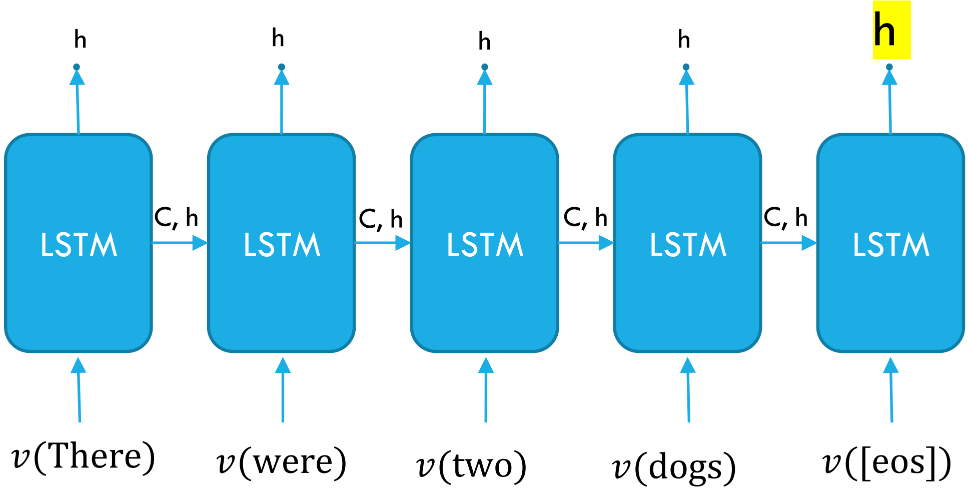
TRANSFORMER AND NLP

Topics of Data Engineering

TRANSFORMER

SEQUENTIAL INPUT TO LSTM

- Word embedding vectors are sequentially input to the LSTM model
- The output vector for the sequential data is obtained after eos (end of sentence) is input
- *HOWEVER, LSTM tends to forget past inputs…*



THE REASON LSTM TENDS TO FORGET PAST INPUTS

- ✓ The past inputs $\widetilde{\boldsymbol{C}}_{t-1}, \widetilde{\boldsymbol{C}}_{t-2}, \cdots$ are multiplied to the factors

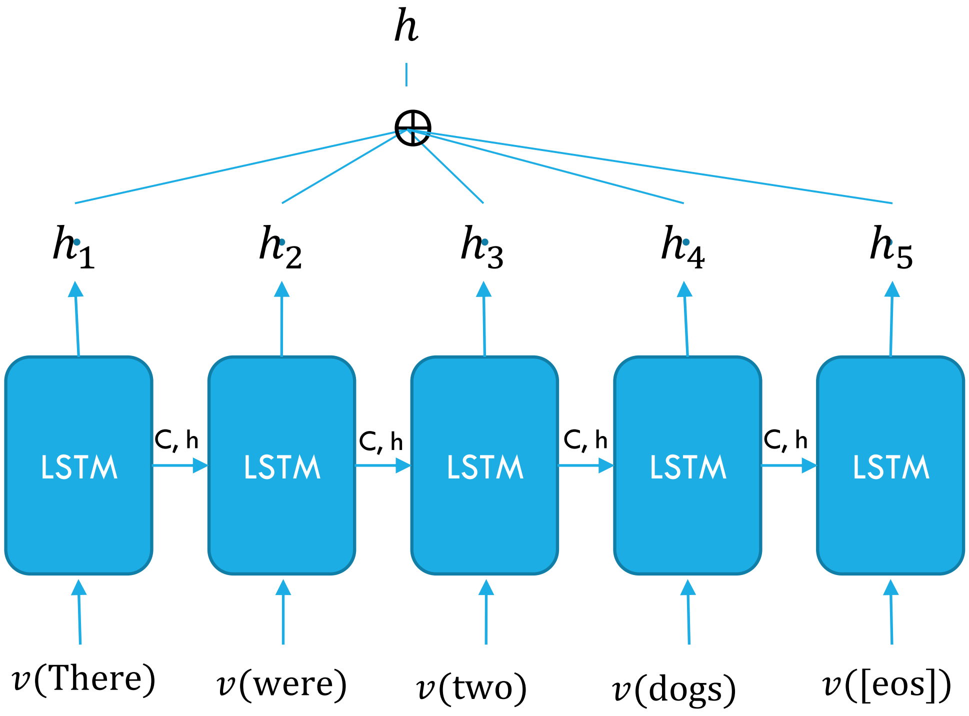
$$\prod_{\boldsymbol{k}} \boldsymbol{f}_{\boldsymbol{t}-\boldsymbol{k}} = \boldsymbol{f}_{\boldsymbol{t}} * \boldsymbol{f}_{\boldsymbol{t}-\mathbf{1}} * \boldsymbol{f}_{\boldsymbol{t}-\mathbf{2}} * \cdots$$

- ✓ The factors decrease the contribution of past inputs to $\boldsymbol{C}_t$
  - ✓ Because of the property of sigmoid function, $0 < \boldsymbol{f}_{\boldsymbol{t}-\boldsymbol{k}} < \mathbf{1}$

$$\begin{aligned}\boldsymbol{C}_t &= \boldsymbol{f}_{\boldsymbol{t}} * \boldsymbol{C}_{t-1} + \boldsymbol{i}_{\boldsymbol{t}} * \widetilde{\boldsymbol{C}}_t \\ &= \boldsymbol{f}_{\boldsymbol{t}} * (\boldsymbol{f}_{\boldsymbol{t}-\mathbf{1}} * \boldsymbol{C}_{t-2} + \boldsymbol{i}_{\boldsymbol{t}-\mathbf{1}} * \widetilde{\boldsymbol{C}}_{t-1}) + \boldsymbol{i}_{\boldsymbol{t}} * \widetilde{\boldsymbol{C}}_t \\ &= \boldsymbol{f}_{\boldsymbol{t}} * (\boldsymbol{f}_{\boldsymbol{t}-\mathbf{1}} * (\boldsymbol{f}_{\boldsymbol{t}-\mathbf{2}} * \boldsymbol{C}_{t-3} + \boldsymbol{i}_{\boldsymbol{t}-\mathbf{2}} * \widetilde{\boldsymbol{C}}_{t-2}) \\ &\qquad + \boldsymbol{i}_{\boldsymbol{t}-\mathbf{1}} * \widetilde{\boldsymbol{C}}_{t-1}) + \boldsymbol{i}_{\boldsymbol{t}} * \widetilde{\boldsymbol{C}}_t\end{aligned}$$

ATTENTION MECHANISM

The output of LSTM for each step is weighted and summed to get an output that covers the whole of steps

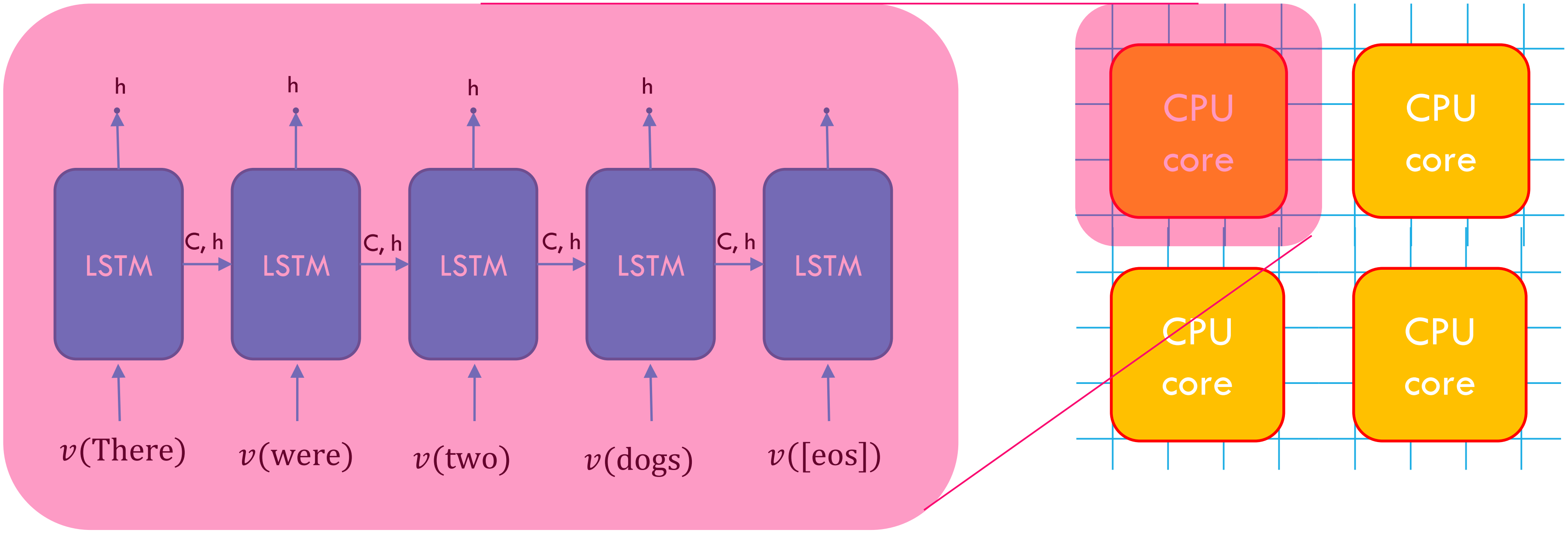


$$h = \sum_i \alpha_i h_i$$

$$\alpha_i = < h_i, h_5 >$$

FROM VIEWPOINT CALCULATION EFFICIENCY…

- ✓For efficient calculation, parallelization is necessary
- ✓LSTM usually causes idle time of CPU/GPU
  - ✓LSTM sequentially calculates the cell state vector $\boldsymbol{C}_t$
  - ✓Parallelization cannot be applied to such sequential calculation



ATTENTION IS ALL YOU NEED

✓In 2017, Google team published a paper, “Attention is all you need”.

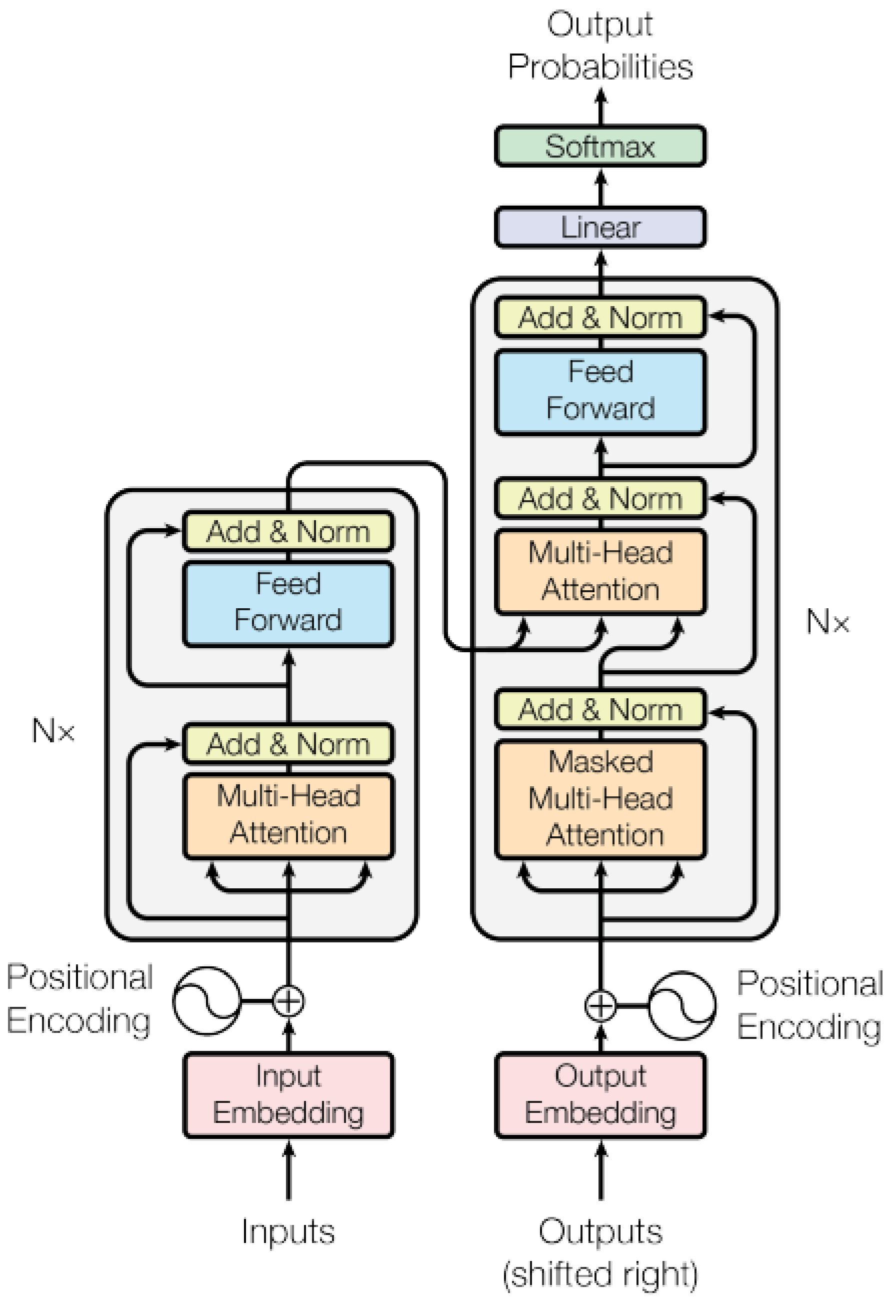
✓It was the first paper that proposed Transformer.

> The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. …

Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

STRUCTURE OF TRANSFORMER

- ✓Transformer consists of the encoder and the decoder part
- ✓It only has multi-head attention parts and feed forward (multilayer perceptron) parts
- ✓The order of sequence is realized by positional encoding



Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

SCALED DOT-PRODUCT SELF-ATTENTION

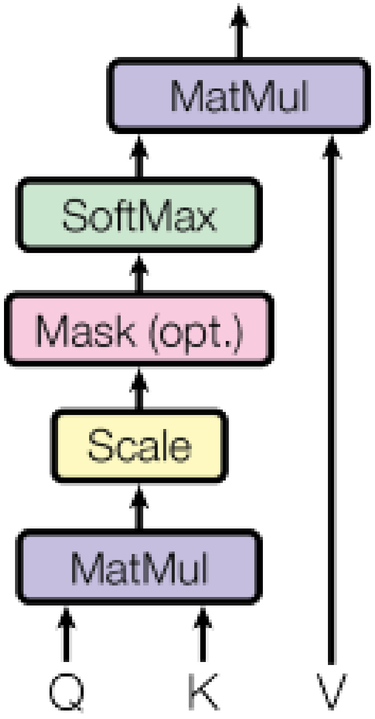
✓Closed the idea $h = \sum_i < h_i, h_5 > h_i$.

✓For the input X, Query $Q = XW_q$, Key $K = XW_k$, Vale $\mathrm{V} = XW_w$ are used

✓The attention is given as

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V$$

where d is the dimension of the key

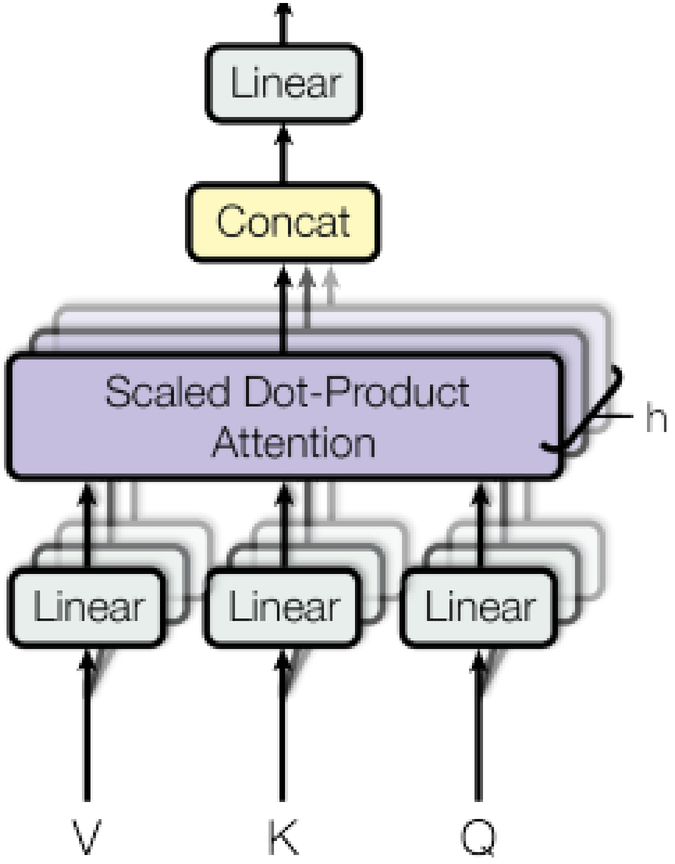


Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

MULTI-HEAD ATTENTION

✓For parallelization, the dot-product attention is divided into *heads*

✓For heads, Q, K and V are linearly projected

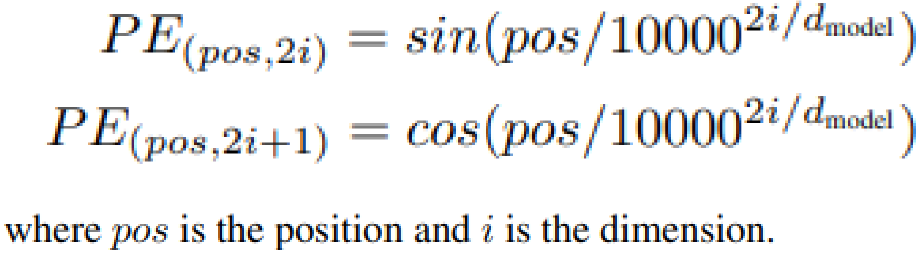


$$\mathrm{MultiHead}(Q, K, V) = \mathrm{Concat}(\mathrm{head}_1, ..., \mathrm{head}_\mathrm{h})W^O$$
$$\text{where } \mathrm{head}_\mathrm{i} = \mathrm{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Where the projections are parameter matrices $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}$, $W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$.

POSITIONAL ENCODING

✓Dot-product attention and multi-head attention consist of linear combination and inner product, which do not take account of the inputs' sequence order



$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

where pos is the position and i is the dimension.

For inputs $X_1, X_2, \cdots, X_{pos}, \cdots$

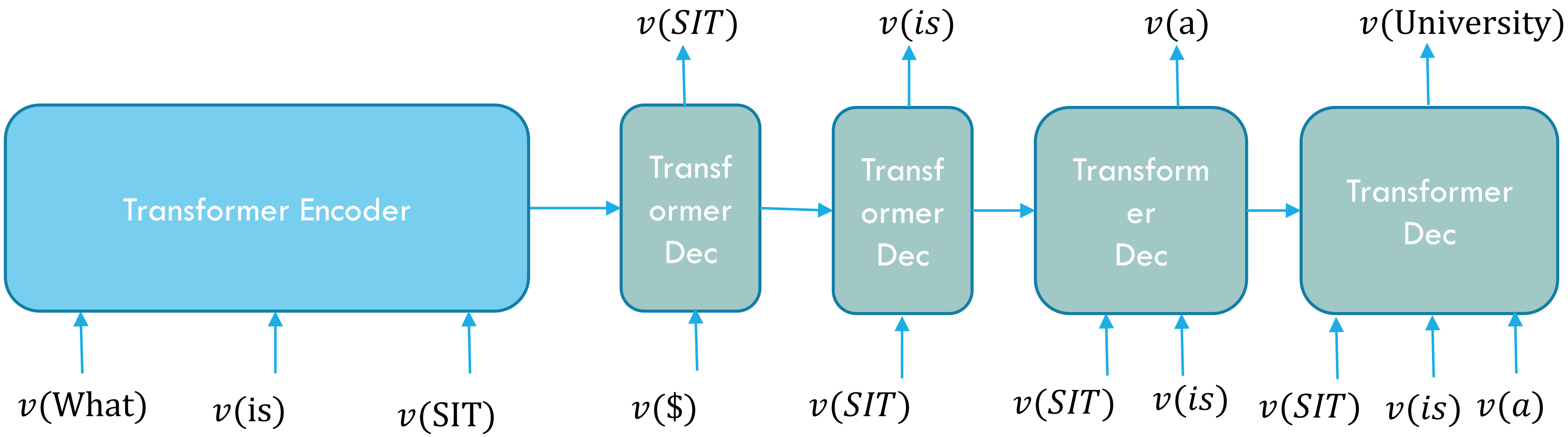
$$PE_{pos} = \left(PE_{pos,1}, PE_{pos,2}, \cdots, PE_{pos,d_{model}}\right)^T$$
$$X_{pos} = X_{pos} + PE_{pos}$$

Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems* 30 (2017).

GPT

✓The idea of GPT is similar to LSTM encoder/decoder model.

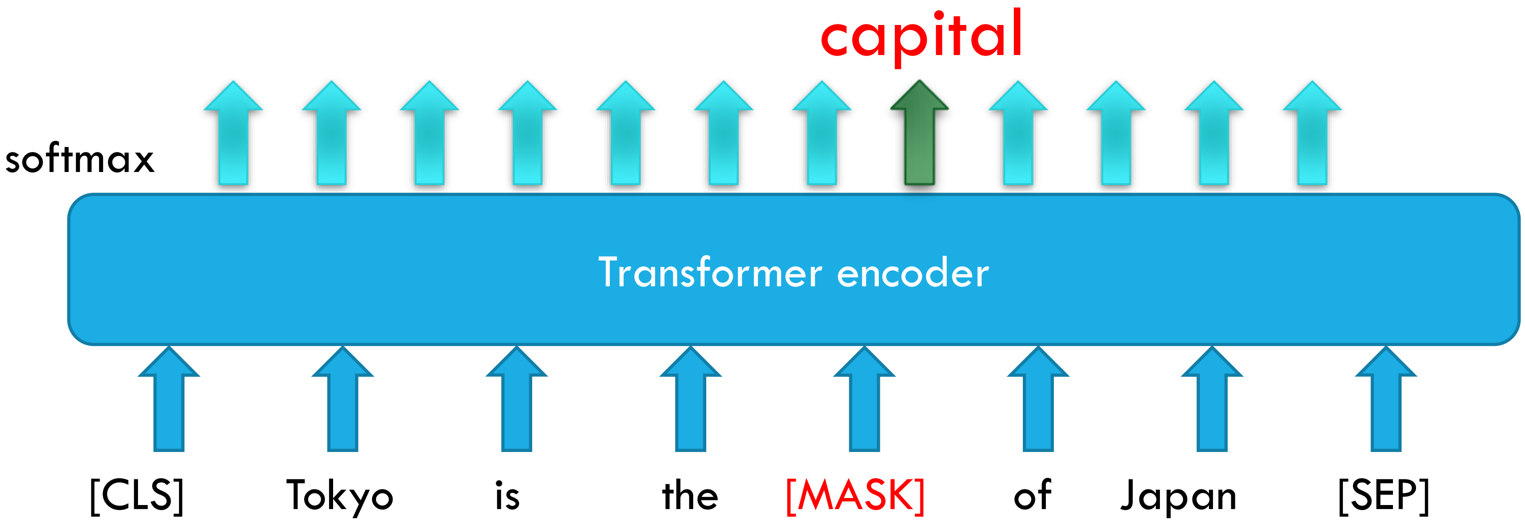
✓After input some words to the transformer encoder, its decoder sequentially outputs words suitable as output sentences.



http://jalammar.github.io/illustrated-gpt2/
https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

BERT

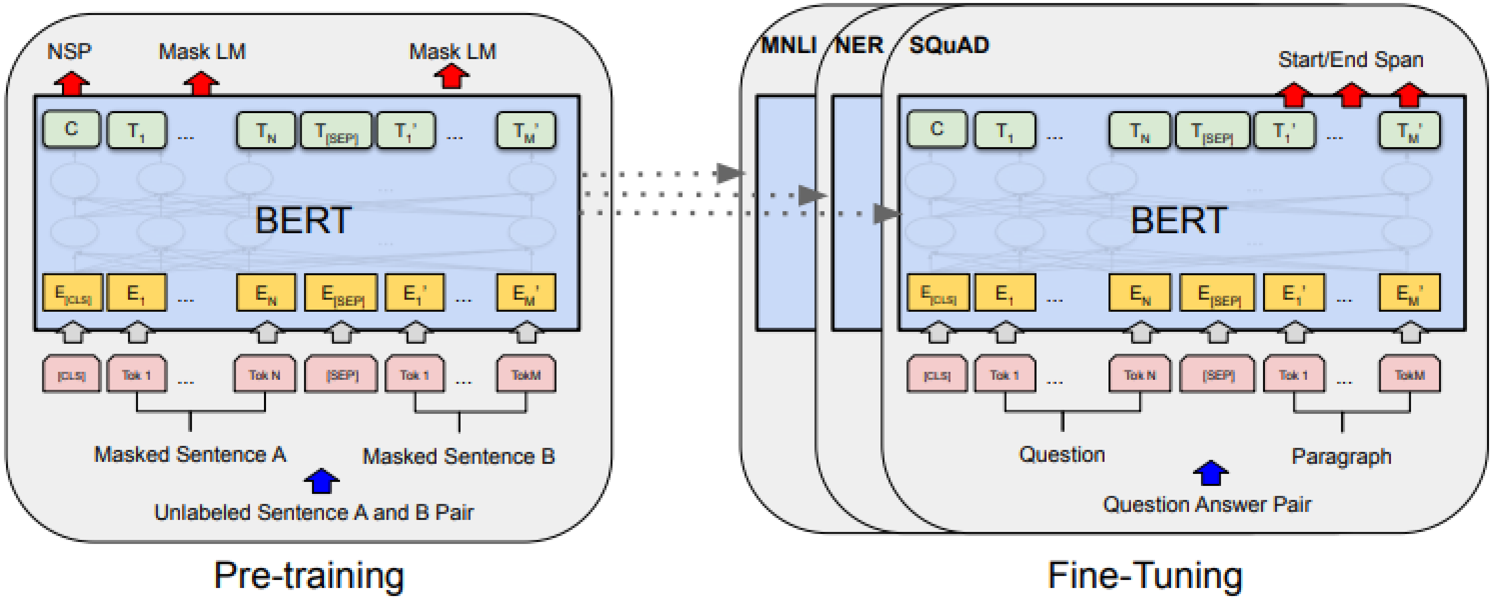
- ✓Using masked language model (MLM), the transformer is trained to output as pre-training phase
- ✓Form MLM, words in input sentences are randomly masked and transformer is trained to output the corresponding word
- ✓Next sentence prediction is the pre-training task for the sentence pairs



BERT(CONT'D)

✓The pre-training phase is followed by fine-tuning phase

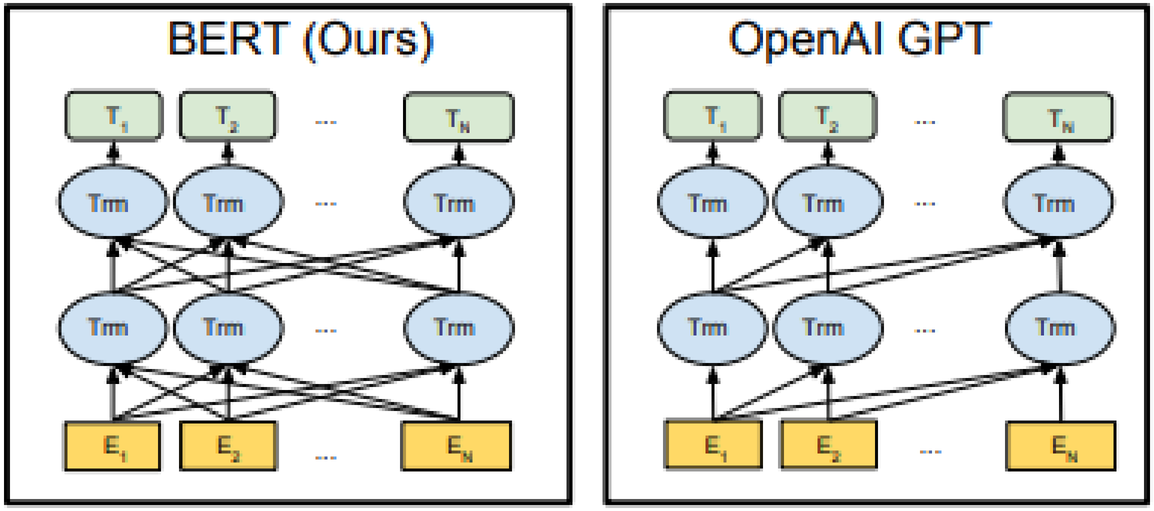
✓ In the fine-tuning phase, the BERT model is embedded to the target model and slightly trained with the target dataset



COMPARISON BETWEEN GPT AND BART

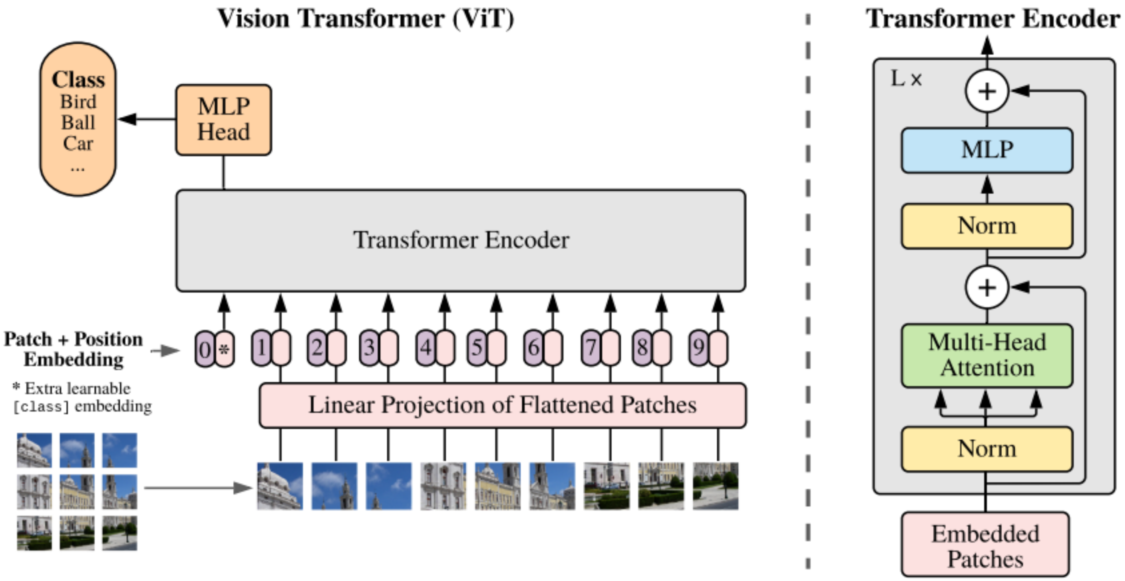
✓GPT predicts output words only based on the left appearing words

✓BERT used both left and right neighbor words to get outputs



VISION TRANSFORMER

- ✓Transformer is not limited to the use of natural language processing
- ✓Instead of word embedding vectors, separated pieces of an image is input to a transformer encoder



Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." *arXiv preprint arXiv:2010.11929* (2020).

XML

XML IS …

eXtensible Markup Language

- whose origin is SGML

in a text format

- XML uses tags as metadata of character strings
  - One can freely define the names of tags
  - Tags need to be closed if they are opened
  - It is possible to assign attributes to tags
- XML is useful to exchange data, since textfiles can be read in any OS.

IS HTML A XML DOCUMENT? - NO

Both HTML and XML are markup languages using tags

- Their origin is common, SGML

A relationship between tags and content strings therein

- Tags of HTML are for display, not giving meaning
- Tags of XML are metadata to define contents' meaning

Constraints for tags

- In HTML, some tags, <P> and <BR>, are not required to be closed
- In XML, all tags need to be closed

ELEMENTS AND ATTRIBUTES

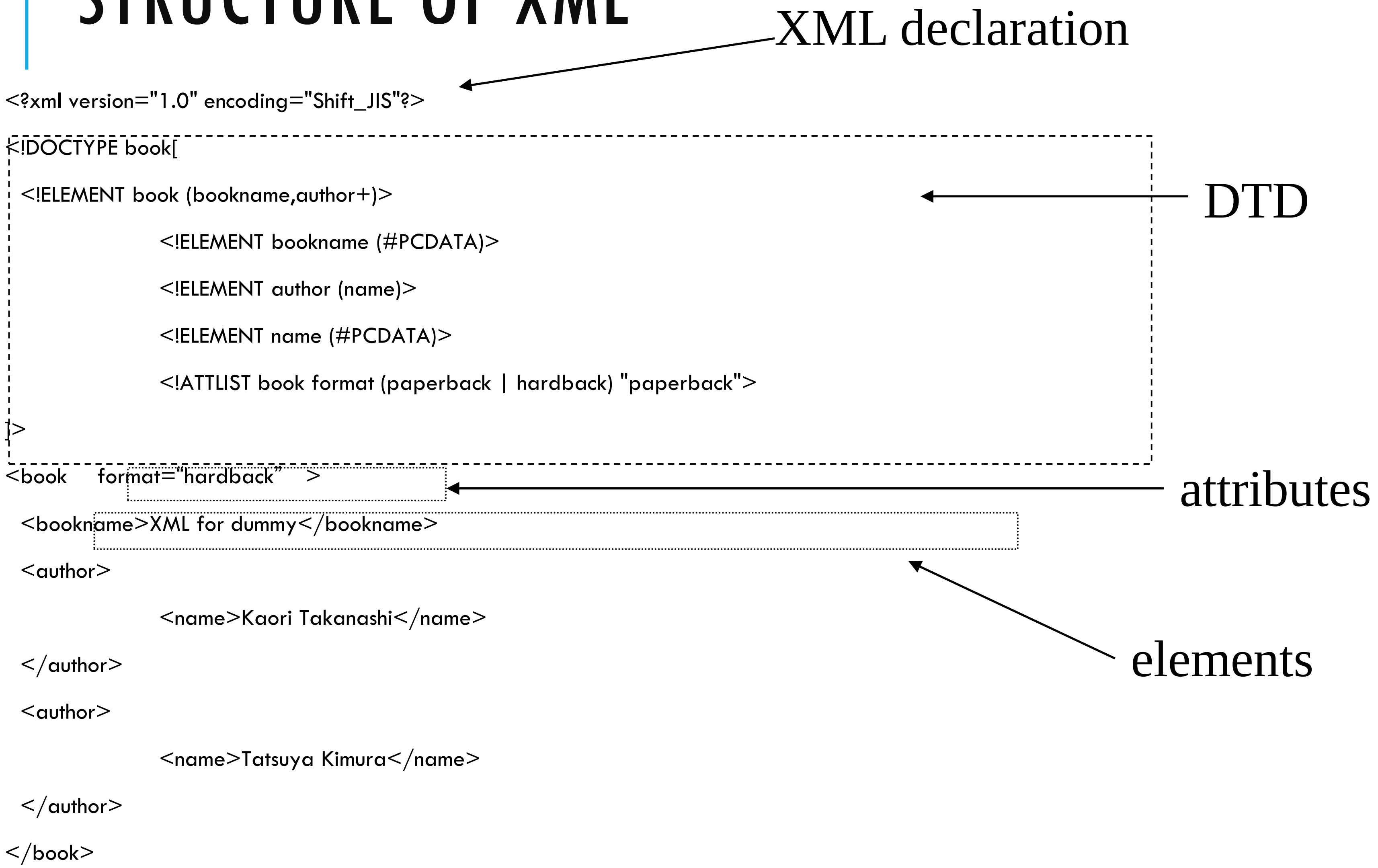
Elements

- A unit surrounded by a tag in a XML document
  - <name>Kimura</name>
- Empty element
  - <name />　=　<name></name>

Attributes

- Additional information in elements
- are included in start-tags
  - <name staff=“yes” >Kimura</name>

STRUCTURE OF XML



```
<?xml version="1.0" encoding="Shift_JIS"?>
<!DOCTYPE book[
  <!ELEMENT book (bookname,author+)>
                    <!ELEMENT bookname (#PCDATA)>
                    <!ELEMENT author (name)>
                    <!ELEMENT name (#PCDATA)>
                    <!ATTLIST book format (paperback | hardback) "paperback">
]>
<book     format=“hardback”    >
  <bookname>XML for dummy</bookname>
  <author>
                    <name>Kaori Takanashi</name>
  </author>
  <author>
                    <name>Tatsuya Kimura</name>
  </author>
</book>
```

XML declaration

DTD

attributes

elements

XML DECLARATION

The declaration that the document is XML

- necessary
- <?xml version=“1.0”?>

contains

- version
    - version=“1.0”
- encoding
    - encoding=“Shift_JIS”
    - if encoding is UTF-8, this is optional
- standalone or not
    - standalone=“no” (default)
    - optional

DTD

defines a structure of XML

<!DOCTYPE *root element* [

<!ELEMENT *element* （*child elements*)>

・・・

<!ATTLIST *element attribute value* ***default***>

・・・

]>