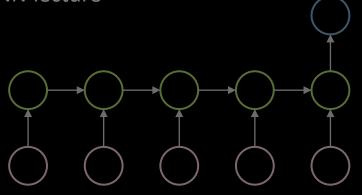
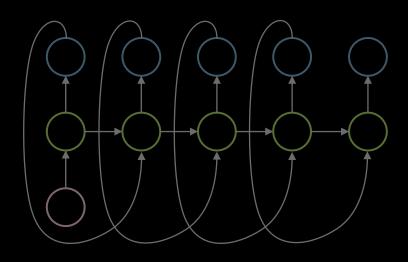
Attention (self/cross, hard/soft)

Dealing with sets

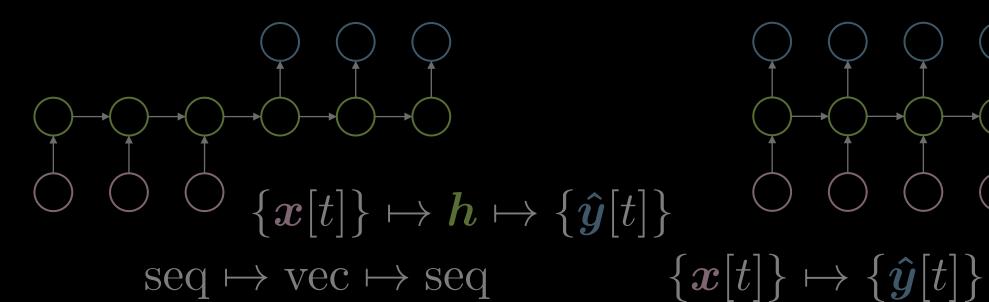
from the RNN lecture

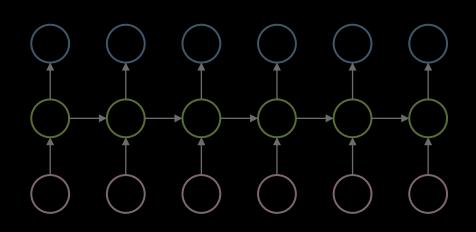


$$\{\boldsymbol{x}[t]\} \mapsto \hat{\boldsymbol{y}}[T] \quad \text{seq} \mapsto \text{vec}$$



$$x[1] \mapsto \{\hat{y}[t]\} \quad \text{vec} \mapsto \text{seq}$$





 $seq \mapsto seq$

Use cases

- img → set: image to bounding box (BB) (DETER)
- set → set: point clouds to BB, surrounding vehicle trajectory pred.
- seq → seq: translation, conditional image generation (DALL-e)
- seq → set: event location and duration
- img → vec: visual image transformer (VIT)
- seq → vec: movies review

Self-attention (I)

$$\{oldsymbol{x}_i\}_{i=1}^t = \{oldsymbol{x}_1, oldsymbol{x}_2, \cdots oldsymbol{x}_t\} \sim oldsymbol{X} \in \mathbb{R}^{n imes t}, \quad oldsymbol{x}_i \in \mathbb{R}^n$$
 $oldsymbol{h} = oldsymbol{lpha}_1 oldsymbol{x}_1 + oldsymbol{lpha}_2 oldsymbol{x}_2 + \cdots + oldsymbol{lpha}_t oldsymbol{x}_t = oldsymbol{X} oldsymbol{a}_t \in \mathbb{R}^n$
 $oldsymbol{lpha}_i = oldsymbol{a}_t oldsymbol{a}_t oldsymbol{a}_t oldsymbol{a}_t oldsymbol{a}_t oldsymbol{a}_t oldsymbol{a}_t = oldsymbol{a}_t oldsymbol{a}_$

Self-attention (II)

$$\boldsymbol{a} = \operatorname{softargmax}_{\beta}(\boldsymbol{X}^{\top}\boldsymbol{x}) \in \mathbb{R}^{t}$$

$$\{\boldsymbol{x}_i\}_{i=1}^t \leadsto \{\boldsymbol{a}_i\}_{i=1}^t \leadsto \boldsymbol{A} \in \mathbb{R}^{t \times t}$$

$$\{\boldsymbol{a}_i\}_{i=1}^t \leadsto \{\boldsymbol{h}_i\}_{i=1}^t \leadsto \boldsymbol{H} \in \mathbb{R}^{n \times t}$$

$$oldsymbol{H} = oldsymbol{X}oldsymbol{A} \in \mathbb{R}^{n imes t}$$

Key-value store

- Paradigm for
 - storing (saving)
 - retrieving (querying)
 - managing

an associative array (dictionary / hash table)

Queries, keys, and values

$$\{oldsymbol{q}_i\}_{i=1}^t \leadsto oldsymbol{Q} \in \mathbb{R}^{d' imes t}$$

$$oldsymbol{q} = oldsymbol{W_q} oldsymbol{x}, \quad oldsymbol{k} = oldsymbol{W_k} oldsymbol{x}, \quad oldsymbol{v} = oldsymbol{W_v} oldsymbol{x}$$
 $eta = oldsymbol{W_l} oldsymbol{1}$

$$oldsymbol{q}, oldsymbol{k} \in \mathbb{R}^{d'}, \quad oldsymbol{v} \in \mathbb{R}^{d''}$$

$$\{m{x}_i\}_{i=1}^t \leadsto \{m{q}_i\}_{i=1}^t, \{m{k}_i\}_{i=1}^t, \{m{v}_i\}_{i=1}^t \leadsto m{Q}, m{K}, m{V}$$

$$\boldsymbol{a} = \operatorname{softargmax}_{\beta}(\boldsymbol{K}^{\top}\boldsymbol{q}) \in \mathbb{R}^{t}$$

$$oldsymbol{h} = oldsymbol{V}oldsymbol{a} \in \mathbb{R}^{d''}$$

$$\{\boldsymbol{q}_i\}_{i=1}^t \leadsto \{\boldsymbol{a}_i\}_{i=1}^t \leadsto \boldsymbol{A} \in \mathbb{R}^{t \times t} \qquad \boldsymbol{H} = \boldsymbol{V}\boldsymbol{A} \in \mathbb{R}^{d'' \times t}$$

Queries, keys, and values

$$\{oldsymbol{q}_i\}_{i=1}^t \leadsto oldsymbol{Q} \in \mathbb{R}^{d' imes t}$$

$$q = W_q x, \quad k = W_k z, \quad v = W_v z^{\xi}$$
 $\beta = \frac{1}{\sqrt{d'}}$

$$oldsymbol{q}, oldsymbol{k} \in \mathbb{R}^{d'}, \quad oldsymbol{v} \in \mathbb{R}^{d''}$$

$$\{\boldsymbol{\xi}_j\}_{j=1}^{\tau} \leadsto \{\boldsymbol{k}_j\}_{j=1}^{\tau}, \{\boldsymbol{v}_j\}_{j=1}^{\tau} \leadsto \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{\{d',d''\} \times \tau}$$

$$\boldsymbol{a} = \operatorname{softargmax}_{\beta}(\boldsymbol{K}^{\top}\boldsymbol{q}) \in \mathbb{R}^{\tau}$$

$$oldsymbol{h} = oldsymbol{V}oldsymbol{a} \in \mathbb{R}^{d''}$$

$$\{\boldsymbol{q}_i\}_{i=1}^t \leadsto \{\boldsymbol{a}_i\}_{i=1}^t \leadsto \boldsymbol{A} \in \mathbb{R}^{\tau \times t} \qquad \boldsymbol{H} = \boldsymbol{A}$$

$$oldsymbol{H} = oldsymbol{V} oldsymbol{A} \in \mathbb{R}^{d'' imes t}$$

$$d' = d'' \stackrel{\downarrow}{=} d$$

Implementation

 $egin{bmatrix} oldsymbol{q} \ oldsymbol{k} \ oldsymbol{v} \end{bmatrix} = egin{bmatrix} oldsymbol{W_q} \ oldsymbol{W_k} \ oldsymbol{V} \end{matrix} \end{bmatrix} oldsymbol{x} \in \mathbb{R}^{3d}$

from the RNN lecture

$$h[t] = g(W_h[\mathbf{x}[t]] + b_h)$$

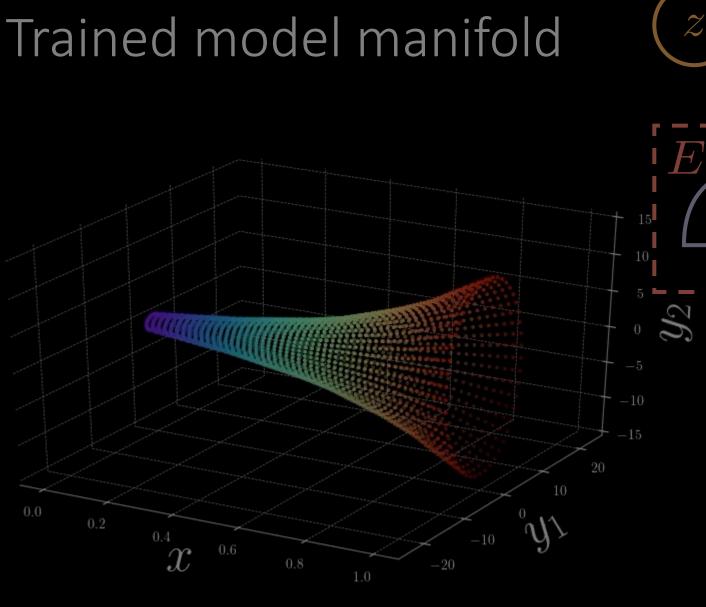
$$\boldsymbol{h}[0] \doteq \boldsymbol{0}, \boldsymbol{W_h} \doteq \begin{bmatrix} \boldsymbol{W_{hx}} \ \boldsymbol{W_{hh}} \end{bmatrix}$$

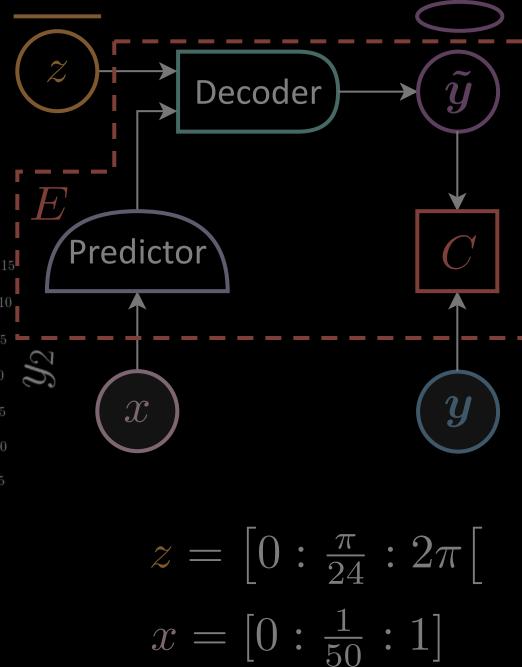
considering h heads we get a vector in \mathbb{R}^{3hd} using a $oldsymbol{W_h} \in \mathbb{R}^{d imes hd}$ to go back to \mathbb{R}^d

$$egin{bmatrix} oldsymbol{q}^1 \ oldsymbol{q}^2 \ oldsymbol{q}^h \ oldsymbol{q}$$

Transformer

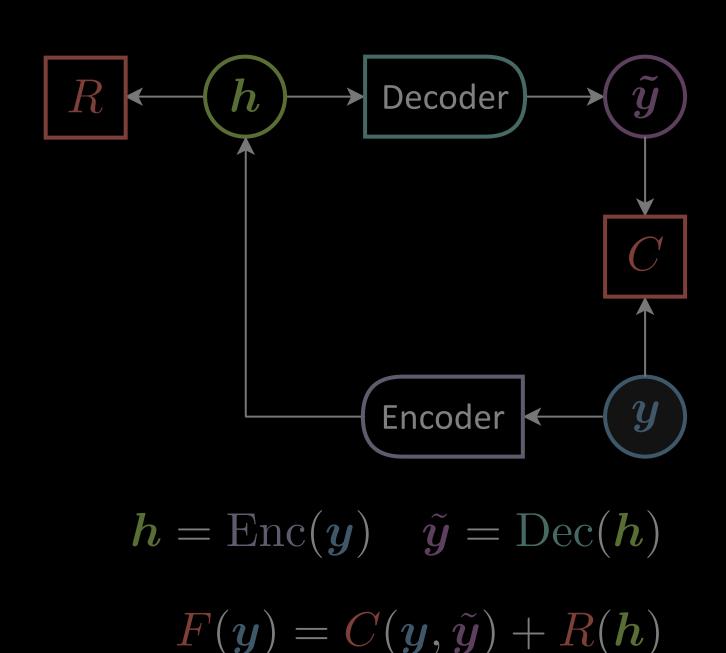
Encoders-predictor-decoder architecture (for Neural Machine Translation)





Autoencoder

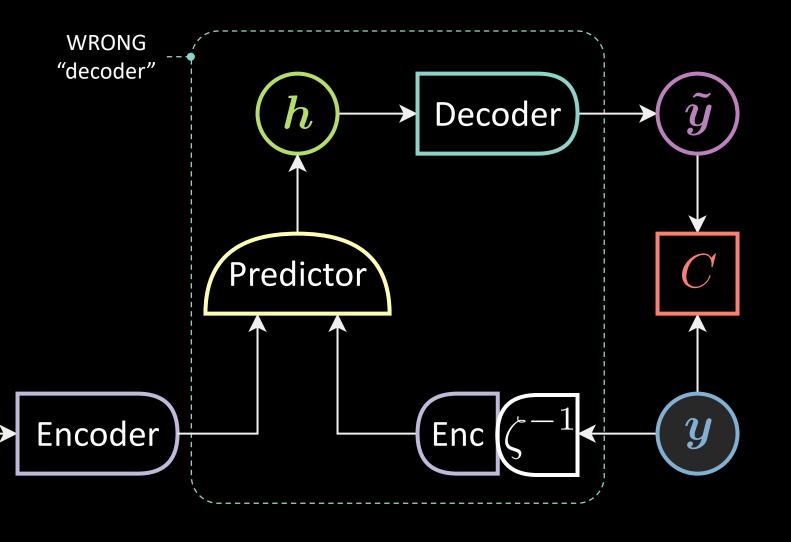
$$egin{aligned} m{h} &= f(m{W_hy} + m{b_h}) \ m{ ilde{y}} &= g(m{W_yh} + m{b_y}) \ m{y}, m{ ilde{y}} &\in \mathbb{R}^n \ m{h} &\in \mathbb{R}^d \ m{W_h} &\in \mathbb{R}^{d imes n} \ m{W_y} &\in \mathbb{R}^{n imes d} \end{aligned}$$



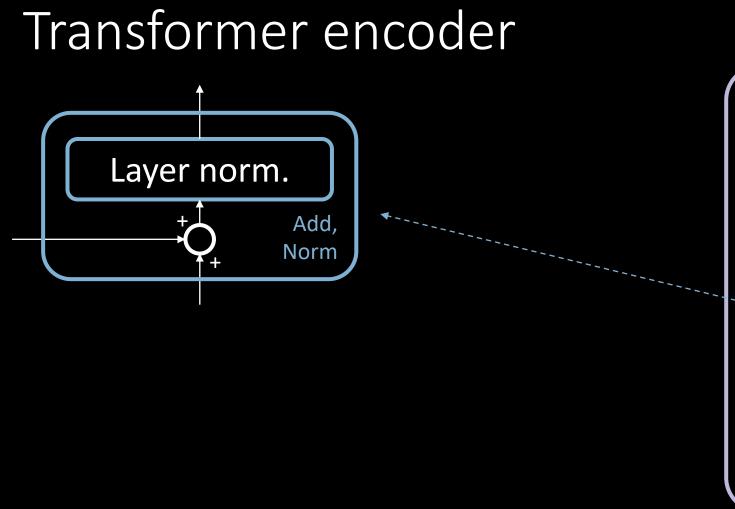
Transformer

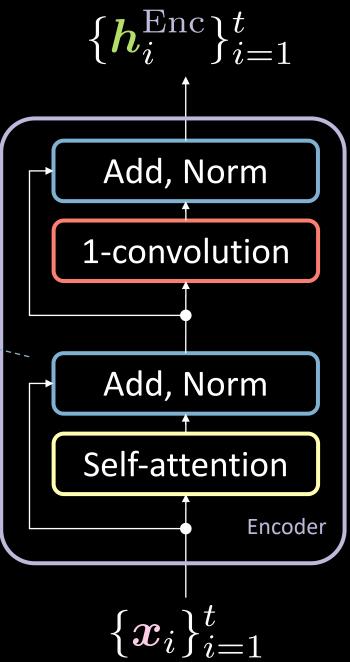
- $oldsymbol{x}$ source sentence
- y target sentence
- $ilde{oldsymbol{y}}$ predicted sentence

 \boldsymbol{x}



$$oldsymbol{y}[j-1] \longleftarrow oldsymbol{\zeta^{-1}} \longleftarrow oldsymbol{y}[j]$$
 unit delay





Add, Norm Layer norm. 1-convolution Add, Norm "Decoder" Add, Norm Add, Norm $\{oldsymbol{h}_i^{ ext{Enc}}\}_{i=1}^t$ Self-attention **Cross-attention** inference $\{\tilde{\boldsymbol{y}}_i\}_{i=0}^{\tau-1}$ $\{{m y}_j\}_{j=0}^{ au-1}$ training Add, Norm Layer norm. 1-convolution Add, Norm Decoder Add, Norm Add, Norm $\{oldsymbol{h}_i^{ ext{Enc}}\}_{i=1}^t$ Self-attention **Cross-attention** Encoder **Predictor** inference $\{ ilde{m{y}}_i \}_{i=0}^{ au-1}$ $\{{m y}_j\}_{j=0}^{ au-1}$ training