

# Advanced Topics in Deep Learning

Summer Semester 2024

2. Attention and Transformers

17.04.2023

Prof. Dr. Vasileios Belagiannis

Chair of Multimedia Communications and Signal Processing

# Course Topics

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

1. Interpretability.
2. **Attention and Transformers.**
3. Self-supervised Learning.
4. Similarity Learning.
5. Generative Models.
6. Model Compression.
7. Transfer learning, domain adaptation, few-shot learning.
8. Uncertainty Estimation.
9. Geometric Deep Learning.
10. Recap and Q&A.
  - The exam will be written.
  - We will have an exam preparation test.

## Acknowledgements

- Special thanks Arij Bouazizi, Julia Hornauer, Julian Wiederer, Adrian Holzbock and Youssef Dawoud for contributing to the lecture preparation.

# Today's Agenda and Objectives

---

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

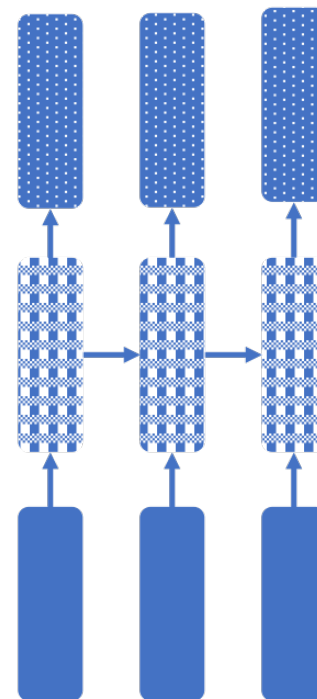
- Sequence Modelling
- Attention
- Self-attention
- Multi-head attention
- Encoding-Decoding

# Language Modelling

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Natural language processing (NLP) is a field where sequence modelling the de facto approach.
- Recurrent neural networks (RNNs) are a common approach to modelling sentences and text in general.
- However, they face some limitations due to the challenges of modelling sequences. In sequence modelling, we deal with long and short dependencies, e.g. according to the sentence length.
- In addition, the input and output sequences are of dynamic length.

Ich bin Vasilis

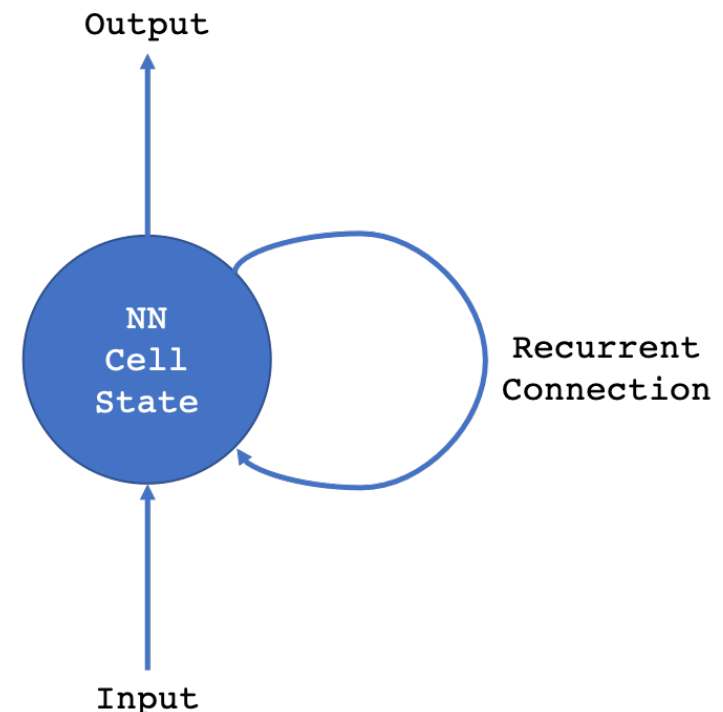


I am Vasilis

# Language Modelling (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- RNNs used to be the standard approach to language modelling.
  - LSTMs, GRUs and other related approaches have been proposed to address the challenges of long sentences. However, they have failed because of the number of words contained in a long sentence.
  - Inspired by humans, a focus mechanism was necessary for focusing on the relevant words. That was the motivation for introducing attention to RNN<sup>\*,\*\*</sup> language modelling.



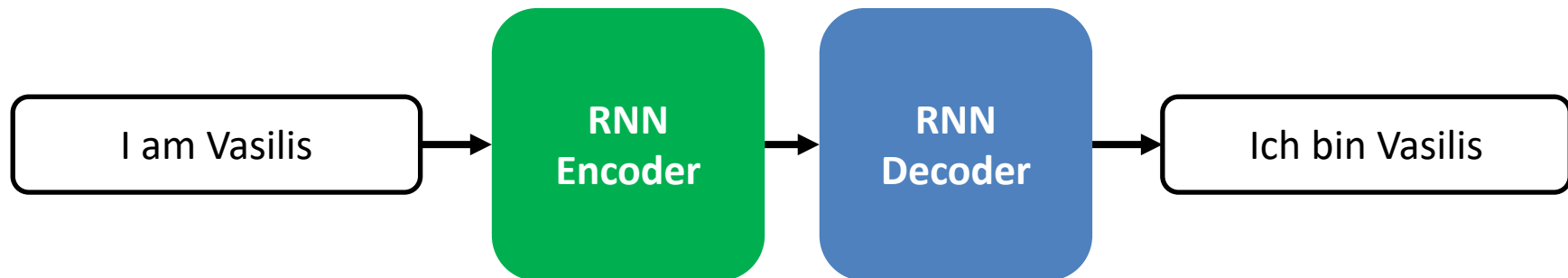
\*Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems 27 (2014).

\*\*Cho, Kyunghyun, et al. "Learning phrase representations using RNN encoder-decoder for statistical machine translation." , Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing ({EMNLP}).

# Sequence Modelling (Language)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Consider the language modelling example, e.g. text translation, and the RNN modelling approach.
- We need an encoder to model the input (English sentence) and the decoder to model the output (German translation).



- Note that the input word is first transformed to an embedding. Each input (word) will be converted to a vector using approaches like GloVe\* or fastText\*\*.

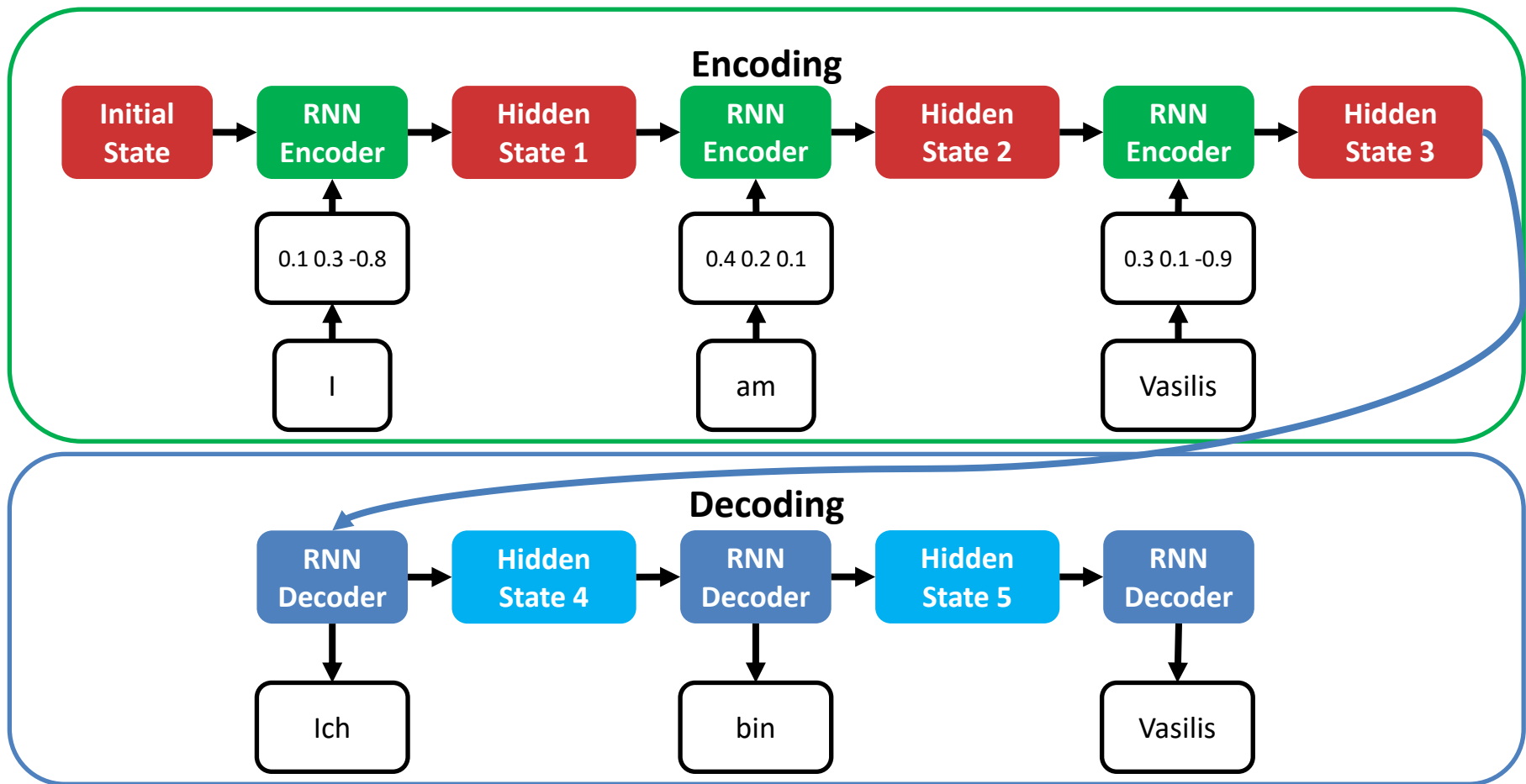
\*Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. "Glove: Global vectors for word representation." Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). 2014.

\*\*Bojanowski, Piotr, et al. "Enriching word vectors with subword information." Transactions of the association for computational linguistics 5 (2017): 135-146.

# Language Modelling

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

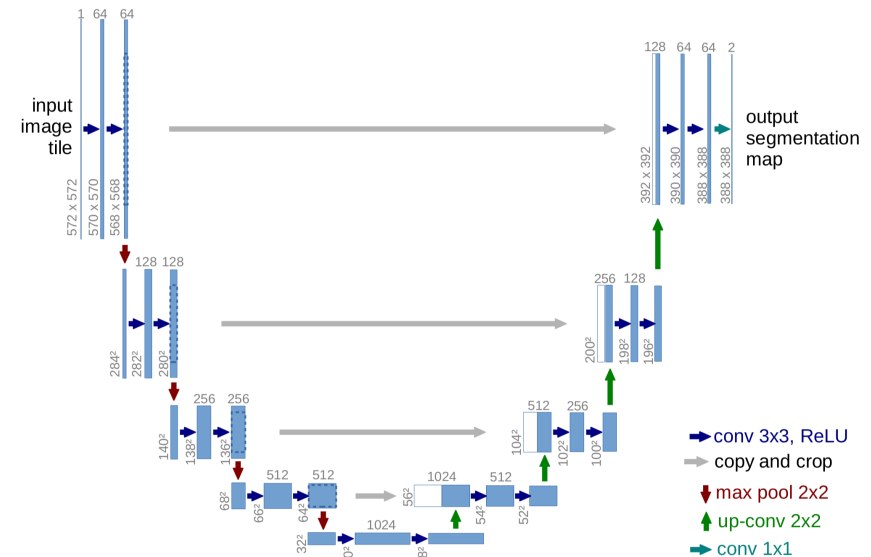
- The output of the decoder summarises the context.



# Language Modelling (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Based only on the context information, it is challenging to model long sentences.
- Recall the encoder-decoder models for vision, e.g. semantic segmentation. To address this limitation, they make use of intermediate connections from the encoder to the decoder.
- One motivation behind the attention mechanism is to provide more information from the encoder to the decoder.



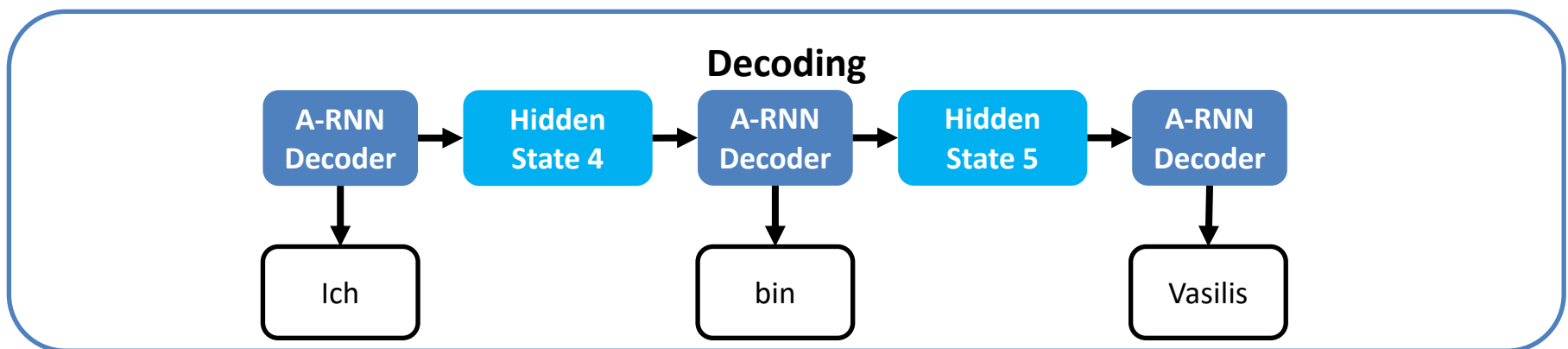
Source: <https://arxiv.org/pdf/1505.04597.pdf>



# Language Modelling (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Now consider passing all the hidden states instead of just the last one. Then use them during each decoding step.
- In this way, additional information is provided to the decoder.
- The attention decoder (A-RNN below) now has a weighting mechanism to decide how much information to withhold from the hidden states.
- The weighting mechanism indicates how much to attend to the hidden state.



# Attention Decoder

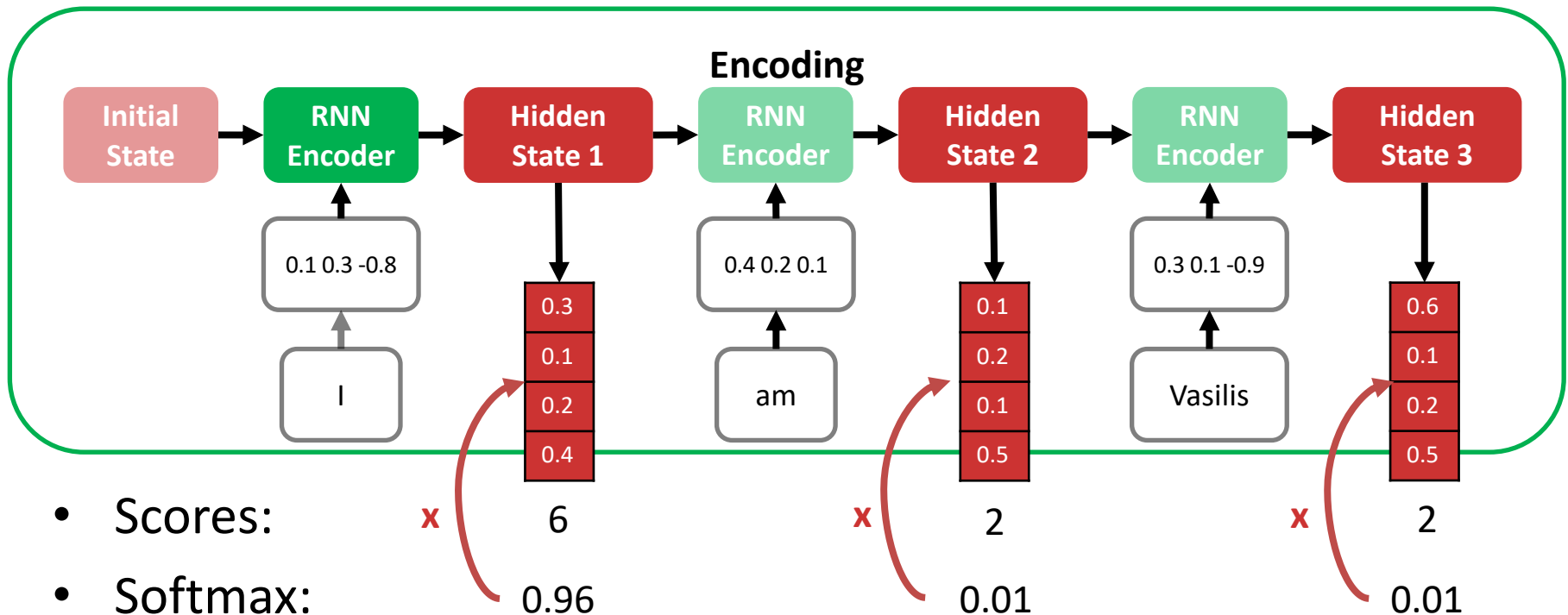
\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- At each decoding step the attention decoder looks where to focus given the encoder's the hidden states.
- For each encoder hidden state, a score is computed. There are several ways to define a score function. A common way is to multiply the dot product of the hidden states of the encoder and decoder.
- Next the scores converted to probabilities using a softmax function.
- Each encoder hidden state is weighted by each softmax probability. In this way, each encoder hidden state end us with a different importance.
- Finally, the weighted encoder hidden state vectors are summed up. The resulting vector is the new context vector.
- Consider each encoder hidden state as a word. With the softmax output, it is indicated which word(s) contribute to the current prediction.
- The aforementioned steps are performed for each decoder iteration.

# Attention Decoder (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Input to the decoder's hidden state 4.



- Multiply the input hidden states with the softmax outputs.
- The sum up the hidden states to result in a single vector.

# Attention Decoder (Cont.)

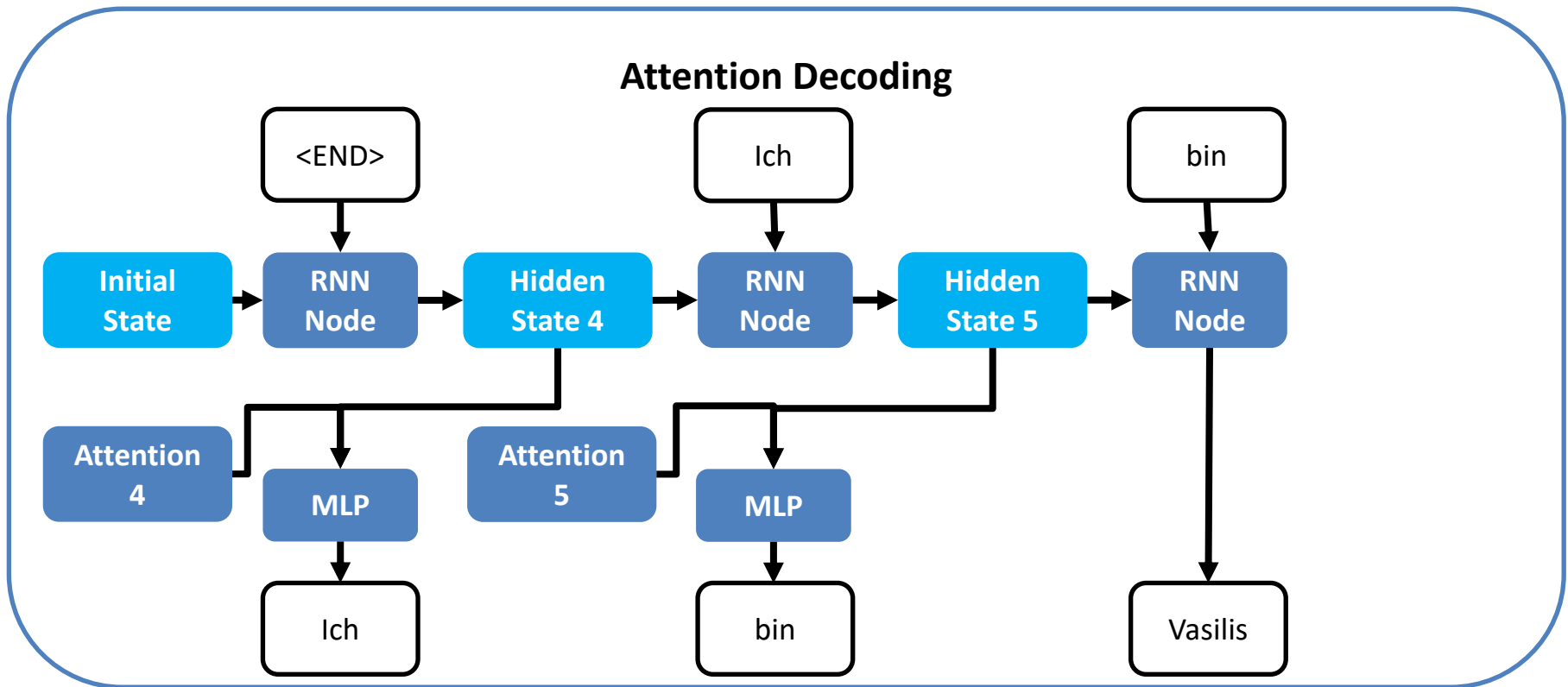
\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The context vector is computed for each decoding iteration.
- Every decoding iteration receives the previous hidden state and previous output as input.
- At the first decoding iteration, an initial hidden state and a token are used to initialise the process. For language modelling the token <END> is placed at the beginning.
- The context vector computed from the attention at the current stage and the hidden decoder state are concatenated and passed through a feed-forward network to predict the output.
- Similar to standard RNNs, the attention decoder has weights for the hidden state vector and input vector.
- The attention decoder learns to align the input with output sequences.

# Attention Decoder (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The input word to the RNN node is first encoded with an embedding. A composition of RNN nodes is also possible.



# Types of attention

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- In our example, we used all available encoder hidden states to compute an attention scores for each decoder iteration. This a global attention because we attend all encoder hidden states.
- When we use only a subset of the hidden states of the encoder to compute the attention scores, we refer to this as local attention.
- There is also the hard attention where for each decoder hidden state, we rely on a single encoder hidden state.

# The Transformer

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

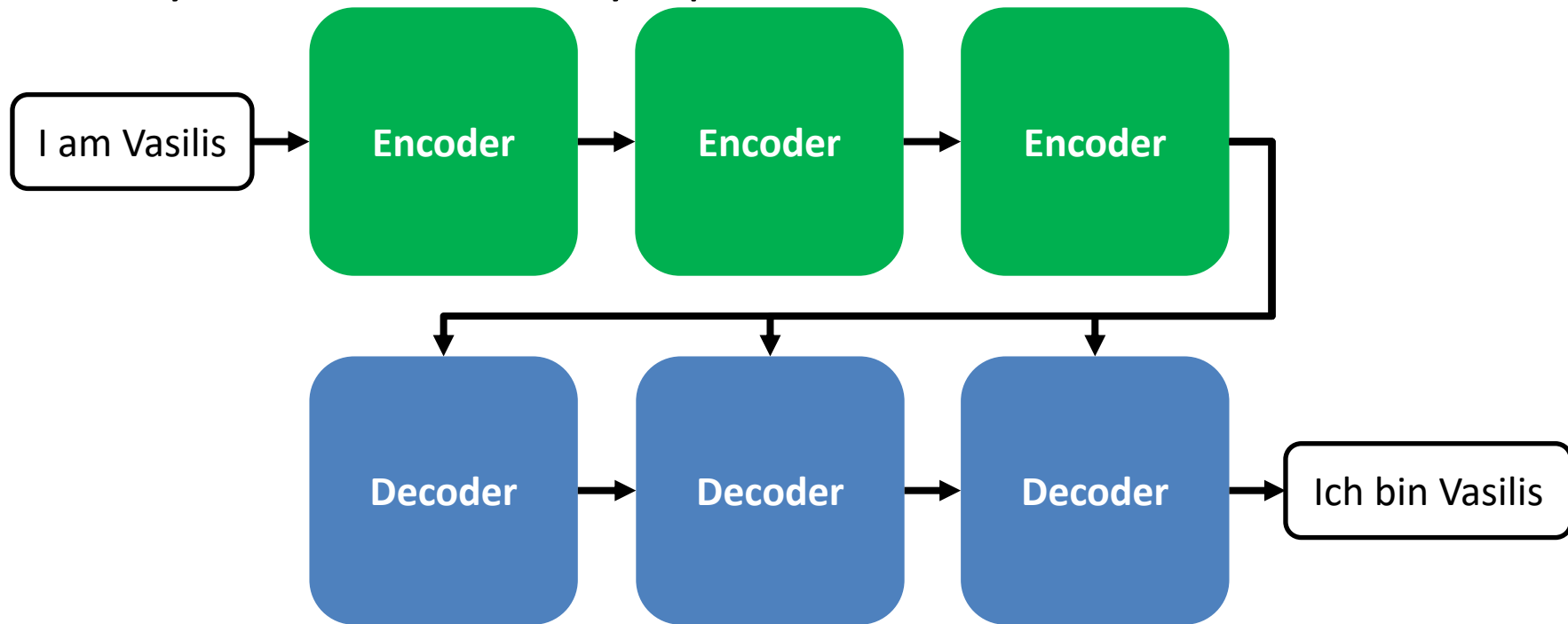
- The attention on RNN encoder-decoder networks has been a great innovation and has allowed to process long sentences in language processing. However, the sequential processing can be characterised as a bottleneck of the RNN approach.
- The transformer\* architecture allows parallel processing for the encoding part while it also presents the idea of self-attention. We focus on using it for natural language processing, such as machine translation.
- Self-attention is a mechanism for relating the input embedding to the remaining inputs of the sentences. It enables context learning, since the meaning of a word varies based on the sentence content.
- Unlike RNN attention, it is a standalone mechanism without the need to use an RNN. It can process on parallel the inputs.
- The transformer architecture is based on the idea of encoding and decoding, with multiple encoders stacked on top of each other. Similarly, there are several stacked decoders. All of them do not share weights.

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

# The Transformer (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Each encoder (or decoder) block is the same, without sharing the parameters.
- Every word in indecently inputted to the encoder.

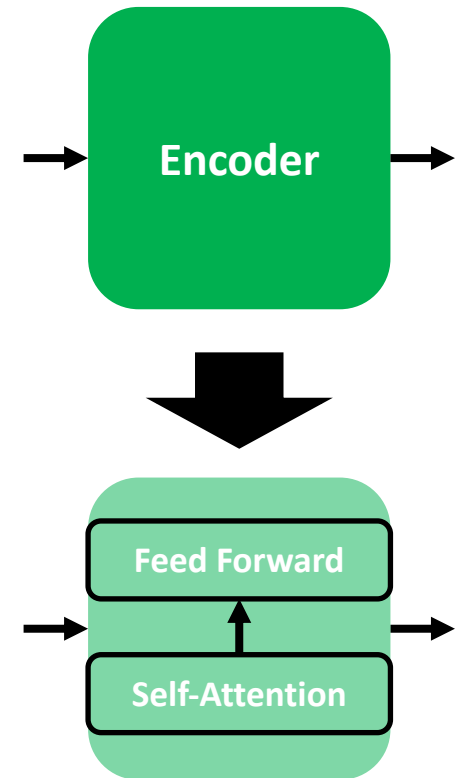




# Encoder Block

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

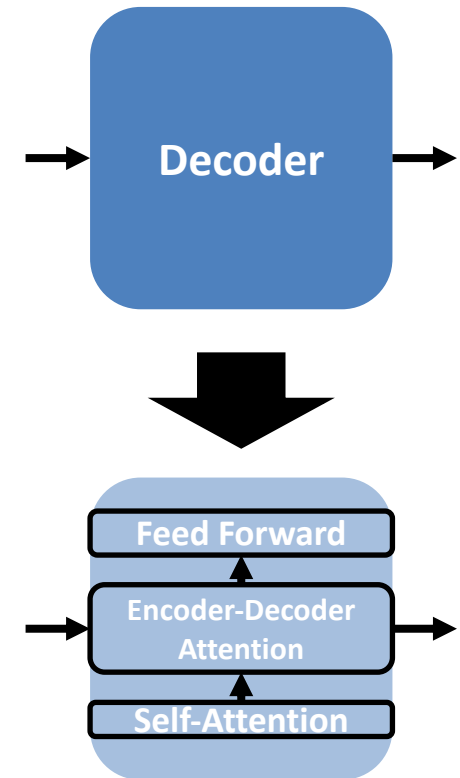
- Each encoder has a self-attention layer that is followed by a feed-forward network layer.
- For the processed word, the self-attention layer helps to attend the rest of the words of the sentence.
- Note that each word is processed independently by the feed-forward network. All input words can be processed together by the encoder.



# Decoder Block

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

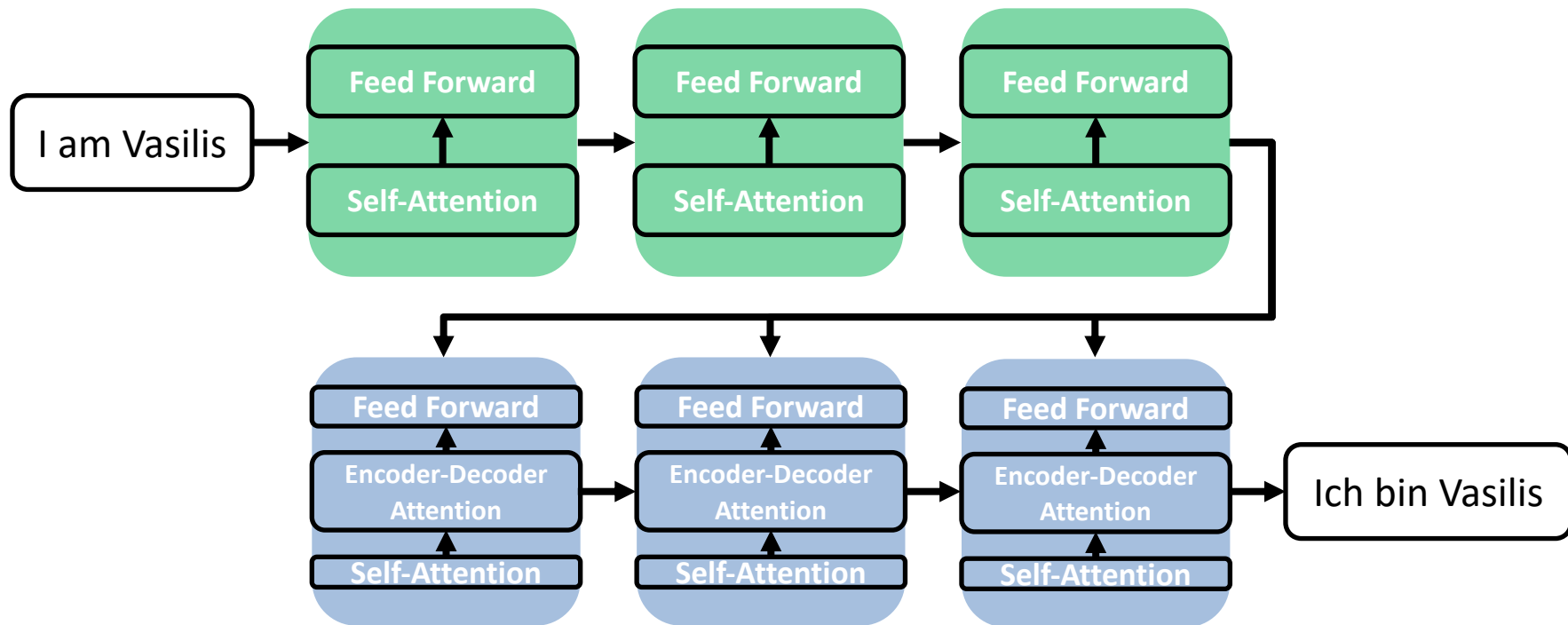
- Each decoder has a self-attention layer and the feed-forward network layer, similar to the encoder.
- In addition, it has the encoder-decoder attention layer which implements the attention to the input words.
- Note that the inputs are represented by word embeddings.



# Encoder-Decoder Blocks

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

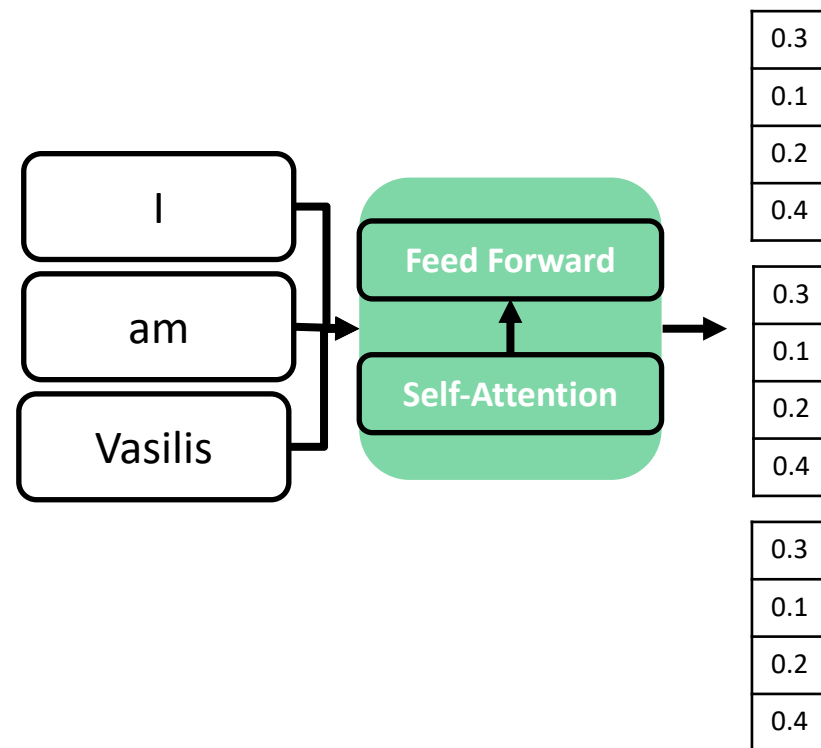
- The encoder processes all words at once.



# Encoder Processing

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The encoder processes all words at once. This allows a high degree of parallelisation with the encoder.
- The outputs of each encoder layer move to the next encoder layer.
- The self-attention layer takes care of the relations between the input words, e.g. I and Vasilis.



# Self-Attention

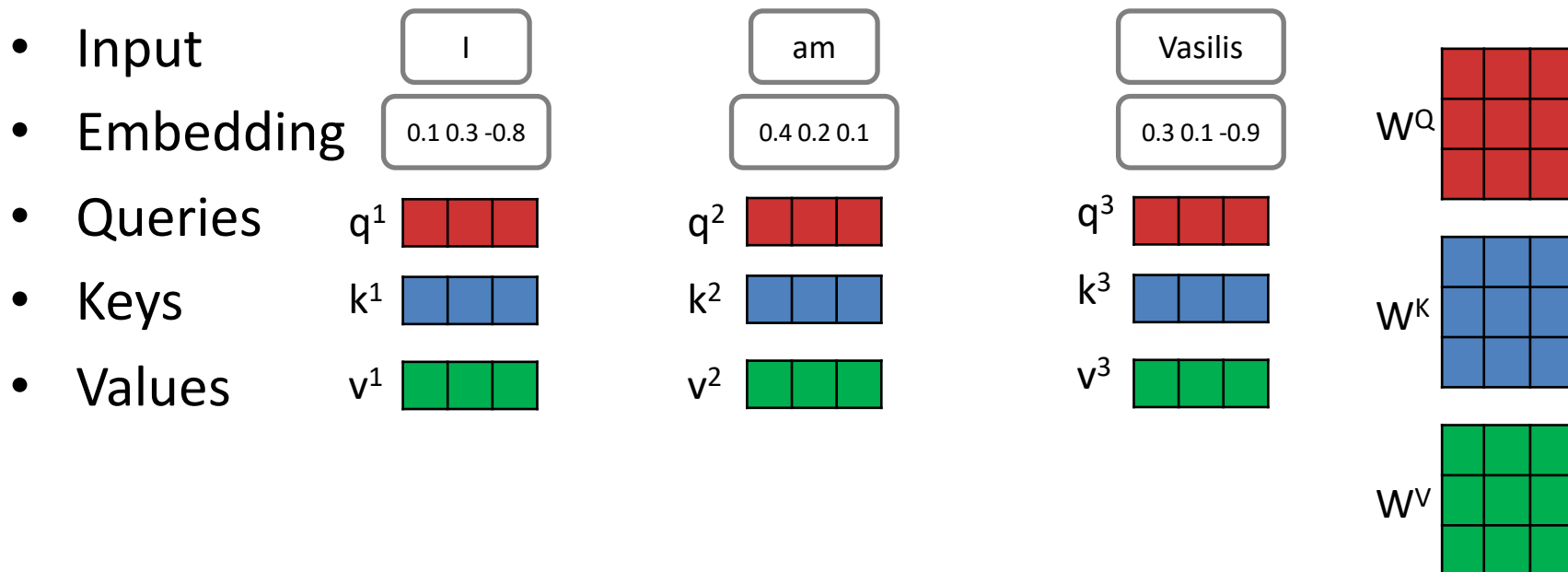
\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Consider the sentence: *Crossing the river was not possible for the turtle due to its exhaustion.*
- *What does the "its" refer to? To the river or the turtle?*
- Self-attention is the mechanism to focus on the relevant words of the sentence.
- For each input word, three vectors are created, namely the Query vector, Key vector, and Value vector. To obtain each vector, the input embedding is multiplied with three matrices. Moreover, these matrices are learned during the model training.

# Query, Key, and Value Vectors

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Matrices: Query  $W^Q$ , Key  $W^K$ , Value  $W^V$ .



- We multiply the embeddings with the three matrices to obtain the three type of vectors.

# Scores










\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

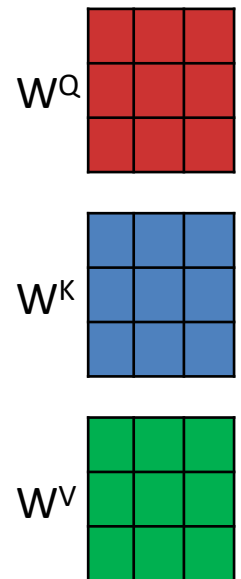
- Next, a score needs to be computed similar to the attention mechanism.
- For each word, we need to calculate a score compared to every other word in the sentence.
- For the current word, the score indicates how much attention to put on other words.
- To compute the score, we take the dot product of the query vector with the key vector of the corresponding word.

# Score Calculation

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The scores for  $q^1$  are calculated below.

• Input	I	am	Vasilis
• Embedding	0.1 0.3 -0.8	0.4 0.2 0.1	0.3 0.1 -0.9
• Queries	$q^1$ 	$q^2$ 	$q^3$ 
• Keys	$k^1$ 	$k^2$ 	$k^3$ 
• Values	$v^1$ 	$v^2$ 	$v^3$ 
• Score ( $q^1$ )	$q^1 \cdot k^1 = 30$	$q^1 \cdot k^2 = 60$	$q^1 \cdot k^3 = 10$



- The same process is repeated for  $q^2$  and  $q^3$ .



# Normalisation

---










\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

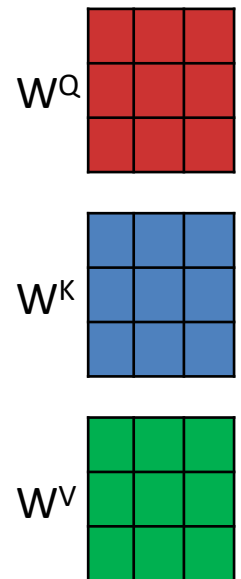
- The normalisation step follows after the scores have been calculated.
- The scores are divided by the square root of the dimension of the key vector. This helps to have stable gradients during back-propagation.
- Finally, the softmax function is used to convert the scores to probabilities.

# Normalisation

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The scores for  $q^1$  are calculated below.

• Input	I	am	Vasilis
• Embedding	0.1 0.3 -0.8	0.4 0.2 0.1	0.3 0.1 -0.9
• Queries	$q^1$ 	$q^2$ 	$q^3$ 
• Keys	$k^1$ 	$k^2$ 	$k^3$ 
• Values	$v^1$ 	$v^2$ 	$v^3$ 
• Score ( $q^1$ )	$q^1 \cdot k^1 = 6$	$q^1 \cdot k^2 = 2$	$q^1 \cdot k^3 = 4$
• Normalise	$\frac{6}{\sqrt{3}} = 3.46$	$\frac{3}{\sqrt{3}} = 1.15$	$\frac{4}{\sqrt{3}} = 2.30$
• Softmax	0.70	0.07	0.22



- How do we interpret the softmax scores?*

# Value Vector Weighting

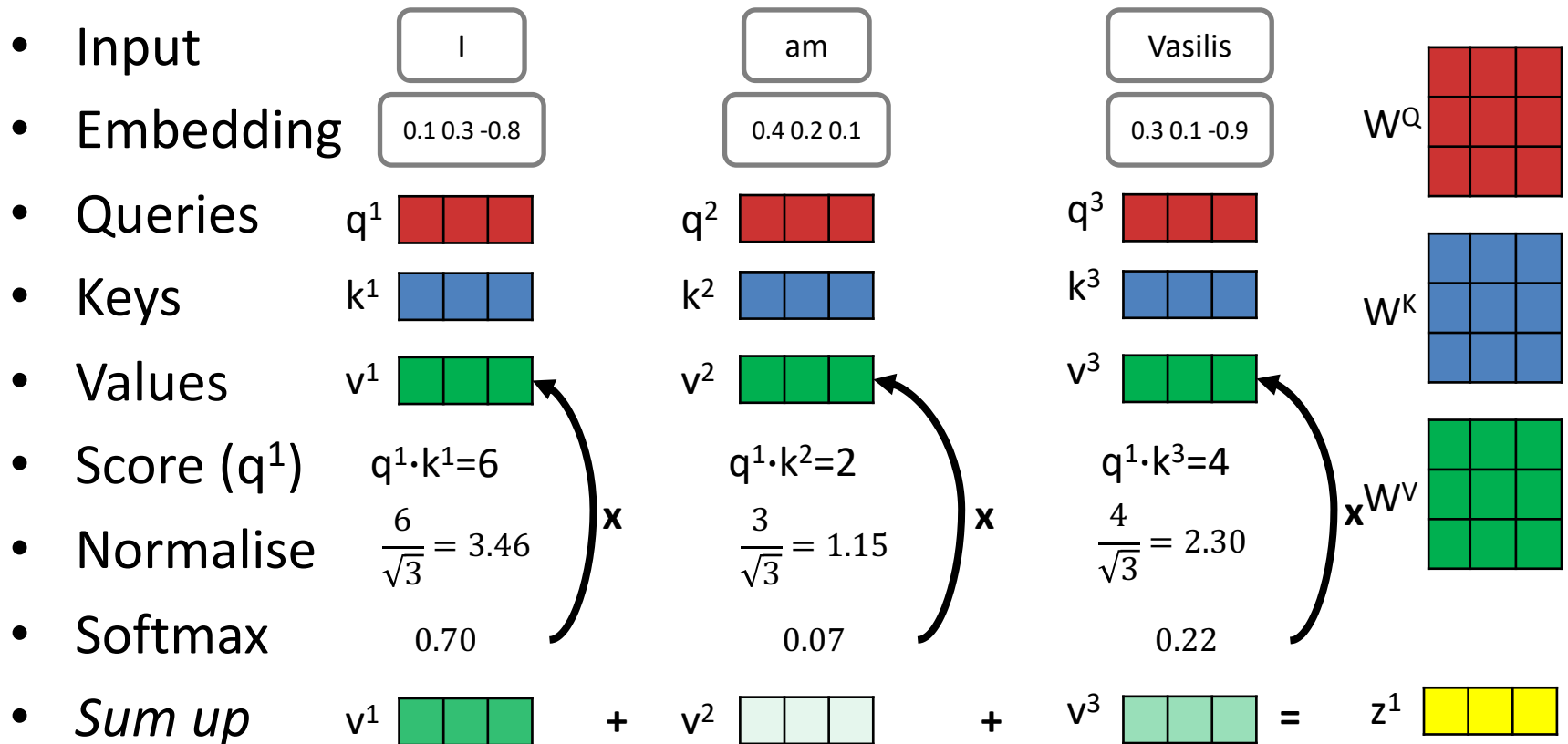
\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- After computing the softmax probabilities, the value vectors are weighted with the softmax score.
- At last they are summed up and this is the output of the self-attention layer.
- This process takes place for each individual word.

# Self-Attention Output

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The vector  $z^1$  is the output for "I".



# Feed-Forward Network

---

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The resulted vector after the self-attention layer goes to the feed-forward network.
- We examined the process for a single token (word). Next, we move to second word. However, this is a slow process. To accelerate it, the whole calculations are taking place in matrix form.

# Self-Attention (Matrix Form)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- $d$  are the dimensions of the key vector.

$$X \begin{matrix} \text{3x3 grid} \end{matrix} W^Q \begin{matrix} \text{3x3 grid} \end{matrix} = Q \begin{matrix} \text{3x3 grid} \end{matrix}$$

$$X \begin{matrix} \text{3x3 grid} \end{matrix} W^K \begin{matrix} \text{3x3 grid} \end{matrix} = K \begin{matrix} \text{3x3 grid} \end{matrix}$$

$$X \begin{matrix} \text{3x3 grid} \end{matrix} W^V \begin{matrix} \text{3x3 grid} \end{matrix} = V \begin{matrix} \text{3x3 grid} \end{matrix}$$

$$\text{softmax} \left( \frac{\begin{matrix} \text{3x3 grid} & \text{3x3 grid} \end{matrix}}{\sqrt{d}} \right) \begin{matrix} \text{3x3 grid} \end{matrix} = Z \begin{matrix} \text{3x3 grid} \end{matrix}$$

# Multi-Head Attention

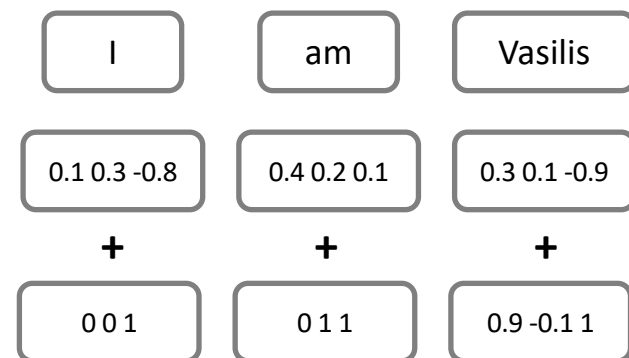
\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- One set of query, key and value matrices form one head of attention.
- Using multiple heads helps to focus on different positions.
- A multi-head attention mechanism is a common approach.
- We have now  $Z_1, Z_2, \dots, Z_N$  attention outputs.
- However, the feed-forward network expects a single attention output of fixed size. For that reason, all attention heads  $Z_1, Z_2, \dots, Z_N$  are concatenated and multiplied with an additional weight matrix to condense to a single  $Z$ .

# Positional Encoding

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Since there is no sequential encoding of the inputs, the word order is missing from the encoder.
- To overcome this limitation, a positional vector is added to the word embedding before the transformer network processing begins.
- In this way, it is possible to determine the position of each word.
- The positional encoding will be used both from the encoder and decoder.

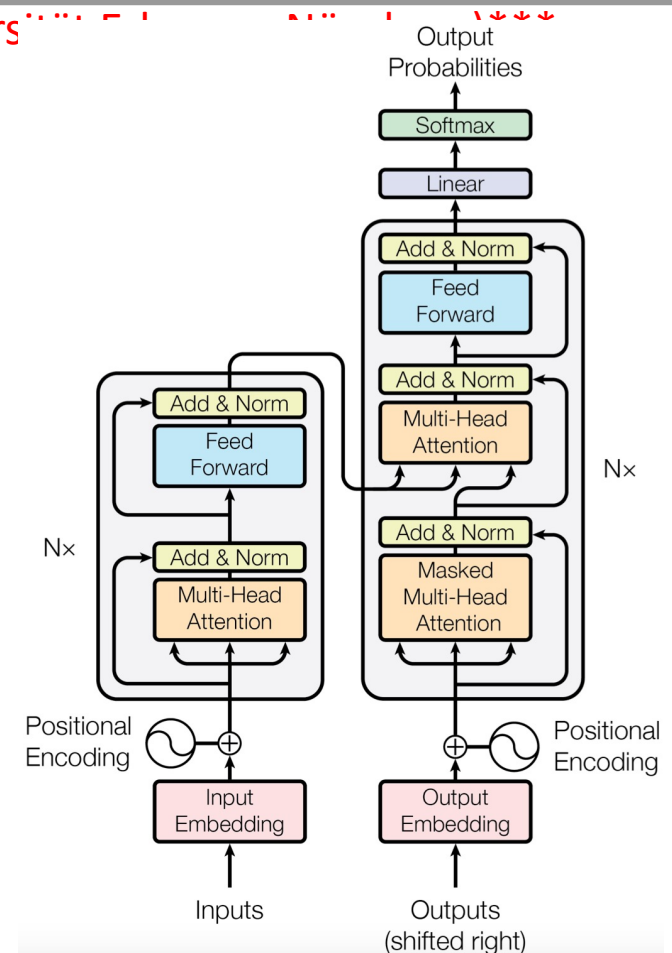




# Residual Connections & Layer Normalisation

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)

- The encoder has additionally residual connections\* and layer normalisation\*\* (add & norm block).
- These operations are part of the decoder too.
- The encoder is repeated several times. The output of the last encoder is used by the decoder.



Source: <https://arxiv.org/pdf/1706.03762.pdf>

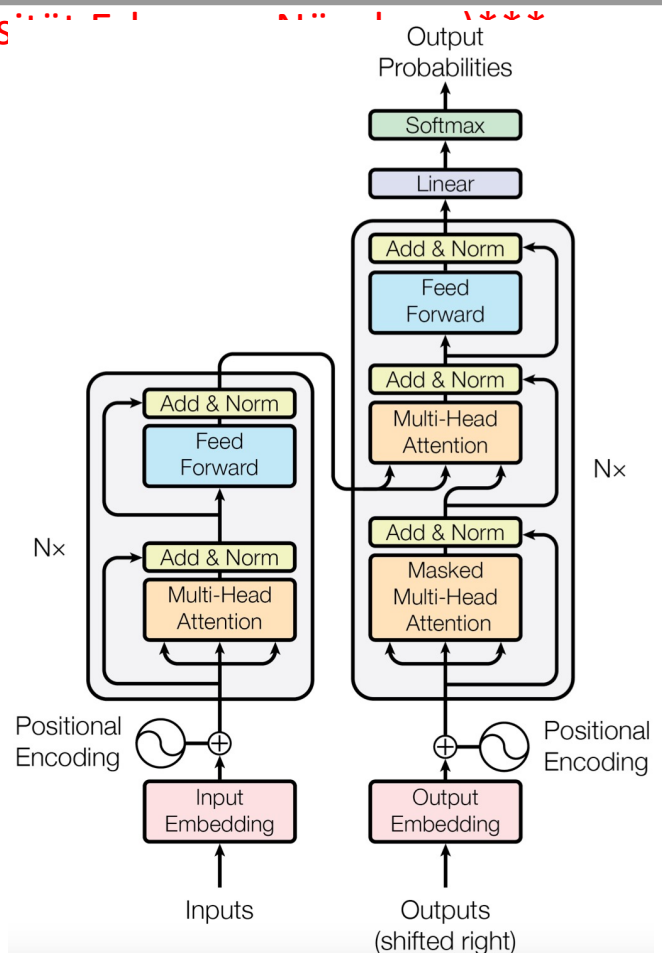
\*He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

\*\*Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer normalization." arXiv preprint arXiv:1607.06450 (2016).

# Transformer Decoder

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Univers

- The last encoder produces a set of key and value matrices to be used by the decoder in the encoder-decoder attention layer. This is another multi-head self-attention layer. The query matrix comes from the previous decoder layer.
- The encoder-decoder attention layer helps to attend the important input words for producing an output at the current position.
- The output of the decoder is used as input on the next decoding iteration. The iterations stop once a specific symbol has been predicted, e.g. <END>.
- The input to the decoder is transformed to an embedding and position encoding is added to it. Then, it is passed through a multi-head self-attention block. This is a masked attention block. This means that we only attend to the preceding positions and not the following ones.

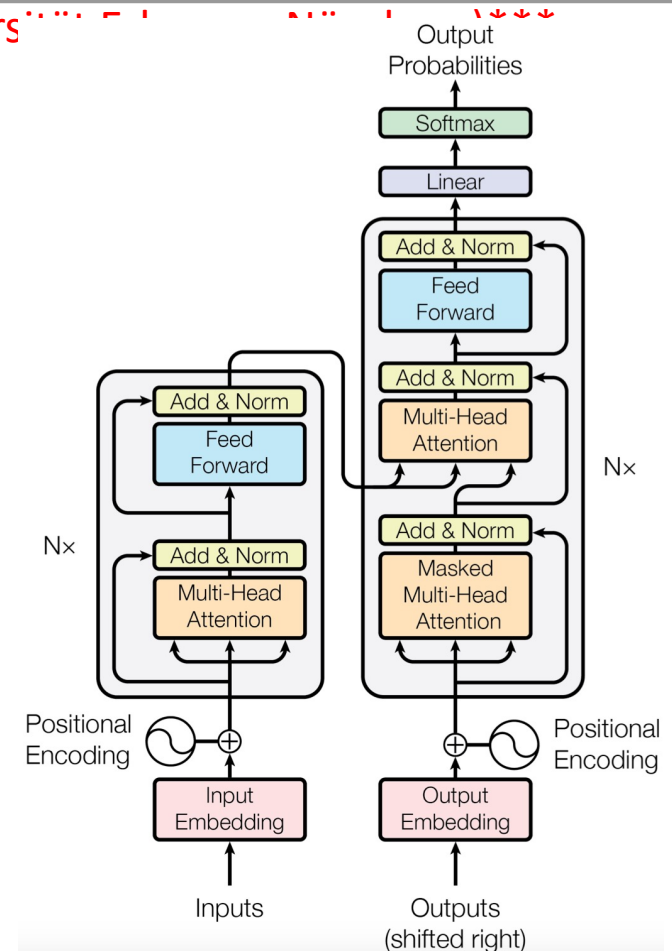


Source: <https://arxiv.org/pdf/1706.03762.pdf>

# Transformer Decoder (Cont.)

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Univers

- The output of the last decoder layer is a vector. To output probabilities for each word, a linear layer (MLP) and afterwards follows a softmax function.
- The softmax function corresponds to the output vocabulary, e.g. 30k words and symbols.
- Note that the transformer decoder processes sequential its inputs.



Source: <https://arxiv.org/pdf/1706.03762.pdf>

# Transformer Training

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- The original transformer was trained on WMT 2014 English-German\* dataset that contains about 4.5 million sentence pairs.
- The sentence encoding was based on Byte-Pair Encoding tokenization\*\* with a source-target vocabulary of around 37000 tokens.
- In addition, there were English-French training.
- Each training batch consists of sentence pairs containing around 25000 source and 25000 target tokens.
- Training can be done using the cross-entropy as loss function.

\* <https://metatext.io/datasets/wmt-14-english-german>

\*\* <https://huggingface.co/learn/nlp-course/chapter6/5?fw=pt>

# Transformer Observations

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- Transformers are the de facto state-of-the-art (SOTA) model for language models.
- Meanwhile, they have reached promising performance without being trained with supervision, GPT-2\*, RoBERTa\*\*.
- Transformer architectures started appearing for different data modalities. For instance, Vision Transformers (ViT) delivery SOTA results on standard benchmarks.
- Transformer models are generally large and complex models. Training them requires a special infrastructure. Also, the inference is not real-time without some kind of model compression.

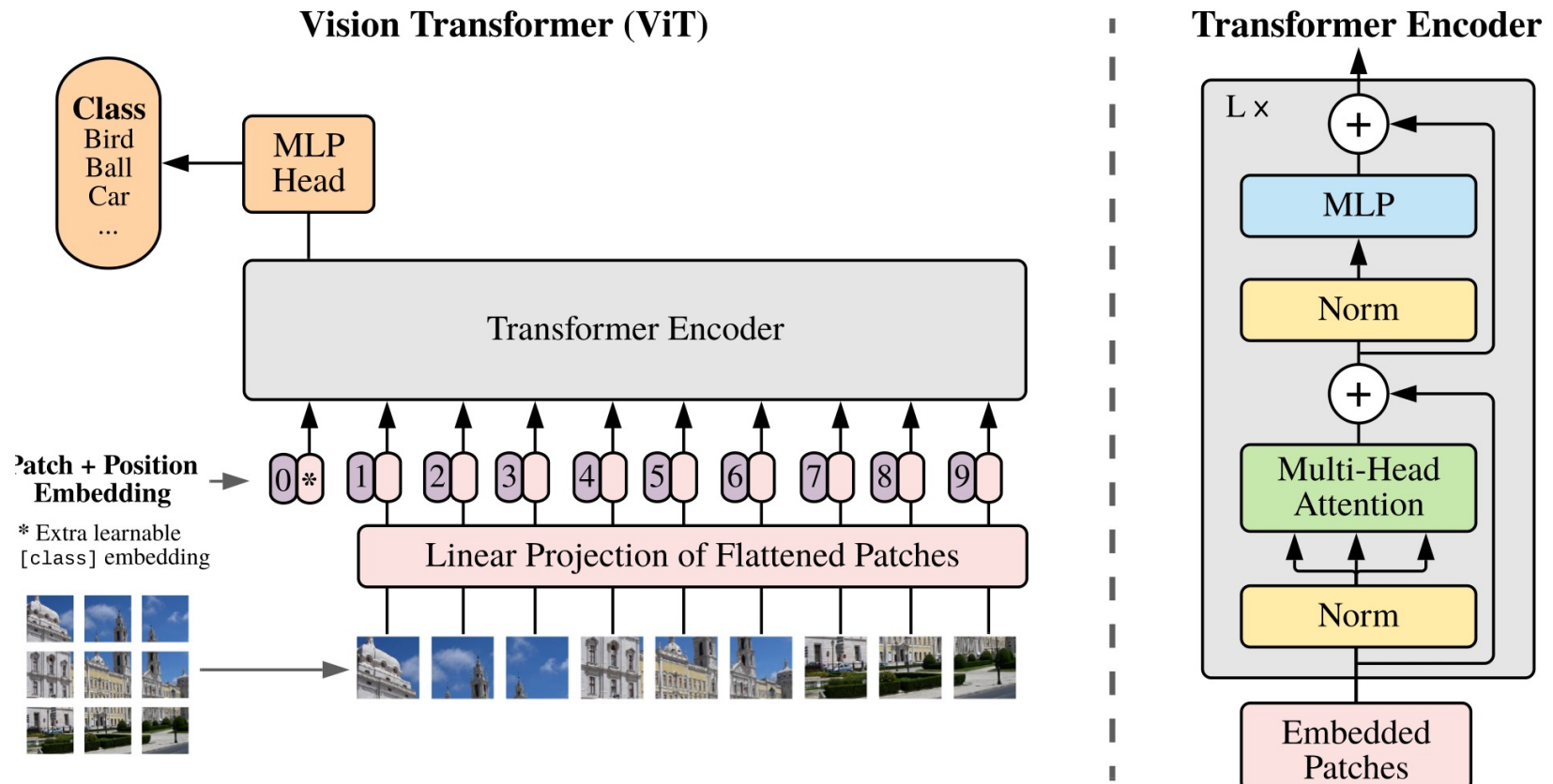
\*Liu, Yinhan, et al. "Roberta: A robustly optimized bert pretraining approach." arXiv preprint arXiv:1907.11692 (2019).

\*\*Radford, Alec, et al. "Language models are unsupervised multitask learners." OpenAI blog 1.8 (2019): 9.

# Vision Transformer

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- What is the difference from the language transformer?*



\*Dosovitskiy, Alexey, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020). Image Source: <https://arxiv.org/pdf/2010.11929.pdf>

# Study Material

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

- *Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le. "Sequence to sequence learning with neural networks." Advances in neural information processing systems 27 (2014).*
- *Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).*
- <https://sebastianraschka.com/blog/2023/self-attention-from-scratch.html>
- <https://jalammar.github.io/illustrated-transformer/>
- <https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/>
- <https://www.kdnuggets.com/2021/01/attention-mechanism-deep-learning-explained.html>

# Next Lecture

\*\*\*Not for sharing (LMS, Friedrich-Alexander-Universität Erlangen-Nürnberg)\*\*\*

## Self-supervised learning