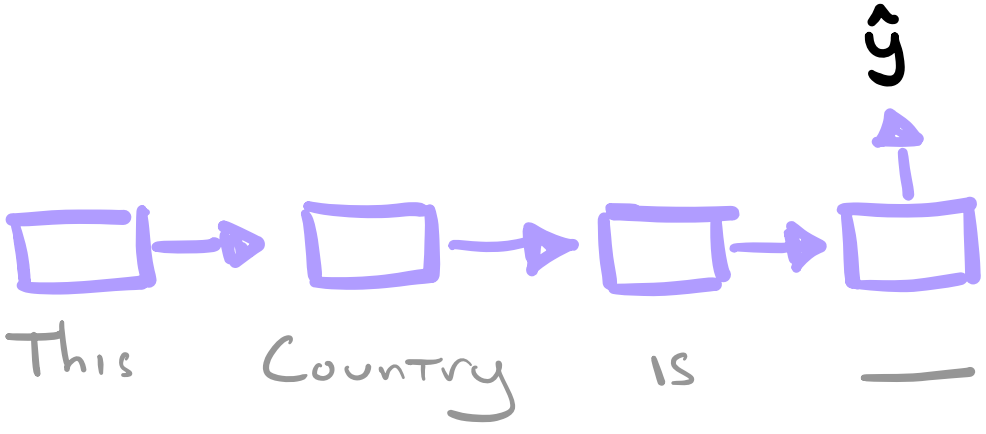
DS 4440 RNNs → Contextual embeddings → Transformers

Autoregressive generative model:

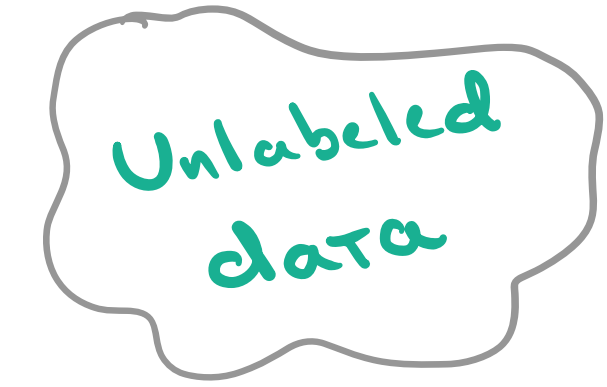
$$P(x_T \mid x_{T-1} \cdots x_1)$$

$$P(x) = \prod_T P(x_T \mid x_{<T})$$

e.g. Language Modeling.

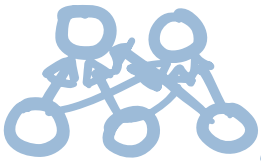


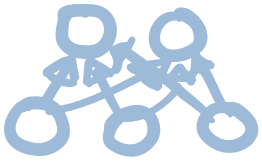
Language Modeling is a kind of Self-Supervised learning, often used for Pre-Training



e.g. the Internet

+ Self Supervised loss | Pretrain





+ $\mathcal{L}$ and (x, y) | Fine-Tune

OR

(x, y) | In Context learning (later)

Contextualized Word Vectors

Recall (10.5 lecture) that W2V yields embeddings of words

"Cat" → W2V → [x y z] | Static; always x y z

"Cute Cat" → [x y z]

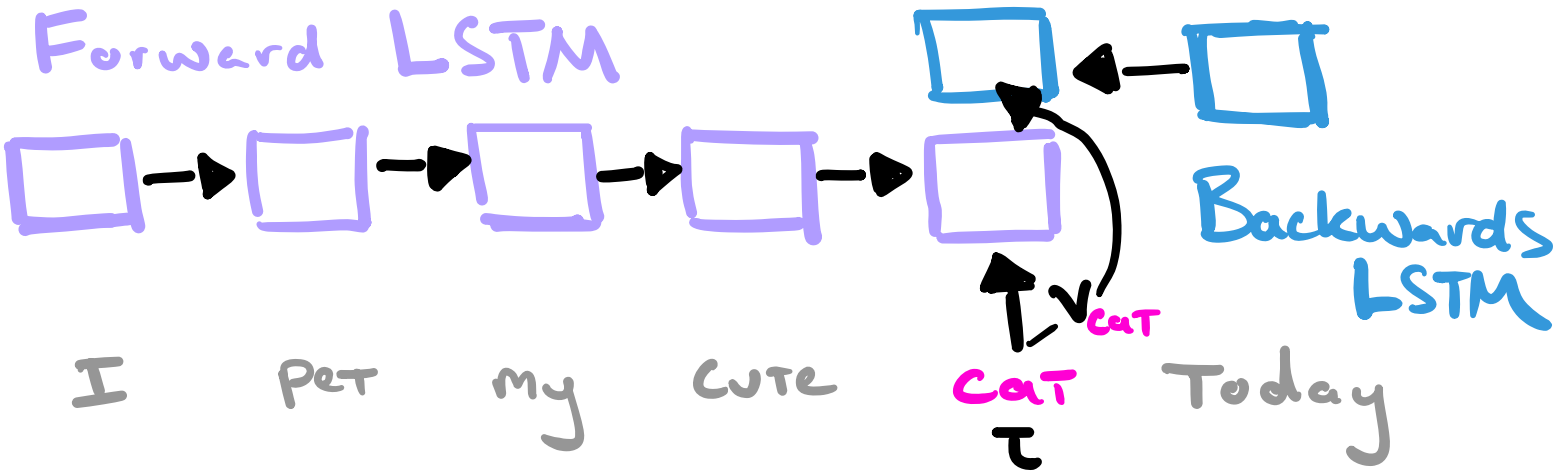
"Fat Cat" → [x y z]

IDEA Train an RNN for language modeling; extract contextualized embeddings

ELMo

Embeddings from Language Models

[Peters et al., '18]



$$\mathcal{L} = -P(cat \mid \overrightarrow{h}_t) - P(cat \mid \overleftarrow{h}_t)$$

} Self Supervised! LM

$$ELMo_{cat} = [v_{cat}; \overrightarrow{h}_t; \overleftarrow{h}_t]$$

Can now use $ELMo_{cat}$ vectors as features for downstream tasks

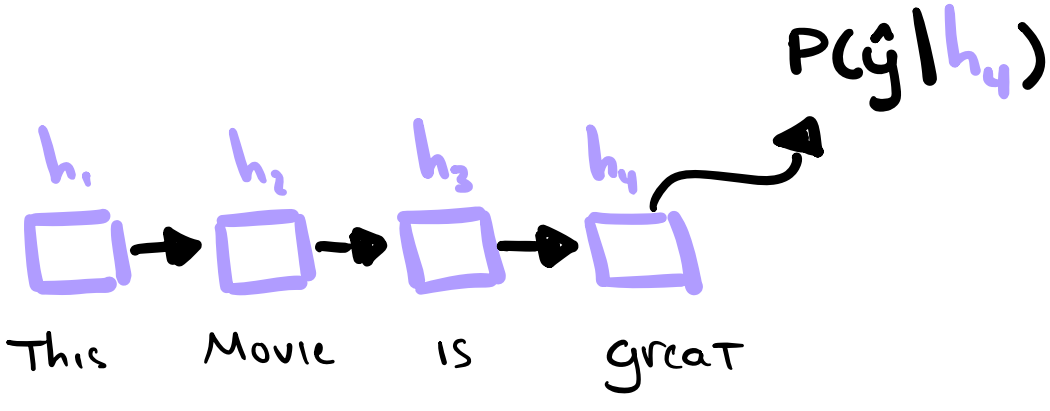
fine-Tuning!

This works pretty well! But, ELMo is based on RNNs and so its Slooooooow due to Sequential processing

Enter Transformers which dive away with recurrence using Self-attention.

(Vanilla)

Attention!



Idea Weight hidden states

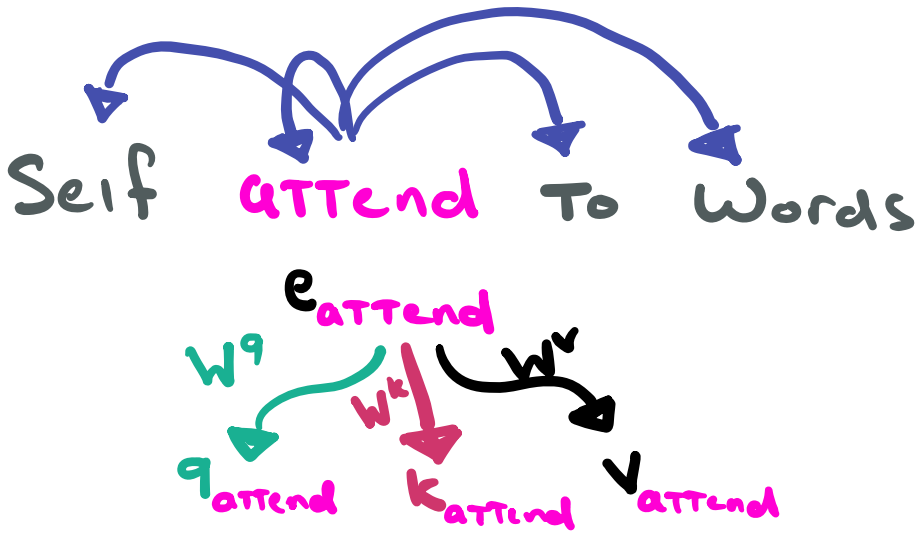
$$a_j = \underset{\langle 1 \times h \rangle \langle h \rangle}{W_a \cdot h_j}$$

$$\bar{h} = \sum_{j=2}^{n} \alpha_j h_j$$

$\alpha = \text{SoftMax}(a)$ | $P(y|\bar{h})$

Self-Attention

Generalizing: Assume keys k and queries q.



$a_j = S(k_j, q)$ $\alpha = \text{SoftMax}(a)$

$$\bar{h} = \sum_j \alpha_j v_j$$

e.g. $(k_j \cdot q)/\sqrt{d}$ "dot product Attn"

Scale by dims

or

$v_j^T \tanh(W_k k_j + W_q q)$ "MLP Attn"

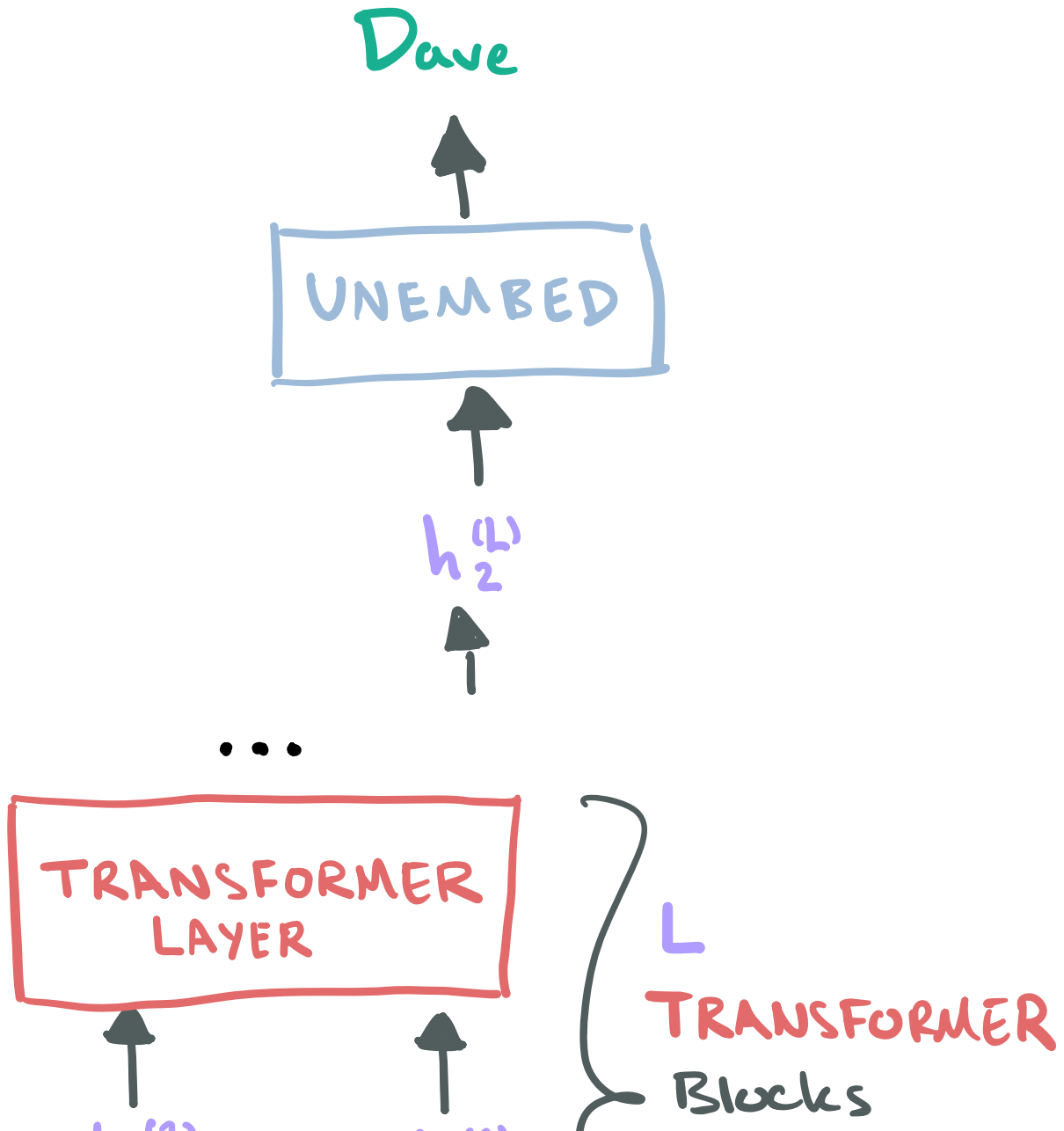
TRANSFORMERS

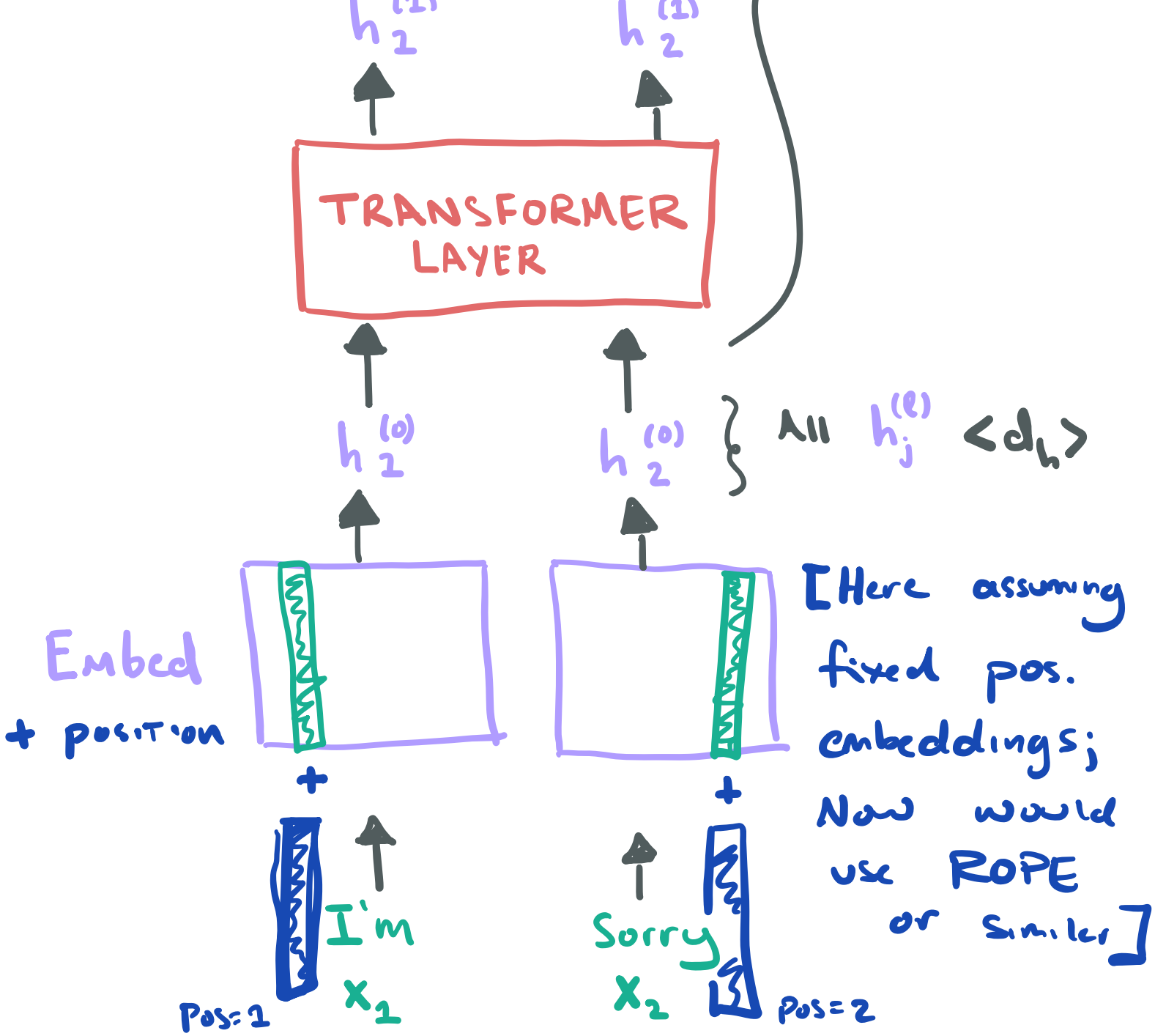
Tokens → Tokens

Assume auto regressive generation

Prefix → Next token
(context)

Consider: I'm sorry ——





OK; let's unpack the blocks.

UNEMBED is easy.

$$\hat{y} = W_U h_N^{(L)} + b_U$$

logits $\langle V \times d_h \rangle$ $\langle V \rangle$

$$y = \text{SoftMax}(\hat{y})$$

distribution over vocab

Recall that

$$\text{SoftMax}(y)_i = \frac{\exp\{y_i\}}{\sum_j^V \exp\{y_j\}}$$

Note: Could add Temp

$$[\exp\{y_i/t\}]/[\sum_j^V \exp\{y_j/t\}]$$

$\tau > 1 \rightarrow$ More random

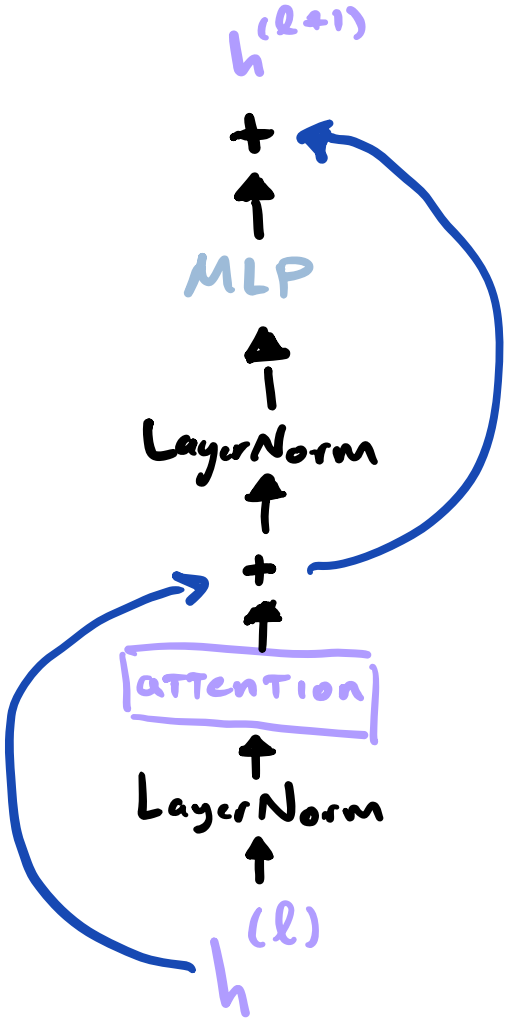
TRANSFORMER LAYER is more involved.

Two components:

(Self) attention

and MLP (with

Skip Connections)



Self-Attention

Intuition: Update $h_i^{(l)}$ given Context

Map all $h_i^{(l)}$ to query, key, & values.

$a_{ij} = S(q_i, k_j)$ $\quad \alpha_i = \text{SoftMax}(a_i)$

Attn dist. for token i.

$$\bar{h}_i = \sum_j \alpha_{ij} v_j$$

attend

Self $\quad S(q_{attend}, K_{self})$

attend $\quad S(q_{attend}, K_{attend})$

to $\quad S(q_{attend}, K_{to})$

Words $\quad S(q_{attend}, K_{words})$

$\rightarrow SM($ $)$

α^{attend}

$$\tilde{x}_{attend} = \sum_j \alpha_j^{attend} v_j$$

For auto regressive generation we use Causal Masking, setting attention on future tokens to 0.

A worked example of Self Attention

"The cat sat" $\quad d=2$

	q	k	v
The	[1 0]	[0 1]	[1 1]
Cat	[0 1]	[2 0]	[.2 0]
Sat	[2 0]	[0 1]	[0 2]

$$S_{ij} = q_i k_j / \sqrt{2}$$

$$S = \begin{matrix} \\ \text{the} \\ \text{Cat} \\ \text{Sat} \end{matrix} \begin{matrix} \begin{matrix} \text{The} & \text{Cat} & \text{Sat} \end{matrix} \\ \begin{bmatrix} 0 & 1.414 & 0 \\ .707 & 0 & .707 \\ 0 & 2.828 & 0 \end{bmatrix} \end{matrix}$$

$2/\sqrt{2}$

So this is NOT Symmetric!

k says what a Token "offers"; q What it cares about; V the signal.

Soft Max per row:

$$\alpha = \begin{matrix} \\ \text{The} \\ \text{cat} \\ \text{Sat} \end{matrix} \begin{matrix} \begin{matrix} \text{Th} & \text{Cat} & \text{Sat} \end{matrix} \\ \begin{bmatrix} .164 & .673 & .164 \\ .401 & .198 & .401 \\ .053 & .894 & .053 \end{bmatrix} \end{matrix}$$

$$\bar{h}_i = \sum_j \alpha_{ij} V_j = \begin{bmatrix} .298 & .491 \\ .441 & 1.203 \\ .272 & .159 \end{bmatrix}$$

$\bar{h}_3$ is interesting because most attn on "Cat", but all of 2nd dim from elsewhere

Causal mask version:

$$\alpha = \begin{array}{c} \\ \text{The} \\ \text{cat} \\ \text{sat} \end{array} \begin{array}{c} \begin{array}{ccc} \text{the} & \text{cat} & \text{sat} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ .67 & .33 & 0 \\ .05 & .89 & .05 \end{bmatrix} \end{array}$$

off diag's are Zero'd!

In Multi-headed Attention we define H attention heads with independent params W_q^h, W_k^h, W_v^h.

Outputs are combined ($\oplus$).

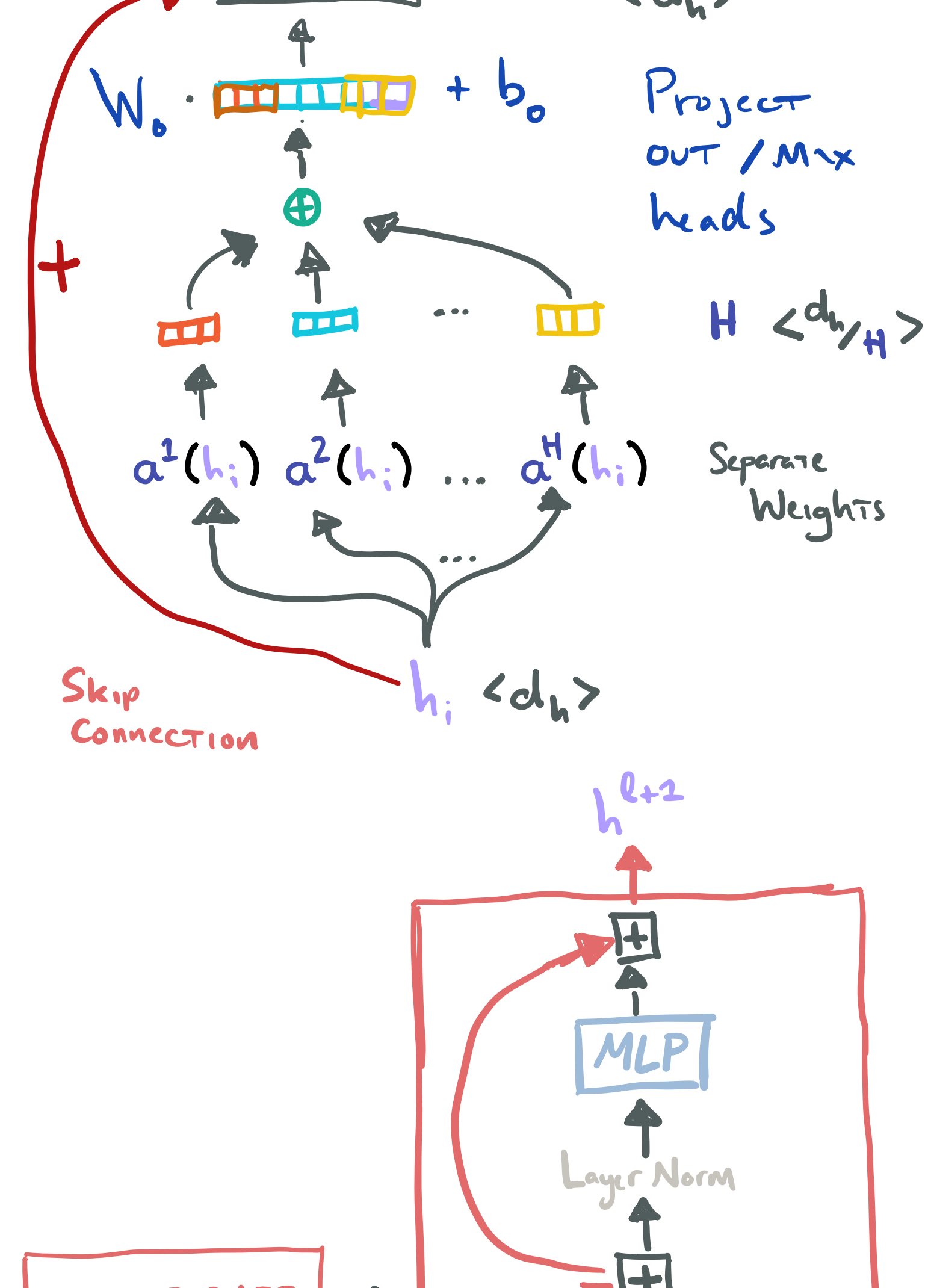
W_q^h, W_k^h, W_v^h all have shape

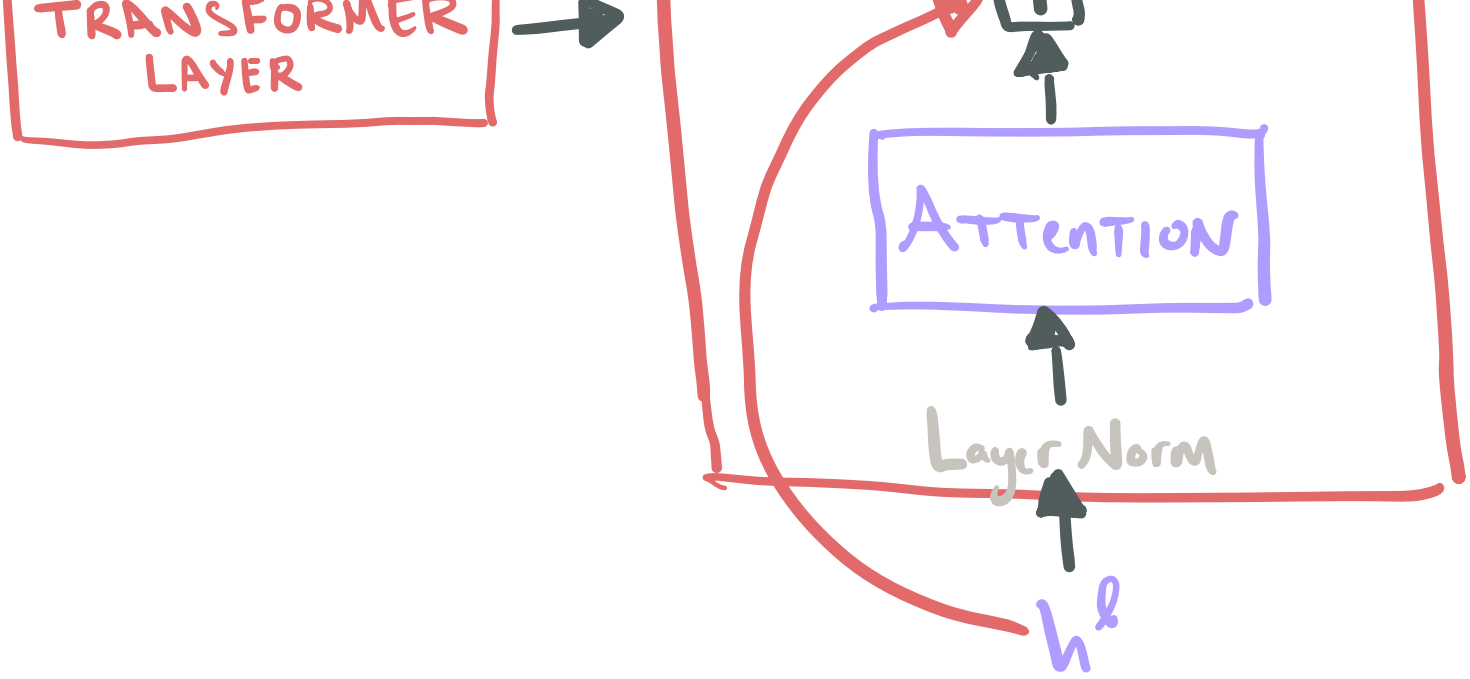
$$\langle d_h \times d_k \rangle$$

Where $d_k = d_h / H$.

Can view This as H transformations




$W_o \cdot$ + b_o
Project out / Mix heads
H $\langle d_h/H \rangle$
$a^1(h_i)$ $a^2(h_i)$ ... $a^H(h_i)$
Separate Weights
Skip Connection
h_i $\langle d_h \rangle$
h^{l+2}
MLP
Layer Norm



LayerNorm keeps magnitude of hidden states under control, which eases optimization

i is token position

$$LN(h_i)_K = \gamma_K \frac{h_{iK} - \mu_i}{\sqrt{\sigma_i^2 + \epsilon}} + \beta_K$$

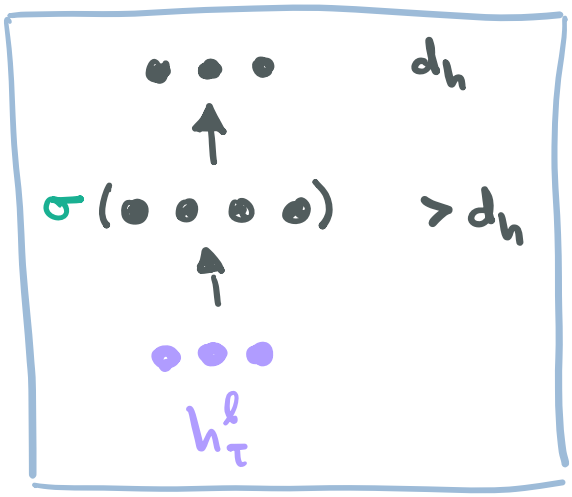
K: hidden dimension!

learned

μ_i and σ_i^2 are feature value mean and variances for token i (over h_i)

→ after, h_i has mean $\emptyset$, Var ~ 1

MLP is just a position wise feed forward Network



Note: Can think of

Implications for Interpretability

- **Attention Heads** or mix tokens **ROUTE** info
- **MLPs** Transform individual tokens (nonlinear)