AI in Built Environment
DCP4300

Week 12: Natural Language Processing

Transformers

Dr. Chaofeng Wang
Jianhao Gao (TA)

University of Florida
College of Design Construction and Planning

Word embeddings

Word2vec, 2013
Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.
https://arxiv.org/abs/1301.3781

GloVe, 2014
Pennington, J., Socher, R., & Manning, C. D. (2014, October). Glove: Global vectors for word representation. In Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP) (pp. 1532-1543).
https://aclanthology.org/D14-1162.pdf

RNNs

LSTM, 2000
Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with LSTM. Neural computation, 12(10), 2451-2471.
https://ieeexplore.ieee.org/abstract/document/6789445

GRU, 2014
Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555. https://arxiv.org/abs/1412.3555

Transformers (BERT, GPT)

2017/2018 -

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). **Attention is all you need**. Advances in neural information processing systems, 30. https://arxiv.org/abs/1706.03762

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
https://arxiv.org/abs/1810.04805

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.
https://arxiv.org/abs/2005.14165

Self-attention layer:
Help encoder look at other words in the input sentences as it encodes a specific word.

Language models

ChatGPT Decoder:
predict the next word in a sentence or complete a partial sentence

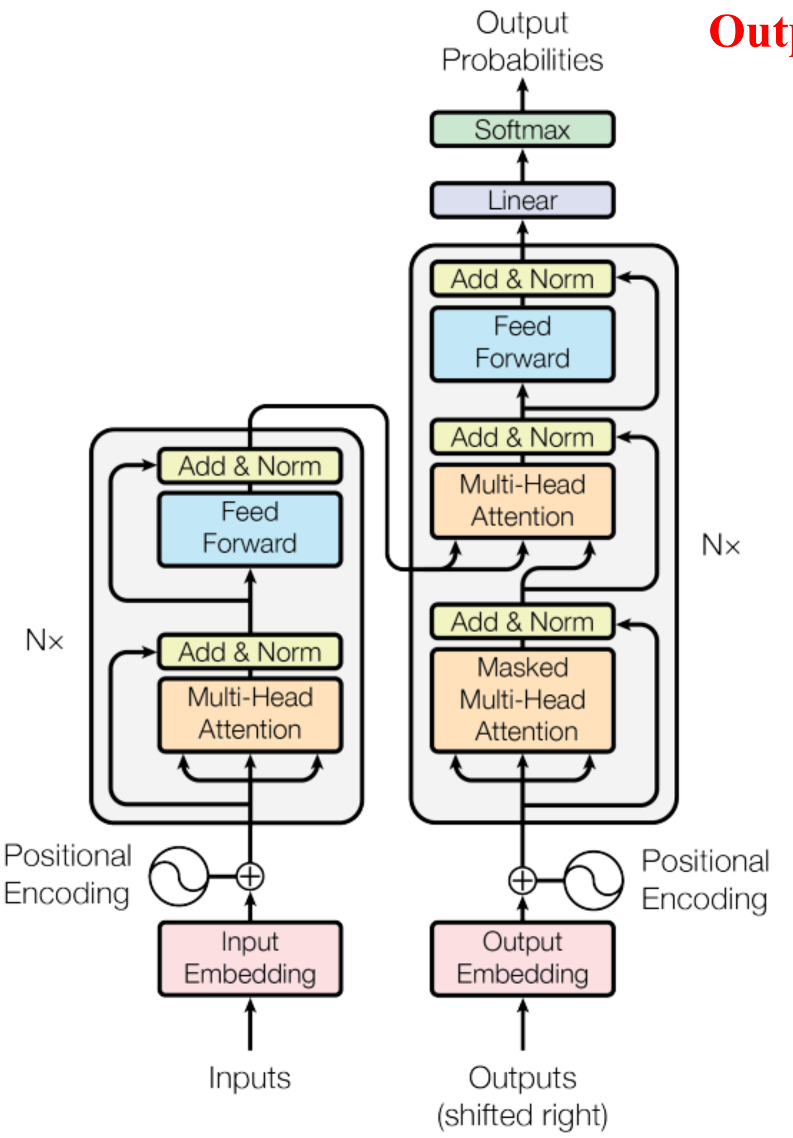
$$P(w_1\, w_2 \ldots w_n) = P(w_n \mid w_1 \ldots w_{n-1}) * P(w_{n-1} \mid w_1 \ldots w_{n-2}) \ldots P(w_2 \mid w_1) * P(w_1)$$

P(The quick brown fox jumps over the lazy dog) =

P(The) * P(quick | The) * P(brown | The quick) * P(fox | The quick brown) * P(jumps | The quick brown fox) * P(over | The quick brown fox jumps) * P(the | The quick brown fox jumps over) * P(lazy | The quick brown fox jumps over the) * P(dog | The quick brown fox jumps over the lazy)

predict the next word

Output: [begin] How

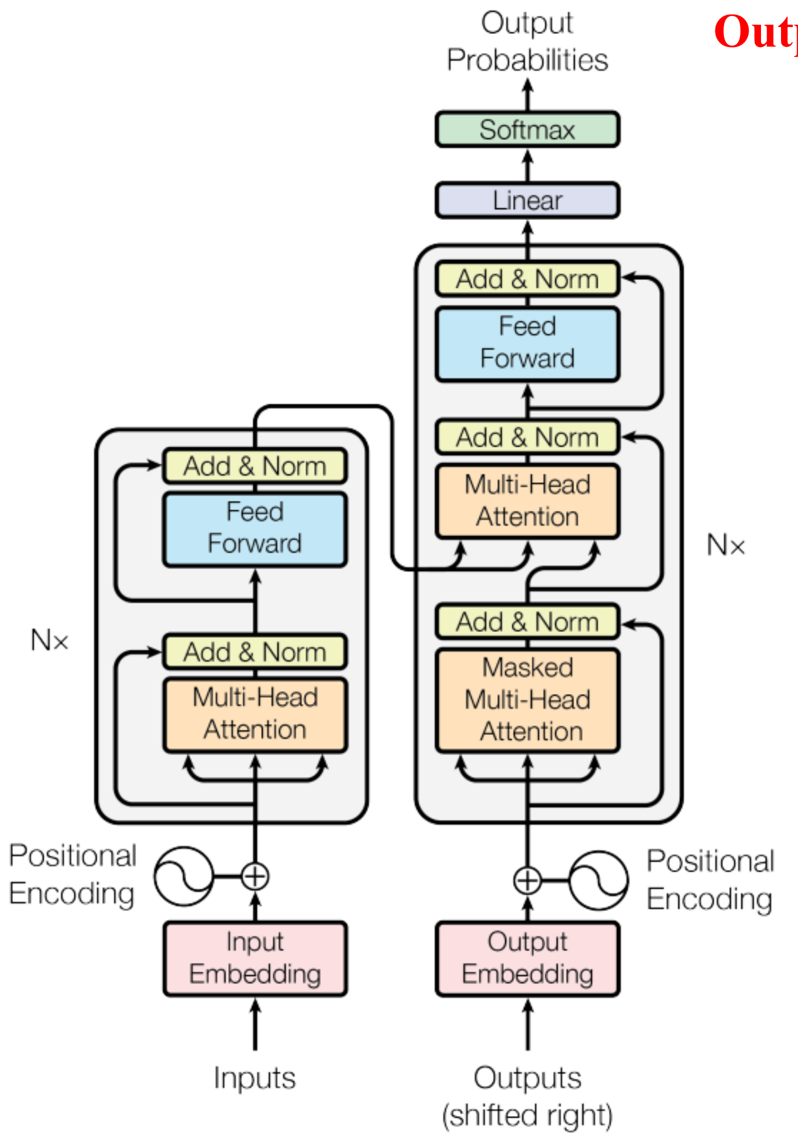


Input: Give me a random sentence

Output: [begin]

predict the next word

Output: [begin] How may

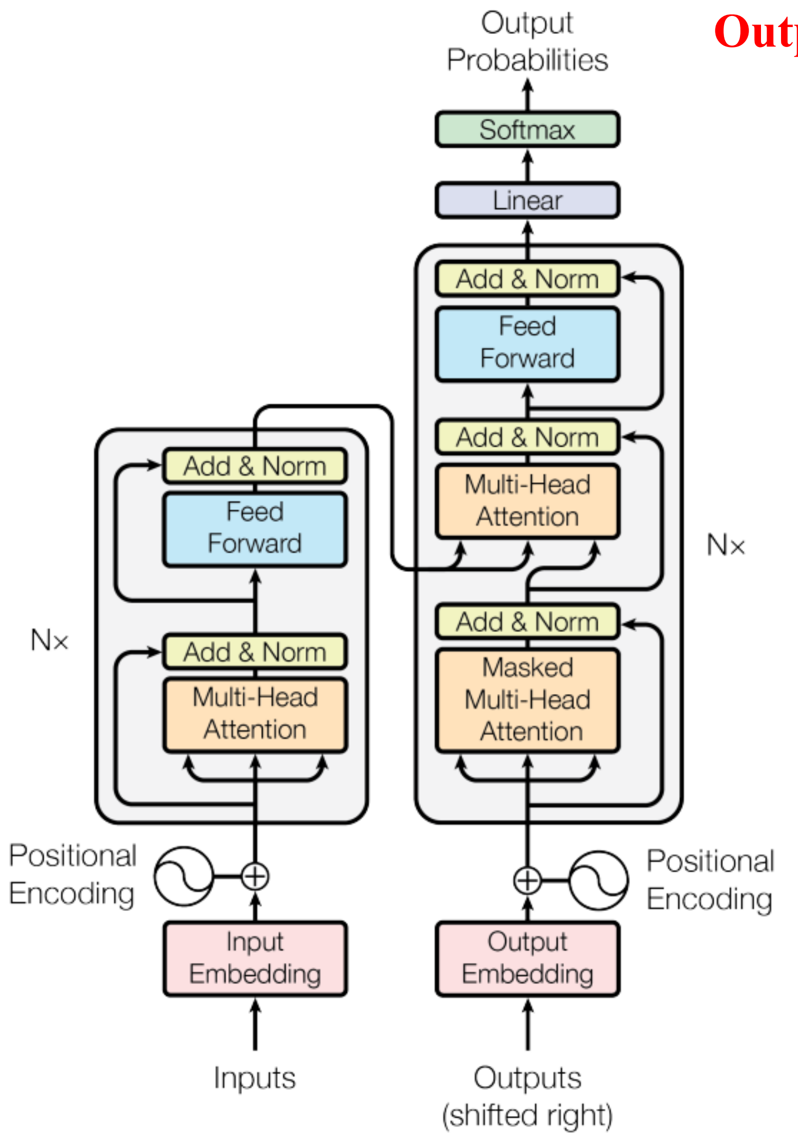


Input: Give me a random sentence

Output: [begin] How

predict the next word

Output: [begin] How may I

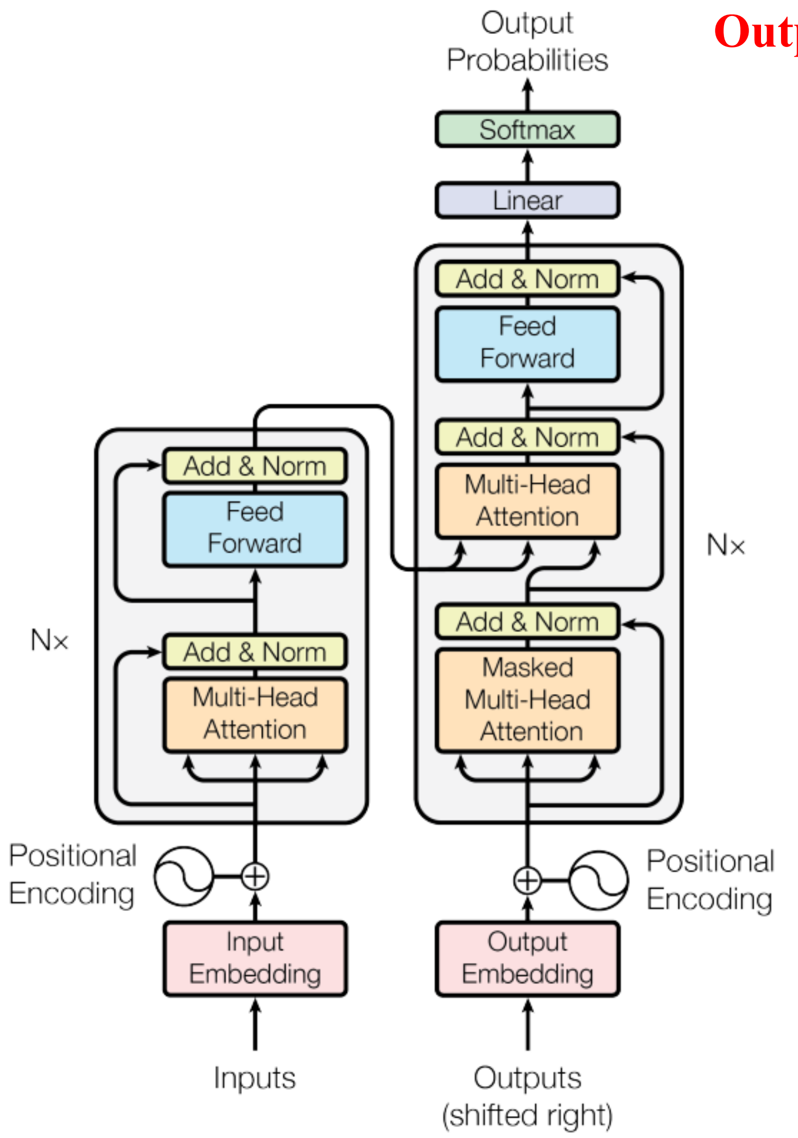


Input: Give me a random sentence

Output: [begin] How may

predict the next word

Output: [begin] How may I help

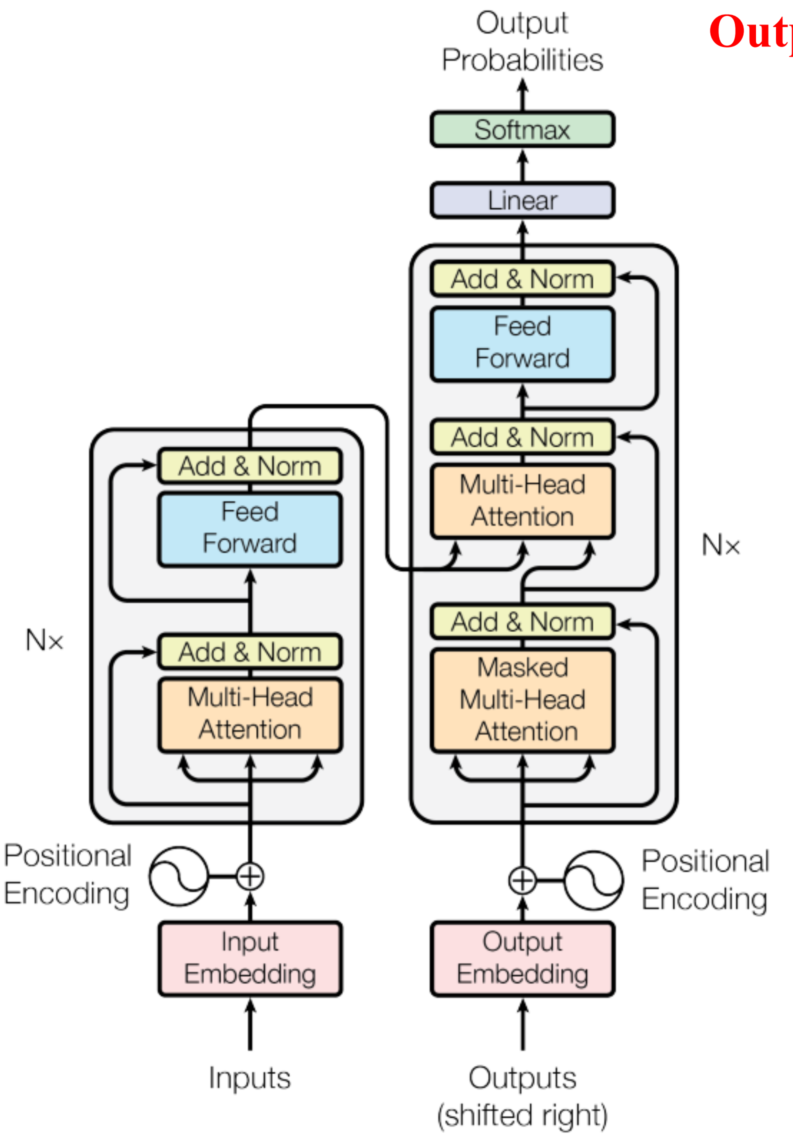


Input: Give me a random sentence

Output: [begin] How may I

predict the next word

Output: [begin] How may I help you

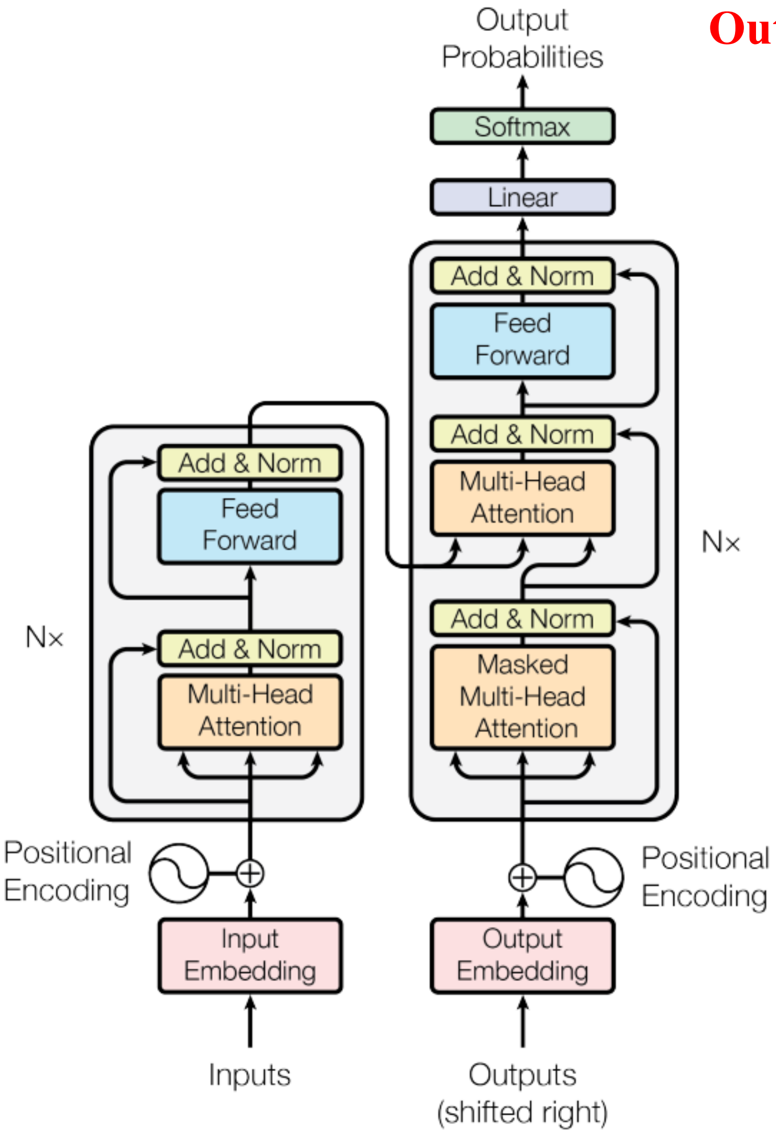


Input: Give me a random sentence

Output: [begin] How may I help

predict the next word

Output: [begin] How may I help you?

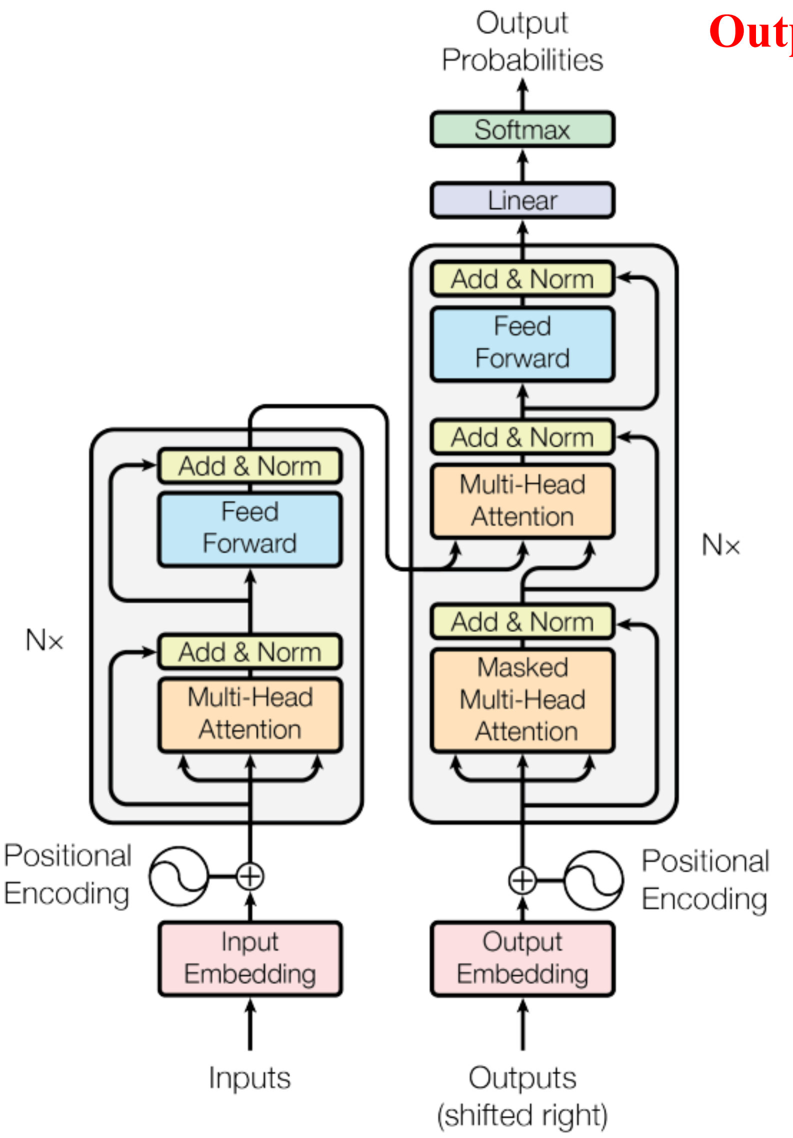


Input: Give me a random sentence

Output: [begin] How may I help you

predict the next word

Output: [begin] How may I help you? [end]



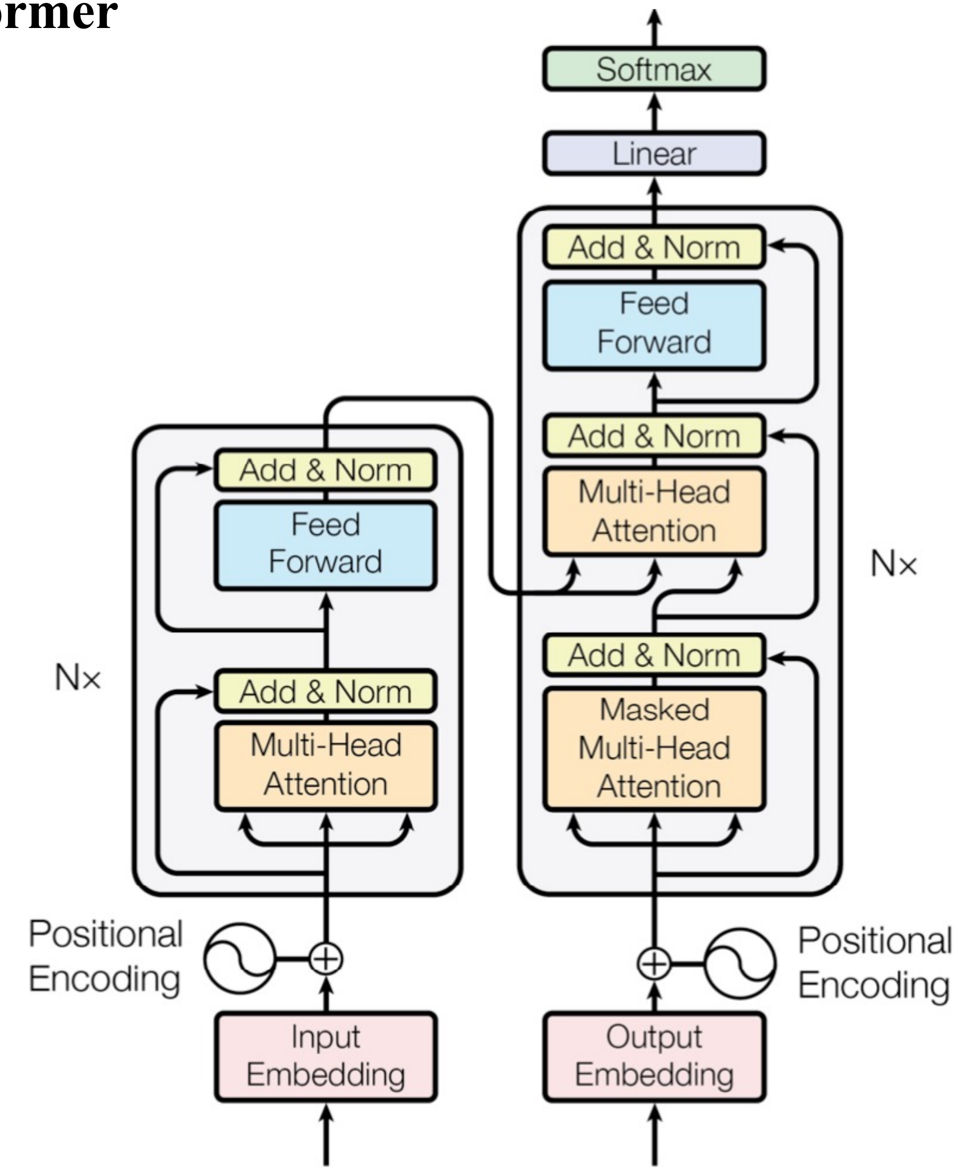
Input: Give me a random sentence

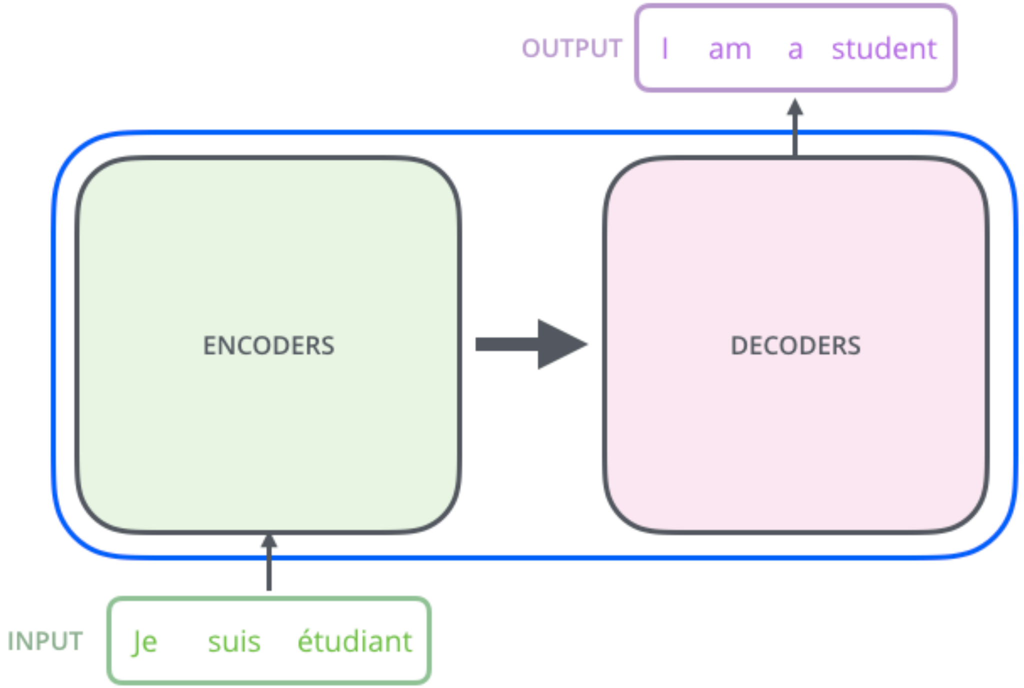
Output: [begin] How may I help you?

- A good video to illustrate the structure:

Illustrated Guide to Transformers Neural Network: A step by step explanation https://www.youtube.com/watch?v=4Bdc55j80l8

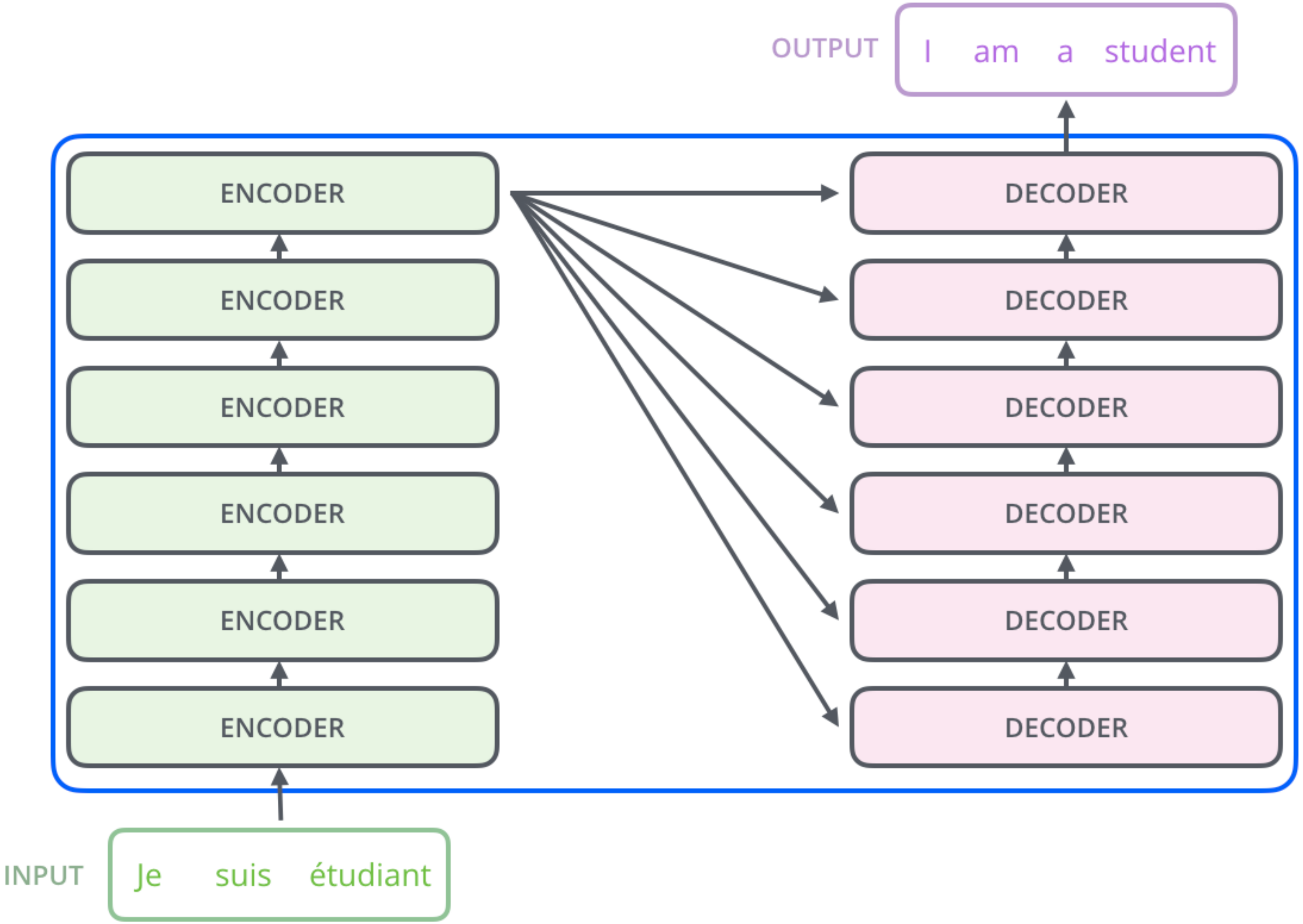
Transformer





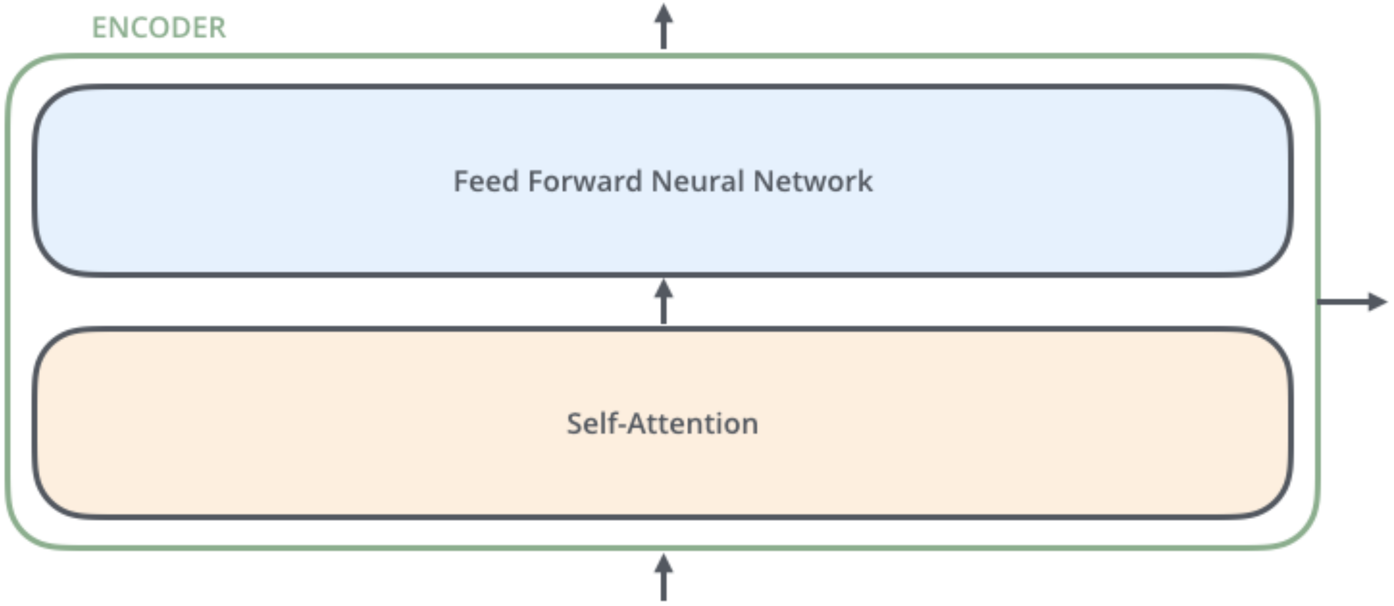
Transformer

Do not share weights.



Transformer

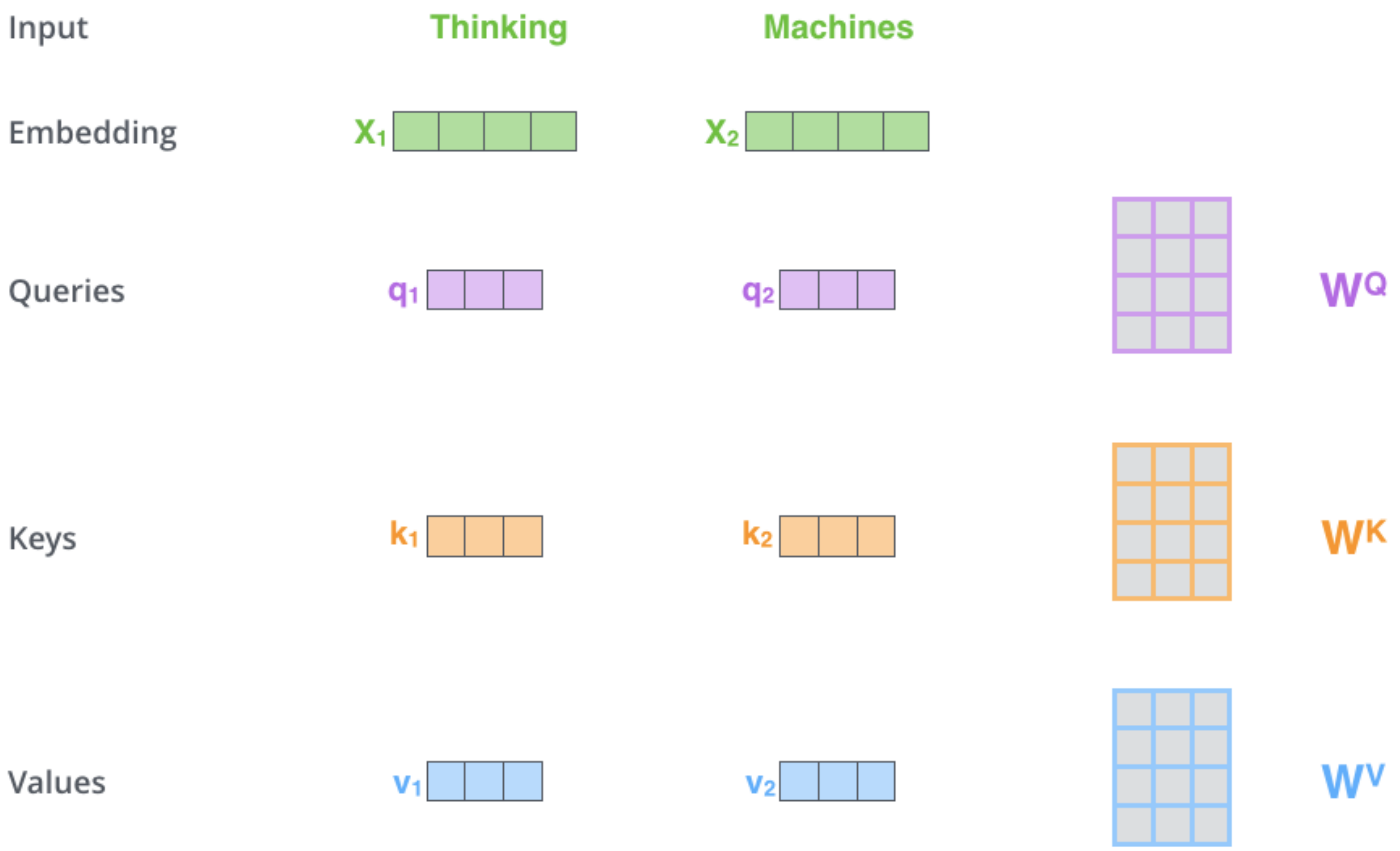
Take a closer look into an encoder.



Self-attention layer: A layer that helps the encoder look at other words in the input sentence as it encodes a specific word

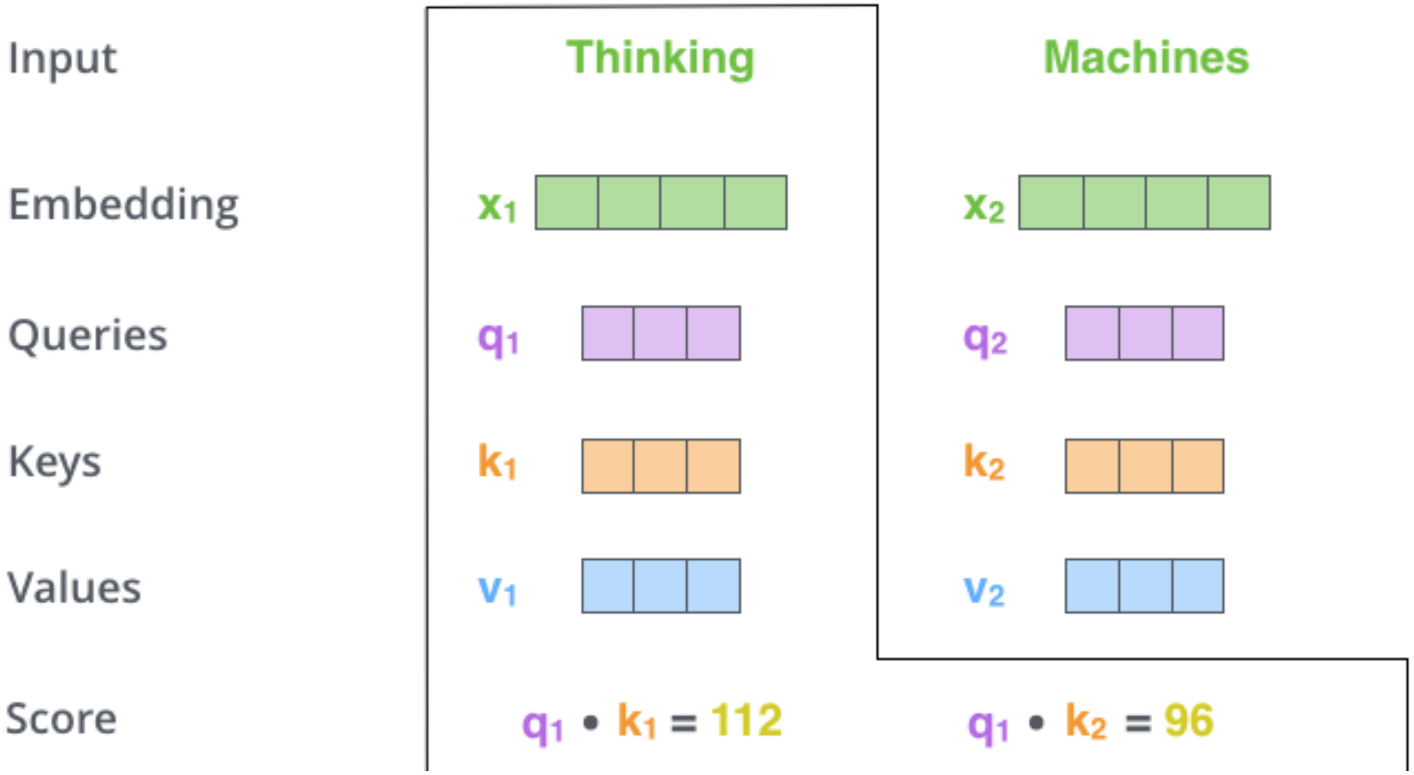
Transformer

Self-attention



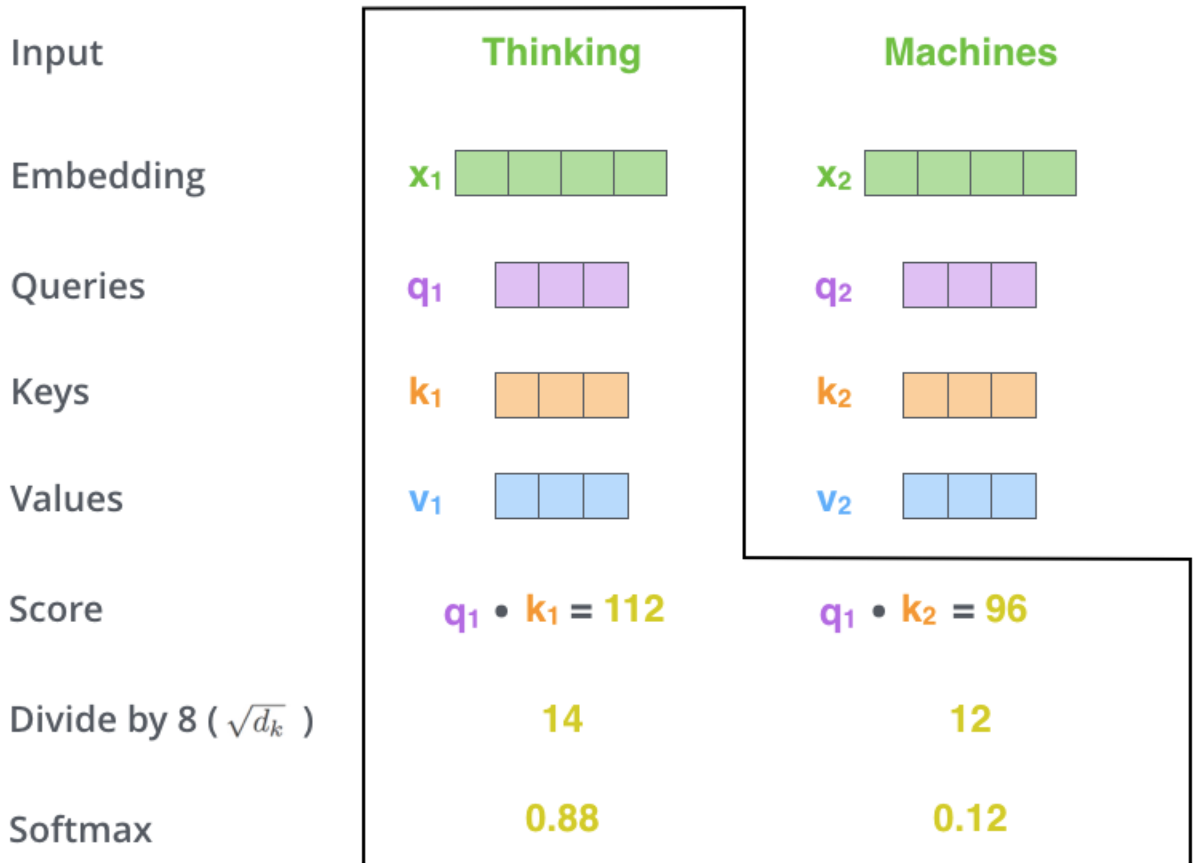
Transformer

Self-attention



Transformer

Self-attention



Transformer

Self-attention

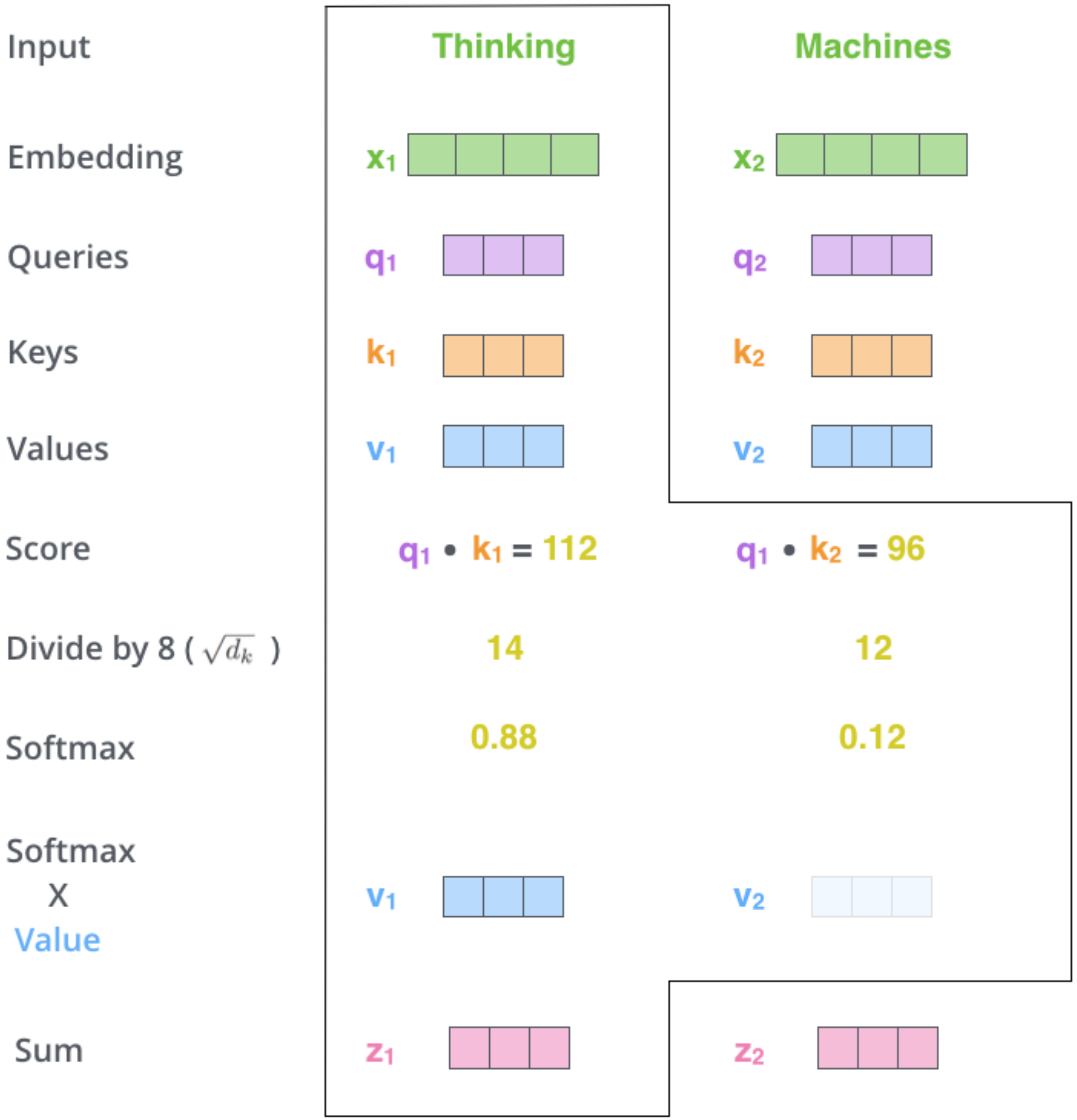
I saw Anchorage flying back to China.

I saw Anchorage flying back to China.

Attention vector used to encode/embed/understand the word ‘flying’ , in the context of this sentence.

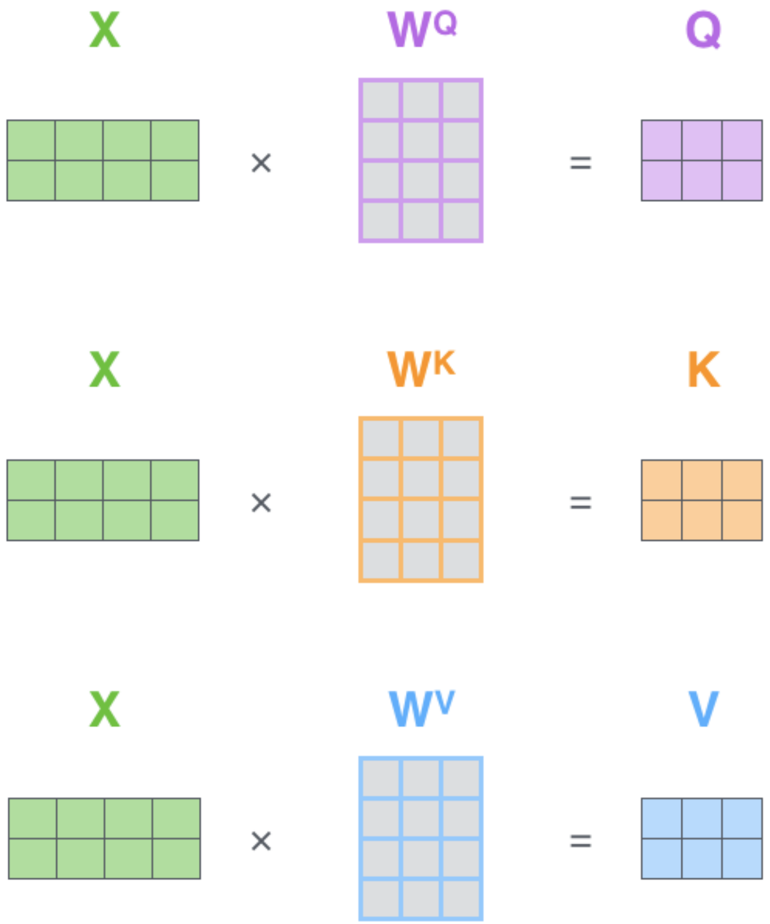
Transformer

Self-attention



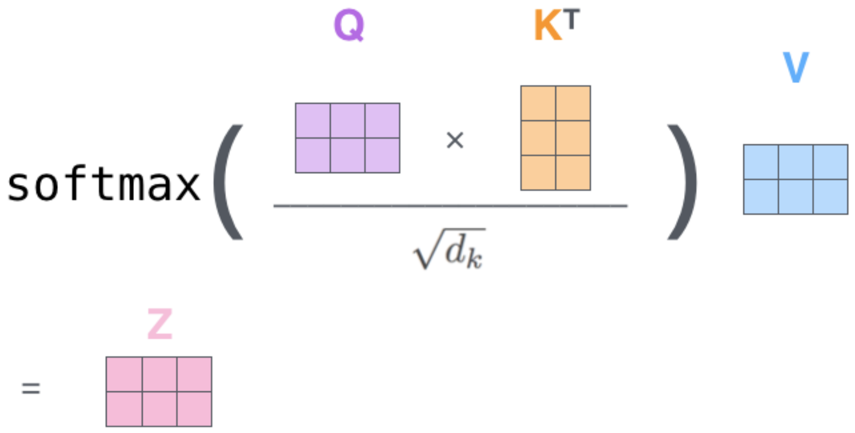
Transformer

Self-attention
Matrix Calculation

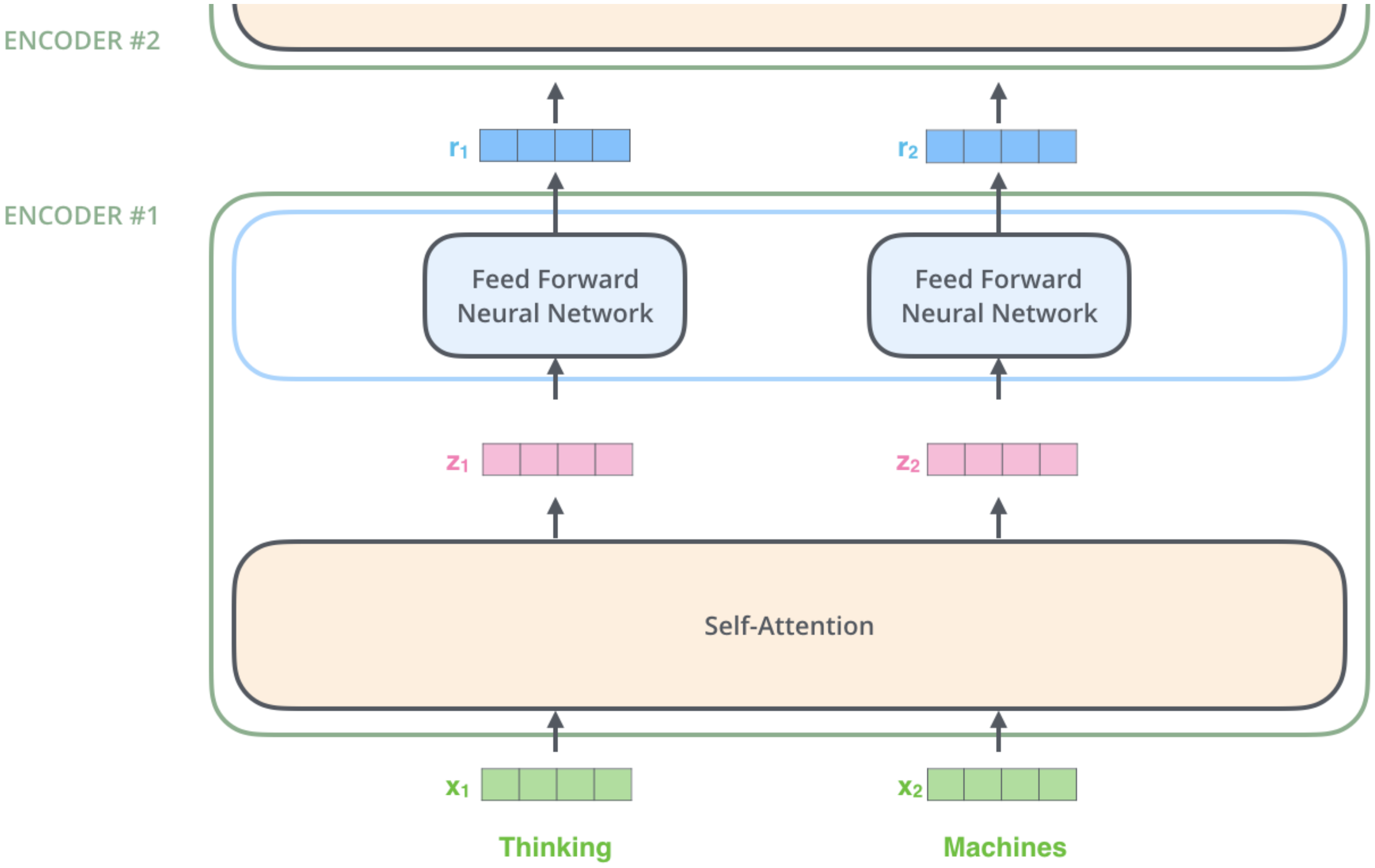


Transformer

Self-attention
Matrix Calculation

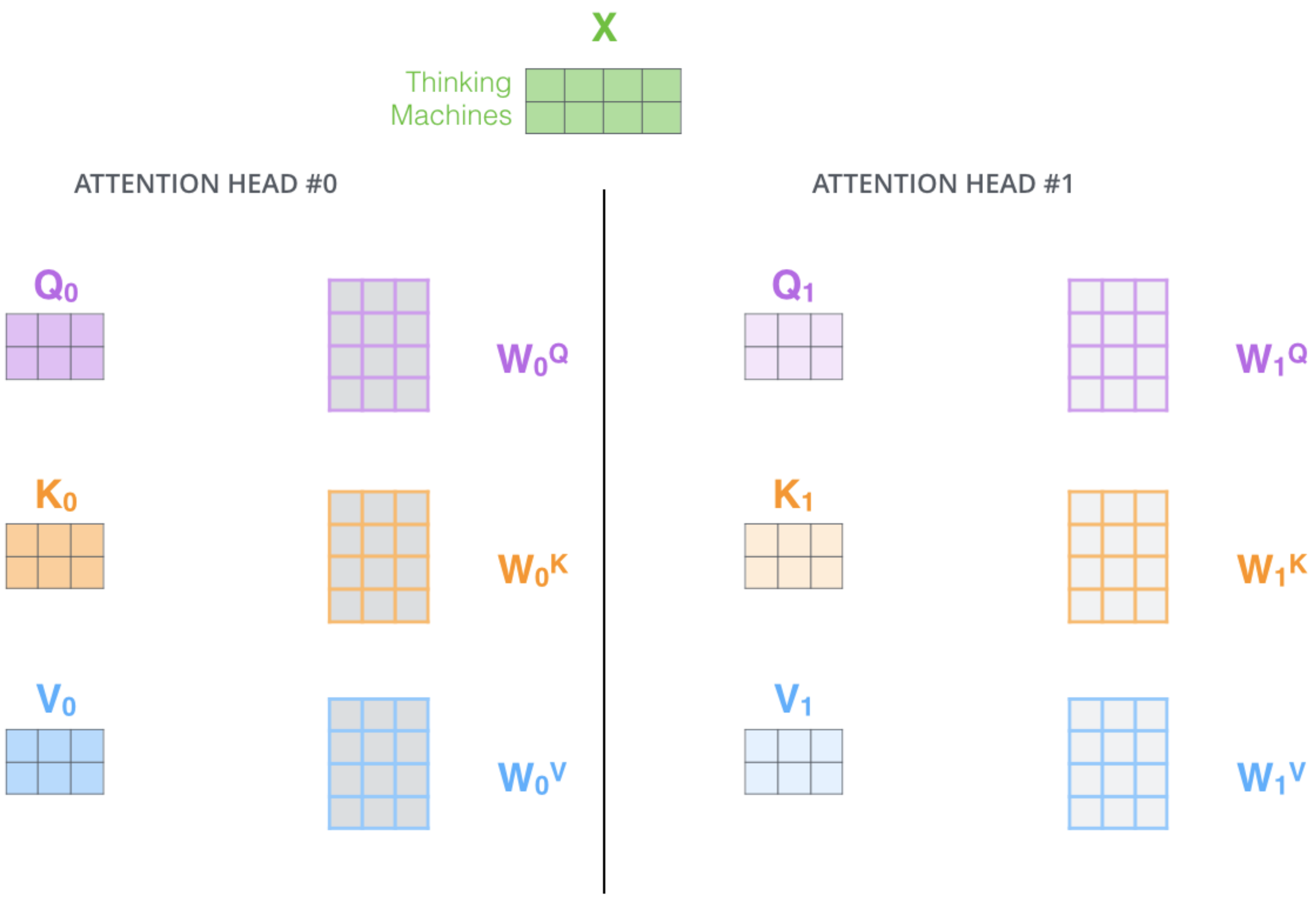


Transformer



Transformer

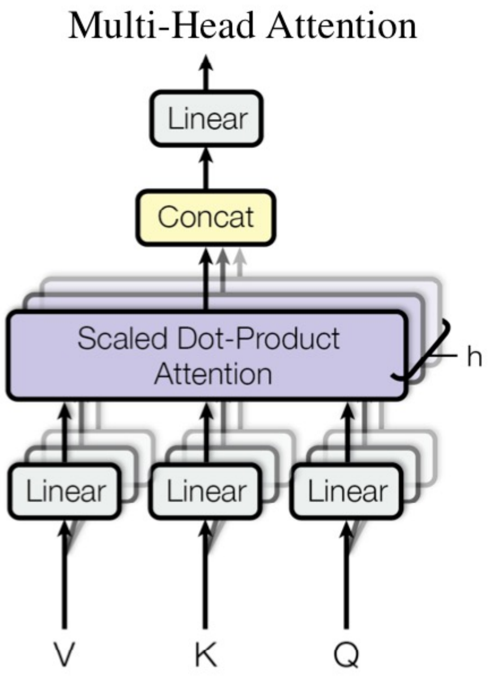
Multi-head attention



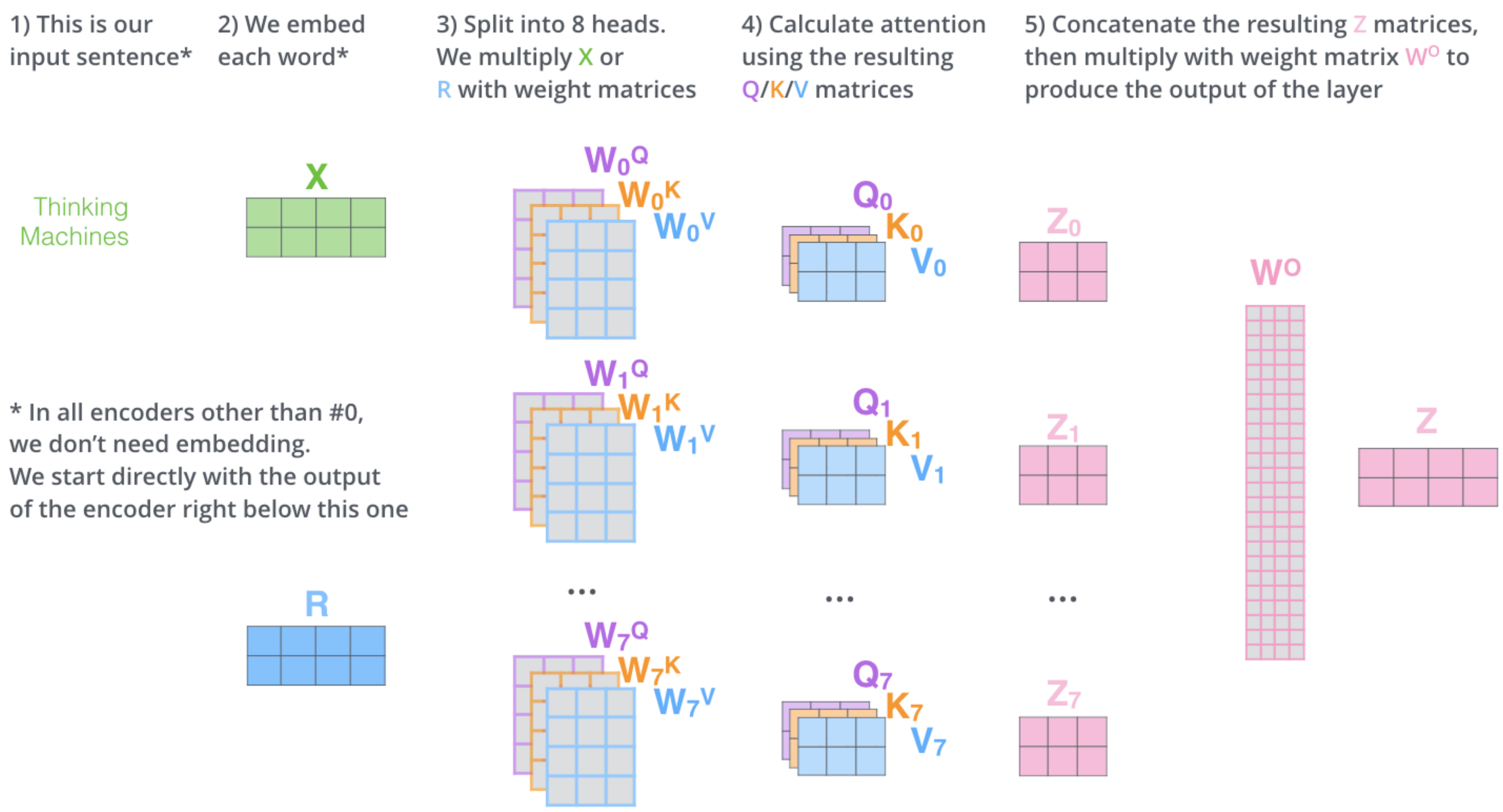
Each head can project the input into a **different representation subspace**

Transformer

Multi-head attention

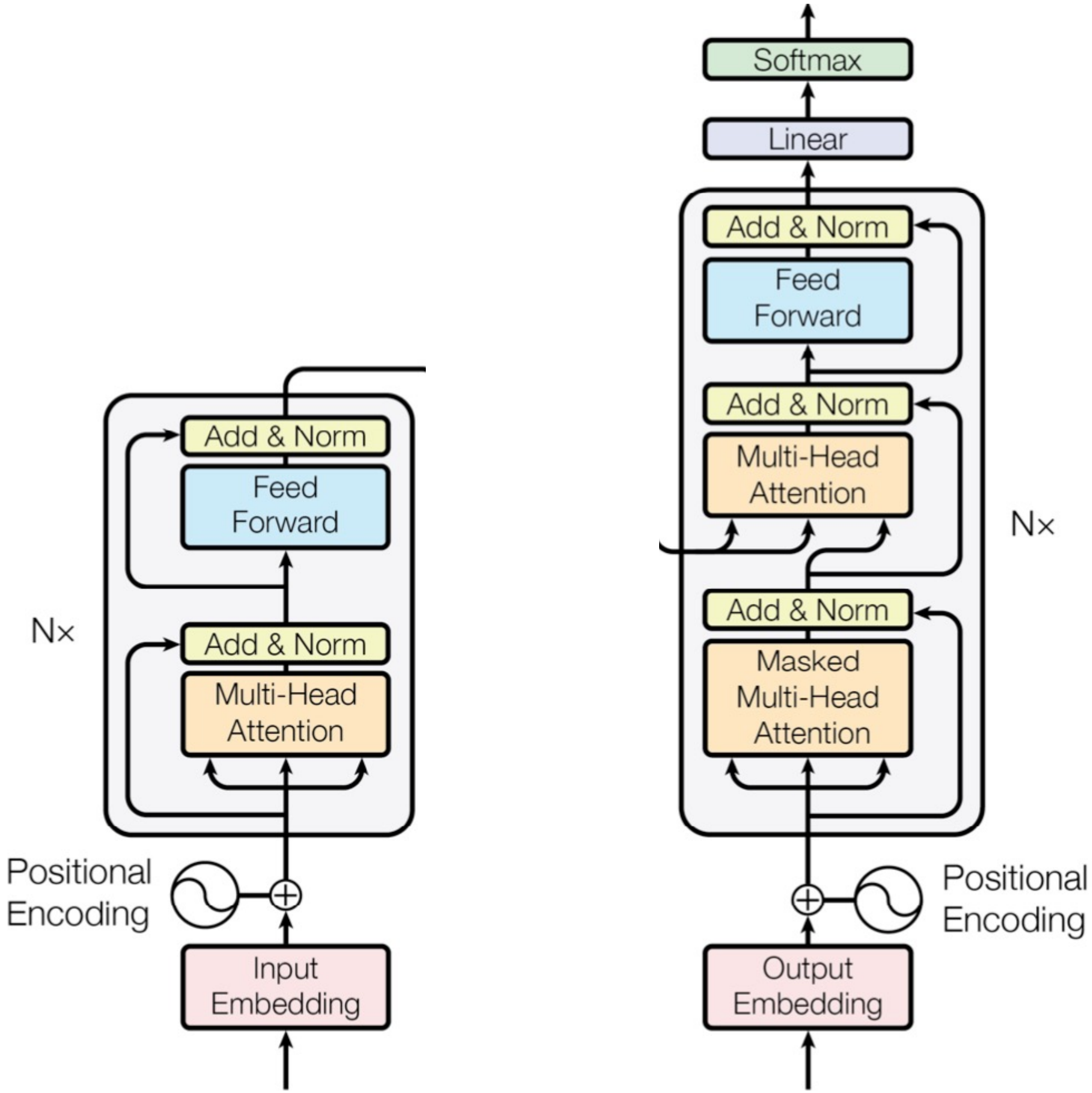


Detail->

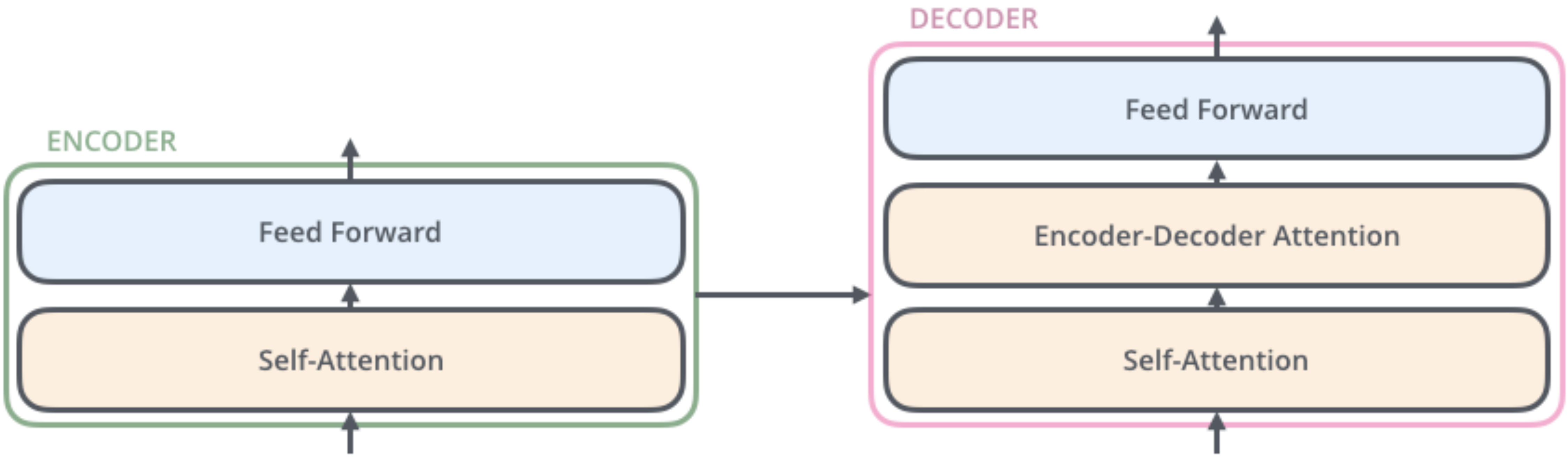


Transformer

Decoder



Transformer



Encoder-Decoder Attention layer: An attention layer that helps the decoder focus on relevant parts of the input sentence

Transformer

Decoder