Transformers II

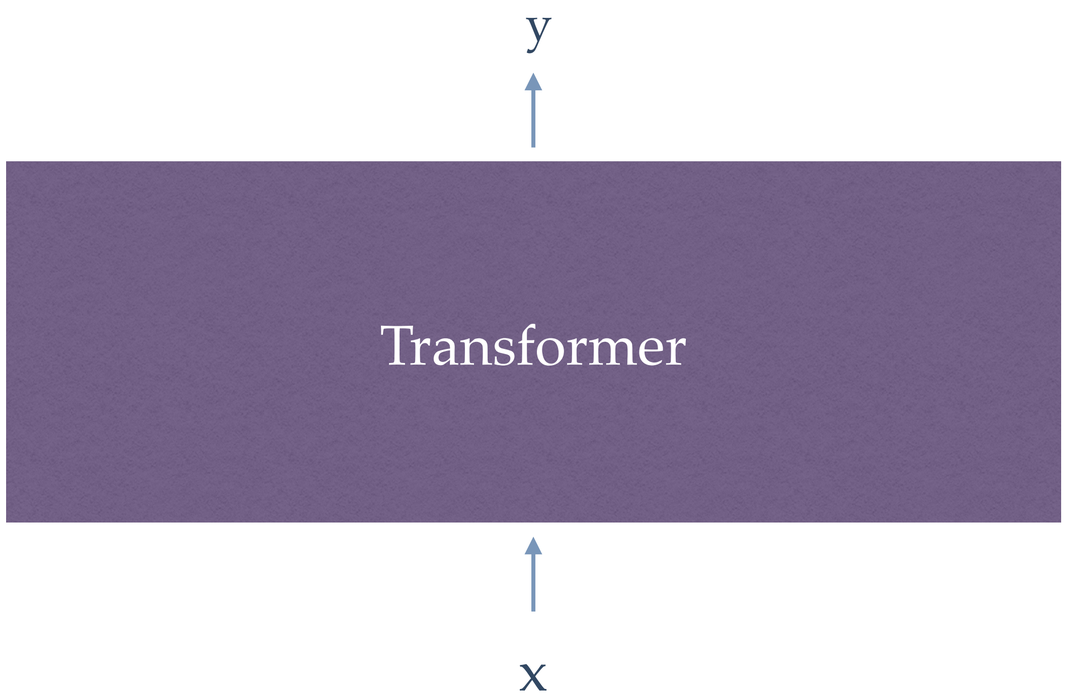
Sandeep Soni

03/05/2024

Story so far

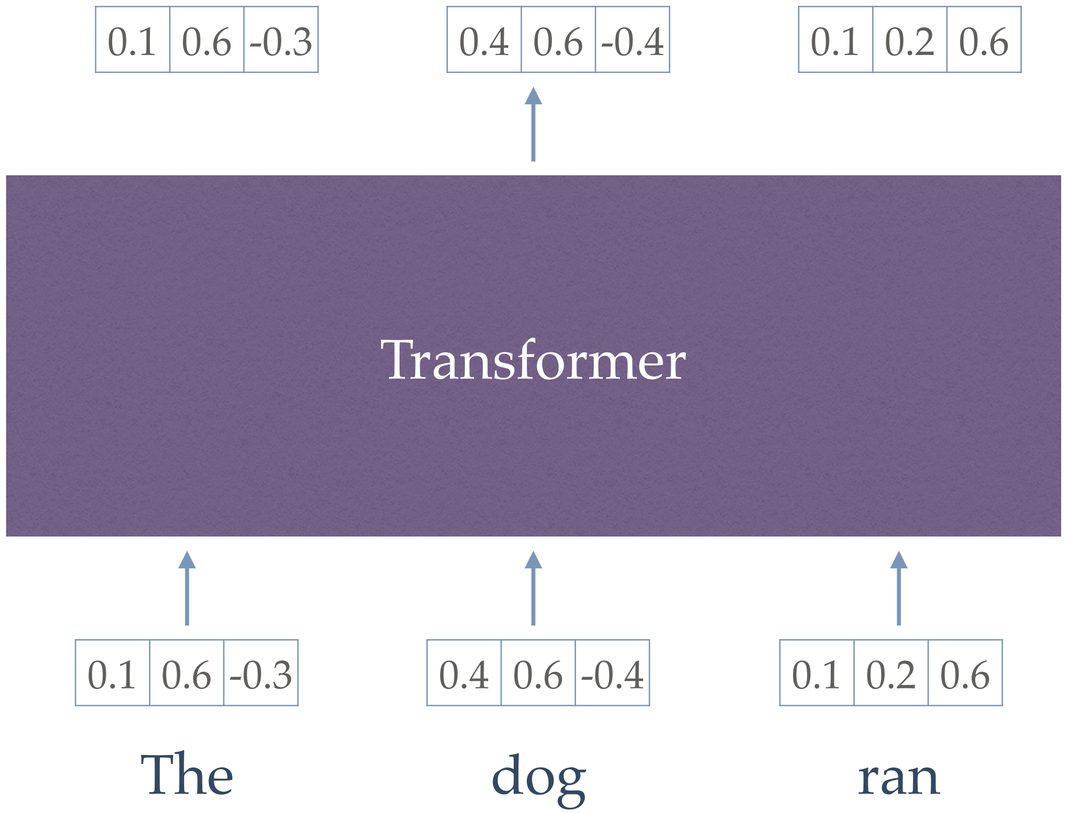
- Language model estimates P(x) for a sequence x
- Count and normalize language models (e.g., N-gram LMs) assume fixed-length conditioning context
- RNN LM can condition on arbitrarily long context but has sequential computation

Transformers



- Transform an input sequence of vectors into an output sequence of vectors

Transformers

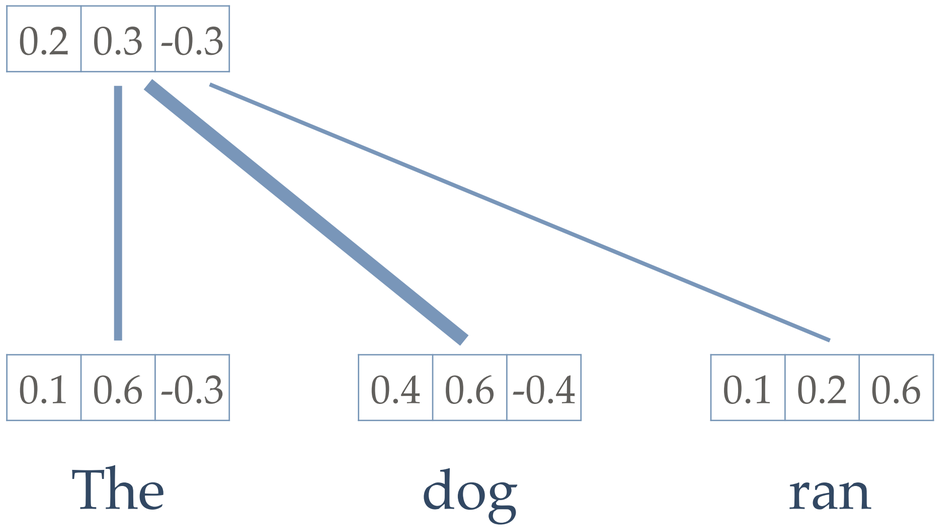


- Transform an input sequence of vectors into an output sequence of vectors

Key ideas

- A vector associated for every token in a sequence that depends on every other token in the sequence.
- Multiple transformations of vector sequences.
- Parallel computation

Self-Attention



- Create a token vector as a weighted combination of input token vectors
- Weights are the importance or attention to other tokens

0.2	0.3	-0.3	Value
0.1	0.2	-0.3	Key
0.3	0.1	-0.5	Query
0.1	0.6	-0.3	Previous value

The

$v \in \mathbb{R}^{768} = eW_V$

$k \in \mathbb{R}^{96} = eW_K$

$q \in \mathbb{R}^{96} = eW_Q$

$e \in \mathbb{R}^{768}$

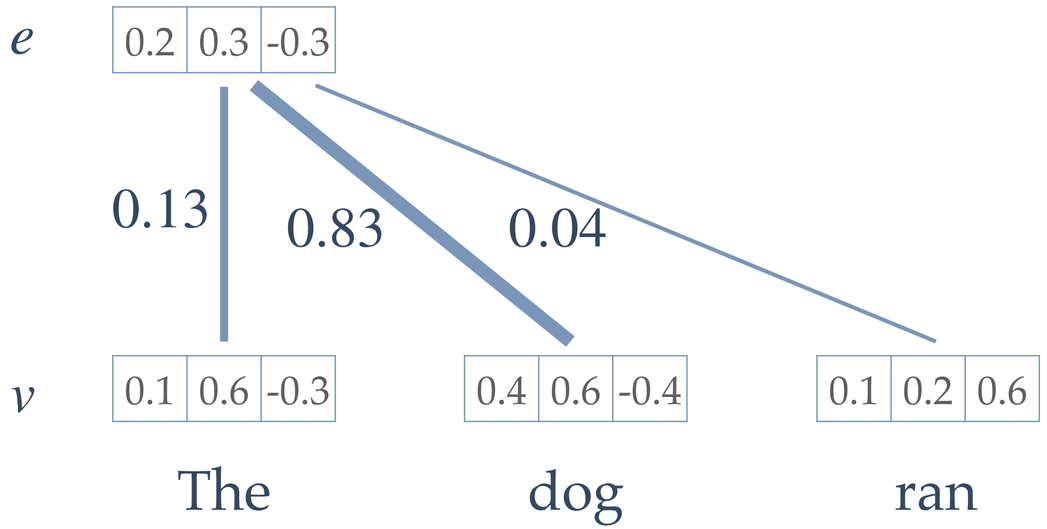
$W_V \in \mathbb{R}^{768\times768}$

$W_K \in \mathbb{R}^{768\times96}$

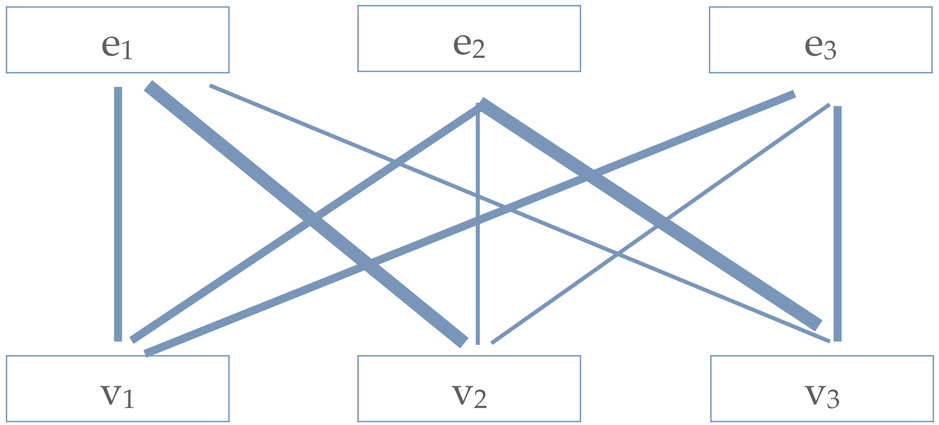
$W_Q \in \mathbb{R}^{768\times96}$

$$a_{1j} = \text{softmax}(\text{dot}(q_1, k_j))$$

0.13	0.83	0.04	Attention weights
0.3	1.9	-0.7	Compatibility scores
$q_1 \cdot k_1$	$q_1 \cdot k_2$	$q_1 \cdot k_3$	
0.1 0.6 -0.3	0.4 0.6 -0.4	0.1 0.2 0.6	
The	dog	ran	

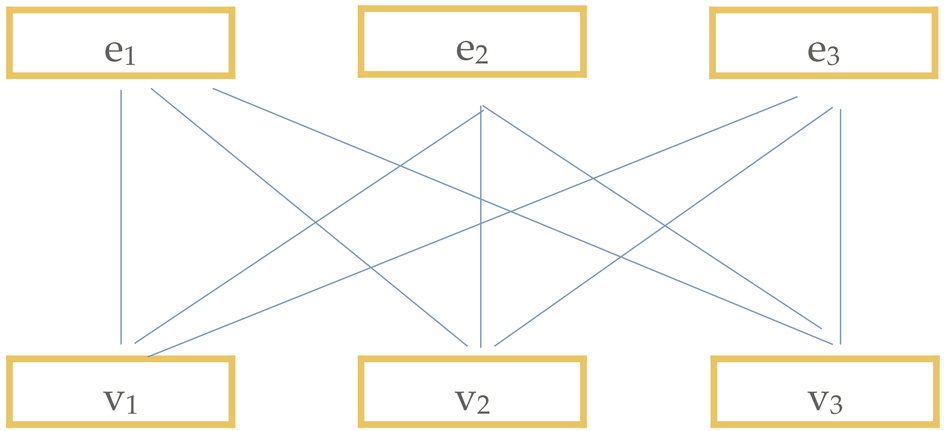


- Query and key vectors are used to calculate compatibility or attention scores.
- Attention weights are used to combine value vectors to create a new vector



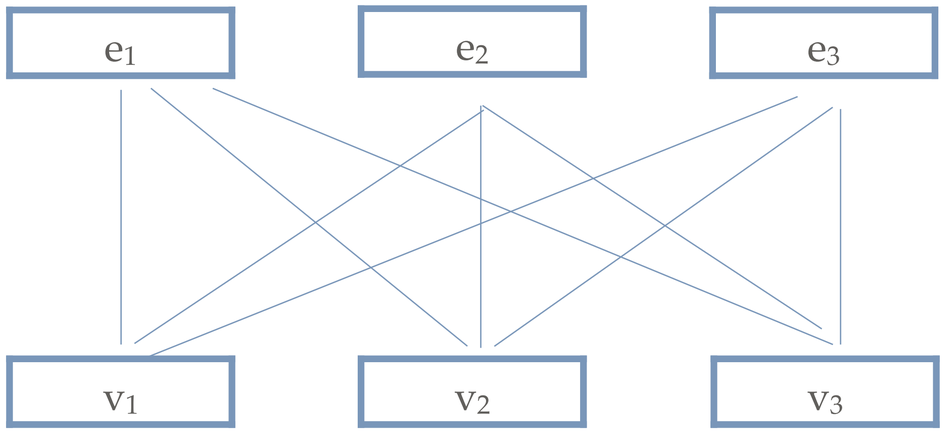
- Compatibility for all the pairs in a sequence is calculated

Multi-head Attention



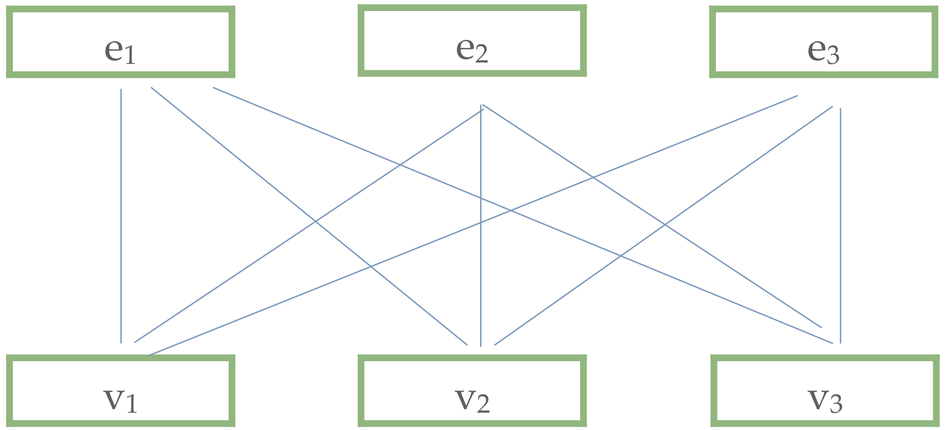
- Each attention head can have its own query, key, and value vectors such that every attention head gives one view of the data

Multi-head Attention



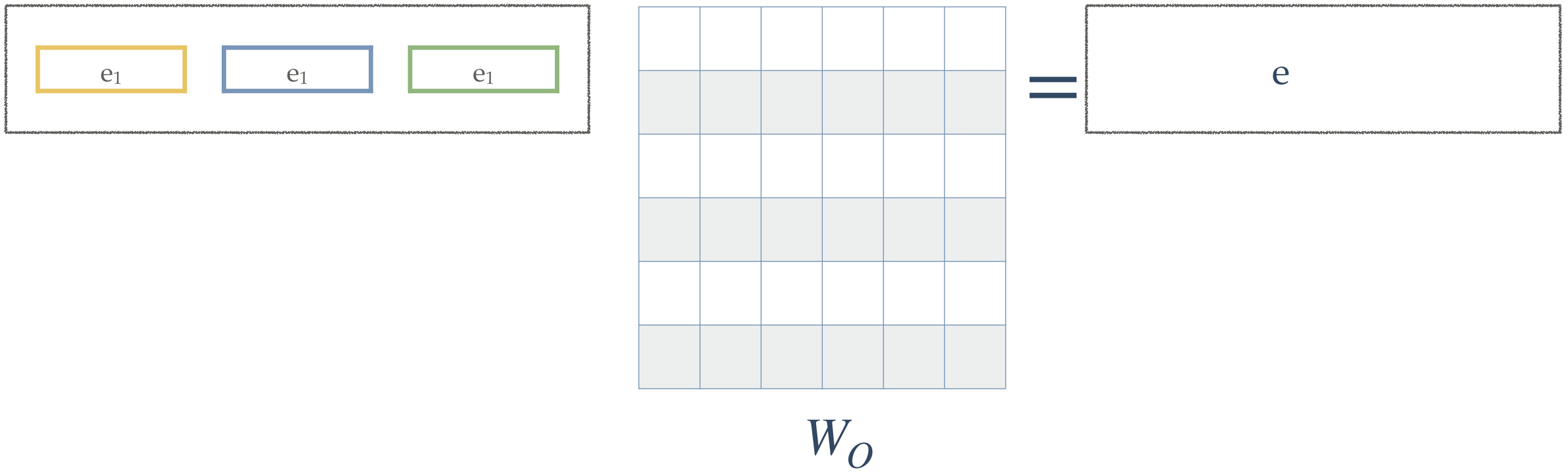
- Each attention head can have its own query, key, and value vectors such that every attention head gives one view of the data

Multi-head Attention

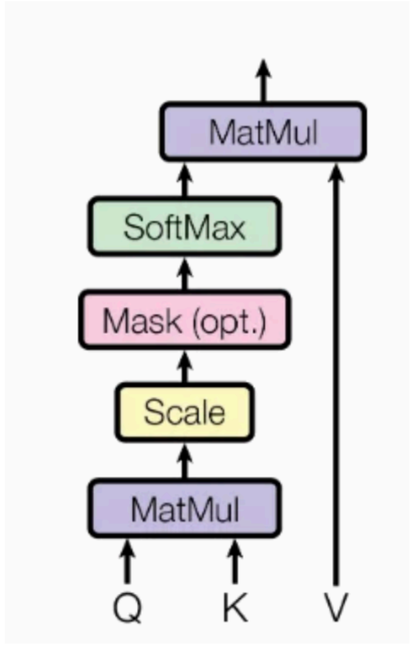


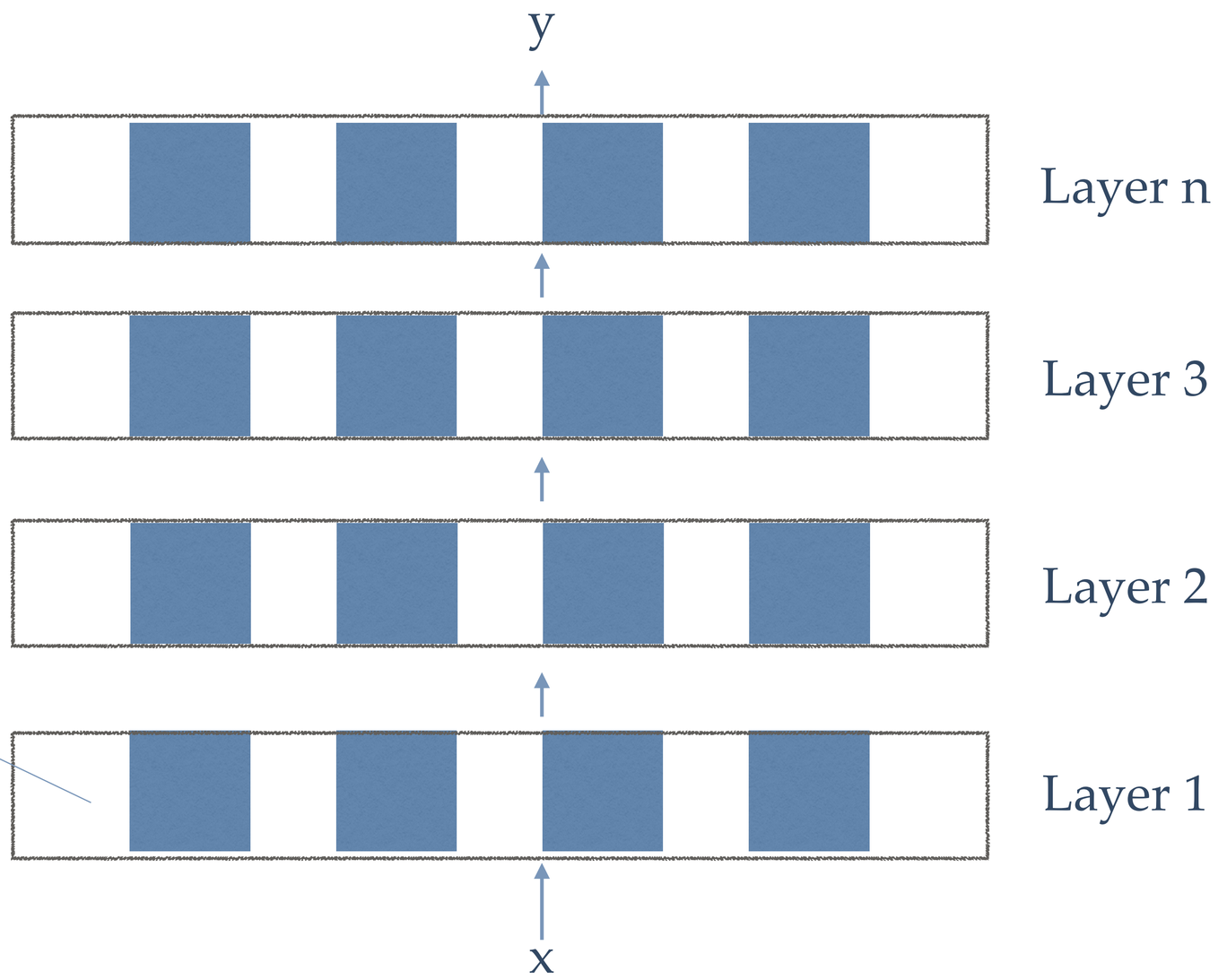
- Each attention head can have its own query, key, and value vectors such that every attention head gives one view of the data

Multi-head Attention



Single head attention





Self attention units
within an attention block

Transformers

- There are some more transformations within an attention block (layer normalization, feedforward network)
- To account for the position of each token, positional embeddings are used in place to augment token embeddings

LARGE LANGUAGE MODELS

- Most modern LLMs are based on the transformer architecture (e.g., BERT, GPT, etc)
- Many layers, high dimensional vector representations, and large corpora for pretraining are all hallmarks of contemporary LLMs

Variations of Language modeling

Masked LM:

$$P(x) = \prod_{i=1}^{n} P(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$

Encoder-Decoder LM:

$$P(y) = \prod_{i=1}^{n} P(y_i \mid y_1, \ldots, y_{i-1}, x)$$

Causal LM:

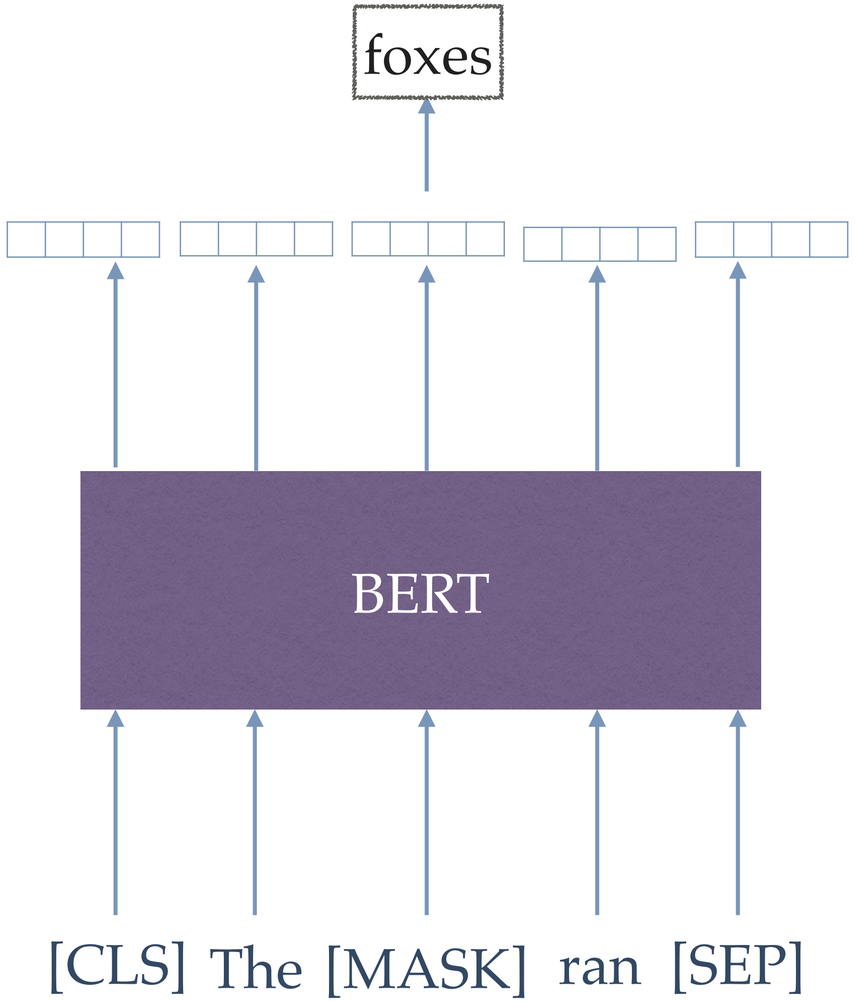
$$P(x) = \prod_{i=1}^{n} P(x_i \mid x_1, x_2, \ldots, x_{i-1})$$

Masked Language modeling

Fill in the blank by using the surrounding context

The _____ ran

$P(w_t \mid \neg w_t)$



BERT is a language model trained to predict the missing word

BERT

- Masked LM $$P(x) = \prod_{i=1}^{n} P(x_i \mid x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$$
- The ____ ran → BERT → foxes
- Transformer based model
  - trained on 512 sized contexts
  - 12 or 24 transformer blocks
  - 768 or 1024 sized representations
  - pretrained on billions of tokens from wikipedia and books

T5

- Encoder-Decoder LM: $P(y) = \prod_{i=1}^{n} P(y_i | y_1, \ldots, y_{i-1}, x)$
- The ~~foxes ran~~ in the forest ~~after~~ seeing the humans

- Pretrained on 750GB of web text

GPT

- Causal LM: $$P(x) = \prod_{i=1}^{n} P(x_i \mid x_1, x_2, \ldots, x_{i-1})$$
- Also called as left-to-right or autoregressive language modeling
- Fill in the blank but always predict the next word in the sequence (e.g., The foxes ___)
- GPT-3 has 175B parameters and trained on 570GB of web text, books, wikipedia