Department of Computational and Data Sciences

INDIAN INSTITUTE OF SCIENCE

भारतीय विज्ञान संस्थान

AI Module: Transformer Models

Deepak Subramani

Assistant Professor

Dept. of Computational and Data Science

Indian Institute of Science Bengaluru

Recap Quiz

1. The core idea of modern deep learning is
   - to process data and find representations that help in downstream tasks
   - to develop models with limited data
   - to use pytorch and tensorflow to develop models
   - to apply neural networks to all types of data
2. Which of the following is TRUE
   - CNNs form a representation of image data by extracting important features
   - Dense Layers are a more efficient feature extractor for images
   - Subword tokenization schemes like BPE need [UNK] tokens
   - BPE cannot be trained offline before text processing tasks

Outline for Week 02

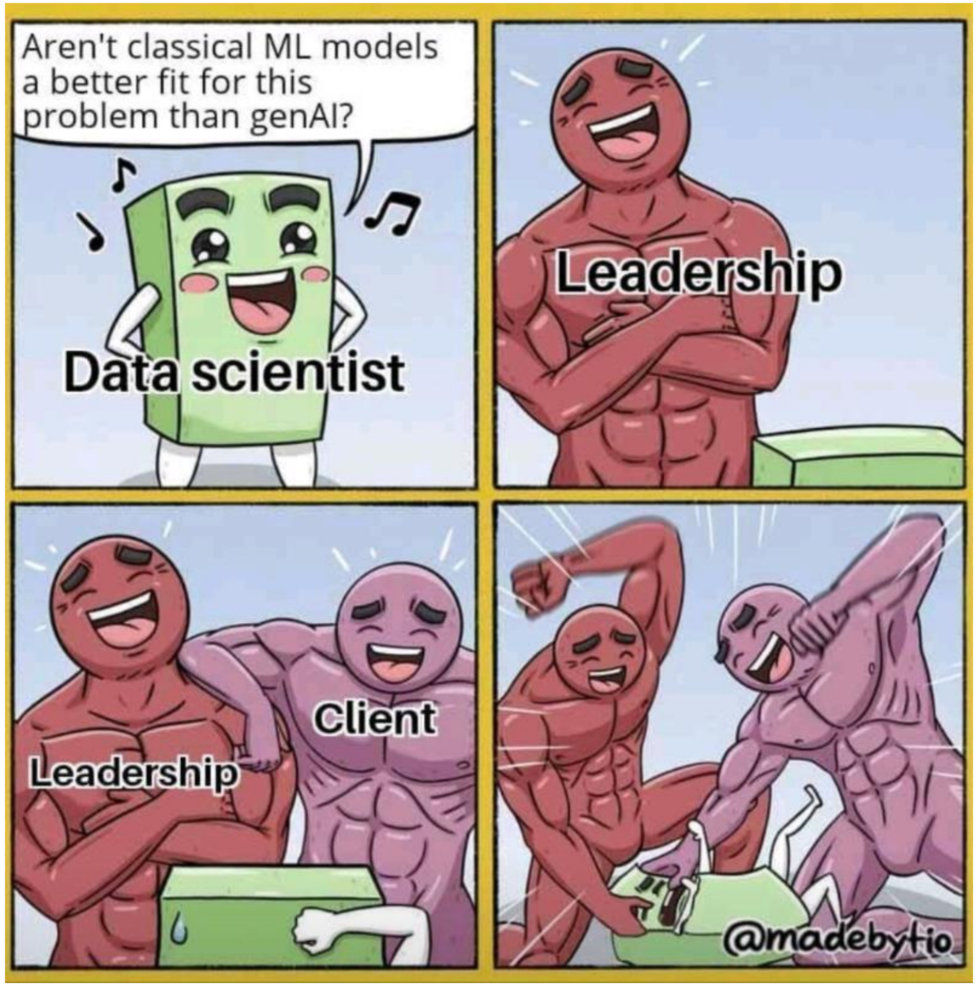
- Part 01: Sequence Modeling
  - NLP as a sequence modelling task
  - Names of recurrent architectures and introduction
  - Details in an additional slide deck on RNN – not included in the main content
    - Other Seq Tasks (Time Series Modeling; Not NLP)
    - Additional notebooks on RNN and LSTM (not compulsory)
- Part 02: Attention Mechanism and Transformer Encoders
  - Key, Query, Value
  - Multi Head Attention
  - MHA as text representation learners for use in discriminative tasks
  - Positional Embedding
  - Trick of the trade: Layer Normalization
  - Assignment on Attention Mechanism

Outline for Week 02

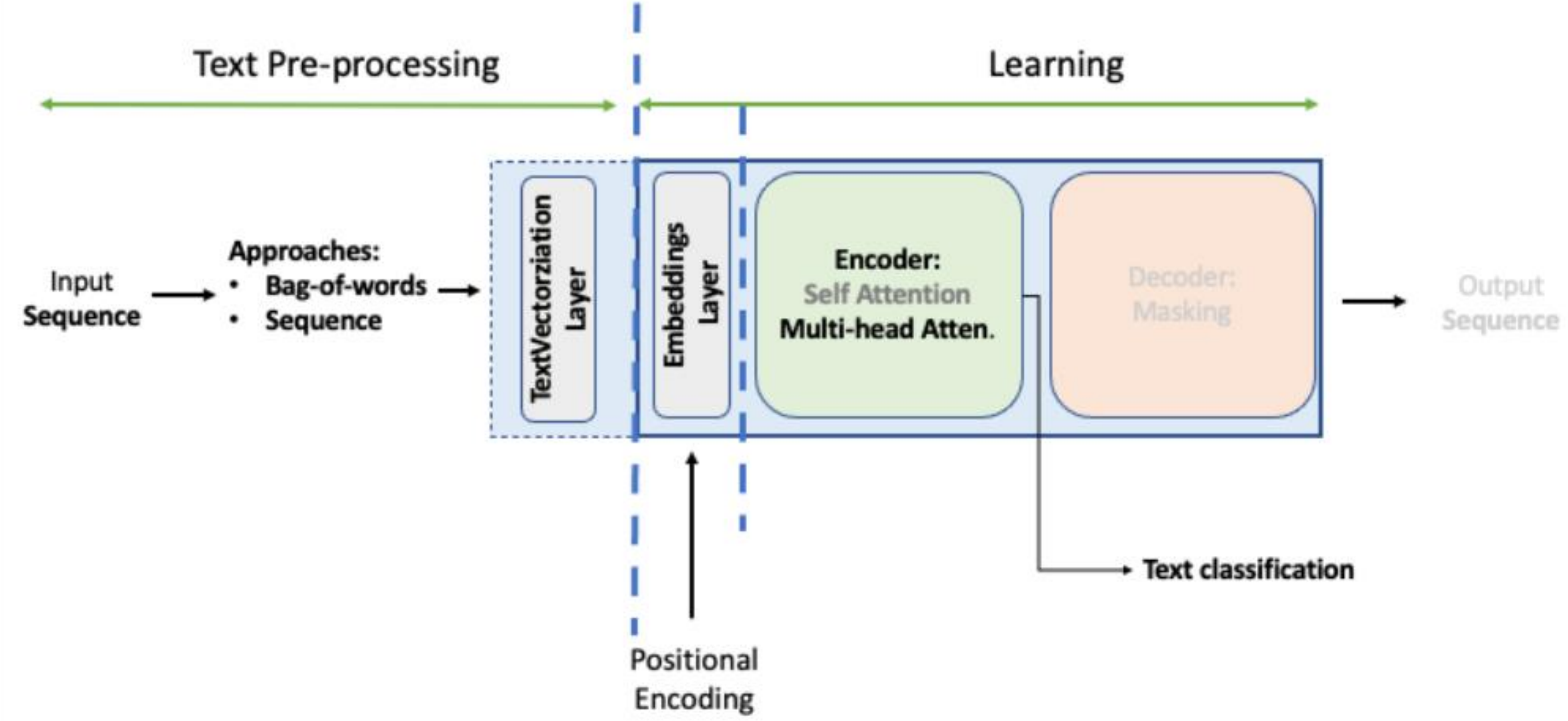
- Part 01: Sequence Modeling
  - NLP as a sequence modelling task
  - Names of recurrent architectures and introduction
  - Details in an additional slide deck on RNN – not included in the main content
    - Other Seq Tasks (Time Series Modeling; Not NLP)
    - Additional notebooks on RNN and LSTM (not compulsory)
- Part 02: Attention Mechanism and Transformer Encoders
  - Key, Query, Value
  - Multi Head Attention
  - MHA as text representation learners for use in discriminative tasks
  - Positional Embedding
  - Trick of the trade: Layer Normalization
  - Assignment on Attention Mechanism

Sequence Model Approach

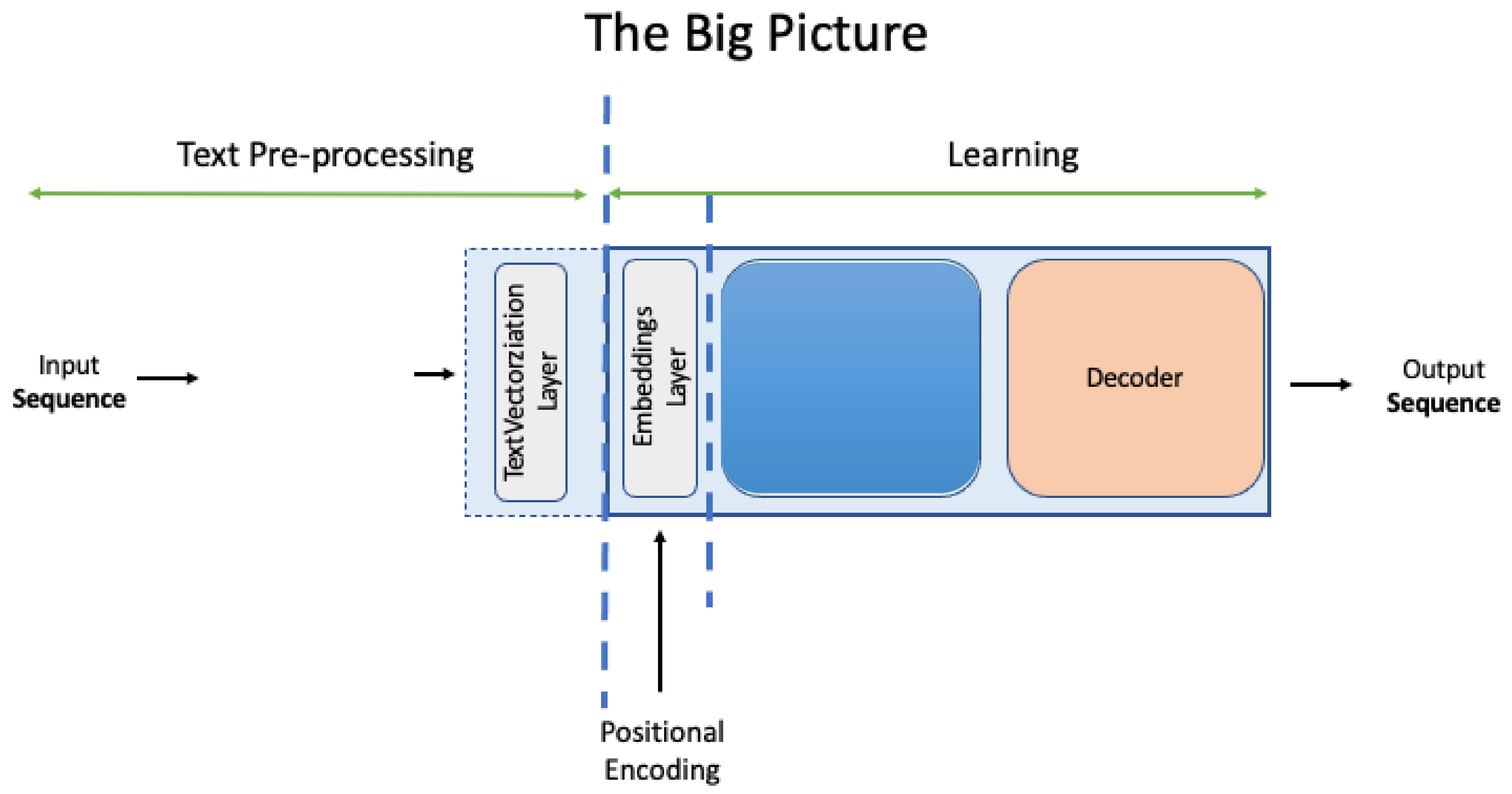
1. Represent input text as a sequence of integers (1 integer per word)
2. Map it into a vector space (one-hot embedding in vocab dim space projected to embed dimensional space)
3. Feed this sequence of vectors to a stack of layers capable of handling sequences (Transformer or RNN/LSTM/Bi-LSTM)


Aren't classical ML models a better fit for this problem than genAI?
Data scientist
Leadership
Leadership
Client
@madebytio

Text Processing Schematic



Decoder Only Models



Deepak Subramani, deepakns@iisc.ac.in

Transformers

- Transformer Encoder
  - Converts a sequence of words to a vector representation
  - This vector representation can be used for text-understanding tasks
- Transformer Decoder
  - Uses the context of the sequence of words so far (sometimes with an additional context from encoder or retrieval) to predict next token in the sequence

Language Models

- Language Model (LM) is a probabilistic model of the natural language
- It understands the statistical relationship between words and can be used to fill missing words and complete sentences
- Language Models are trained by artificially introducing missing words and creating sentence completion tasks
- This procedure is called as "Pretraining" as the LM so learnt can be used with fine tuning for other particular tasks
- Large Language Models are usually transformer based neural models with a large number of parameters that are pretrained on a large number of tokens using masked model pretraining

What is learned through pretraining?

- IISc is located in ___________, India. [Trivia]
- He put ___ fork down on the table. [syntax]
- Naruto is the hokage of the village and _______ feels responsible for all the inhabitants. [coreference]
- I went to the ocean to see the fish, turtles, seals, and ______. [lexical semantics/topic]
- Miyazaki has created a complete, complex world with this film, and it's certainly a magical journey. The movie was ___. [sentiment]
- San went into the kitchen to make some tea. Standing next to San, Ashitaka pondered his destiny. Ashitaka left the _______. [some reasoning]

Outline for Week 02

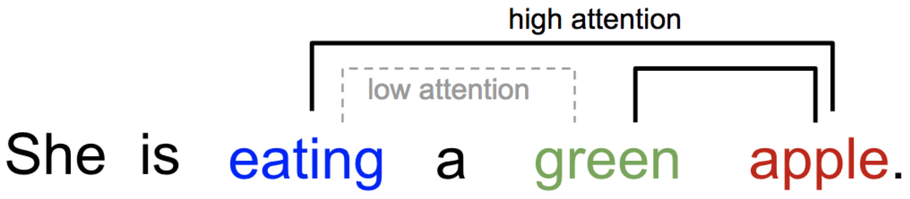
- Part 01: Sequence Modeling
  - Essential tasks in sequence modeling
  - When to use a seq model?
  - Names of recurrent architectures and introduction (details in additional slide deck on RNN – not included in the main content)
  - Additional notebooks on RNN and LSTM (not compulsory)
- Part 02: Attention Mechanism and Transformer Encoders
  - Key, Query, Value
  - Multi Head Attention
  - MHA as text representation learners for use in discriminative tasks
  - Positional Embedding
  - Trick of the trade: Layer Normalization
  - Assignment on Attention Mechanism

Transformer

- Introduced in the seminal paper Vaswani et al (2017) – Attention is all you need
- This paper introduced transformer as an Encoder-Decoder Neural Machine Translation model
- It was a sequence-to-sequence model for language translation
- The encoder part can be used for other language tasks such as text classification, sentiment analysis
- Gist of the paper: “Neural Attention” can be used for building sequential models without any recurrent or convolutional layers

Self Attention

- Key idea: Some part of the input is more important than other parts
- Model must pay “more attention” to important parts and “less attention” to other parts – A simple mechanism
- Consider the word “station”
  - Are we talking about train station, pizza station, police station or space station?
- Context determines the meaning
- So we need to adaptively change the word vector in the embedding space to represent the meaning of the word in its context
- Self attention provides this mechanism to modulate the representation of a token using the representation of tokens nearby



Self-Attention

- Consider the sentence: 'The animal didn't cross the street because it was too tired'.
- What does 'it' refer to?
- While performing self attention, the association of 'it' is made with 'animal'.

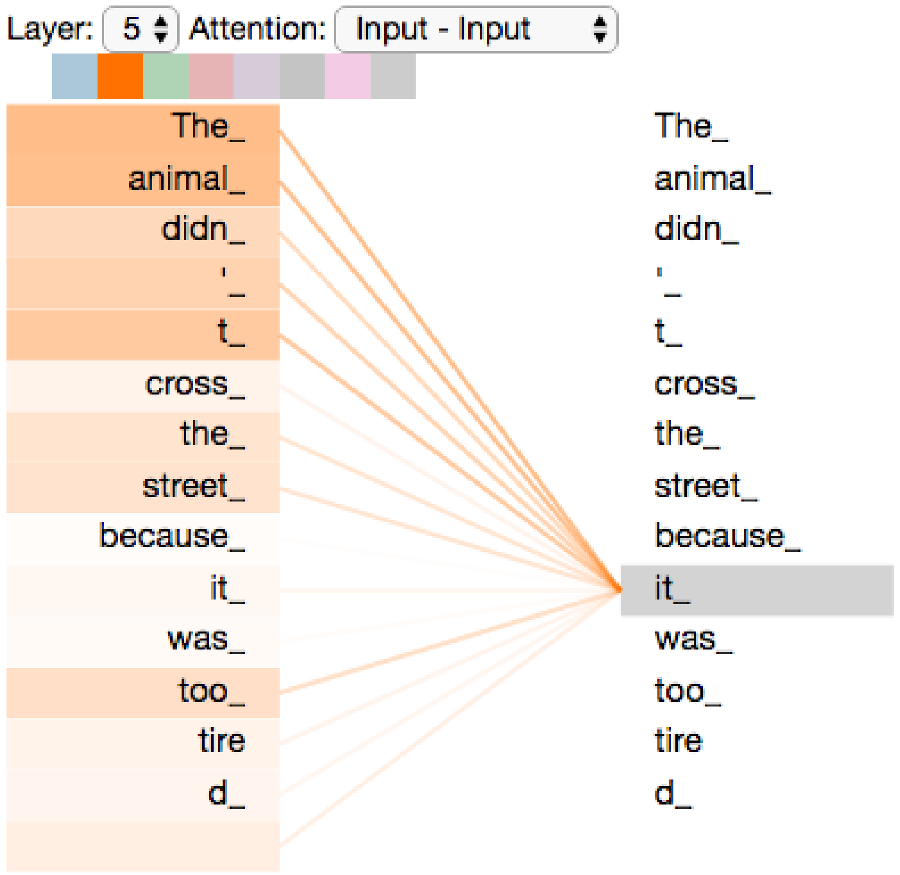


Image source: [Transformer visualization: Tensor2tensor]

Self Attention for Context Aware Vector

- Step 1: Compute the relevancy score of “station” with every other word in the sequence
- This score (just a dot product, cosine distance) scaled and softmaxed is our attention score
- Step 2: Compute sum of all word vectors weighted with attention scores to replace the representation of station
- Here the word station to itself is also used, so everything works well!
- Repeat the process for every word in the sequence and get a new sequence of vectors encoding the sentence
- output=sum(input * pairwise_score(input, input))

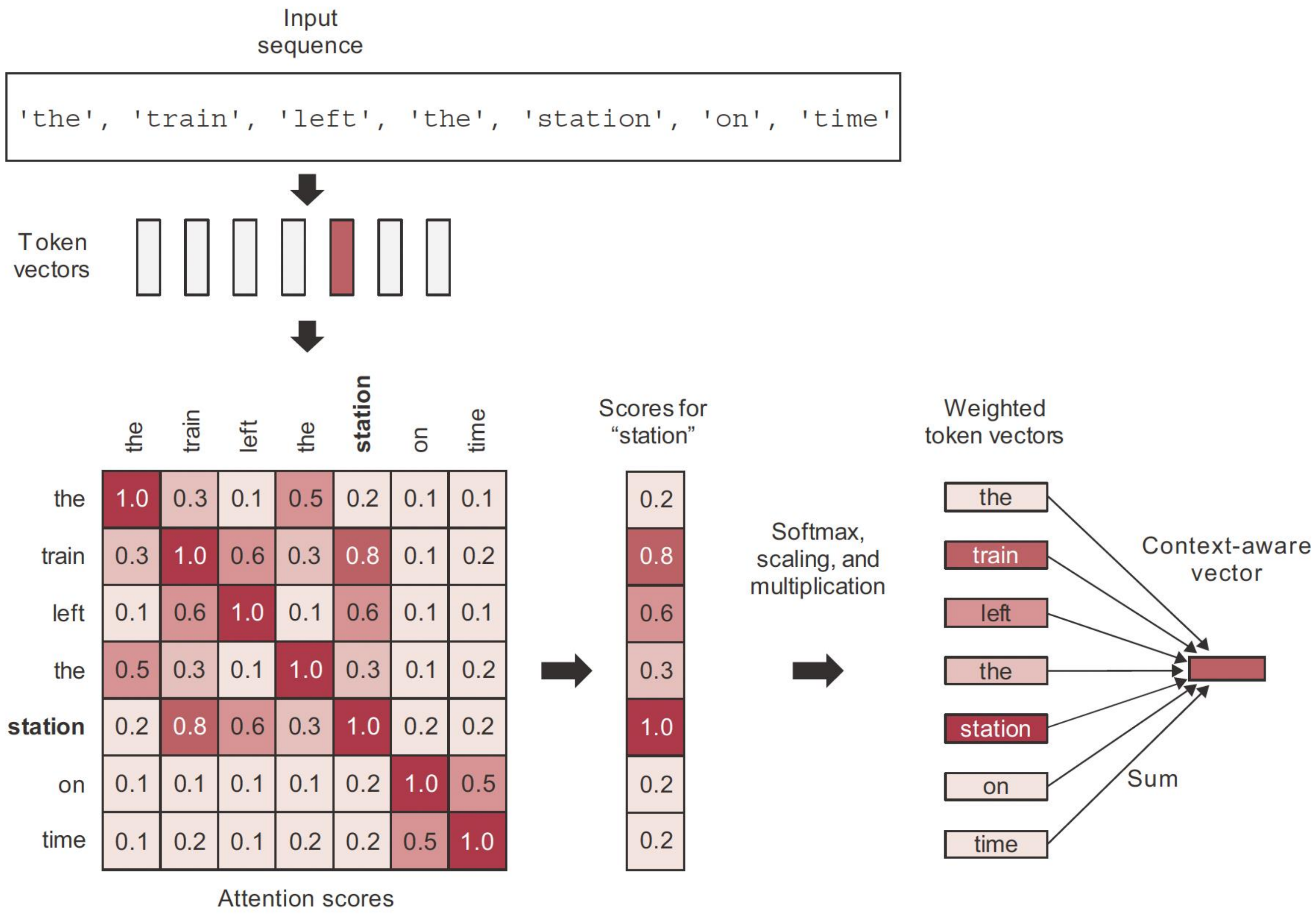


Figure 11.6 Self-attention: attention scores are computed between “station” and every other word in the sequence, and they are then used to weight a sum of word vectors that becomes the new “station” vector.

Simple Calculator

Generalized Self Attention – Query-Key-Value Model

- Consider a search engine example
- Write the Self Attention of the stattion example as a schematic
- output=sum(input * pairwise_score(input, input))
- Generalize it to a QKV Model
- output=sum(values * pairwise_score(query, key))

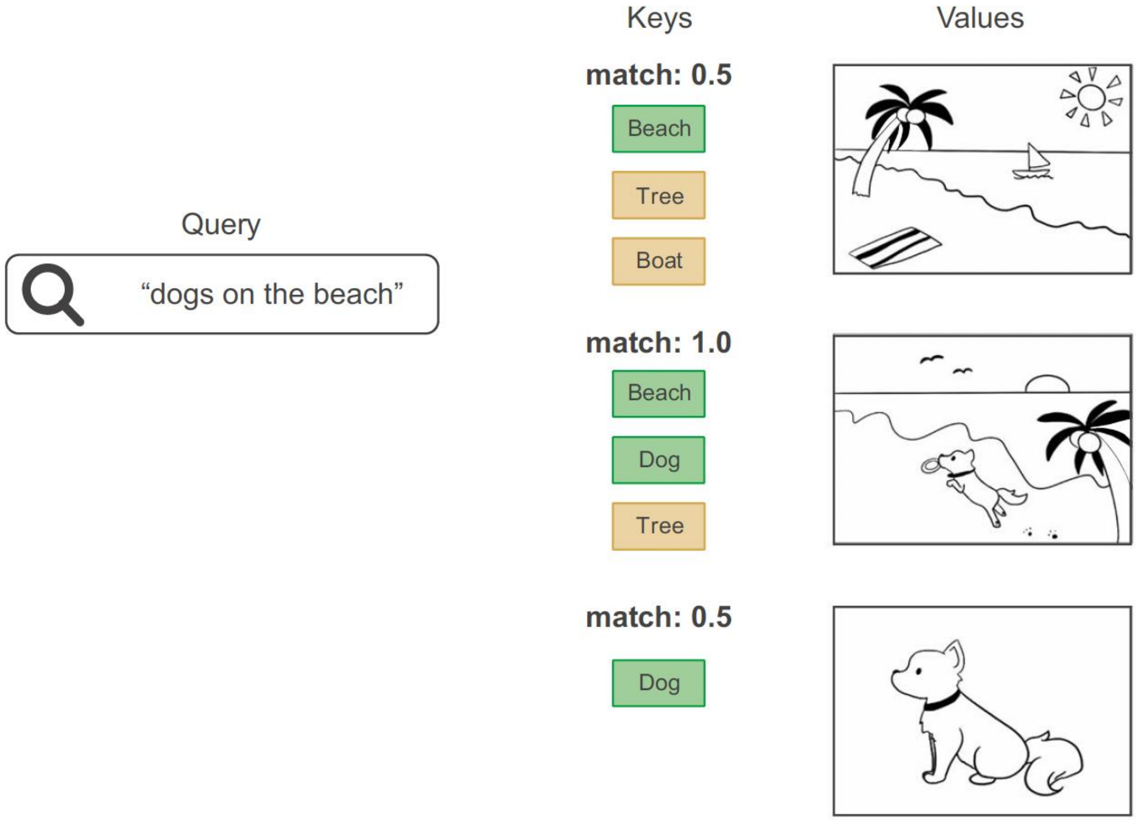
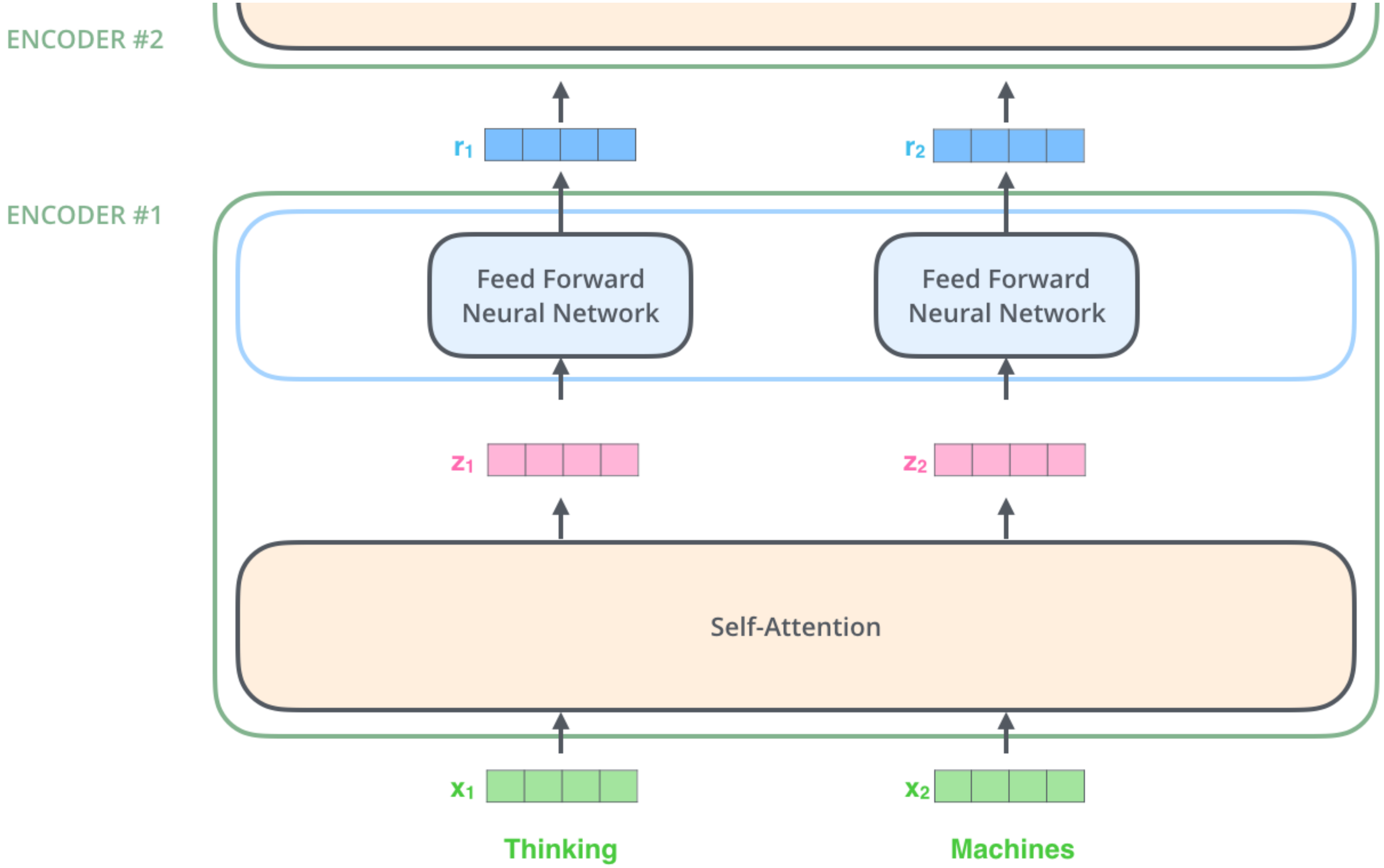
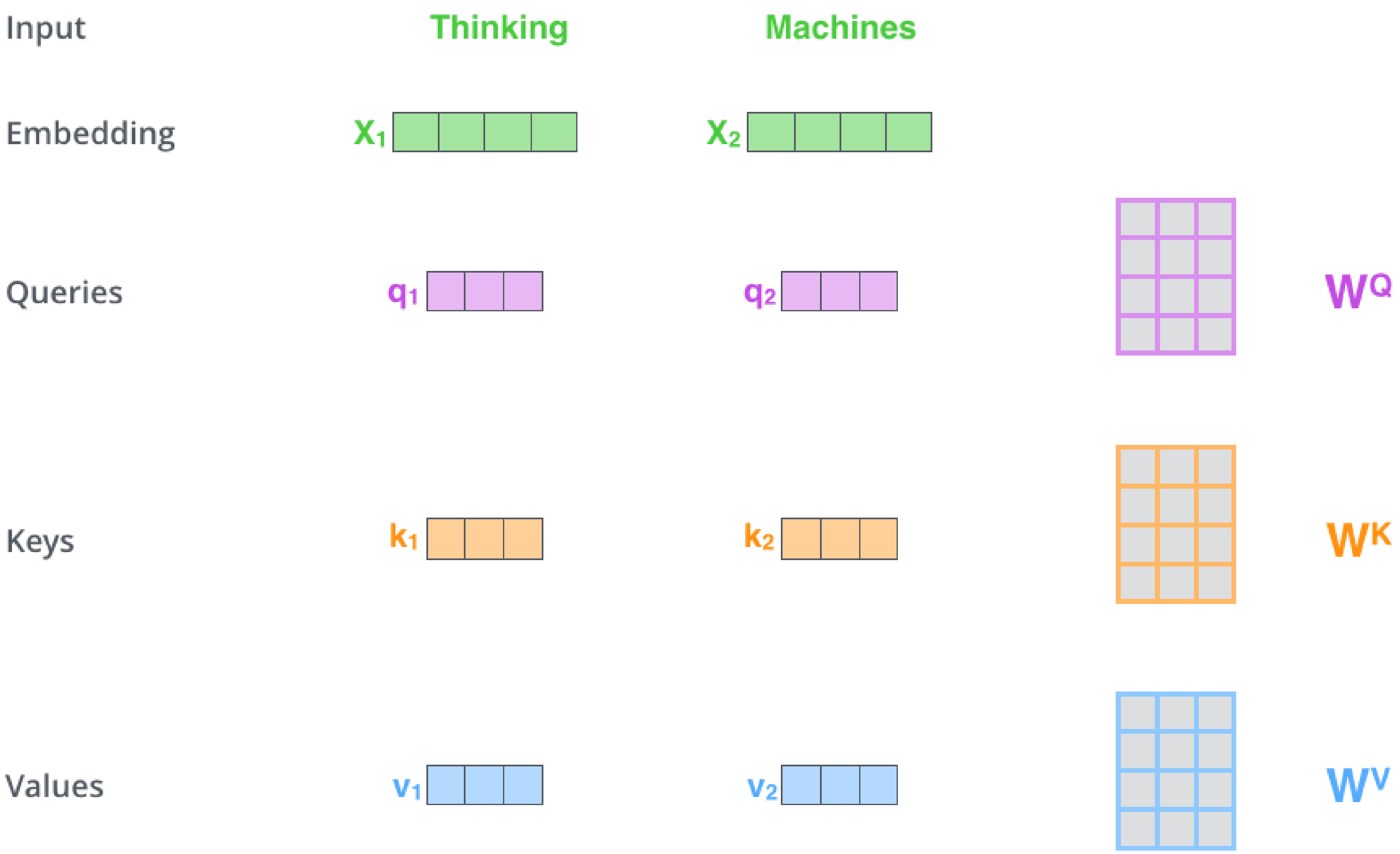


Figure 11.7 Retrieving images from a database: the "query" is compared to a set of "keys," and the match scores are used to rank "values" (images).

Attention through figures

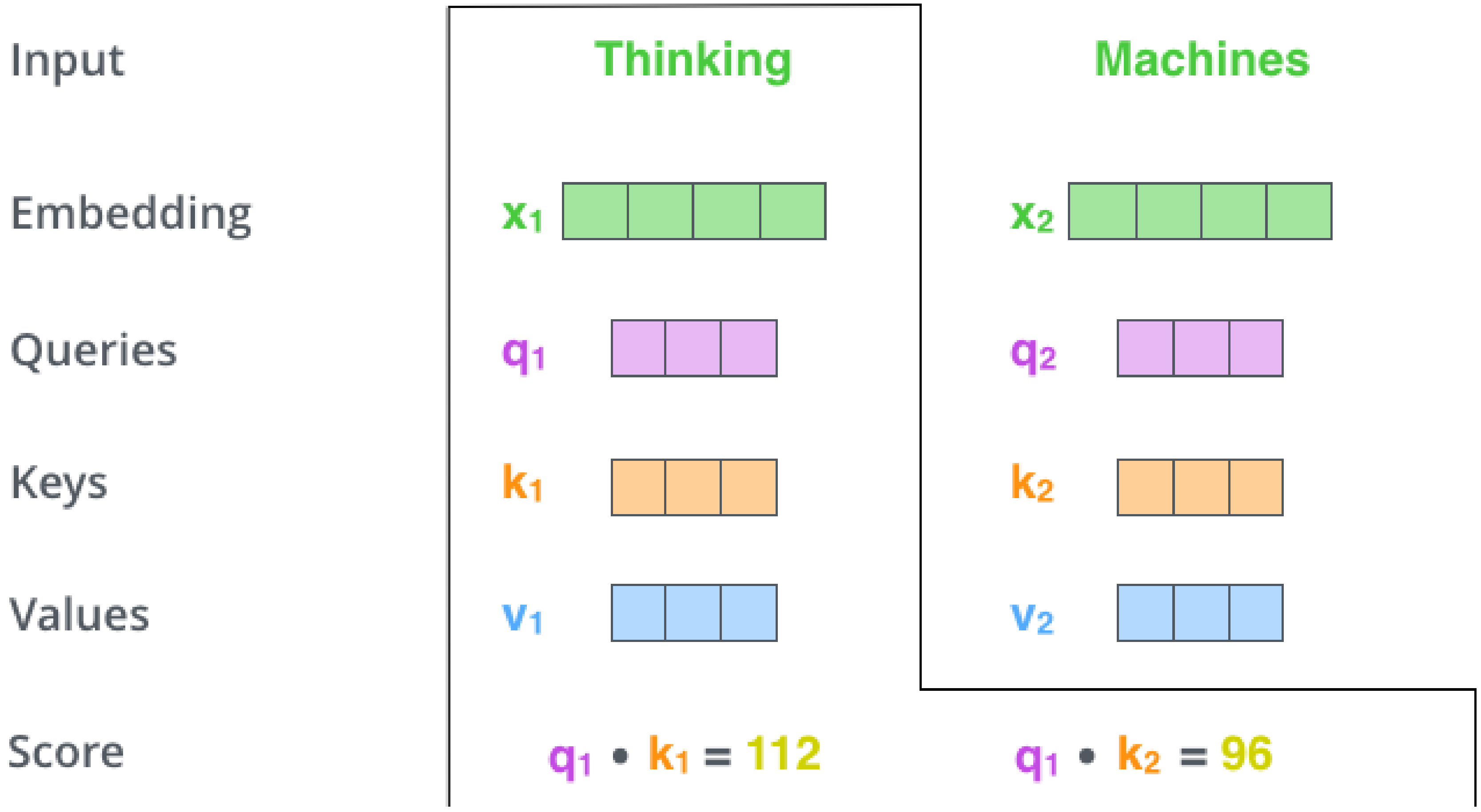


1st Step: Key, Query, Value



Deepak Subramani, deepakns@iisc.ac.in

2^{nd} Step: q.k pairwise score



3,4: Divide and Normalize

Input	Thinking	Machines
Embedding	x_1	x_2
Queries	q_1	q_2
Keys	k_1	k_2
Values	v_1	v_2
Score	$q_1 \bullet k_1 = 112$	$q_1 \bullet k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12

5,6: Multiply and Add

Input	Thinking	Machines	
Embedding	x_1	x_2	x_3
Queries	q_1	q_2	
Keys	k_1	k_2	
Values	v_1	v_2	
Score	$q_1 \bullet k_1 = 112$	$q_1 \bullet k_2 = 96$	
Divide by 8 ($\sqrt{d_k}$)	14	12	
Softmax	0.88	0.12	
Softmax X Value	v_1	v_2	
Sum	z_1	z_2	z_3

Query-Key-Value as Matrices

- How is self attention calculated?
- Attention operates on queries, keys and values.
- There are generated from the same source in case of self-attention.
- The weight matrices are learned during training.
- $q_i = W^Q x_i$
- $k_i = W^K x_i$
- $v_i = W^V x_i$
- $Attention(Q, K, V) = softmax\left(\frac{QK^T}{scaling}\right)V$

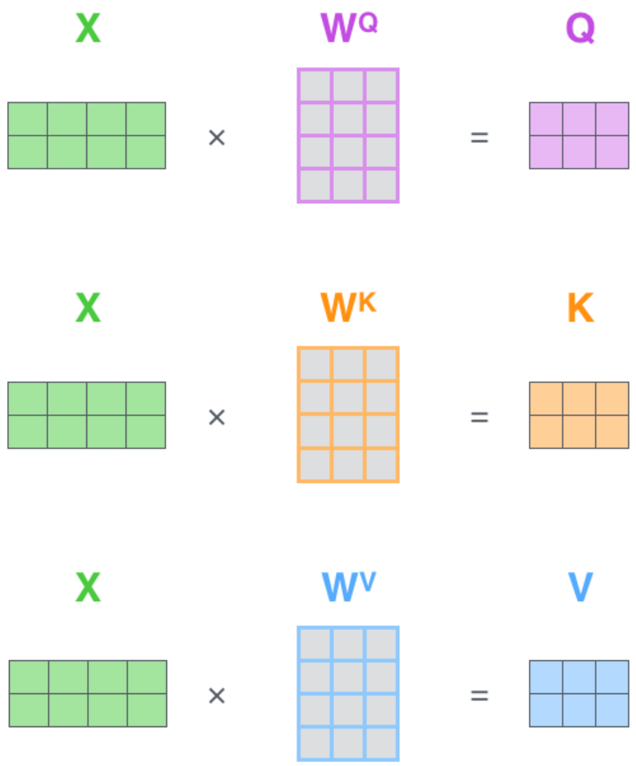
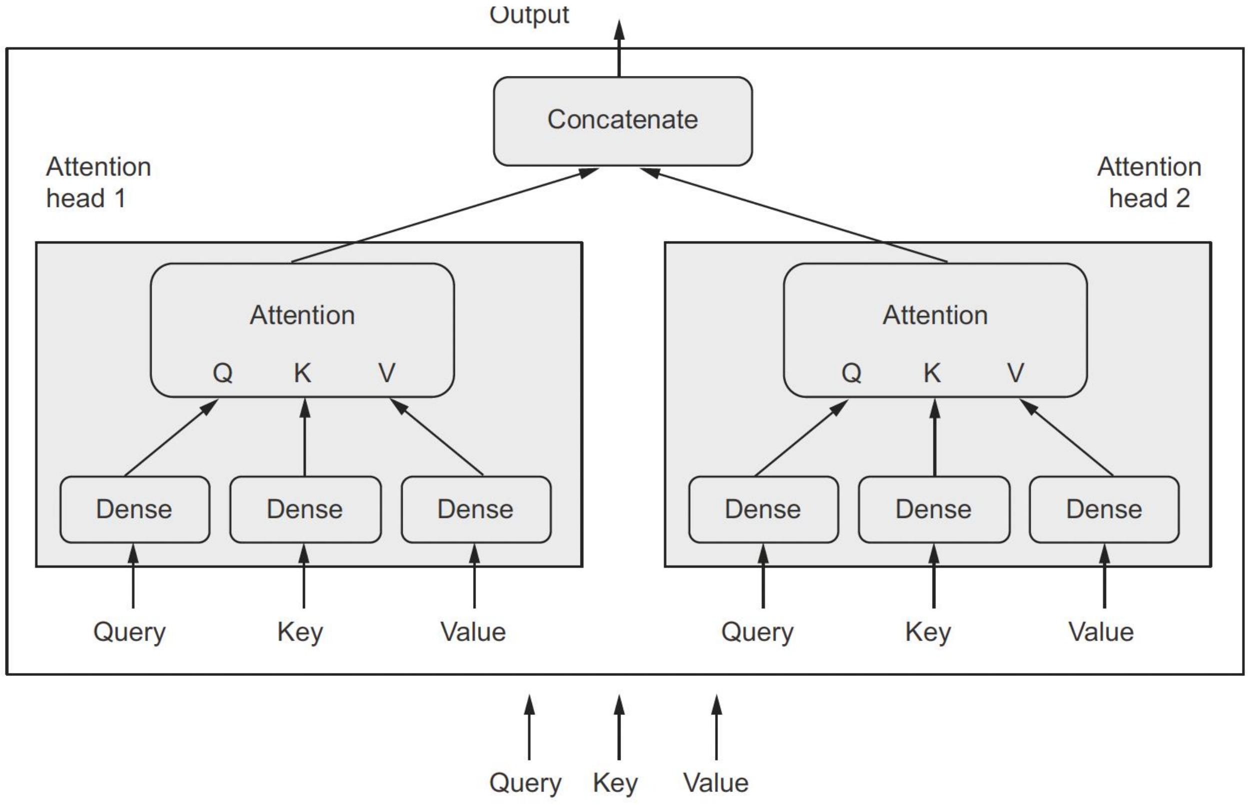


Image source: https://jalammar.github.io/illustrated-transformer/

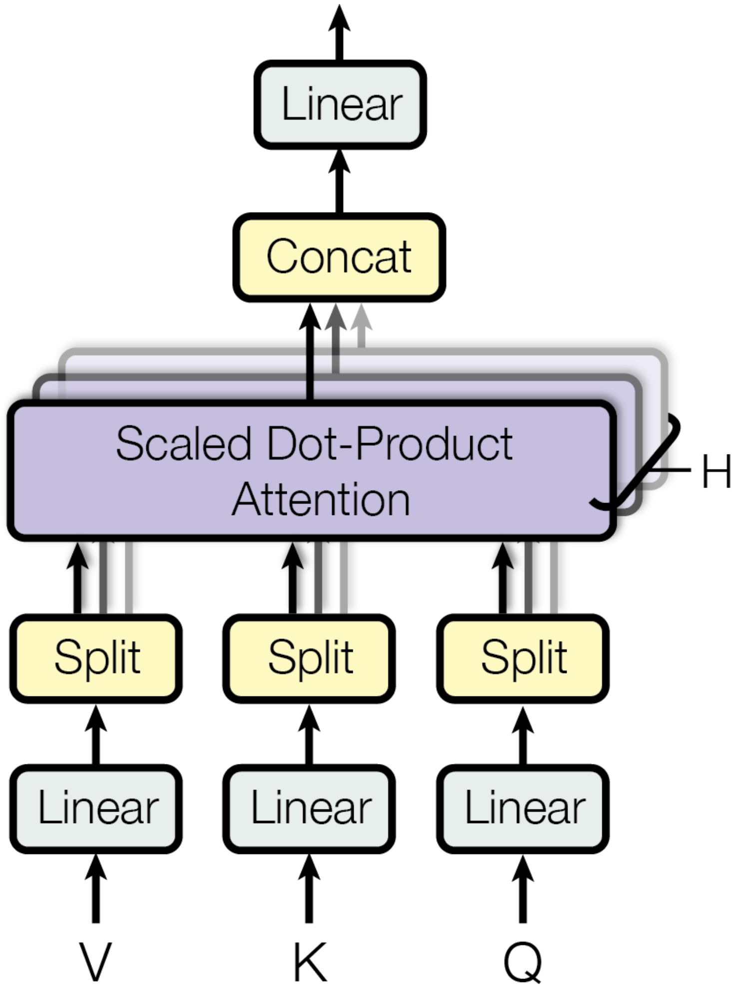
Multi Head Attention

- In an attention head: The initial query, key and value are sent through three separate dense layers before self attention is applied
- Several heads are applied in parallel to an input sequence, making it a multi-head attention layer



Benefit of Dense Layers in MHA

- This mechanism has two benefits
  - Dense layers allows learning within the attention layer
  - The output space is factored into three subspaces much like the depthwise separable convolution



All Calculations: Summary

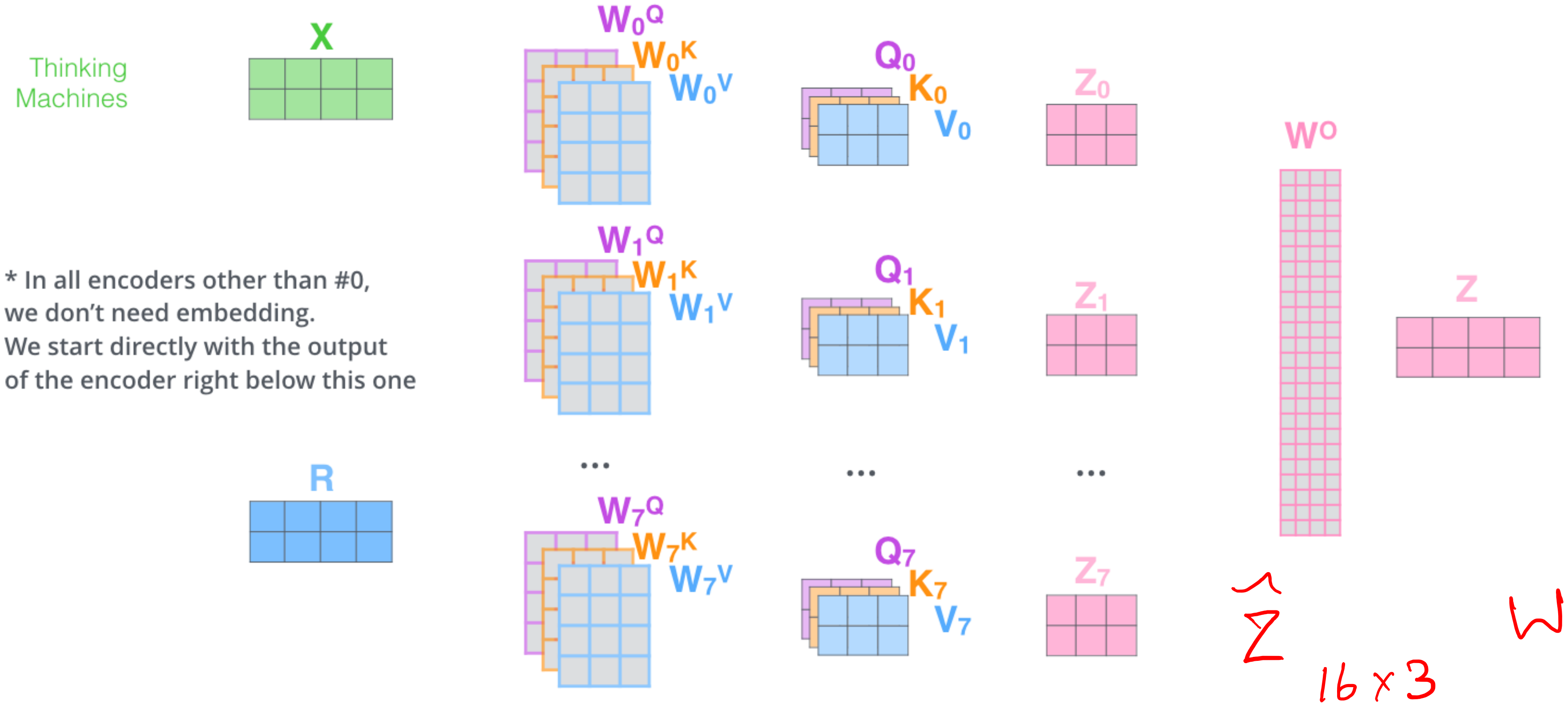
1) This is our input sentence*

2) We embed each word*

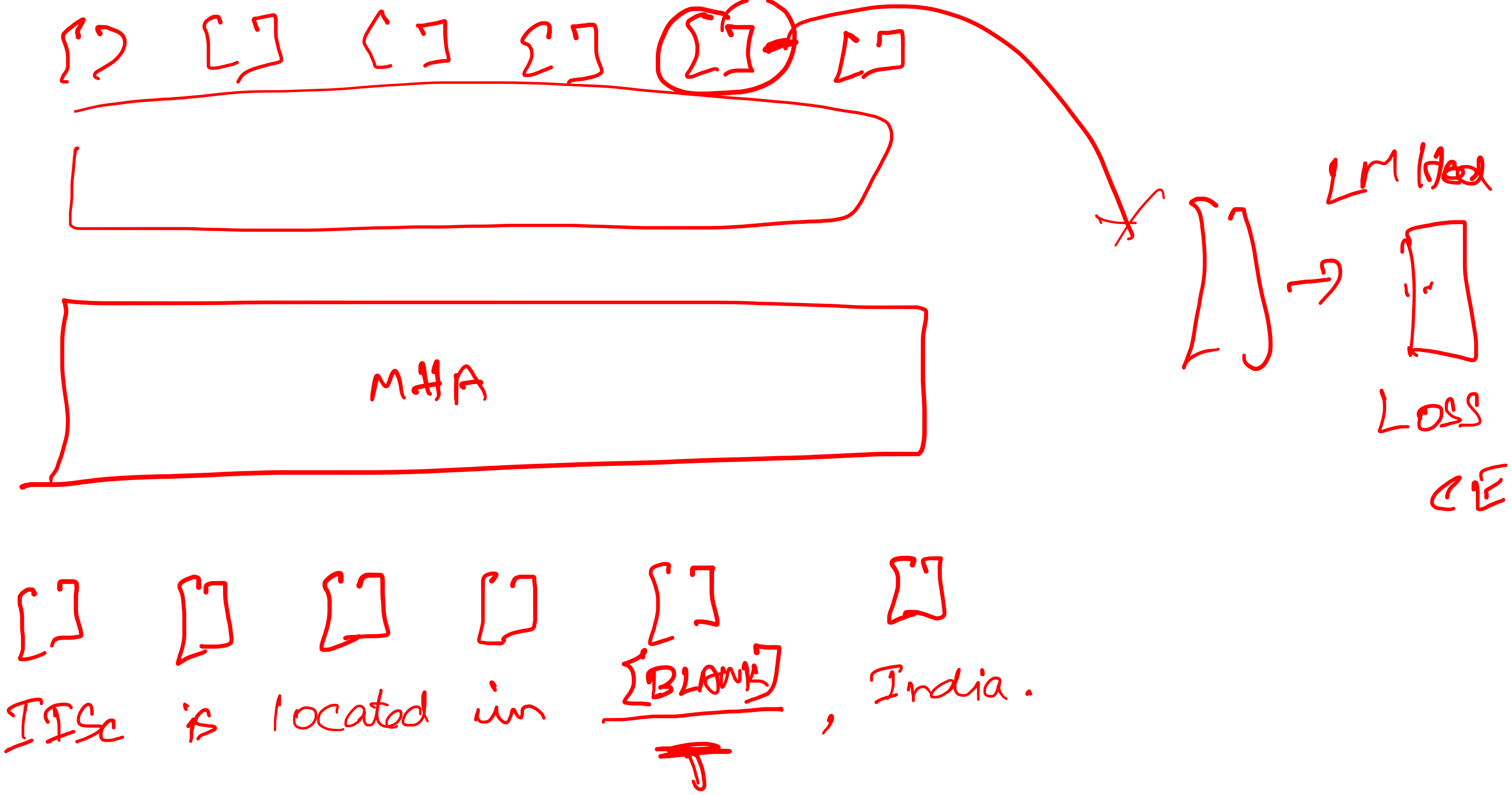
3) Split into 8 heads. We multiply X or R with weight matrices

4) Calculate attention using the resulting Q/K/V matrices

5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer



$\hat{Z}_{16\times3}$ $\quad W \quad$ $Z_{4\times2}$

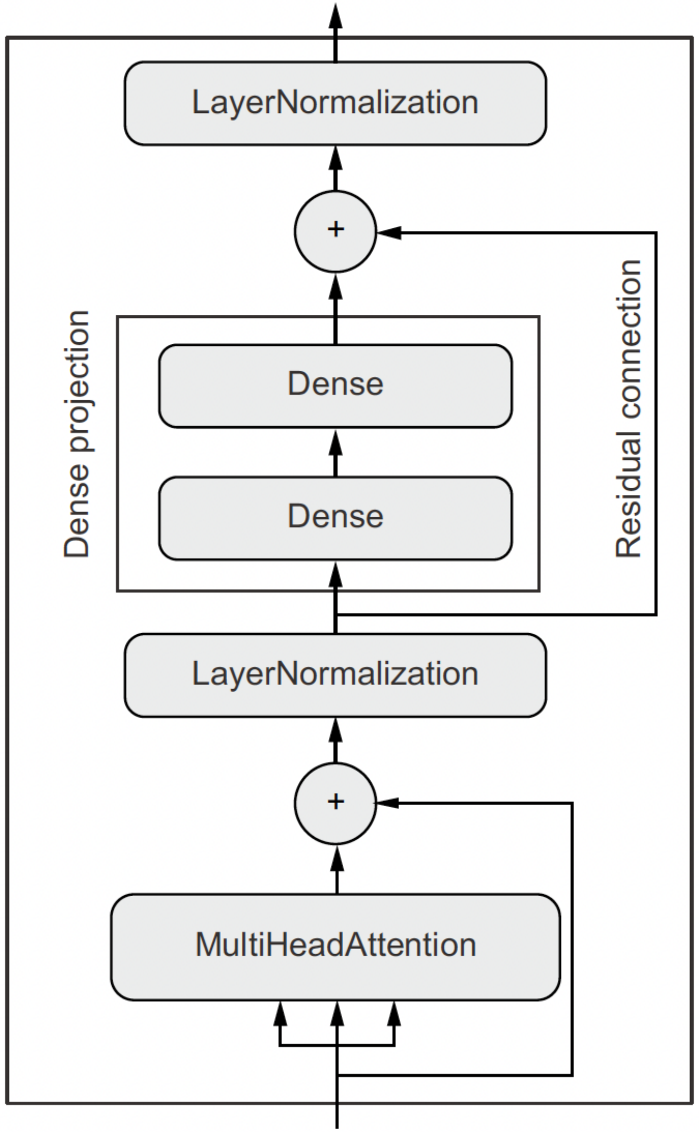

MHA
LM Head
LOSS
CE
IISc is located in [BLANK], India.

Poll

1. Which of the following is FALSE
   - Self Attention scores modify the word embeddings to be more context aware
   - Attention scores can be calculated only for one document with itself
   - Attention mechanism tells a model to pay more importance to certain parts of the input sequence
   - For a sequence with n tokens, n^2 self attention scores are needed.

Transformer Encoder

- Adding dense layers before the attention mechanism helped, so why not after?
- Yes, this leads to what is called as the transformer encoder
- While at it, add in layer normalization and residual connections
- Adding a dense layer is also called as a Dense projection
- This architecture is called as a transformer encoder that allows learning from text corpora well
- Transformer Encoder has to be written as a subclassed Layer in Keras
- Keras NLP has an inbuilt layer https://keras.io/api/keras_nlp/layers/transformer_encoder/



Batch vs Layer Normalization

```
def layer_normalization(batch_of_sequences):
    mean = np.mean(batch_of_sequences, keepdims=True, axis=-1)
    variance = np.var(batch_of_sequences, keepdims=True, axis=-1)
    return (batch_of_sequences - mean) / variance
```

Input shape: (batch_size, sequence_length, embedding_dim)

To compute mean and variance, we only pool data over the last axis (axis -1).

Compare to BatchNormalization (during training):

```
def batch_normalization(batch_of_images):
    mean = np.mean(batch_of_images, keepdims=True, axis=(0, 1, 2))
    variance = np.var(batch_of_images, keepdims=True, axis=(0, 1, 2))
    return (batch_of_images - mean) / variance
```

Input shape: (batch_size, height, width, channels)

Pool data over the batch axis (axis 0), which creates interactions between samples in a batch.

The Missing Ingredient

- Wait a minute for one minute – Aren't we learning deep sequential models? Where is the multi head attention layer using sequence information?
- Well, it does not
- Models with Multi Head Attention (as seen so far) will process text data as a set of words
- Now we add in the missing ingredient – Positional Encoding to inject order information to the transformer architecture

	Word order awareness	Context awareness (cross-words interactions)
Bag-of-unigrams	No	No
Bag-of-bigrams	Very limited	No
RNN	Yes	No
Self-attention	No	Yes
Transformer	Yes	Yes

Positional Encoding

- Add the word's position in the sentence to each of the word embeddings
- Word Embeddings = Usual word vector (without seq info) + position of the word in the sequence
- Simple approach is to give an integer, but neural networks like to have normalized numbers, so a transformation must take place
- The original paper proposed a sinusoidal position embedding
- A simpler and more powerful approach is to make the position embedding similar to word embedding and make it learnable
- After that let the Multi Head Attention act on the combined word and position embedding

Positional Encoding

- RNNs processed words at each time step.
- How to we add the notion of position in case of transformers?
- Since self-attention doesn't build in order information, we need to encode the order of the sentence in our keys, queries, and values.

$$\boldsymbol{p}_i \in \mathbb{R}^d, \text{ for } i \in \{1,2,\ldots,n\} \text{ are position vectors}$$

- x_i is the embedding of the word i. The position information is added with the embedding.

$$\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i + \boldsymbol{p}_i$$

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

Positional Embedding

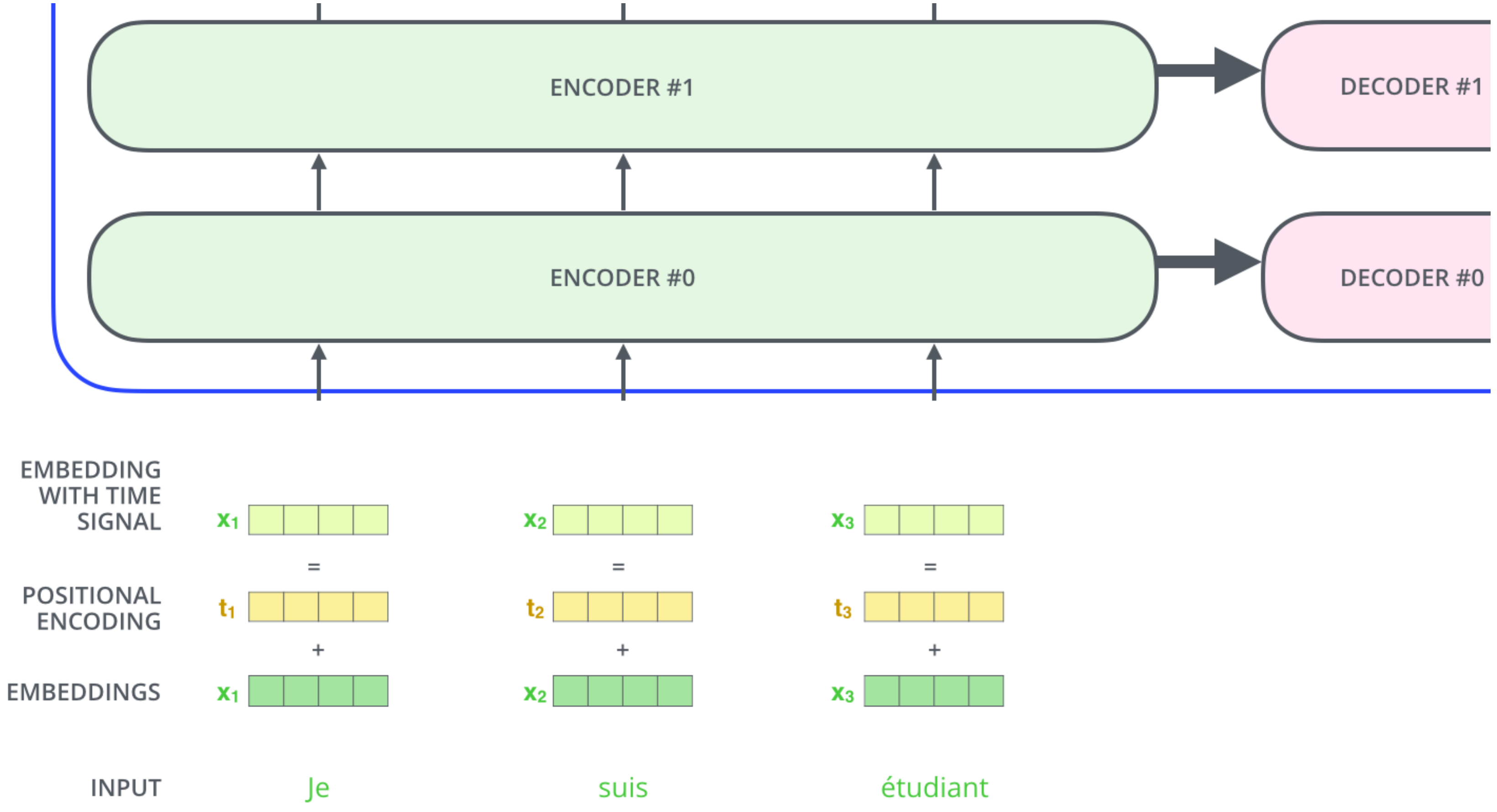
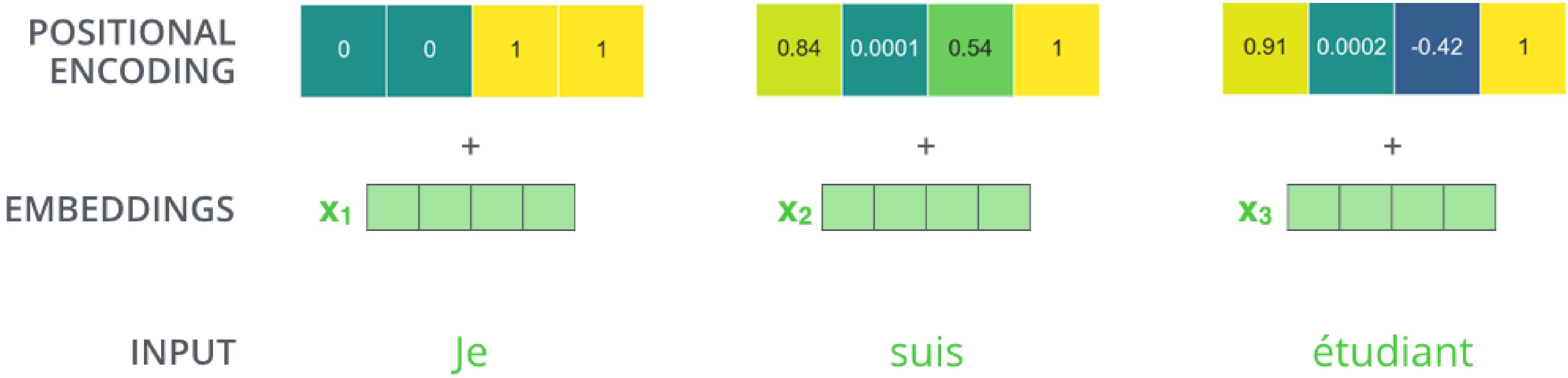


Image source: https://jalammar.github.io/illustrated-transformer/



Deepak Subramani, deepakns@iisc.ac.in

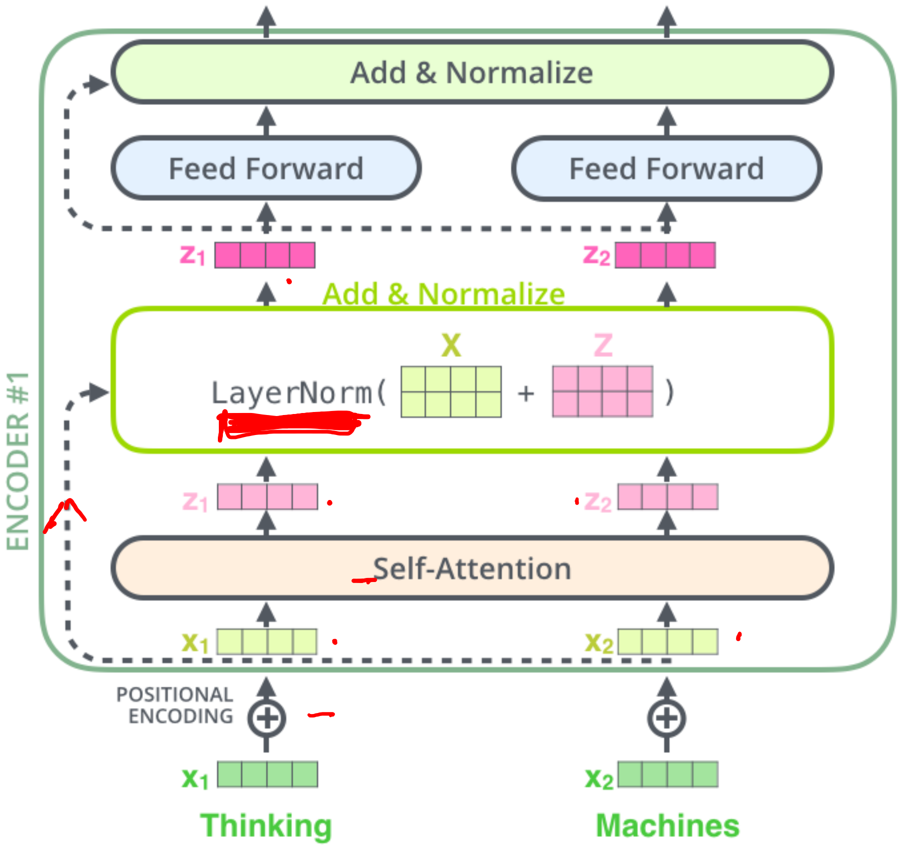
Visualization of Residual Connection and Layer Normalization – Full Transformer Encoder

Department of Computational and Data Sciences

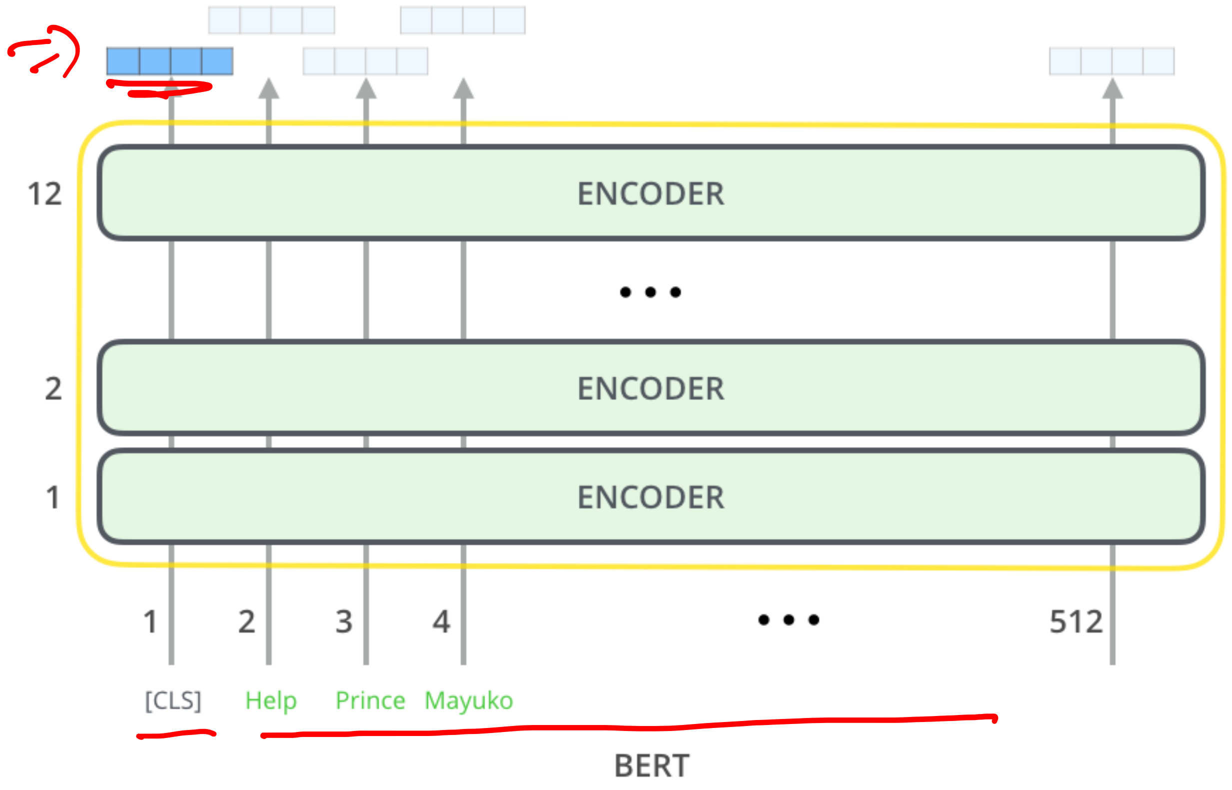
INDIAN INSTITUTE OF SCIENCE
भारतीय विज्ञान संस्थान

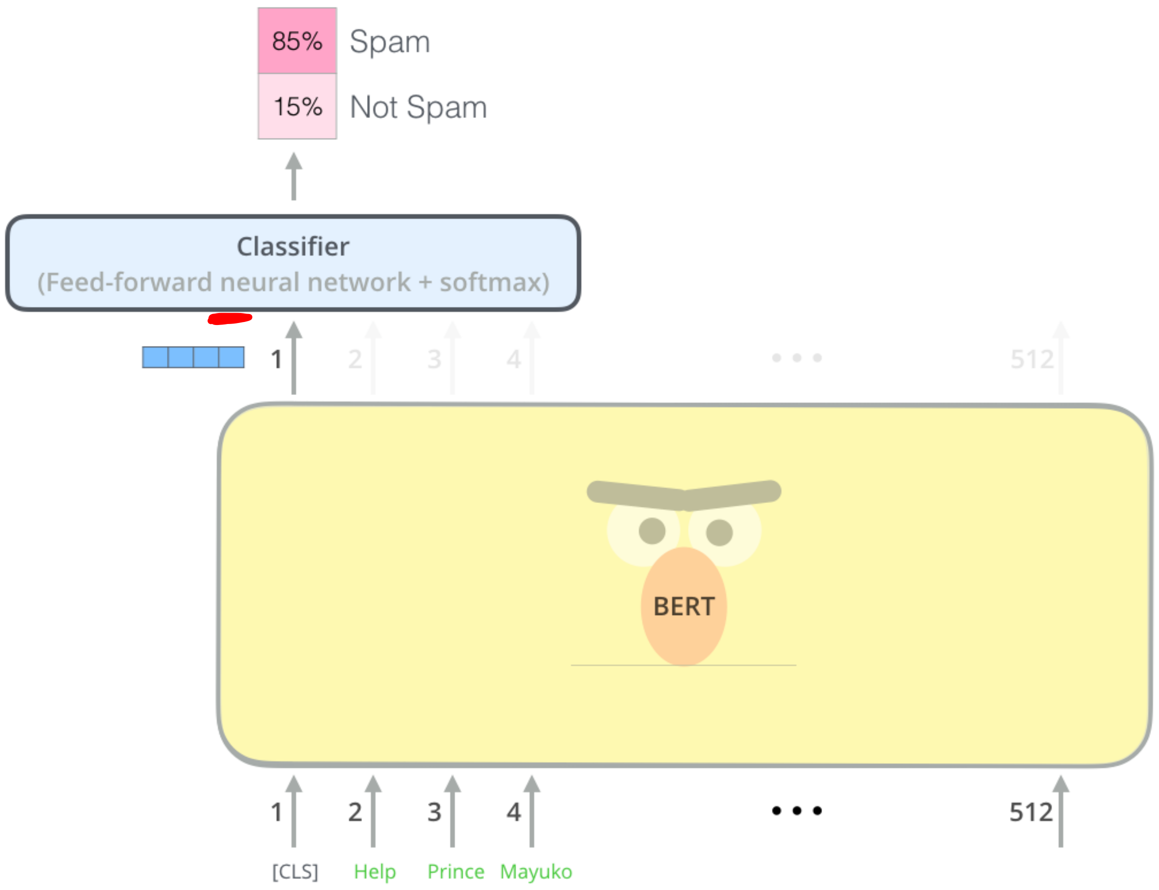
$$Z_2 = \sum_{i=1}^{3} (q_2 \cdot k_i) v_i$$

$$Z_2 = (q_2 \cdot k_1) v_1 + (q_2 \cdot k_2) v_2 + (q_2 \cdot k_3) v_3$$



Bidirectional Encoder Representation with Transformers (BERT) for Text Classification





Poll

1. Layer Normalization is not used in Transformer Encoder
   - True, False
2. In Batch Normalization, normalization is done across the features
   - True, False
3. Positional Encoding is learnt as an embedding layer
   - True, False