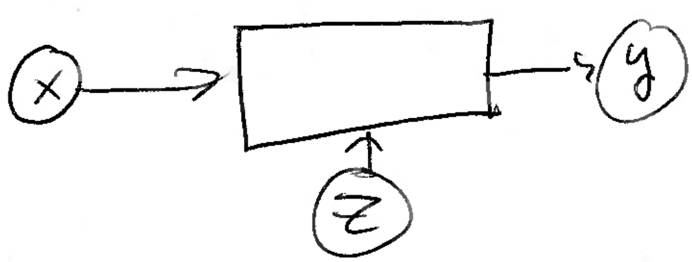
- Attention
- Code example
- Some more examples of Hamiltonian systems

We point to ENM 531 and Murphy's textbook for discussion about architectures, but ~~for~~ final projects I want to make sure everyone has access to a modern transformer architecture

Consider the <u>conditional neural field</u>



$$y = f(x \mid z ; \theta)$$

<u>Idea</u> z modulates the x, y input/output relationship

<u>ex</u>

y - finite difference stencil
x - grid function
z - material properties / microscopy

Many options : - deep ONet (Lu Lu)
- PCA / FNO (Stewart)
- Cross Attention

<u>Attention</u> as soft dictionary lookup (see Murphy 15.4.1)

- Consider $(k_i, v_i)_{i=1}^{N}$ <u>key</u>- <u>value</u> pairs describing input/output labels

- Consider a <u>query</u> q where model is evaluated

<u>def</u> $$\text{Attn}(q, \{k_i, v_i\}) = \sum_{i=1}^{N} \alpha_i(q, \vec{k})\, v_i$$

for attn weights

$$0 \leq \alpha_i \leq 1$$
$$\sum_i \alpha_i = 1$$

introducing an <u>attention score</u> $a(q, k_i)$

$$\alpha_i = \text{softmax}(a(q, \vec{k})) = \frac{\exp[a(q, k_i)]}{\sum_j \exp[a(q, k_j)]}$$

<u>Remark</u> This is a data driven basis

e.g. Taking $\alpha_i(q, \vec{K}) = \phi_i(q)$

$\phi_i \in CPWL$

satisfies the same properties

$$f(x) = \sum_i \alpha_i(x, x_n)\, y_i$$

$q \to x$

$K \to x_n$ (nodal values)

$y_i \to$ nodal basis functions

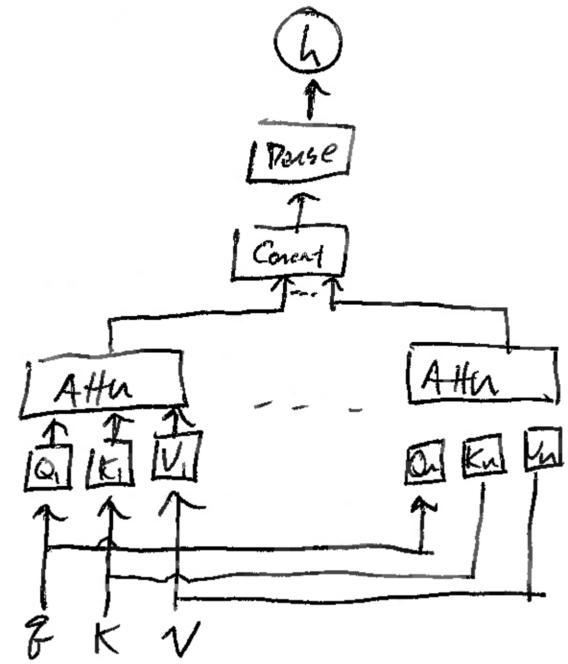
<u>Self-Attention</u> $\quad q, K, V = X$

<u>Multihead Attention</u>

- Introduce Dense MLP Blocks $i = 1, \dots M$

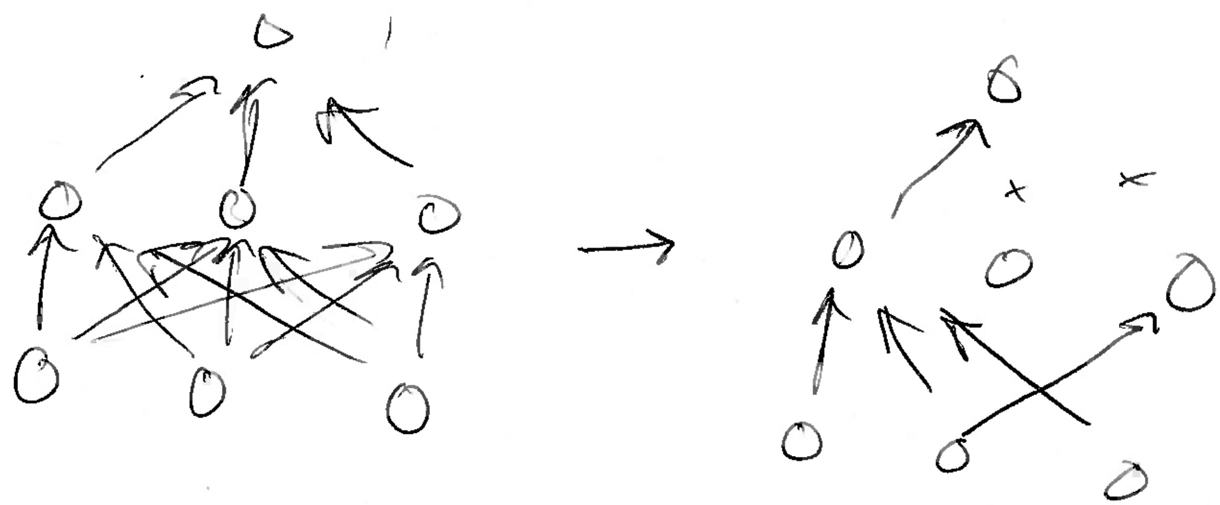
$$\hat{q}_i = Q_i(q)$$
$$\hat{K}_i = K_i(k)$$
$$\hat{v}_i = V_i(v)$$

$$h = MHA(q, \{K, v\})$$



Dropout

With probability p, at each forward pass the output of a given neuron will be dropped



Replace weights

$$\theta_{lji} = W_{lji} \, \varepsilon_{li}$$

$$\varepsilon_{li} \sim \mathrm{Ber}(1-p)$$

Graph Attention

Steps

1. <u>Input features</u>

   Each node $n_i \in N$ has feature vector $h_i^n \in \mathbb{R}^F$

2. <u>Linear Transform</u> $h_i' = W_i h_i^n$, $W \in \mathbb{R}^{F' \times F}$

3. <u>Preattention coeffs</u>

$$e_{ij} = \underset{\text{leaky ReLU}}{\sigma}\Big(\underset{\text{trainable}}{a} \cdot [Wh_i \underset{\text{concat}}{\|} Wh_j]\Big)$$

4. <u>Attention mech.</u>

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \sim i} \exp(e_{ik})}$$

5. <u>Aggregate</u>

$$h^{n+1} = \sigma\Big(\sum_{j \sim i} \alpha_{ij} W h_j^n\Big)$$

Physics-Inspired Arch.

GRAND

$$h_i^{n+1} = h_i^n + \sigma\left(\sum_{j \sim i} \alpha_{ij} \left(h_j^n - h_i^n\right)\right)$$

or in graph calculus

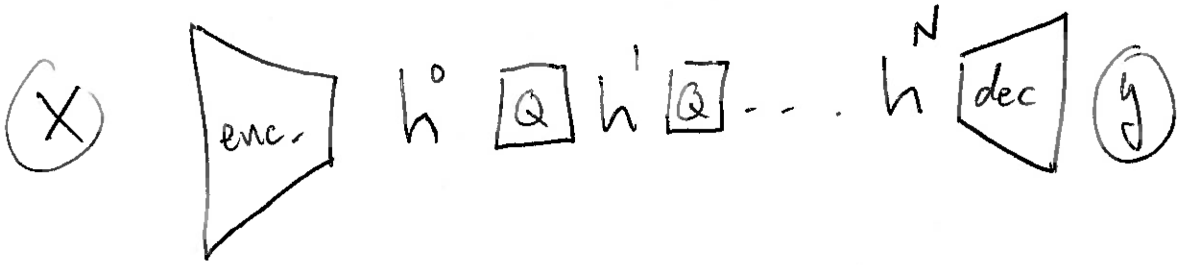
$$h_i^{n+1} = h_i^n + \sigma\left(\delta^* \delta h^n\right)$$

$$\langle f, \delta^* g \rangle = \langle \delta f, g \rangle_\alpha$$

view attention as learning inner-product

Writing $h^{n+1} = Q^n h^n$

Arbitrary graph data $\mathcal{D} = \{X_i, y_i\}$



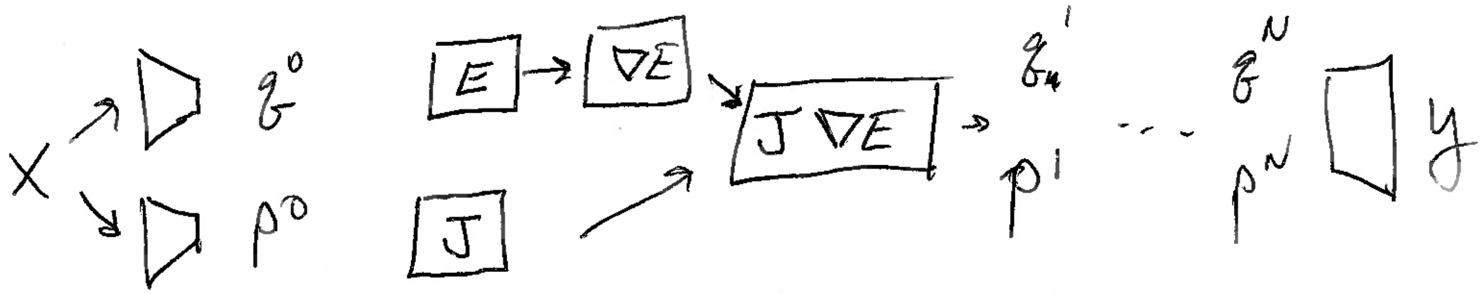
Lets revisit Hamiltonian mech.

$$\frac{d}{dt}\begin{pmatrix} q \\ p \end{pmatrix} = \begin{pmatrix} 0 & -\delta_0^* \\ \delta_0 & 0 \end{pmatrix}\begin{pmatrix} \partial_q E \\ \partial_p E \end{pmatrix}$$

$$E = NN(p, q)$$

<u>Thm</u> $\dot{E} = 0$

Same as GRAND

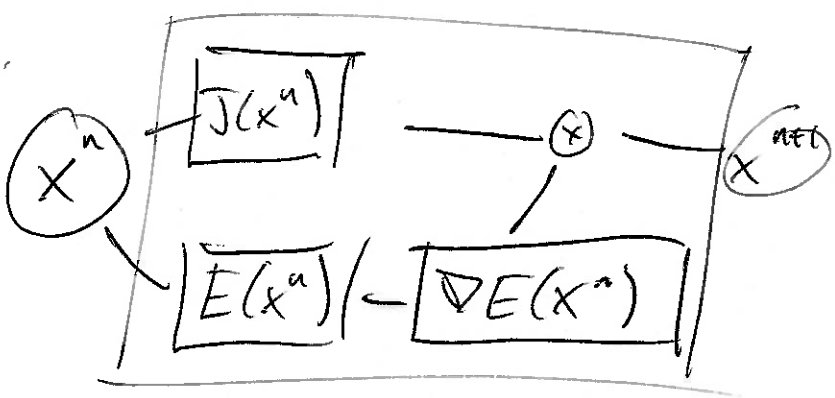


Double Bracket dynamics

$$\frac{dx}{dt} = J \nabla E + - J^T J \nabla E$$

$$\frac{dE}{dt} = \frac{\partial E}{\partial x} \cdot \frac{dx}{dt}$$

$$= \underbrace{\cancel{\frac{\partial E}{\partial x} J \frac{\partial E}{\partial x}}}_{=0} + - \underbrace{\frac{\partial E}{\partial x} J^T J \frac{\partial E}{\partial x}}_{\geq 0}$$



Q^n

$x_0 \; \Box \; h^0 \; \boxed{Q^0} \; h^1 \; \boxed{Q^1} \ldots \; h^N \; \Box \; y$