CS772: Deep Learning for Natural Language Processing

Attention and Transformer

Pushpak Bhattacharyya

Computer Science and Engineering Department

IIT Bombay

Week 13 of 28th March, 2022

Attention: Linguistic and Cognitive View

Task *vs.* Technique Matrix

Task (row) vs. Technique (col) Matrix	**Rules Based/Knowledge-Based**	**Classical ML**				**Deep Learning**		
		Perceptron	Logistic Regression	SVM	Graphical Models (HMM, MEMM, CRF)	Dense FF with BP and softmax	RNN-LSTM	CNN
Morphology, POS, Chunking- Shallow Parsing								
Parsing								
NER, MWE								
Coreference, cohesion, coherence- discourse								
WSD								
Language Modelling								
Text Entailment								
Sentiment and Emotion								
Semantic Role Labeling								
Machine Translation								
Question Answering								

Is Logistic Regression-NER a good technique-task combination?

- Yes: if NEI (named entity identification): name or not-name: *puja_name ne puja_not-name ke liye phul kharidaa*

- No: if NER (disambiguation amongst name categories): *Washington_location voted Washington_person to power*

Build up to attention

EnCoder-DeCoder (1/2)

- Two RNNs/LSTMs. One we call the encoder – this reads the input sentence and tries to make sense of it, before summarizing it. It passes the summary (context vector) to the decoder which translates the input sentence by just seeing it.

Moot question

- Does the context vector REALLY represent sentence meaning?
- Look deeper: each word vector represents its ALL possible context
- “The” is very non-descript!- occurs in almost all possible contexts
- Encoder builds the context vector from word vectors which represent ALL context besides the current context
- Question: can this methodology really capture sentence meaning?

Encoder-Decoder (2/2)

- Cho et al (2014), who proposed the encoder-decoder network, demonstrated that **the performance of the encoder-decoder network degrades rapidly as the length of the input sentence increases.**
- A paper with negative observation, bringing sanity to the euphoria!
- Recall NP-completeness

Attention introduced

- Bahdanau et al (2015) came up with a simple but elegant idea where they suggested that not only can all the input words be taken into account in the context vector, but relative importance should also be given to each one of them.

Seq2Seq- Encoder

- An **encoder** processes the input sequence and compresses the information into a context vector (also known as sentence embedding or “thought” vector) of a *fixed length*. This representation is expected to be a good summary of the meaning of the *whole* source sequence.

S2S- Decoder

- A **decoder** is initialized with the context vector to emit the transformed output. The early work only used the last state of the encoder network as the decoder initial state.
- Both the encoder and decoder are recurrent neural networks, i.e. using LSTM or GRU units.

A point about decoder

- Without context vector’s conditioning, decoder is essentially an LM
- *P(next word| previous seq of words)=LM*

- But, *P(next word| previous seq of words, context vector)=Decoder*

Long distance dependency: WSD

The *bank*

Long distance dependency: WSD

The *bank* that Ram

Long distance dependency: WSD

The ***bank*** that Ram used to visit

Long distance dependency: WSD

The ***bank*** that Ram used to visit 30 years before

Long distance dependency: WSD

The ***bank*** that Ram used to visit 30 years before was closed

Long distance dependency: WSD

The *bank* that Ram used to visit 30 years before was closed due to

Long distance dependency: WSD

The *bank* that Ram used to visit 30 years before was closed due to the lockdown

Long distance dependency: WSD

The *<u>bank</u>* that Ram used to visit 30 years before was closed due to the lockdown with the Govt

Long distance dependency: WSD

The ***bank*** that Ram used to visit 30 years before was closed due to the lockdown with the Govt. getting worried that

Long distance dependency: WSD

The *bank* that Ram used to visit 30 years before was closed due to the lockdown with the Govt. getting worried that crowding of people

Long distance dependency: WSD

The *bank* that Ram used to visit 30 years before was closed due to the lockdown with the Govt. getting worried that crowding of people during the

Long distance dependency: WSD

The ***bank*** that Ram used to visit 30 years before was closed due to the lockdown with the Govt. getting worried that crowding of people during the immersion ceremony

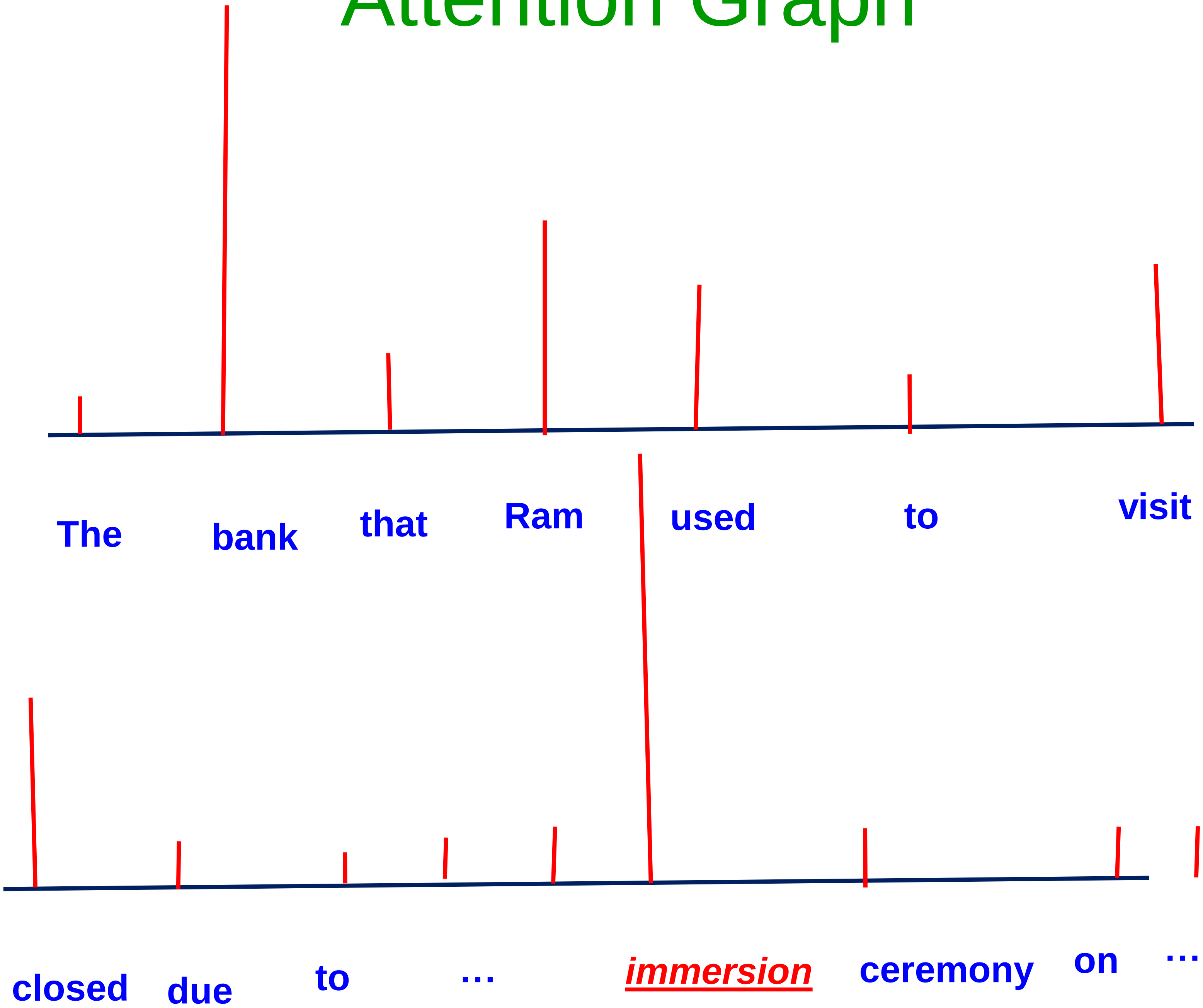
Long distance dependency: WSD

The ***bank*** that Ram used to visit 30 years before was closed due to the lockdown with the Govt. getting worried that crowding of people during the immersion ceremony on the river will aggravate the situation.

Movement of probability mass for “bank”

- Seeing “closed”, probability mass edges toward “financial” sense, because of strong association between “bank” and “closed/open”
- “lockdown” pushed this probability mass towards “river bank”
- Push further strengthened by arrival of “crowding”, “immersion” and “river” one after the other; “river” closes the case!

Attention Graph



The bank that Ram used to visit

closed due to … ***immersion*** ceremony on …

Different forms of Attention

- Morphological Attention: For predicting the token ‘jayega’, attention should be given to the token ‘Ram’ from the morphological perspective in order to render the correct form of the verb (in gender, person, number etc)
- Shallow Parsing Attention: The previous two tokens e.g might carry enough syntactic context for predicting the correct part of speech at a given position.
- Semantic Attention: From the sentence for example, ‘university’ and ‘higher studies’ are semantically related.

Backward Chaining for learning Attention

As an example, for different forms of jaana: jayega, jayegi, jayenge, jaoge etc

- We have to produce a ranking for all the different forms of ‘jaana’.
- The ranking is based on probabilities, in particular the softmax computation.
- Softmax depends on computing e^{net_input}.
- The net input is computed from the dot product of word vectors...

Attention and Eye Tracking

Eye-tracking Technology

Invasive and non-invasive eye-trackers

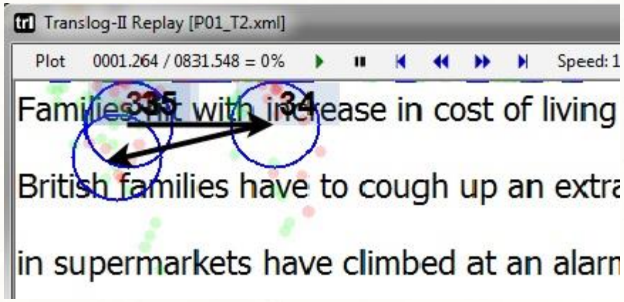
(image - sources: http://www.tobii.com/)

For linguistic studies non-invasive eye-trackers are used

- **Data delivered by eye-trackers**
  - Gaze co-ordinates of both eyes (binocular setting) or single eye (monocular setting)
  - Pupil size
- **Derivable data**
  - Fixations, Saccades, Scanpaths, Specific patterns like progression and regression.

Nature of Gaze Data

- **Gaze Point:** Position (co-ordinate) of gaze on the screen
- **Fixations :** A long stay of the gaze on a particular object on the screen
- **Saccade:** A very rapid movement of eye between the positions of rest.
  - Progressive Saccade / Forward Saccade / Progression
  - Regressive Saccade / Backward Saccade / Regression
- **Scanpath:** A path connecting a series of fixations.



Families hit with increase in cost of living

British families have to cough up an extra

in supermarkets have climbed at an alarm

Eye-movement and Cognition

- Eye-Mind Hypothesis (Just and Carpenter, 1980)

When a subject is views a word/object, he or she also processes it cognitively, for approximately the same amount of time he or she fixates on it.

- Considered useful in explaining theories associated with reading (Rayner and Duffy,1986; Irwin, 2004; von der Malsburg and Vasishth, 2011)

- Linear and uniform-speed gaze movement is observed over texts having simple concepts, and often non-linear movement with non-uniform speed over more complex concepts (Rayner, 1998)

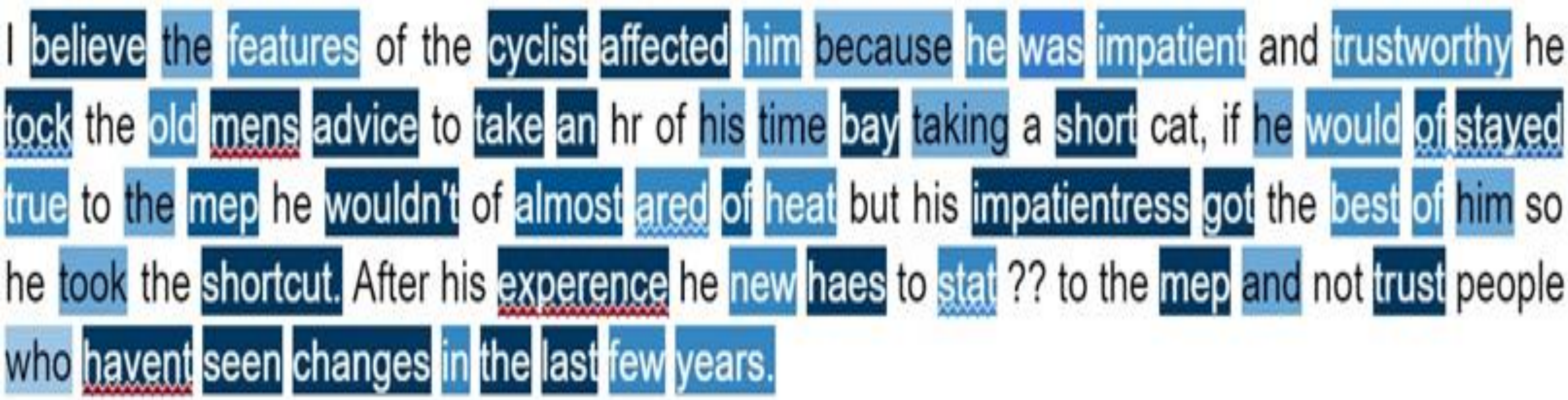
Using Gaze Behaviour to Predict Text Quality Rating

Sandeep Mathias, Diptesh Kanojia, Kevin Patel, Samarth Agrawal, Abhijit Mishra, and Pushpak Bhattacharyya. Eyes are the windows to the soul: Predicting the rating of text quality using gaze behaviour. **ACL 2018**.

Aim

- To predict the rating of different properties of text quality using text and gaze behaviour features

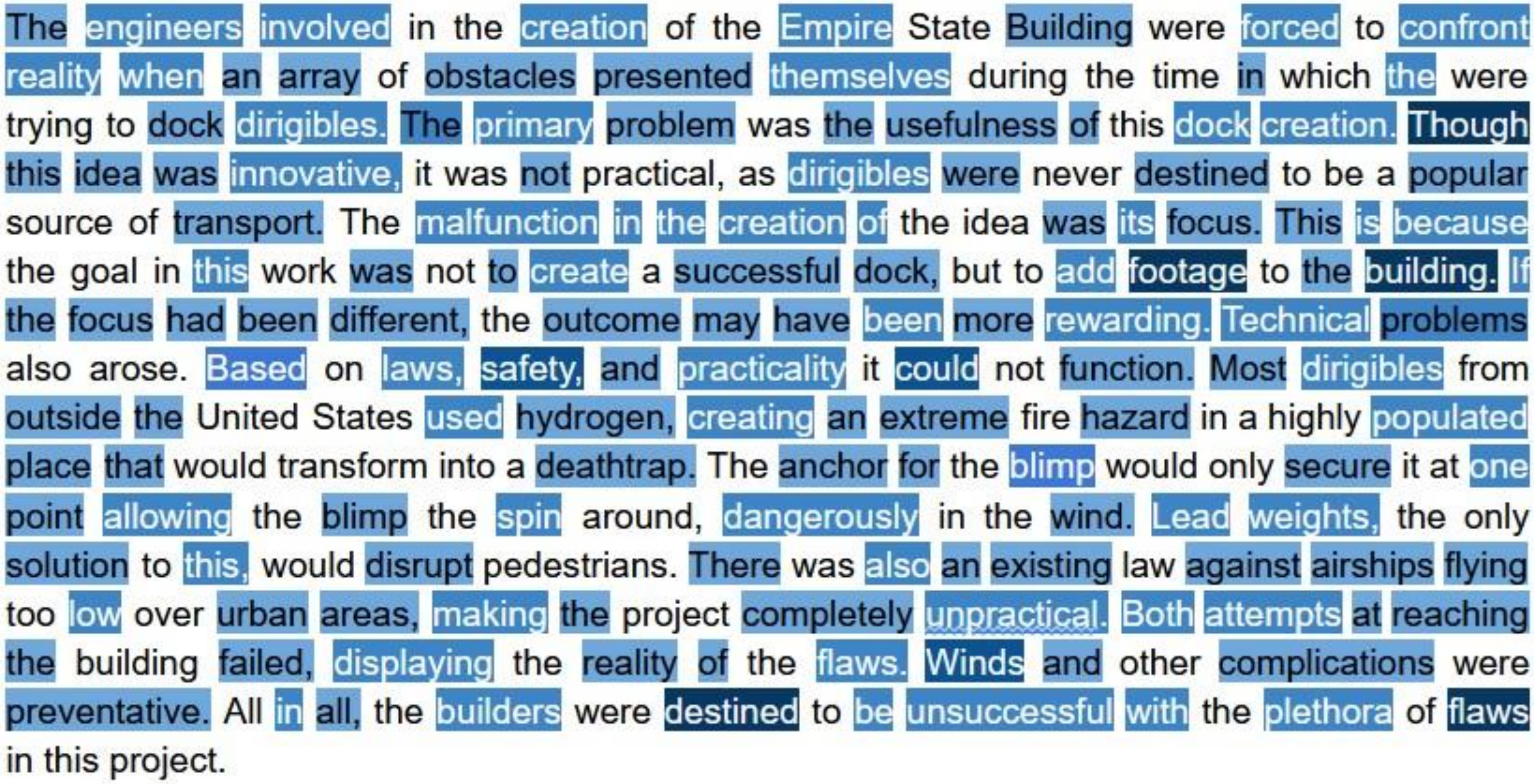
Example



I believe the features of the cyclist affected him because he was impatient and trustworthy he tock the old mens advice to take an hr of his time bay taking a short cat, if he would of stayed true to the mep he wouldn't of almost ared of heat but his impatientress got the best of him so he took the shortcut. After his experence he new haes to stat ?? to the mep and not trust people who havent seen changes in the last few years.

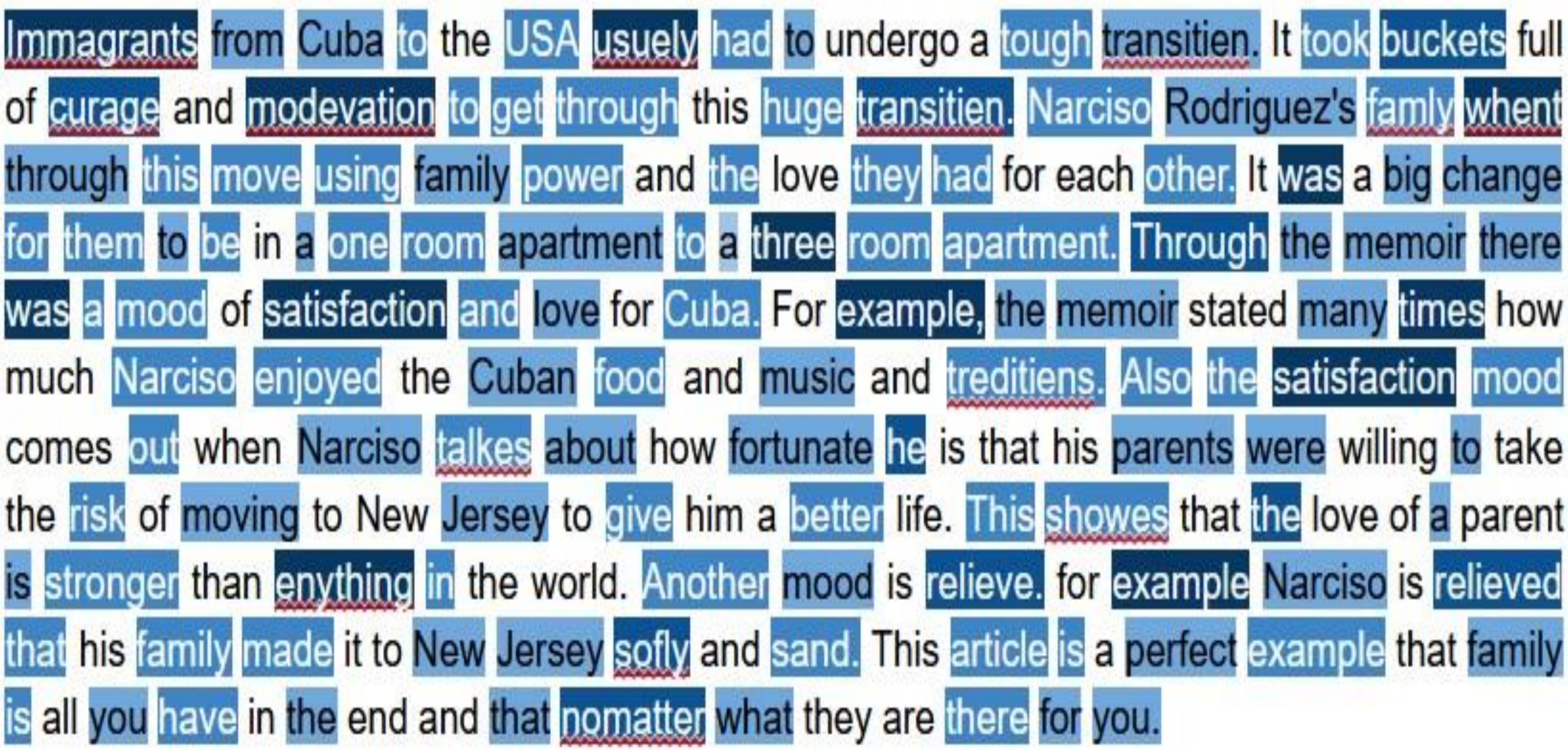
Dwell Time of a reader for one of the essays. The darker the blue, the larger the dwell time.

Dwell Time example for a good essay: notice the lack of dark blue parts



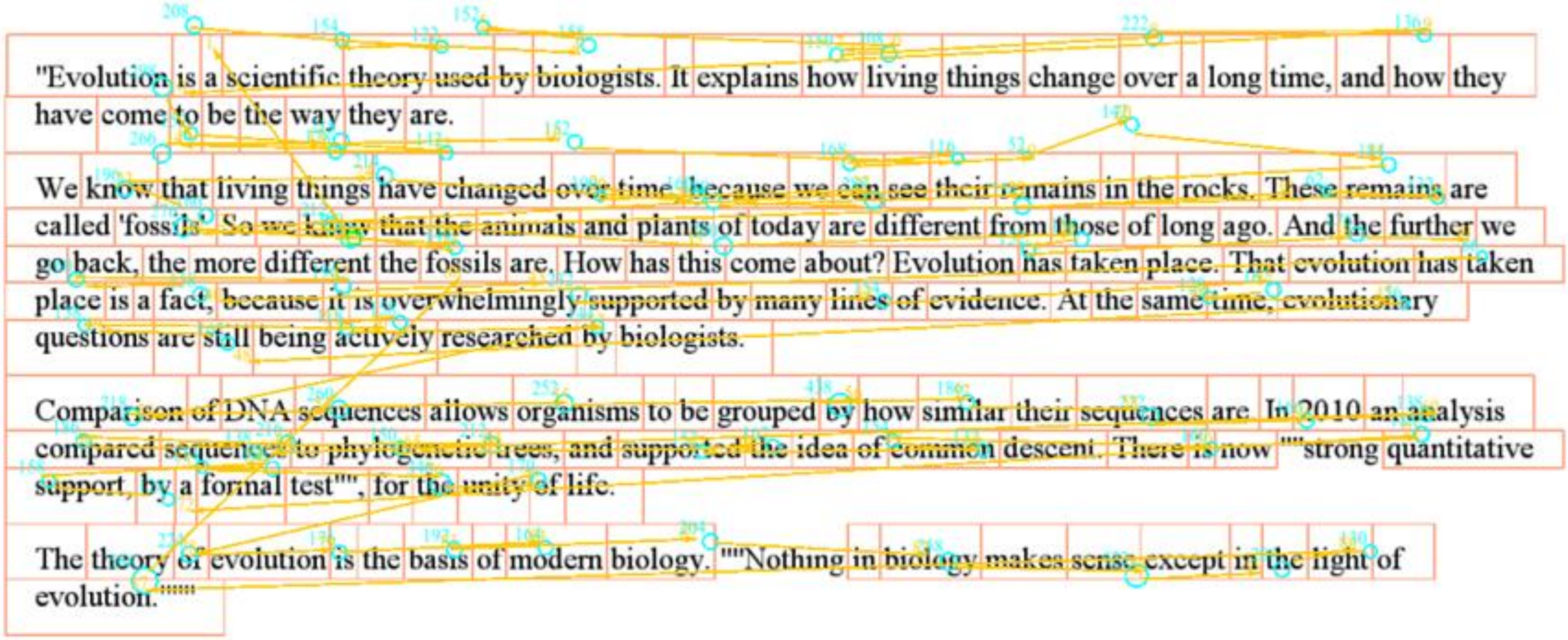
The engineers involved in the creation of the Empire State Building were forced to confront reality when an array of obstacles presented themselves during the time in which the were trying to dock dirigibles. The primary problem was the usefulness of this dock creation. Though this idea was innovative, it was not practical, as dirigibles were never destined to be a popular source of transport. The malfunction in the creation of the idea was its focus. This is because the goal in this work was not to create a successful dock, but to add footage to the building. If the focus had been different, the outcome may have been more rewarding. Technical problems also arose. Based on laws, safety, and practicality it could not function. Most dirigibles from outside the United States used hydrogen, creating an extreme fire hazard in a highly populated place that would transform into a deathtrap. The anchor for the blimp would only secure it at one point allowing the blimp the spin around, dangerously in the wind. Lead weights, the only solution to this, would disrupt pedestrians. There was also an existing law against airships flying too low over urban areas, making the project completely unpractical. Both attempts at reaching the building failed, displaying the reality of the flaws. Winds and other complications were preventative. All in all, the builders were destined to be unsuccessful with the plethora of flaws in this project.

Dwell Time example for a bad essay: Dark blue parts are more common



Immagrants from Cuba to the USA usuely had to undergo a tough transitien. It took buckets full of curage and modevation to get through this huge transitien. Narciso Rodriguez's famly whent through this move using family power and the love they had for each other. It was a big change for them to be in a one room apartment to a three room apartment. Through the memoir there was a mood of satisfaction and love for Cuba. For example, the memoir stated many times how much Narciso enjoyed the Cuban food and music and treditiens. Also the satisfaction mood comes out when Narciso talkes about how fortunate he is that his parents were willing to take the risk of moving to New Jersey to give him a better life. This showes that the love of a parent is stronger than enything in the world. Another mood is relieve. for example Narciso is relieved that his family made it to New Jersey sofly and sand. This article is a perfect example that family is all you have in the end and that nomatter what they are there for you.

There are also saccades, regress and progress



"Evolution is a scientific theory used by biologists. It explains how living things change over a long time, and how they have come to be the way they are.

We know that living things have changed over time, because we can see their remains in the rocks. These remains are called 'fossils'. So we know that the animals and plants of today are different from those of long ago. And the further we go back, the more different the fossils are. How has this come about? Evolution has taken place. That evolution has taken place is a fact, because it is overwhelmingly supported by many lines of evidence. At the same time, evolutionary questions are still being actively researched by biologists.

Comparison of DNA sequences allows organisms to be grouped by how similar their sequences are. In 2010 an analysis compared sequences to phylogenetic trees, and supported the idea of common descent. There is now ""strong quantitative support, by a formal test"", for the unity of life.

The theory of evolution is the basis of modern biology. ""Nothing in biology makes sense except in the light of evolution.""""

Text Quality Rating Properties

Organization - How well-structured the text is.
Coherence - How much sense the text makes.
Cohesion - How well-connected the text is.

Each of these properties is rated on a Likert scale on a range of 1 to 4 [12].

Text Quality Rating: An overall measure of the text quality. We compute it as the sum of the organization, coherence and cohesion scores, scaled to a range of 1 to 10.

Quality = f(Organization, Coherence, Cohesion)

Method of Collecting Gaze Behaviour Data

1. Prior to starting the experiment, the camera is calibrated, and the calibration is validated.
2. The reader reads a text and answers 2 comprehension questions about the text.
3. The reader then scores the text for organization, coherence and cohesion.
4. The quality score of the text is then calculated.

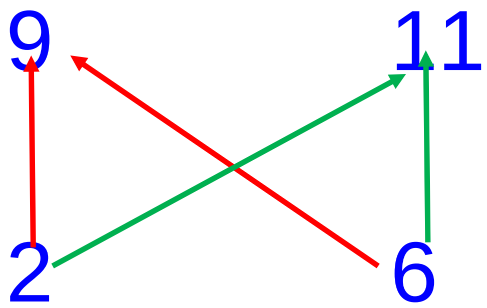
Another case for Attention: Coreference

Samarth Agrawal, Aditya Joshi, Joe Cheri Ross, Pushpak Bhattacharyya, Harshawardhan M. Wabgaonkar, *Are Word Embedding and Dialogue Act Class-based Features Useful for Coreference Resolution in Dialogue?*, PACLING 2017

Joe Cheri Ross, Abhijit Mishra and Pushpak Bhattacharyya, Leveraging Annotators' Gaze Behaviour for Coreference Resolution, ACL 2016 Workshop on Cognitive Aspects of Computational Language Learning (CogACLL 2016) at ACL 2016, Berlin, Germany, August 11, 2016

Coreference: a Foundational Problem

- *The cat went near the dog, and it bit it.*
- *The*$_1$ *cat*$_2$ *went*$_3$ *near*$_4$ *the*$_5$ *dog*$_6$ *,*$_7$ *and*$_8$ *it*$_9$ *bit*$_{10}$ *it*$_{11}$ *.*$_{12}$ (who bit whom?)
- Two possibilities:



Mentions- *it*$_9$ and *it*$_{11}$; **mention-pairs**: *<2,9>, <2,11>, <6,9>, <6,11>*

Coreference resolution

- Coreference resolution concerns
    - Finding different linguistic expressions that refers to the same entity
- Eg:
    - Binding a pronoun with corresponding noun: **Anaphora Resolution**
- The cameraman shot the batsman when he was near the minister.
    - Ambiguity:
        - “He” refers to “The cameraman”, or
        - “He” refers to ”batsman”

Pro-drop phenomenon

- *Ram promised Shyam to give a party* (who gave the party?) (subject controlled)
- *Ram forced Shyam to give a party* (who gave the party?) (object controlled)
- Elliptic Pro-Drop; no-tense marked on the clause

Complex linguistic processing

- Uncover “he” from pro-dropped sentence
- Bind to correct noun
- Then answer “who”

- Challenge (very hard!): “to give a party” is an infinitive (verb not carrying tense) and subject missing

Transformer

Presented by
Tamali Banerjee
Research scholar, IIT Bombay
tamali@cse.iitb.ac.in

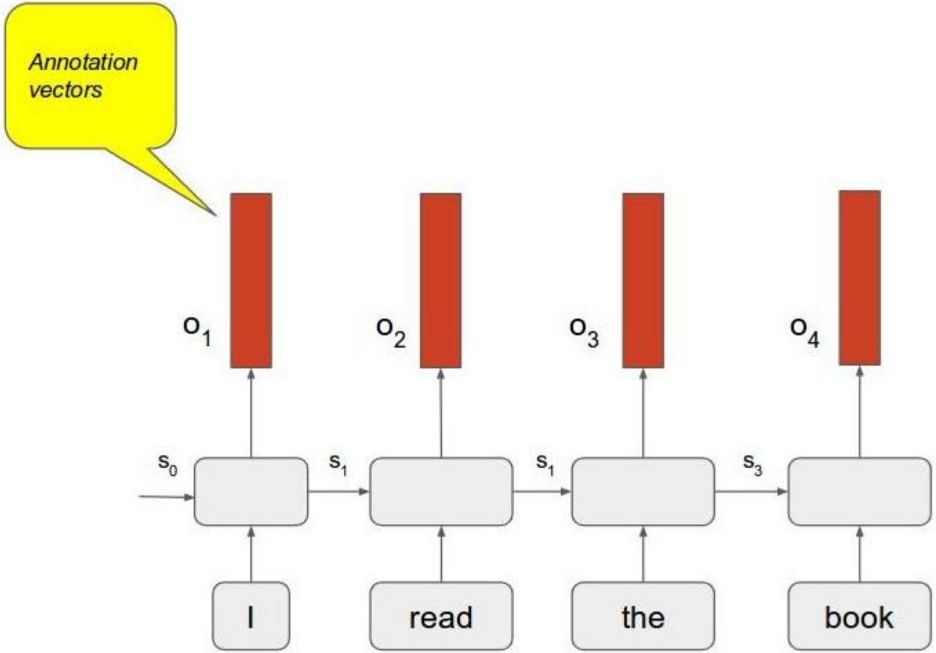
For
CS772 at IIT Bombay, March 2022
Course Instructor: Prof. Pushpak Bhattacharyya

Roadmap

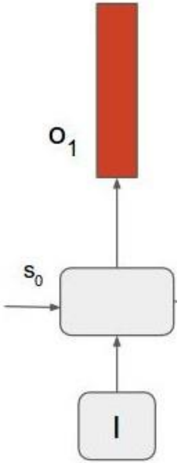
- Quick recap of attention-based encoder-decoder (using RNN)
- Benefits of Transformers-based encoder-decoder over RNN-based encoder-decoder
- Transformer architecture

Attention-based encoder-decoder architecture (recap)

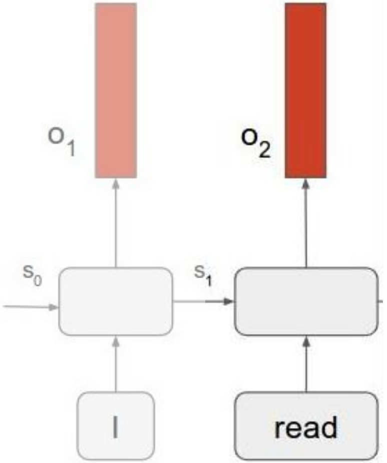
Encoder of RNN-based seq2seq architecture



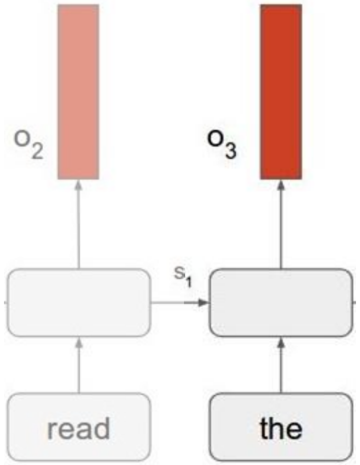
Generation of encoder output



Generation of encoder output

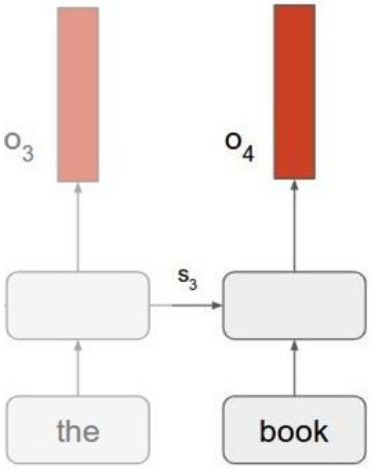


Generation of encoder output

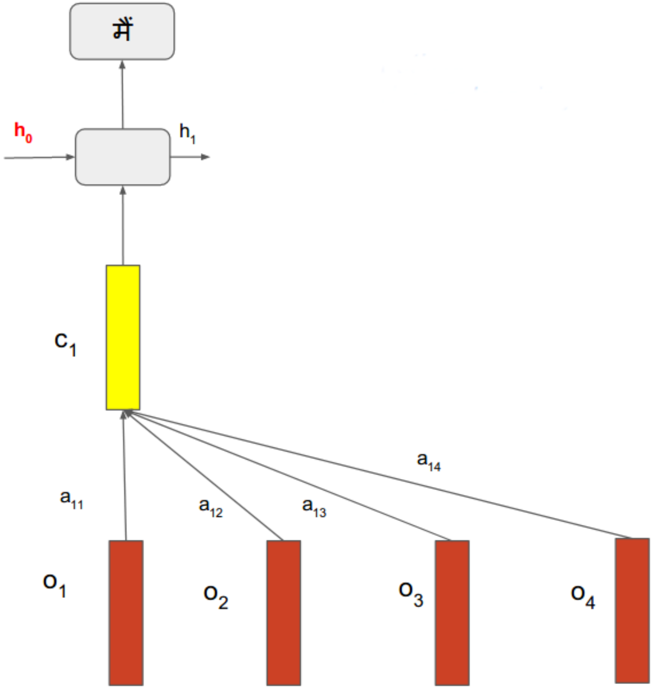


Generation of encoder output

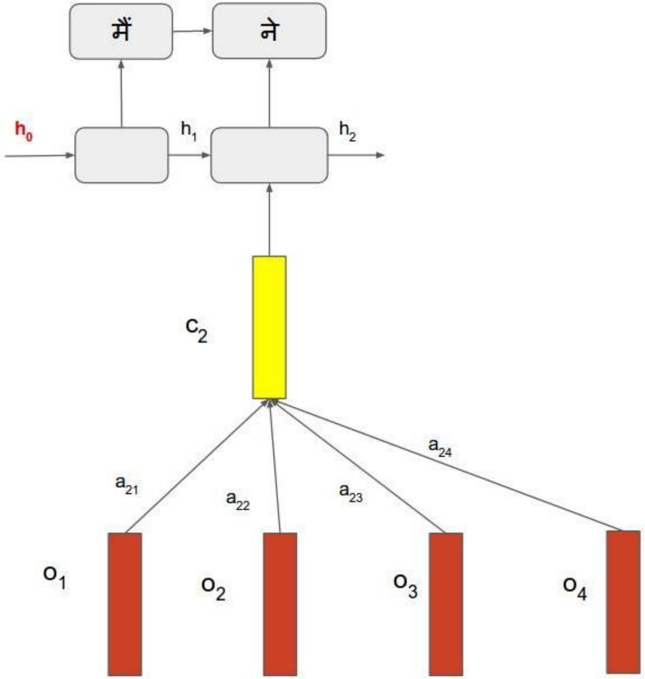
- Time consuming
- Short-term memory



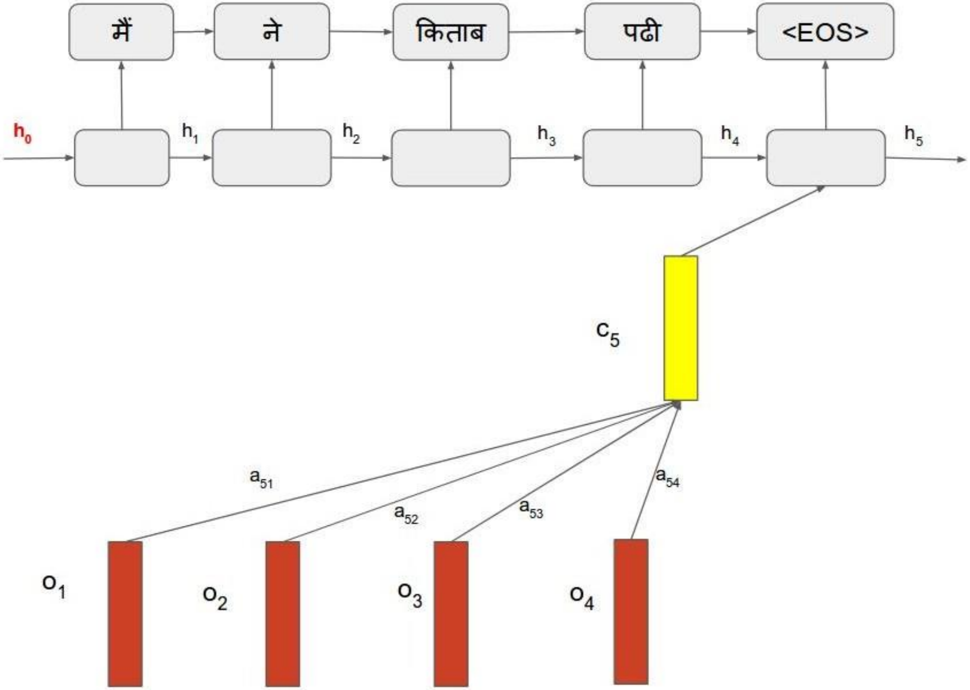
Encoder-Decoder attention



Encoder-Decoder attention



Encoder-Decoder attention



Annotation weights

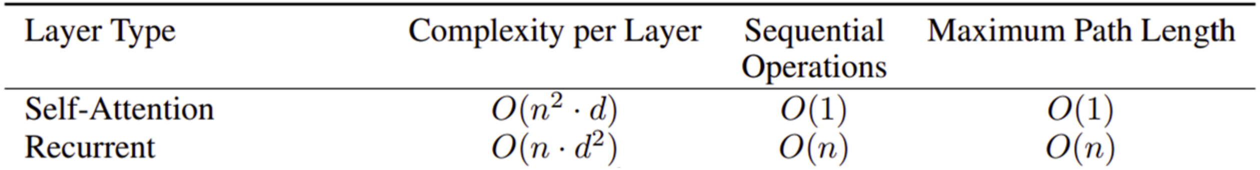
- How do we find the annotation weights?
    - For a word in target sentence, these are softmax computed **alignment vectors**.

- How do we find the alignment vectors?
    - Pick the attention weights that maximize the translation accuracy (more precisely, decrease training data loss)
        - Jointly learn to align and translate

Why Transformer?

Motivation

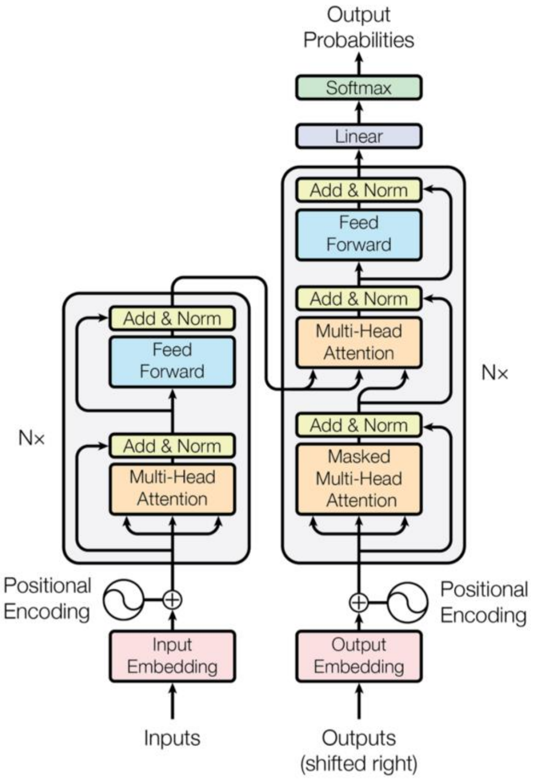
- Transformer was introduced as a seq2seq architecture in the context of **machine translation**
    - to allow parallel computation (to reduce training time)
    - to reduce drops in performance due to long-term dependencies for long sentences



Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$

Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension.

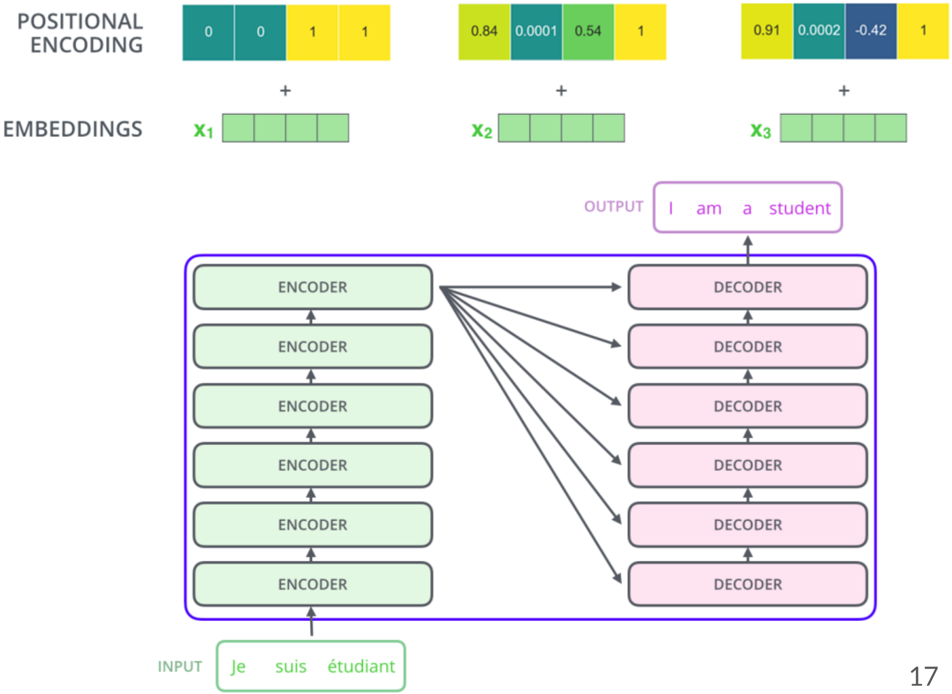
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "**Attention is all you need.**" Advances in neural information processing systems 30 (2017).



Transformer architecture

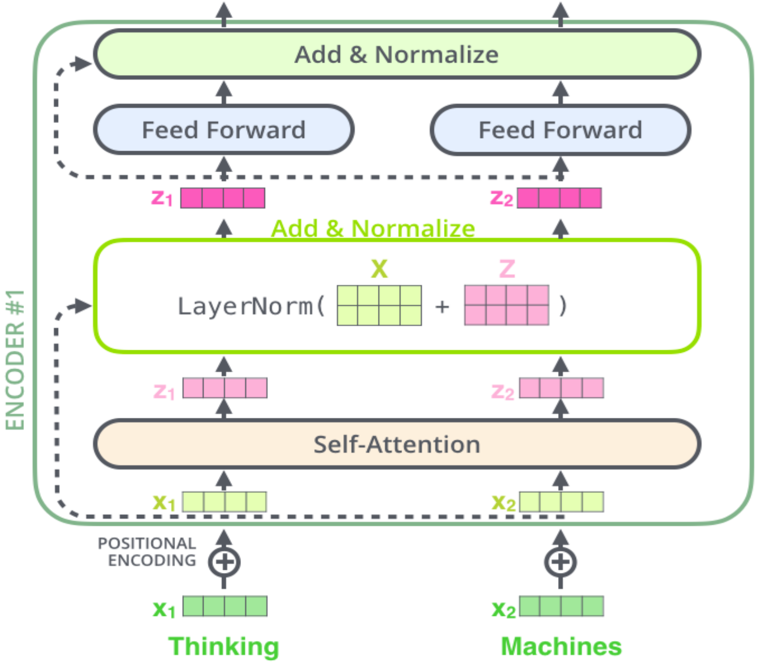
Transformer

- Embedding-
  - With word-embedding they add positional encoding. To set a constant and small vector_size they use a strategy for which model can translate a sentence long sentences of training set.
- After that, there is an encoder-decoder architecture with Transformers instead of RNN.



Transformer encoder

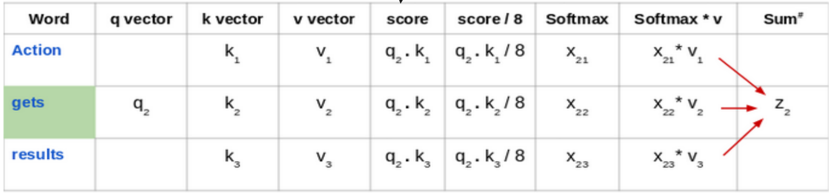
1. **Self attention (input x; output z)-**
   a. Multi-head attention- We will process input vectors with all these for 8 sets (parameter). Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions.
   b. Train W_Q, W_K, W_V to get Q, K, V. (8 sets for each word).
   c. Prepare Z_is.
   d. Z_i to Z —> concatenate then train W_o to to transform into a z_i size vector.
2. Residual connection: Add and normalise
   LayerNorm(x+z)
3. Feedforward
4. Add and normalise



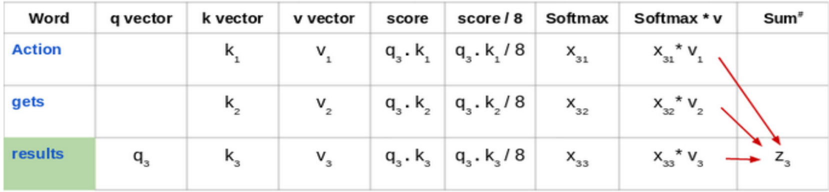
Self attention

1. Self attention: For each word x_i of position i
   a. Multi-head attention- We will process input vectors with all these for 8 sets (parameter). Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions.
   b. Get Q, K, V (8 sets for each word) by multiplying x_i with W_Q, W_K, W_V (trainable).
   c. Prepare Z_js from these for each head j.
   d. Z_j to Z —> concatenate then train W_o to transform into a x_i size vector.
2. Residual connection: Add and normalise LayerNorm(x+z)
3. Feedforward
4. Add and normalise

Attention scores

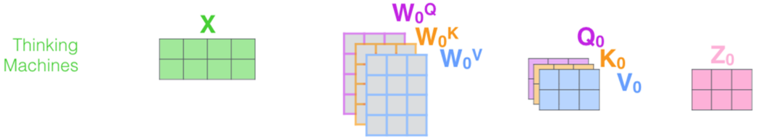


Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum[#]
Action		k_1	v_1	$q_2 . k_1$	$q_2 . k_1 / 8$	x_{21}	$x_{21} * v_1$	
gets	q_2	k_2	v_2	$q_2 . k_2$	$q_2 . k_2 / 8$	x_{22}	$x_{22} * v_2$	z_2
results		k_3	v_3	$q_2 . k_3$	$q_2 . k_3 / 8$	x_{23}	$x_{23} * v_3$	

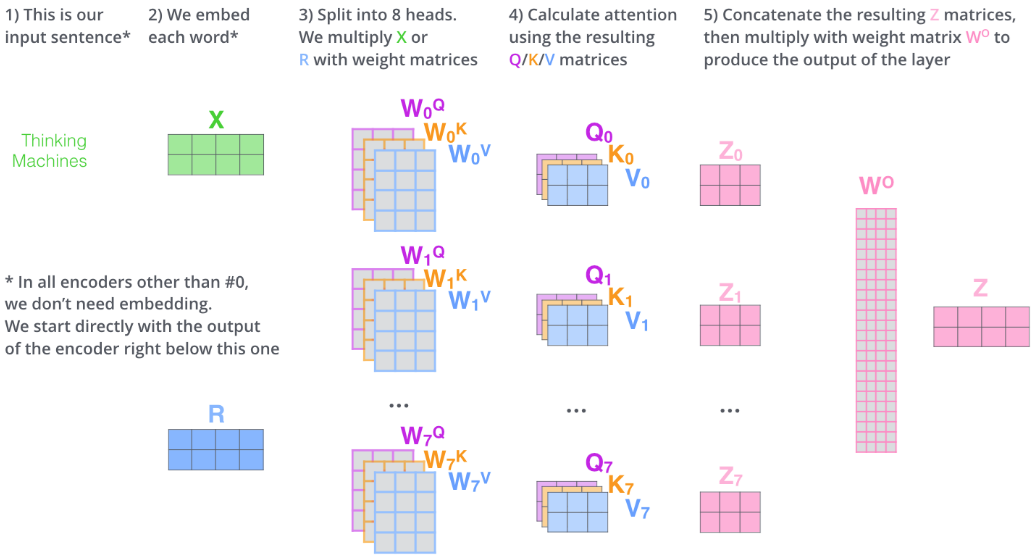


Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum[#]
Action		k_1	v_1	$q_3 . k_1$	$q_3 . k_1 / 8$	x_{31}	$x_{31} * v_1$	
gets		k_2	v_2	$q_3 . k_2$	$q_3 . k_2 / 8$	x_{32}	$x_{32} * v_2$	
results	q_3	k_3	v_3	$q_3 . k_3$	$q_3 . k_3 / 8$	x_{33}	$x_{33} * v_3$	z_3

Demo of self-attention (Query size is 64 for each head)



Multi-head attention



1) This is our input sentence*

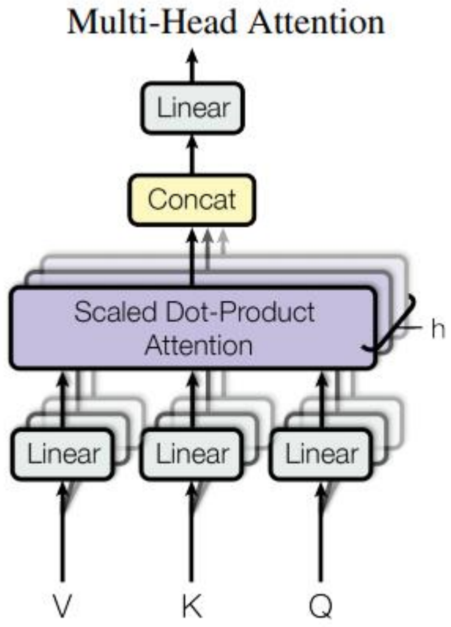
2) We embed each word*

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

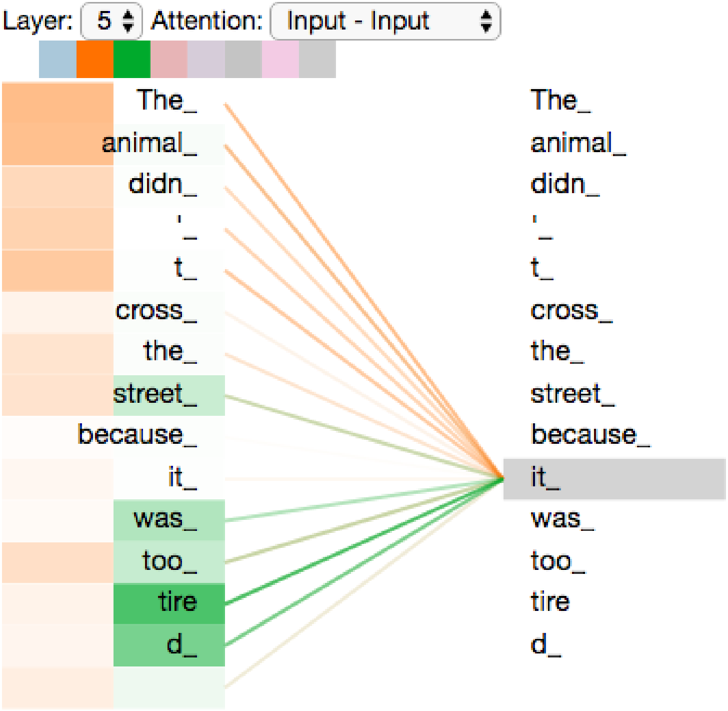
4) Calculate attention using the resulting Q/K/V matrices

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

* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one



Why multi-head attention?

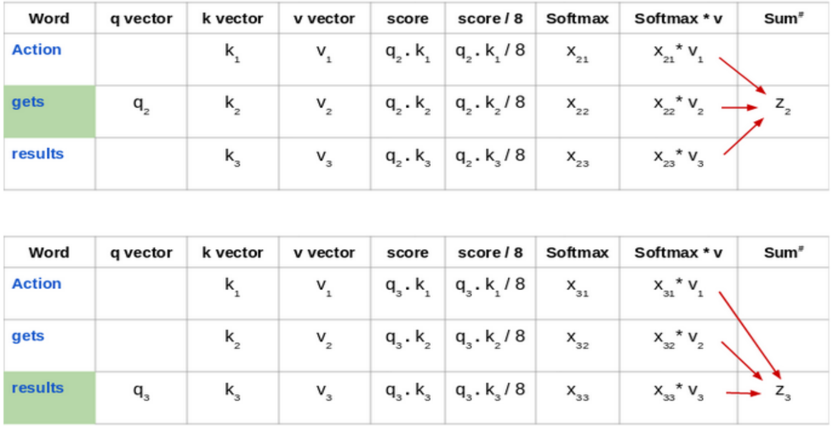


- The figure is a visualization of the outputs upon using 2 heads.
- If the Query word is ‘it’, the first head focuses more on the words ‘the’, ‘animal’, and the second head focuses more on the word ‘tired’.

Source: https://blogs.oracle.com/

Transformer encoder

1. Self attention: For each word x_i of position i
   a. Multi-head attention- We will process input vectors with all these for 8 sets (parameter). Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions.
   b. Get Q, K, V (8 sets for each word) by multiplying x_i with W_Q, W_K, W_V (trainable).
   c. Prepare Z_js from these for each head j.
   d. Z_j to Z —> concatenate then train W_o to transform into a x_i size vector.
2. Residual connection: Add and normalise LayerNorm(x+z)
3. Feedforward
4. Add and normalise



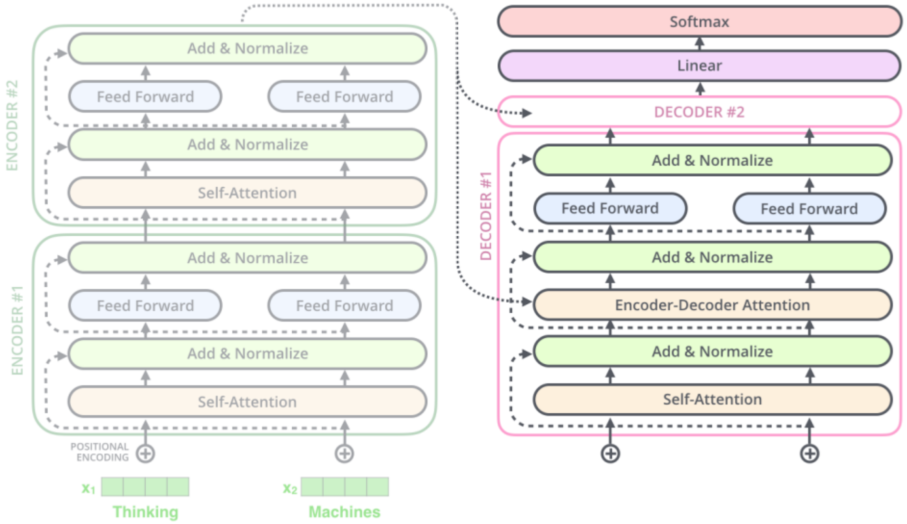
Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum#
Action		k_1	v_1	$q_2 . k_1$	$q_2 . k_1 / 8$	x_{21}	$x_{21} * v_1$	
gets	q_2	k_2	v_2	$q_2 . k_2$	$q_2 . k_2 / 8$	x_{22}	$x_{22} * v_2$	z_2
results		k_3	v_3	$q_2 . k_3$	$q_2 . k_3 / 8$	x_{23}	$x_{23} * v_3$	

Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum#
Action		k_1	v_1	$q_3 . k_1$	$q_3 . k_1 / 8$	x_{31}	$x_{31} * v_1$	
gets		k_2	v_2	$q_3 . k_2$	$q_3 . k_2 / 8$	x_{32}	$x_{32} * v_2$	
results	q_3	k_3	v_3	$q_3 . k_3$	$q_3 . k_3 / 8$	x_{33}	$x_{33} * v_3$	z_3

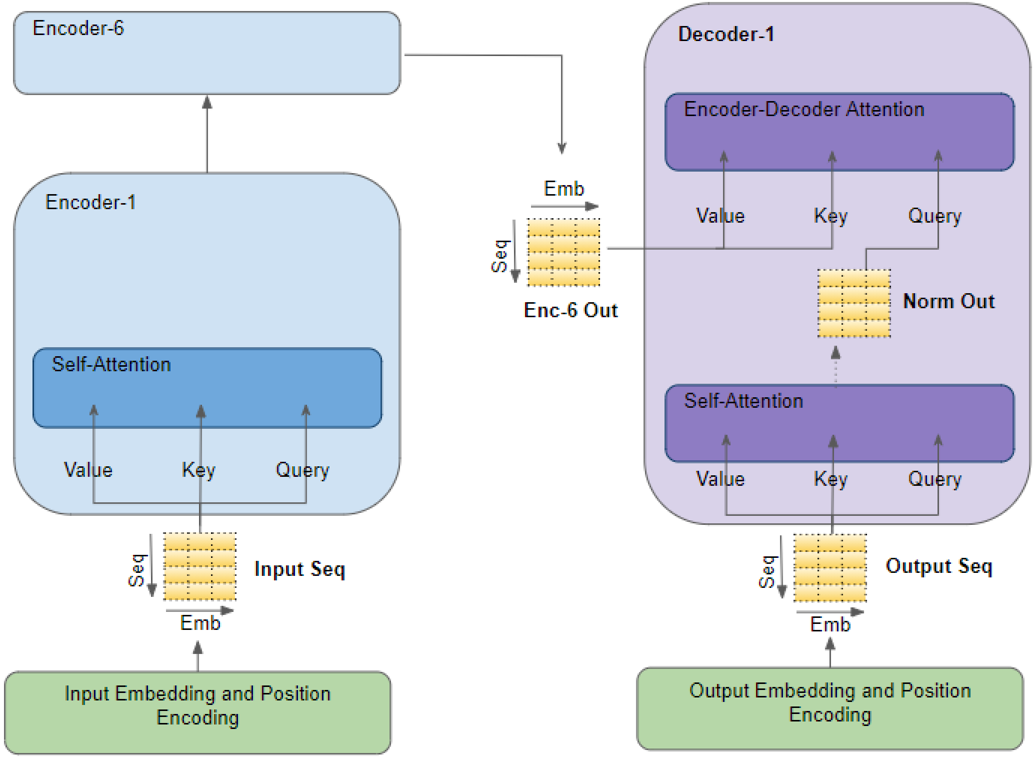
Demo of self-attention (Query size is 64 for each head)

Transformer decoder

1. Self-attention: In decoder side the self-attention layer is **only allowed to attend to earlier positions in the output sequence**. This is done by masking future positions (setting them to -inf)
2. Add and normalize
3. **Encoder-decoder attention**: Just like multiheaded self-attention, except **it creates its Queries matrix from the layer below it, and takes the Keys and Values matrix from the output of the encoder stack.**
4. Add and normalize
5. Feedforward
6. Add and normalize

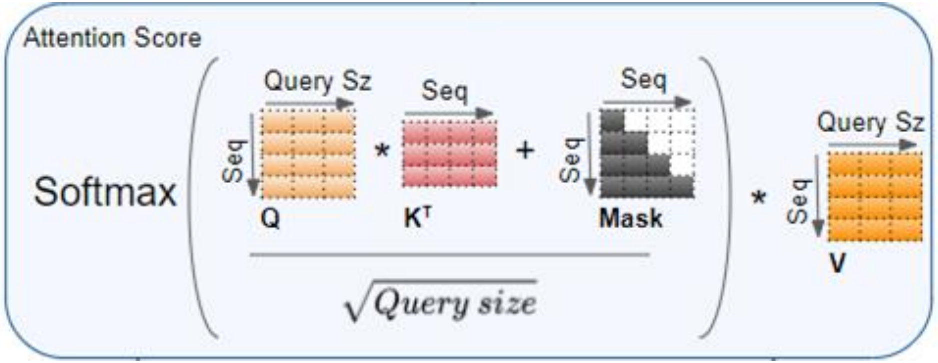


Encoder-Decoder attention

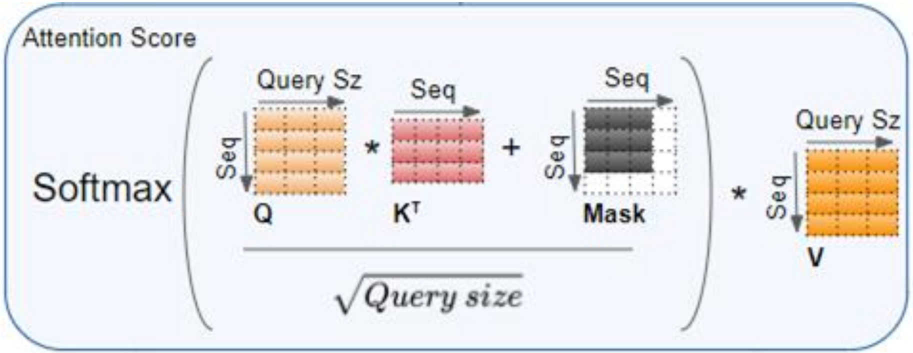


It creates its Queries matrix from the layer below it, and takes the Keys and Values matrix from the output of the encoder stack.

Decoder attention with masking

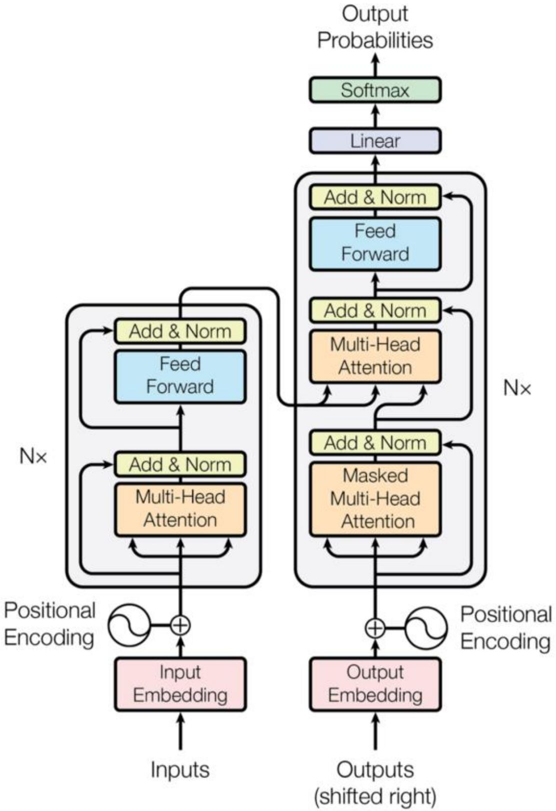


Decoder self attention with masking



Encoder-Decoder attention with masking

Full architecture



Summary

- Transformer is faster than RNN as it allows parallel computation.
- Transformer improves output performance by handling long-term dependencies for long sentences.
- Sentences are processed non-sequentially as a whole rather than word by word.
- Self attention computes similarity scores between words in a sentence.
- Positional embeddings are used to incorporate position information.

References

[1] Vaswani, Ashish, et al. "Attention is all you need." Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017.

Acknowledgement

- http://jalammar.github.io/illustrated-transformer/
- https://towardsdatascience.com/transformers-explained-visually-part-3-multi-head-attention-deep-dive-1c1ff1024853

Thank you!

Attention mechanism

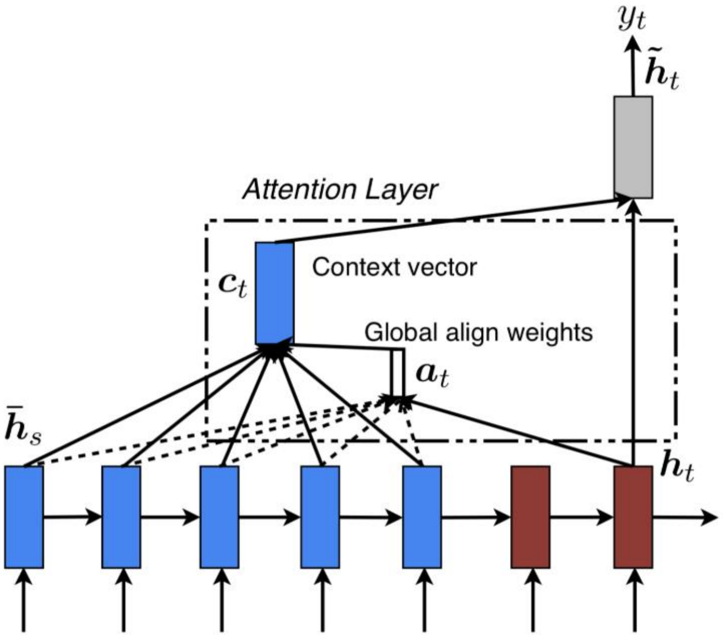
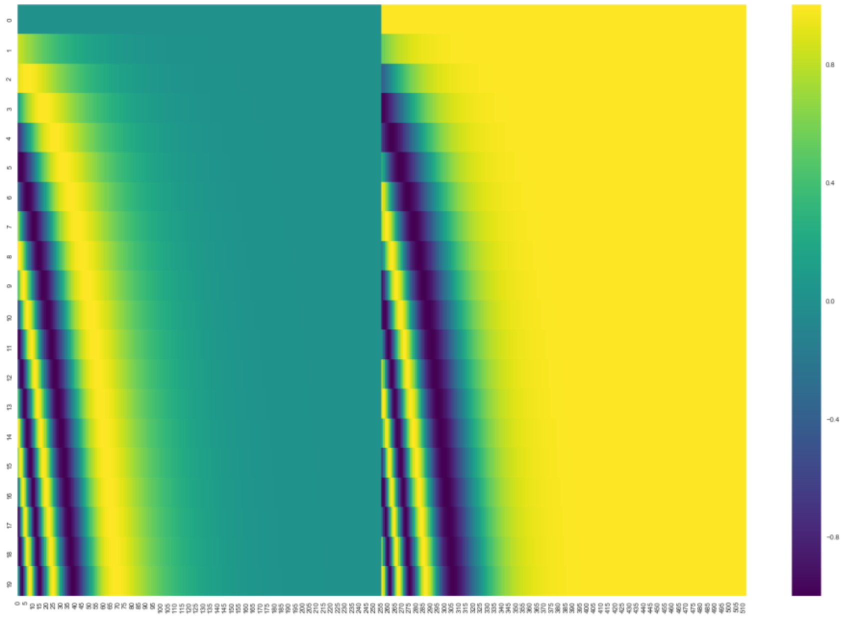
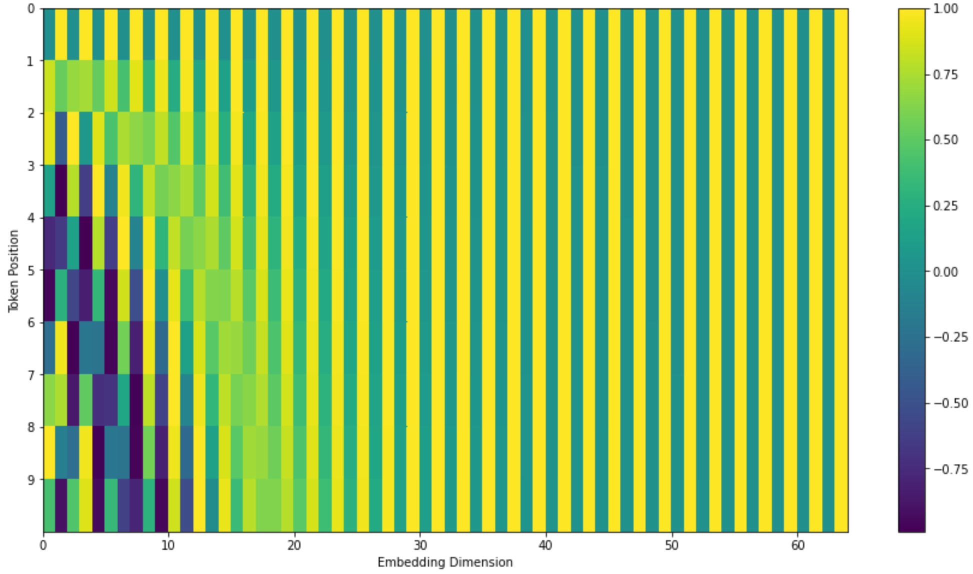


Image source- [6]

Positional Encoding: Sinusoidal





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

Image source- [2]

Understanding Transformer and its variants

Presented by
Tamali Banerjee
Research scholar, IIT Bombay
tamali@cse.iitb.ac.in

For
CS772 at IIT Bombay, March 2022
Course Instructor: Prof. Pushpak Bhattacharyya

- Main components of Transformer are:
  - Positional encoding
  - Multihead self-attention
  - Feed-forward layer
  - Residual connection
  - Decoder masking
  - Enc-dec attention
  - Linear classifier

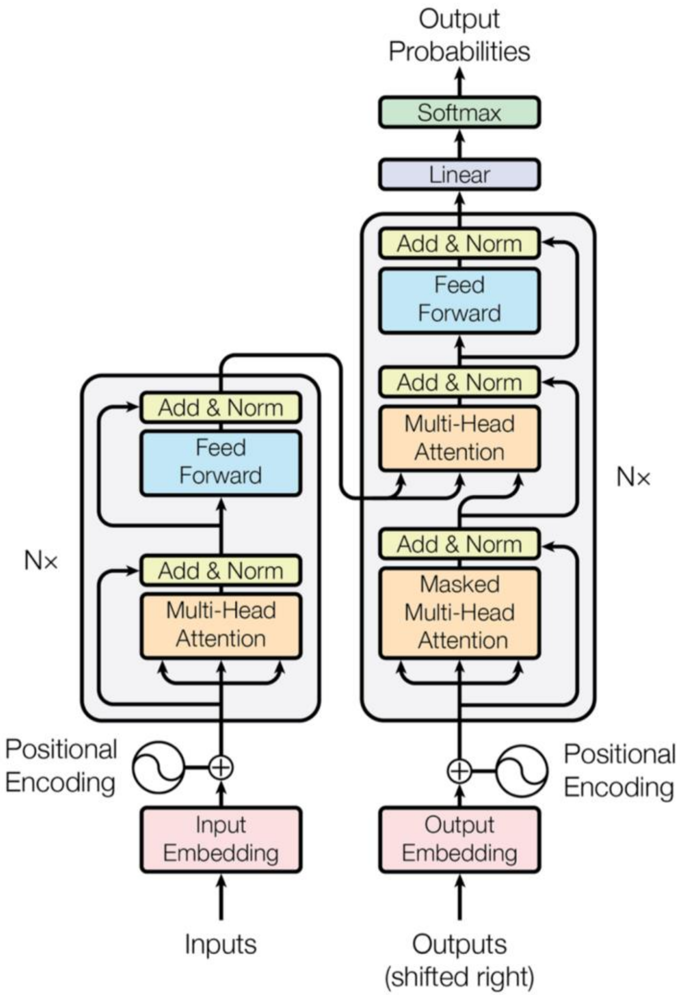


Image source: [1]

Positional encoding

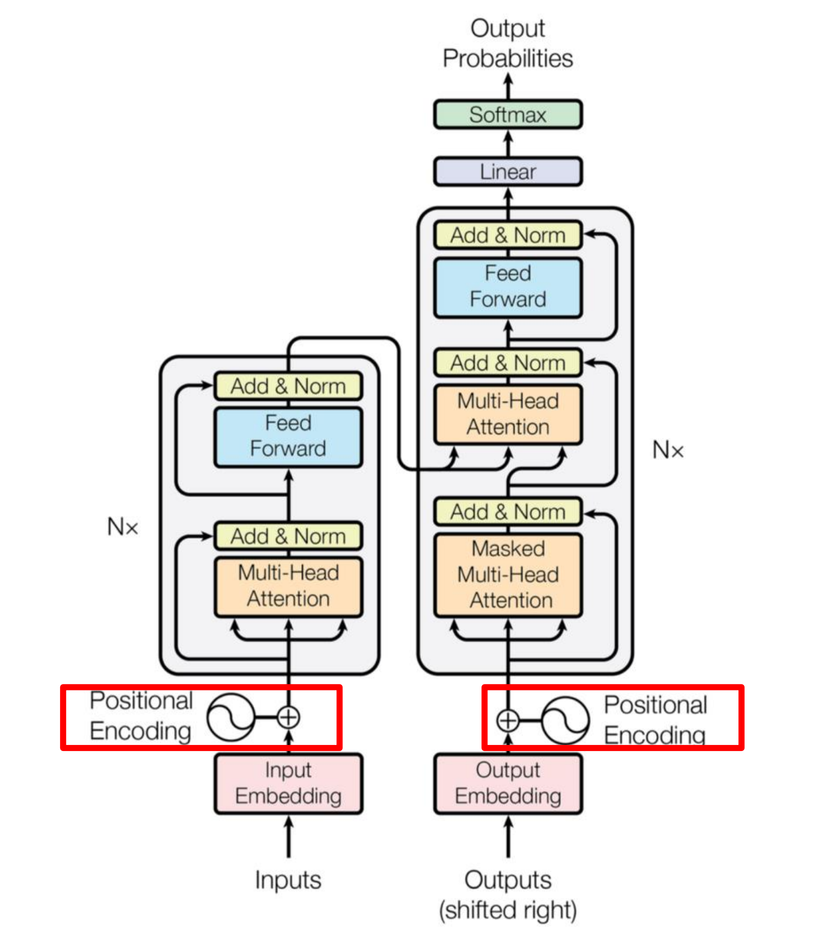


Image source: [1]

Need for Positional encoding/embedding

- RNNs have positional information as it takes input of one word-vector at a time.
- To incorporate positional information in Transformer we need to give positional encoding as a part of input.
- Positional Encoding (vector) should be of a fixed length.
- Encoder gets input of vectors. Each of these vectors is summation of the Positional Encoding (depends on position of the word) and the word-embeddings (depends on meaning of the word).

Criteria of an ideal positional encoding/embedding

- It should output a unique encoding for each time-step (word’s position in a sentence).
- Distance between any two time-steps should be consistent across sentences with different lengths.
- The model should generalize to longer sentences without any efforts.
- Its values should be bounded.
- It must be deterministic.

Possible positional encoding strategies

Strategy 1:

P1	P2	P3
1	2	3
1	2	3
1	2	3
1	2	3

👎 Words appearing in the later part of the sentence will be more distorted than first word of the sentence. These should be within a range.

Strategy 2:

P1	P2	P3
0	0.33	1
0	0.33	1
0	0.33	1
0	0.33	1

👎 It will be different for the same position for different sentences. It can be accidentally same for different position because of different sentence length.

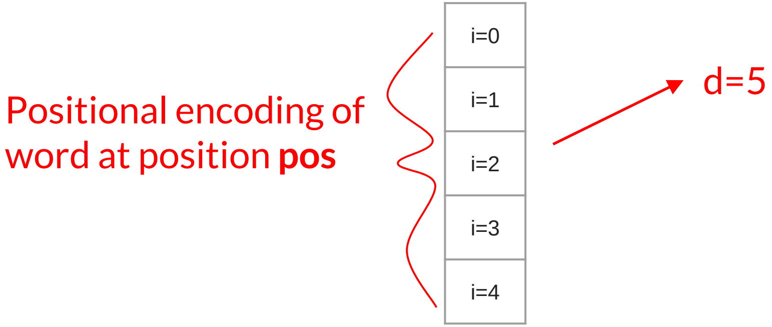
Strategy 3:

P1	P2	P3
0	0	0
0	0	0
0	0	1
0	1	0

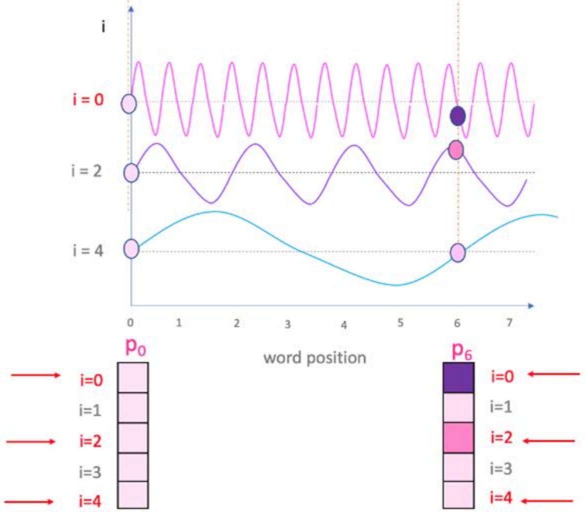
👎 These vectors cannot be fitted into given vector size.

Sinusoidal function with different frequencies

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



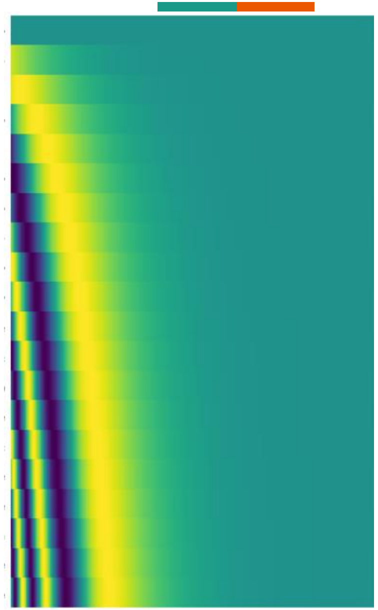
Positional encoding of word at position **pos**



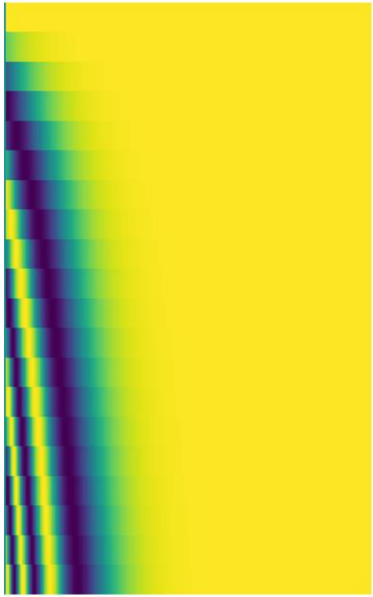
- Values for lower indexes are changing quickly higher indexes require a lot of positions to change a value.
- Cosine_similarity between PE(pos) and PE(pos+1) will be greater than PE(pos) and PE(pos+10).

Image source- [https://tamoghnasaha-22.medium.com/transformers-illustrated-5c9205a6c70f]

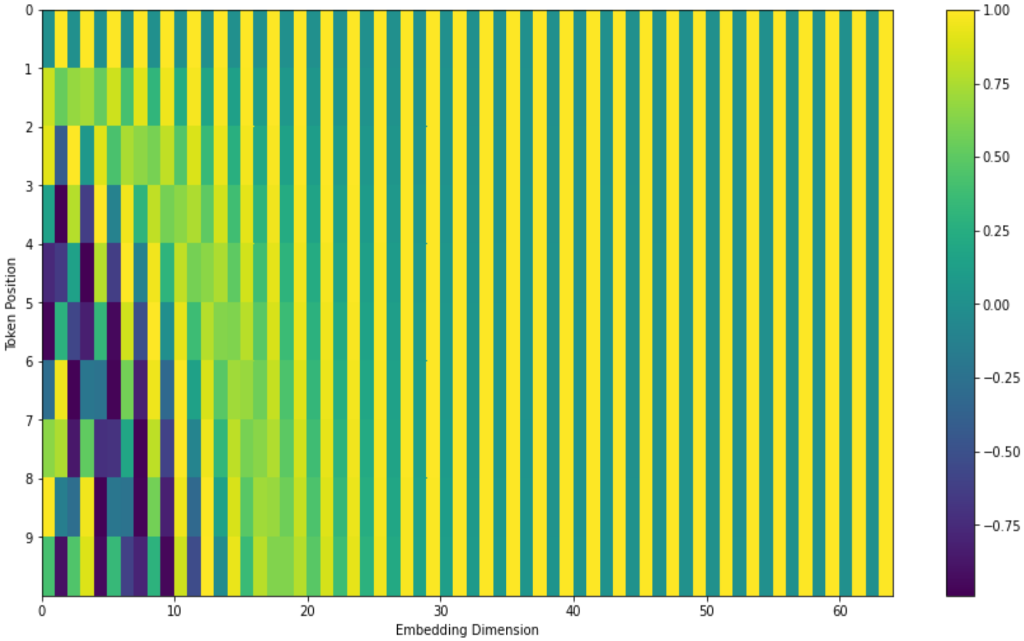
Visualizing the Positional Matrix



sin func for even positions in the sentence



cos func for odd positions in the sentence



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

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

Multi-head attention

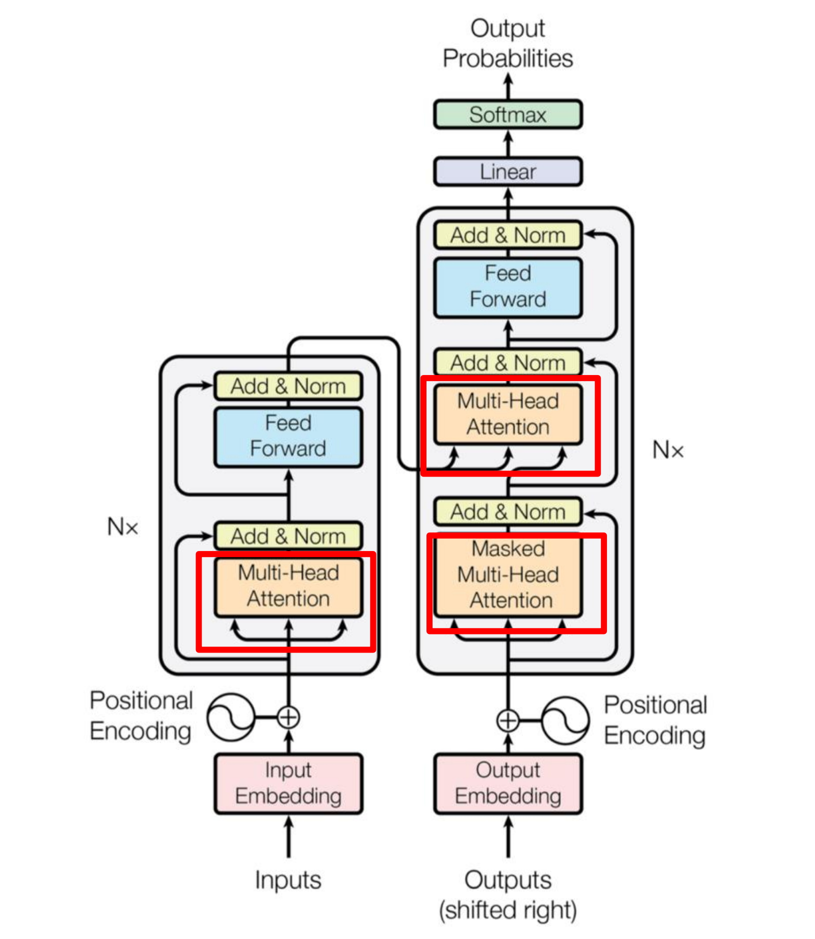
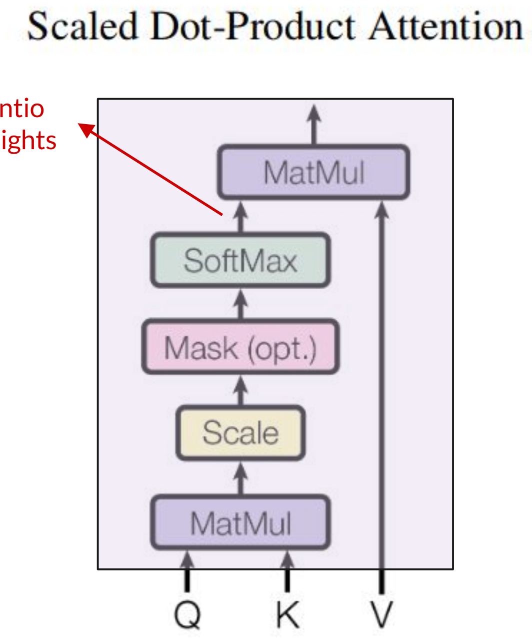


Image source: [1]

Query, Key and Value

- This concept is analogous to IR systems.
  - Query: The vector that helps to search its important neighboring words.
    - Query = input_vector * W_Q
  - Key: The vector that signifies its key features to be important
    - Key = input_vector * W_K
  - Value: It consists of meaning of the word in the sentence which the word contributes as information to form self-attention output.
    - Value = input_vector * W_V
- Scaled dot product attention of Q (of word 'x') and K (of word 'y') is attention weight of word 'y' while calculating self-attention output of word 'x'.
  - Note here, attention scores are not symmetric.

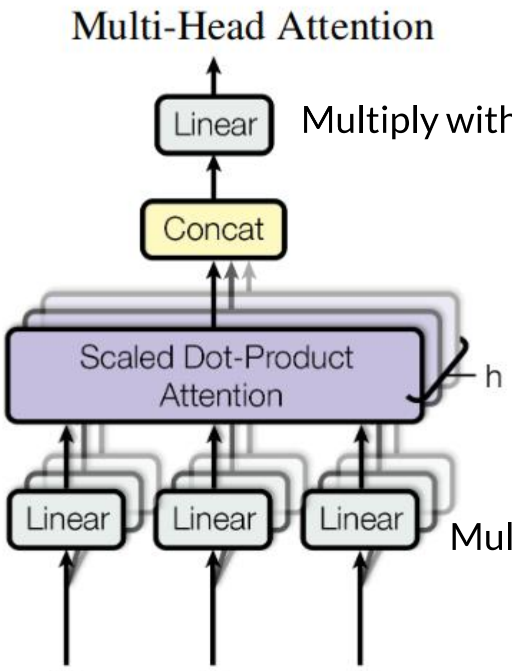
Multi-head attention



Scaled Dot-Product Attention

Attention weights

Weighted average of Values



Multi-Head Attention

Multiply with W_o

Multiply with W_Q , W_K , W_V

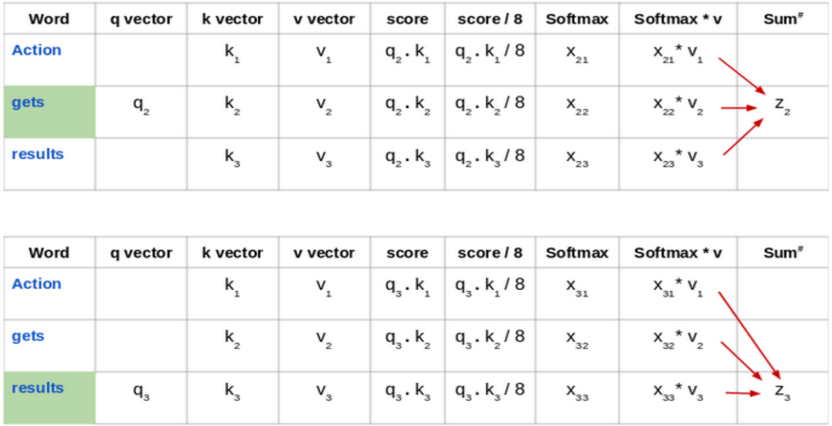
Image source: [1]

Transformer encoder

1. Self attention: For each word x_i of position i
   a. Multi-head attention: We will process input vectors with all these for 8 sets (parameter). Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions.
   b. Get Q, K, V (8 sets for each word) by multiplying x_i with W_Q, W_K, W_V (trainable).
   c. Prepare Z_js from these for each head j.
   d. Z_j to Z → concatenate then train W_o to transform into a x_i size vector.
2. Residual connection: Add and normalise LayerNorm(x+z)
3. Feedforward
4. Add and normalise

Attentio
n weights

Weighted
average of
Vs



Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum$^{\#}$
Action		k_1	v_1	$q_2 \cdot k_1$	$q_2 \cdot k_1 / 8$	x_{21}	$x_{21} * v_1$	
gets	q_2	k_2	v_2	$q_2 \cdot k_2$	$q_2 \cdot k_2 / 8$	x_{22}	$x_{22} * v_2$	z_2
results		k_3	v_3	$q_2 \cdot k_3$	$q_2 \cdot k_3 / 8$	x_{23}	$x_{23} * v_3$	

Word	q vector	k vector	v vector	score	score / 8	Softmax	Softmax * v	Sum$^{\#}$
Action		k_1	v_1	$q_3 \cdot k_1$	$q_3 \cdot k_1 / 8$	x_{31}	$x_{31} * v_1$	
gets		k_2	v_2	$q_3 \cdot k_2$	$q_3 \cdot k_2 / 8$	x_{32}	$x_{32} * v_2$	
results	q_3	k_3	v_3	$q_3 \cdot k_3$	$q_3 \cdot k_3 / 8$	x_{33}	$x_{33} * v_3$	z_3

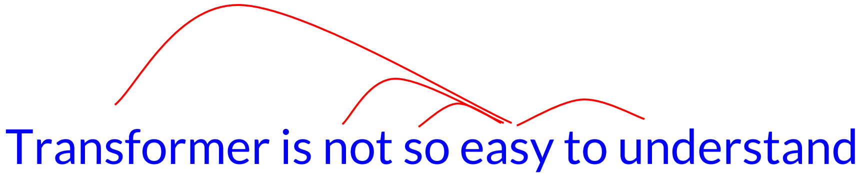
Demo of self-attention (Query size is 64 for each head)

Study on multi-heads

- Researchers [3] identified 3 types of important heads by looking at their attention matrices:
    - Positional heads that attend mostly to their neighbor.
    - Syntactic heads that point to tokens with a specific syntactic relation.
    - Heads that point to rare words in the sentence.

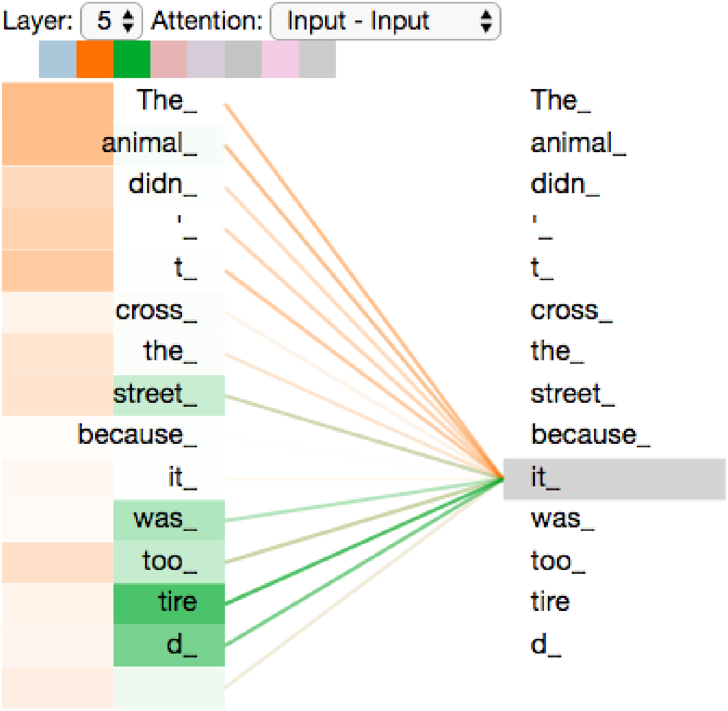
Purpose of multiple heads

- A sentence is a mixture of multiple information.
- A single word can have multiple type of relations with other words of the sentence.
- Multi-headed attention was introduced due to the observation that different words relate to each other in different ways.



Transformer is not so easy to understand

Example of Multi-head attention



Layer: 5 Attention: Input - Input

- The figure is a visualization of the outputs upon using 2 heads.
- If the Query word is ‘it’, the first head focuses more on the words ‘the’, ‘animal’, and the second head focuses more on the word ‘tired’.

Source: https://blogs.oracle.com

Position-wise feed-forward layer

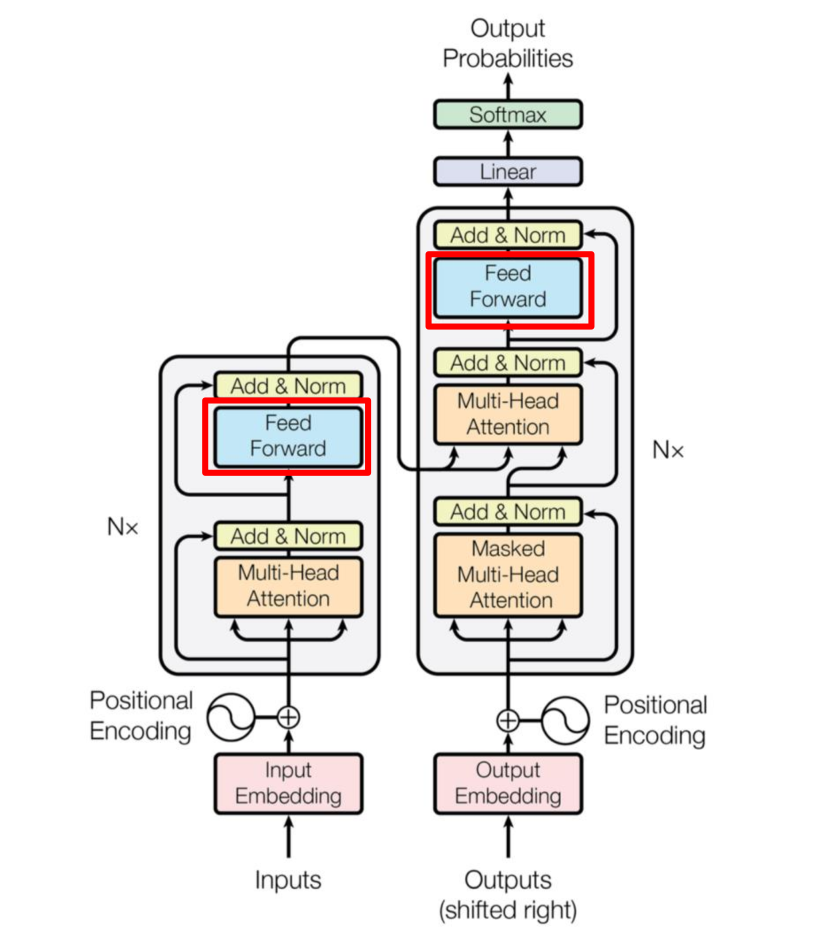


Image source: [1]

Position-wise feed-forward layer

- After going through multi-headed self-attention layer, output vector of every input word is now aware of its context.
- After that, it processes these information.
- The FF function is applied to each position **separately** and **identically**.
- It consists of two linear transformations with a ReLU activation in between.

$$\mathrm{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2$$

- A study [3] found that, simply stacking self-attention modules without FF layer causes a training issues.

Residual connection

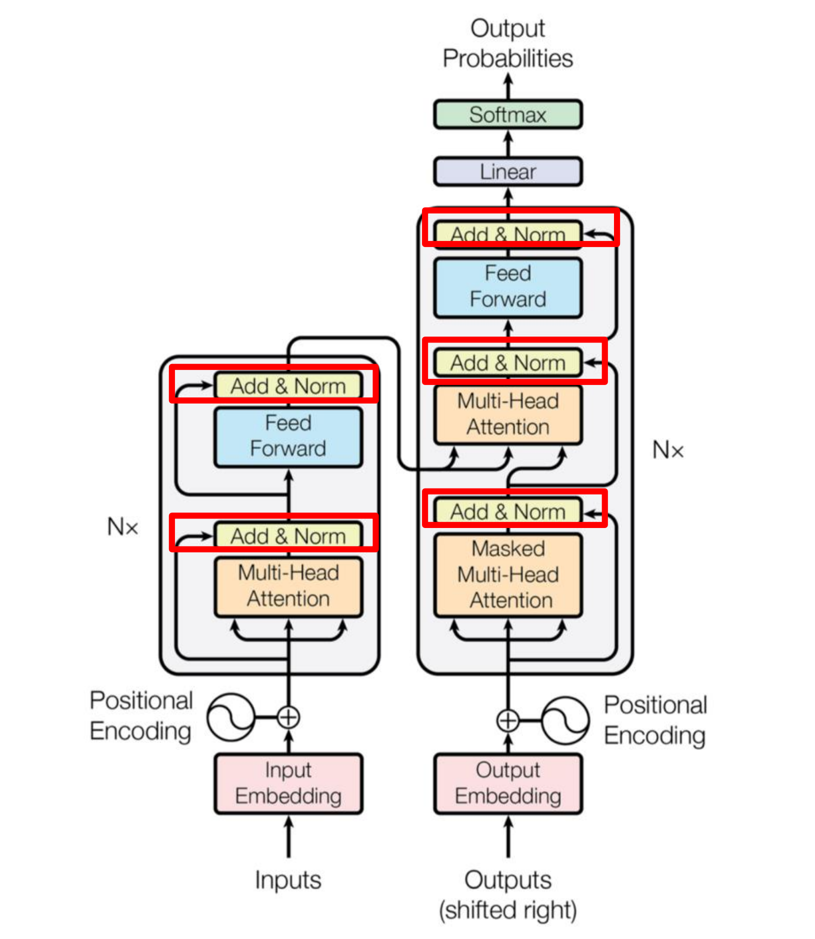
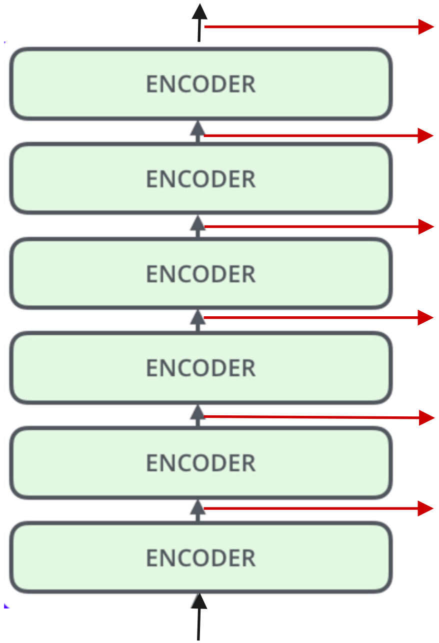


Image source: [1]

Purpose of Residual connection

- Summation of input and output: layernorm(input+output)
- Residual connection makes the gradient flow smoothly to previous layer.
- When gradient flows through multi-head attention layer, vanishing gradient problem is observed.
    - **Vanishing gradient:** As the backpropagation algorithm advances downwards(or backward) from the output layer towards the input layer, the gradients often get smaller and smaller and approach zero which eventually leaves the weights of the initial or lower layers nearly unchanged.
- The information from the input of the model (which contains positional embeddings) can efficiently propagate to further layers where the more complex interactions are handled.
- The layer normalizations are used to stabilize the network which results in substantially reducing the training time necessary.
- The layer norm trainable parameters (0.1% of the parameters) to be the most crucial for fine-tuning transformers, after pre-training.

Stack of encoders



layer6_op$_1$ layer6_op$_2$ … layer6_op$_n$

layer5_op$_1$ layer5_op$_2$ … layer5_op$_n$

layer4_op$_1$ layer4_op$_2$ … layer4_op$_n$

layer3_op$_1$ layer3_op$_2$ … layer3_op$_n$

layer2_op$_1$ layer2_op$_2$ … layer2_op$_n$

layer1_op$_1$ layer1_op$_2$ … layer1_op$_n$

ip$_1$ ip$_2$ … ip$_n$

[n no of words]

Decoder masking

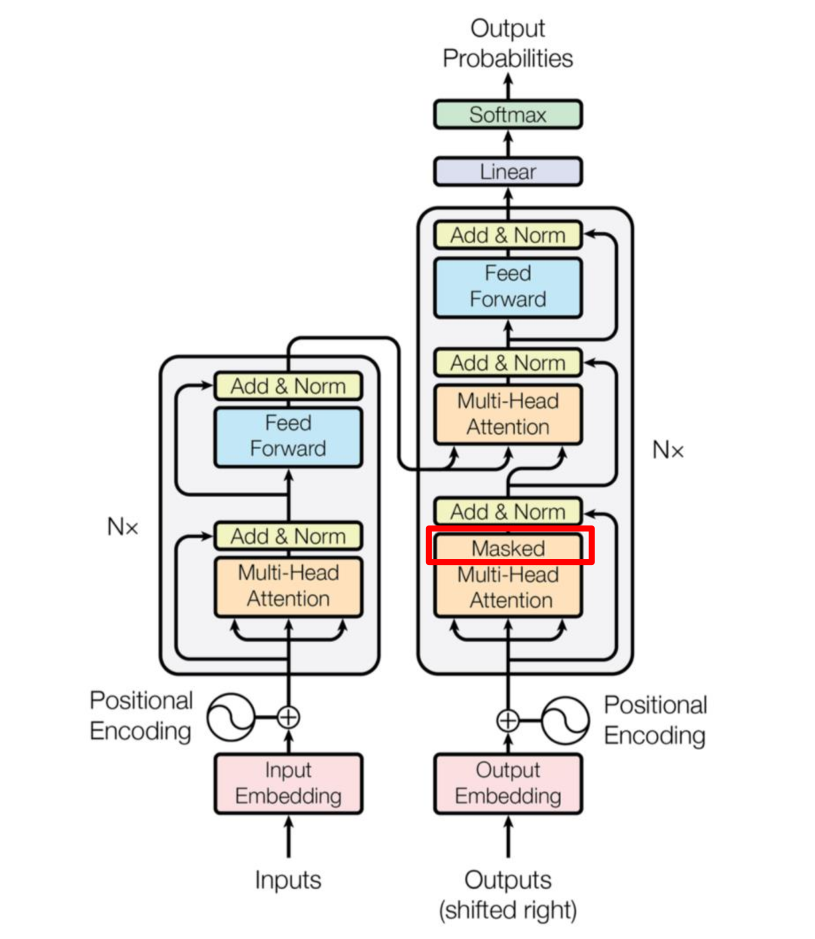
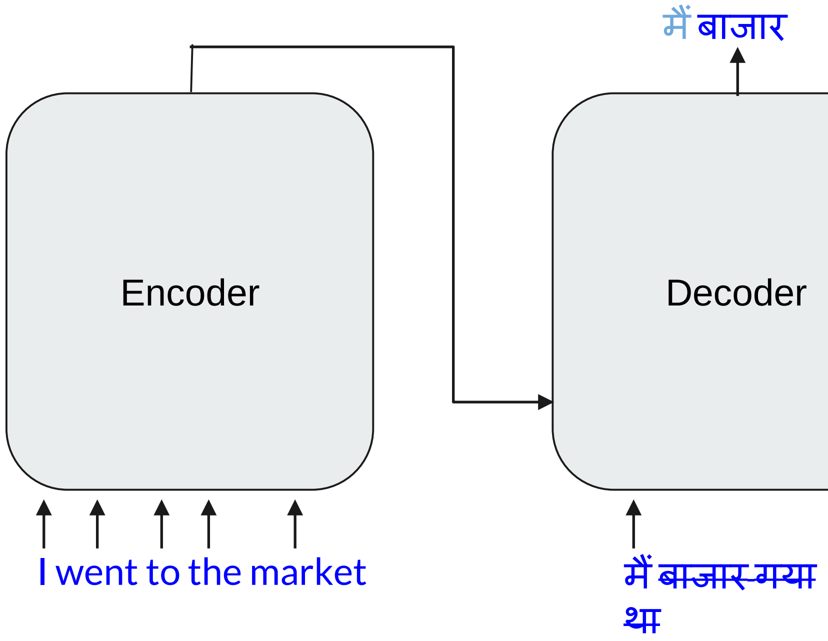


Image source: [1]

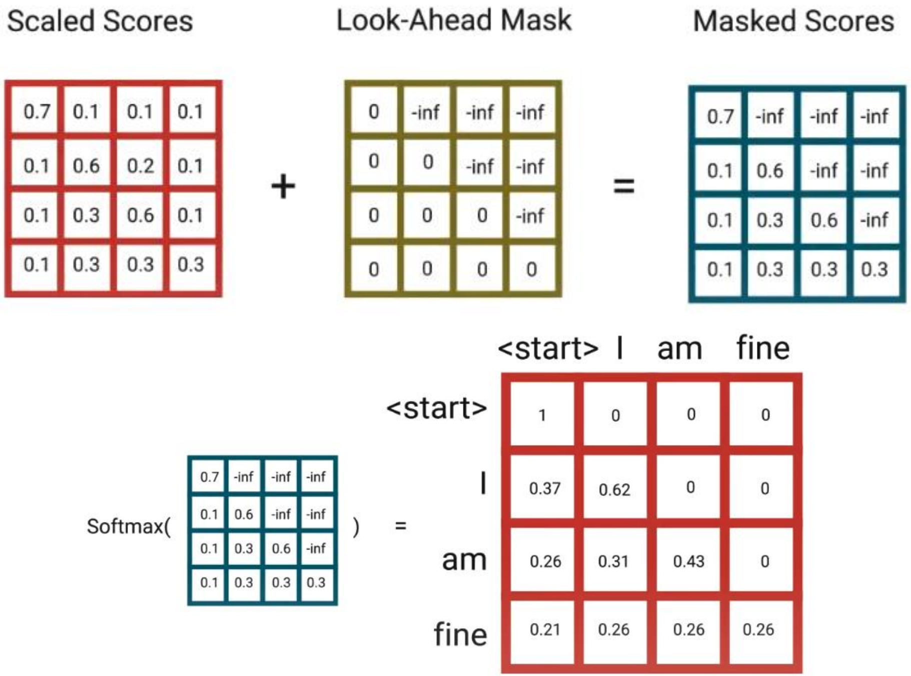
Need for decoder masking



How decoder operates?

- The decoder is autoregressive.
- It begins with a special token <start>.
- Then it generates the next possible word.
- Then it takes the previous output(s) as input(s) and again that encoder outputs.
- Then it generates the next possible word, and this process goes on.
- The decoder stops decoding when it generates <eos> token as an output.

Decoder attention with masking



Scaled Scores

0.7	0.1	0.1	0.1
0.1	0.6	0.2	0.1
0.1	0.3	0.6	0.1
0.1	0.3	0.3	0.3

+

Look-Ahead Mask

0	-inf	-inf	-inf
0	0	-inf	-inf
0	0	0	-inf
0	0	0	0

=

Masked Scores

0.7	-inf	-inf	-inf
0.1	0.6	-inf	-inf
0.1	0.3	0.6	-inf
0.1	0.3	0.3	0.3

Softmax(

0.7	-inf	-inf	-inf
0.1	0.6	-inf	-inf
0.1	0.3	0.6	-inf
0.1	0.3	0.3	0.3

) =

	<start>	I	am	fine
<start>	1	0	0	0
I	0.37	0.62	0	0
am	0.26	0.31	0.43	0
fine	0.21	0.26	0.26	0.26

- To mask the positions [-inf] is added.
- After softmax they become 0.
- That means, attention weights for the words of these positions become 0.

Image source: [https://tamoghnasaha-22.medium.com/transformers-illustrated-5c9205a6c70f]

Encoder-decoder attention

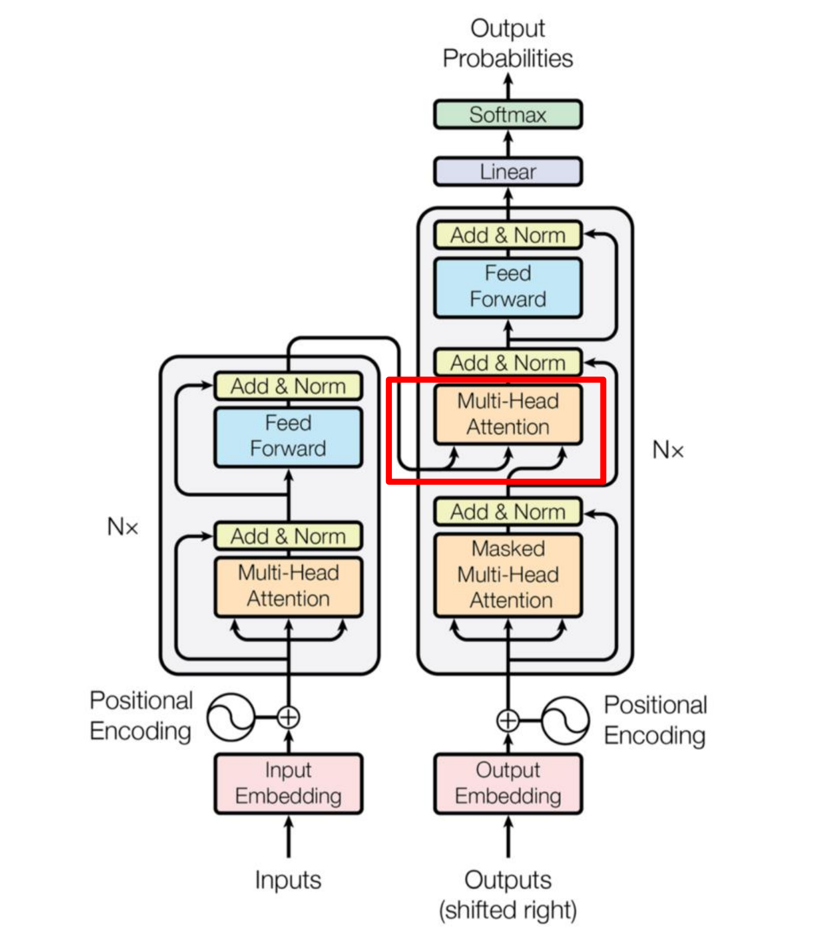


Image source: [1]

Query, Key, Value of enc-dec attention

- **Query**: Query comes from output of last layer.
- **Key**: Key comes from output of last encoder layer.
- **Value**: Value comes from output of last encoder layer.

➔ It means that- an output word uses its **Query** vector to generate the **context vector** from the input sentence from last-layer encoder outputs.

- ◆ Recall how context vectors were generated in RNN-based enc-attn-dec models.

Stack of encoders and decoders

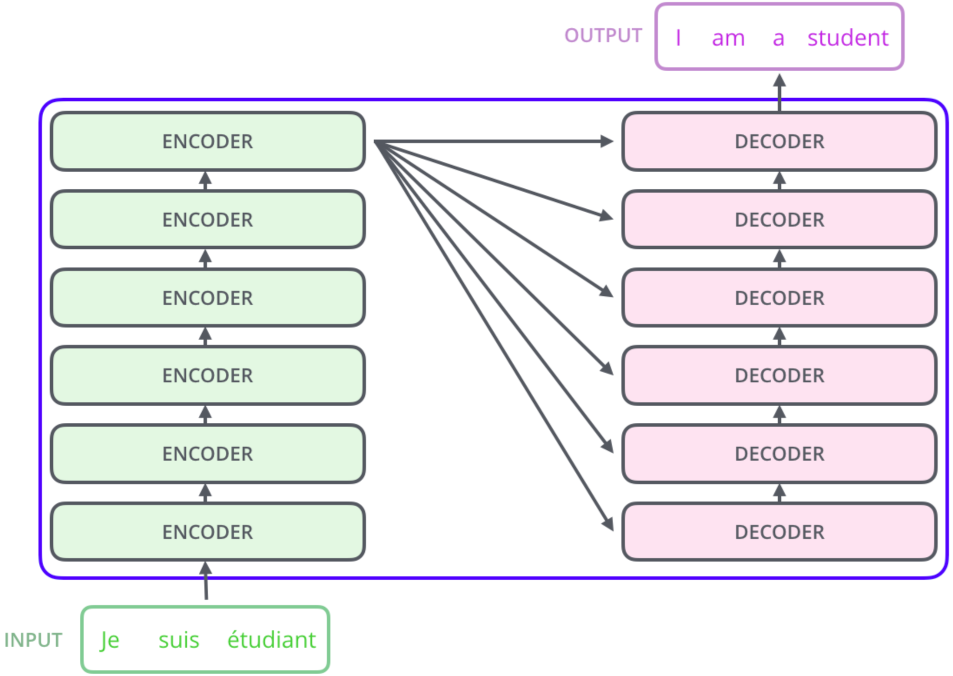


Image source- [https://jalammar.github.io/illustrated-transformer]

Linear classifier and softmax

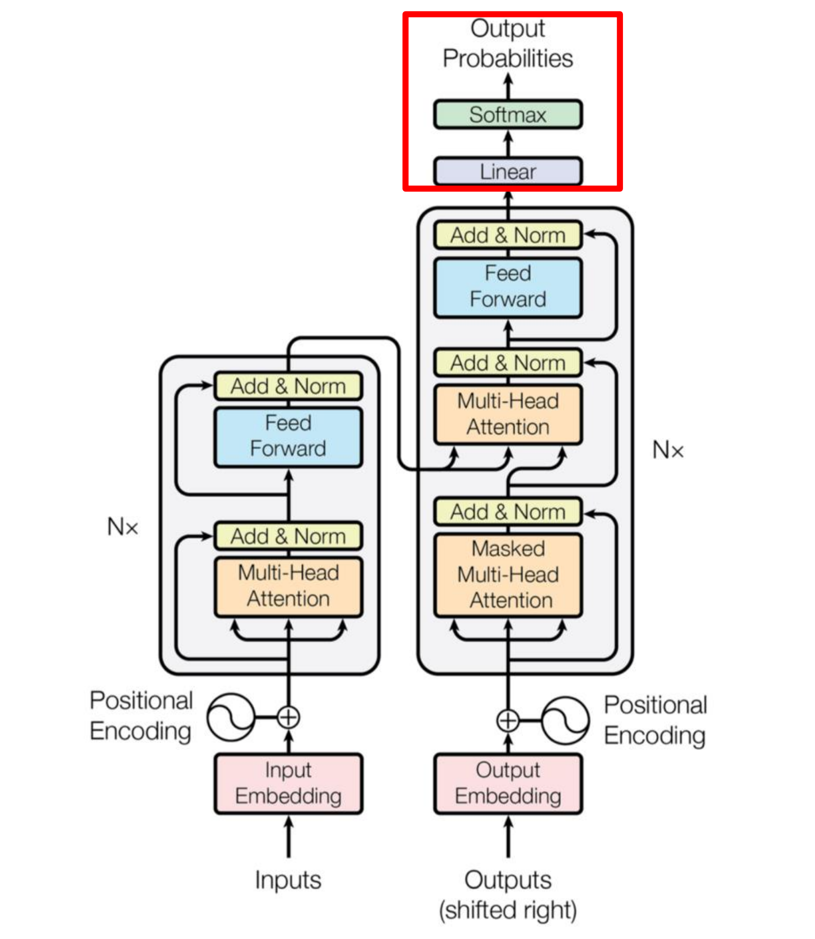


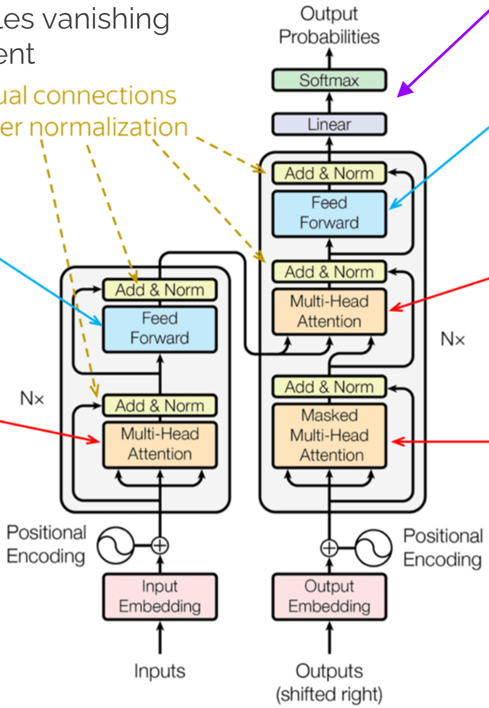
Image source: [1]

Linear classifier and softmax

- The output of the final pointwise feedforward layer goes to a linear classifier as input.
- The size of the classifier is same as the output vocabulary size.
- The output of the classifier then gets fed into a softmax layer, which will produce probability scores of each word of vocabulary between 0 and 1.
- Output is the word for which the probability score is highest.

Summary

Summary



Classifier:
outputs probability of each word to be generated at this position

Handles vanishing gradient

Residual connections and layer normalization

Feed-forward network:

processes each word separately

↑

Encoder self-attention:
tokens look at each other

queries, keys, values are computed from encoder states

Feed-forward network:

processes each word separately

↑

Decoder-encoder attention:
target token looks at the source

queries – from decoder states; keys and values from encoder states

↑

Decoder self-attention (masked):
tokens look at the previous tokens

queries, keys, values are computed from decoder states

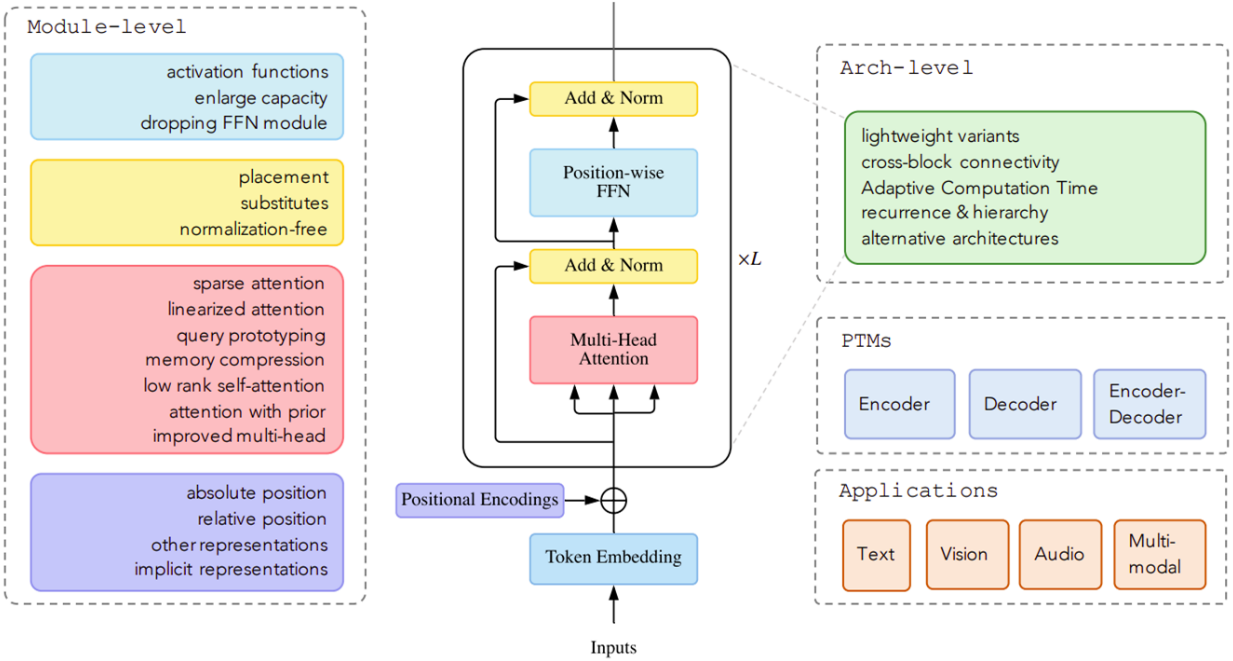
Image source: [1]

Transformer variants

Challenges of Transformer

- Model Efficiency: A key challenge of applying Transformer is its inefficiency at processing long sequences mainly due to the computation and memory complexity of the self-attention module.
- Model Generalization: Since the transformer is a flexible architecture and makes few assumptions on the structural bias of input data, it is hard to train on small-scale data.
- Model Adaptation: This line of work aims to adapt the Transformer to specific downstream tasks and applications.

Categorization of Transformer variants



- Image source: [2]

Attention-level variants of Transformers

- **Sparse Attention:** introduces sparsity bias into the attention mechanism, leading to reduced complexity. (*e.g.*, Longformer, Star-Transformer, Reformer)
- **Linearized Attention:** disentangles the attention matrix with kernel feature maps. The attention is then computed in reversed order to achieve linear complexity. (*e.g.*, Linear Transformer, Performer).
- **Prototype and Memory Compression:** reduces the number of queries or key-value memory pairs to reduce the size of the attention matrix. (*e.g.*, Informer, Linformer)
- **Low-rank Self-Attention:** captures the low-rank property of self-attention. (*e.g.*, Low-rank Attention)
- **Attention with Prior:** explores supplementing or substituting standard attention with prior attention distributions. (*e.g.*, Local Transformer, Realformer)
- **Improved Multi-Head Mechanism:** different alternative multi-head mechanisms. (*e.g.*, Collaborative MHA, Adaptive Attention Span)

Module-level variants of Transformers

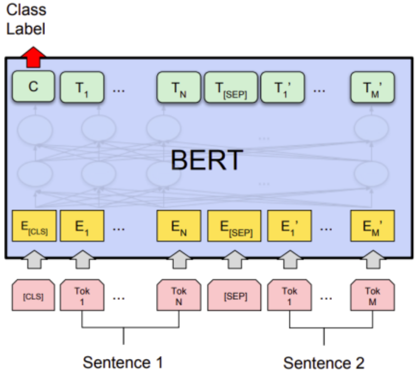
- **Positional representation:** changes the function or let it be trained. (*e.g.*, T5, Transformer-XL)
- **Layer normalisation:** changes placement of layer normalisation, normalisation method. (*e.g.*, post-LN, AdaNorm)
- **FF layer:** explores the case of dropping FF layer, changes activation function, increases parameter. (*e.g.*, Product-key Memory, All-Attention layer)

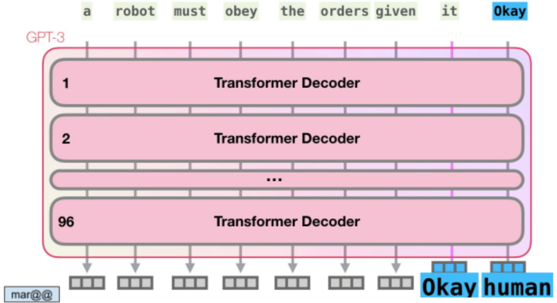
Architecture-level variants of Transformers

- **Lightweight:** module-level or higher-level changes to use it with memory constraints. (*e.g.*, Lite Transformer)
- **Strengthening Cross-Block Connectivity:** reuses attention distributions from previous block to guide attention of current block. (*e.g.*, Realformer, Transformer-XL)
- **Adaptive Computation Time:** applies more computations for data that are hard to process, for easy examples, a shallow representation to reduce computation time. (*e.g.*, Conditional Computation Transformer, DeeBERT)
- **Using Divide-and-Conquer Strategies:** decomposes an input sequence into finer segments that can be efficiently processed by Transformer. (*e.g.*, Transformer-XL, HIBERT)
- **Exploring Alternative Architecture:** (*e.g.*, Sandwich Transformer, Mask Attention Network (MAN))

Pre-training of Transformers

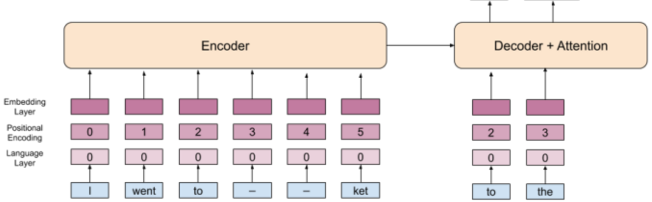
- **Pretraining encoder** (*e.g.*, BERT)





- **Pretraining decoder** (*e.g.*, GPT)

- **Pretraining full model** (*e.g.*, MASS)



References

[1] Vaswani, Ashish, et al. "Attention is all you need." Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017.

[2] Lin, Tianyang, et al. "A survey of transformers." arXiv preprint arXiv:2106.04554. 2021.

[3] Voita, Elena, et al. "Analyzing Multi-Head Self-Attention: Specialized Heads Do the Heavy Lifting, the Rest Can Be Pruned." 57th Annual Meeting of the Association for Computational Linguistics. ACL Anthology, 2019.

[4] Dong, Yihe, Jean-Baptiste Cordonnier, and Andreas Loukas. "Attention is not all you need: Pure attention loses rank doubly exponentially with depth." International Conference on Machine Learning. PMLR, 2021.

Acknowledgement

- Dr. Rudramurthy V, Dr. Sukanta Sen, Jyotsana Khatri.
- http://jalammar.github.io/illustrated-transformer/
- https://kazemnejad.com/blog/transformer_architecture_positional_encoding/
- https://towardsdatascience.com/master-positional-encoding-part-i-63c05d90a0c3

Thank you!

- Why positional embeddings are summed with word embeddings instead of concatenation?
    - To reduce the cost
- Why are both sine and cosine used?
    - They makes it easier to attend to relative positions because of their rotation property. This means sin(w+k)/cos(w+k) can be represented as rotations of sin(w)/cos(w).
    - Given any input PE(pos), the model can create the attention query matrix Q that targets PE(pos+k) by multiplying the PE(pos) with a weight matrix T (the transformation matrix). The weight matrix T, which could be parameters of a single feed-forward layer, can be learned during the training process.