Machine Learning

2024 (ML-2024)

Lecture 9. Recurrent neural networks

by Alexei Valerievich Kornaev, Dr. habil. in Eng. Sc.,
Researcher at the RC for AI, Assoc. Prof. of the Robotics and CV Master's Program, Innopolis University
Researcher at the RC for AI, National RC for Oncology n.a. NN Blohin
Professor at the Dept. of Mechatronics, Mechanics, and Robotics, Orel State University

Agenda

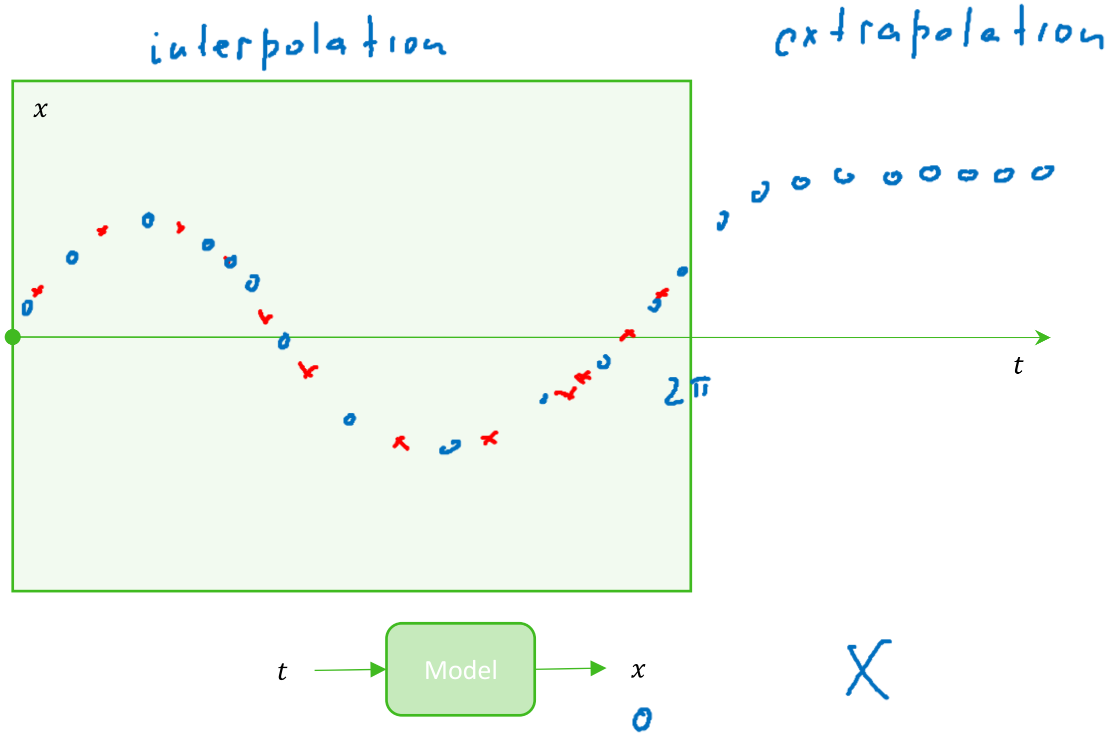
I. TEMPORAL DATA PROCESSING INTUITION

II. RECURRENT NEURAL NETWORKS (RNNs)

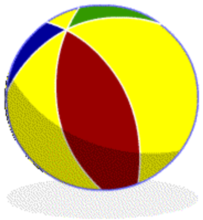
III. TRANSFORMERS (Ts)

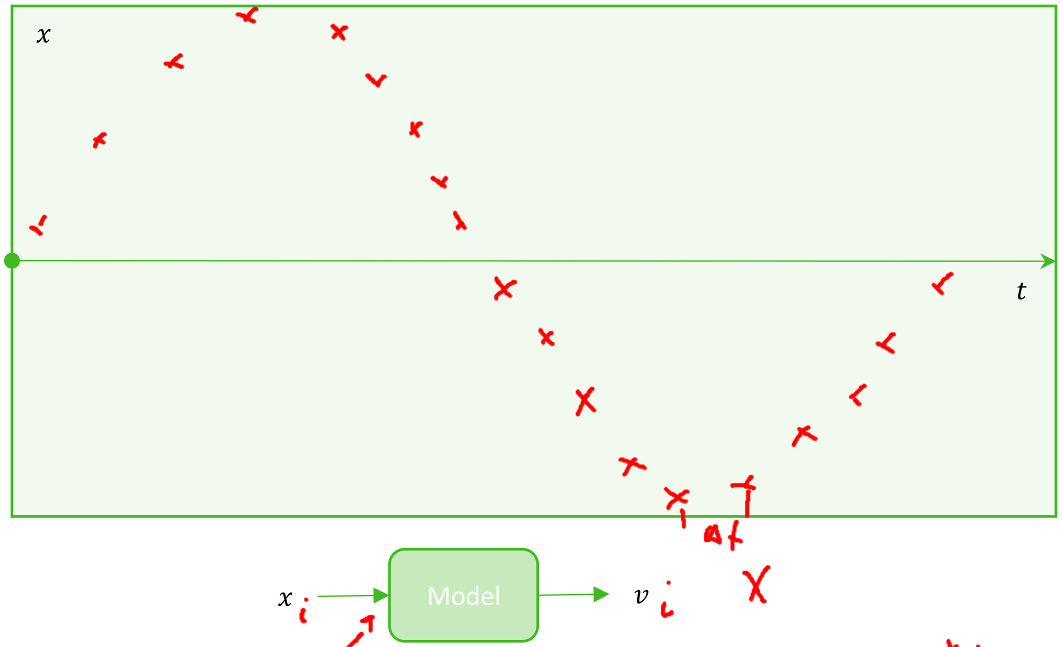
All you need is love (all together now)
All you need is love (everybody)
All you need is love, love
Love is all you need
/Lennon, McCartney/

Why sequence processing is a challenging problem?



Why sequence processing is a challenging problem?

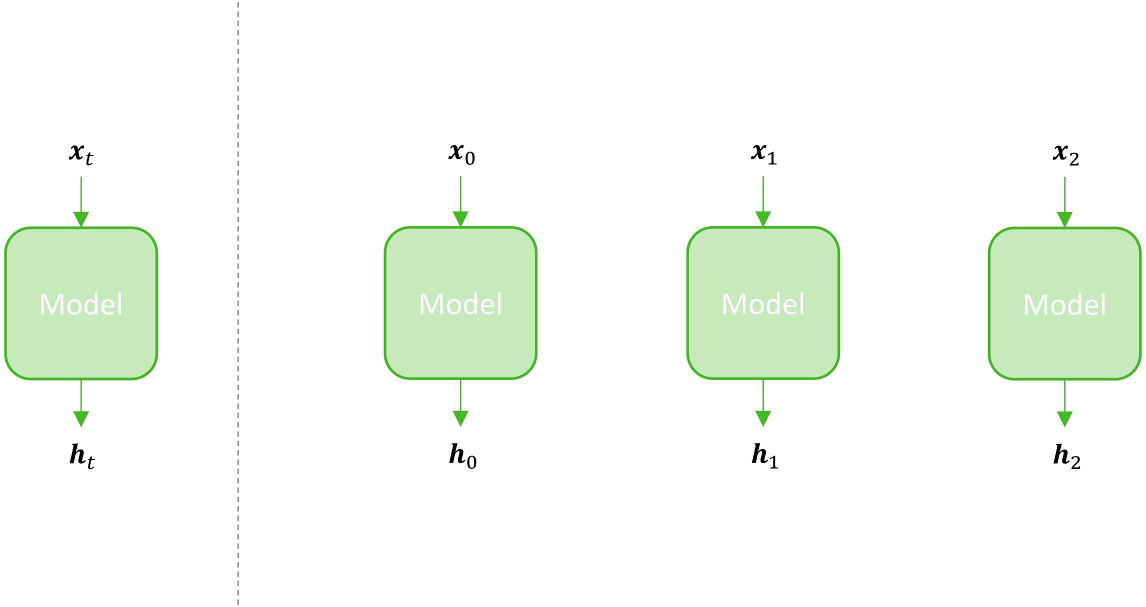




x_{i-1}

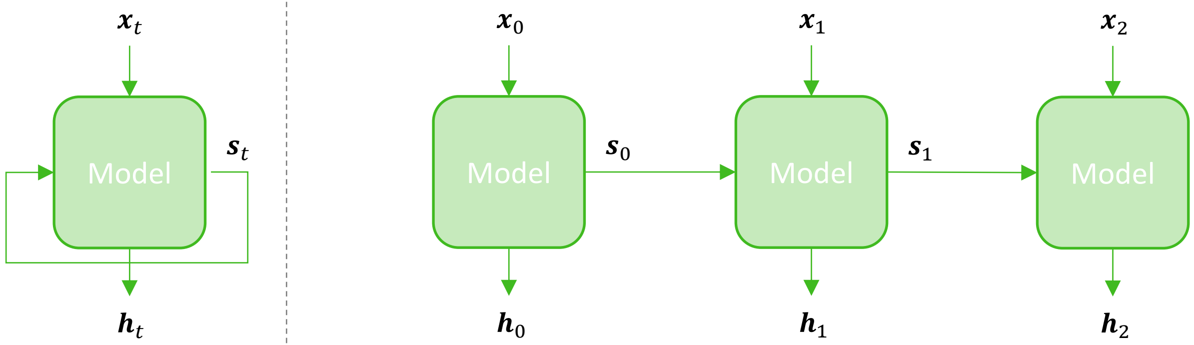
$$v_i = \frac{x_i - x_{i-1}}{\Delta t}$$

Sequence modeling intuition



[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

Sequence modeling intuition



$$h_t = (x_t, \quad s_{t-1})$$

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

Sequence modeling applications

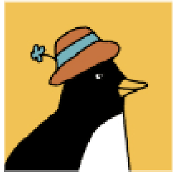
Many-to-one

Было сложно, мне понравилось

↓

Позитивный

One-to-many



→ Пингвин в шляпе

Many-to-many, не синхронизированный

I seem to have overfitted

↓

Кажется, я переобучился

Many-to-many, синхронизированный

Будет	сложно,	вам	понравится
↓	↓	↓	↓
глагол	наречие	местоимение	глагол

[Yandex Handbook on ML]

Agenda

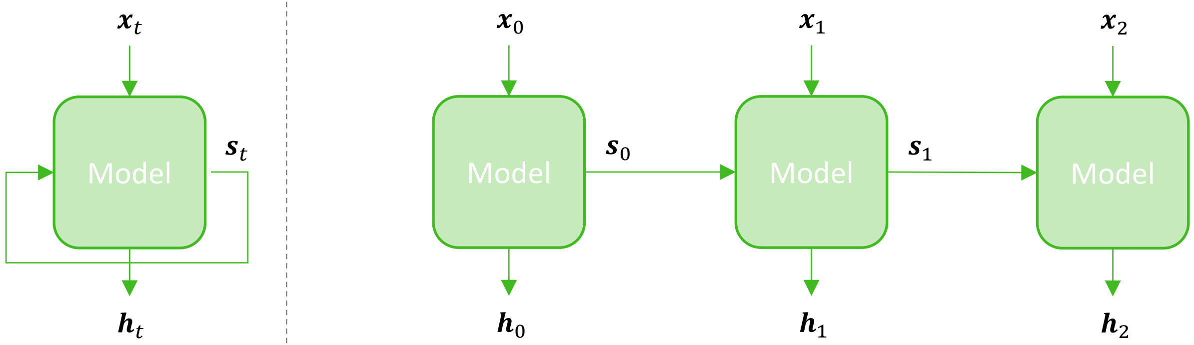
I. TEMPORAL DATA PROCESSING INTUITION

II. RECURRENT NEURAL NETWORKS (RNNs)

III. TRANSFORMERS (Ts)

All you need is love (all together now)
All you need is love (everybody)
All you need is love, love
Love is all you need
/Lennon, McCartney/

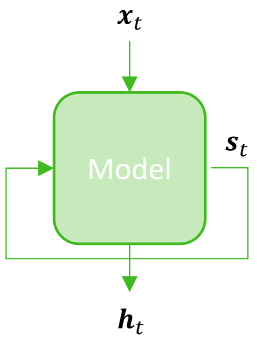
RNNs intuition: apply a recurrence relation to process a sequence



$$\boldsymbol{s}_t = (\boldsymbol{x}_t, \quad \boldsymbol{s}_{t-1})$$

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

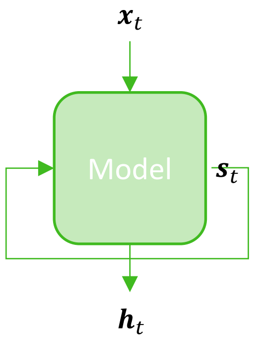
RNNs intuition: apply a recurrence relation to process a sequence




```
#initialization
s = [0,0,0,0]

#data
x = ['all','you','need','is']
l = len(x)

#model
model = my_RNN()

#training
for i in range(l):
    #prediction, hidden state
    h_i, s_i_plus_1 = my_RNN(x[i],s[i])

#test
next_word_prediction = prediction
>>'love'
```

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

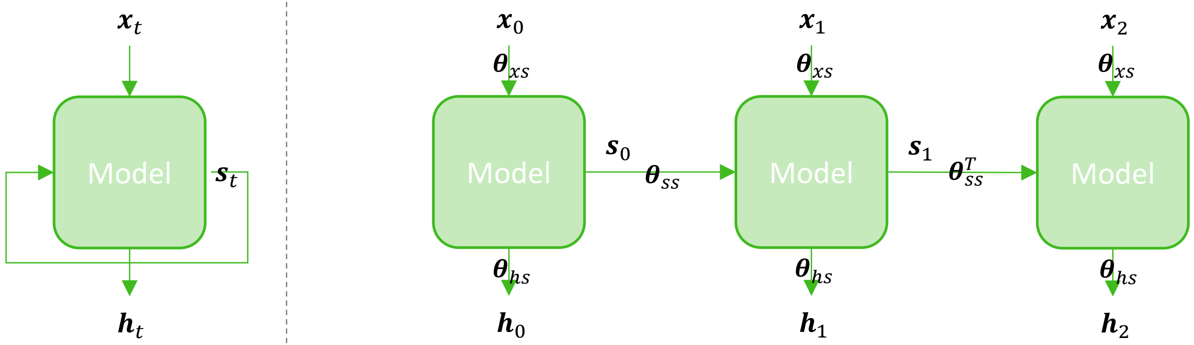
RNNs intuition: apply a recurrence relation to process a sequence



1. Input $\boldsymbol{x}_t$
2. Update hidden state: $\boldsymbol{s}_t = (\boldsymbol{\theta}_{xs}^T \boldsymbol{x}_t + \boldsymbol{\theta}_{ss}^T \boldsymbol{s}_{t-1})$
3. Make prediction: $\boldsymbol{h}_t = \boldsymbol{\theta}_{hs}^T \boldsymbol{s}_t$.

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

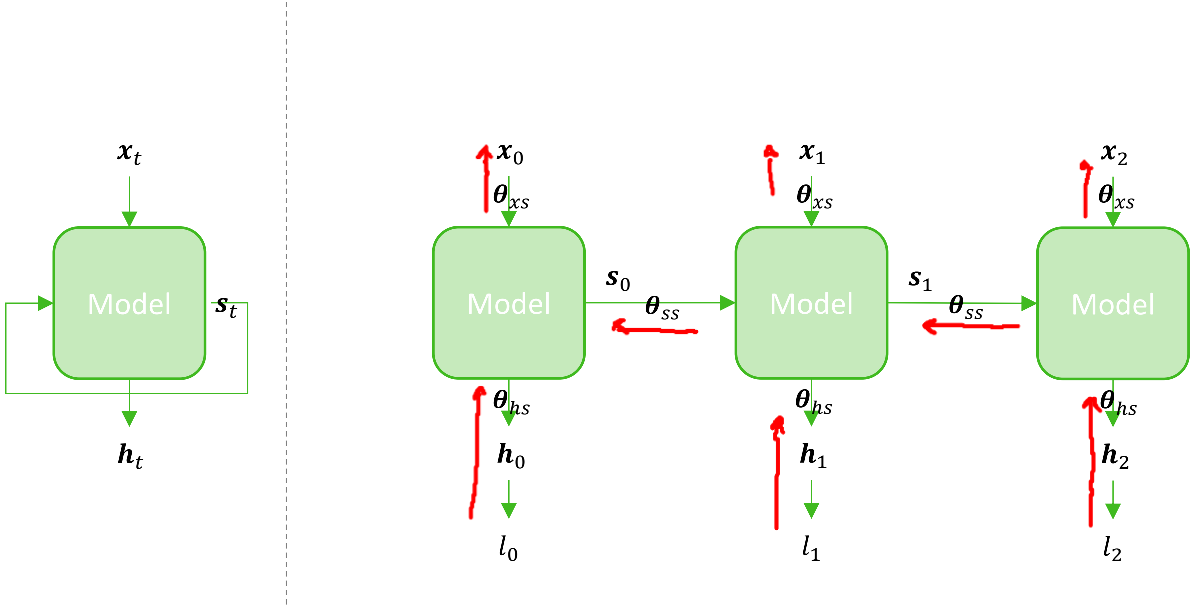
RNNs intuition: apply a recurrence relation to process a sequence



The model re-uses same weight matrices at every time step

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

RNNs gradient flow: exploding and vanishing gradients



Many values < 1: vanishing

1. Activation functions
2. Weights initialization
3. Network architecture

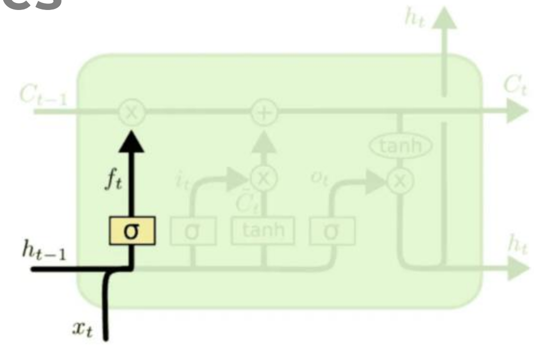
$$L = \sum_t l_t$$

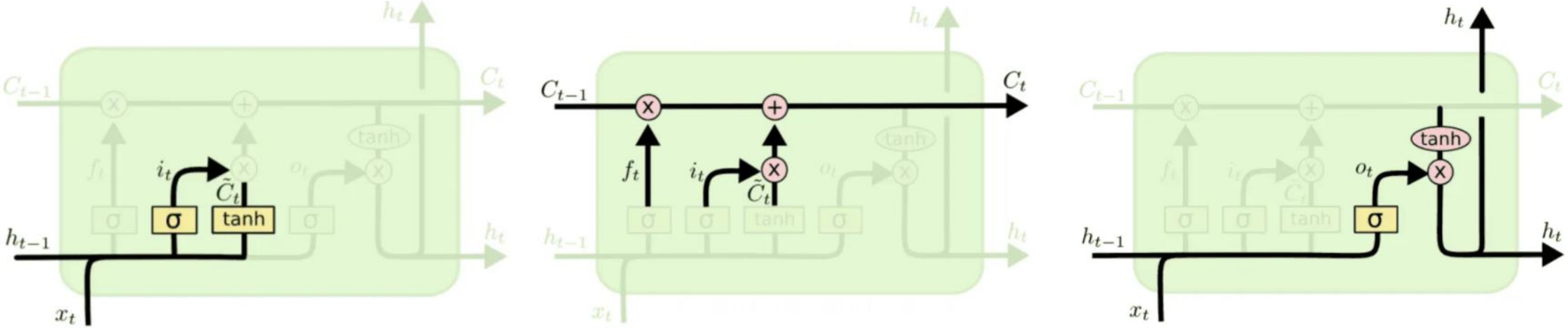
Many values > 1: exploding

Gradient clipping to scale big gradients

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

Long short-term memory networks (LSTMs): forget, store, update, output gates





[Understanding-LSTMs]

Why recurrent neural networks (RNNs) are good at sequences processing?

1. **Sequential Memory:** RNNs have a built-in memory mechanism that allows them to maintain information across time steps. This is crucial for tasks where the context from previous time steps is important for making predictions at the current time step.
2. **Variable-Length Inputs:** Unlike traditional neural networks, RNNs can handle sequences of variable lengths. This flexibility is essential for processing data where the length of the input sequence can vary, such as sentences in natural language or time series data.
3. **Contextual Understanding:** RNNs can capture dependencies and patterns in the data that span multiple time steps. This is particularly useful in tasks like language modeling, where understanding the context of a word often requires looking at the words that come before it.
4. **Stateful Processing:** RNNs can maintain a hidden state that summarizes the information from previous time steps. This stateful processing allows RNNs to model long-term dependencies in the data.
5. **Versatility:** RNNs can be adapted for various tasks by modifying their architecture. For example, Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) are specialized types of RNNs designed to better capture long-range dependencies and mitigate the vanishing gradient problem.

DeepSeek-V2.5 speaks

Agenda

I. TEMPORAL DATA PROCESSING INTUITION

II. RECURRENT NEURAL NETWORKS (RNNs)

III. TRANSFORMERS (Ts)

All you need is love (all together now)
All you need is love (everybody)
All you need is love, love
Love is all you need
/Lennon, McCartney/

Transformer is a game changer

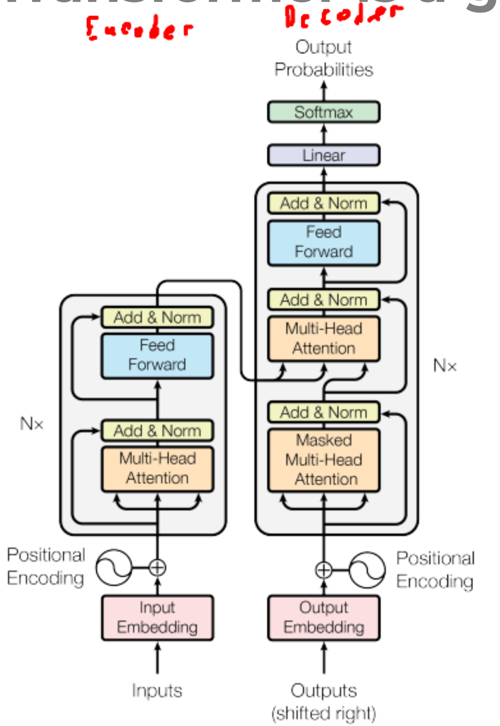
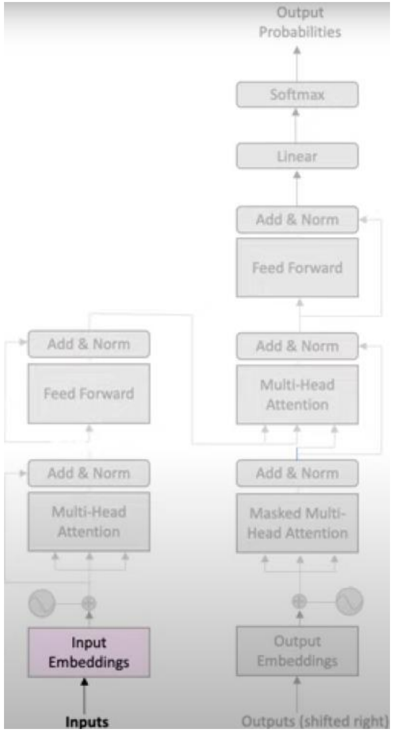


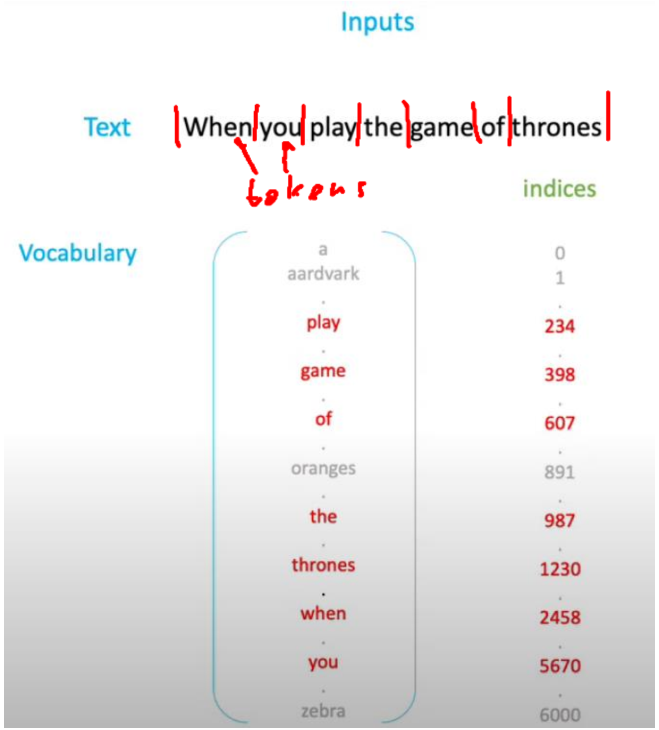
Figure 1: The Transformer - model architecture.

[Attention is all you need, 2017]

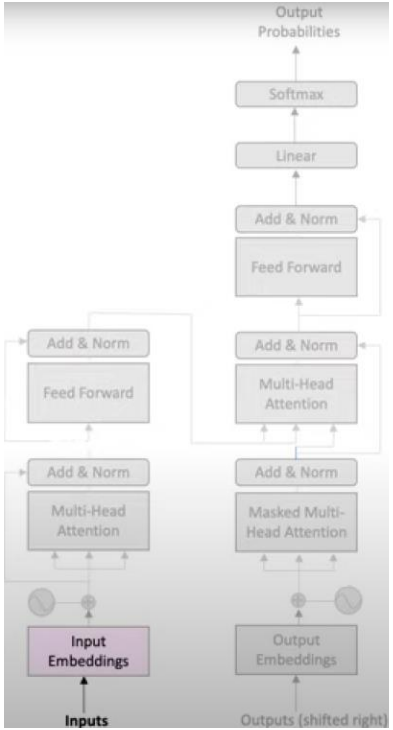
Inputs to tokens and embeddings

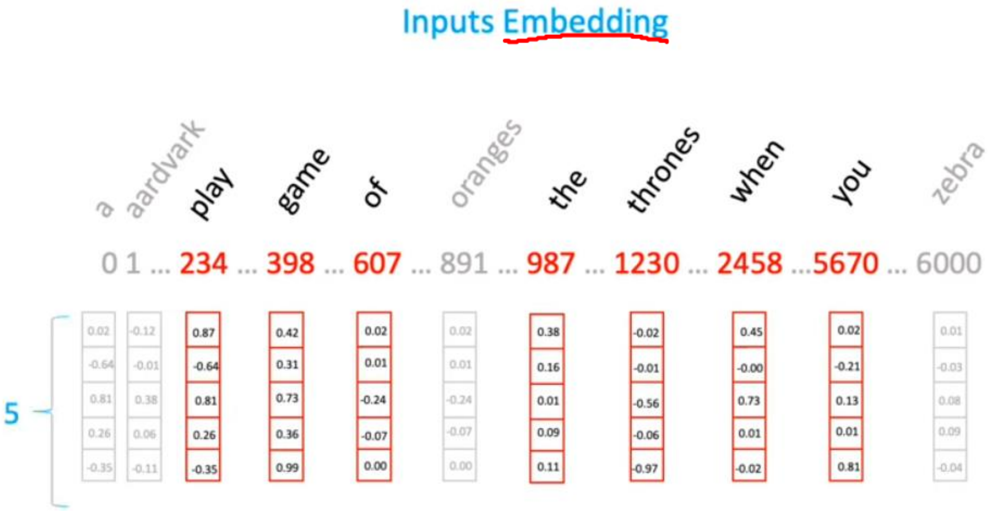


[Visual guide to Transformers]



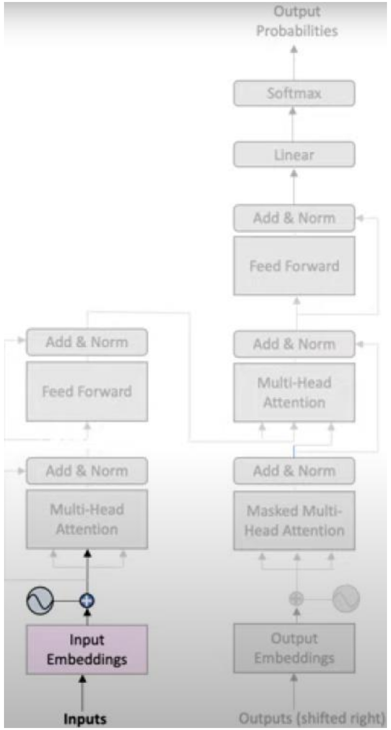
Inputs to tokens and embeddings

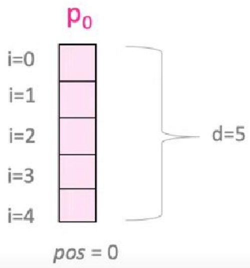




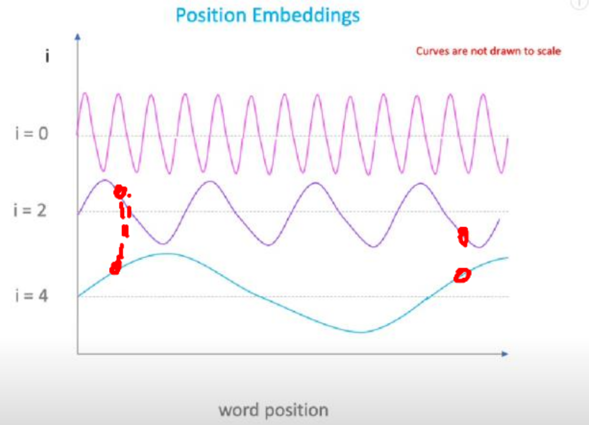
[Visual guide to Transformers]()

Positional encoding



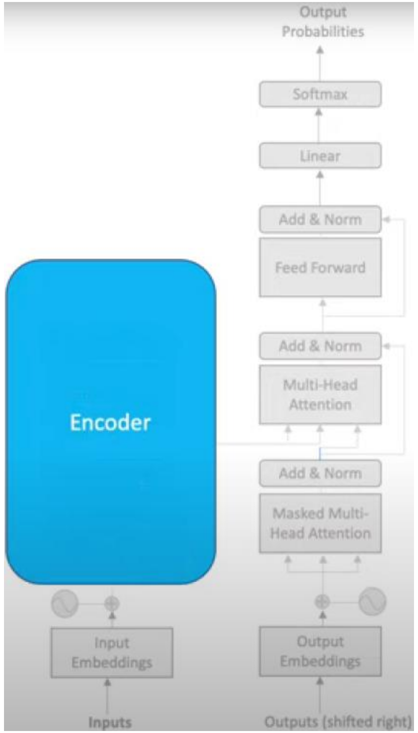


$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d}}}\right)$$



[Visual guide to Transformers]

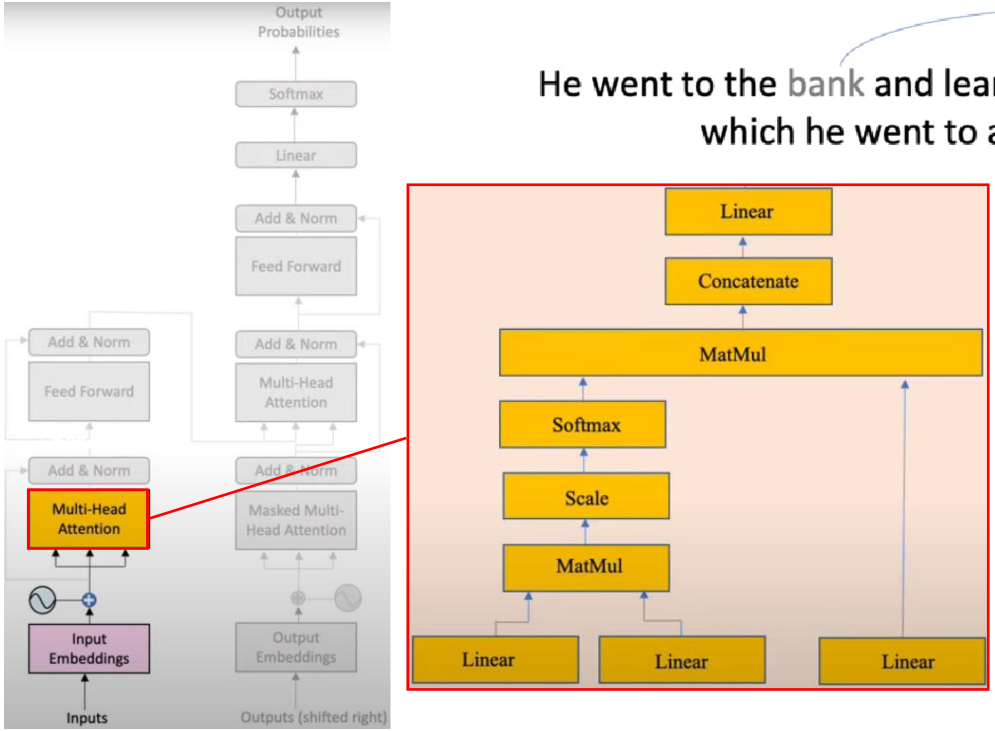
T encoder and self-attention intuition



[Visual guide to Transformers]()

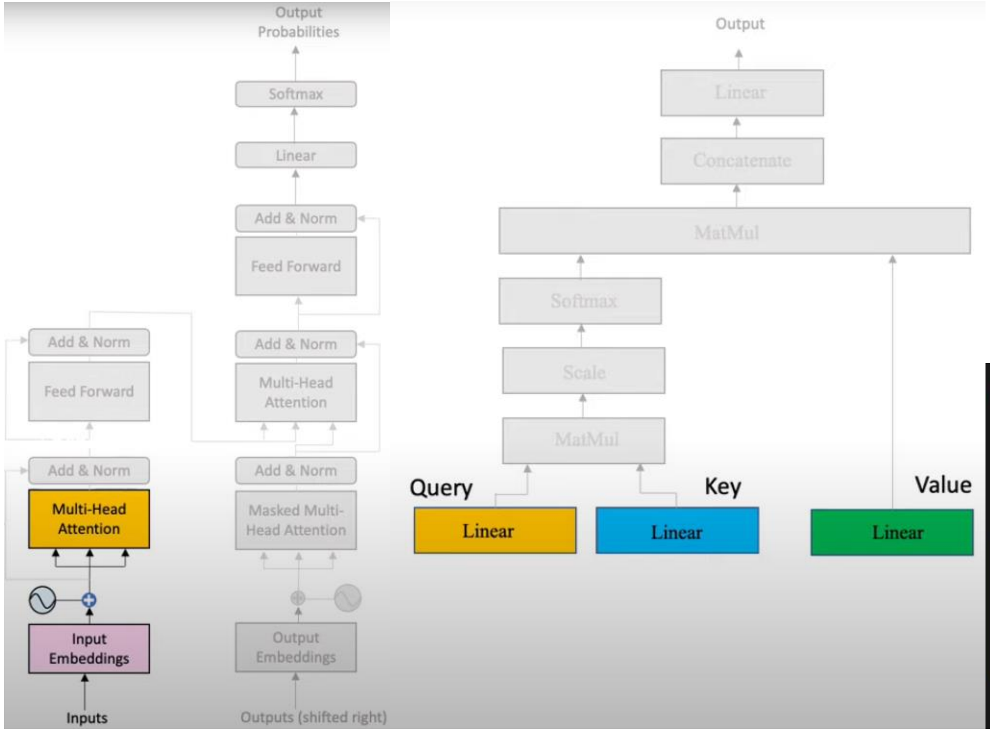
T encoder and self-attention intuition

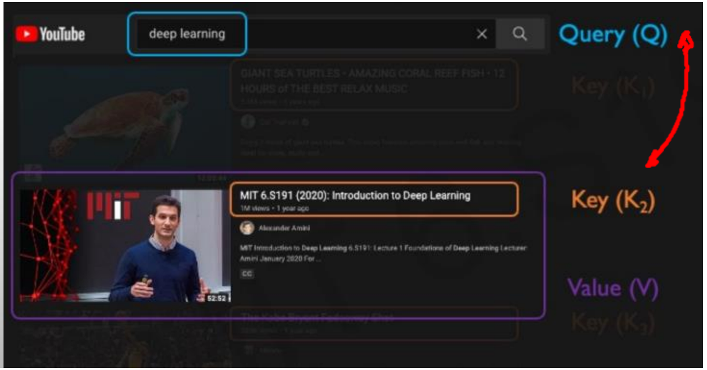
He went to the bank and learned of his empty account, after which he went to a river bank and cried.



[Visual guide to Transformers]

Self-attention intuition

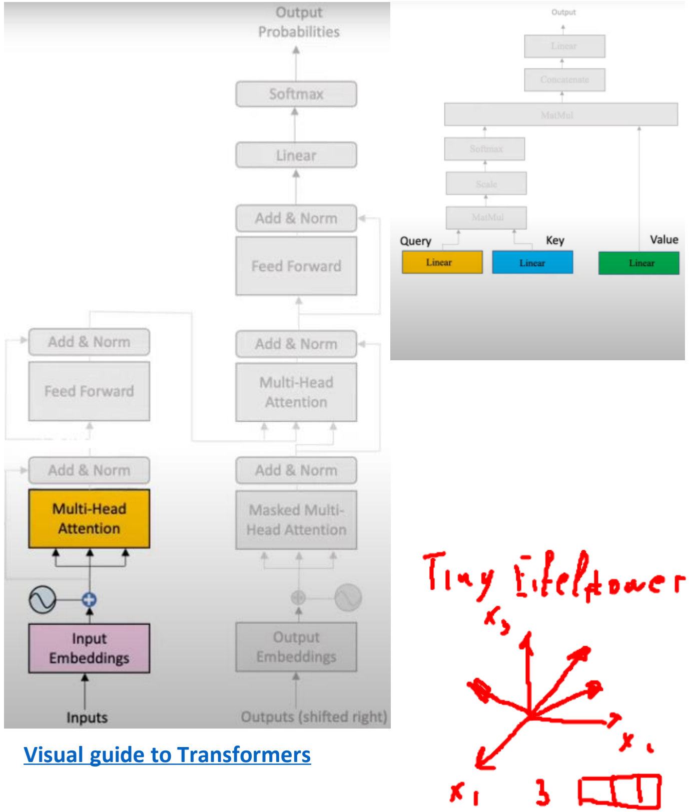


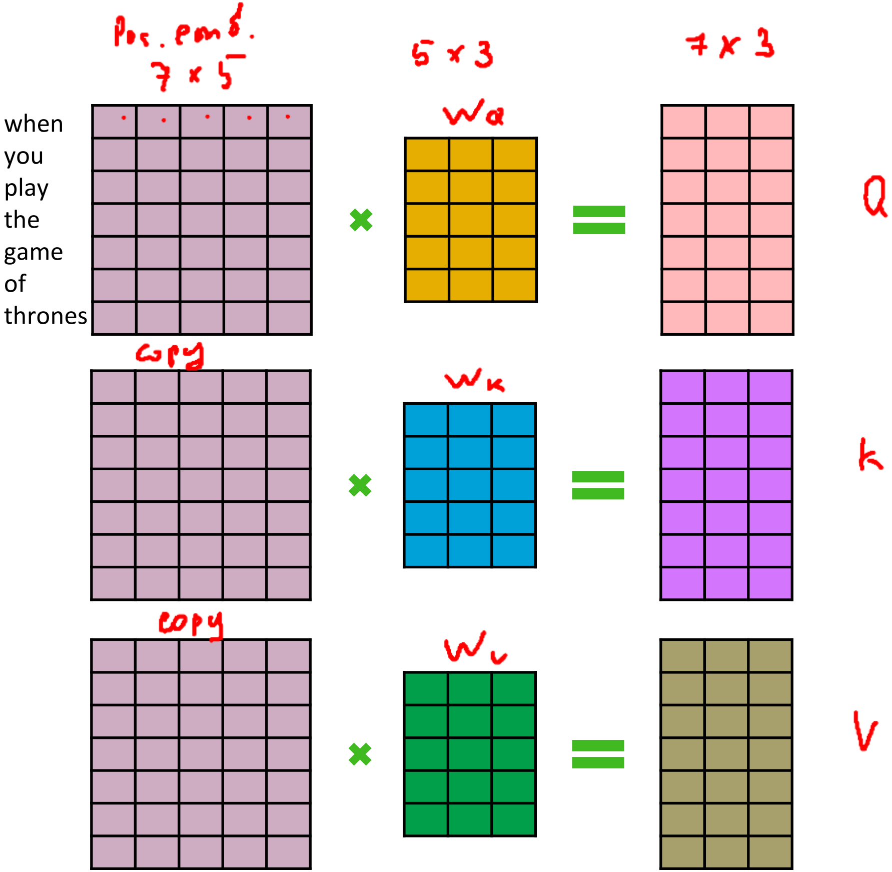


[Visual guide to Transformers]

[MIT 6.S191: Recurrent Neural Networks, Transformers, and Attention]

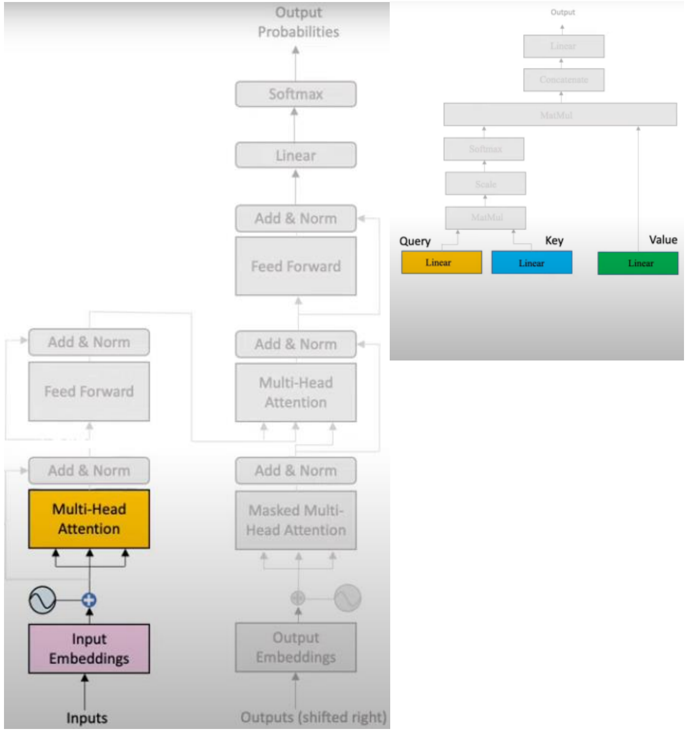
Self-attention intuition

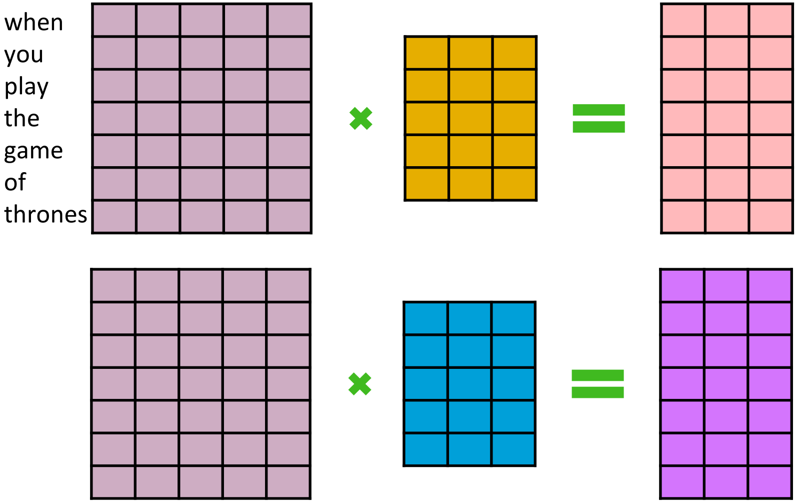




[Visual guide to Transformers]

Self-attention intuition

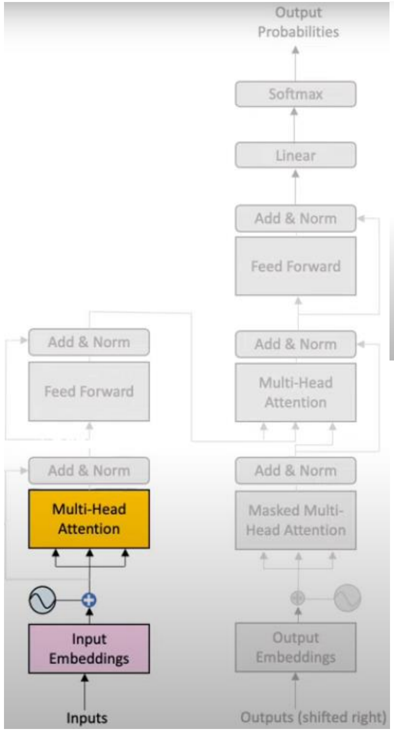


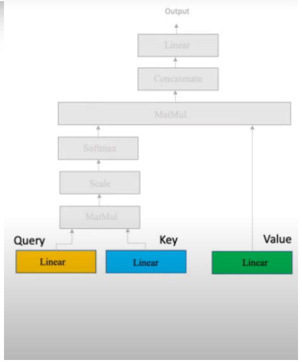


$$similarity(Q, K) = \frac{Q \quad K^T}{scaling}$$

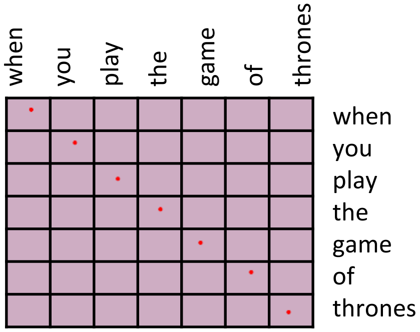
[Visual guide to Transformers]

Self-attention intuition





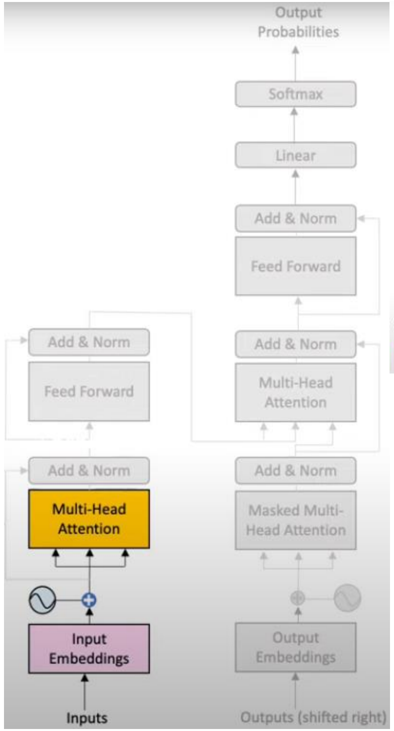
$$similarity(Q, K) = \frac{Q\ \ K^T}{scaling} =$$

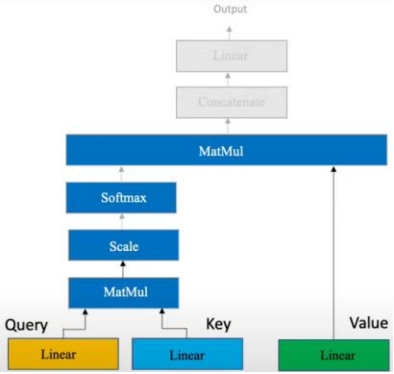


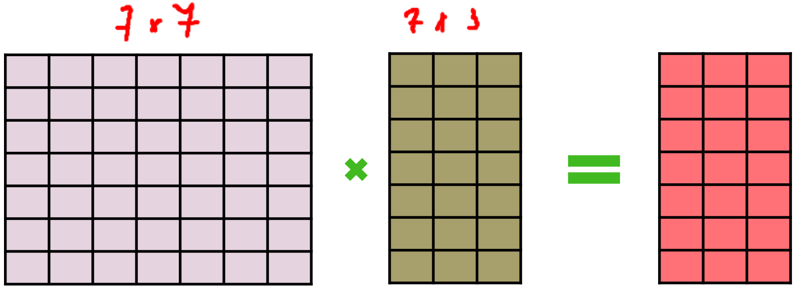
Att. filter

[Visual guide to Transformers]

Self-attention intuition





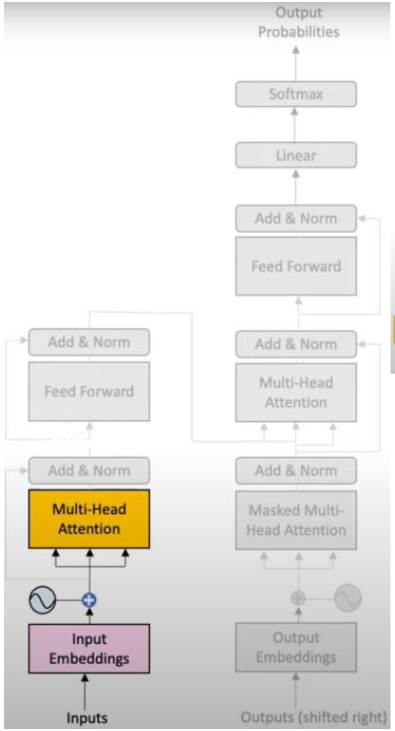


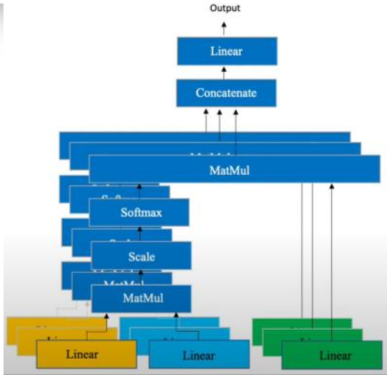
$$Attention(Q, K, V) = softmax\left(\frac{Q \quad K^T}{scaling}\right)V$$

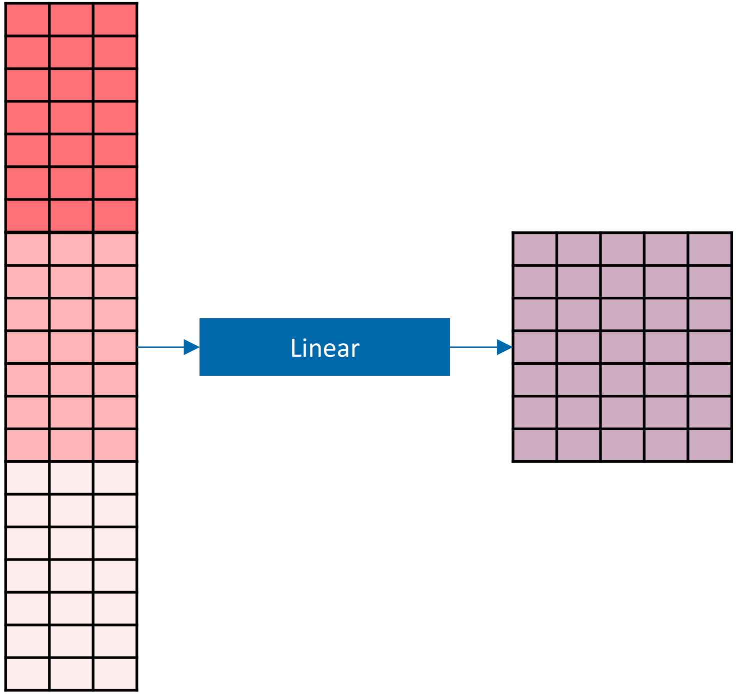
[Visual guide to Transformers]

ML-2024
by Alexei Kornaev

Multi-head attention intuition







[Visual guide to Transformers]

T's encoder updates the input embeddings

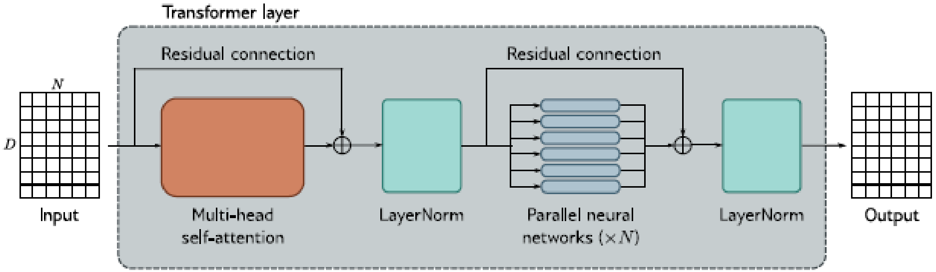
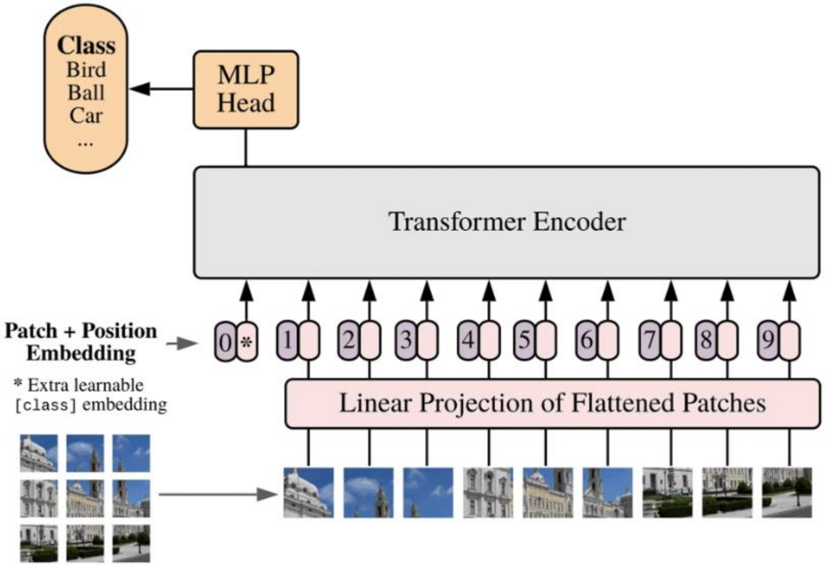


Figure 12.7 Transformer layer. The input consists of a $D \times N$ matrix containing the D-dimensional word embeddings for each of the N input tokens. The output is a matrix of the same size. The transformer layer consists of a series of operations. First, there is a multi-head attention block, allowing the word embeddings to interact with one another. This forms the processing of a residual block, so the inputs are added back to the output. Second, a LayerNorm operation is applied. Third, there is a second residual layer where the same fully connected neural network is applied separately to each of the N word representations (columns). Finally, LayerNorm is applied again.

S. J. Prince. Understanding Deep Learning. MIT Press, 2023.

ML-2024
by Alexei Kornaev

Visual Ts



[Yandex Handbook on ML]

Visual Ts: self-attention intuition

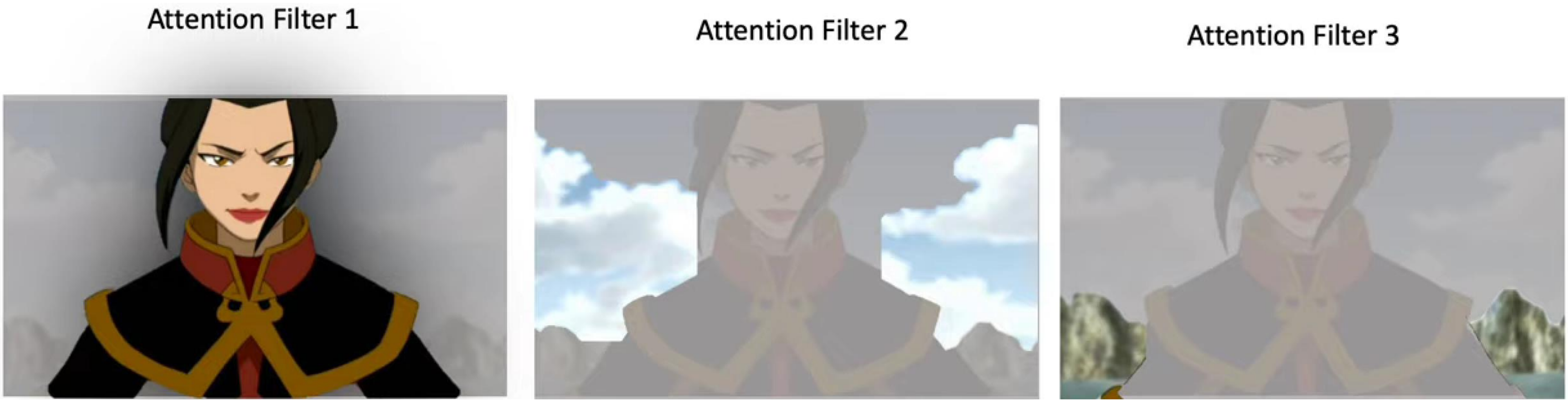
Attention Filter | Original Image | Filtered Image

*

=

[Visual guide to Transformers]

Visual Ts: self-attention intuition



[Visual guide to Transformers]

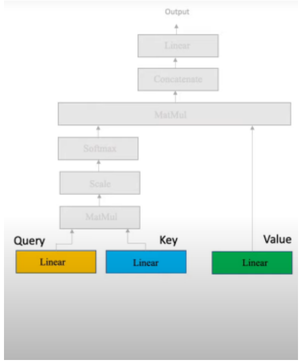
Why transformers (Ts) are the best at sequences processing?

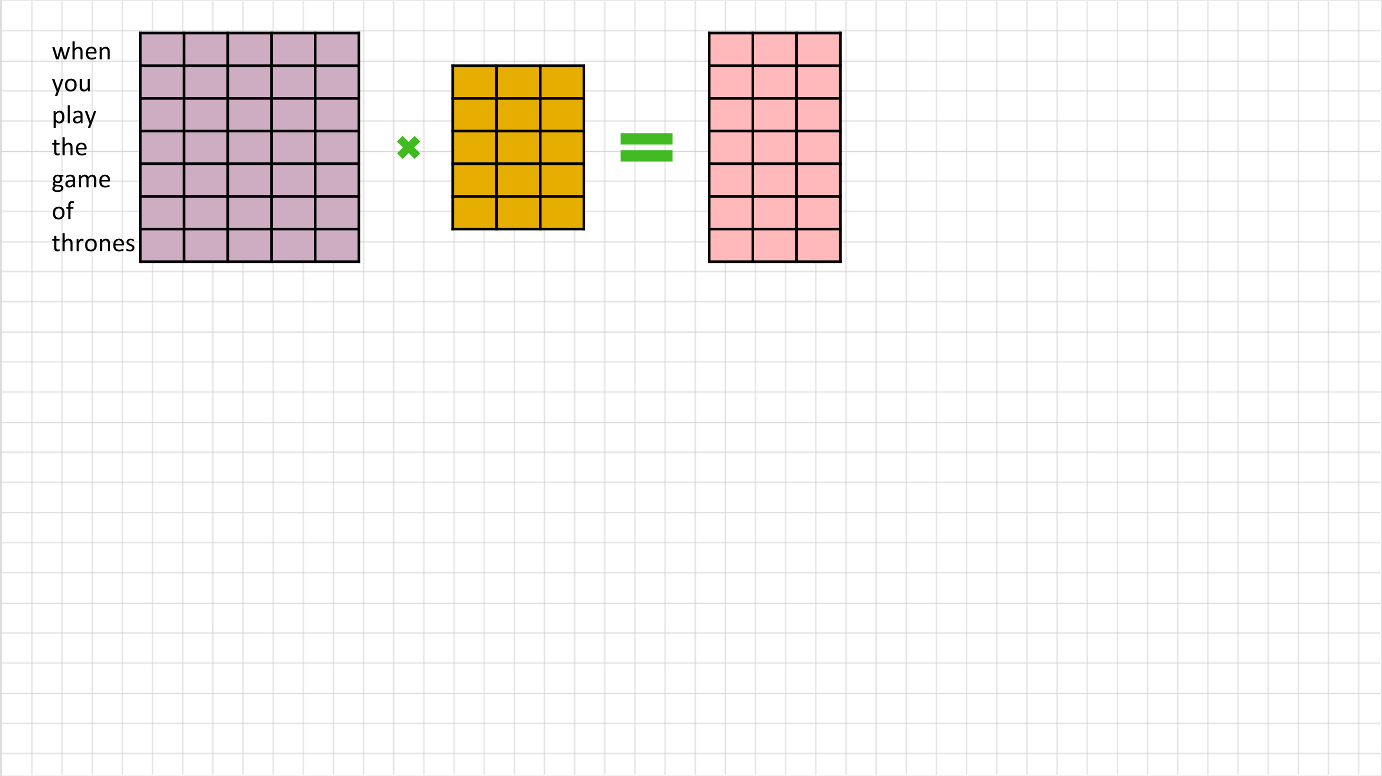
1. **Parallel Processing:** ransformers process entire sequences in parallel, unlike RNNs which process data sequentially. This parallelism significantly speeds up training and inference times.
2. **Attention Mechanism:** The core innovation of Transformers is the self-attention mechanism, which allows the model to weigh the importance of different elements in the input sequence when making predictions. This mechanism helps in capturing long-range dependencies more effectively.
3. **Handling Long-Range Dependencies:** Transformers can capture long-range dependencies more effectively than RNNs. The self-attention mechanism ensures that distant elements in the sequence can influence each other directly, without the need for intermediate steps. Unlike RNNs, which suffer from the vanishing gradient problem, Transformers do not have this issue due to their parallel processing and attention mechanism.
4. **Scalability:** Transformers can be scaled to handle very large datasets and complex tasks. The attention mechanism allows for the incorporation of more data without a significant increase in computational complexity.
5. **Versatility:** Transformers can be adapted for a wide range of sequence processing tasks, including natural language processing, speech recognition, and even computer vision (e.g., Vision Transformers).
6. **State-of-the-Art Performance**

DeepSeek-V2.5 speaks

Thank you for your attention!

a.kornaev@innopolis.ru, @avkornaev


Output
Linear
Concatenate
MatMul
Softmax
Scale
MatMul
Query
Key
Value
Linear
Linear
Linear


when
you
play
the
game
of
thrones