CS 6180 10/08

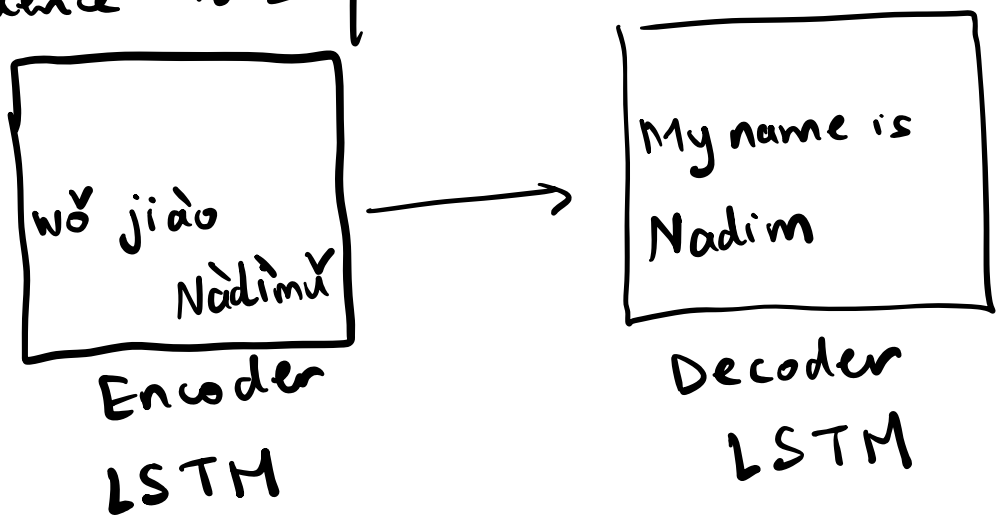
HW2 released Yaaa'Y

Review from last time

Attention

Translation source language → target language

sequence to sequence model



Attention (Cross-Attention)
(between different models)
put attention on the source sentence

to generate a word in the target sentence.

$\vec{e}_i$

Key

$\vec{k}_i = K\vec{e}_i$

Value

$\vec{v}_i = V\vec{e}_i$

query

$\boxed{\vec{q}_i = Q\vec{e}_i}$

decoder

Key representation

My name is

query

$\vec{q}_t$

source sentence

$\vec{e}_1^{\,enc}, \vec{e}_2^{\,enc}, \ldots, \vec{e}_m^{\,enc}$

$\vec{h}_1^{\,enc}, \vec{h}_2^{\,enc}, \ldots, \vec{h}_m^{\,enc}$

$\vec{K}_1 \quad \vec{K}_2 \quad \cdots \quad \vec{K}_m$

$\vec{K}_1 \quad \vec{K}_2 \quad \vec{K}_m$

similarity between keys and queries

$$\vec{K}_1^T \vec{q}_t \quad \vec{K}_2^T \vec{q}_t \quad \cdots \quad \vec{K}_m^T \vec{q}_t$$

$$\begin{array}{ccc} \| & \| & \| \\ s_{1t} & s_{2t} & s_{mt} \end{array}$$

(attention scores)

softmax to scale the scores

$$\frac{\exp(\vec{K}_i^T q_t)}{\sum_{j=1}^{m} \exp(\vec{K}_j^T q_t)} = \alpha_{it}$$

scaled attention scores

(attention distribution)

attention output

$$\vec{a}_t = \sum_{i=1}^{m} \alpha_{it} \vec{V}_i$$

$$\begin{bmatrix} \vec{a}_t \\ \vec{h}_t^{dec} \end{bmatrix} \xrightarrow{\text{generating word}}$$

taking into account words in the original sentence (with attention)

$$\vec{h}_t^{dec}, \vec{c}_t^{dec} = LSTM(\vec{h}_{t-1}^{dec}, \vec{c}_{t-1}^{dec}, \vec{e}_t)$$

replace

$$\begin{bmatrix} \vec{e_t} \\ a_t \end{bmatrix}$$

how much info was propagating from the source sentence to the target one

O(length of source sentence)

between relevant words in the source sentence and target

(previously)

with attention $O(1)$

Je m'appelle Nadim et j'aime le chocolat beaucoup

h_T^{enc}, c_T^{enc}

input decoder

interaction distance

between the word I want to generate now and a relevant source word is in the order of the length of the source sentence

My

With attention

interaction distance = 1

(getting some direct info from every source word)

Attention also helps with propagating info from previous words. That's why we looked at RNNs.

Why not give up on RNNs (LSTMs)

and build models that are entirely based on attention?

<u>Self attention:</u> attention framework WITHIN the same sentence.

$$\vec{x}_1 \xrightarrow{E} \vec{e}_1 \qquad \vec{x}_2 \to \vec{e}_2 \qquad \vec{x}_3 \to \boxed{\vec{e}_3} \qquad \cdots \qquad \vec{x}_m \to \vec{e}_m$$

$$\vec{e}_i \to \begin{cases} \vec{q}_i = Q\vec{e}_i \\ \vec{k}_i = K\vec{e}_i \\ \vec{v}_i = V\vec{e}_i \end{cases}$$

$$s_{ij} = \vec{q}_i^{\,T}\vec{k}_j \qquad \alpha_{ij}, \quad \vec{a}_t = \sum \alpha_{ij} v_j$$

*Need to make sure we cannot peek at future words

Come up with a general framework for every word in the sentence

* Order (position) of words is not taken into account.

Zuko made his uncle some tea.

vs

His uncle made Zuko some tea.

Same words in both sentences but

different meanings.

* Need non-linearities to be able to handle complex data.

---

* Let's fix first the issue of word position

Let's add position vectors

$\vec{x_1} \rightarrow \vec{e_1} + \vec{p_1}$

$\vec{x_2} \rightarrow \vec{e_2} + \vec{p_2}$

$\vdots$

$\vec{x}_m \rightarrow \vec{e}_m + \vec{p}_m$

Zuko made his uncle tea at Zuko's place

vs

His uncle made Zuko tea at Iroh's place.

if we're only using self attention the output at Zuko's / Iroh's should be the same. We don't want the model to learn that. Adding position parameters to the model will give it flexibity to learn the difference between the sentences.

*Need to make sure that we cannot peek at future words.

Masking

$t=4$

Zuko made his uncle tea. at Zuko's

s_{1t} s_{2t} s_{3t} s_{4t} $\underbrace{s_{5t}\ s_{6t}\ s_{7t}}$

need to not contribute to my attention output

$$\vec{a}_t = \sum_{i=1}^{7} \alpha_{it} \vec{V}_i$$

$$\alpha_{6t} = \frac{\exp(s_{6t})}{\sum_{i=1}^{7} \exp(s_{it})}$$

$\downarrow$

0

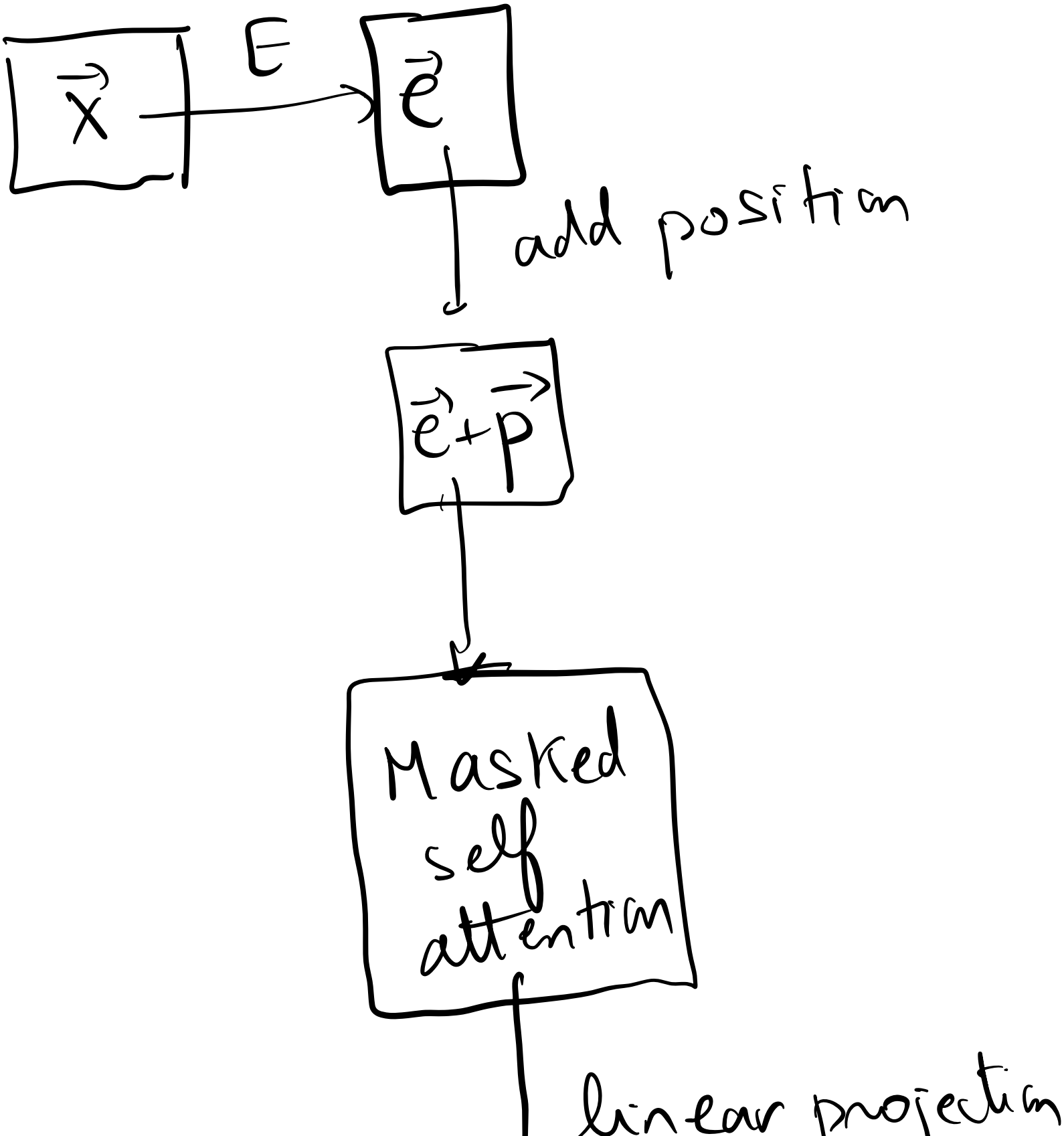
set $s_{6t} = -\infty$

$s_{7t} = -\infty$

$$s_{it} = \begin{cases} \vec{q}_t^{\,T} \vec{k}_i & \text{if } i \leq t \\ -\infty & \text{if } i > t \end{cases}$$

ensures that we are not peeking

Yay for masking



softmax

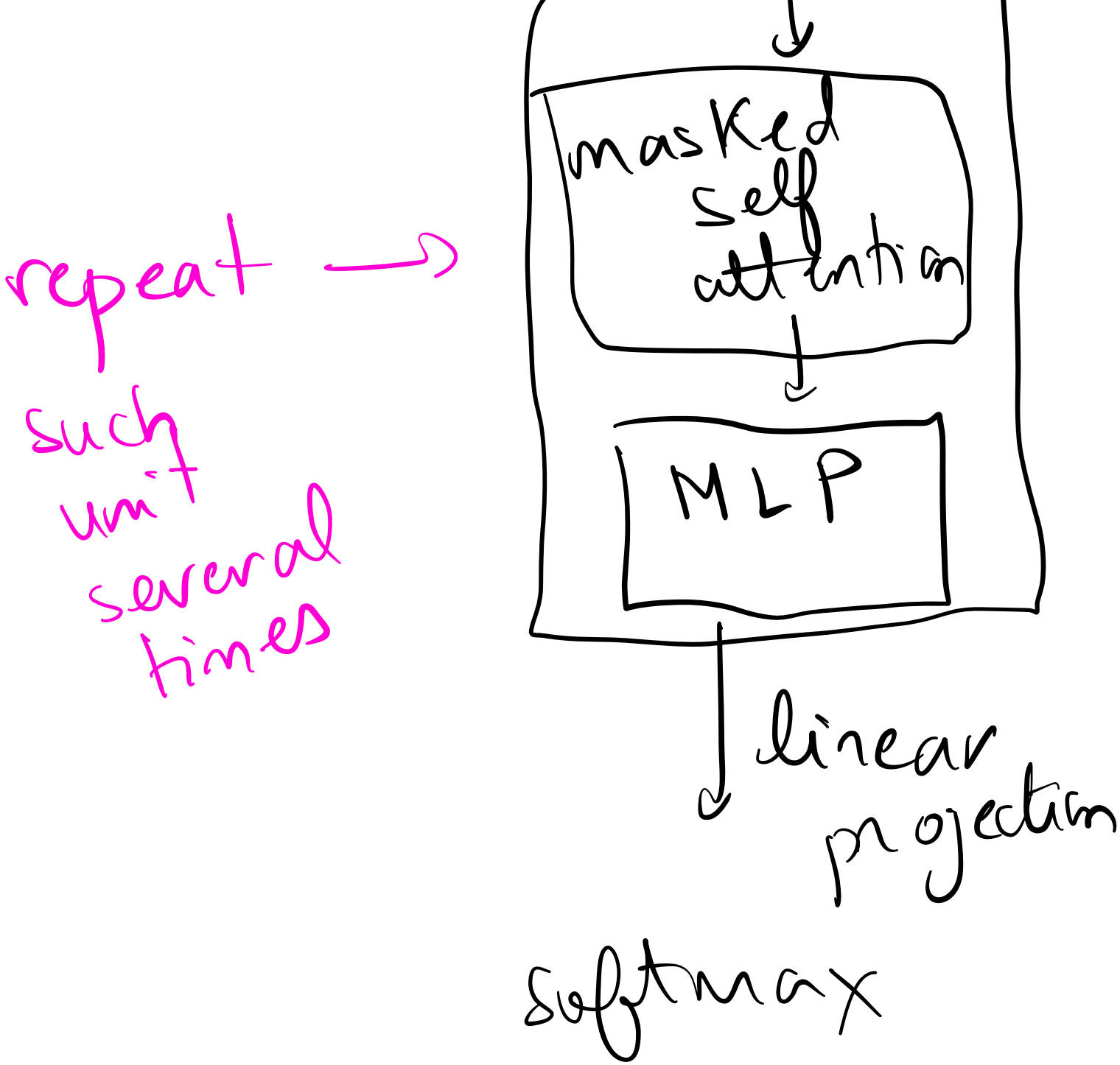
Multilayer perceptron

$$\vec{h}^{(1)} = \sigma\left(W^{[1]}\vec{x} + \vec{b}^{[1]}\right)$$

$$\vec{h}^{(2)} = \sigma\left(W^{[2]}\vec{h}^{(1)} + \vec{b}^{[2]}\right)$$

$$\vdots$$

$$\boxed{\vec{x}} \xrightarrow{E} \boxed{\vec{e}} \xrightarrow{\text{position}} \boxed{\vec{e} + \vec{p}}$$



(Almost)

Transformer (Decoder) architecture

3 main components to add to get the full transformer architecture

* have more than one attention head (Q, V, K)

meaning

grammar

syntax