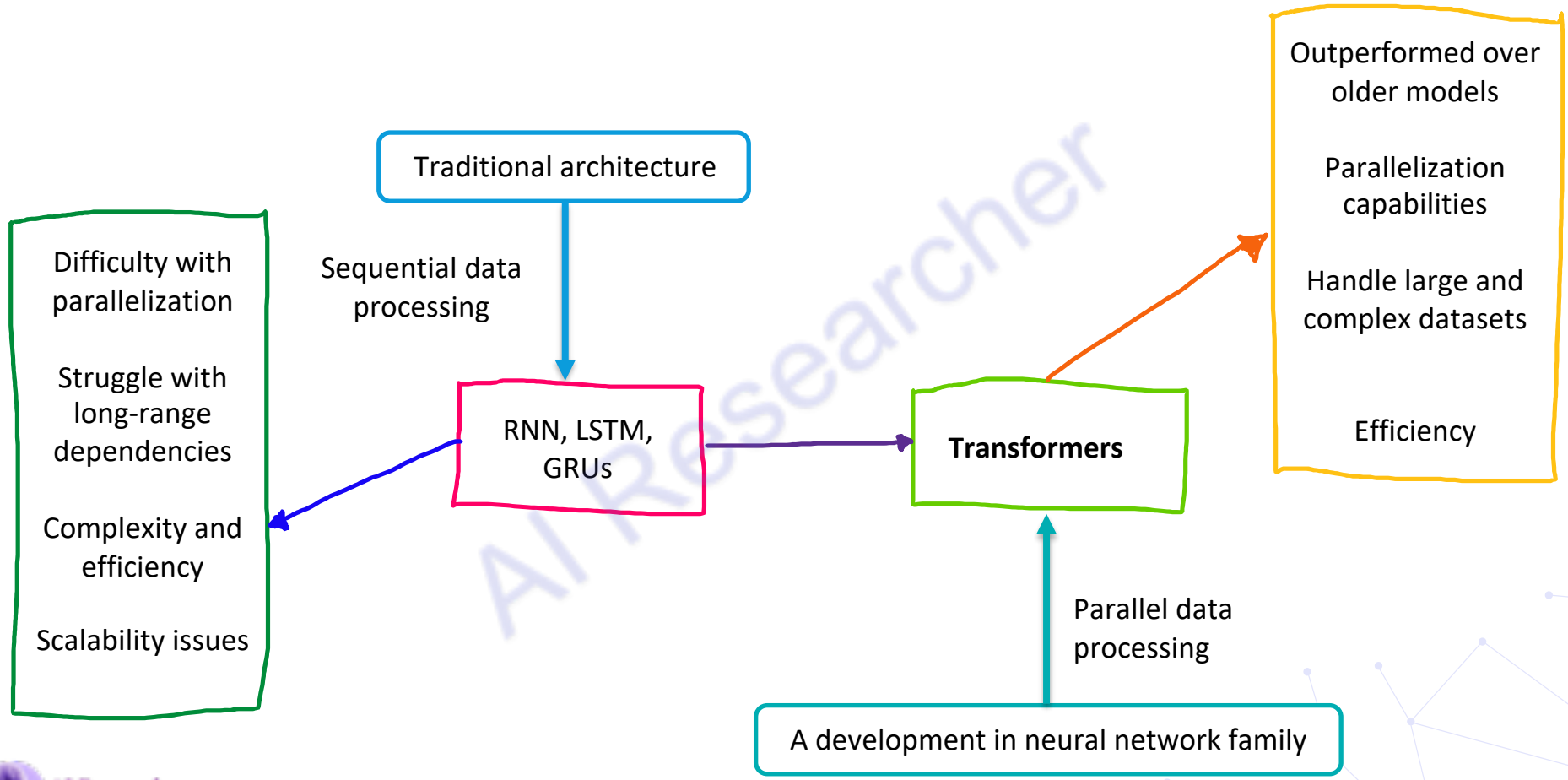
Understanding Transformer Architecture

Attention is all you need – Research study

Introduction to Transformers



Attention Is All You Need

Ashish Vaswani*
Google Brain
avaswani@google.com

Noam Shazeer*
Google Brain
noam@google.com

Niki Parmar*
Google Research
nikip@google.com

Jakob Uszkoreit*
Google Research
usz@google.com

Llion Jones*
Google Research
llion@google.com

Aidan N. Gomez* †
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser*
Google Brain
lukaszkaiser@google.com

Illia Polosukhin* ‡
illia.polosukhin@gmail.com

Architecture Overview

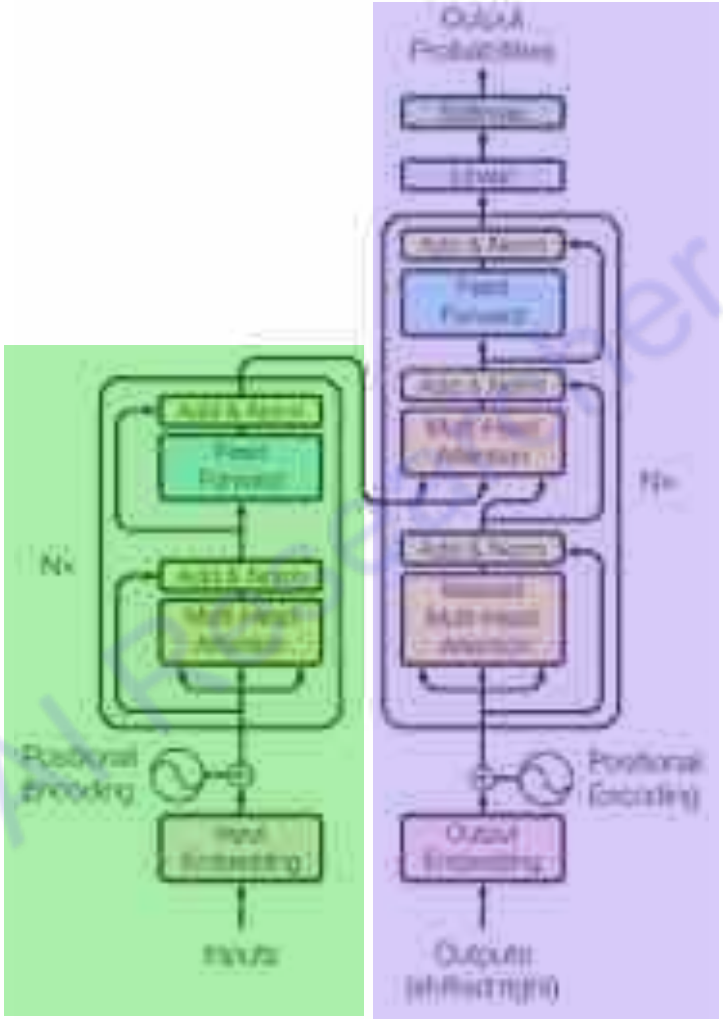


Figure 1: The Transformer - model architecture.

Transformer Core Components

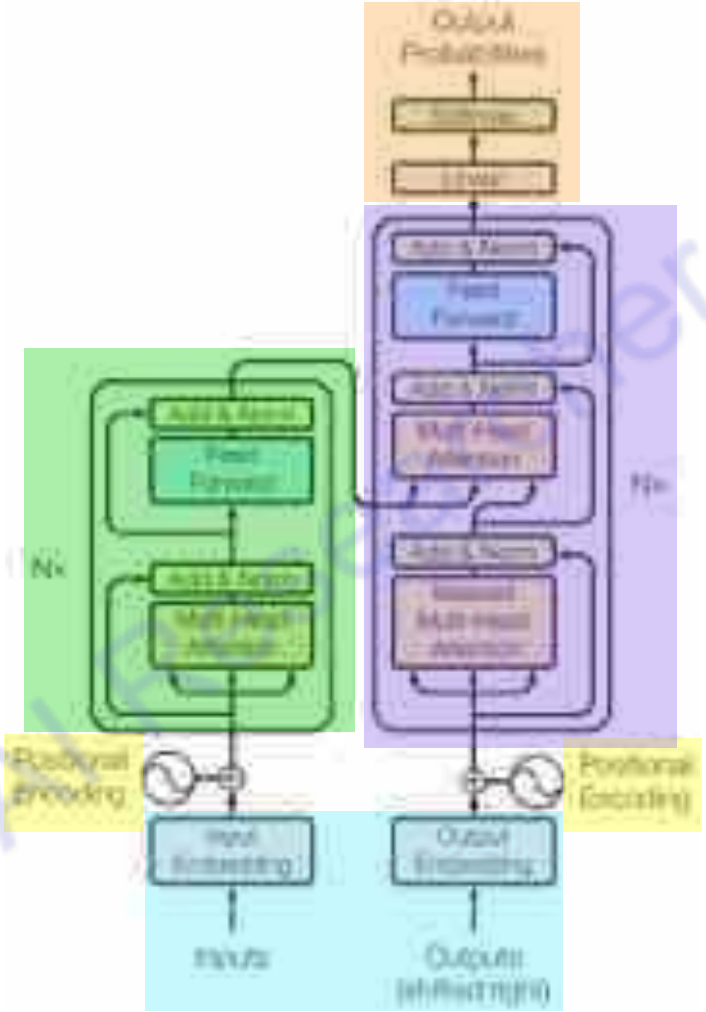


Figure 1: The Transformer - model architecture.

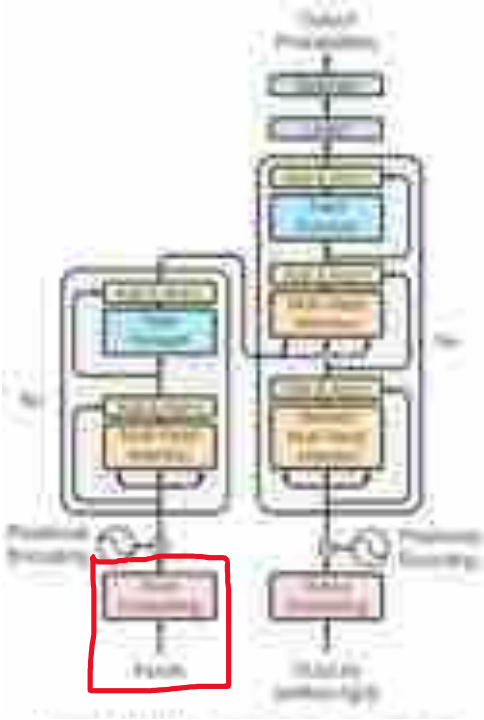
Input Embedding

Converts input tokens/words into vectors…

Example: sunset over the ocean

Embedding matrix

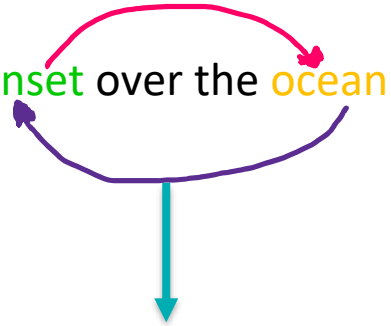
sunset	[0.25, -0.75, 0.50, ..., 0.10]
over	[0.10, -0.70, 0.60, …, -0.40]
the	[0.30, 0.20, -0.10, …, 0.50]
ocean	[-0.60, 0.40, 0.70, …, -0.20]



Positional Encoding

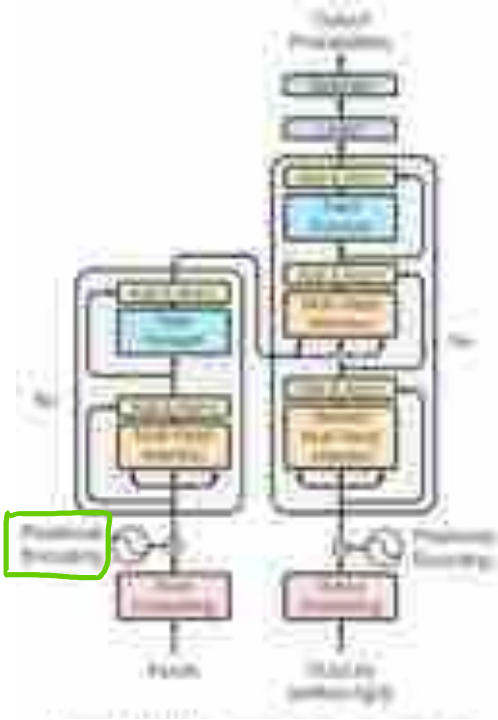
Adds information about the position of each word in the sequence to the input embeddings…

Example: sunset over the ocean



ocean over the sunset

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

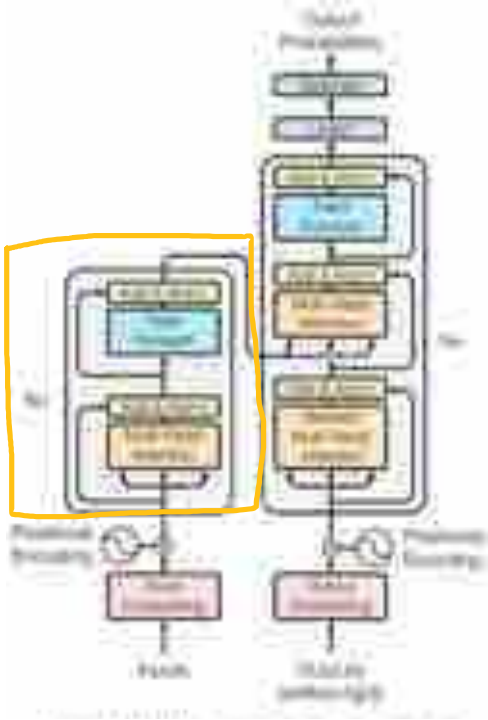


Embedding matrix with position

sunset	[0.25, -0.75, 0.50, ..., 0.10, 1]
over	[0.10, -0.70, 0.60, ..., -0.40, 2]
the	[0.30, 0.20, -0.10, ..., 0.50, 3]
ocean	[-0.60, 0.40, 0.70, ..., -0.20, 4]

Encoder Layers

- It processes the input sentence
- It has multiple identical layers (Nx)
- Each layer has two sub-layers
- The first sub-layer is the multi-head self-attention mechanism
- The second is a position-wise fully connected feed-forward network



Multi-Head Attention

Allows the model to focus on different positions of the input sequence, helping it to draw global dependencies between words…

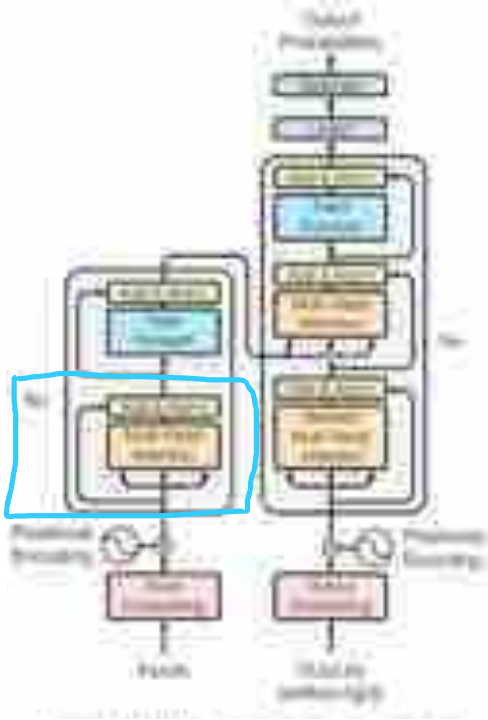
Example: sunset over the ocean

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

Three vectors

Attention score

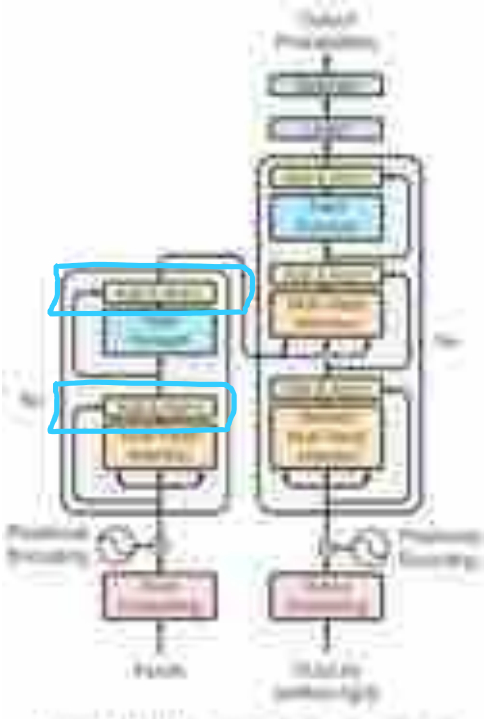
Dimension of the key vectors



$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$
$$\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

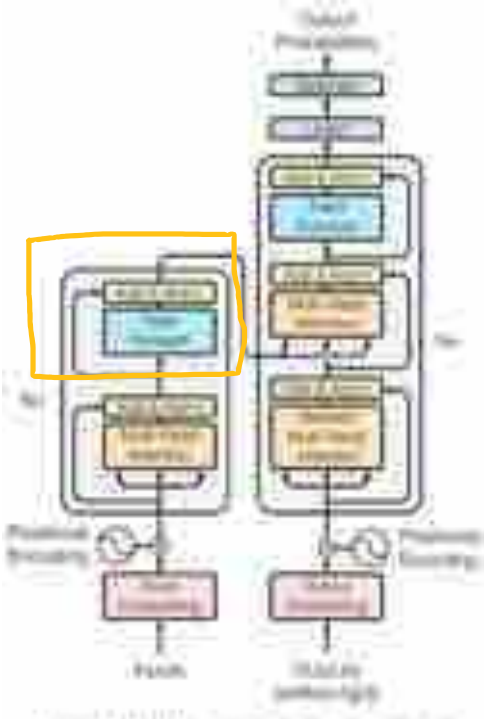
Add and Norm

- Each attention output is then added to the original input vector
- Normalization is applied to stabilize the learning process.



Point-Wise Feed Forward Networks

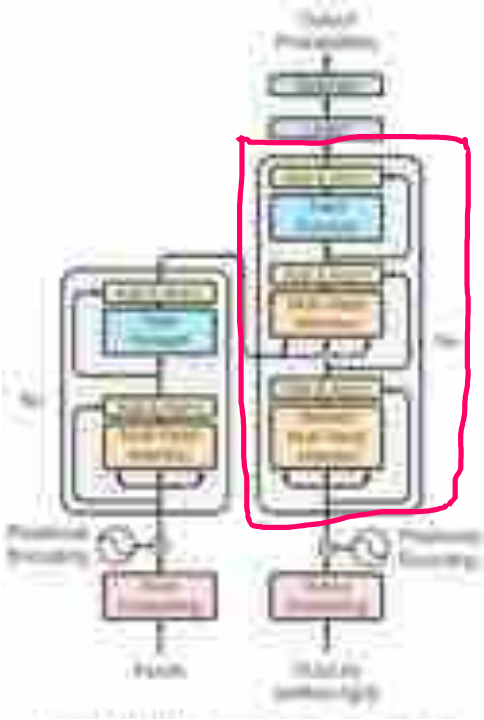
- Allows the Transformer to learn even more about the relationship between words in the sentence…
- Enriched the final word representations with more complex patterns…
- Example: sunset over the ocean



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

Wight matrix | Bias | Wight matrix | Bias

Decoder Layers



It has a similar structure to the encoder, with the addition of a third sub-layer that performs multi-head attention..

Produces output text one token at a time, using both its own output up to that point and the output from the encoder…

Example: sunset over the ocean

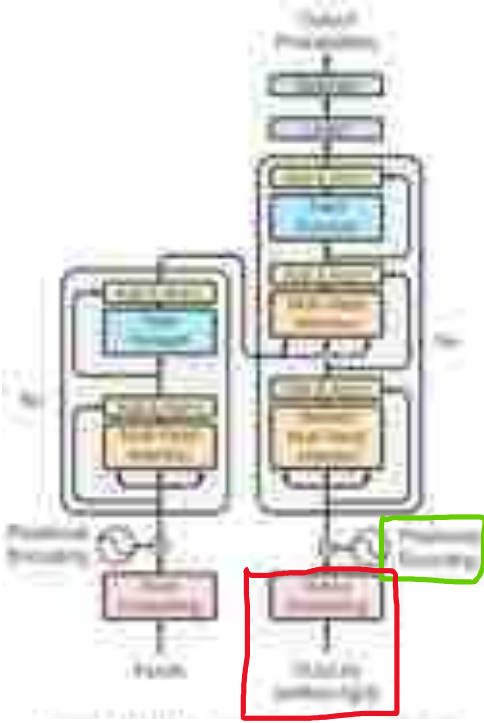
Spanish translation: Puesta de sol sobre el océano

Output Embedding and Positional Encoding

Similar to the input embedding, but for the output sequence...

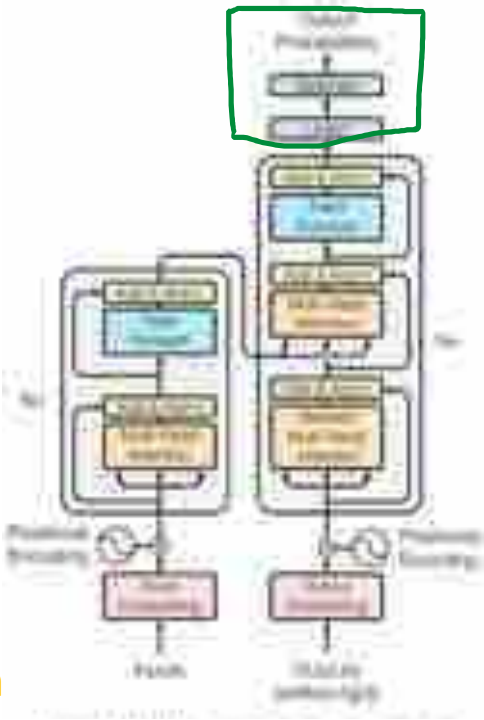
Example: sunset over the ocean

puesta del sol



Final Layers

- Linear Layer maps the decoder's output to a score for each word in the vocabulary.
- Softmax layer then converts these scores to probabilities, and the word with the highest probability is chosen as the translation.



Example: sunset over the ocean

Output Probabilities predicts for the next token in the sequence.

puesta del sol → sobre

Next token

Example

- Machine Translation: Transforming a sentence from one language to another while maintaining the semantic context.
- Text Summarization: Generating a concise summary from a long text.
- Named Entity Recognition: Identifying names, organizations, locations in text.

Thank you!