• So output another graph w/ updated edge & node ind!

+ again, stackable!

\[
\text{graph} \rightarrow \text{MP layer 1} \rightarrow \text{MP layer 2} \rightarrow \ldots
\]

before final layer, must involve permute & aggregating e.g. sum or max poly

c.s. \( \Sigma \) (nodes)

\[ \rightarrow \text{MLP} \rightarrow \text{out} \]

• Example: dynamic graph CNN (1801.07829)

k NN edges \( \rightarrow \) new edges based on k NN of learned features

\[
\mathbf{e}_{ab} = \text{edge}(a, b)
\]

\[ \phi \mathbf{a} = \mathbf{e}_{ac} \rightarrow \text{new edges} \]

• ParticleNet based on DGCNN won top prize competition ca 2019! (10% better R50 than kettle)

• Graph nets—very flexible & powerful framework!

• Domsde: compute scales w/ \( N^2 \) Nodes (\( \text{Nodes} = \text{Particles} \times \text{LHC} \ldots \))
Another application of GNNs

Regression for "inter-atomic potential" in materials science

- Atoms positions $\mathbf{r}_i$
- Atomic $Z_i$

want total potential energy $E_{\text{pot}}$
& forces acting on atoms $F_i$

good approx (?)

$$E_{\text{pot}} = \sum E_{\text{atomic}}$$

$$\sum_i F_i = -\nabla E_{\text{pot}}$$

can be calculated using Density Functional Theory but very computationally expensive.

A lot of data

A literature in regressing DFT w/ NWS -> GNNs especially well-suited!

system already in the form of a graph!
- node attributes $\mathbf{r}_i$, $Z_i$
- edge attributes $\mathbf{r}_{ij}$ (rotational & translational symmetry)

MP-NN -> final graph output gives $E_{\text{atomic}}$

Fully differentiable!
- at each node $E_{\text{pot}}$ from final sum over nodes
- forces
One more architecture: **Transformers** (Vaswani et al. 2017)

- Used in state of the art NLP like ChatGPT
- In simplest form is also permutation invariant but well suited to sequences
- Like an MLP with input-dependent weights $X \rightarrow WAX \cdot X$
  - So much more powerful in principle!

- Key idea: self-attention mechanism
- Key idea: learning embeddings

NLP example: "The animal didn't cross the street because it was tired"

attention $\rightarrow$ it is associated with "the animal" and not "the street"

Previous approaches to NLP were more sequence oriented
- trouble learning relations between distinct words
- "long-term correlations"

Transformers, being inherently orderless, work much better!
Self-Attention (Bahdanau 2014)

First step — embed words into fixed size vector

$\textit{could be learned MLP}$

So each event/sentna/etc

\[ \begin{array}{c}
\text{w/N constants} \\
\rightarrow (N, d)
\end{array} \]

Self-Attention

Output are the transformed embeddings

Note: it is on all—we all process

Through attention output embeddng encodes

What other embeddings or
corresponding other constituents were relevant

Learn 3 matrices $W^Q$, $W^K$, $W^V$ "query, key, value"

and $d$ and $d$

$Q = XW^Q (N, d')$

$K = XW^K (N, d')$

$V = XW^V (N, d')$

$Q K^T$ is $N \times N$

$\text{Softmax} (Q K^T) \rightarrow N \times N$

$\begin{pmatrix}
\pi_1 & \pi_2 & \ldots & \pi_N
\end{pmatrix}$

Each $m \in \{0, 1\}$
\[
\text{softmax}(QK^T)V \\
\rightarrow \text{weighted sum of value vectors}
\]

\[
\begin{pmatrix}
\tilde{v}_1 \\
\vdots \\
\tilde{v}_N
\end{pmatrix} = \begin{pmatrix}
\tilde{v}_1 \\
\vdots \\
\tilde{v}_N
\end{pmatrix} + \alpha \begin{pmatrix}
\sum \tilde{v}_1 \tilde{v}_1 \\
\vdots \\
\sum \tilde{v}_N \tilde{v}_i
\end{pmatrix}
\]
output of self-attention block.

Intuition: softmax output is where attention happens
weights irrelevant parts \( \rightarrow 0 \)
relevant parts \( \rightarrow \infty \).

Example: the "it" word relates to 8 in sentence
\( N = 10 \)

\( \text{weights set: } \tilde{v} = (0.5, 0.5, 0, \ldots, 0) \)

But "it" also relates to "tired" — what about that?

\( \text{"Multi-headed attention" analogous to multiple filters in CNN} \)

Learn multiple \( W^Q, W^K, W^V \)
\( q_{i,j} \)
each gives transformed embedding \( (N, d') \) \( z_a \)
Concatenate \( (N, d' \times N) \) \( (z_1, z_2, \ldots, z_N) \) \( N = Z \)
\( \text{learn final matrix } W^o \) \( (d' \times N, d'') \)
\( z_{d''} = z^o W^o \)