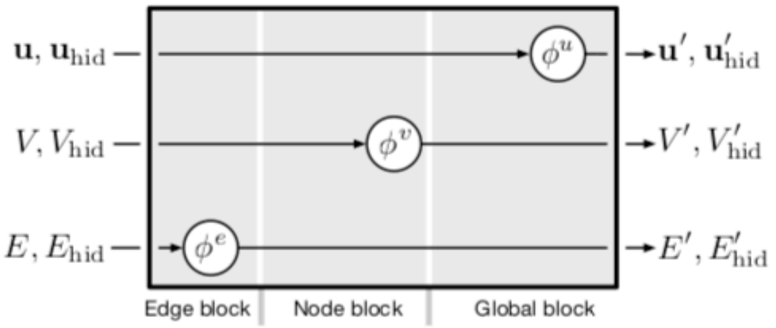
$$\phi^v = \mathrm{NN}_v$$

$$\phi^u = \mathrm{NN}_u$$

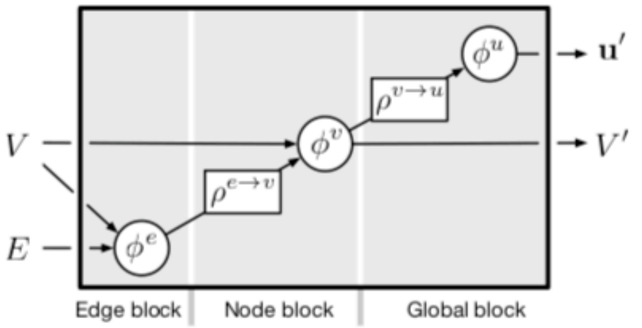
$$\rho^{e\to v} = \rho^{v\to u} = \rho^{e\to u} = \sum$$



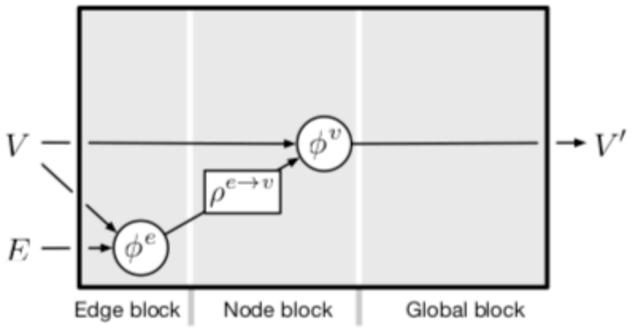

(a) Full GN block



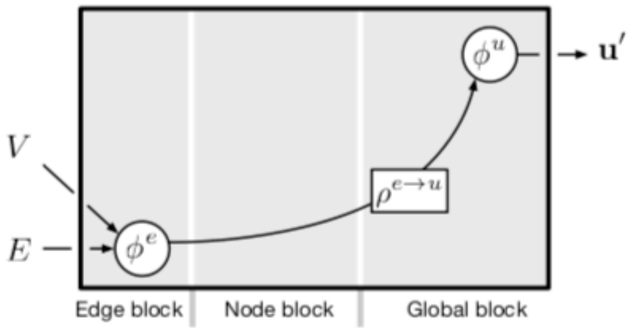

(b) Independent recurrent block



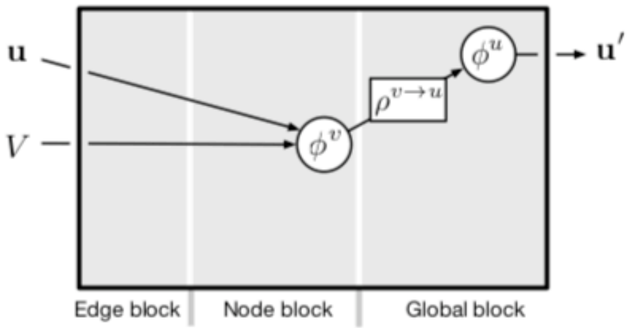

(c) Message-passing neural network




(d) Non-local neural network



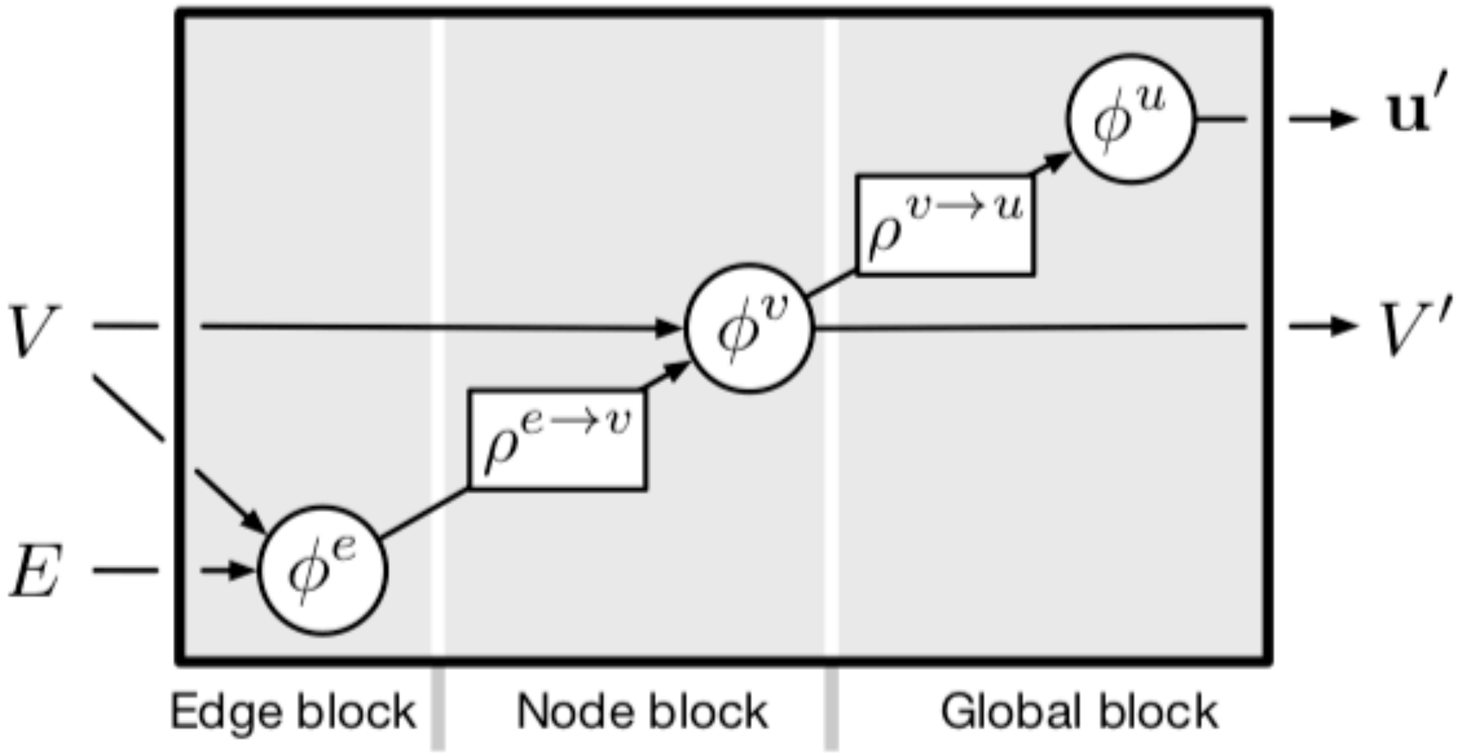

(e) Relation network




(f) Deep set

Battaglia et al.(2018) arXiv:1806.012

# Message Passing Neural Nets (MPNN)



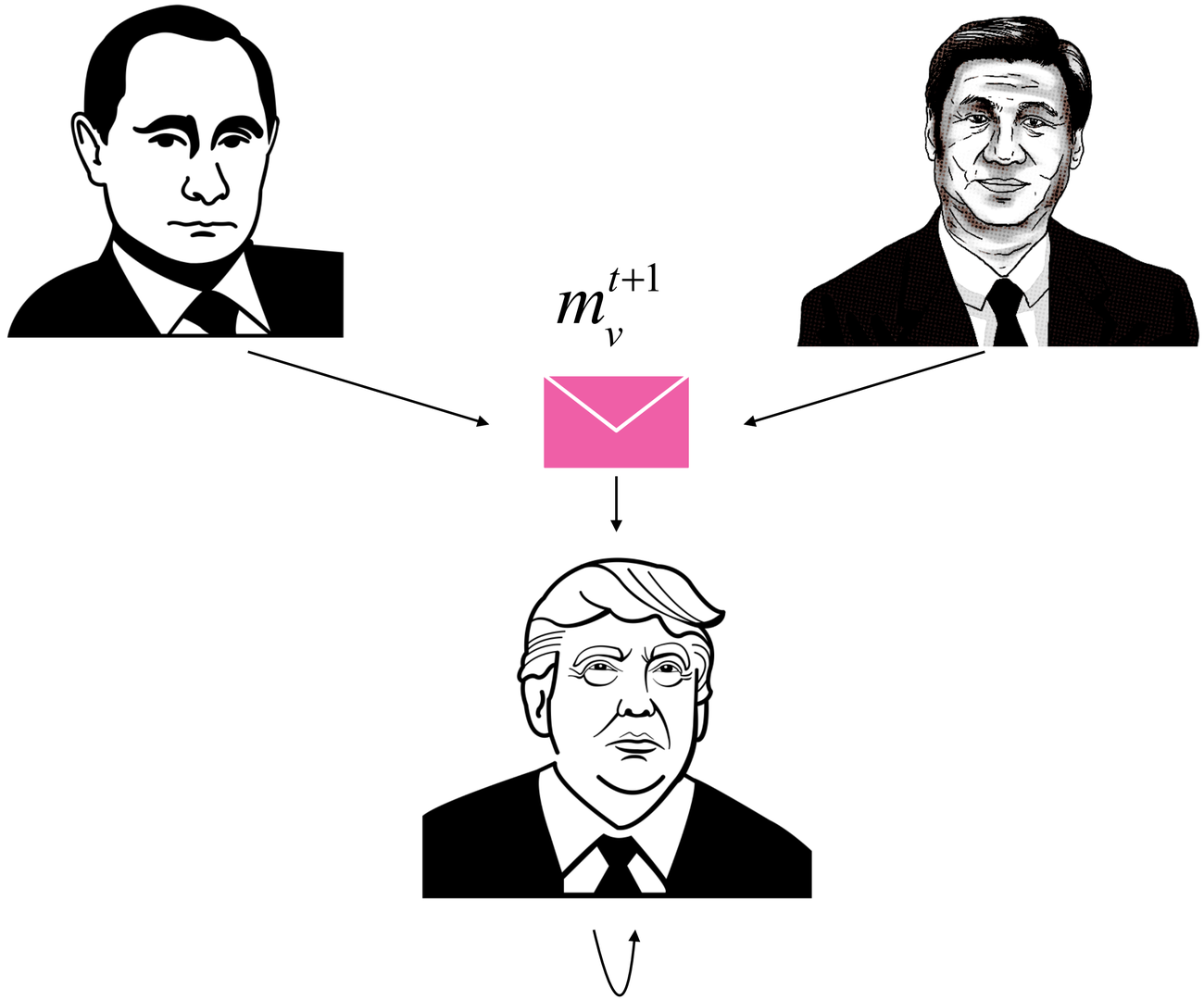

(c) Message-passing neural network

Battaglia et al.(2018) arXiv:1806.012

# Message Passing Neural Nets (MPNN)

$$\overline{e}'_i = \rho^{e \to v}(E'_i) = \sum_{w \in N(v)} M_t(h_v^t, h_w^t, e_{vw}) = m_v^{t+1}$$

$$\mathbf{v}'_i = \phi^v(\overline{\mathbf{e}}'_i, \mathbf{v}_i, \mathbf{u}) = U_t(h_v^t, m_v^{t+1}) = h_v^{t+1}$$


$m_v^{t+1}$

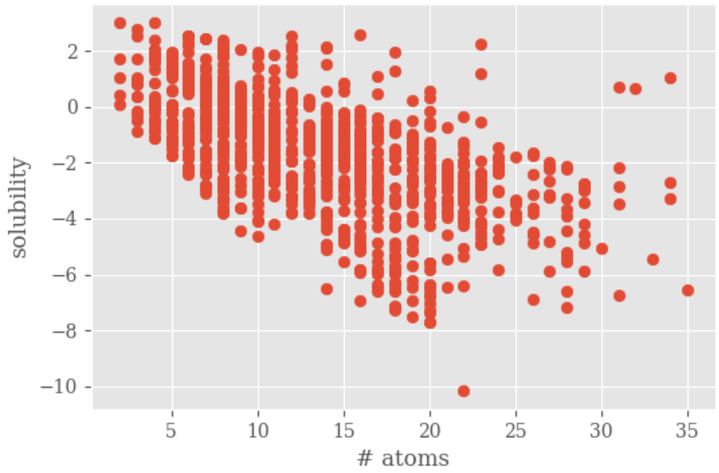
# Message Passing Neural Nets (MPNN)

| | Convolutional<br>Duvenaud<br>et al. (2015) | Gated<br>Li<br>et al. (2016) | Interaction<br>Battaglia<br>et al. (2016) | Deep Tensor<br>Shutt<br>et al. (2017) |
|---|---|---|---|---|
| $U_t(h_v^t, m_v^{t+1})$ | (.,.) | $\mathrm{GRU}(h_v^t, m_v^{t+1})$ | (.,.) | + |
| $M_t(h_v^t, h_w^t, e_{vw})$ | $(\sum h_w^t, \sum e_{vw})$ | $A_{e_{vw}} h_w^t$ | FC | FC |
| $R$ | $f(\sum_{v,t} \mathrm{softmax}(W_t h_v^t))$ | FC | $f(\sum_{v \in \mathcal{G}} h_v^T)$ | FC |
| citation | arXiv:1509.09292 | arXiv:1511.05493 | arXiv:1612.00222 | 10.1038/<br>ncomms13890 |

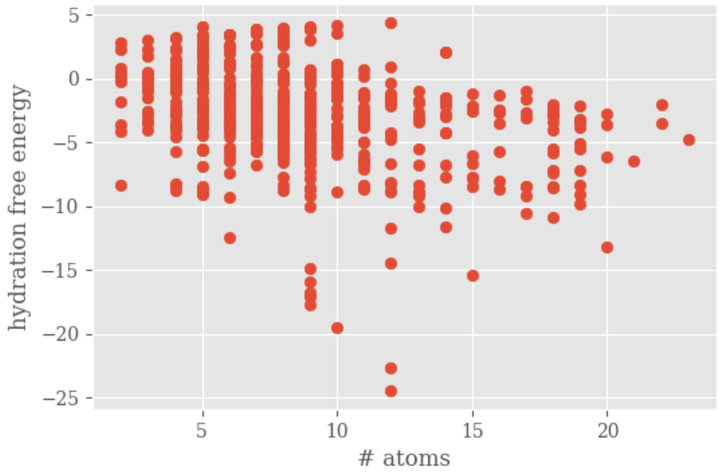
# part 1
# discriminative models: per-graph attributes

# results of per-molecule task

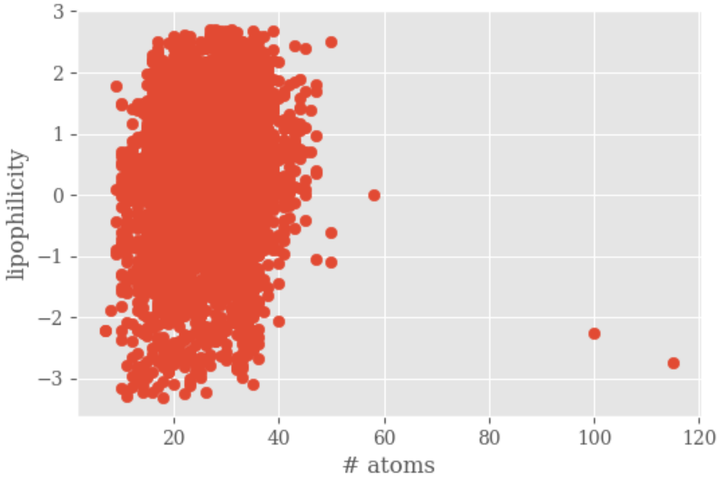
| dataset | $R^2$ of GIMLET | RMSE of GIMLET | SOTA | $R^2$ of SOTA | RMSE of SOTA |
|---|---|---|---|---|---|
| **ESOL** | 0.8682 | 0.5372 | MPNN | 0.939 | 0.58 |
| **SAMPL** | 0.9537 | 0.7388 | MPNN | 0.923 | 1.15 |
| **Lipophilicity** | (mean agg.) 0.5178 | (mean agg.) 0.6990 | GC | 0.655 | 0.662 |
| | (sum agg.) 0.3493 | (sum agg.) 0.9432 | | | |




**ESOL: Water solubility data(log solubility in mols per litre) for common organic small molecules**




**FreeSolv: Experimental and calculated hydration free energy of small molecules in water.**




**Lipophilicity: Experimental results of octanol/water distribution coefficient(logD at pH 7.4).**

To answer this question, we prepared two toy task:

- molecule weight: extensive
- mean atom weight: intensive

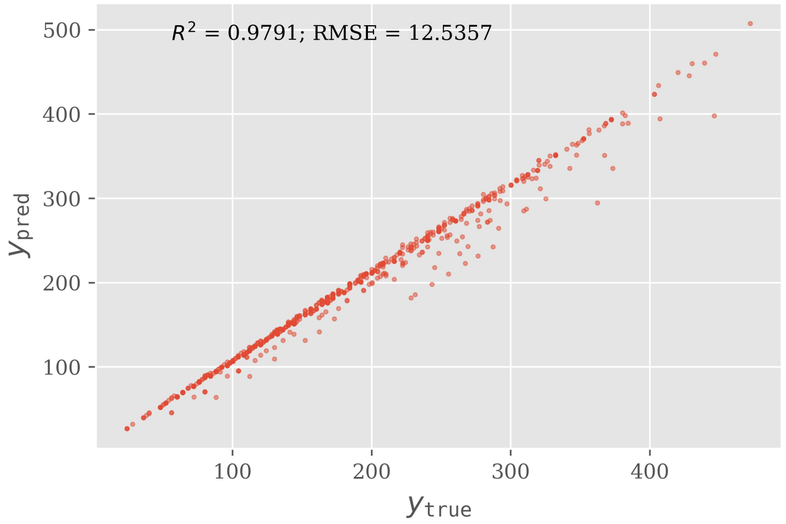
with molecules in ESOL dataset.

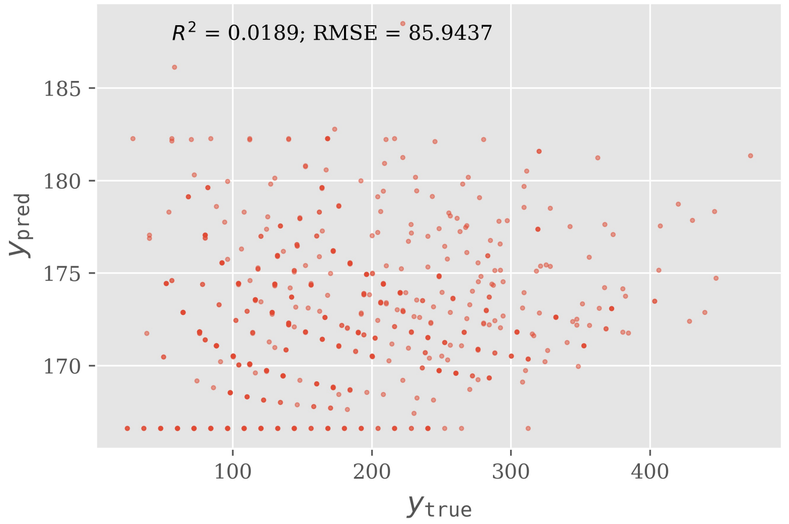
is there an association between the sum in hidden space and the sum in physical space?

## sum aggregation

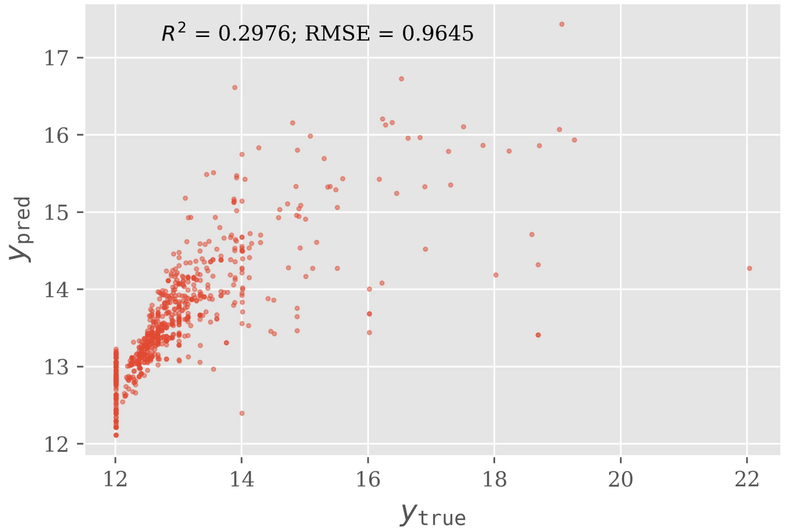
## mean aggregation

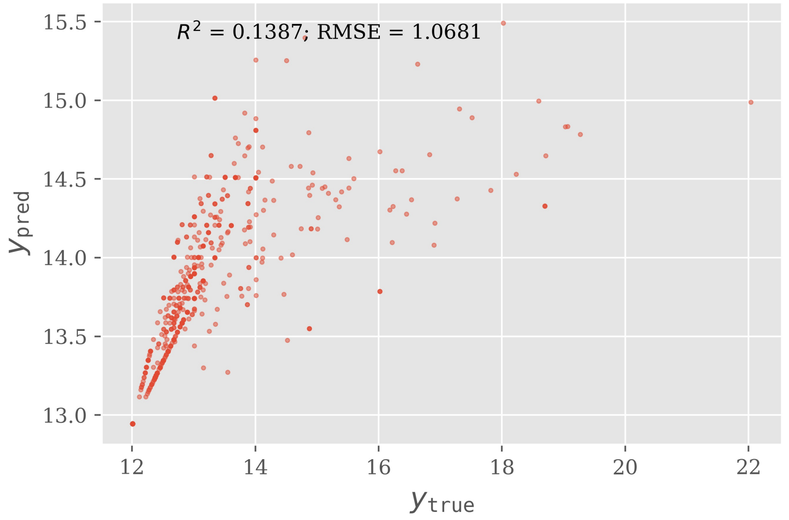
**molecule weight**




**mean atom weight**



# invariance and equivariance

For every graph G and every permutation matrix P, we call function f

invariant if

$$f(\mathbf{P} \star \mathcal{G}) = f(\mathcal{G})$$

and equivariant if

$$f(\mathbf{P} \star \mathcal{G}) = \mathbf{P} \star f(\mathcal{G})$$

# invariance and equivariance

The following conditions are sufficient for an operation on a graph to be invariant for per-graph attributes and equivariant for per-node and per-edge attributes:

- perform on an unlabelled graph or discard node and edge label at readout level

- perform on nodes and edges in synchronous manner

# is invariance and equivariance always a good thing?

"cycle graph"

task: predicting averaged atom weight (12) and molecule weight (12n)

for all edges and all nodes in all graphs this collection:

- at t=0, they are initialized to have the same attributes. $h_{vi}=h_{vj}$ for all i, j; $h_{ei} = h_{ej}$ for all i,j. We call this state of such set locally isomorphic.

- if at t=T, the set is locally isomorphic, and we update $h_v$ and $h_e$ by:

$$\mathbf{e}'_k = \phi^e(\mathbf{e}_k, \mathbf{v}_{rk}, \mathbf{v}_{sk})$$

$$\mathbf{v}'_i = \phi^v(\bar{\mathbf{e}}'_i, \mathbf{v}_i)$$

- then the set is locally isomorphic at t=T+1.

hence this set is locally isomorphic for all t.

To get the final readout,  if we apply a sum function, then the graphs in this set have different values; if we apply a mean function, then the graphs in this set have same value.

Therefore sum aggregation function can only be used to predict molecule weight but not atom weight, mean can only be used to predict mean atom weight but not molecule weight.

# GNs are powerful when  performed on labeled graphs

| *problem* | *bound* | *problem* | *bound* |
|---|---|---|---|
| cycle detection (odd) | $dw = \Omega(n/\log n)$ | shortest path | $d\sqrt{w} = \Omega(\sqrt{n}/\log n)$ |
| cycle detection (even) | $dw = \Omega(\sqrt{n}/\log n)$ | maximum independent set | $dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$ |
| subgraph verification* | $d\sqrt{w} = \Omega(\sqrt{n}/\log n)$ | minimum vertex cover | $dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$ |
| minimum spanning tree | $d\sqrt{w} = \Omega(\sqrt{n}/\log n)$ | chromatic coloring | $dw = \Omega(n^2/\log^2 n)$ for $w = O(1)$ |
| minimum cut | $d\sqrt{w} = \Omega(\sqrt{n}/\log n)$ | girth 2-approximation | $dw = \Omega(\sqrt{n}/\log n)$ |
| diameter estimation | $dw = \Omega(n/\log n)$ | diameter 3/2-approximation | $dw = \Omega(\sqrt{n}/\log n)$ |

*Table 1: Summary of main results. Sugraph verification* entails verifying one of the following predicates for a given subgraph $H$ of $G$: is connected, contains a cycle, forms a spanning tree of $G$, is bipartite, cuts $G$, is an $s$-$t$ cut of $G$. All problems are defined in Appendix A.*

Loukas  (2019) arXiv:1907.0319

# part 2
# discriminative models: per-node and per-edge attributes

# Weisfeiler-Lehman Test

Iteratively:

- aggregates the labels of nodes and their neighborhoods,
- hashes the aggregated labels into unique new labels

Weisfeiler and Lehman (1968)

# GNs could be as powerful as WL test

Xu et al. (2019) Theorem 3.

Let A: G -> $R^d$ be a GNN. With a sufficient number of GNN layers, A maps any graphs that the Weisfeiler-Lehman test of isomorphism decides as non-isomorphic, to different embeddings if the following conditions hold:

a) A aggregates and updates node features iteratively with

$$h_v^{(k)} = \Phi(h_v^{(k-1)}, f(h_u^{(k-1)} : \{u \in \mathcal{N}(v)\}))$$

where the functions f, which operates on multisets, and Φ are injective.

b) A's graph-level readout, which operates on the multiset of node features $\{h_v^{(k)}\}$, is injective.

Xu et al. (2019) arXiv:1810.00826

# automorphic equivalence test

- Two vertices are automorphically equivalent if all the vertices can be re-labeled to form an isomorphic graph with the labels of u and v interchanged.
- We can test automorphic equivalence through WL-like test, where we iteratively
  - aggregates the attribute of nodes and their neighborhoods,
  - hashes the aggregated attribute into ~~unique~~ new attribute

# per-atom attributes: charges

Since charges of atoms are determined by the chemical environment thereof, we hypothesize that two atoms that are not automorphically equivalent have different charges, and thus could be distinguished by graph nets.

- Q: why we need a new charging method for Molecular Dynamics simulation?

- A: charging is critical for MD but current methods suck as they are either expensive (QM) or unreliable (empirical).

# per-atom attributes: charges

Dataset:

- Bleiziffer et al. (2018) Density functional theory.
- In-house dataset: generated by AM1-BCC ELF (Electrostatically Least-interacting Functional) method. Considered to be invariant w.r.t. conformation.

We can find such $\{q_i\}$ by minimizing the error between predicted and reference charges

$$\{\hat{q}\} = \underset{\{q_i\}}{\operatorname{argmin}} \sum_i RMSE(q_i, q_{i0})$$

subject to

$$\sum_i q_i = \sum_i q_{i0}$$

Define the contribution of potential energy by atomic charge as $E_A(Q)$. It has been shown that the second-order Taylor expansion is sufficient to approximate.

$$E_A(Q) \approx E_{A0} + Q_A(\frac{\partial E}{\partial Q})_{A0} + \frac{1}{2}Q_A^2(\frac{\partial^2 E}{\partial Q^2})_{A0}$$

the first- and second-order derivates are termed electronegativity and hardness.

$$e_A \equiv (\frac{\partial E}{\partial Q})_{A0} \approx \frac{1}{2}(E_A(+1) - E_A(-1)) = \frac{1}{2}(\mathrm{IP} + \mathrm{EA})$$

$$s_A \equiv J_{AA}^0 \equiv (\frac{\partial^2 E}{\partial Q^2})_{A0} \approx E_A(+1) + E_A(-1) - 2E_A(0) = \mathrm{IP} - \mathrm{EA}$$

where IP and EA are ionization potential and electron affinity.

Rappe and Goddard (1991)
doi://10.1021/j100161a070?rand=h0p8l69f

Adapting the clever trick by Gilson et al., we predict the first- and second- order derivative of $E_A(Q)$, and form this problem as a double optimization, where,

$$\{\hat{e}_i, \hat{s}_i\} = \underset{e_i, s_i}{\operatorname{argmin}}\left(\underset{q_i}{\operatorname{argmin}} \sum_i e_i q_i + \frac{1}{2} s_i q_i^{\,2}\right)$$

subject to:

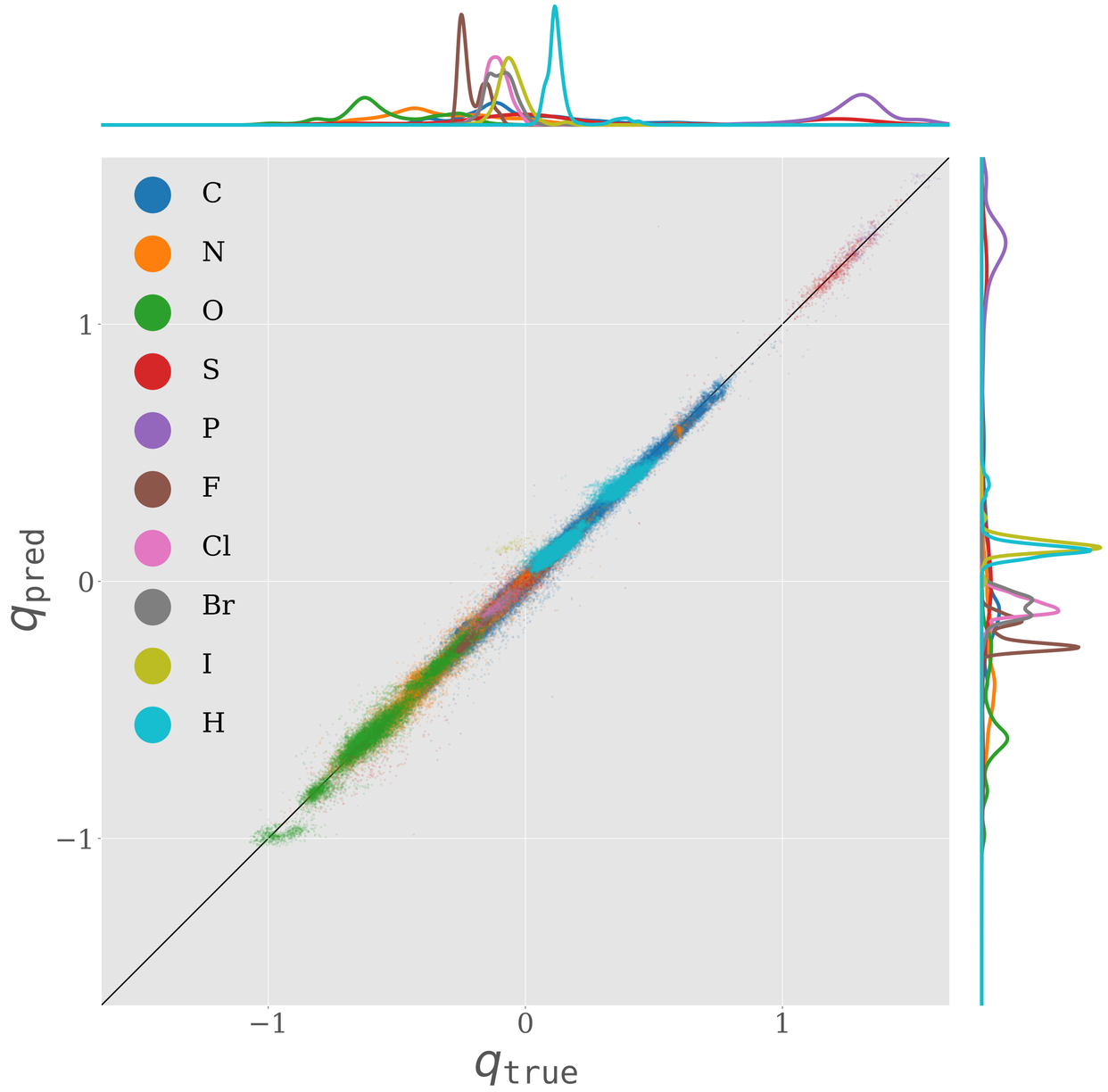
$$\sum_i q_i = \sum_i q_{i0}$$

For the second minimization, i.e. solving {qi} with given {ei} and {si}, it could be solved analytically using Lagrange multipliers,

$$\hat{q}_i = -e_i s_i^{-1} + s_i^{-1} \frac{Q + \sum_i e_i s_i^{-1}}{\sum_j s_j^{-1}}$$

whose Jacobian and Hessian are trivially easy to calculate.

Gilson et al. (2003) doi://10.1021/ci034148o



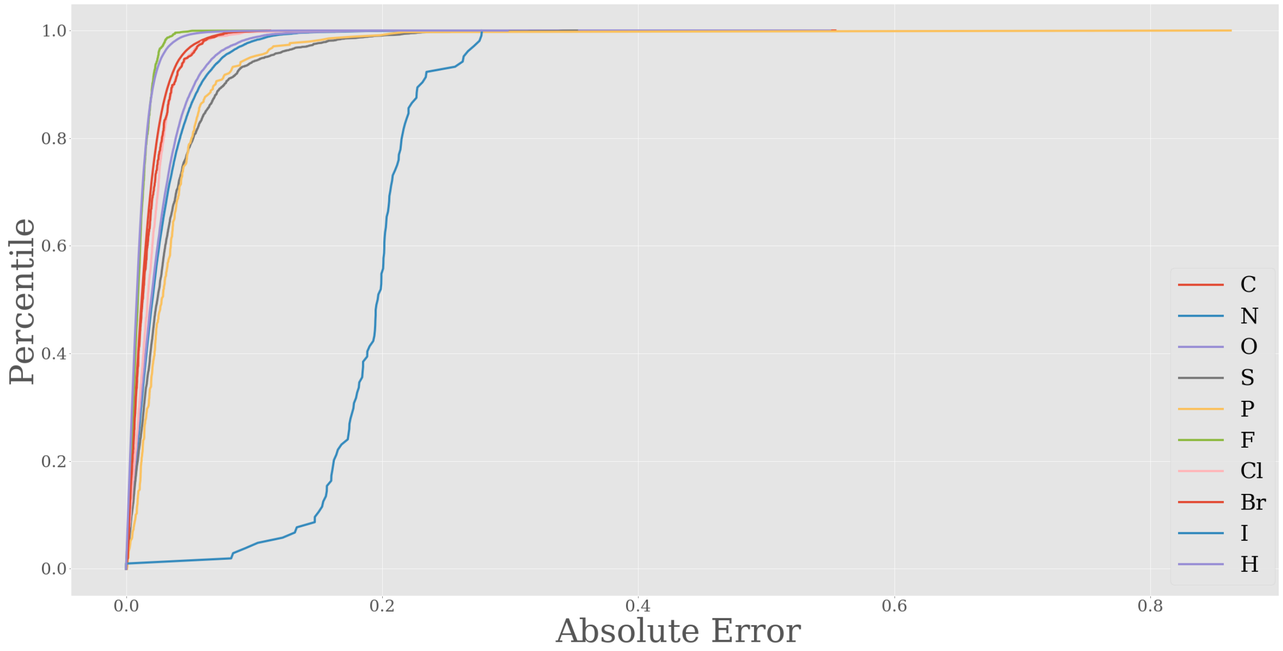

Predicted versus reference charge on held-out test set.

| Element | $R^2$ | $RMSE$ | Number of Data Points |
|---|---|---|---|
| C | $0.9938^{0.9933}_{0.9930}$ | $0.0222^{0.0219}_{0.0215}$ | 116864 |
| N | $0.9938^{0.9805}_{0.9789}$ | $0.0221^{0.0375}_{0.0363}$ | 19490 |
| O | $0.9936^{0.9725}_{0.9701}$ | $0.0223^{0.0348}_{0.0335}$ | 21503 |
| S | $0.9937^{0.9941}_{0.9928}$ | $0.0222^{0.0551}_{0.0496}$ | 2955 |
| P | $0.9931^{0.9929}_{0.7240}$ | $0.0222^{0.0955}_{0.0347}$ | 341 |
| F | $0.9933^{0.9574}_{0.9462}$ | $0.0226^{0.0138}_{0.0126}$ | 1967 |
| Cl | $0.9938^{0.8047}_{0.7526}$ | $0.0218^{0.0270}_{0.0237}$ | 1215 |
| Br | $0.9940^{0.8452}_{0.7885}$ | $0.0211^{0.0252}_{0.0215}$ | 572 |
| I | $0.9954^{0.6596}_{-0.0297}$ | $0.0164^{0.2017}_{0.1875}$ | 105 |
| H | $0.9935^{0.9750}_{0.9738}$ | $0.0224^{0.0145}_{0.0142}$ | 134799 |
| Overall | $0.9936^{0.9937}_{0.9935}$ | $0.0223^{0.0225}_{0.0221}$ | 299811 |

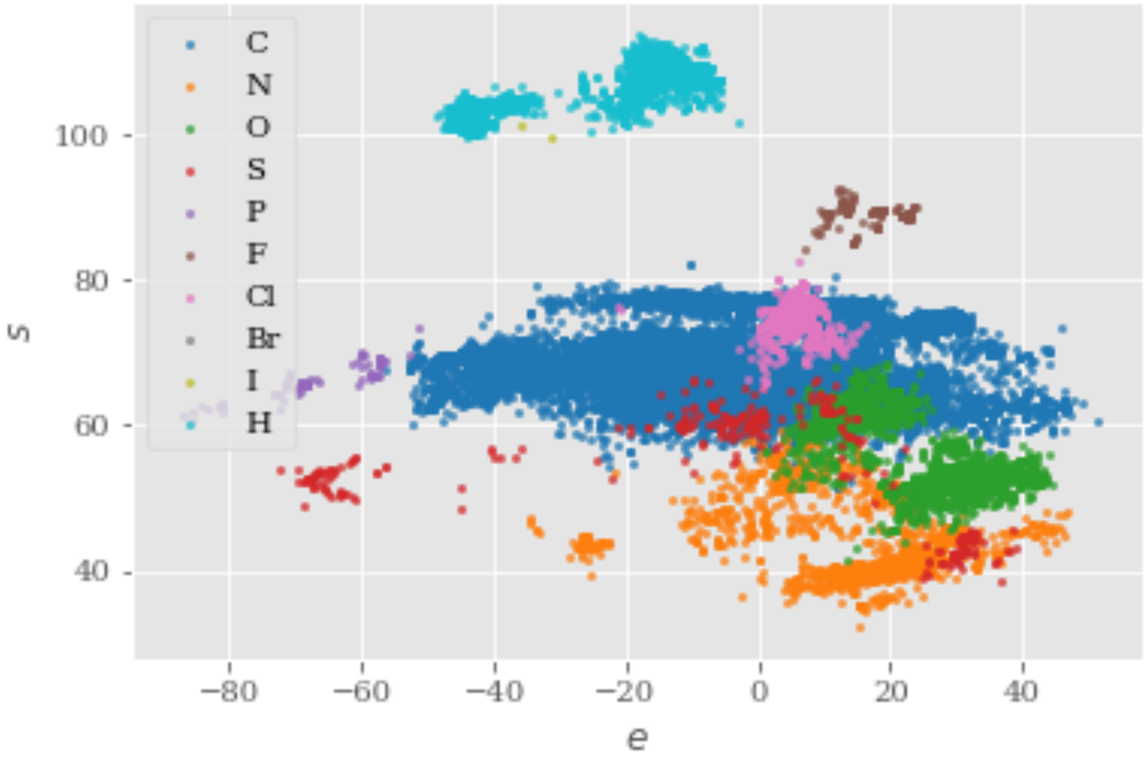
Table 1: $R^2$, $RMSE$, and number of data points in held-out test set grouped by element type. The 95% confidence interval (by bootstrapping 1000 times is reported in brackets.)

# ablation study

| method | $R^2$ | RMSE |
|---|---|---|
| **GN** | 0.9936 | 0.0223 |
| **MPNN without bond order** | 0.9930 | 0.0233 |
| **GN predicting *q*** | -6.9242E-06 | 0.280 |

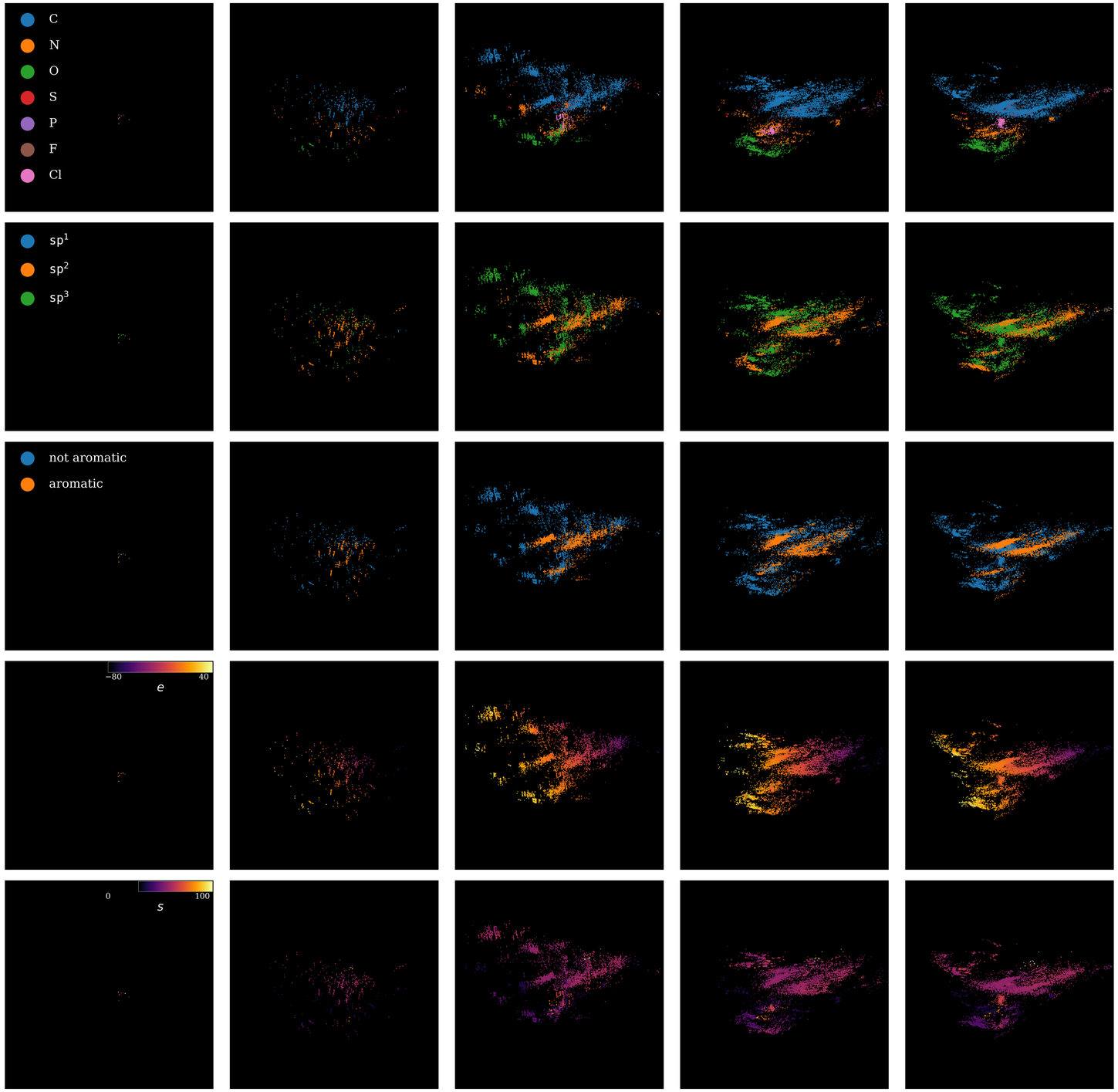


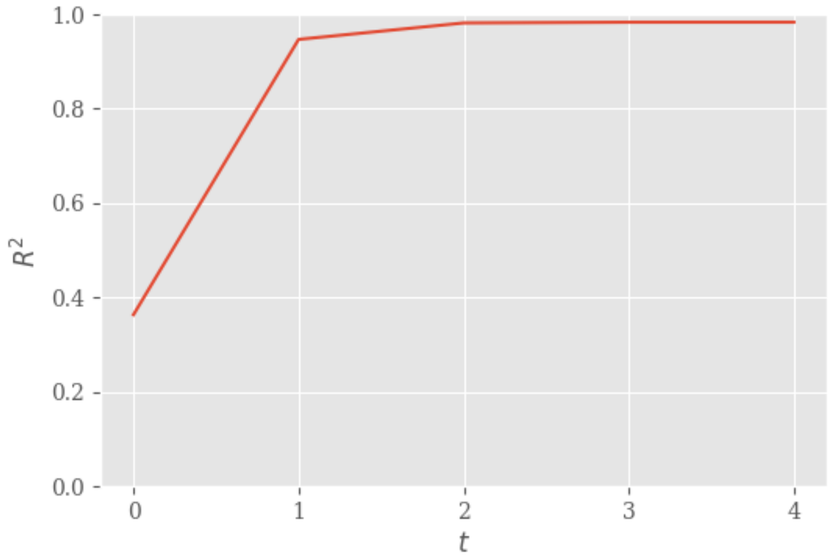

Cumulative fraction of samples as a function of absolute error in held-out test set.




Predicted electronegativity e and hardness s grouped by element.


t = 0



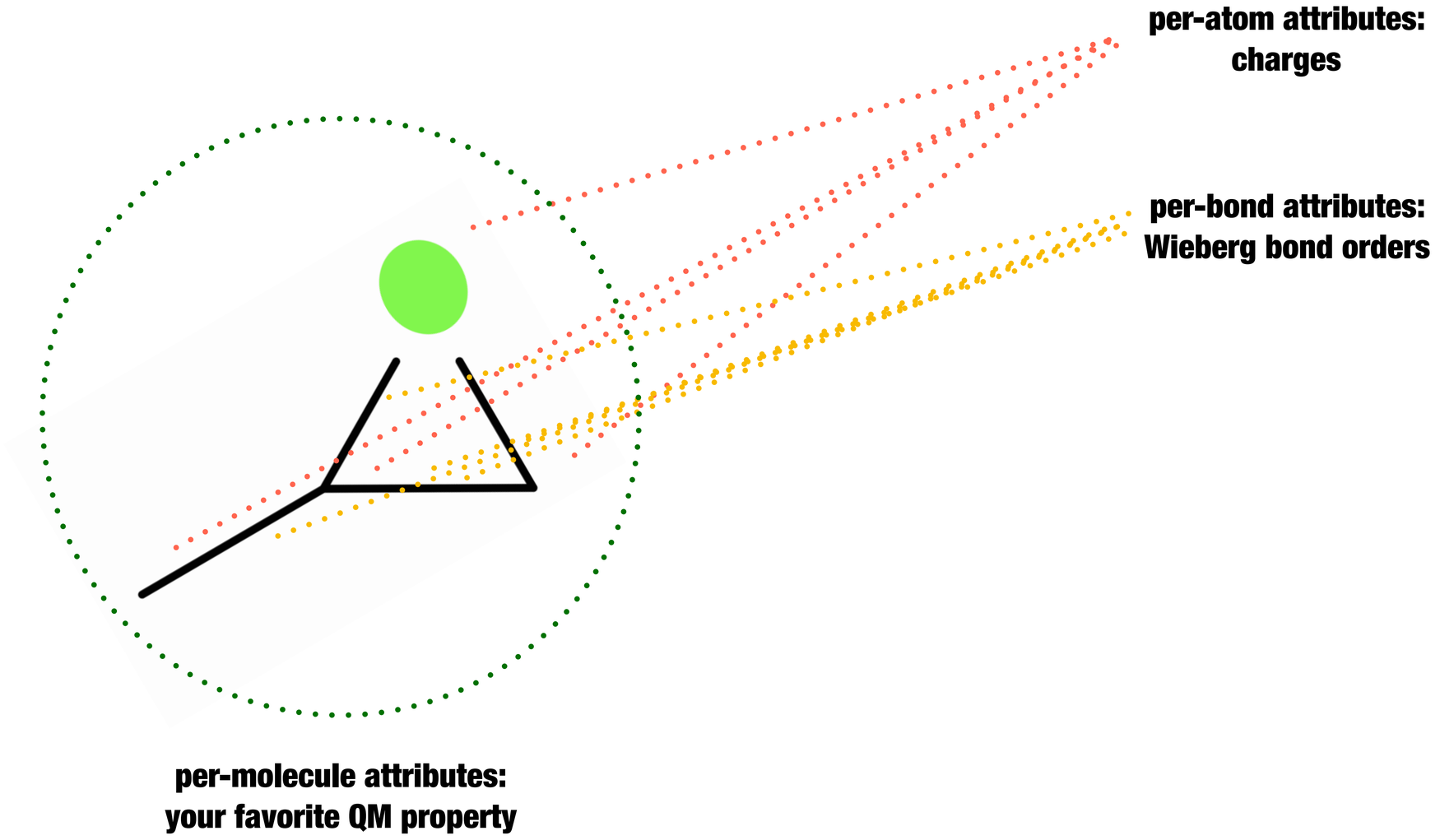

Left: Principal Component Analysis (PCA) of latent representations of node attributes, at different time step (from left to right), and color-coded according to (from top to bottom) elements, hybridizations, aromaticity, electronegativity, and hardness.

Above: $R^2$ of time-series linear regression on latent space.

# scalability of the model

# part 3
# work(s)-in-progress

# inter-hierarchical multitask learning



$$U = \sum_{e \in \mathcal{E}} E_{\text{bond}}(e) + \sum_{a \in \mathcal{A}} E_{\text{angle}}(a) + \sum_{t \in \mathcal{T}} E_{\text{torsion}}(t) + \sum_{v_0, v_1 \in \mathcal{V}, v_0 \notin \mathcal{N}^v(v_1)} E_{\text{non-bonded}}(v_0, v_1)$$

$$E_{\text{bond}}(e) = \frac{1}{2} k_{\text{bond}}(e)(r(e) - r_{\text{eq}}(e))^2$$

$$E_{\text{angle}}(a) = \frac{1}{2} k_{\text{angle}}(a)(\phi(a) - \phi_{\text{eq}}(a))^2$$

$$E_{\text{torsion}}(t) = \sum_{n=1:N_{\text{phases}}(t)} k_{\text{torsion},i}(t)(1 + \cos(n\phi(t) - \phi_{eq}(t)))$$

$$E_{\text{non-bonded}}(v_0, v_1) = \underbrace{4\epsilon(v_0, v_1)\left[\left(\frac{\sigma(v_1, v_2)}{r(v_0, v_1)}\right)^{12} - \left(\frac{\sigma(v_0, v_1)}{r(v_0, v_1)}\right)^{6}\right]}_{\text{van der Waals}} + \underbrace{\frac{1}{4\pi\epsilon_0}\frac{q(v_0)q(v_1)}{r(v_0, v_1)}}_{\text{Coulombic}}$$

$$U = \sum_{e \in \mathcal{E}} E_{\text{bond}}(e) + \sum_{a \in \mathcal{A}} E_{\text{angle}}(a) + \sum_{t \in \mathcal{T}} E_{\text{torsion}}(t) + \sum_{v_0, v_1 \in \mathcal{V}, v_0 \notin \mathcal{N}^v(v_1)} E_{\text{non-bonded}}(v_0, v_1)$$

$$E_{\text{bond}}(e) = \sum_{i=1}^{d} [\theta_e]_i r(e)^i;$$

$$E_{\text{angle}}(a) = \sum_{i=1}^{d} [\theta_a]_i \phi(a)^i;$$

$$E_{\text{torsion}}(t) = \sum_{i=1}^{d} [\theta_t]_i \phi(t)^i;$$

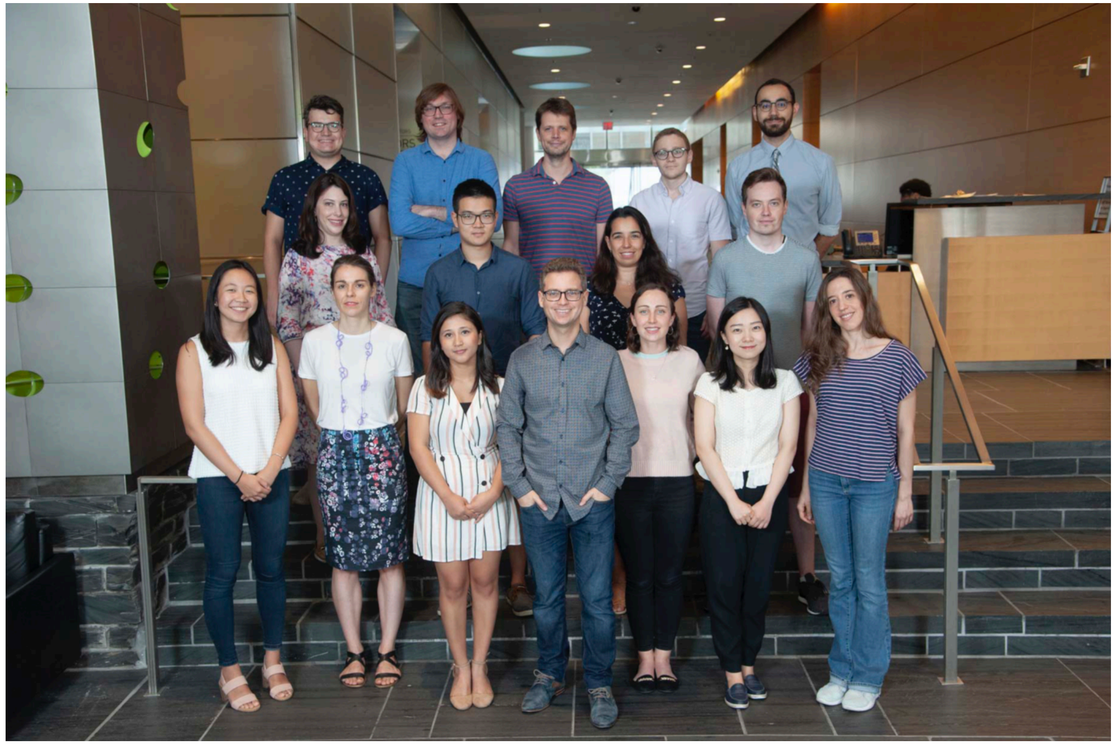
$$E_{\text{pairwise}}(v_0, v_1) = \sum_{i=1}^{d} [\theta_{\text{pairwise}}]_i (r + \epsilon)^{-i}.$$

$$\{\{\theta_v\}, \{\theta_e\}, \{\theta_a\}, \{\theta_d\}\} = f^r(\{\{\mathbf{v}^{(t)}, \mathbf{e}^{(t)}, \mathbf{a}^{(\mathbf{t})}, \mathbf{d}^{(t)}, \mathbf{u}^{(t)}\}, t = 1, 2, ..., T\})$$

$$\{\theta_v\} = \text{NN}_{r,v}(\{\mathbf{v}^{(t)}\}), \{\theta_e\} = \text{NN}_{r,e}(\{\mathbf{e}^{(t)}\}), \{\theta_a\} = \text{NN}_{r,a}(\{\mathbf{a}^{(t)}\}), \{\theta_d\} = \text{NN}_{r,d}(\{\mathbf{d}^{(t)}\})$$

$$\{\theta_{\text{pairwise}}\} = \{\text{NN}_{r,\text{pairwise}}(\theta_{v,0}, \theta_{v,1}), v_0 \notin \mathcal{N}_{v_1}^v\}$$

# acknowledgement




Chodera Lab:

Josh Fass

Chaya Stern

John Chodera

Uli Statistical Learning: Kun Luo


Memorial Sloan Kettering Cancer Center

The City College of New York

MFA PROGRAM IN CREATIVE WRITING

HPC HIGH PERFORMANCE COMPUTING

Weill Cornell Medicine