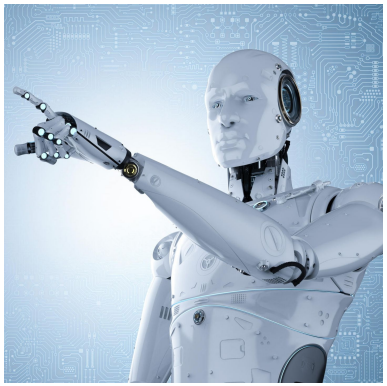
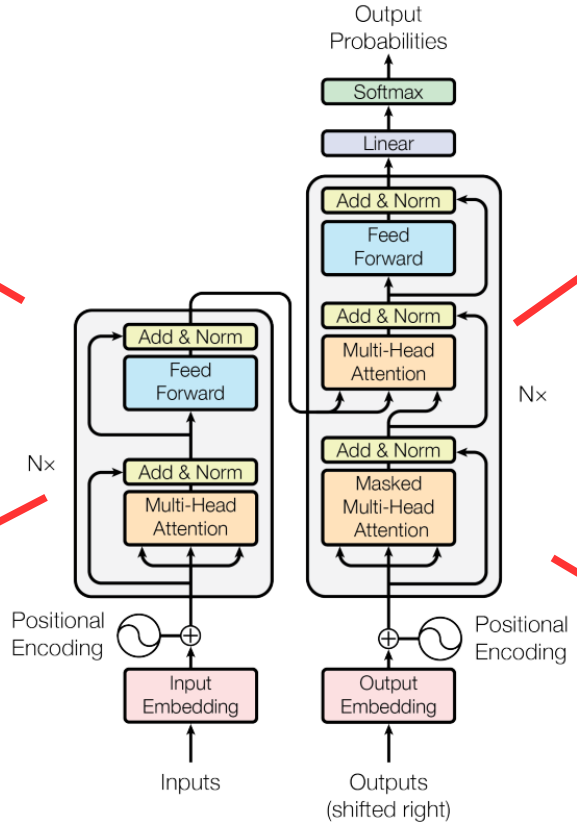
Image source:
https://www.wallpaperflare.com/optimus-prime-the-city-weapons-robots-the-movie-wallpaper-wbsf/download (call date: 20.07.22)

Computer Vision



Deep Reinforcement Learning



Natural Language Processing

Attention!
The Transformers are coming!

GANs

Tim Niklas Witte

Image sources:
https://machinelearningmastery.com/wp-content/uploads/2021/08/attention_research_1-727x1024.png
https://images.ctfassets.net/3viuren4us1n/1Ghw96A2tcYRfRezOwtmjx/e646778f3f53e50ea3e857e9cdb23120/Computer_vision.jpg
https://image.stern.de/7931130/t/NJ/v2/w1440/r1/-/roboter.jpg
https://www.studienscout-nl.de/fileadmin/_processed_/csm_Sprache-und-Sprachkurse_08f53ae680.jpg
https://studip.uni-osnabrueck.de/sendfile.php?type=0&file_id=8a554d27d543b1ce06eb1a68a34e22ae&file_name=vqgan_4_i350.png
(call dates: 20.07.22)

Structure

1. Motivation

2. Intuitive understanding of self-attention
3. Intuitive understanding of transformer architecture

4. Natural Language Processing
5. Computer Vision
6. GANs
7. Deep Reinforcement Learning

7. Modifications of attention: Synthesizer, Linformer, Reformer

8. Conclusions

Discussion: Candy-wise attention

Image sources:
https://www.flaticon.com/free-icon/candy-jar_10751352?related_id=10751352&origin=tag
https://emojipedia.org/de/toss-face/march-2022/gesicht-mit-umarmenden-h%C3%A4nden/
(call dates: 20.07.22)

1. Motivation

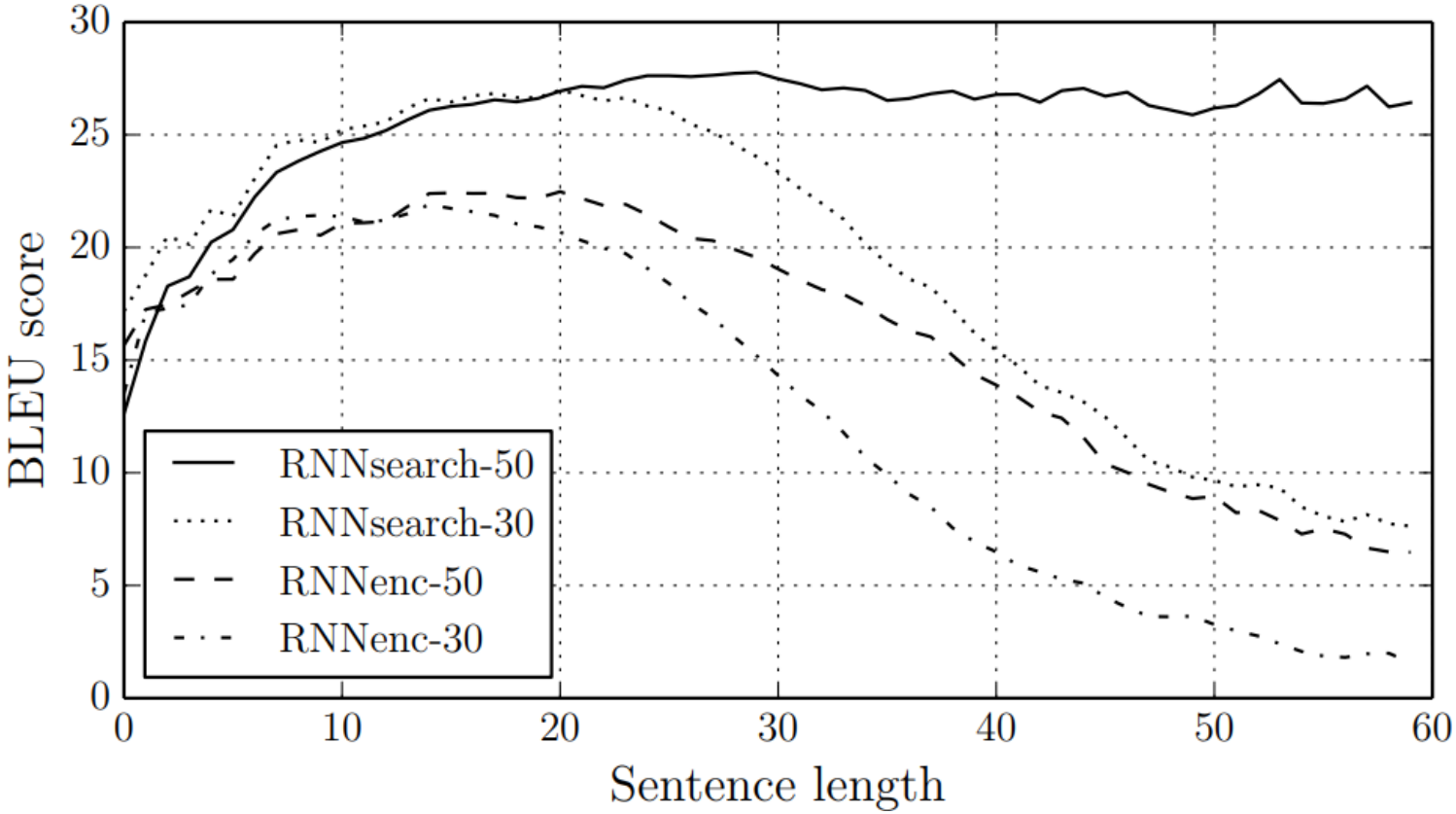
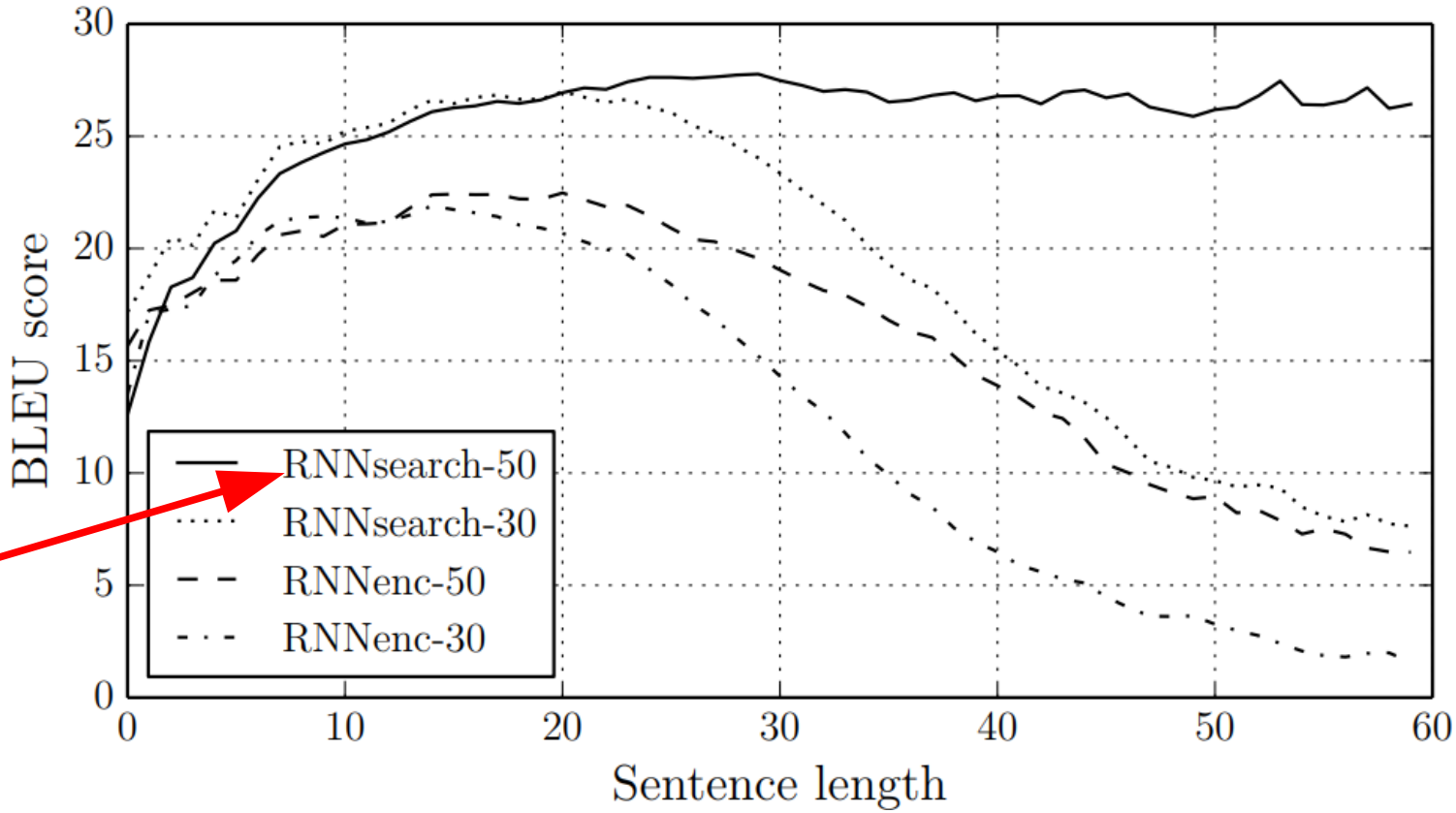


Figure 1: Performance (BLEU score) differences between RNN-based models for neural machine translation [1].

1. Motivation



Bahdanau attention

Figure 1: Performance (BLEU score) differences between RNN-based models for neural machine translation [1].

2. Intuitive understanding of self-attention

Image sources:
https://www.campussafetymagazine.com/wp-content/uploads/2019/03/Bull-horn-announcement-PA-iStock-1000x500.jpg
https://upload.wikimedia.org/wikipedia/commons/thumb/7/7b/Attention_Sign.svg/2302px-Attention_Sign.svg.png
https://upload.wikimedia.org/wikipedia/commons/thumb/9/9a/Simple_Attention.svg/1024px-Simple_Attention.svg.png
(call dates: 20.07.22)

2. Intuitive understanding of self-attention

The server ...

2. Intuitive understanding of self-attention

The server ...

?

Figure 2: A Waiter.

Figure 3: Server in a data center.

Image sources:
https://content.gastivo.de/wp-content/uploads/2019/09/Der-Kellner-Knigge-Perfektes-Servieren-1024x576.jpg
https://i0.wp.com/blog.fyipe.com/wp-content/uploads/2020/01/servers-e1605085513709.jpeg
(call dates: 20.07.22)

2. Intuitive understanding of self-attention

The server is crashed.

?

Figure 2: A Waiter.

Figure 3: Server in a data center.

Image sources:
https://content.gastivo.de/wp-content/uploads/2019/09/Der-Kellner-Knigge-Perfektes-Servieren-1024x576.jpg
https://i0.wp.com/blog.fyipe.com/wp-content/uploads/2020/01/servers-e1605085513709.jpeg
(call dates: 20.07.22)

2. Intuitive understanding of self-attention

The server is crashed.

Figure 2: A Waiter.

Figure 3: Server in a data center.

Image sources:
https://content.gastivo.de/wp-content/uploads/2019/09/Der-Kellner-Knigge-Perfektes-Servieren-1024x576.jpg
https://i0.wp.com/blog.fyipe.com/wp-content/uploads/2020/01/servers-e1605085513709.jpeg
(call dates: 20.07.22)

2. Intuitive understanding of self-attention

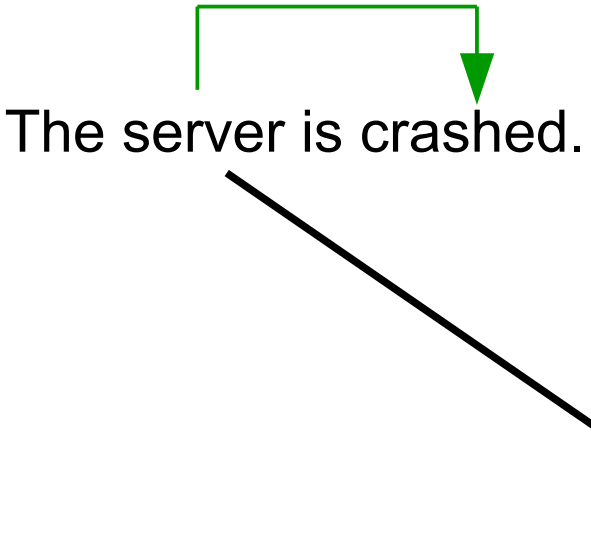


Figure 2: A Waiter.

Figure 3: Server in a data center.

Image sources:
https://content.gastivo.de/wp-content/uploads/2019/09/Der-Kellner-Knigge-Perfektes-Servieren-1024x576.jpg
https://i0.wp.com/blog.fyipe.com/wp-content/uploads/2020/01/servers-e1605085513709.jpeg
(call dates: 20.07.22)

2. Intuitive understanding of self-attention

English → **Portuguese**

This is the first book I did. → Este e o primerio livro que eu fiz.

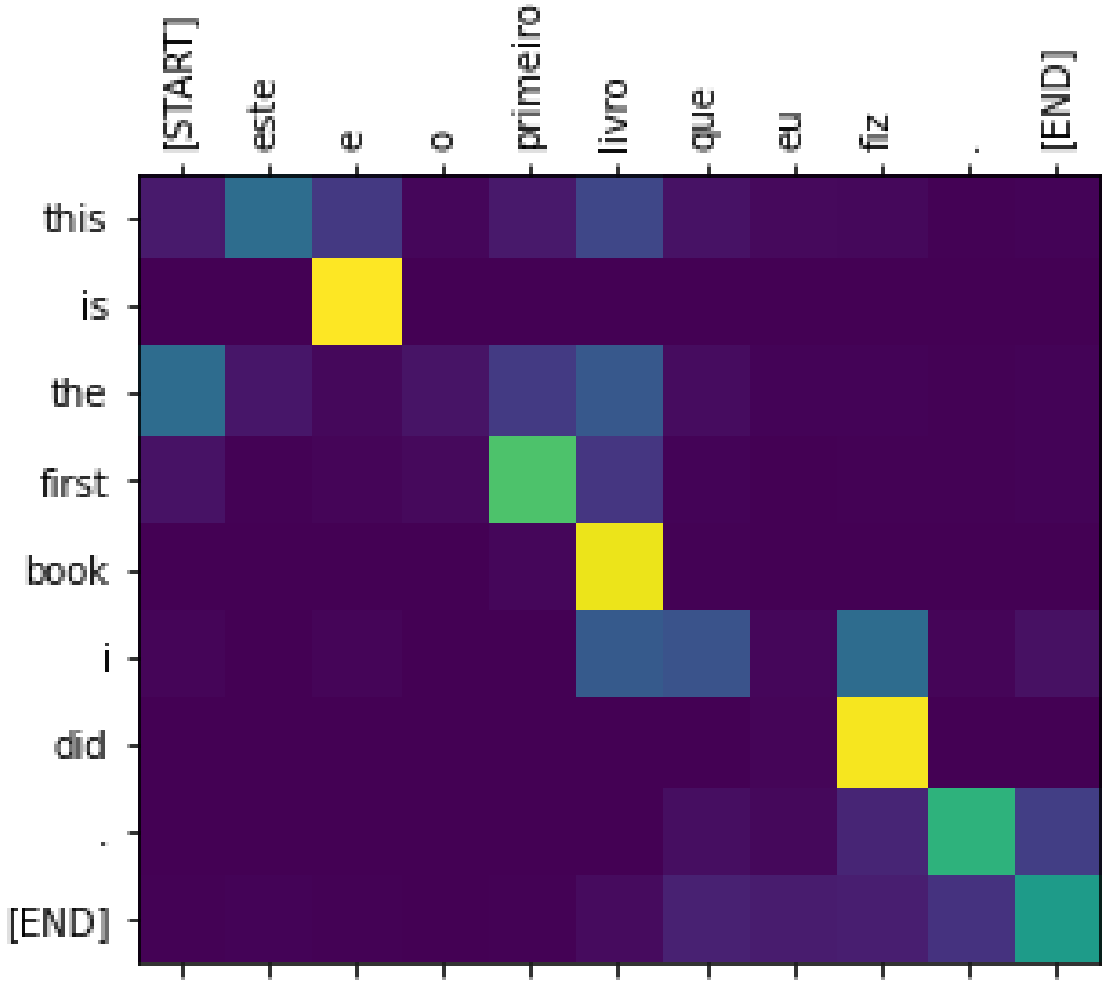


Figure 4: Self-attention heatmap.

Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

2. Intuitive understanding of self-attention

English → **Portuguese**

This is the first book I did. → Este e o primerio livro que eu fiz.

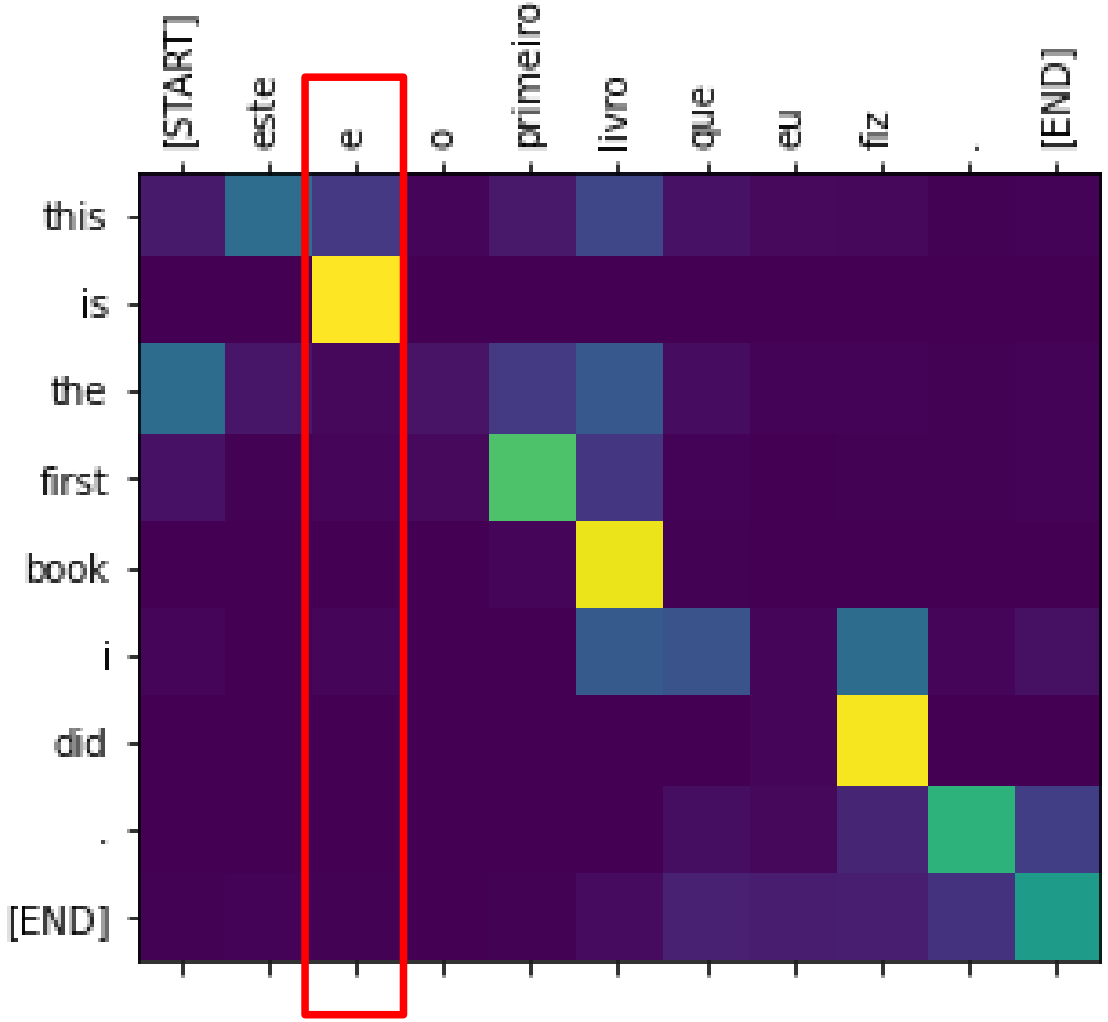


Figure 4: Self-attention heatmap.

Image source: https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png (call date: 20.07.22)

2. Intuitive understanding of self-attention

English → **Portuguese**

This is the first book I did. → Este e o primerio livro que eu fiz.

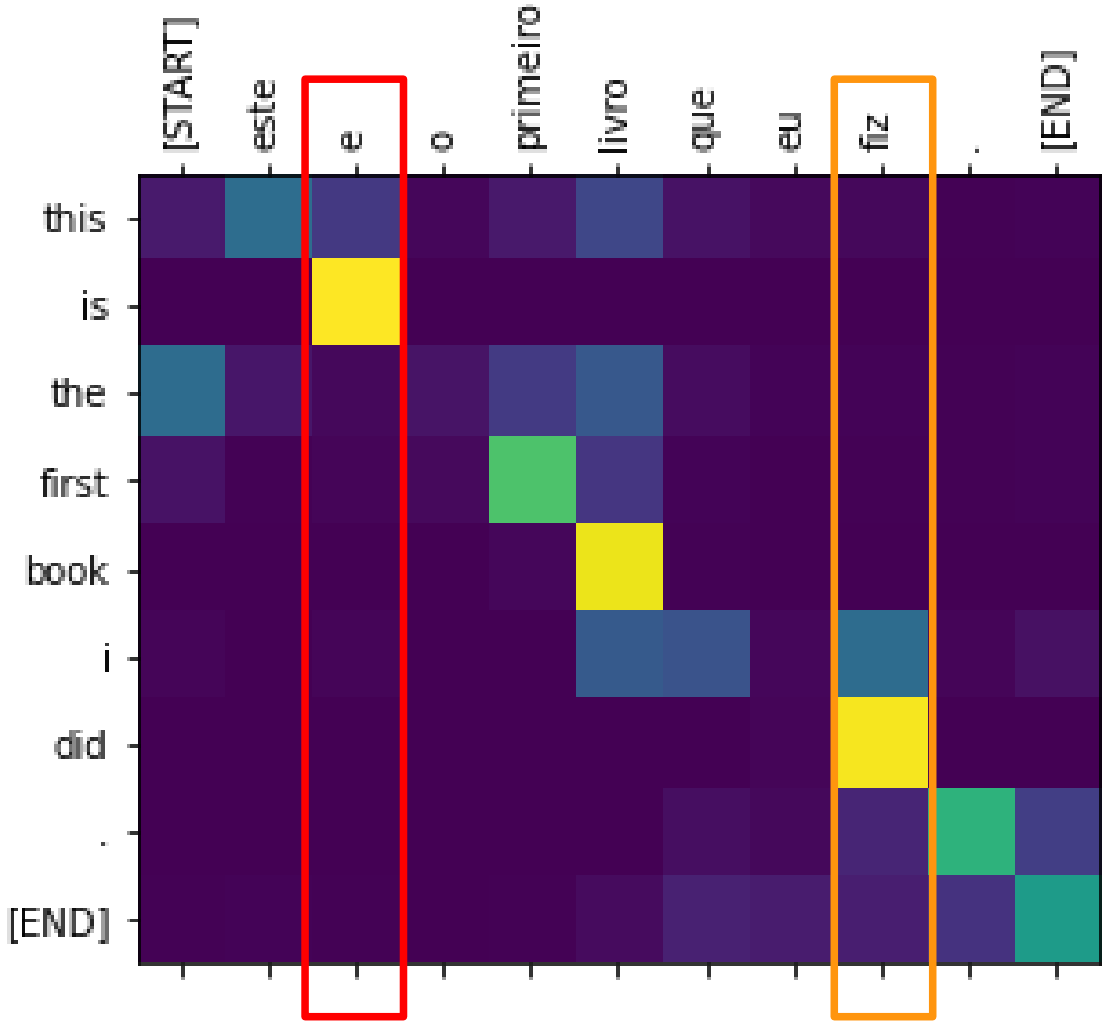
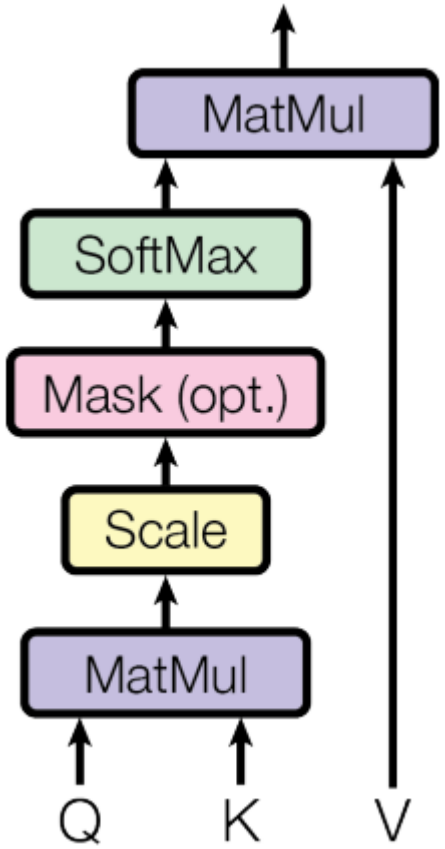


Figure 4: Self-attention heatmap.

Image source: https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png (call date: 20.07.22)

2. Intuitive understanding of self-attention



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

Q: Query matrix shape: d_k x D_H
K: Key matrix shape: d_k x D_H Hidden size
V: Value matrix shape: d_v x D_H

Figure 5: Self-attention [2].

2. Intuitive understanding of self-attention

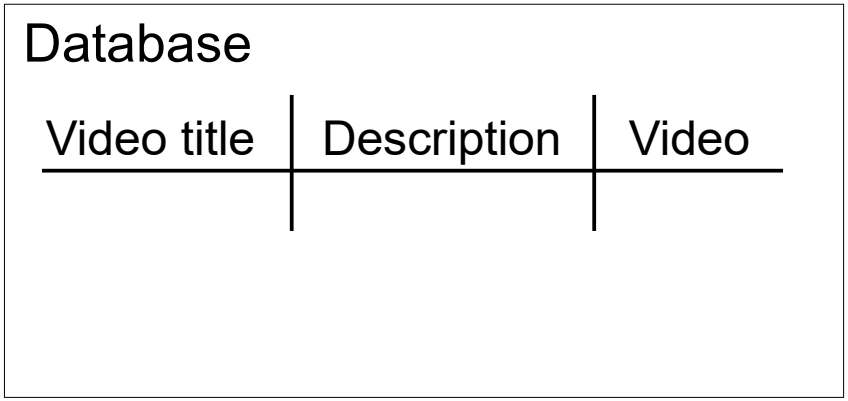


Figure 6: Self-attention example based on a YouTube search query.

2. Intuitive understanding of self-attention

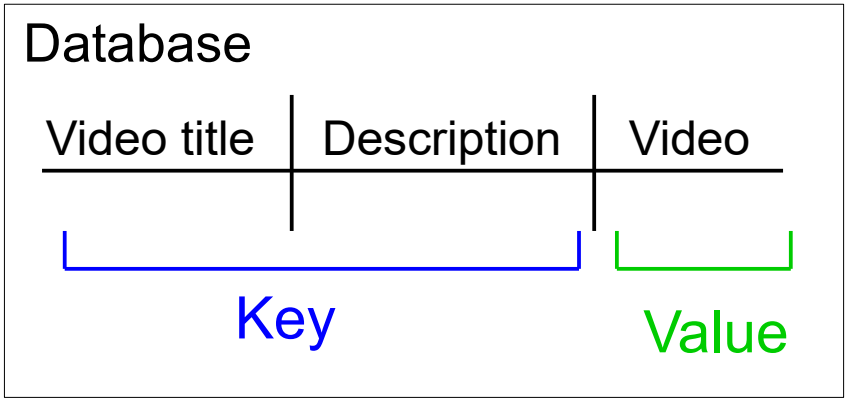


Figure 6: Self-attention example based on a YouTube search query.

2. Intuitive understanding of self-attention

Database

Video title	Description	Video

Key

Value

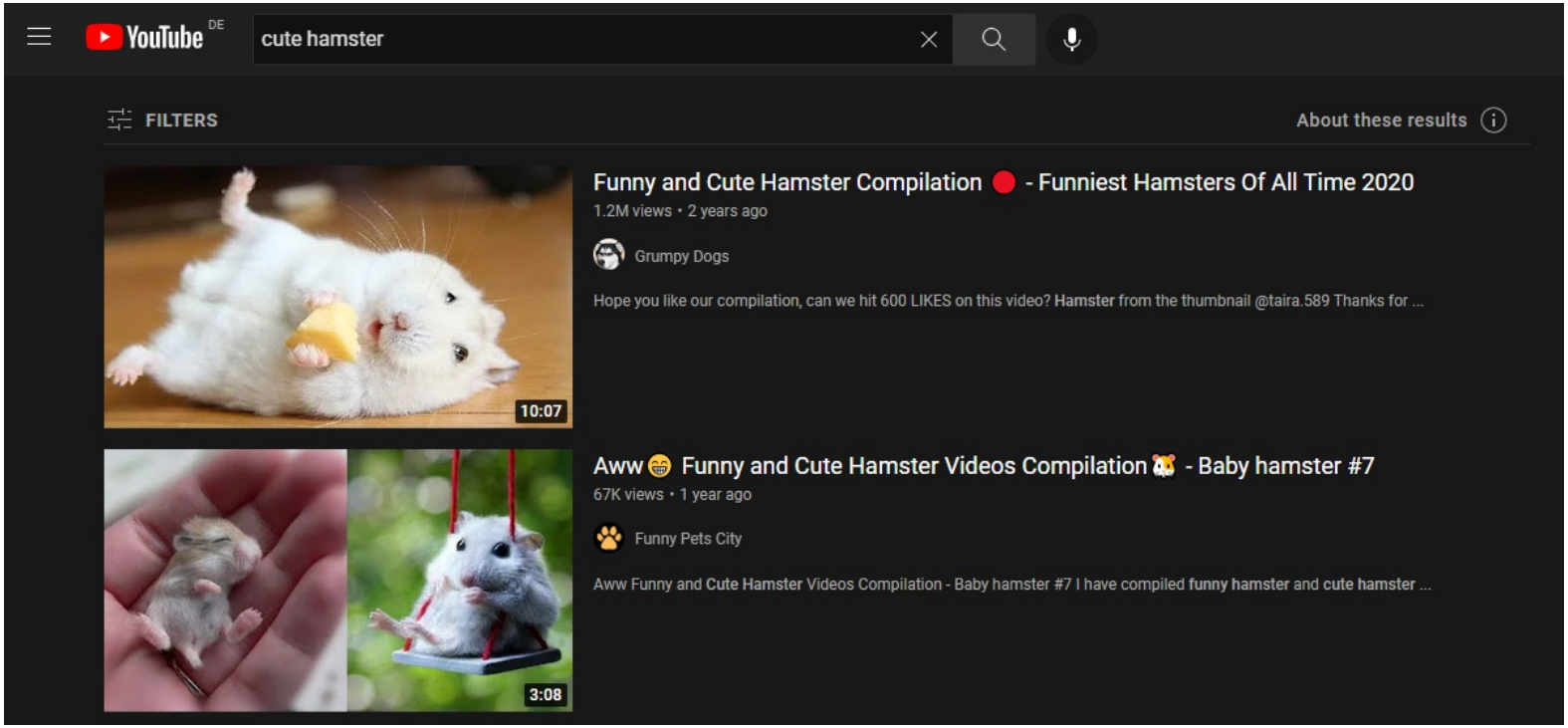
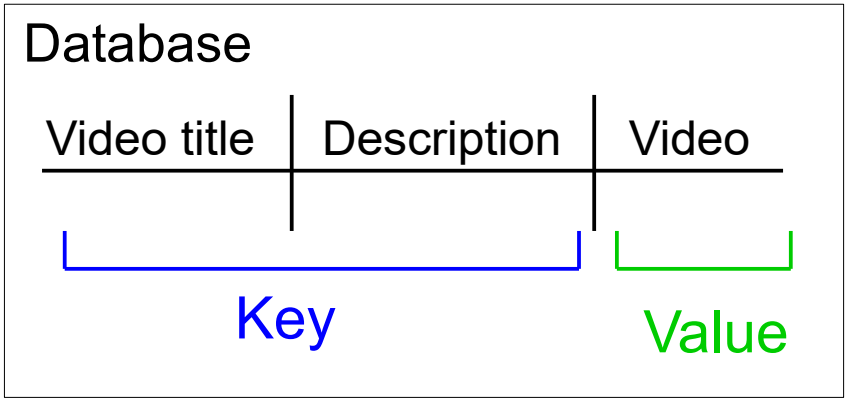


Figure 6: Self-attention example based on a YouTube search query.

2. Intuitive understanding of self-attention



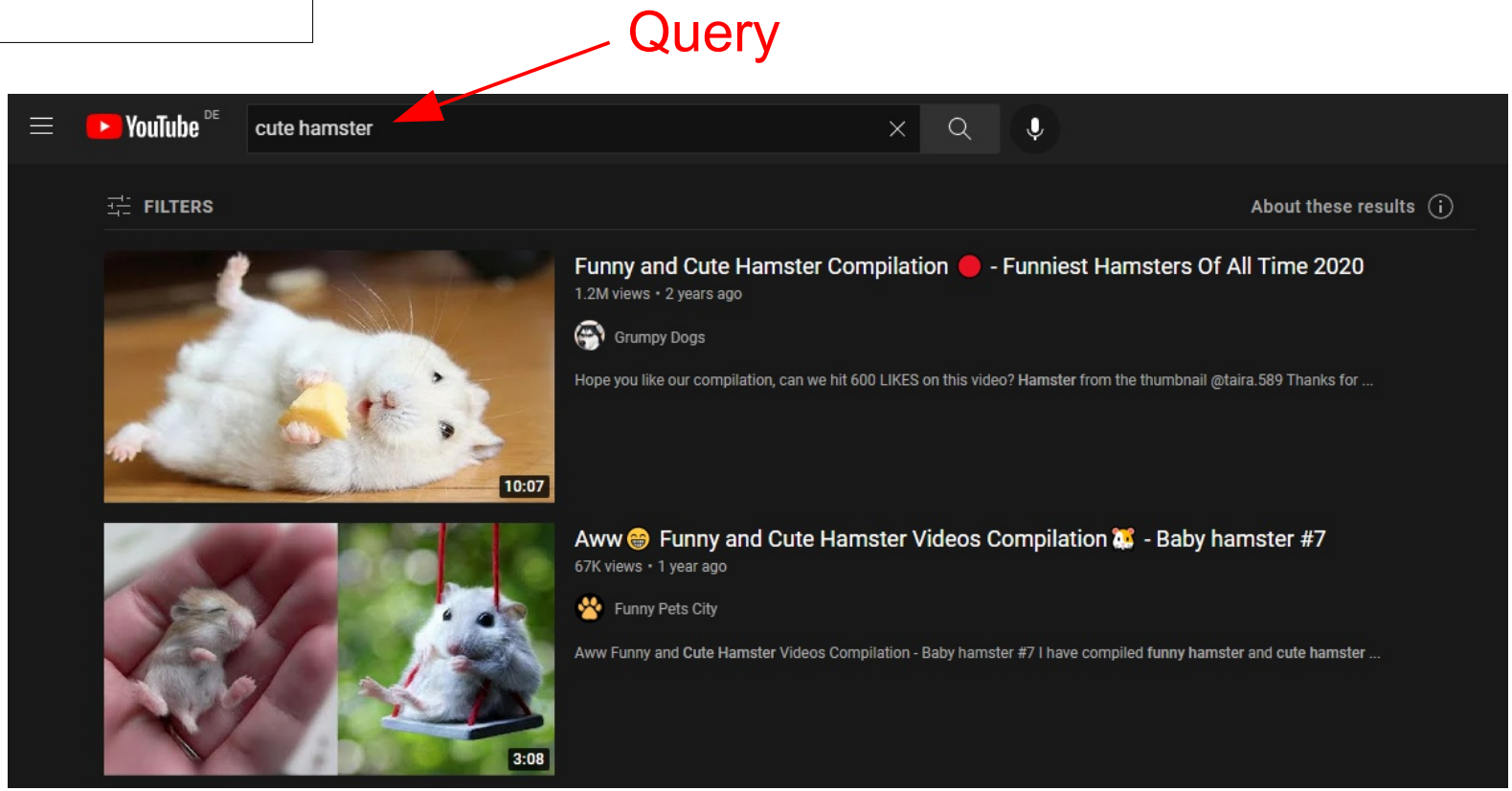


Figure 6: Self-attention example based on a YouTube search query.

2. Intuitive understanding of self-attention

Database

Video title	Description	Video

Key

Value

Query

$$\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

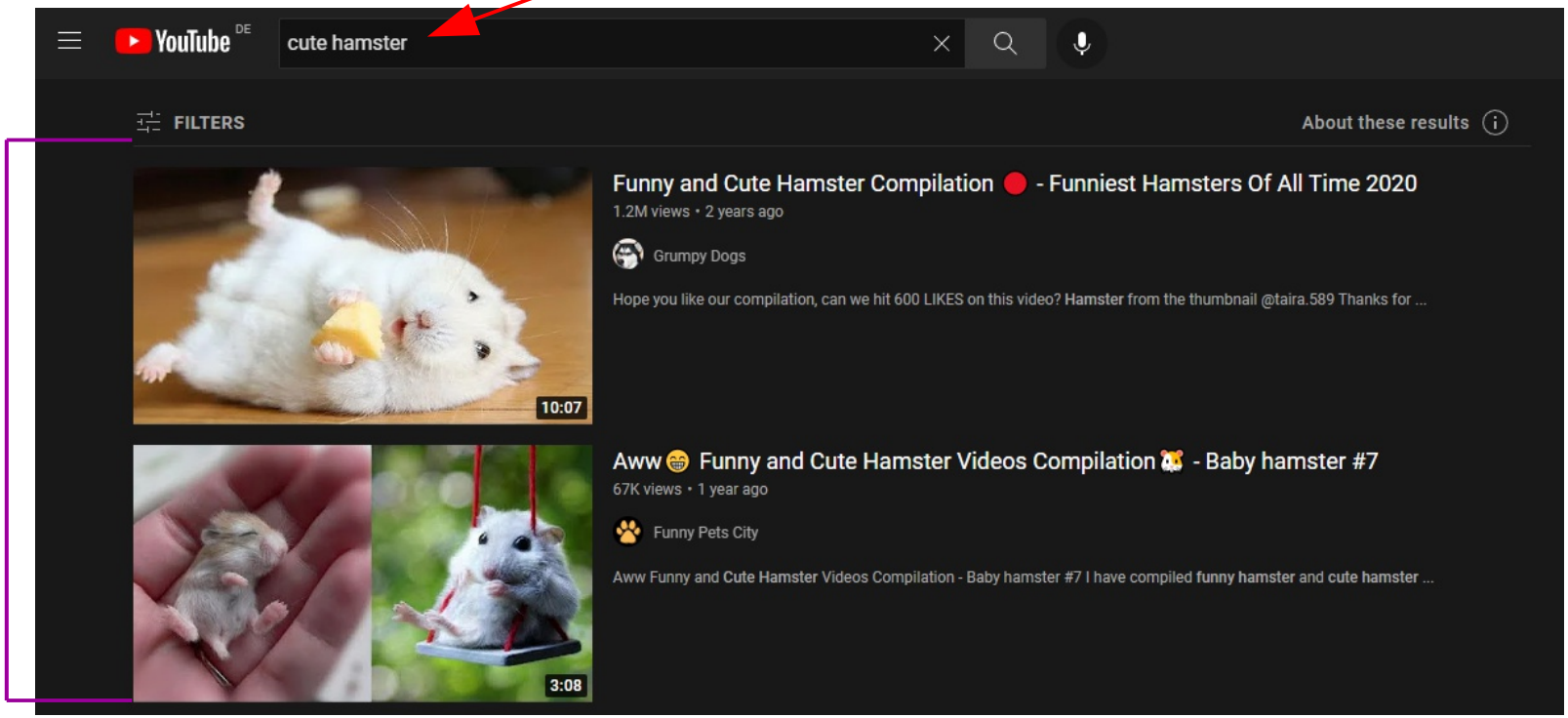


Figure 6: Self-attention example based on a YouTube search query.

2. Intuitive understanding of self-attention

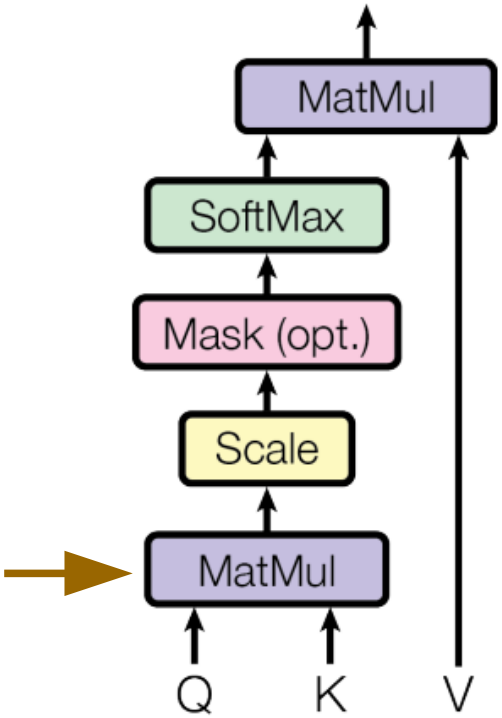


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention

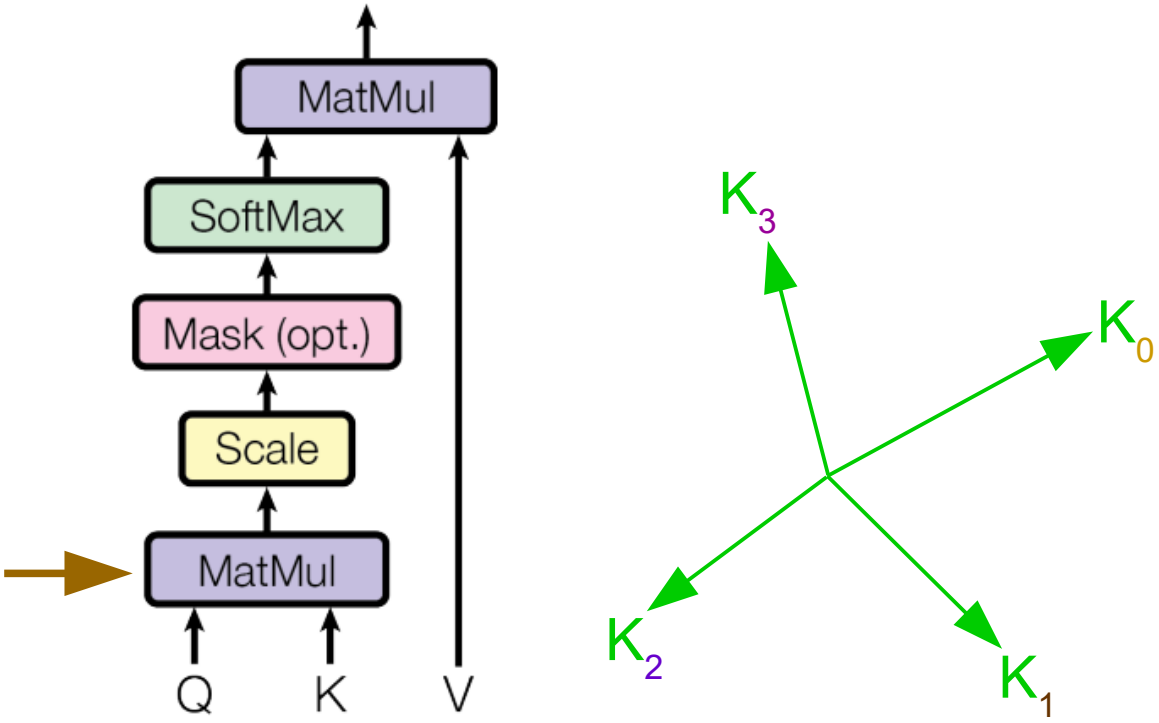


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention

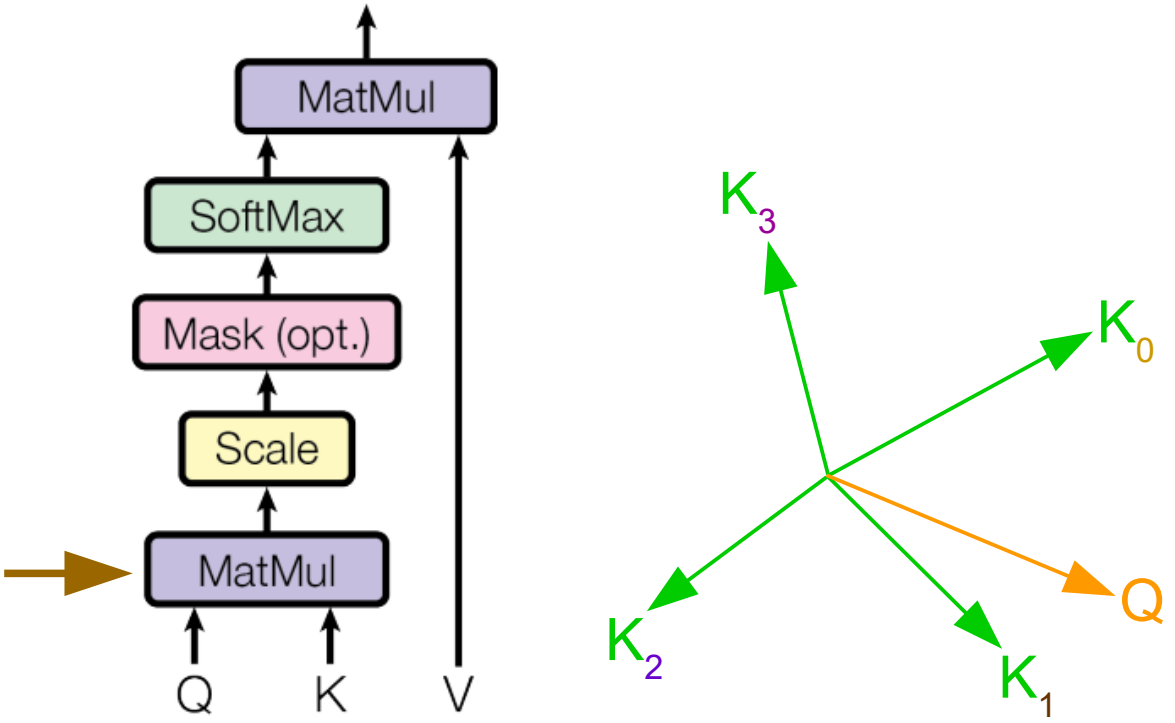
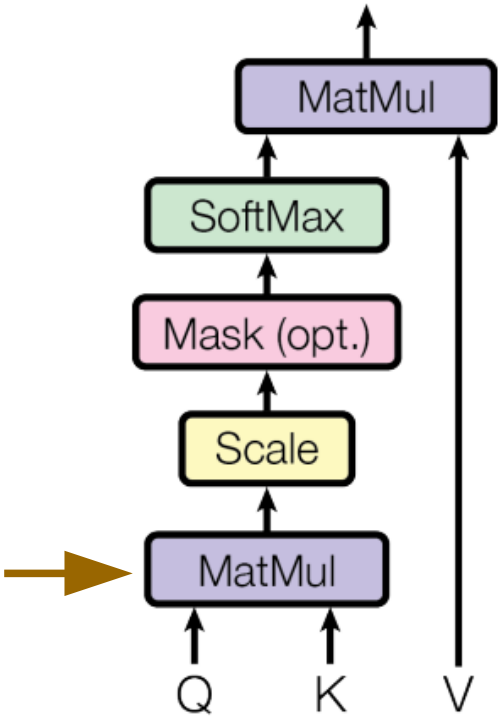
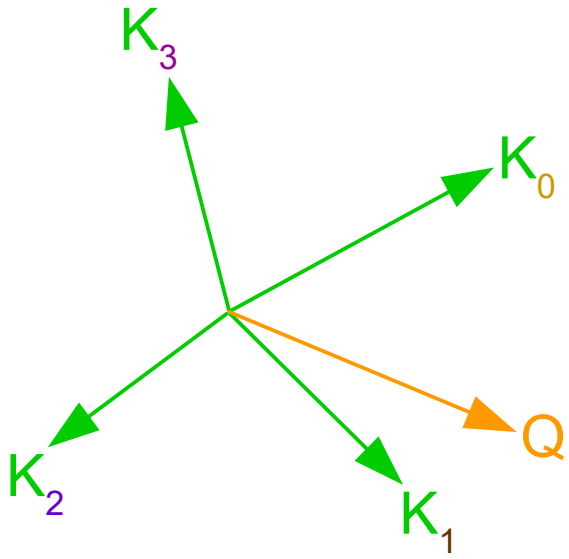


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention





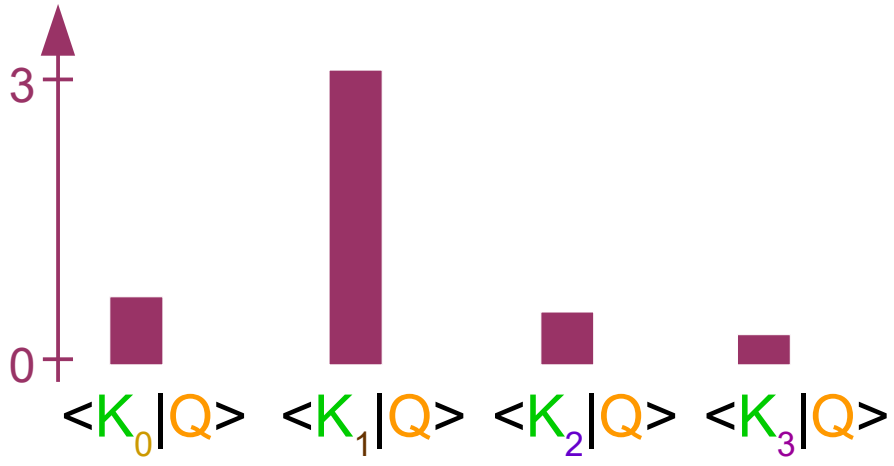


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention

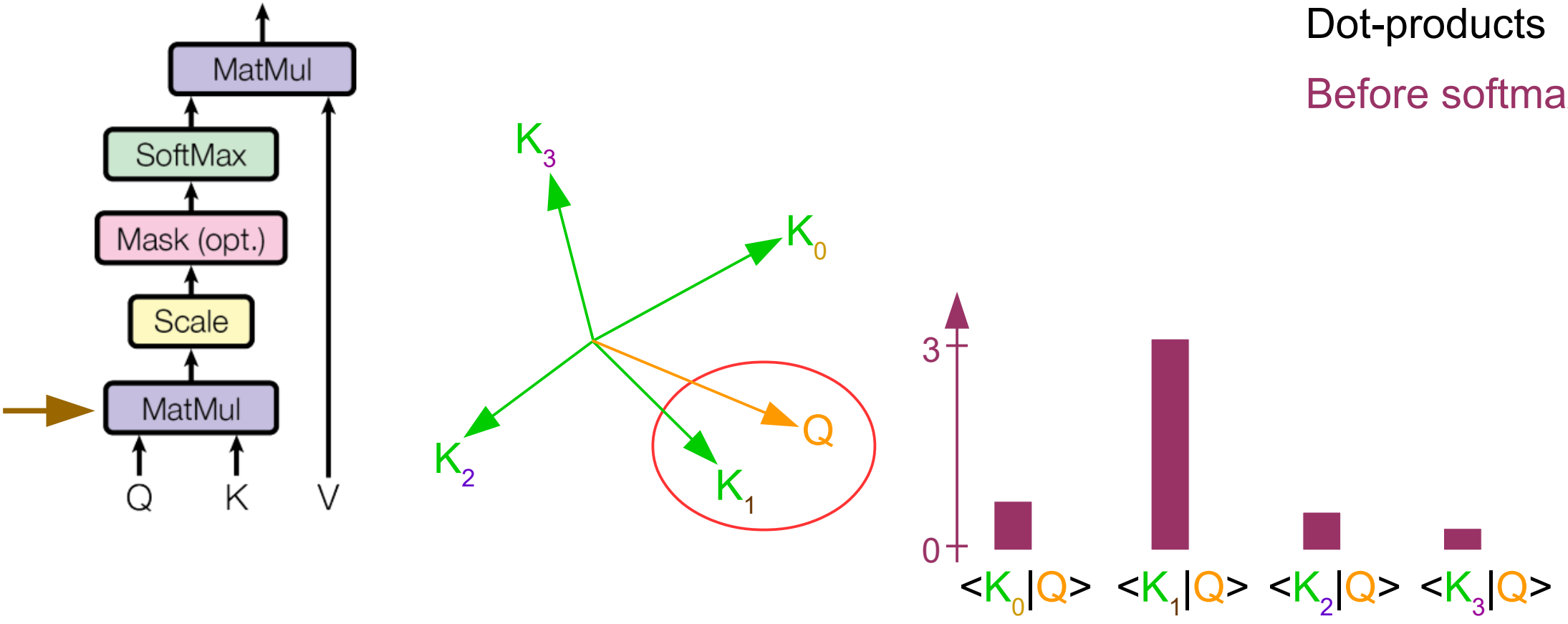


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention

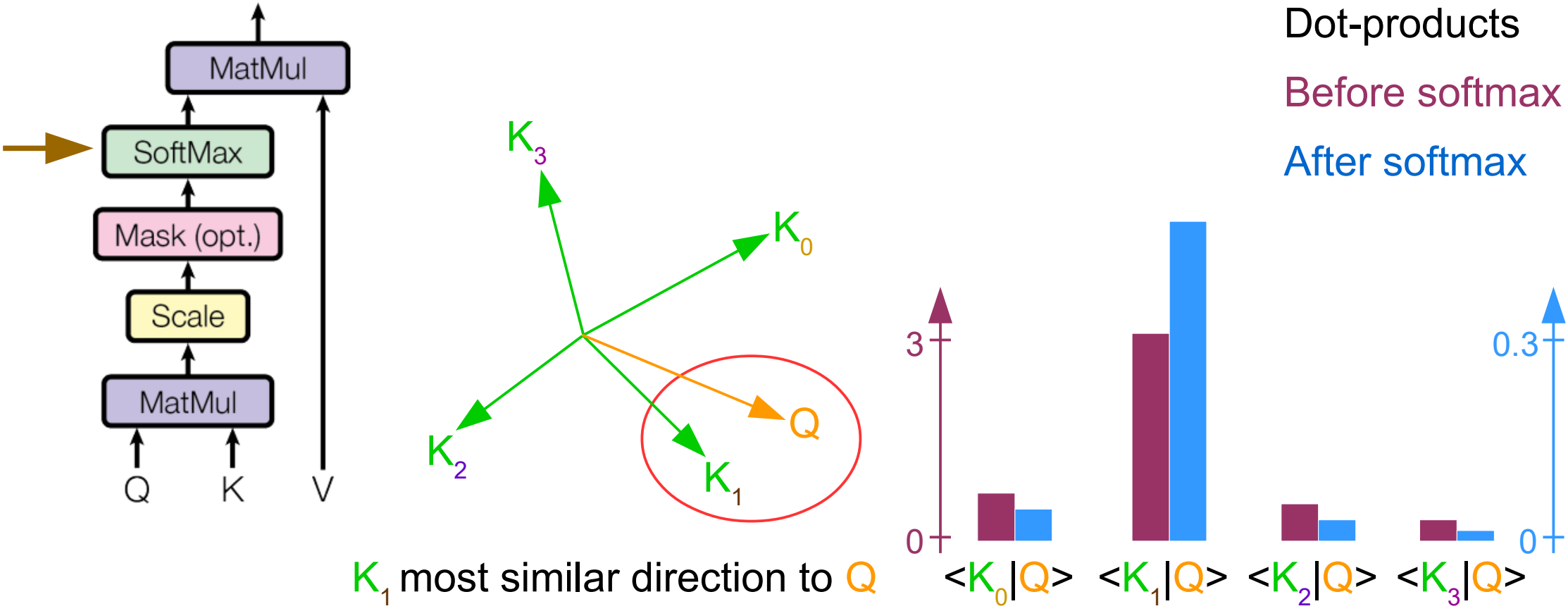


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention

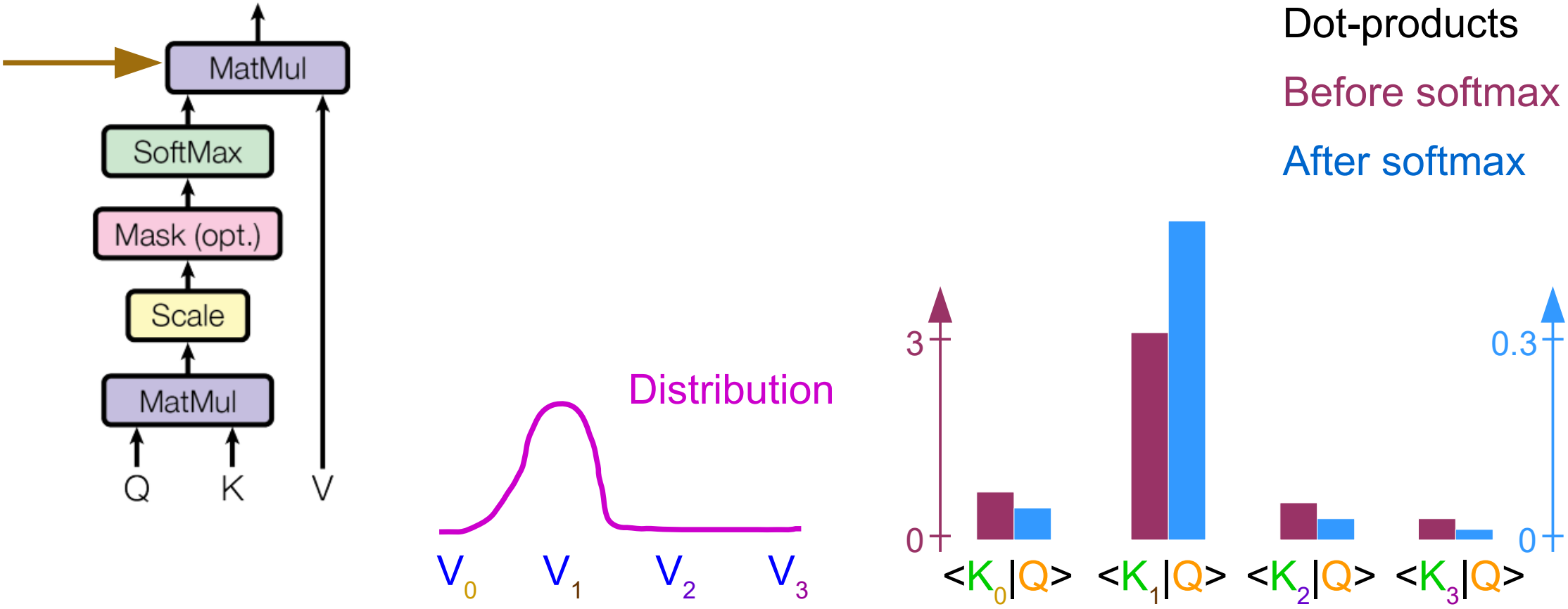


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention

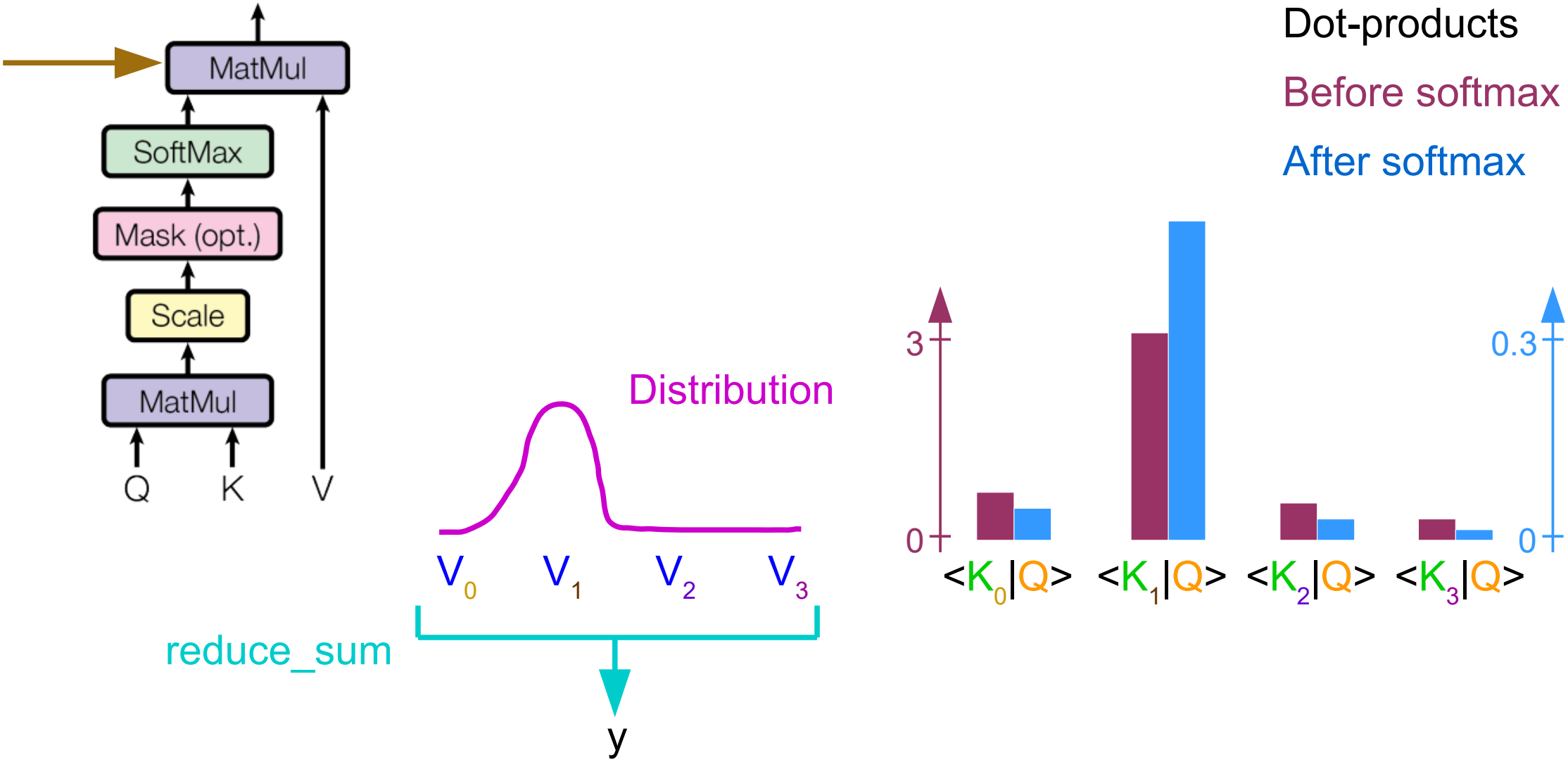
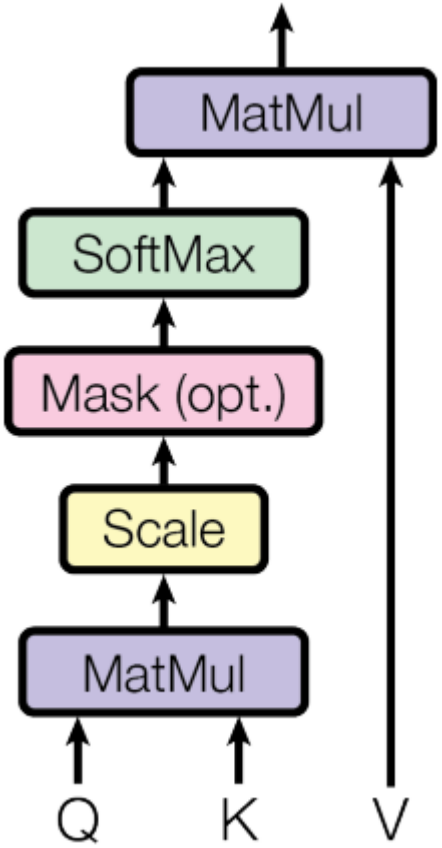
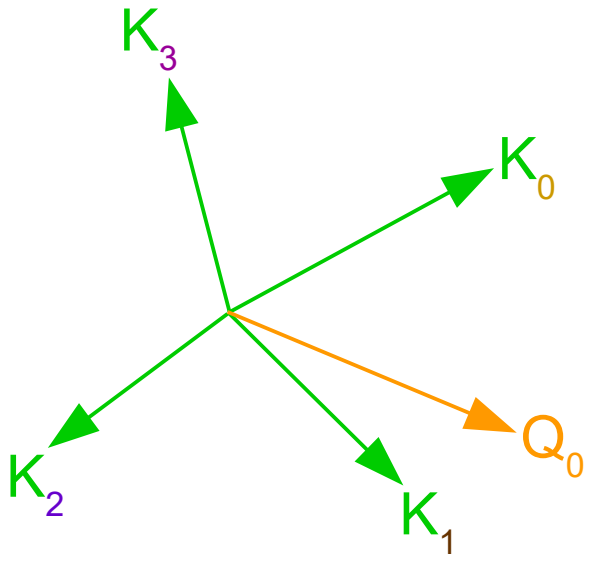


Figure 7: Self-attention visualised [2].

2. Intuitive understanding of self-attention





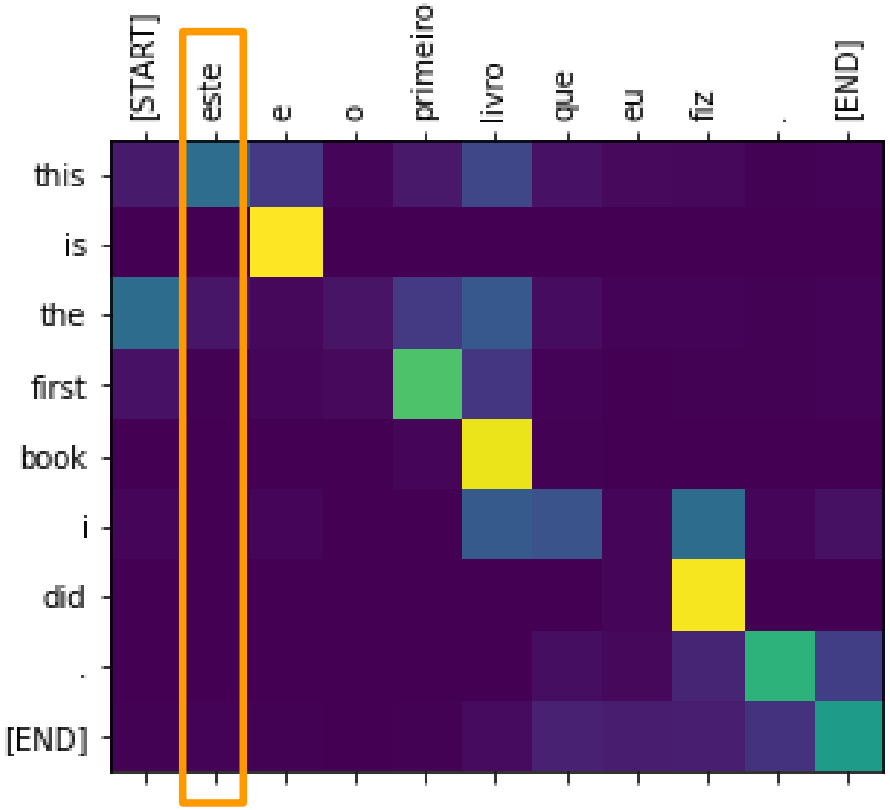


Figure 8: Self-attention – origin of the heatmap [2].

Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

2. Intuitive understanding of self-attention

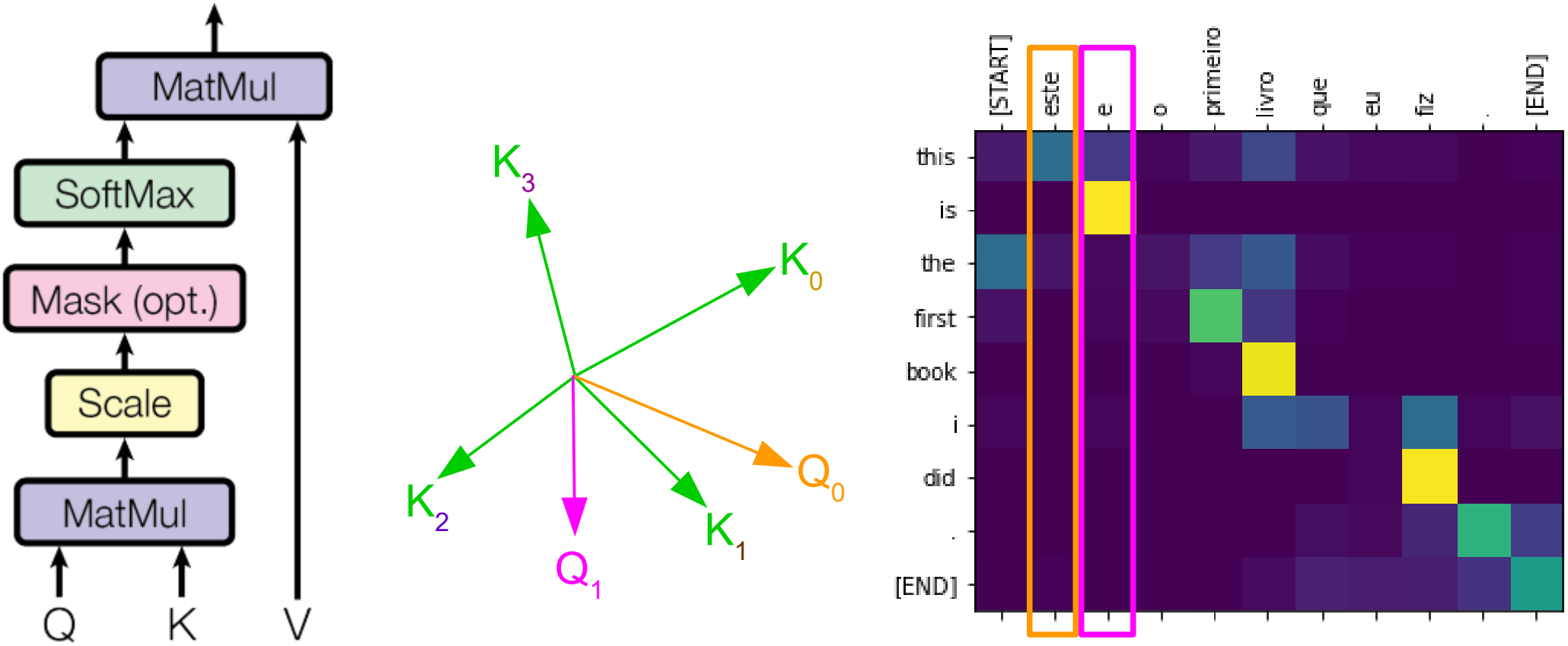
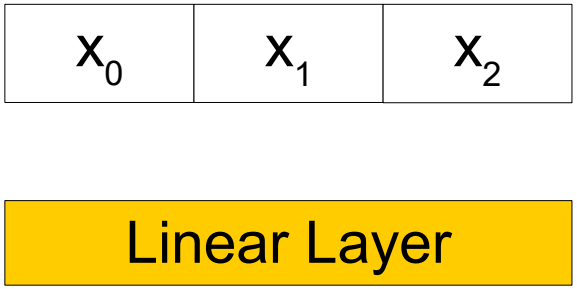
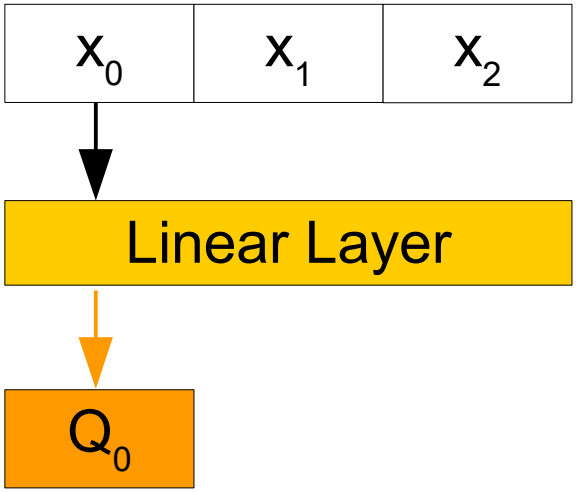


Figure 8: Self-attention – origin of the heatmap [2].

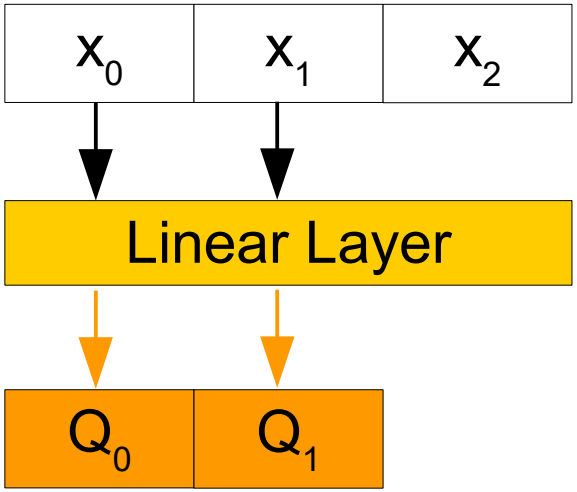
2. Intuitive understanding of self-attention



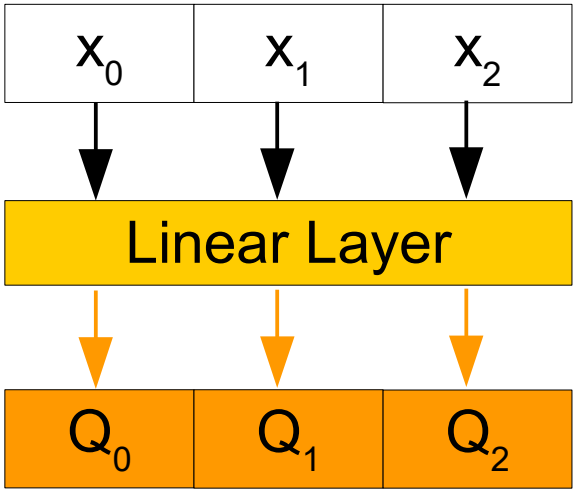
2. Intuitive understanding of self-attention



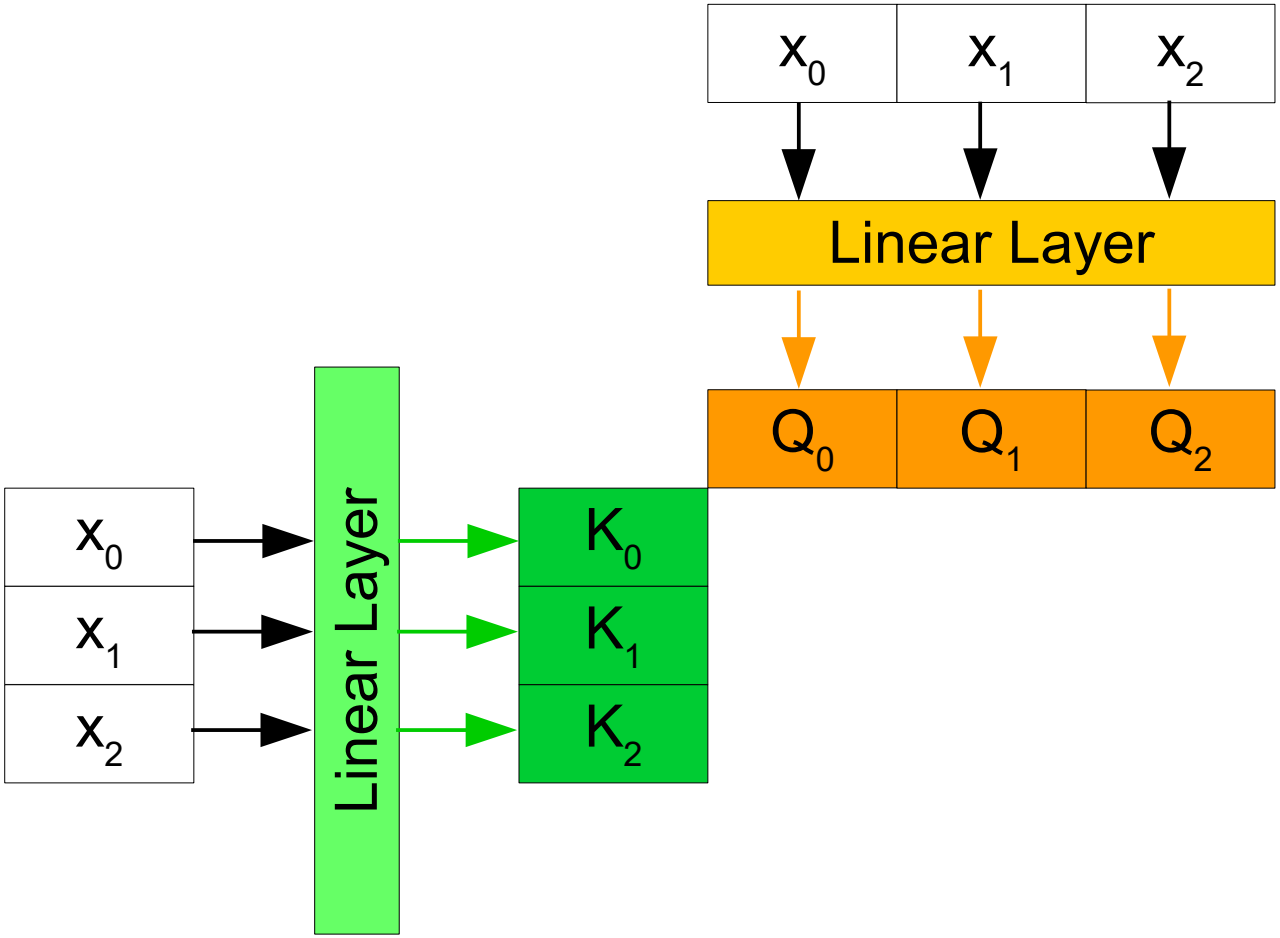
2. Intuitive understanding of self-attention



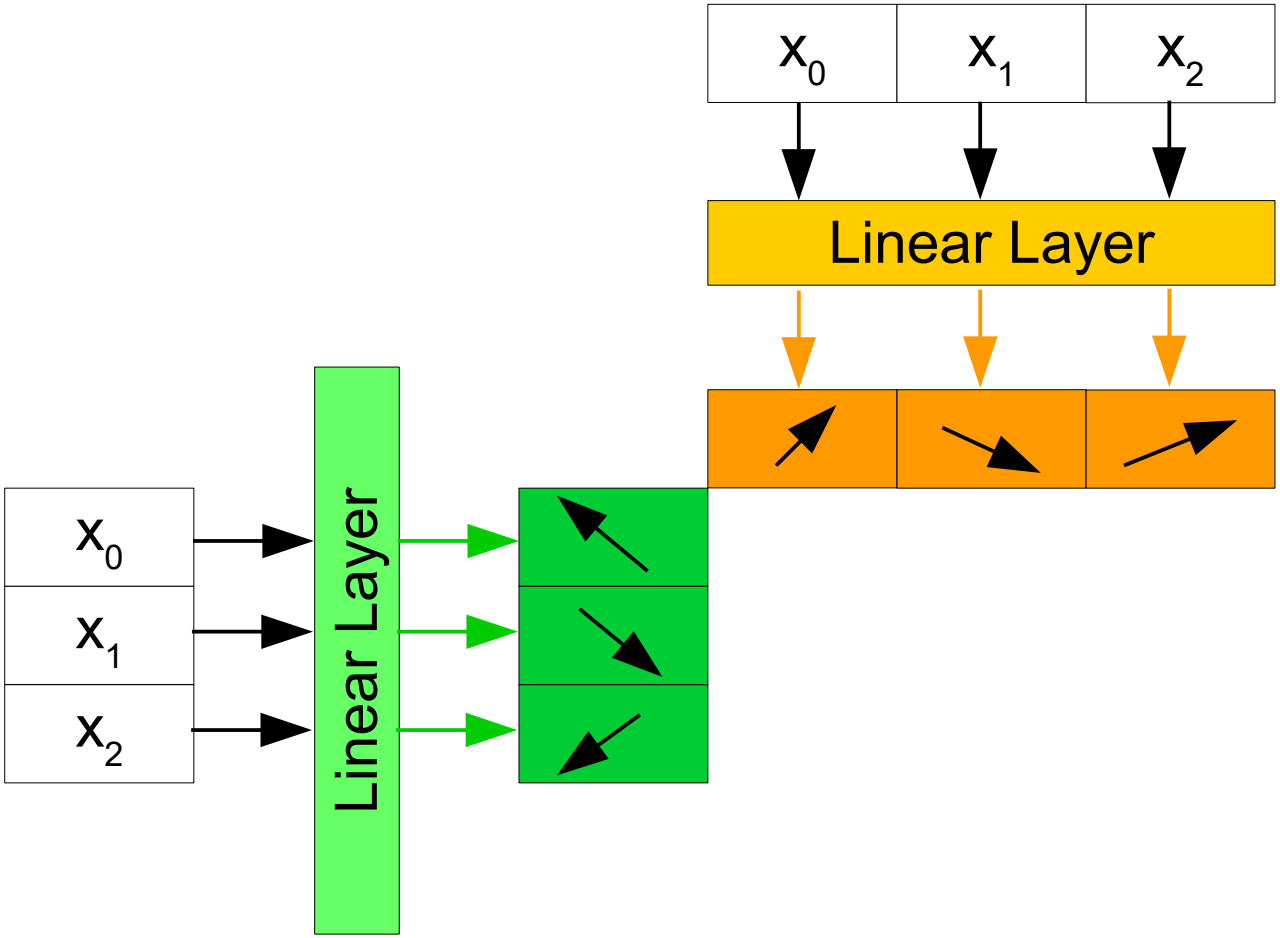
2. Intuitive understanding of self-attention



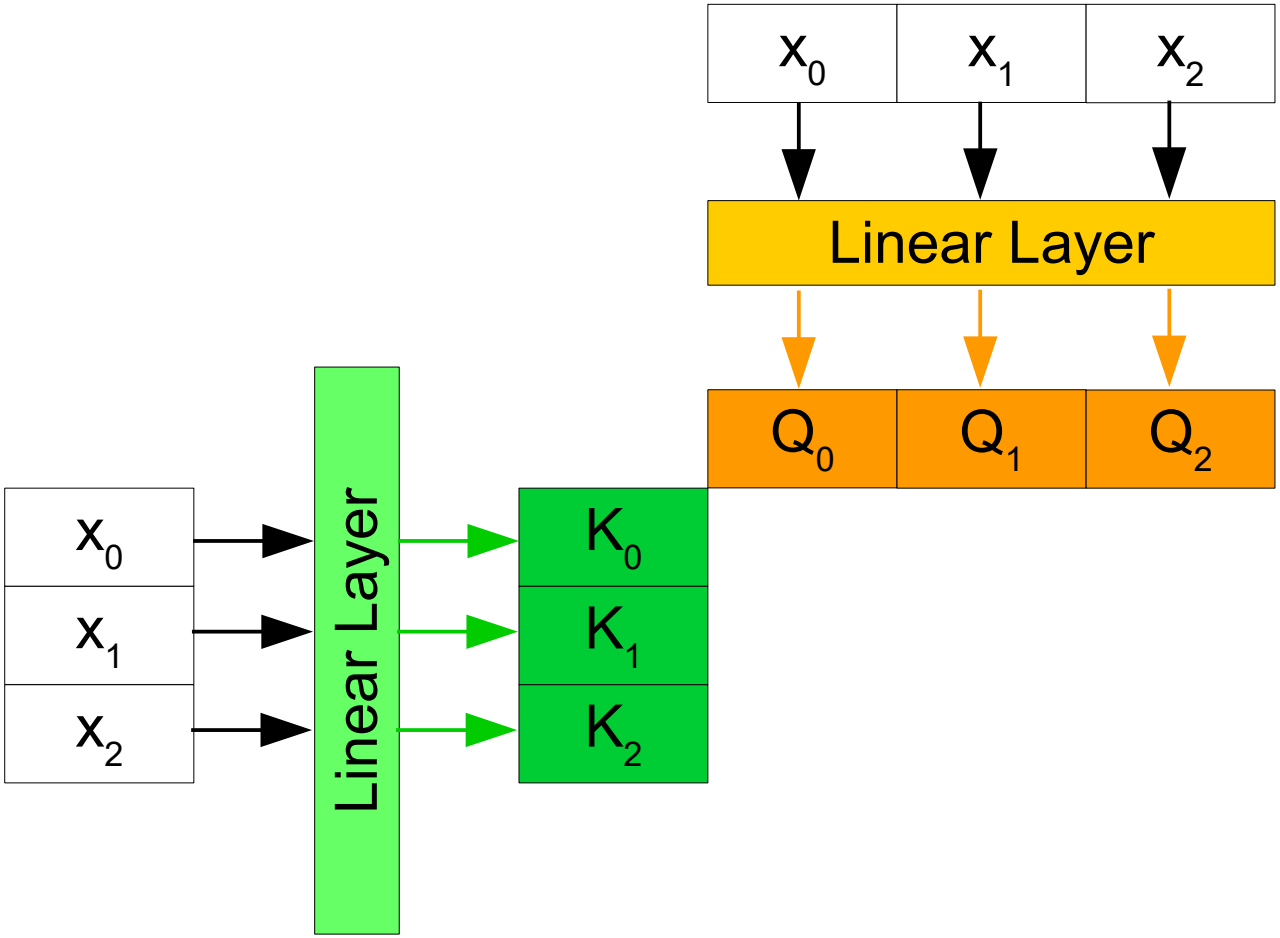
2. Intuitive understanding of self-attention



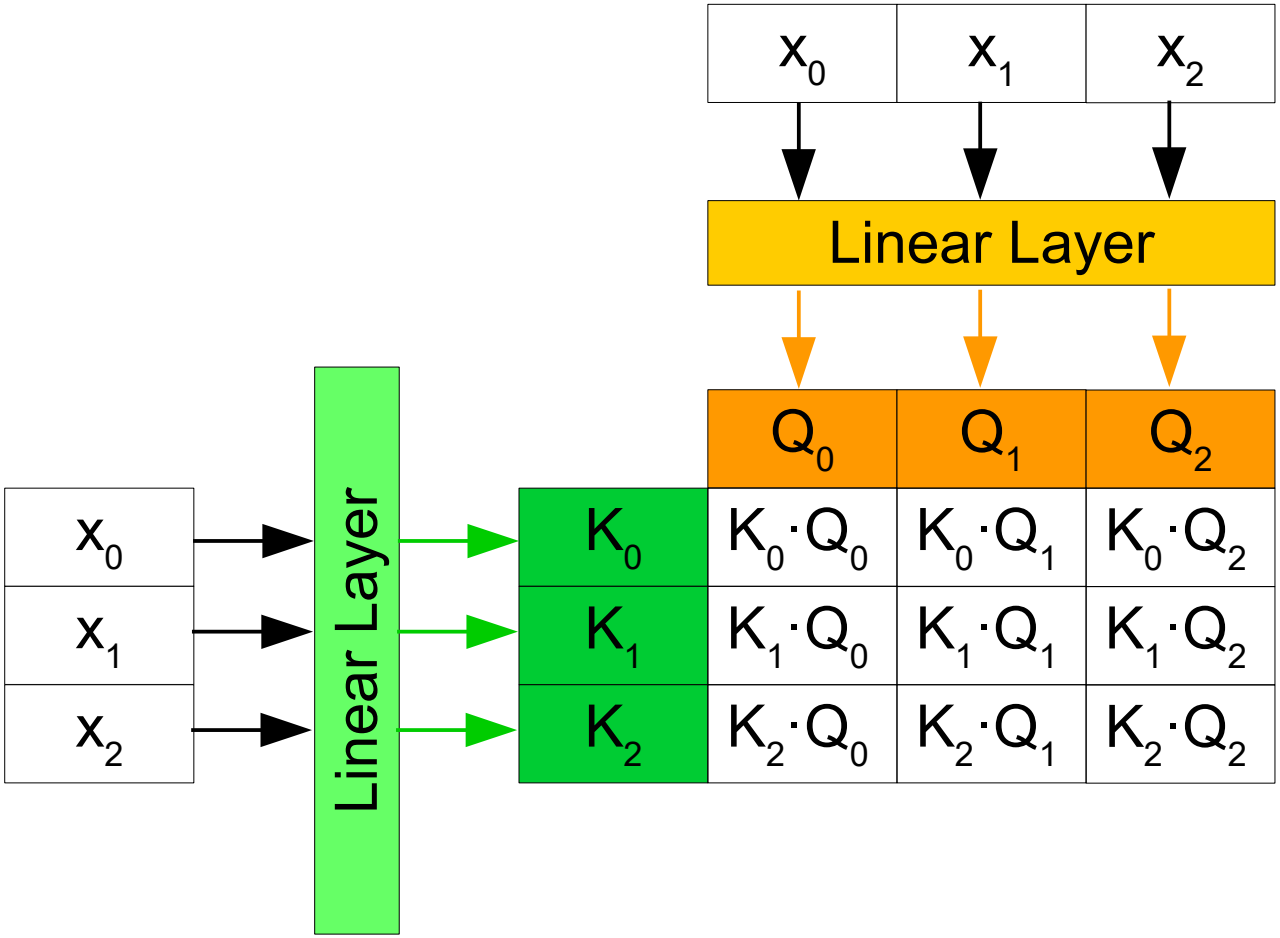
2. Intuitive understanding of self-attention



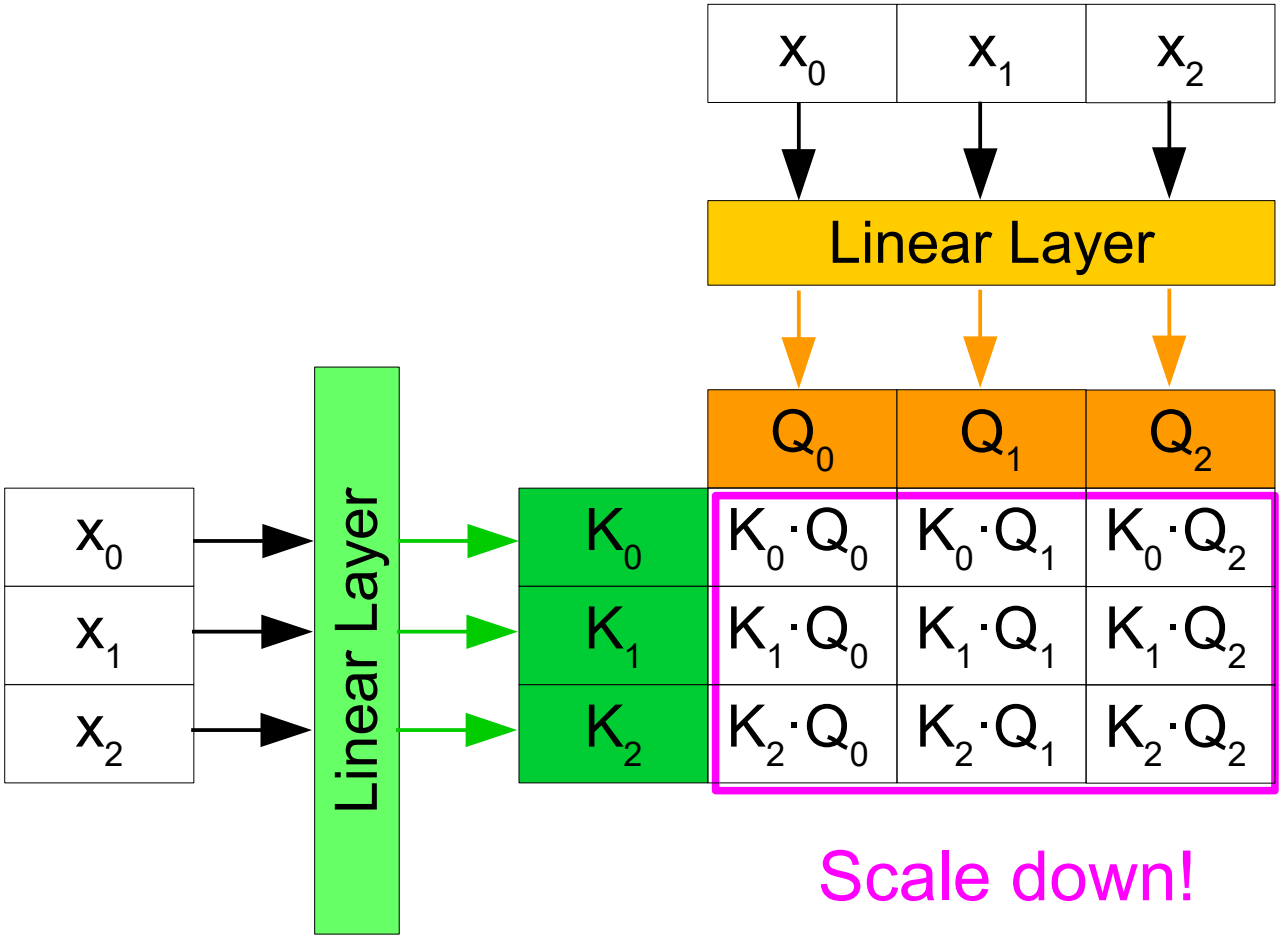
2. Intuitive understanding of self-attention



2. Intuitive understanding of self-attention

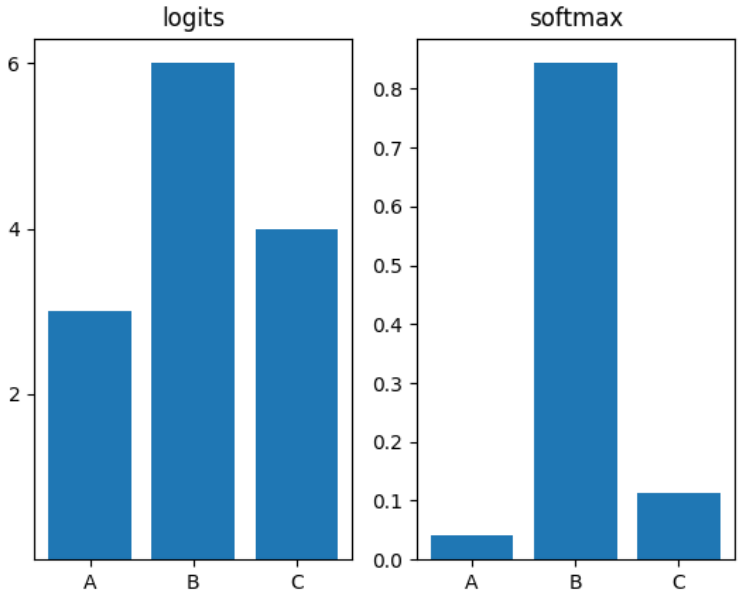


2. Intuitive understanding of self-attention

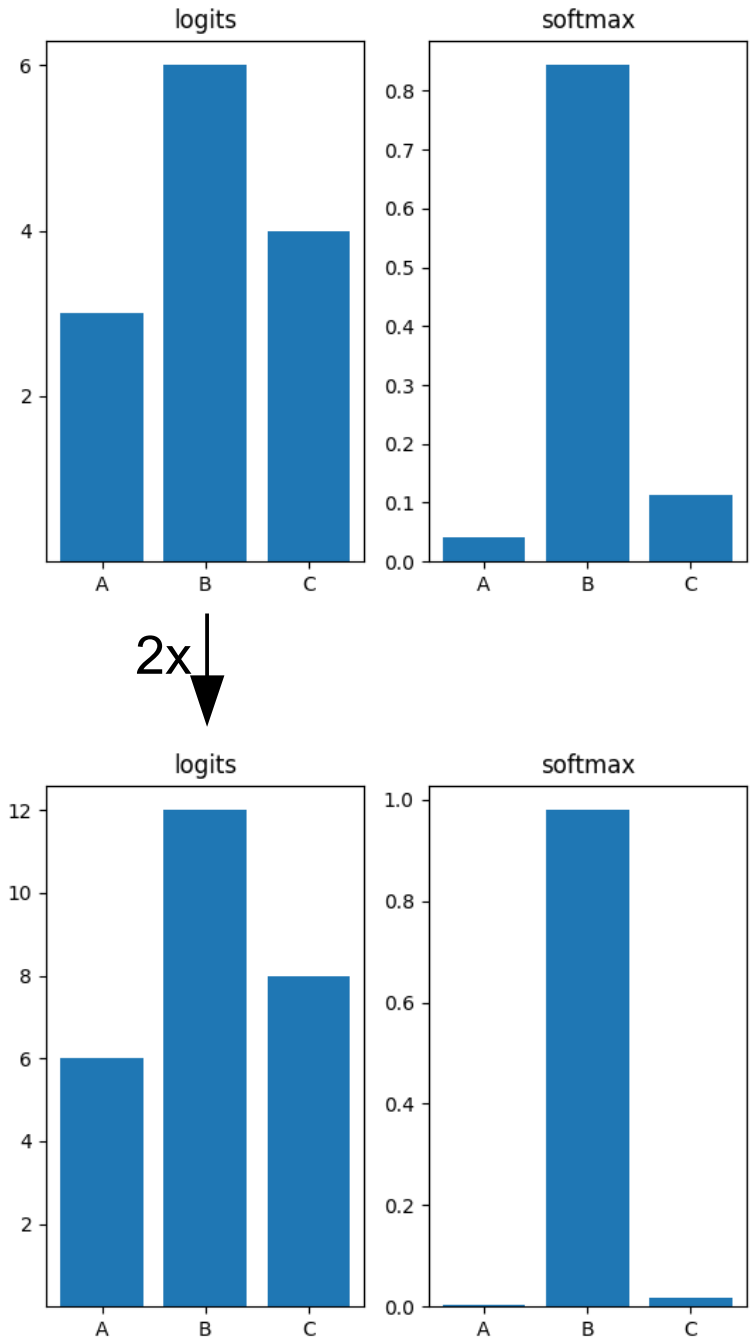


$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

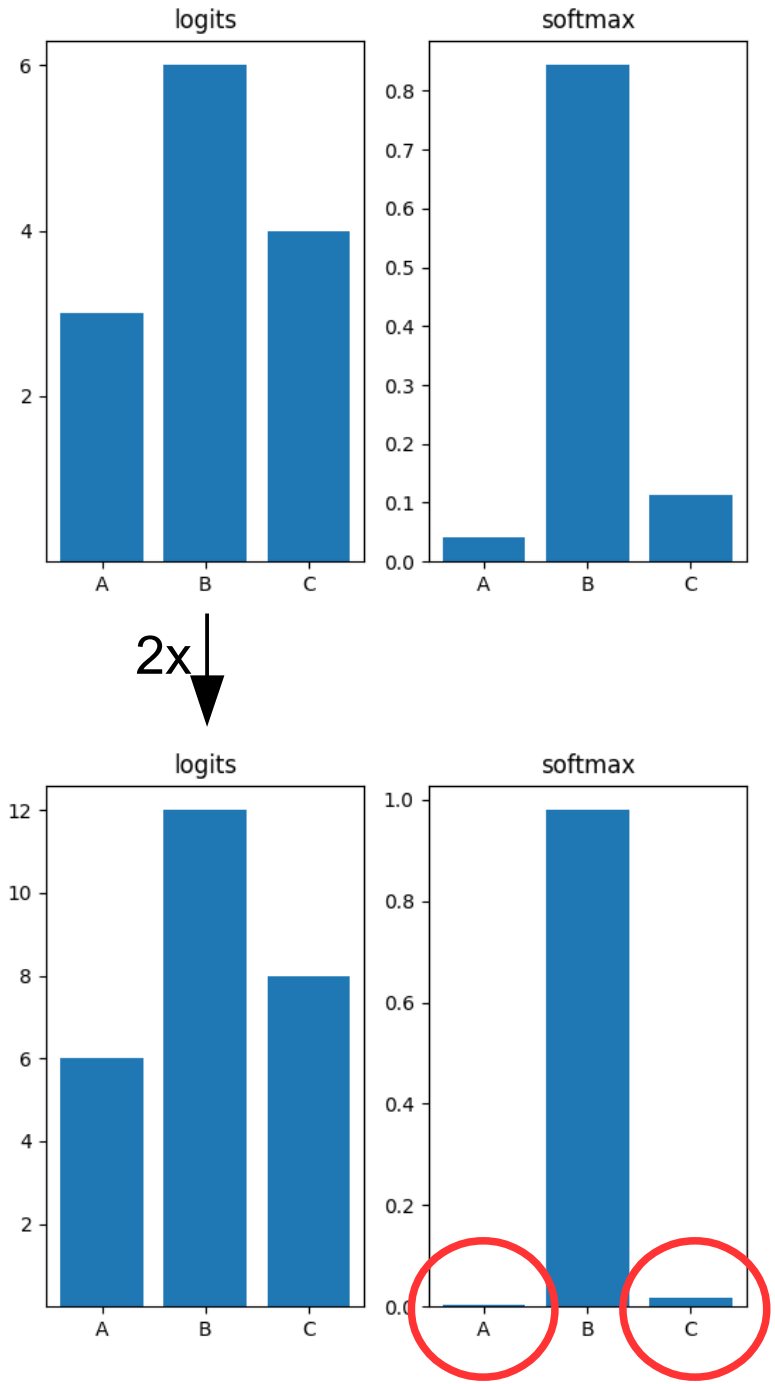
Side note: softmax is not scale invariant



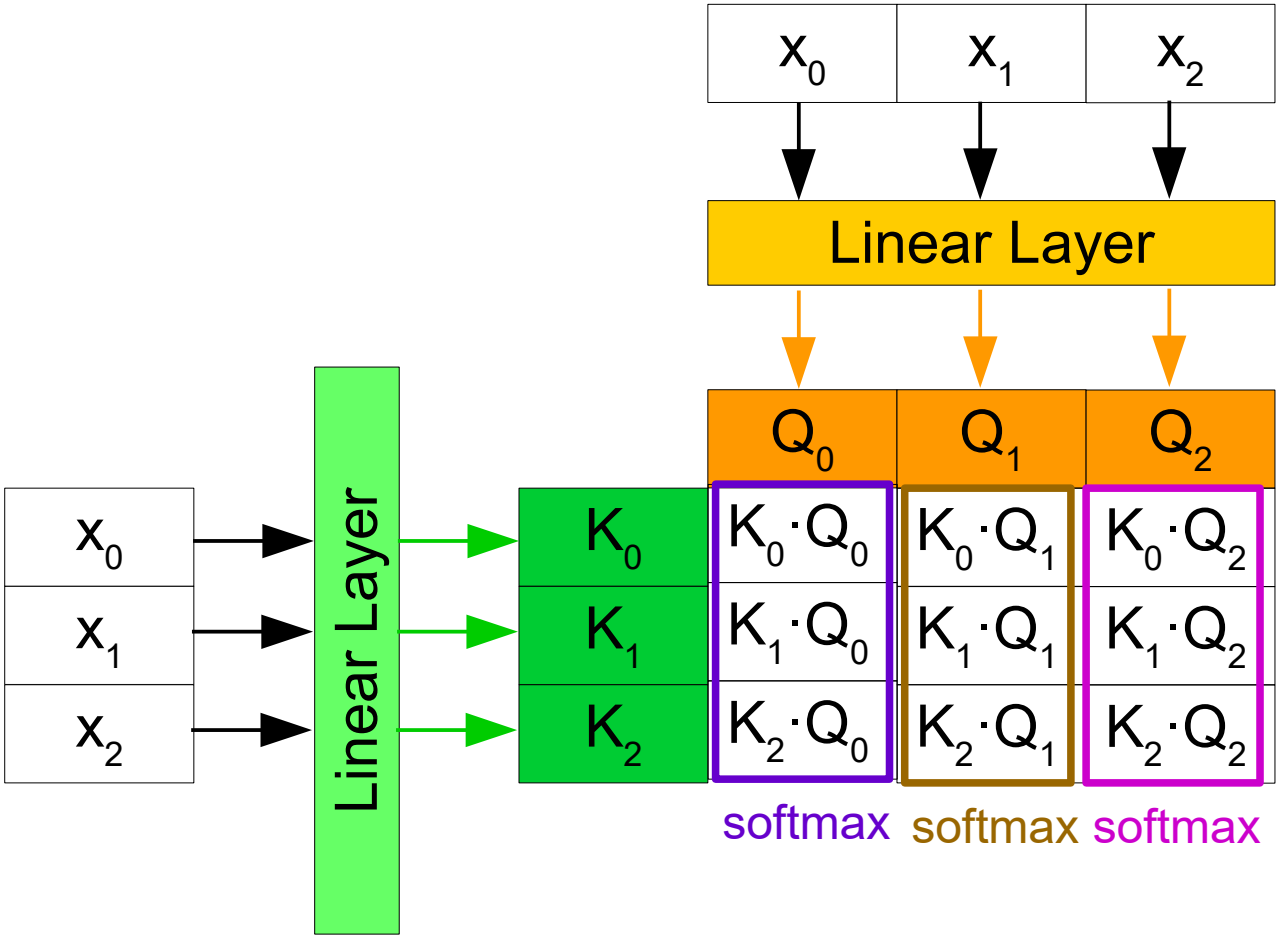
Side note: softmax is not scale invariant



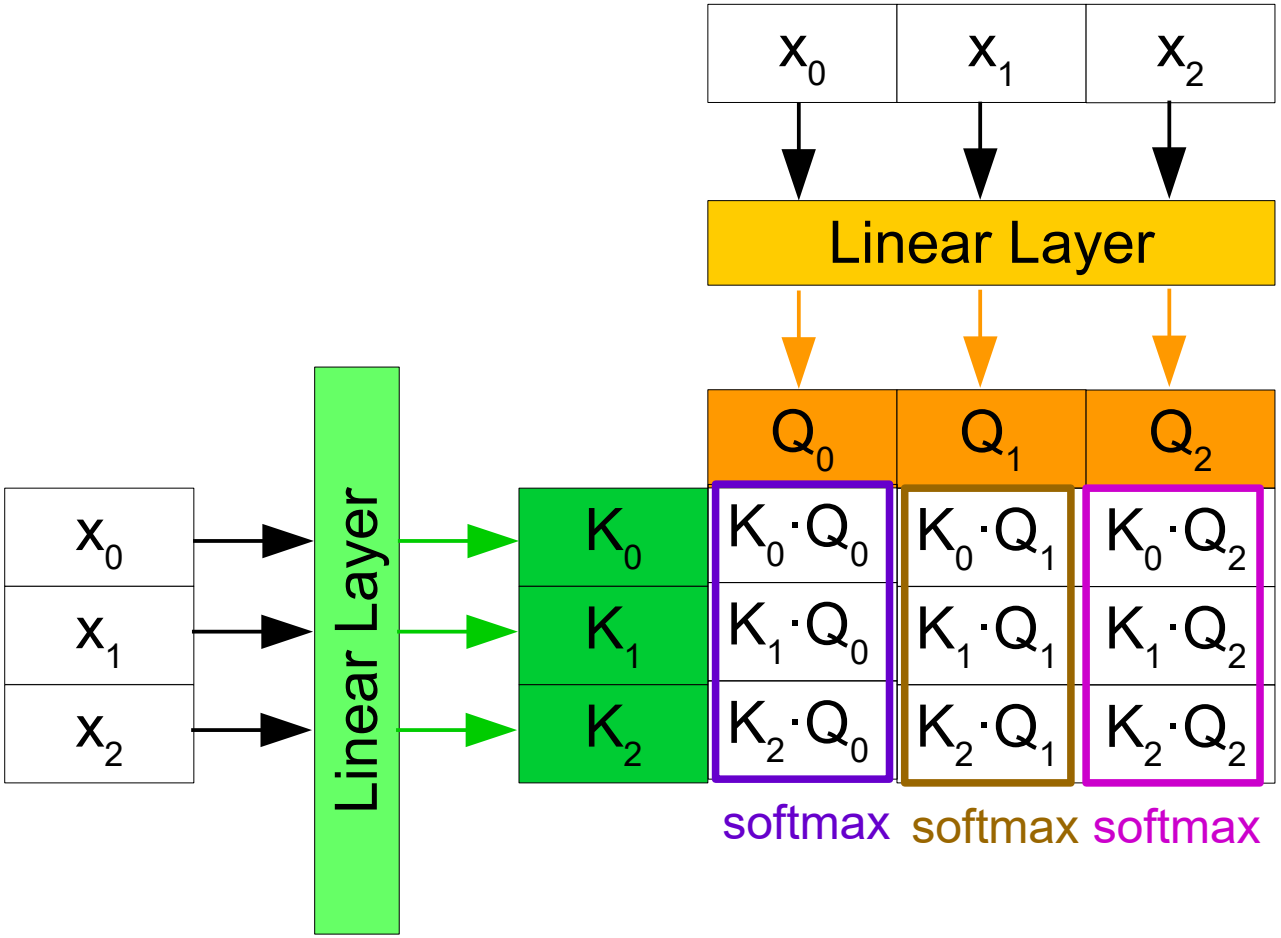
Side note: softmax is not scale invariant



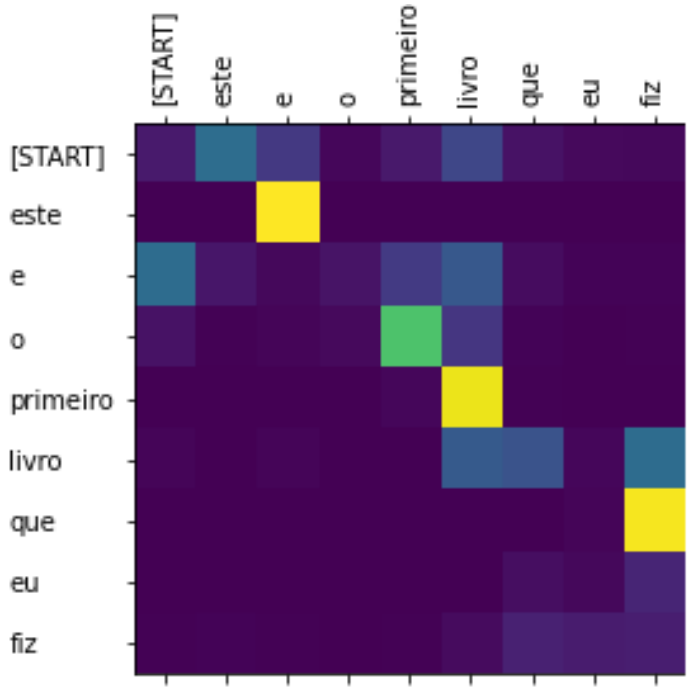
2. Intuitive understanding of self-attention



2. Intuitive understanding of self-attention



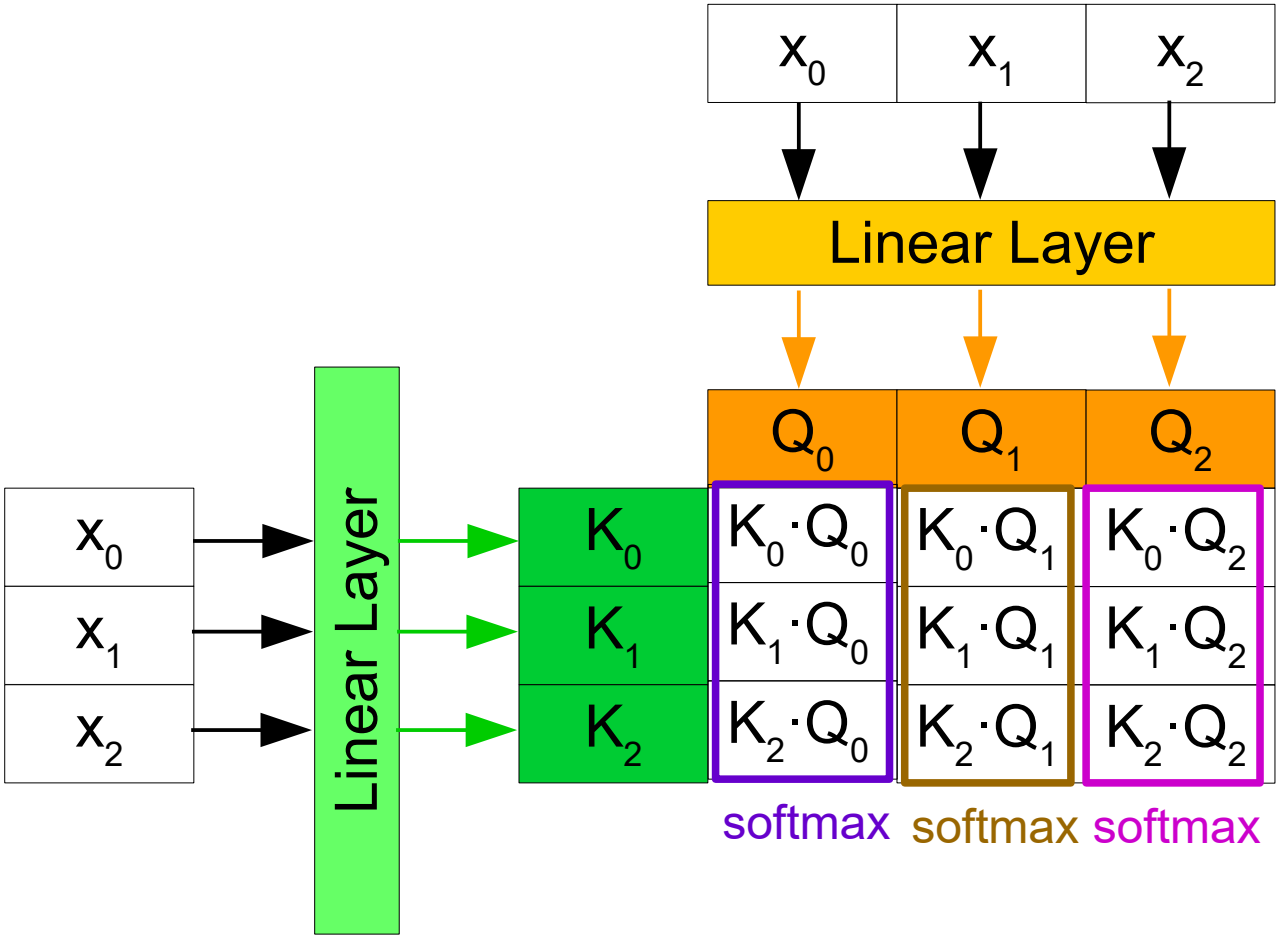
=



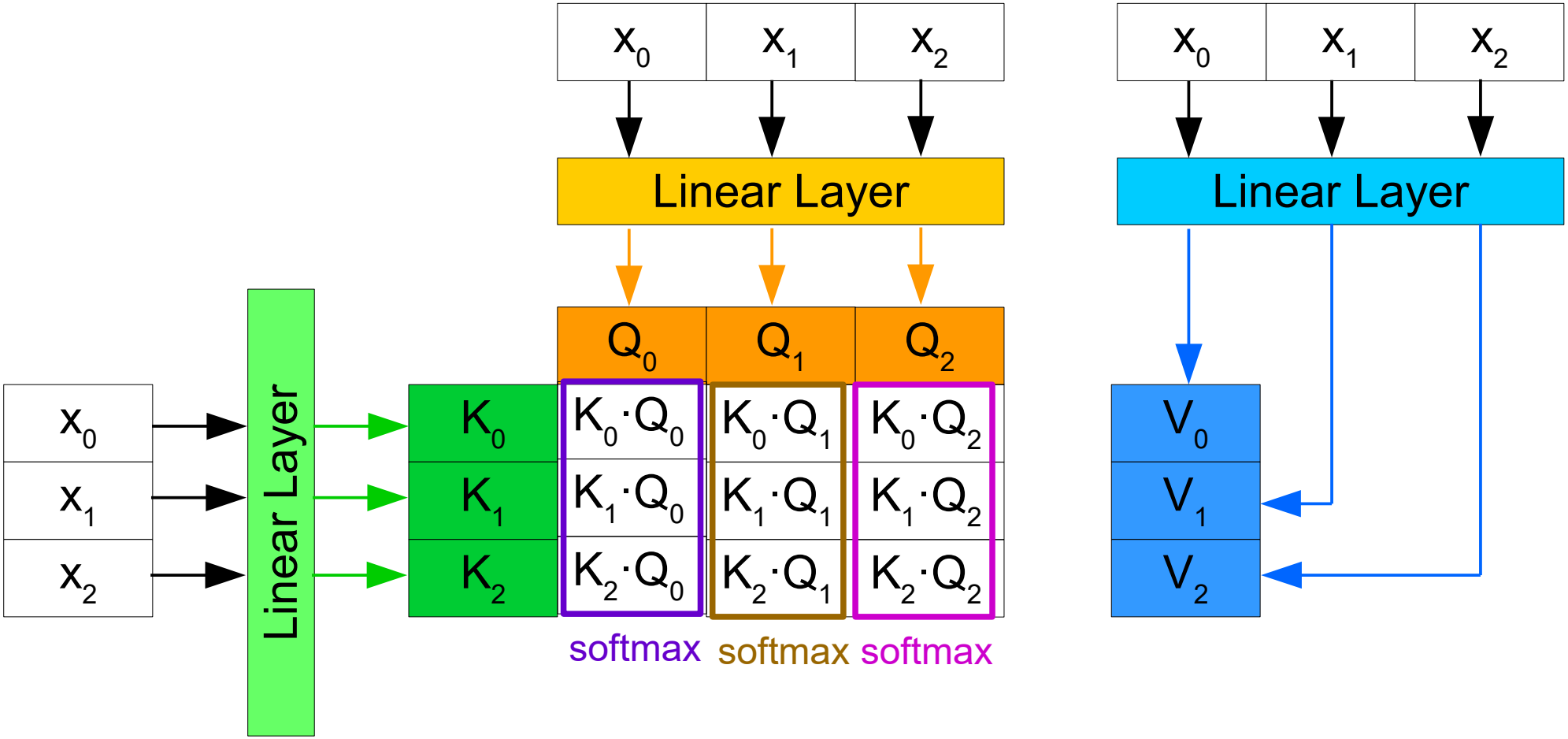
Here: 3 words instead of 9 ;)

Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

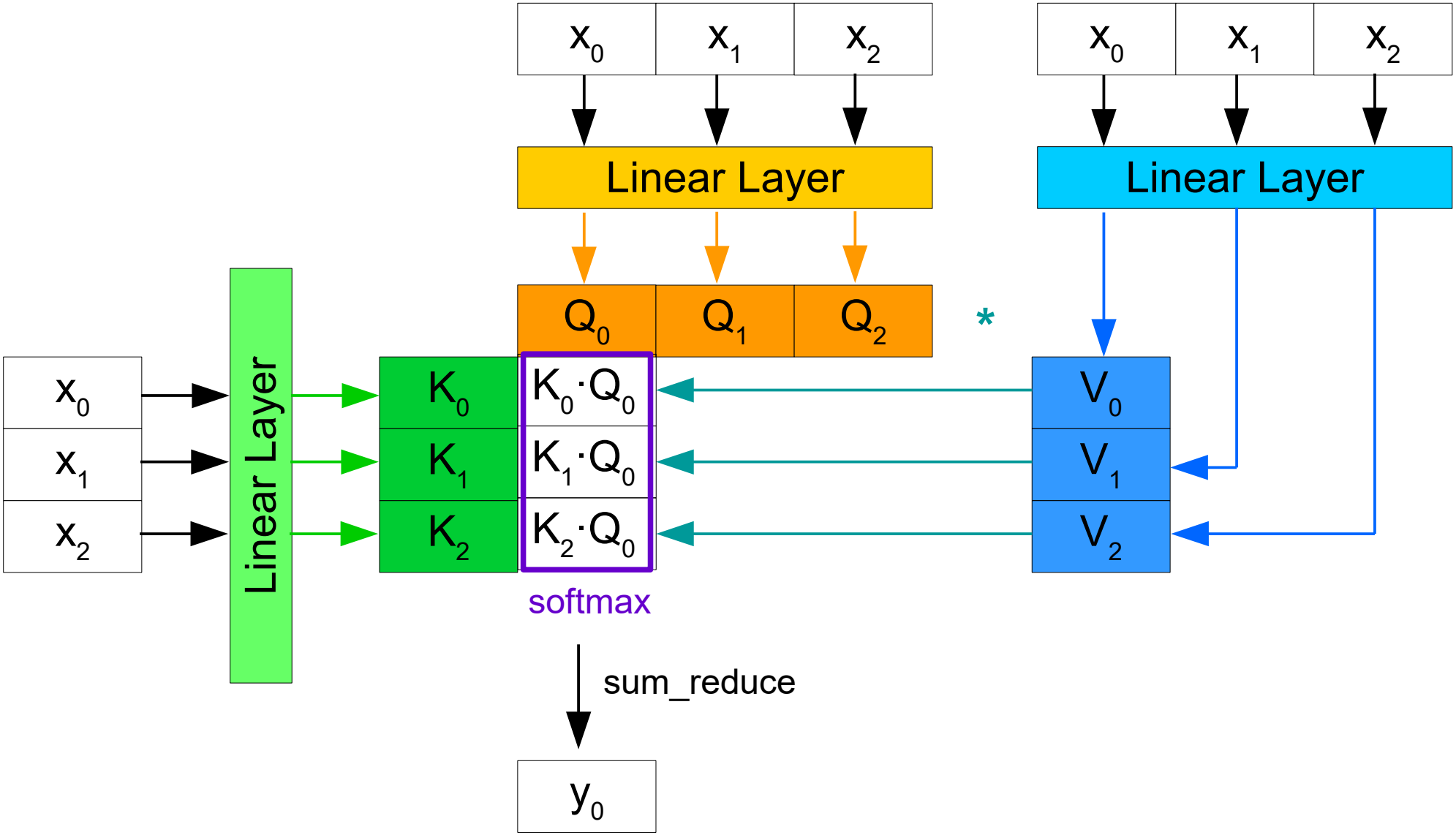
2. Intuitive understanding of self-attention



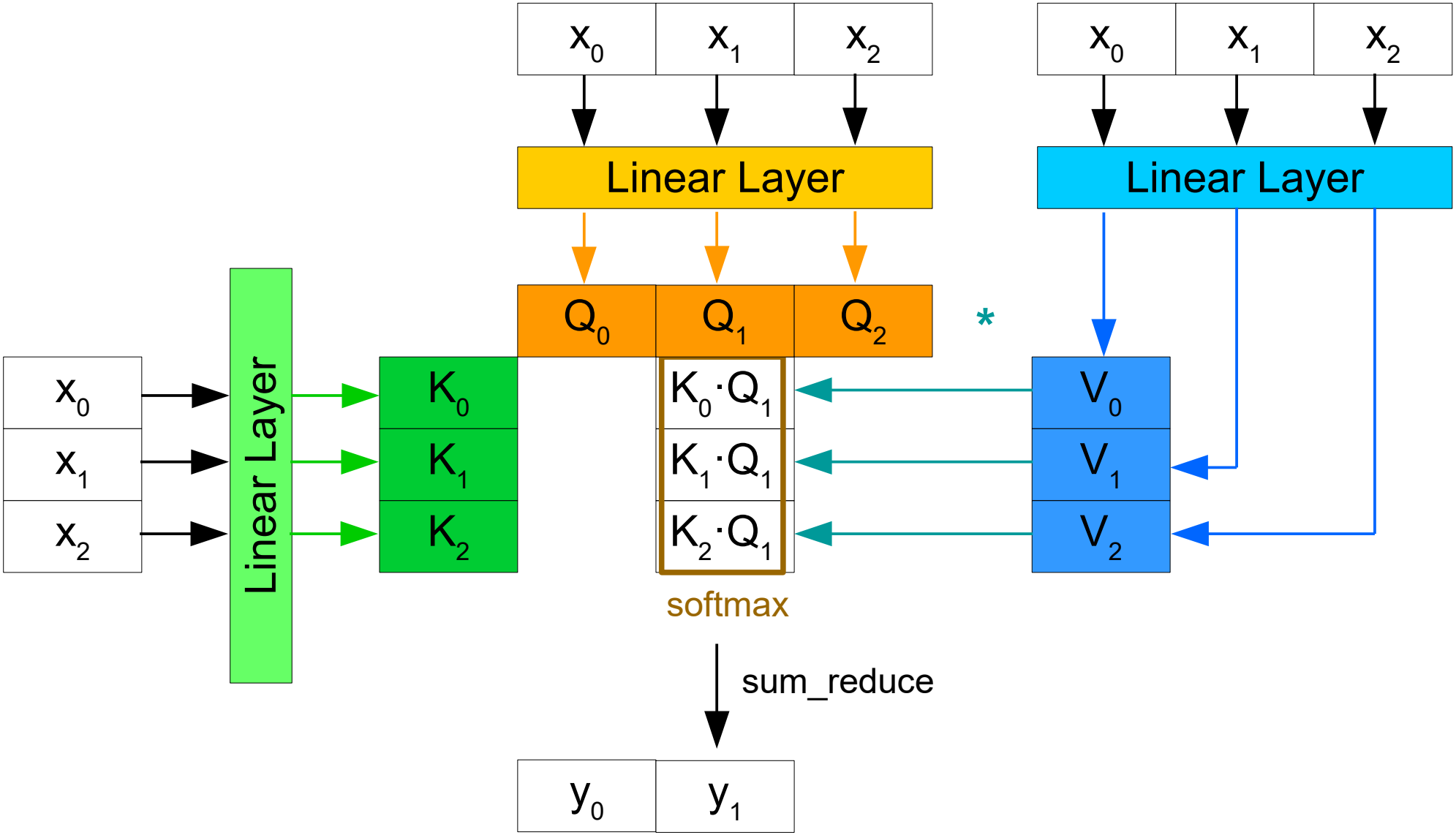
2. Intuitive understanding of self-attention



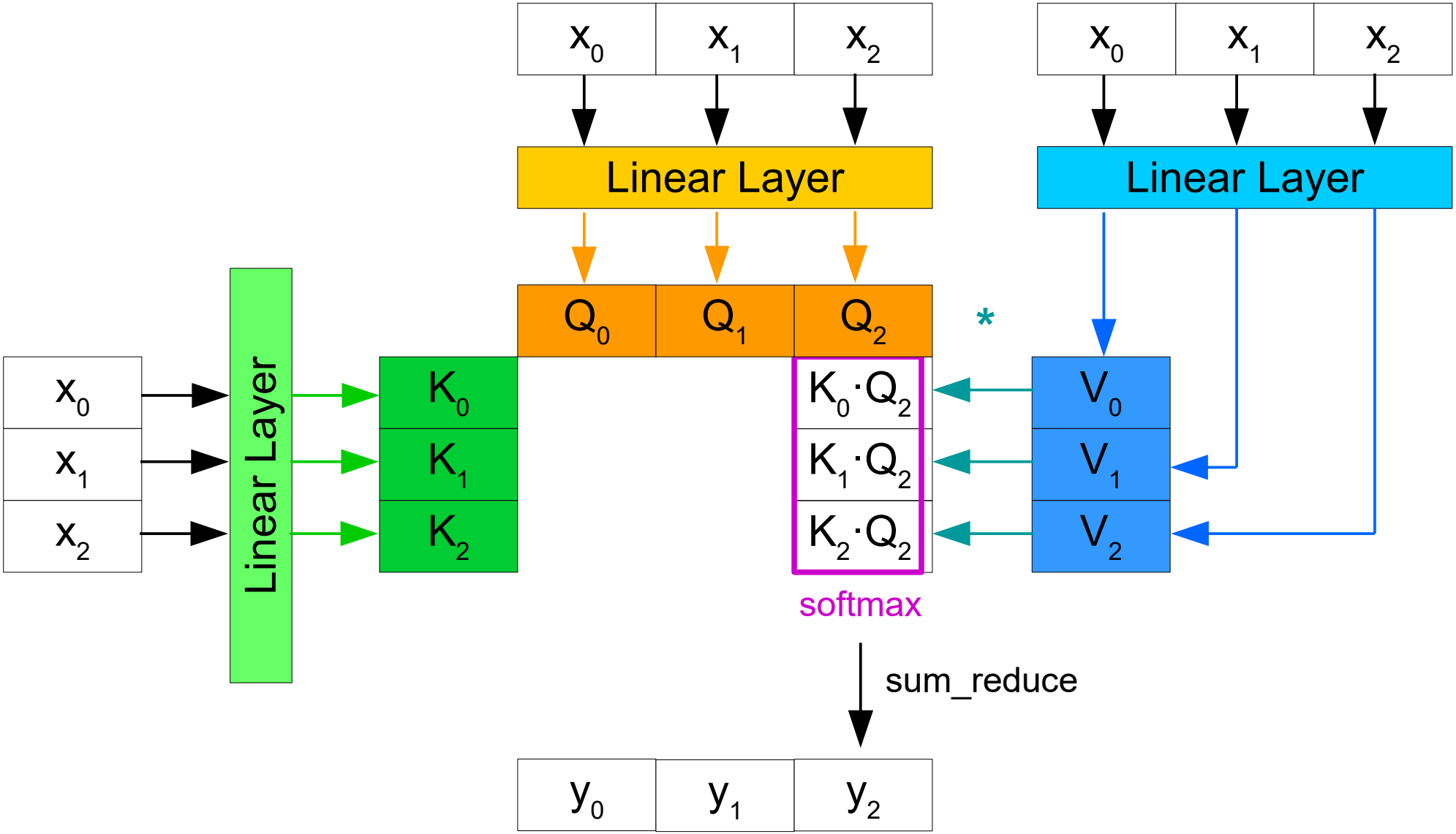
2. Intuitive understanding of self-attention



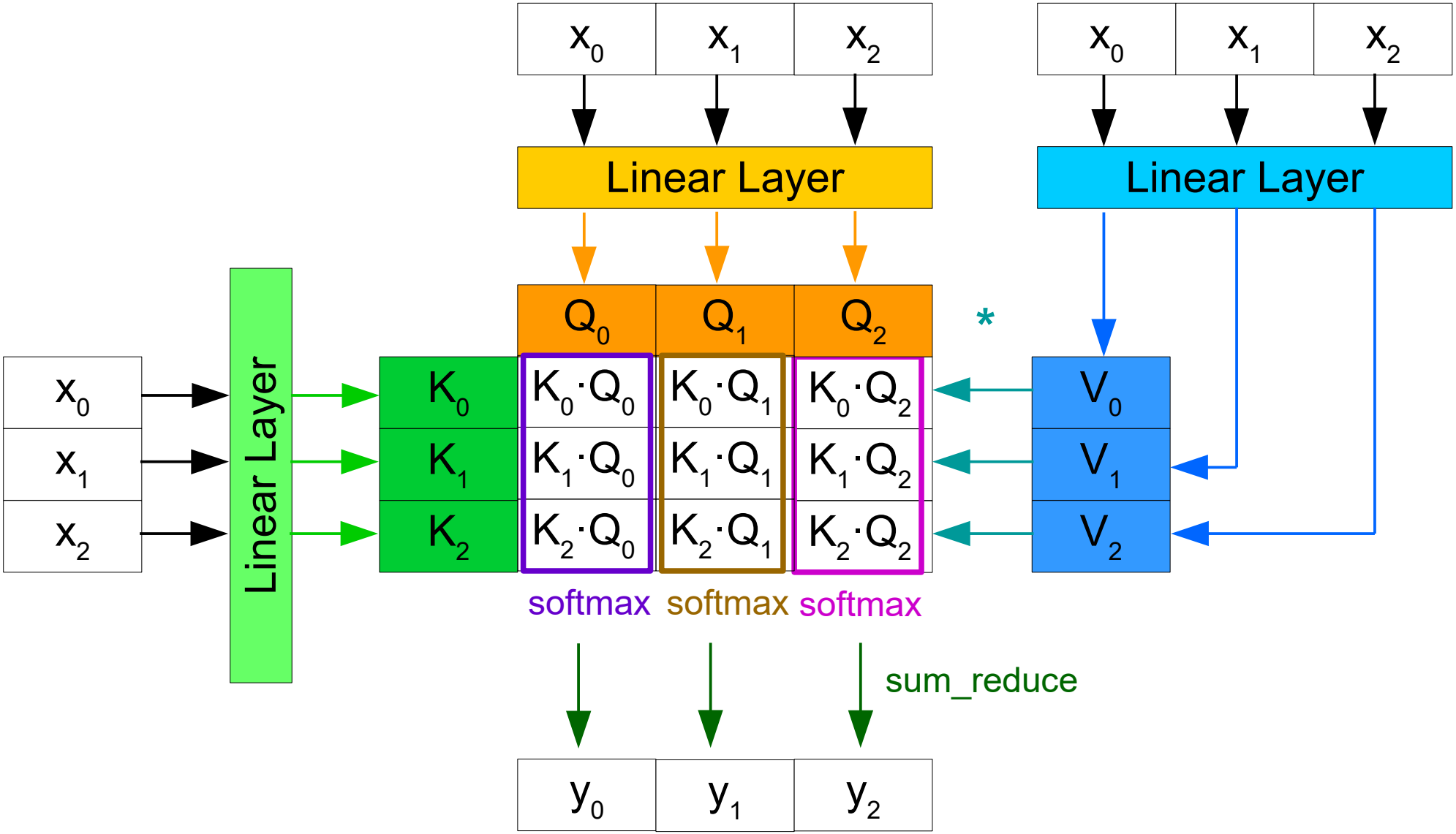
2. Intuitive understanding of self-attention



2. Intuitive understanding of self-attention



2. Intuitive understanding of self-attention



2.1 Multi-head attention

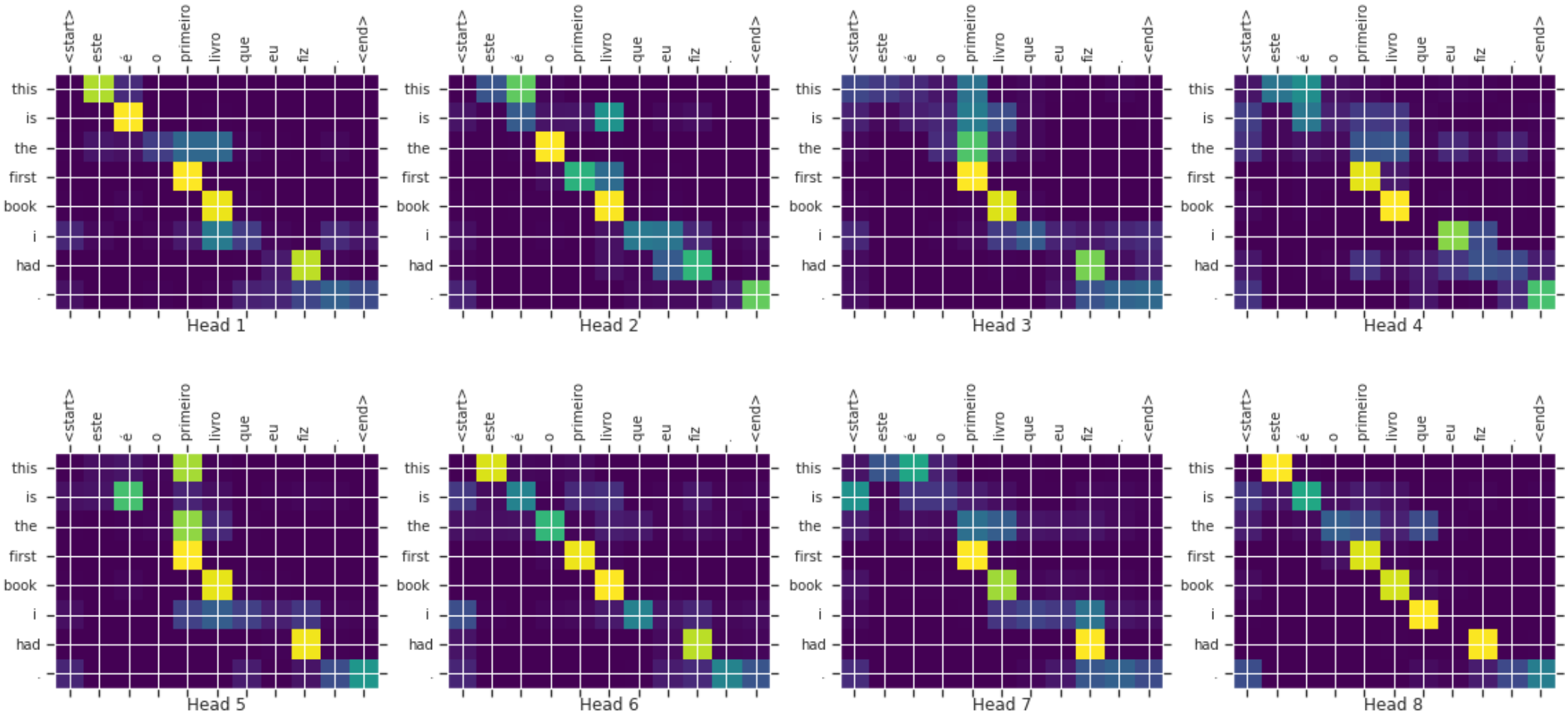


Figure 9: Heatmaps from Multi-head attention [2].

Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

2.1 Multi-head attention

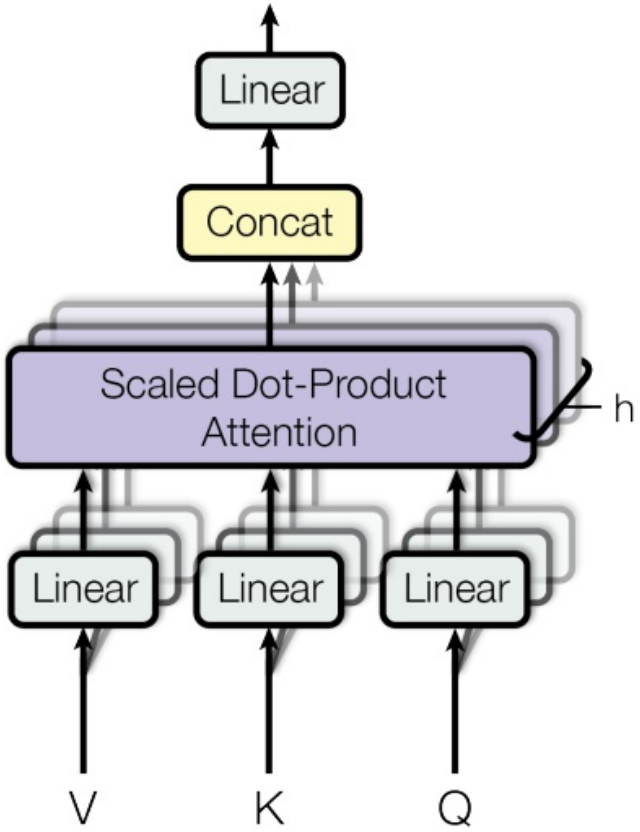
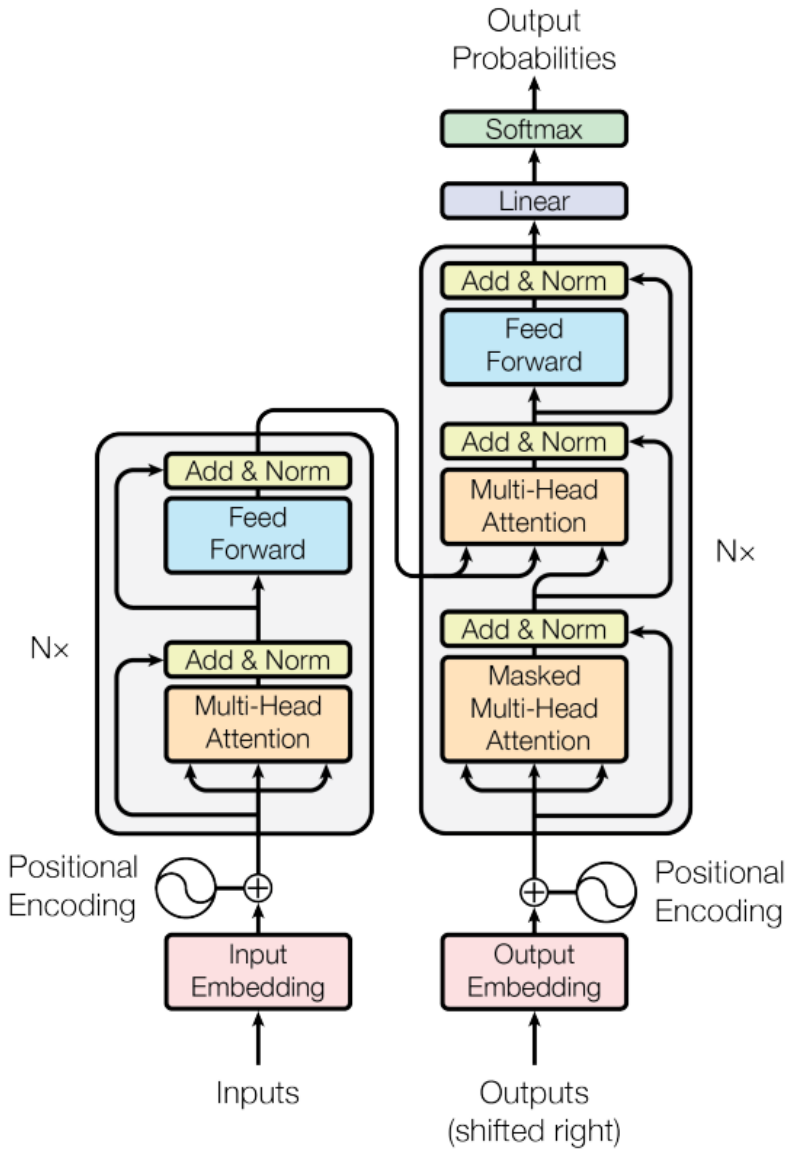


Figure 10: Overview of multi-head attention [2].



3. Intuitive understanding of the transformer architecture

Image sources:
https://machinelearningmastery.com/wp-content/uploads/2021/08/attention_research_1-727x1024.png
(call date: 20.07.22)

3.1 Let's build a transformer

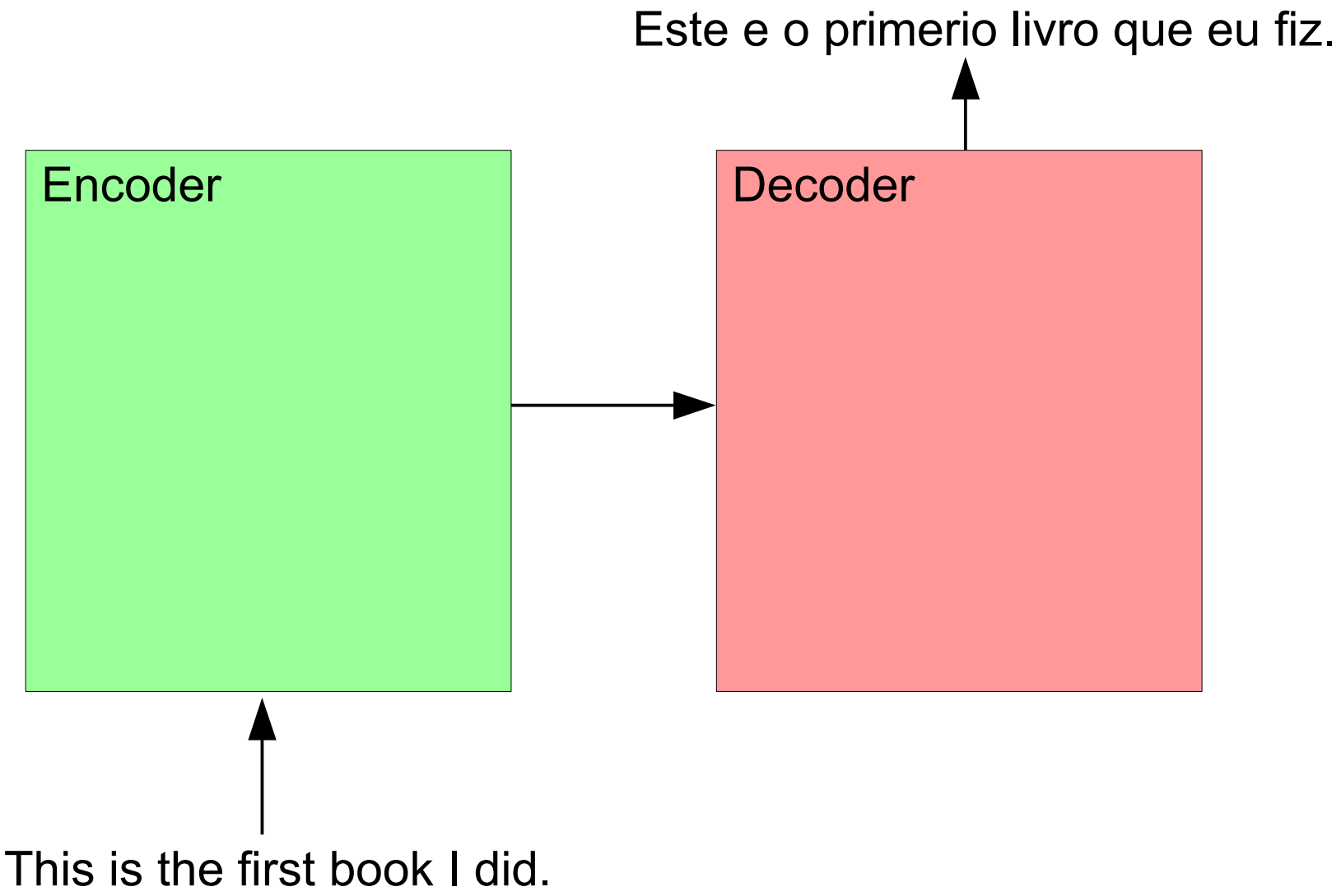


Figure 11: Neural machine translation based on a transformer [2].

3.1 Let's build a transformer

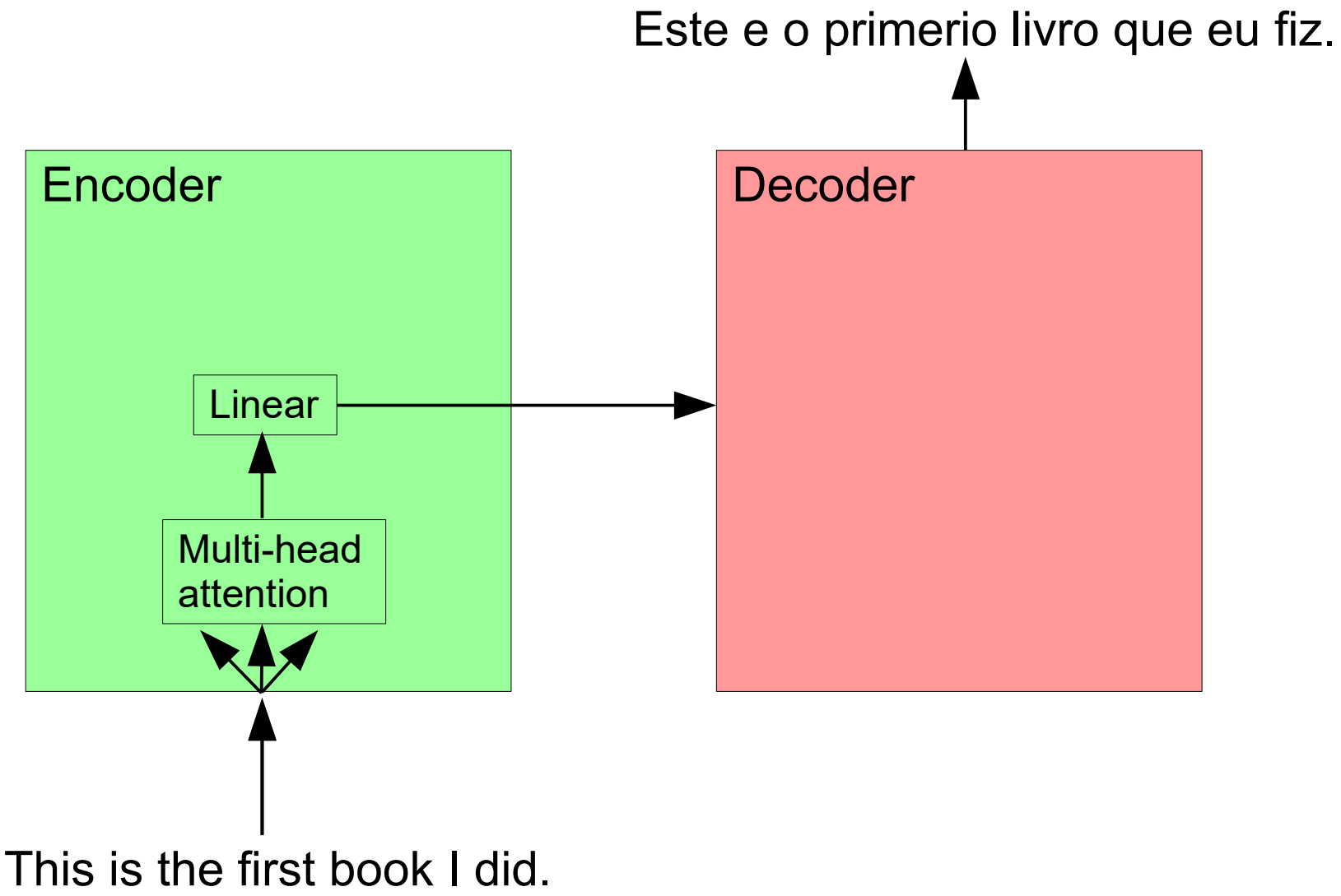


Figure 11: Neural machine translation based on a transformer [2].

3.1 Let's build a transformer

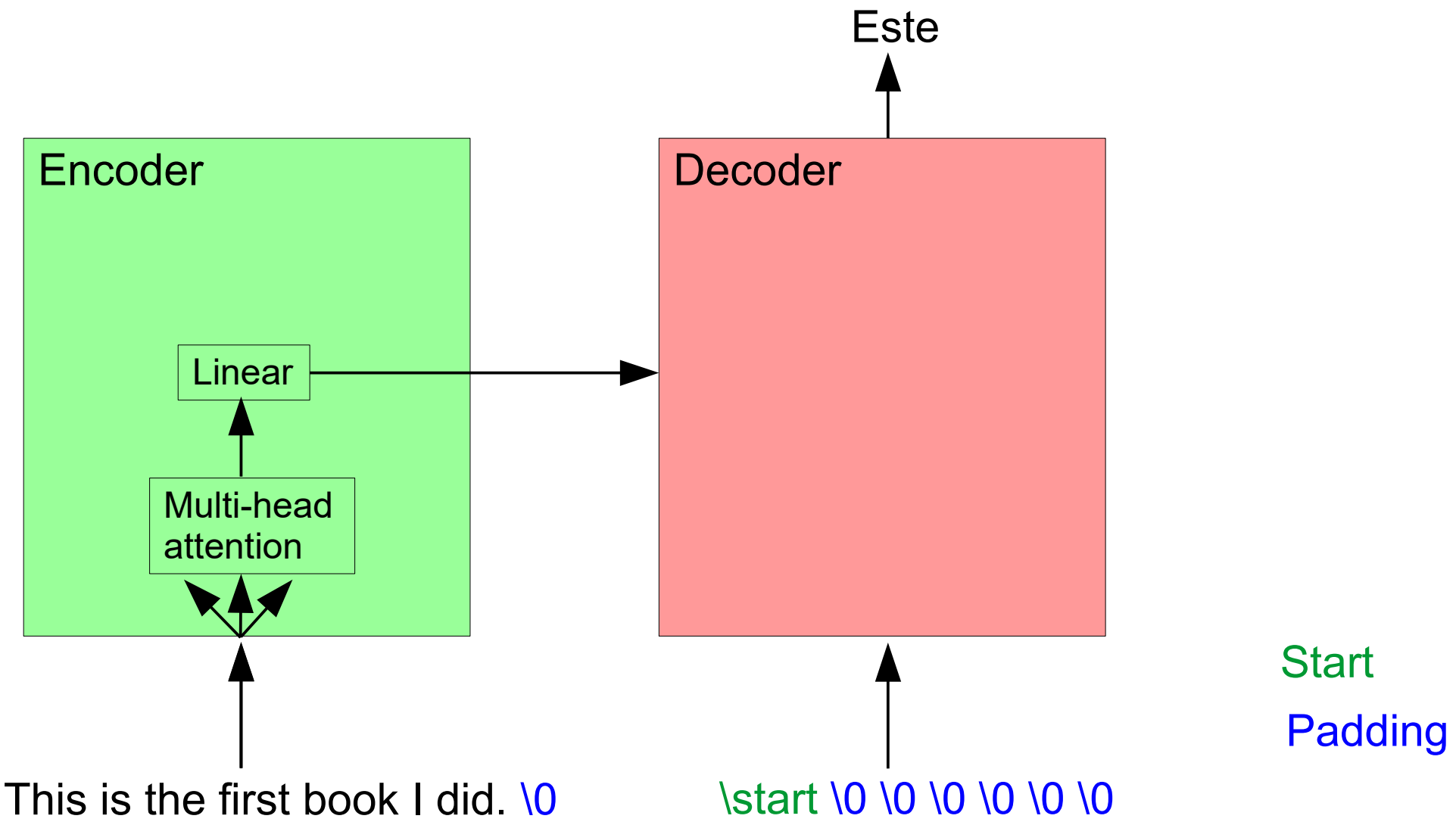


Figure 11: Neural machine translation based on a transformer [2].

3.1 Let's build a transformer

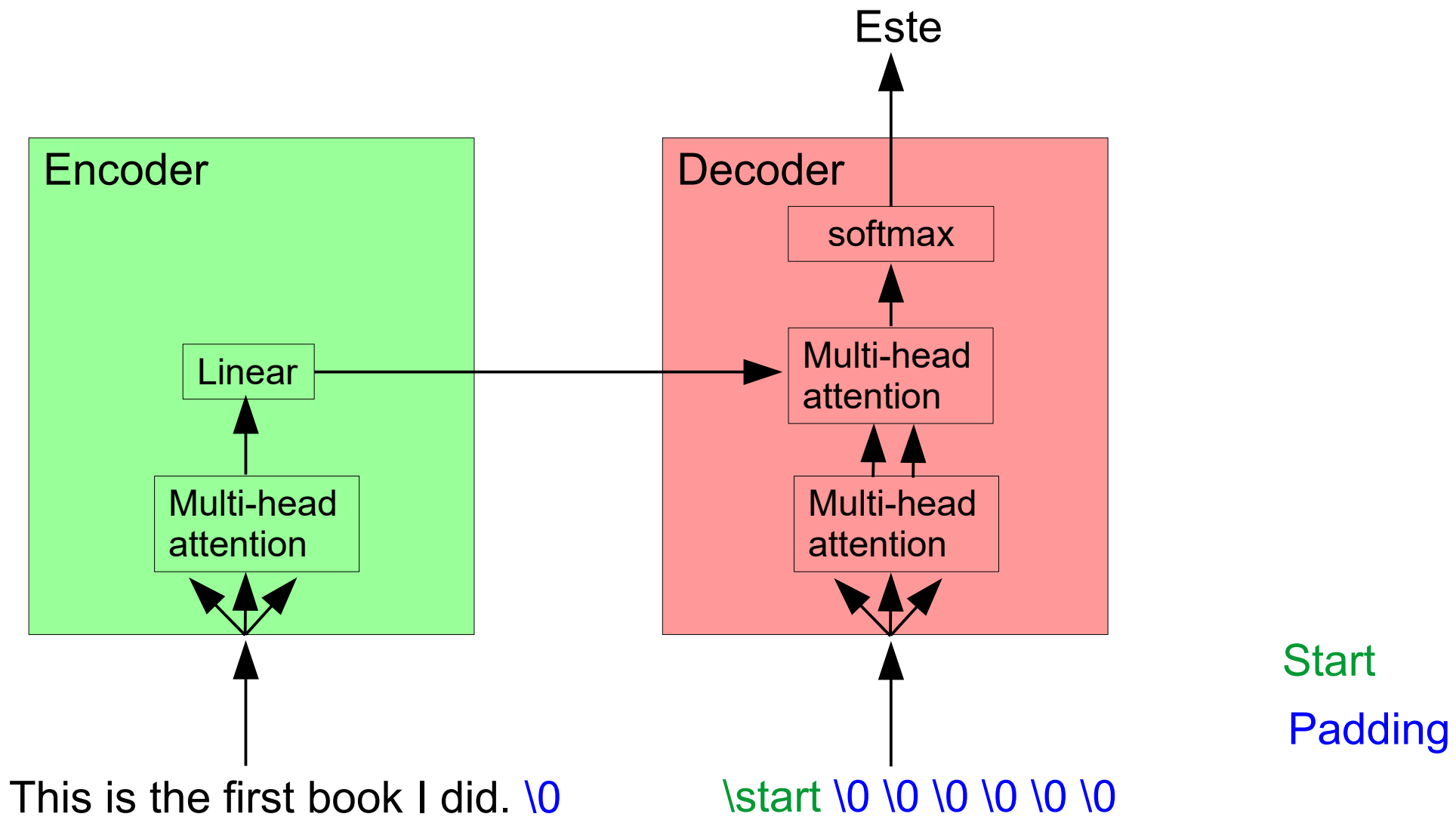


Figure 11: Neural machine translation based on a transformer [2].

3.1 Let's build a transformer

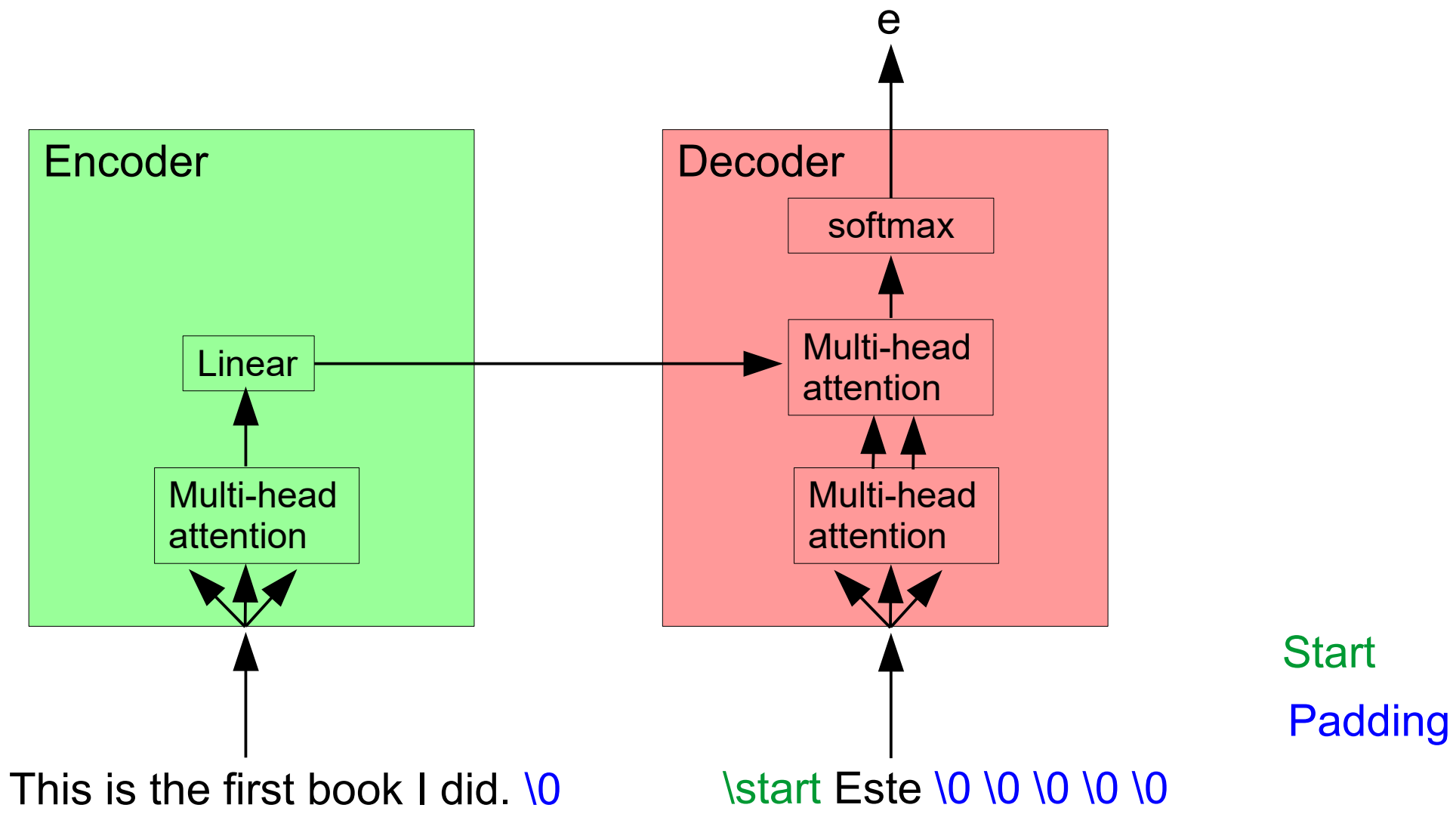


Figure 11: Neural machine translation based on a transformer [2].

3.1 Let's build a transformer

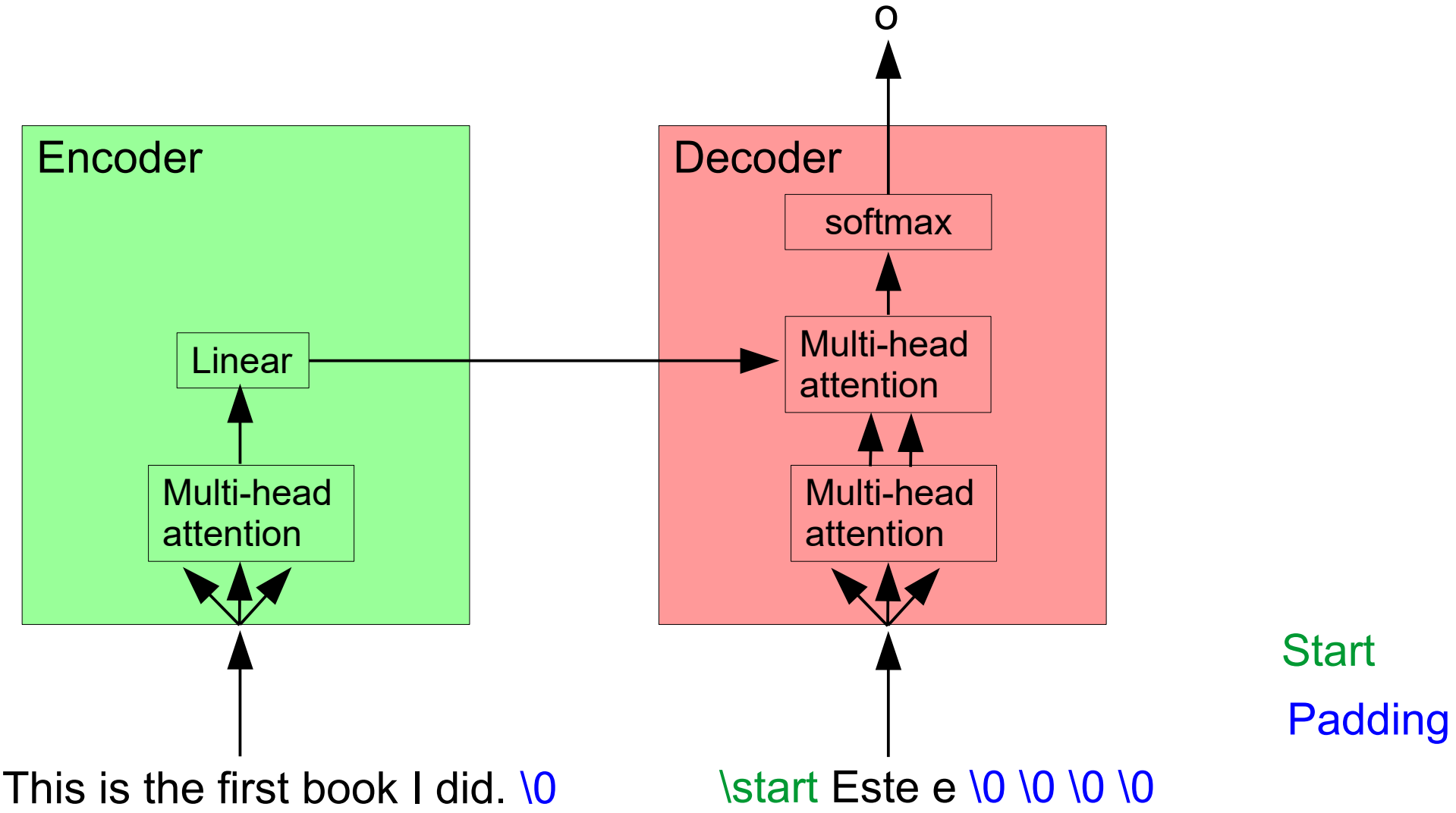


Figure 11: Neural machine translation based on a transformer [2].

3.2 Positional encoding

$$PE_{pos,2i} = \sin\left(pos/10000^{2i/d_{model}}\right)$$

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

depth/dimension index

3.2 Positional encoding

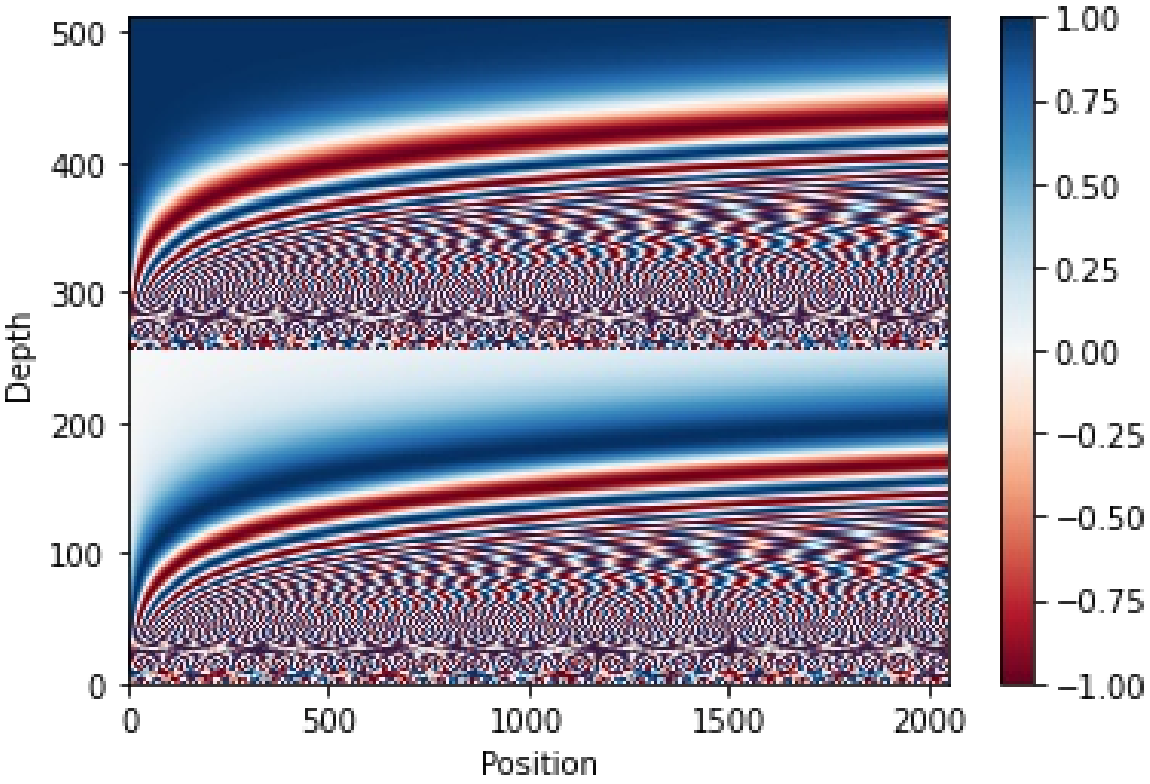


Figure 12: Positional encoding applied in a transformer [13].

$$PE_{pos,2i} = \sin\left(pos/10000^{2i/d_{model}}\right)$$

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

depth/dimension index

Image source:
https://www.tensorflow.org/static/text/tutorials/transformer_files/output_1kLCla68EloE_1.png
(call date: 20.07.22)

3.3 Masking – Encoder and Decoder Padding

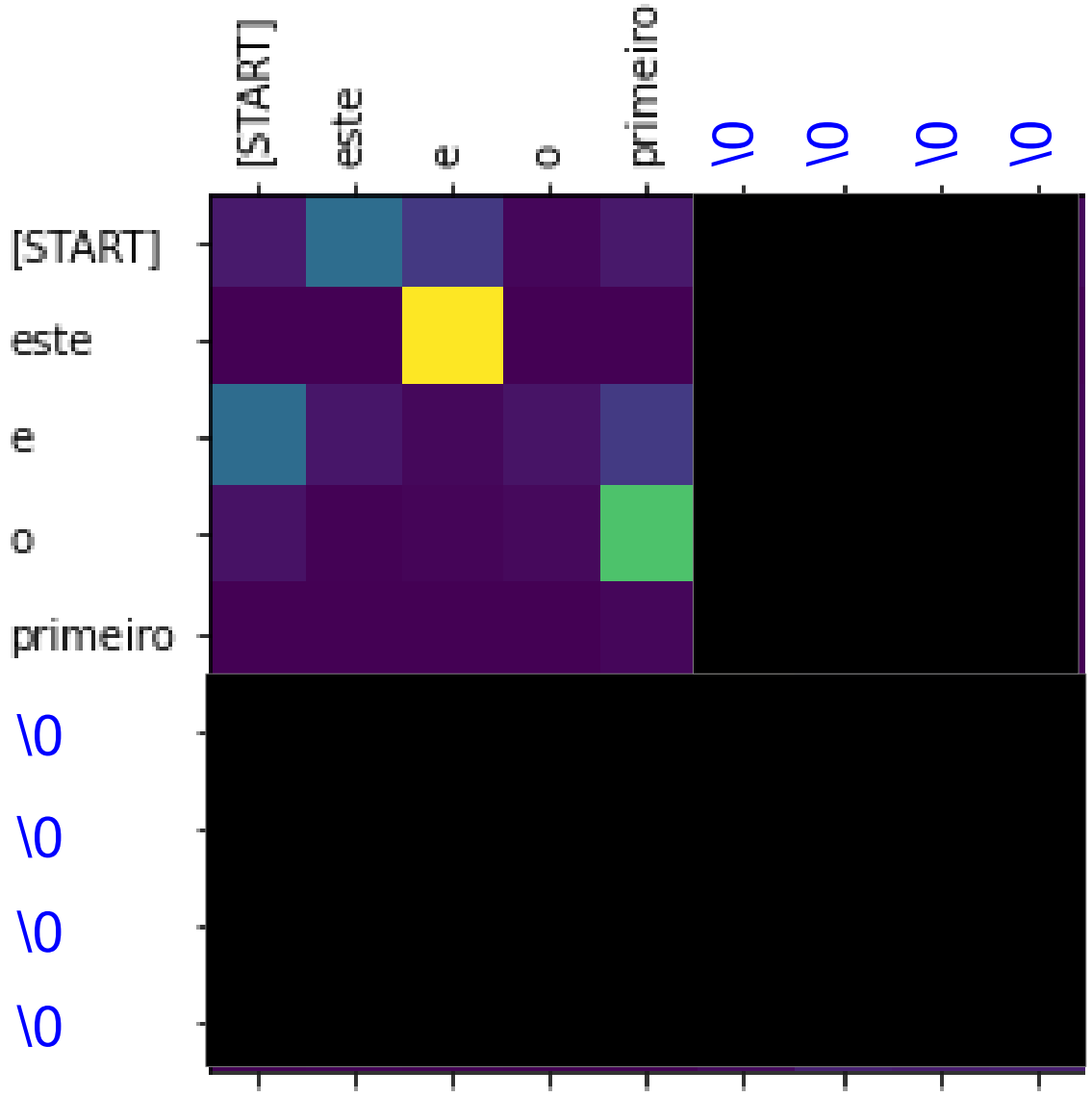


Figure 13: Masking padding words [11].

Image source:
https://www.tensorflow.org/static/text/tutorials/transformer_files/output_1kLCla68EloE_1.png
(call date: 20.07.22)

3.3 Masking – Decoder: Left to right information flow

A word in the output sequence is not allow to attend on words coming after it.

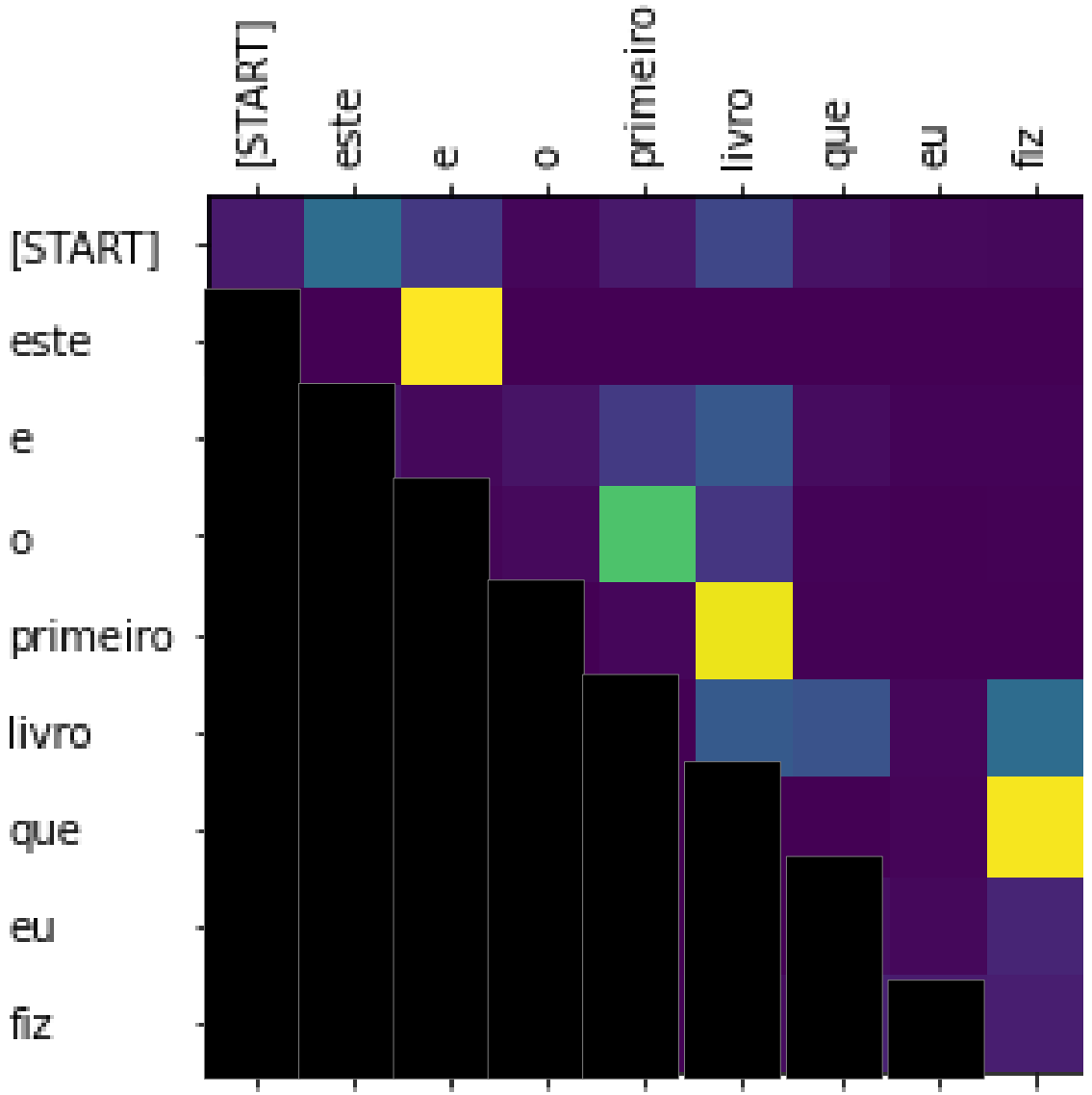


Figure 14: All values in the self-attention matrix above the diagonal are zero [11].

Image source:
https://www.tensorflow.org/static/text/tutorials/transformer_files/output_1kLCla68EloE_1.png
(call date: 20.07.22)

3.3 Masking – Decoder: Left to right information flow

A word in the output sequence is not allow to attend on words coming after it.

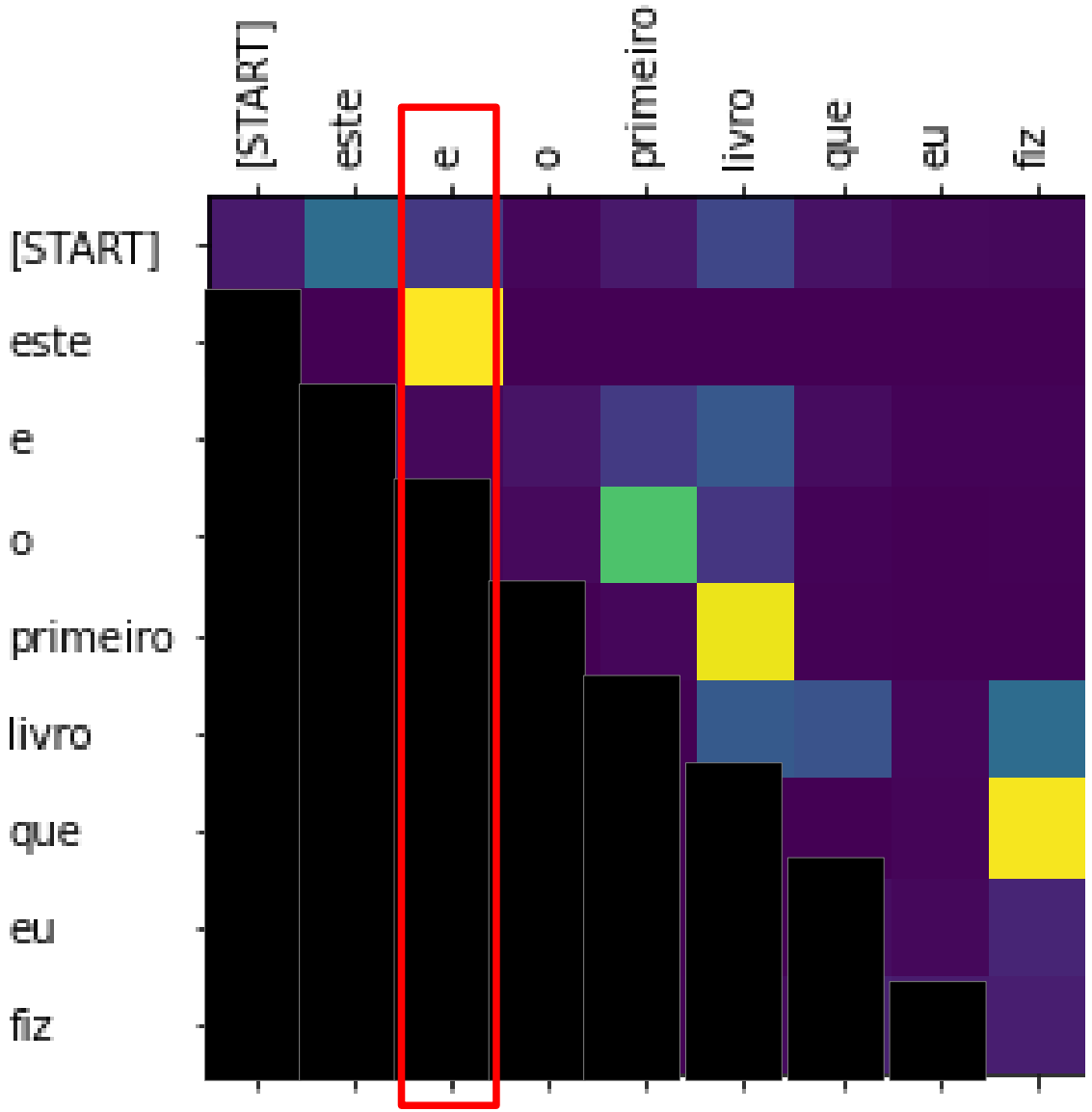


Figure 14: All values in the self-attention matrix above the diagonal are zero [11].

Image source:
https://www.tensorflow.org/static/text/tutorials/transformer_files/output_1kLCla68EloE_1.png
(call date: 20.07.22)

3.1 Let's build a transformer

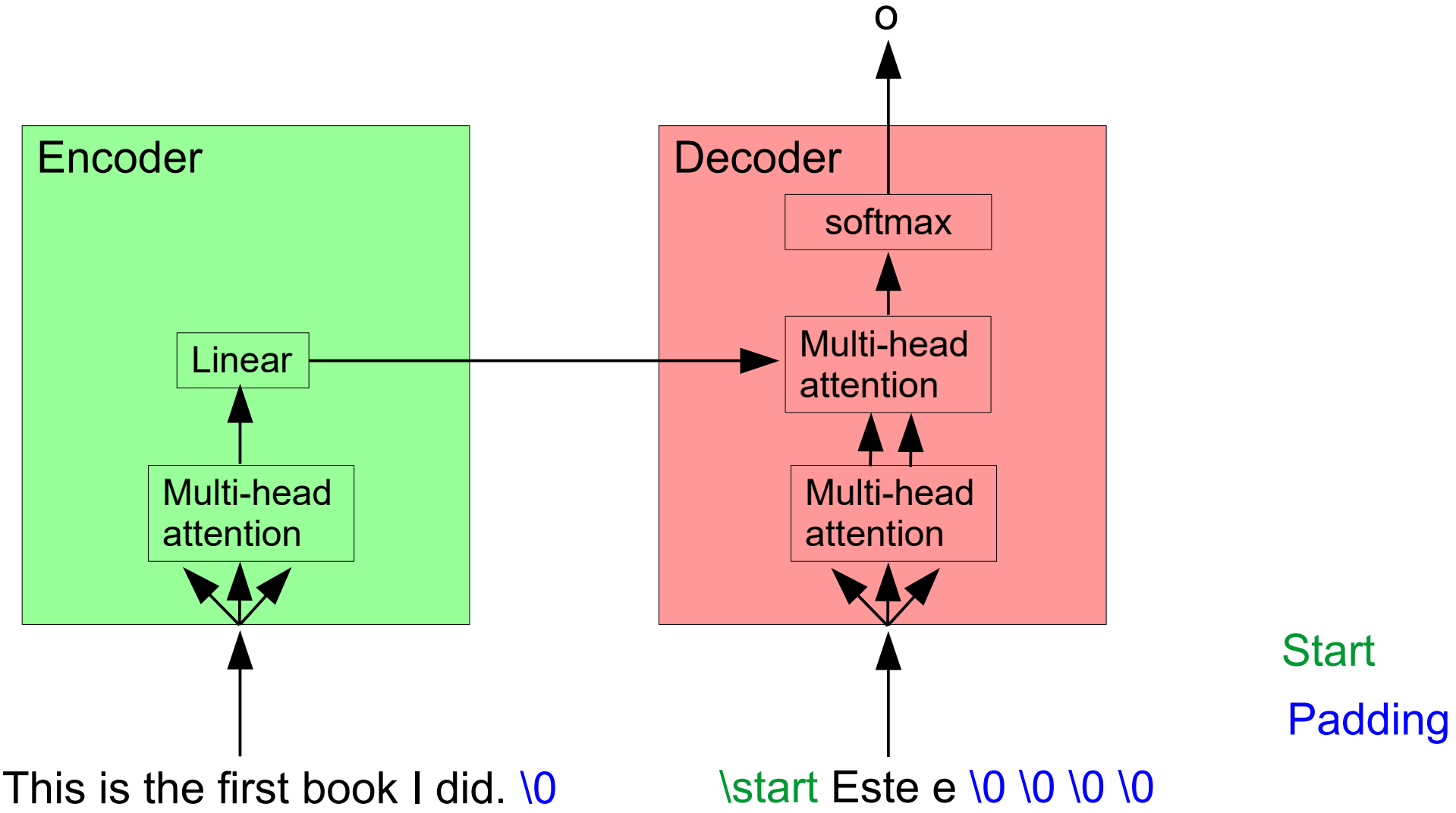


Figure 11: Neural machine translation based on a transformer [2].

3.4 Transformer

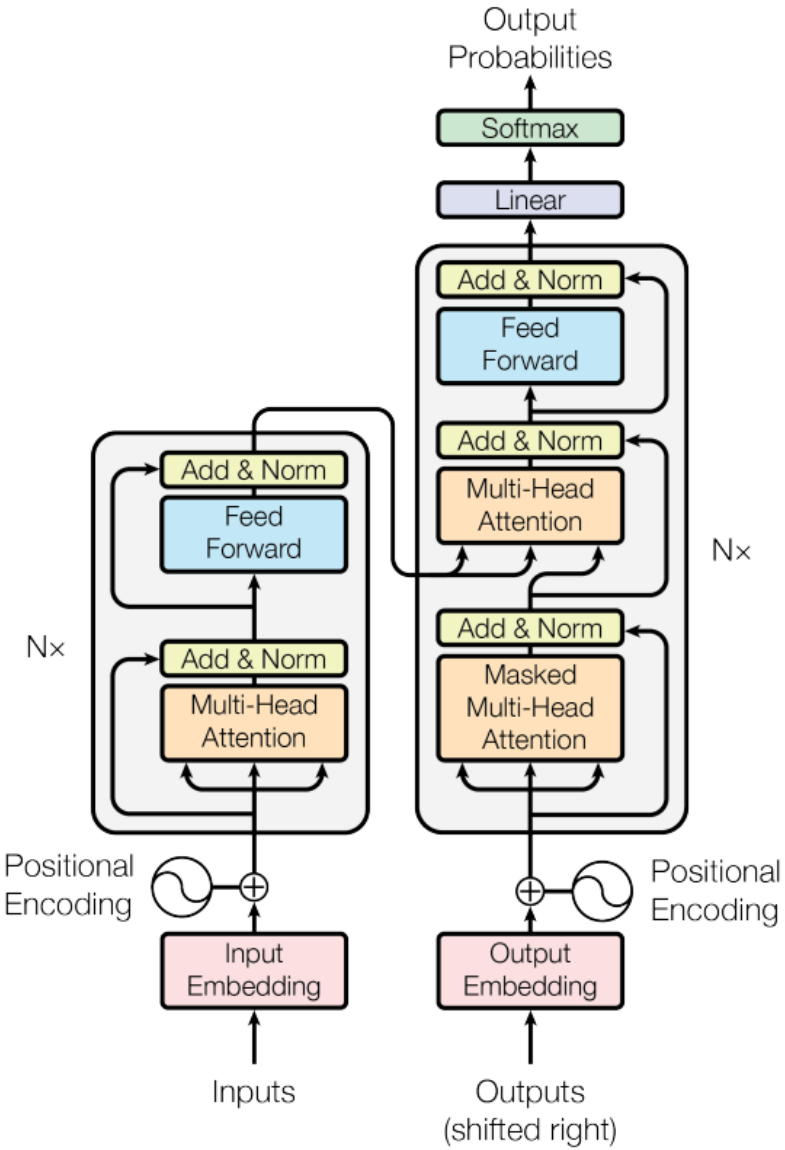


Figure 15: Transformer Architecture [2].

3.4 Transformer

Residual connection

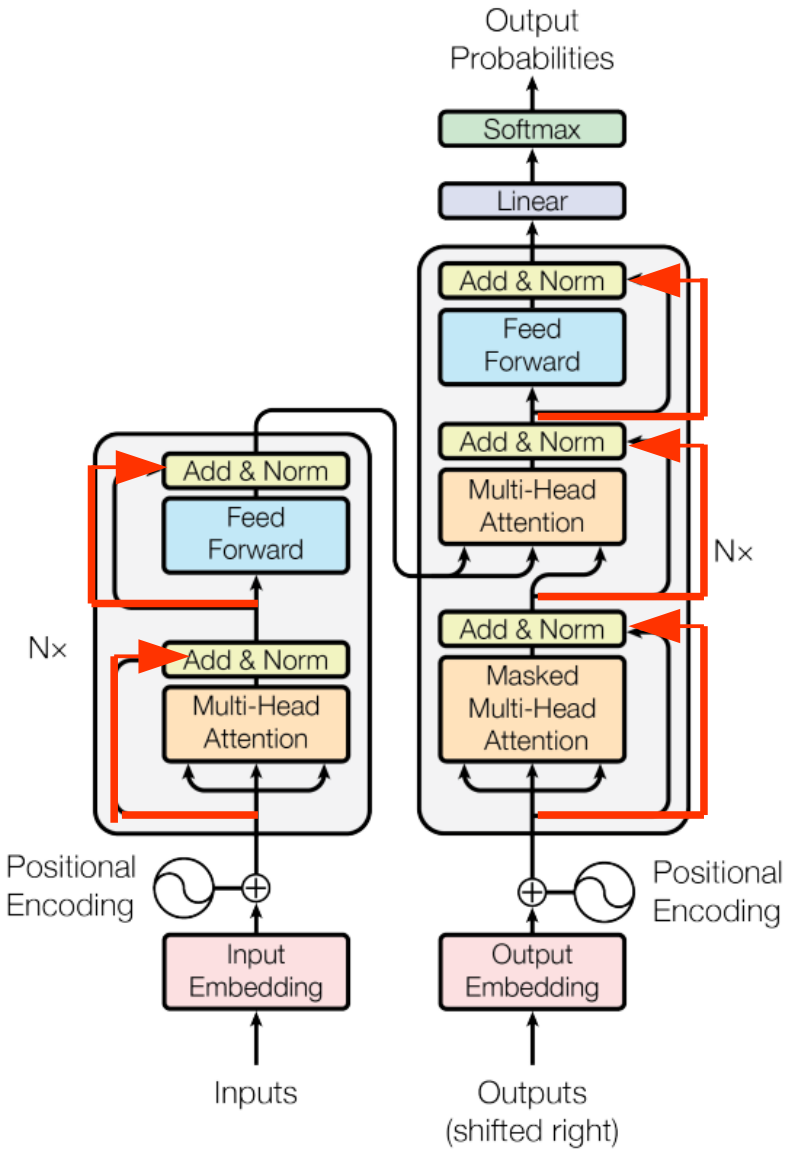


Figure 15: Transformer Architecture [2].

3.4 Transformer

Layer normalization

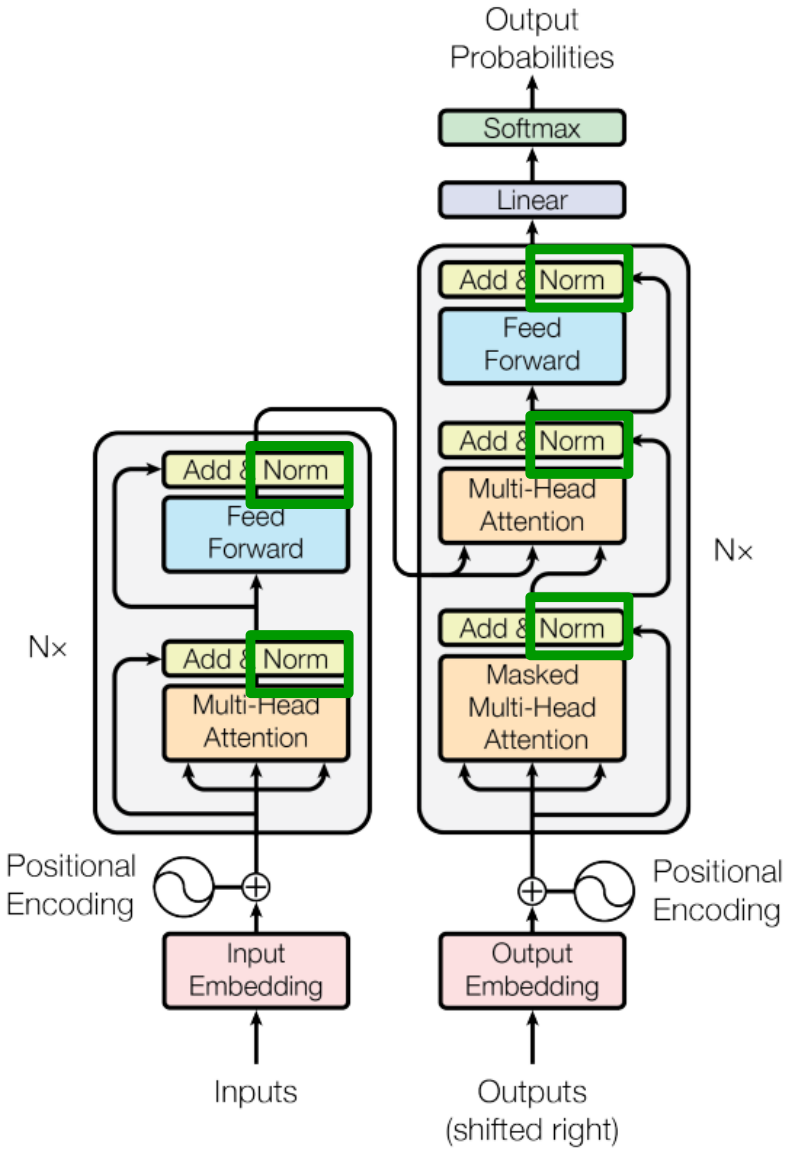


Figure 15: Transformer Architecture [2].

4. Transformers in Natural Language Processing

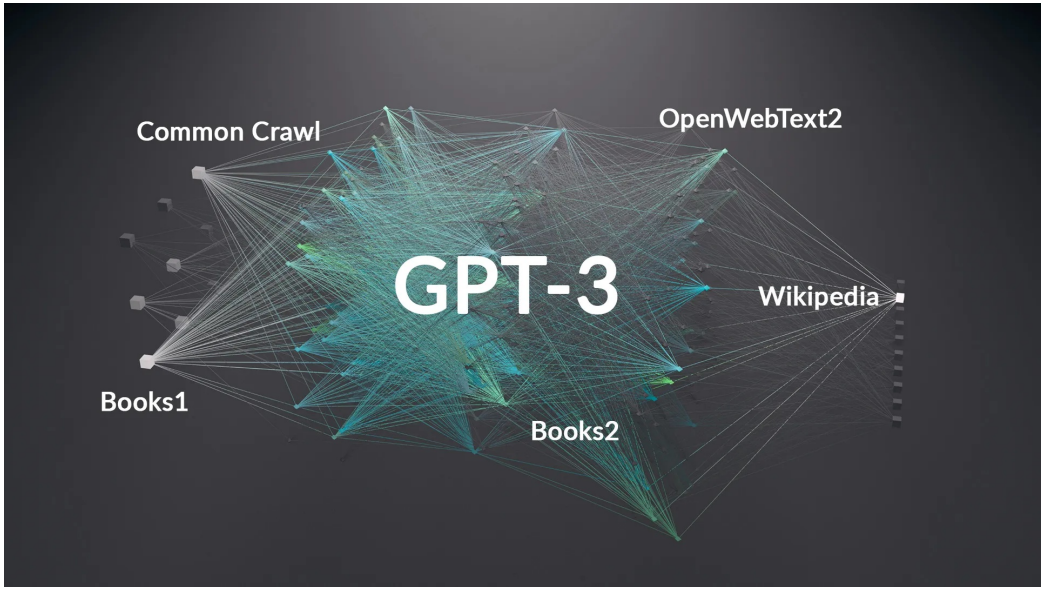


Image sources:
https://cdn-images-1.medium.com/max/428/1*lMlwRSQfzaYw1ckgS-jgZw.jpeg
https://i0.wp.com/katzlberger.ai/wp-content/uploads/2021/04/GPT-3-Datenquellen.jpg?resize=1400%2C800&ssl=1
(call dates: 20.07.22)

4.1 BERT: Bidirectional Encoder Representations from Transformers

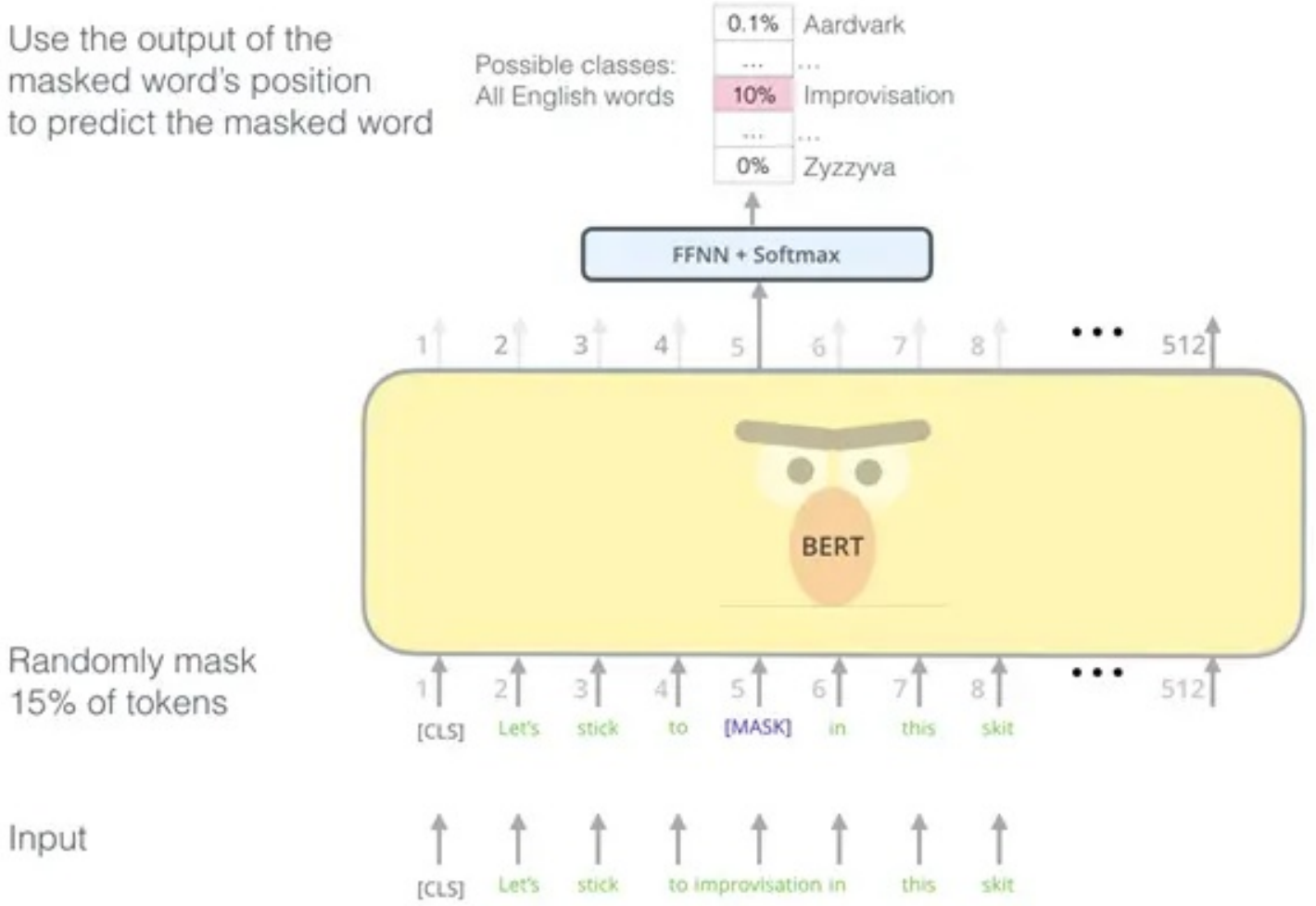


Figure 16: BERT Architecture [8].

Image source: https://www.alexanderthamm.com/wp-content/uploads/High-level-architecture-of-BERT-1.png.webp (call date: 20.07.22)

4.1 BERT: Bidirectional Encoder Representations from Transformers

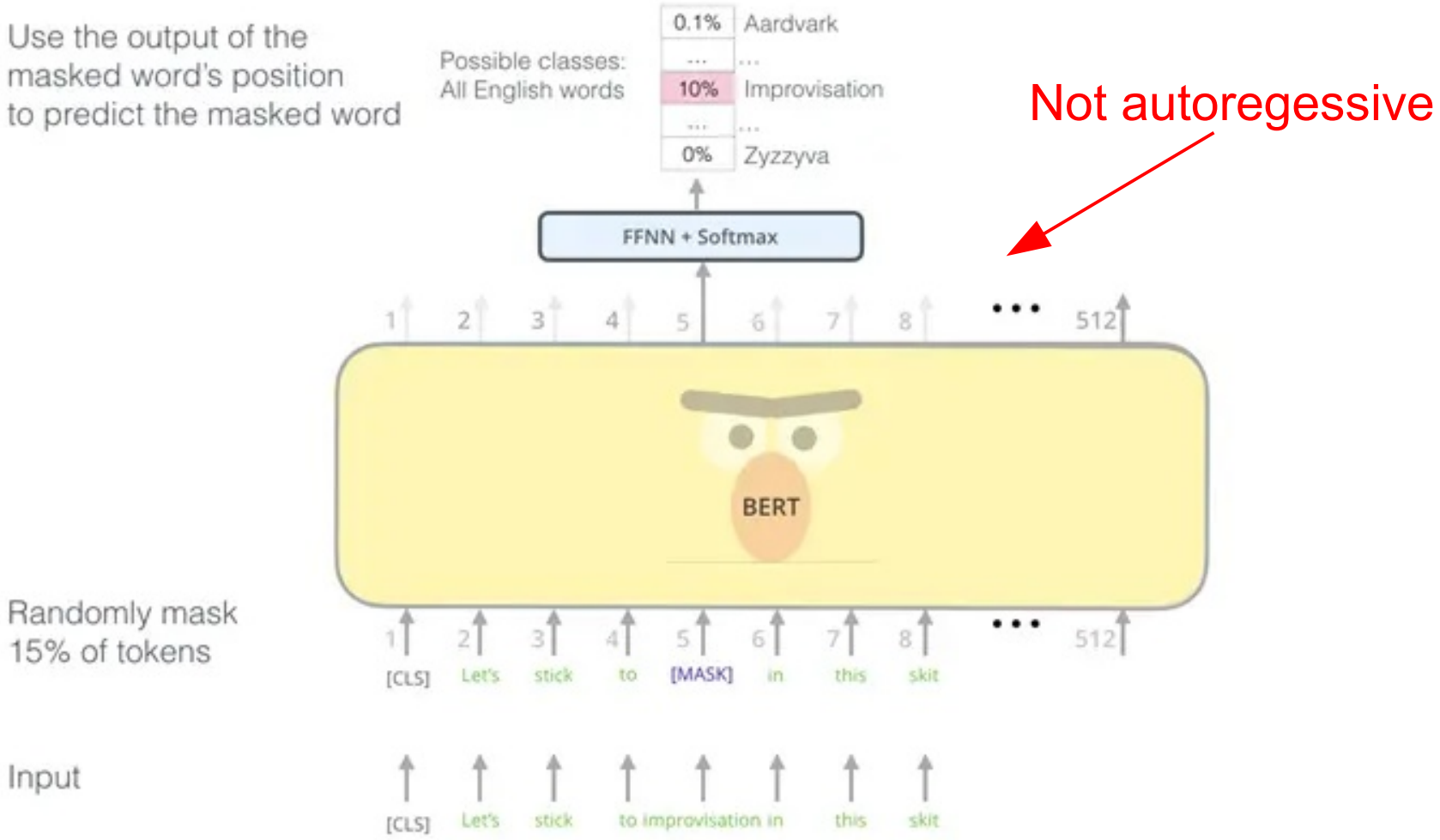


Figure 16: BERT Architecture [8].

Image source: https://www.alexanderthamm.com/wp-content/uploads/High-level-architecture-of-BERT-1.png.webp (call date: 20.07.22)

4.1 BERT: Bidirectional Encoder Representations from Transformers

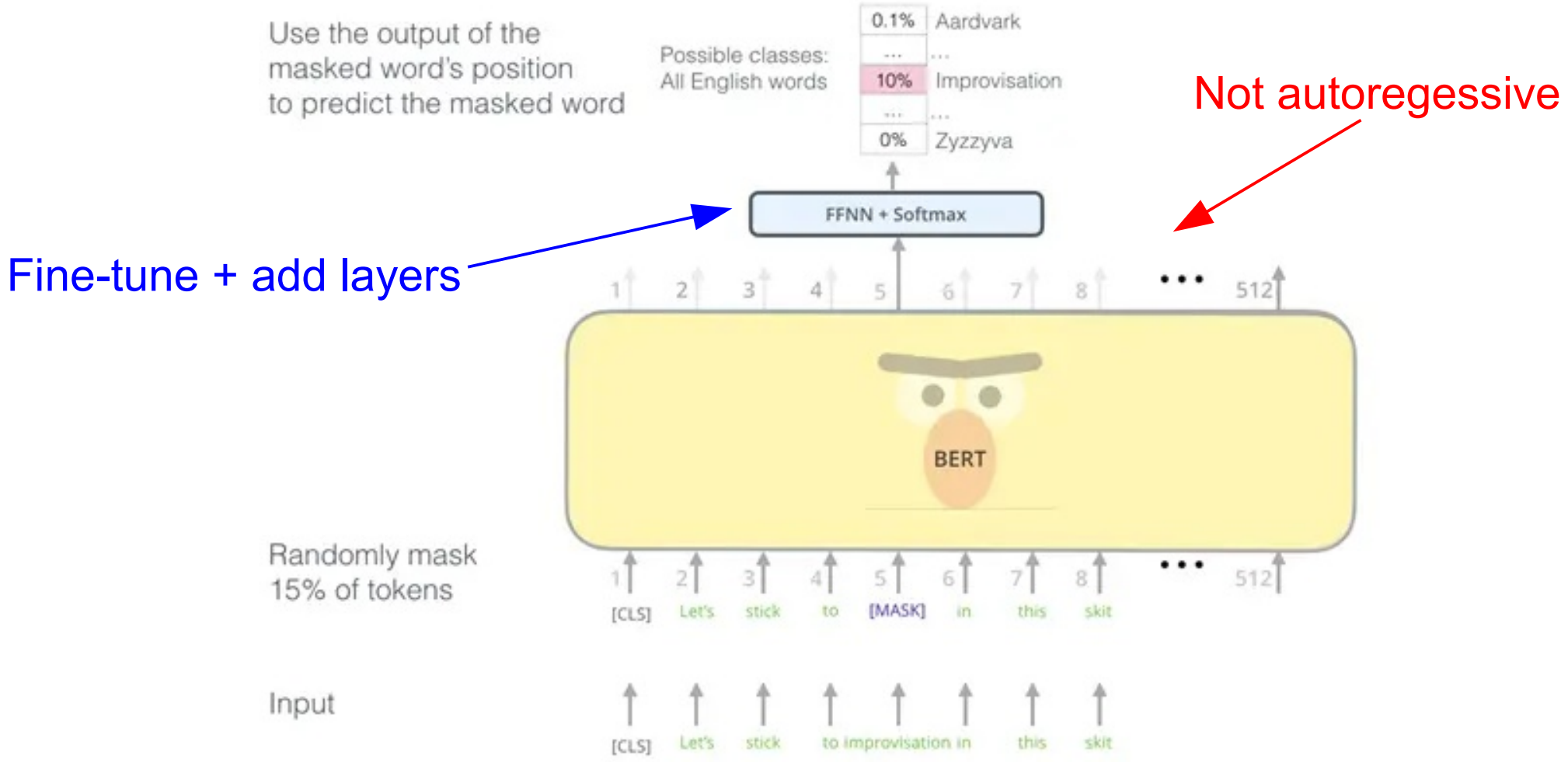


Figure 16: BERT Architecture [8].

Image source: https://www.alexanderthamm.com/wp-content/uploads/High-level-architecture-of-BERT-1.png.webp (call date: 20.07.22)

4.2 GPT-3: Generative Pre-trained Transformer 3

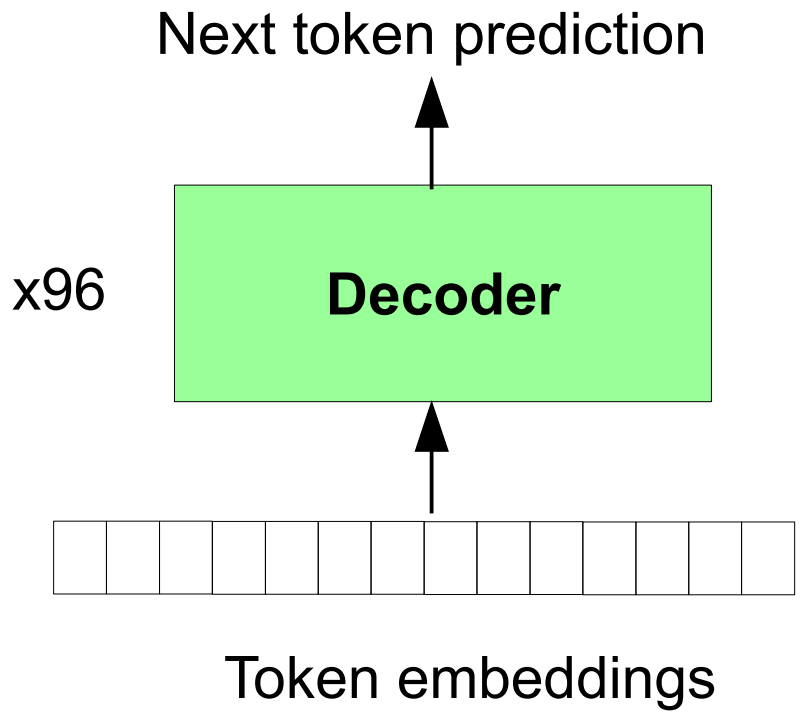


Figure 17: GPT-3 Overview [9].

4.2 GPT-3: Generative Pre-trained Transformer 3

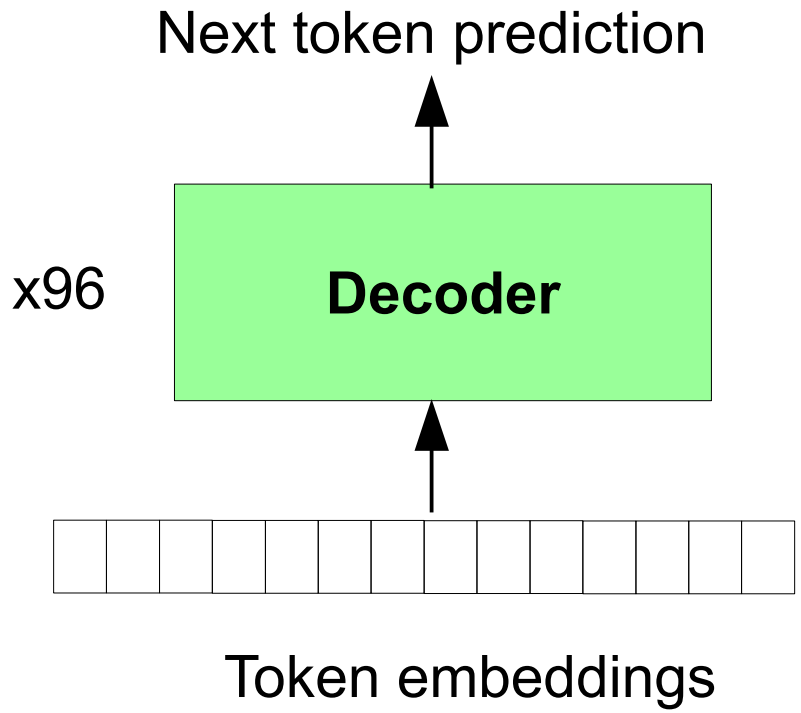


Figure 17: GPT-3 Overview [9].

Image sources:
https://upload.wikimedia.org/wikipedia/commons/thumb/8/80/Wikipedia-logo-v2.svg/250px-Wikipedia-logo-v2.svg.png
https://cdn-icons-png.flaticon.com/512/29/29302.png
https://cdn-icons-png.flaticon.com/512/493/493805.png
(call dates: 20.07.22)

4.2 GPT-3: Generative Pre-trained Transformer 3

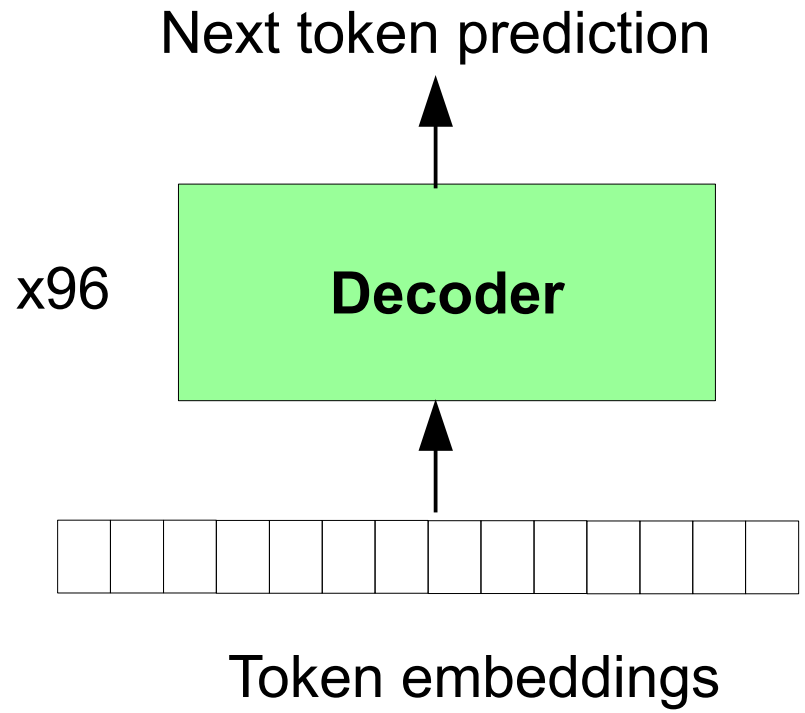


Figure 17: GPT-3 Overview [9].

Fine-tuning: **No gradient updates**
Input:
Translate English to French:
cheese =>

Image sources:
https://upload.wikimedia.org/wikipedia/commons/thumb/8/80/Wikipedia-logo-v2.svg/250px-Wikipedia-logo-v2.svg.png
https://cdn-icons-png.flaticon.com/512/29/29302.png
https://cdn-icons-png.flaticon.com/512/493/493805.png
(call dates: 20.07.22)

4.3 BERT vs. GPT-3

	BERT	**GPT-3**
release year	2018	2020
#params	340M	175B
#attention heads	16	96
hidden size	1024	12288
stacked transformer blocks	encoder (non-autoregressive)	decoder (autoregressive)
attention direction	bidirectional	left-to-right
#stacked transformer blocks	12	96
input	512 token embeddings	2048 token embeddings
output	512 token embeddings + classifiers	1 token embedding + classifier

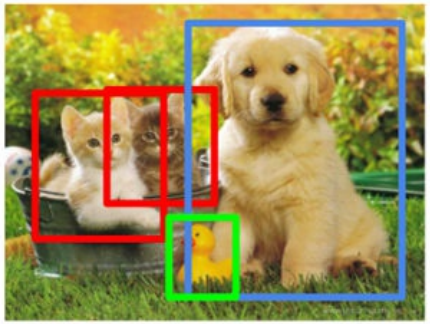
Table 1: Differences between BERT and GPT-3 [8,9].

5. Transformers in Computer Vision

Classification

CAT

Object Detection



CAT, DOG, DUCK

Image source:
https://res.cloudinary.com/dyd911kmh/image/upload/f_auto,q_auto:best/v1522766480/1_6j34dAOTijqP6HDFnjxPFA_udggex.png (call date: 20.07.22)

5.1.1 ViT: Vision Transformer – Image Classification

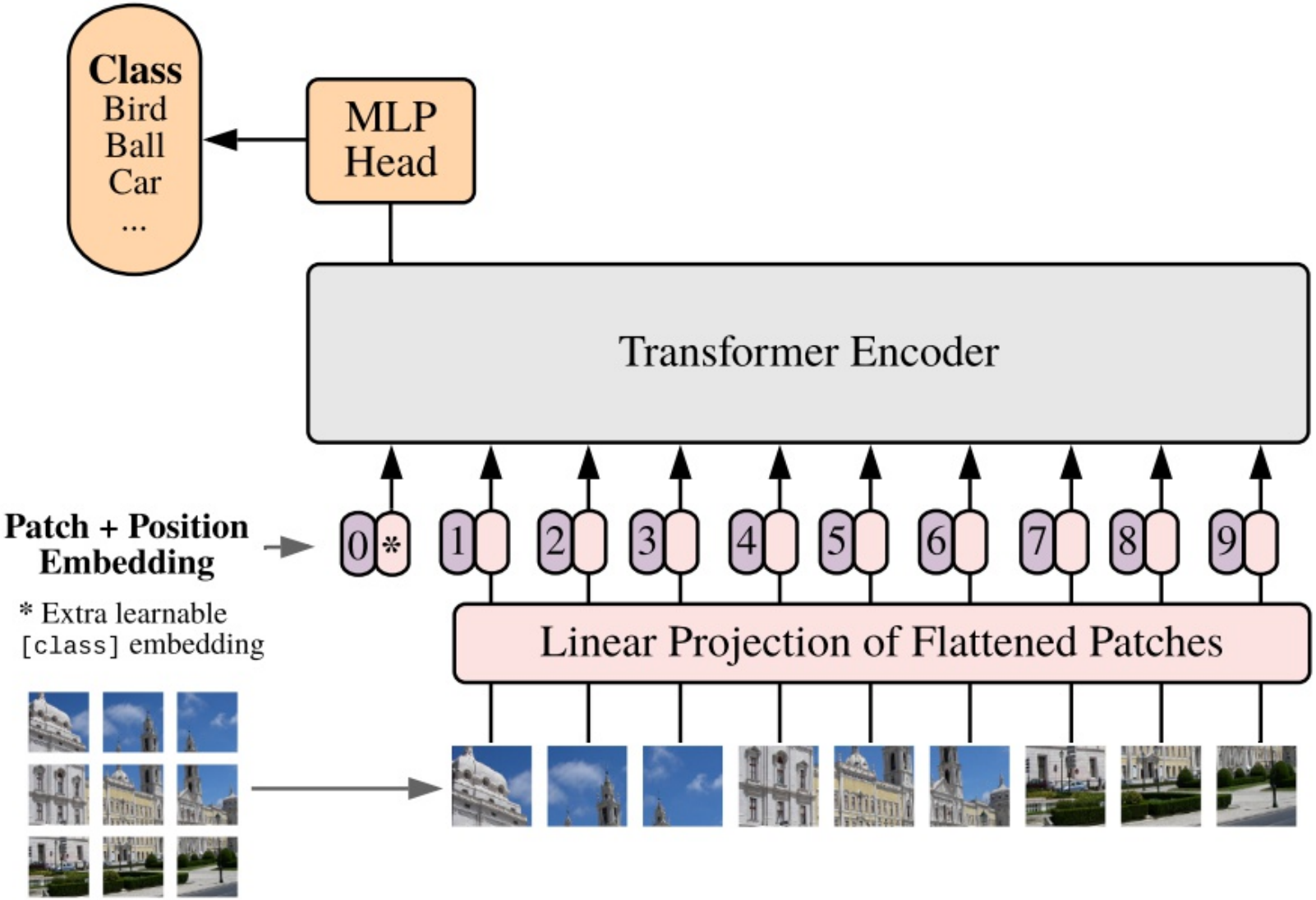


Figure 18: ViT Architecture [3].

5.1.2 ViT: Vision Transformer – Evaluation

	ViT-H/14	ResNet152x4
ImageNet	**88.55**	87.54
ImageNet ReaL	**90.72**	90.54
CIFAR-10	**99.50**	99.37
CIFAR-100	**94.55**	93.51
Oxford-IIIT Pets	**97.56**	96.62
Oxford Flowers-102	**99.68**	99.63
VTAB (19 tasks)	**77.63**	76.29
#param	632M	14M
TPUv3-core-days	2.5k	9.9k

Table 2: Performance of the ViT and ResNet [3].

5.1.3 ViT: Vision Transformer – “Attention in an image”

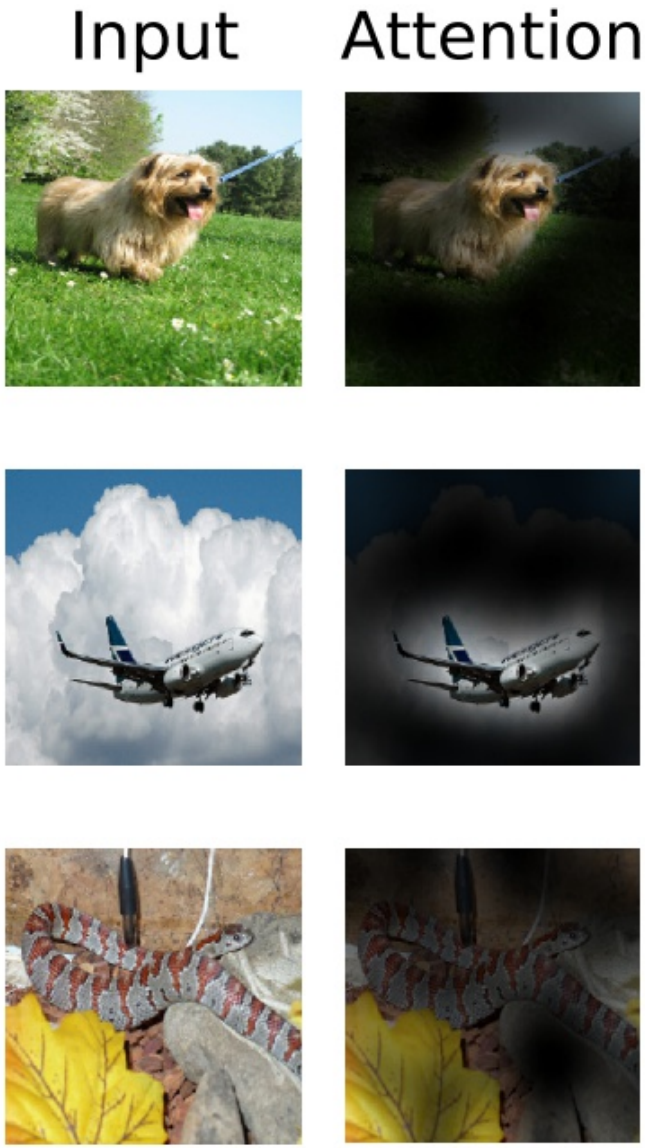


Figure 19: Visualization of self-attention in an image [3].

Side note: Feature Attribution with Occlusion

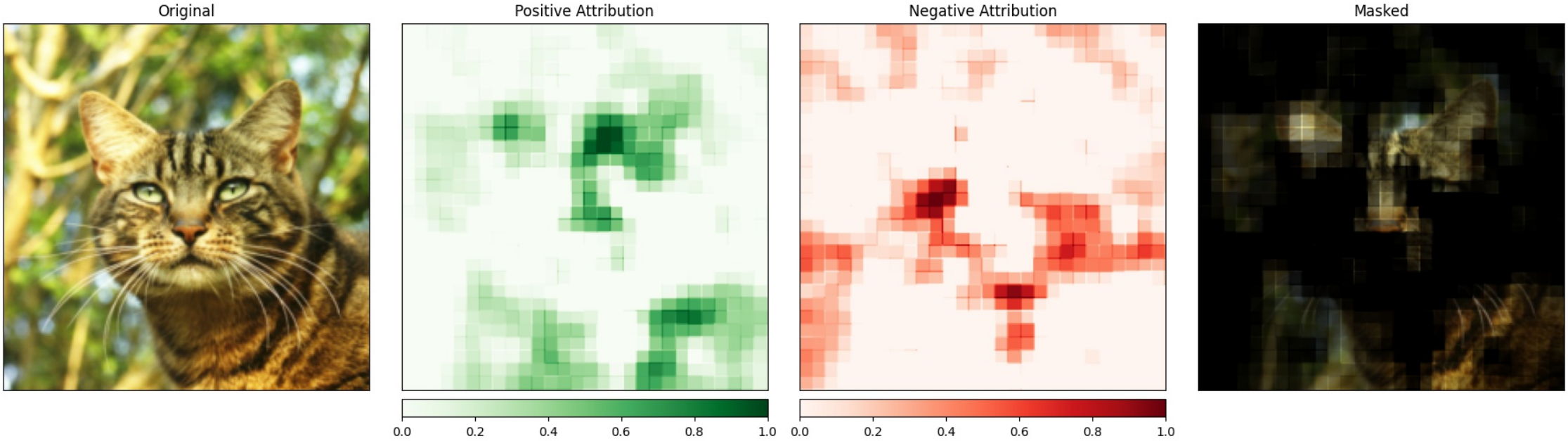


Image source:
https://www.youtube.com/watch?v=Am2EF9CLu-g&list=PL_lsbAsL_o2CTlGHgMxNrKhzP97BaG9ZN&index=7&ab_channel=PyTorch (call date: 26.01.23)

5.1.4 ViT: Vision Transformer – “Learn to be a CNN”

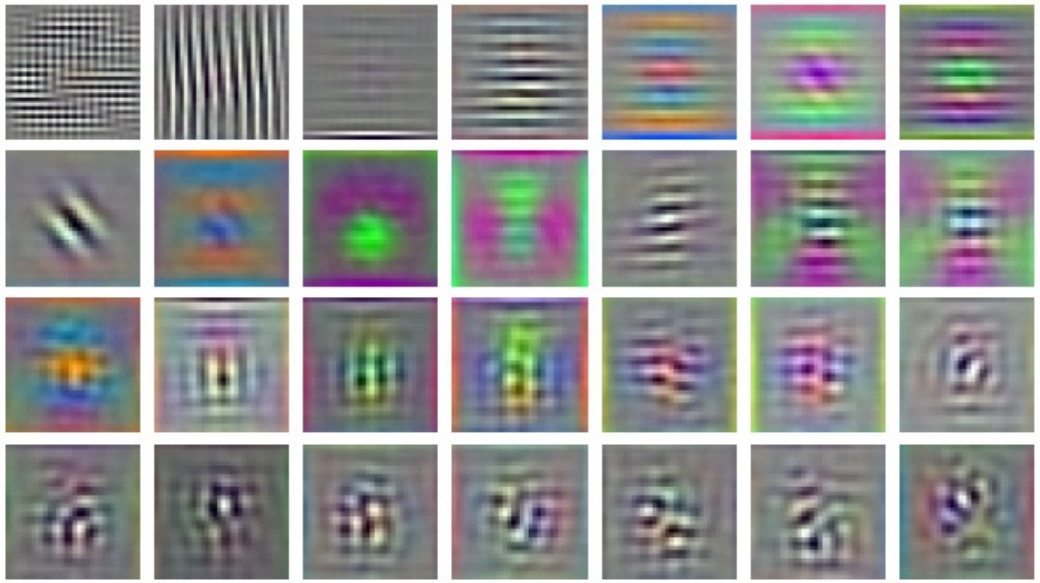


Figure 20: Learned patterns to which the self-attention mechanism reacts to [3].

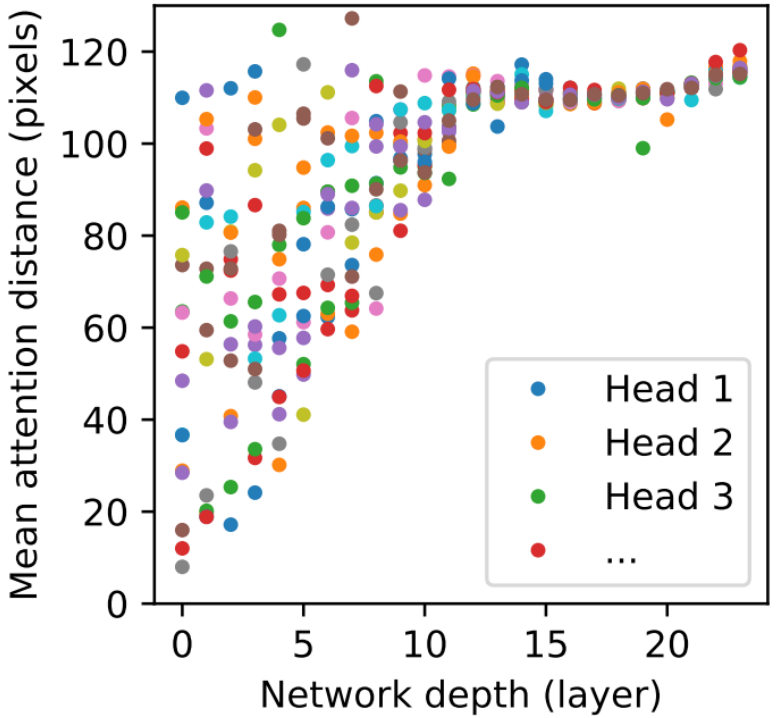


Figure 21: Mean attention distance of each layer [3].

Side note: Receptive Field in a CNN

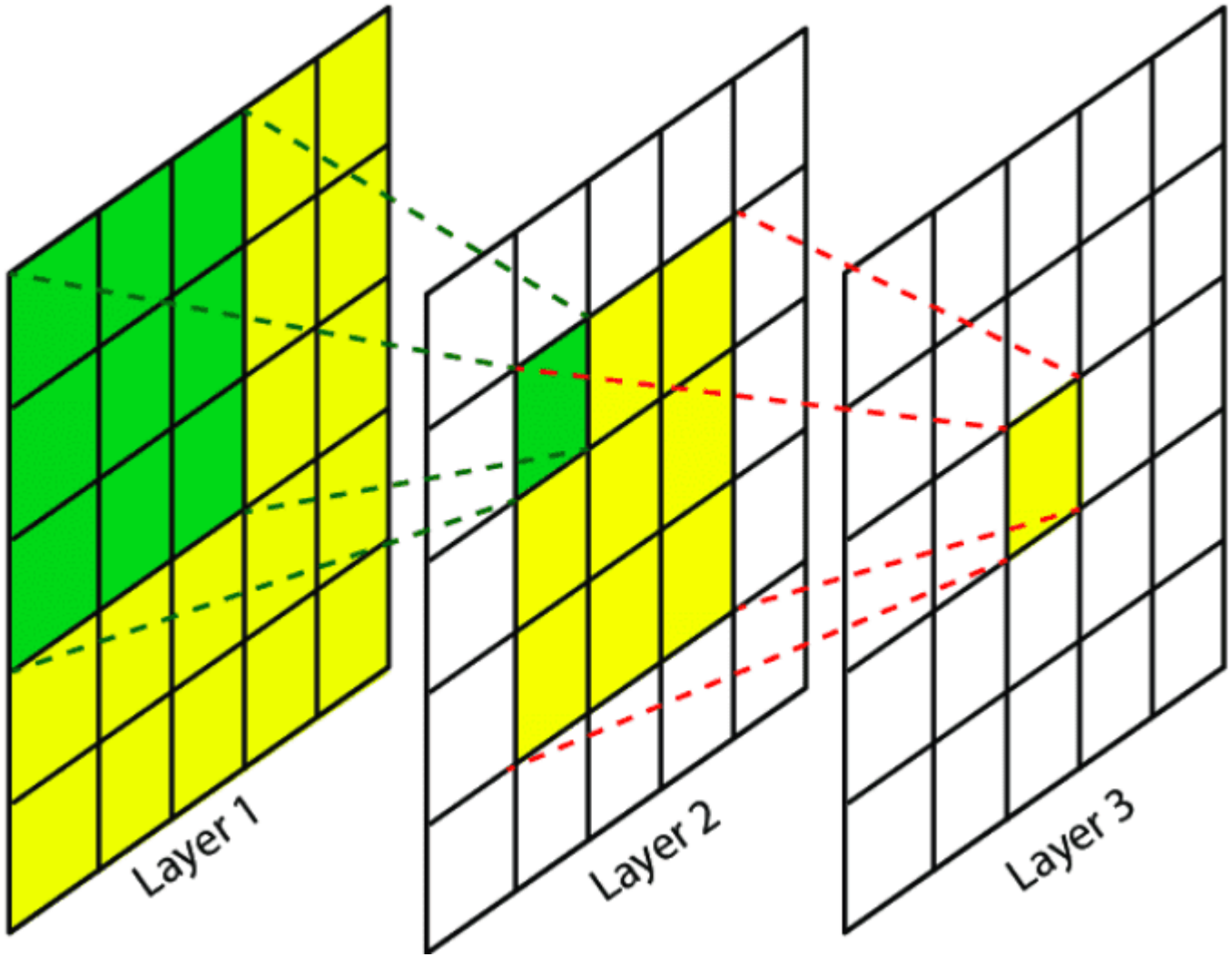


Image source:
https://theaisummer.com/static/490be17ee7f19b78003c3fdf5a6bbafc/83b75/receptive-field-in-convolutional-networks.png (call date: 20.07.22)

5.2.1 DETR: DEtection Transformer – Object detection

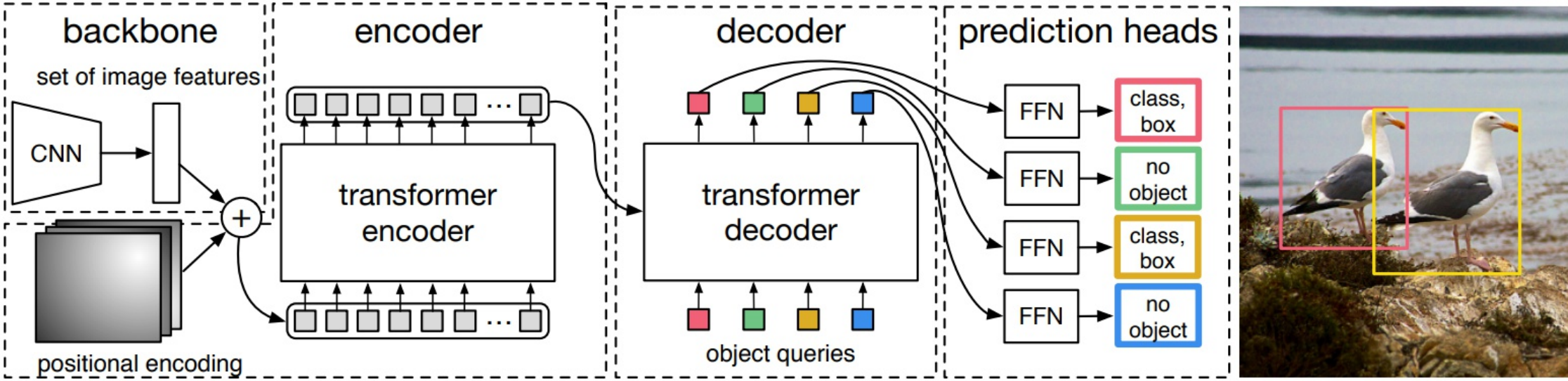


Figure 22: DETR architecture [4].

5.2.1 DETR: DEtection Transformer – Object detection

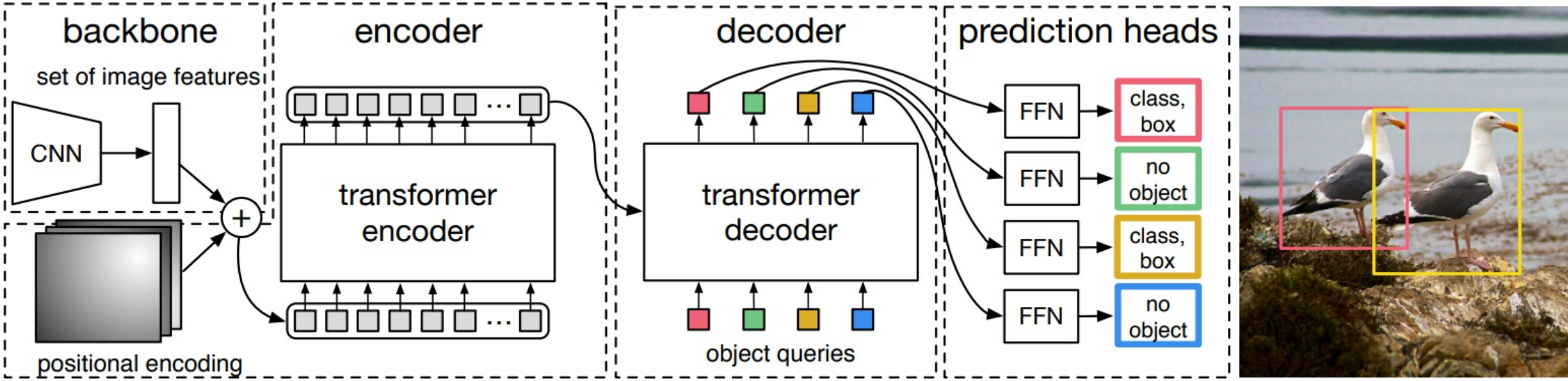


Figure 22: DETR architecture [4].

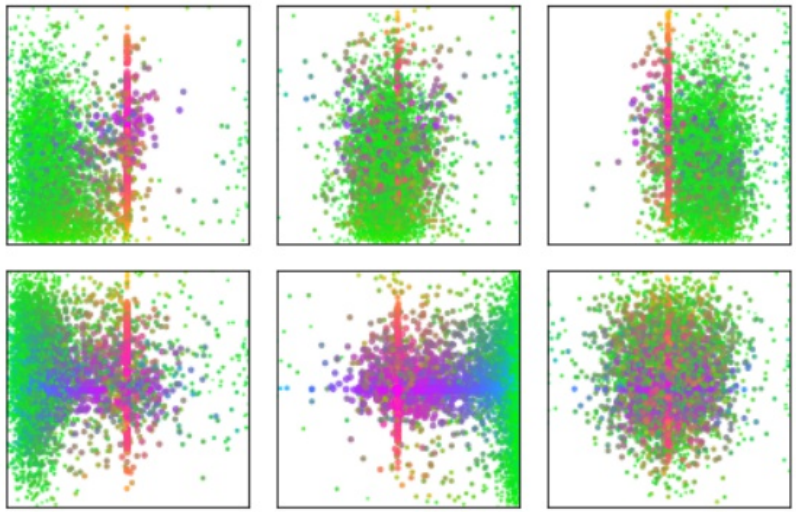


Figure 23: Object queries [4].

5.2.1 DETR: DEtection Transformer – Object detection

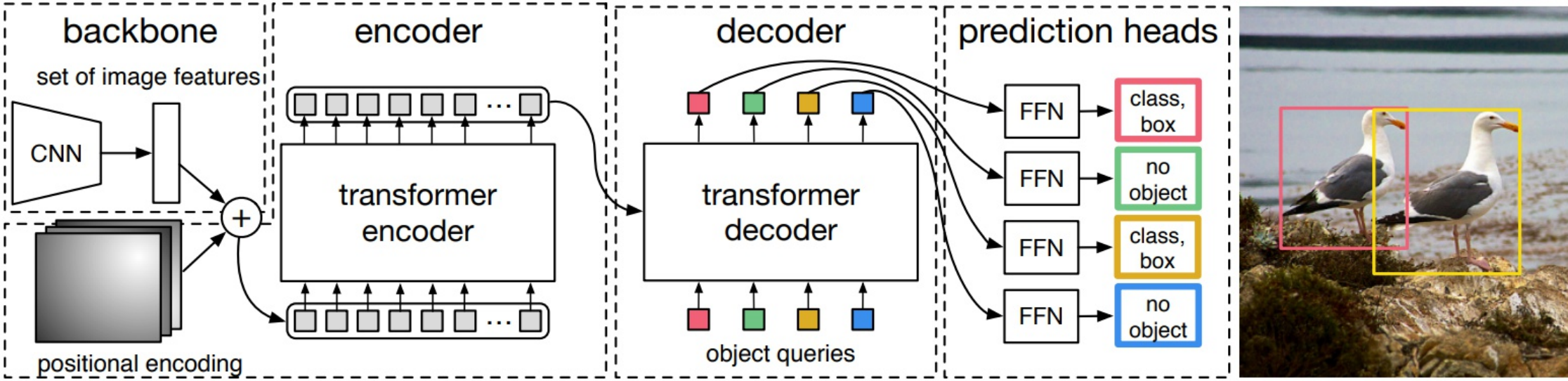


Figure 22: DETR architecture [4].

Training:

Loss = best matching loss between target and prediction → Hungarian Algorithm

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$$

N = amount of boxes

5.2.1 DETR: DEtection Transformer – Object detection

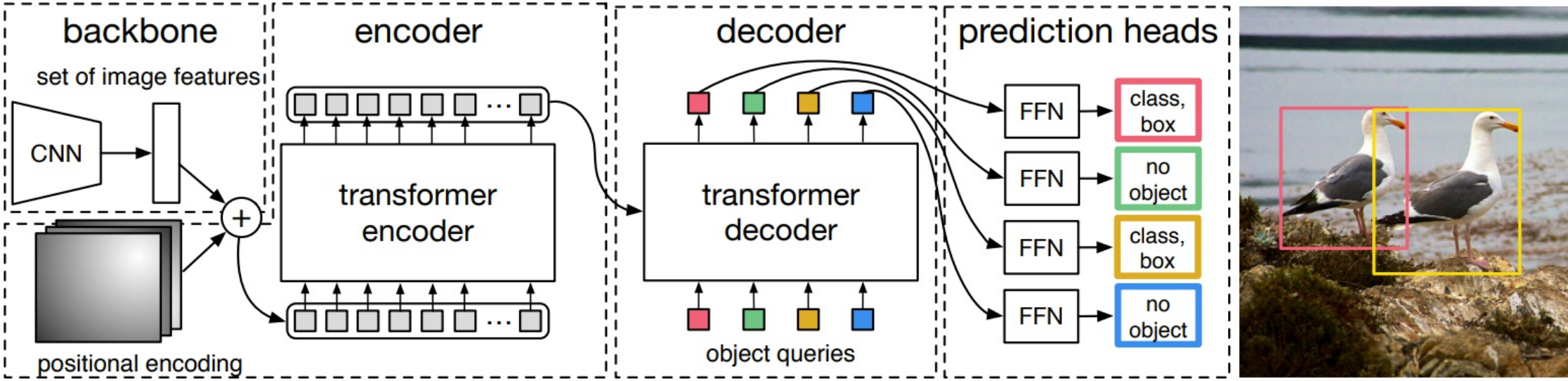


Figure 22: DETR architecture [4].

Training:

Loss = best matching loss between target and prediction → Hungarian Algorithm

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$$

N = amount of boxes

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \left[-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{\{c_i \neq \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \right]$$

Classification loss (cross entropy loss)

Box loss (intersection over union loss)

5.2.2 DETR: DEtection Transformer
“Attention-head communication”

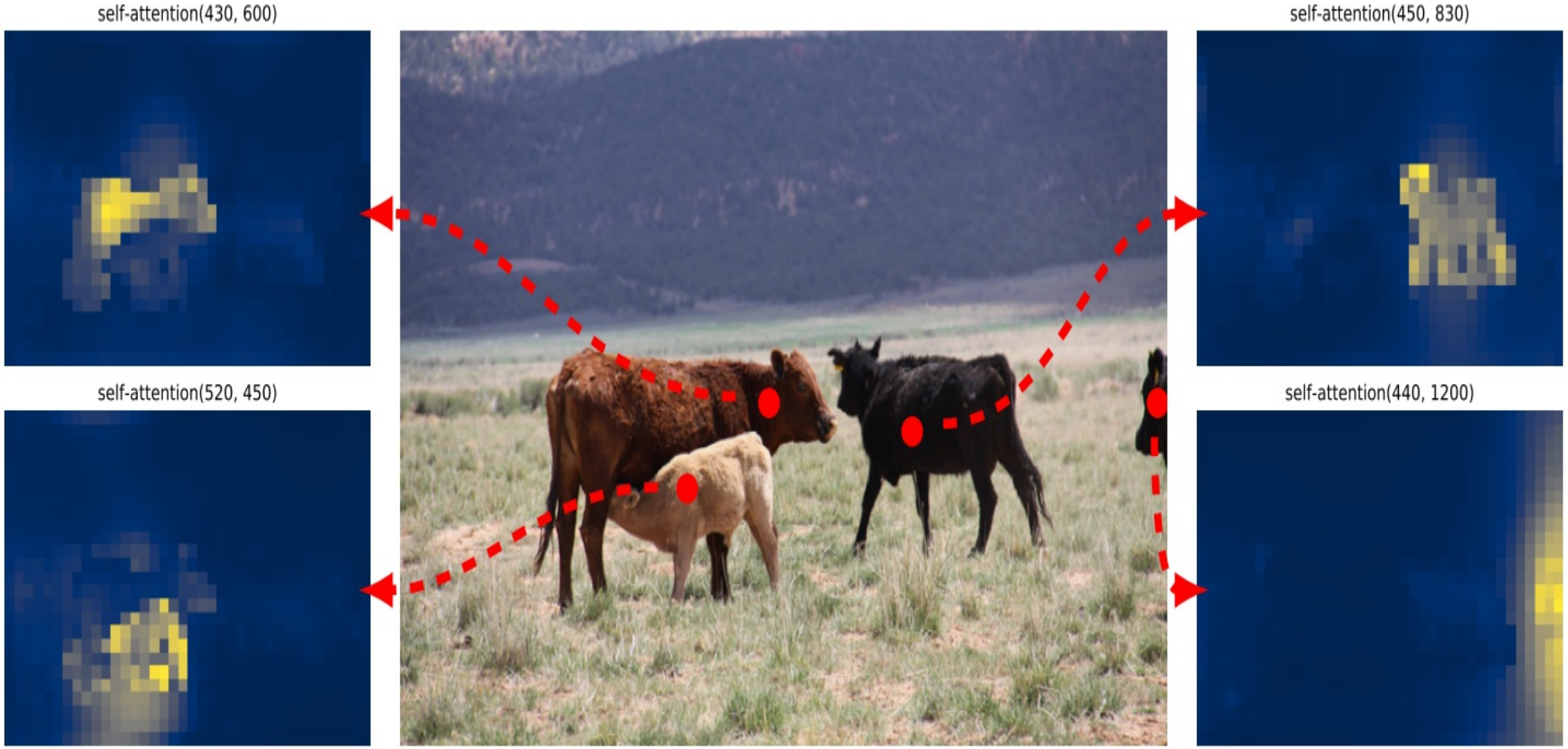


Figure 24: Different attention-heads attend to different objects in the image [4].

This Person Does Not Exist [25].

6. GAN based on a Transformer

Without transposed convolutions!!!

Image source: https://www.thispersondoesnotexist.com/image (call date: 20.07.22)

6.1 TransGAN: Architecture

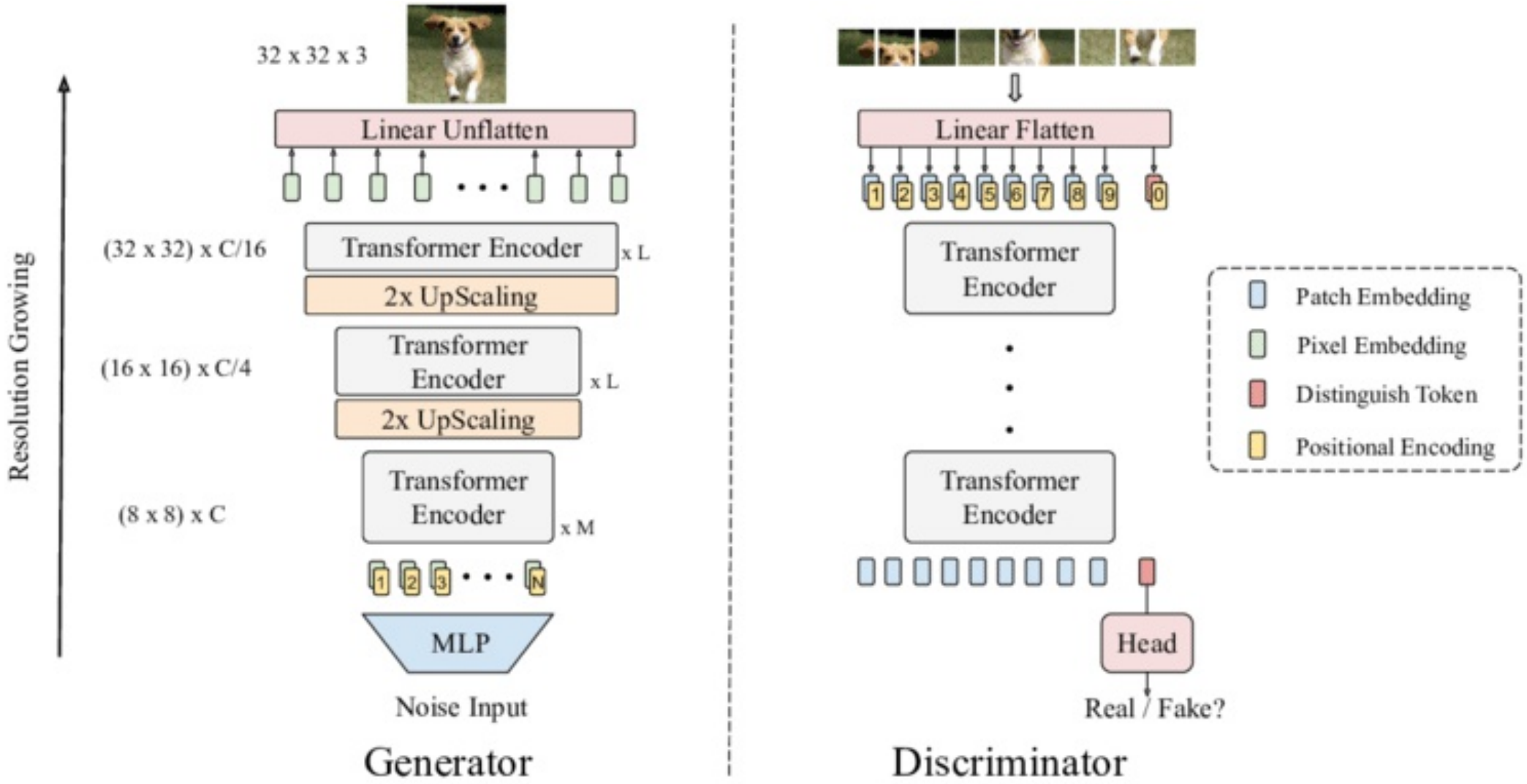


Figure 25: TransGAN architecture [5].

Side note: PixelShuffle

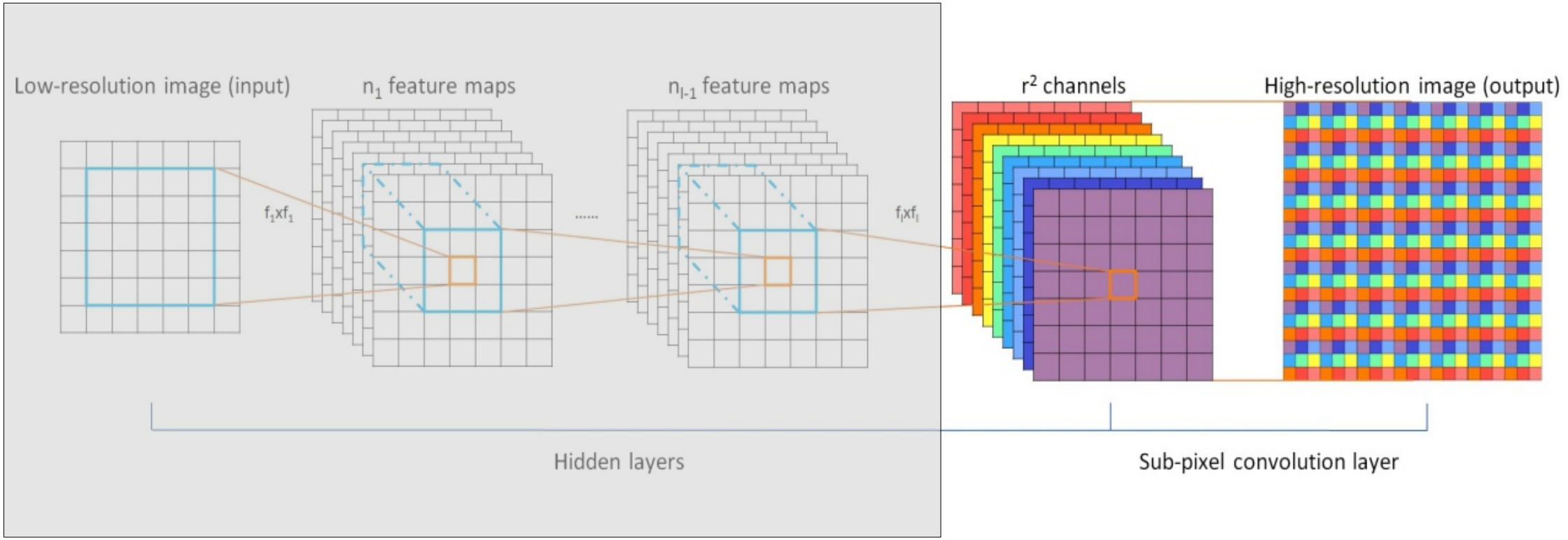


Image source:
Shi, Wenzhe et al. “Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016): 1874-1883.

6.2 TransGAN: Evaluation

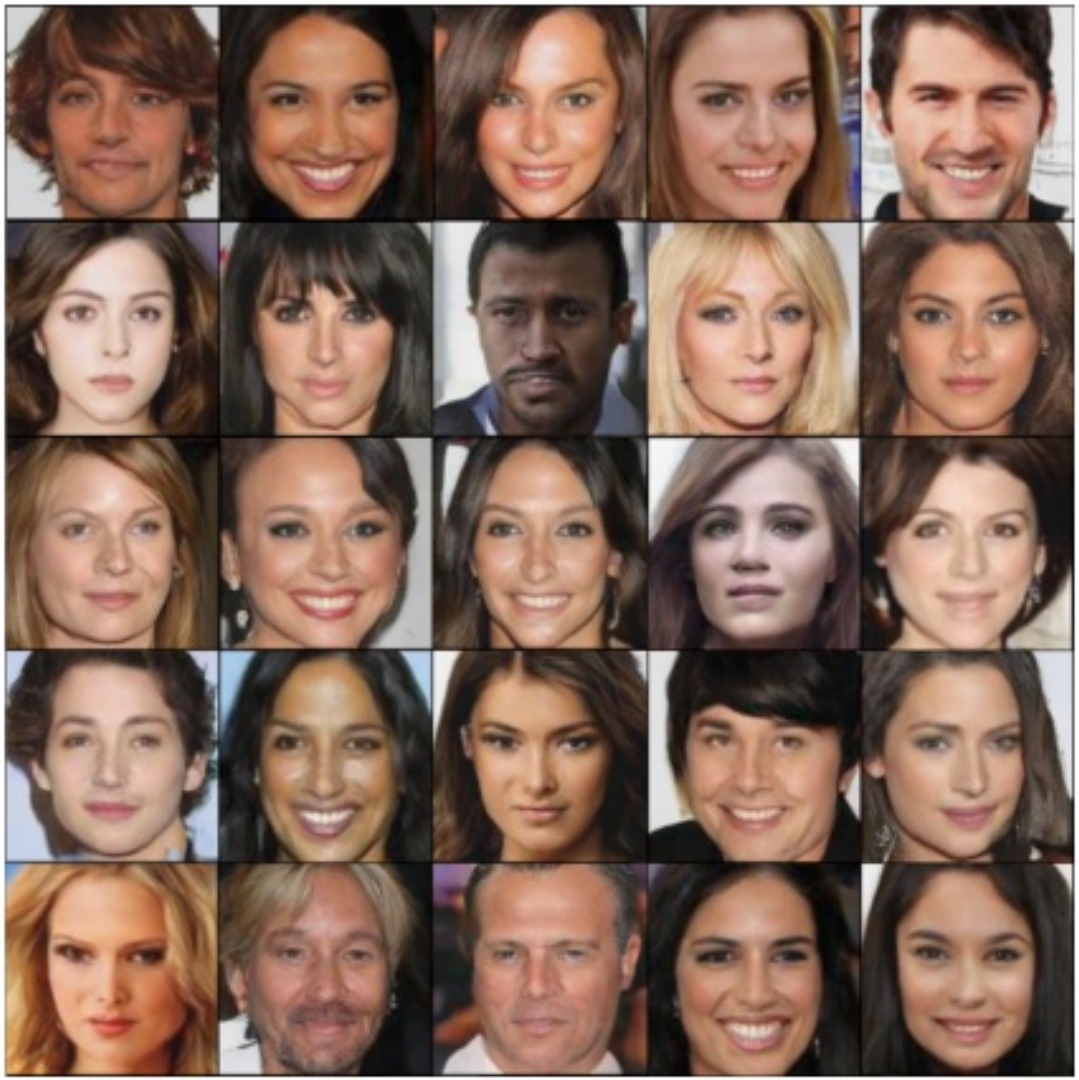


Figure 26: Generated images by the TransGAN [5].

6.2 TransGAN: Evaluation

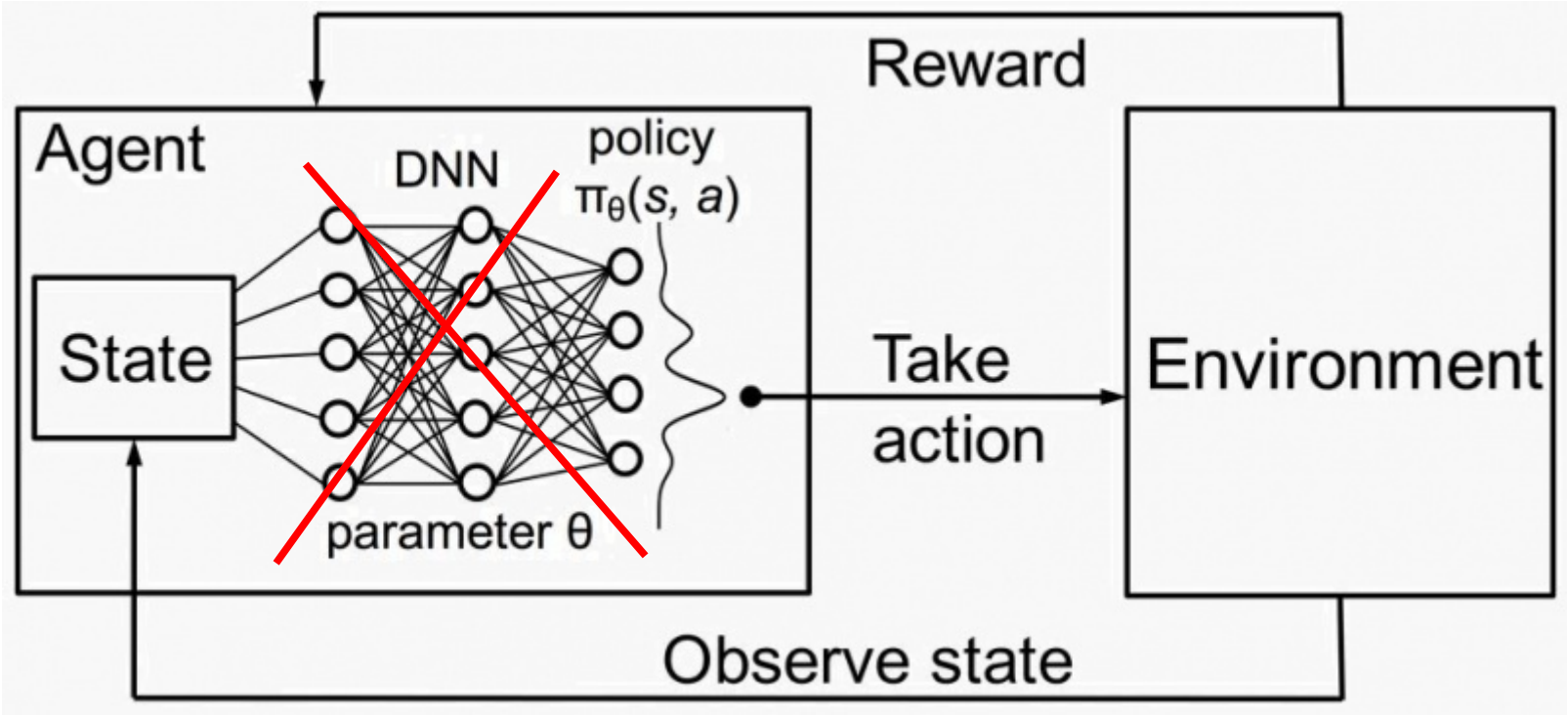
Methods	WGAN-GP		AutoGAN		StyleGAN-V2		TransGAN	
	IS ↑	FID ↓	IS ↑	FID ↓	IS ↑	FID ↓	IS ↑	FID ↓
Original	**6.49**	39.68	8.55	**12.42**	9.18	11.07	8.36	22.53
+ DiffAug [69]	6.29	**37.14**	**8.60**	12.72	**9.40**	**9.89**	**9.02**	**9.26**

Table 3: Performance of the TransGAN compared with state-of-the-art GANs [5].

IS: Inception score “high = good”

FID: Frechet Inception Distance “low = good”

7. Transformers in Deep Reinforcement Learning



Now: Transformer

Image source:
https://miro.medium.com/max/1400/1*ZZJ2FJFDNB9W-kdA2CfmTQ.png (call date: 20.07.22)

7.1.1 DTQN: Deep Transformer Q-network

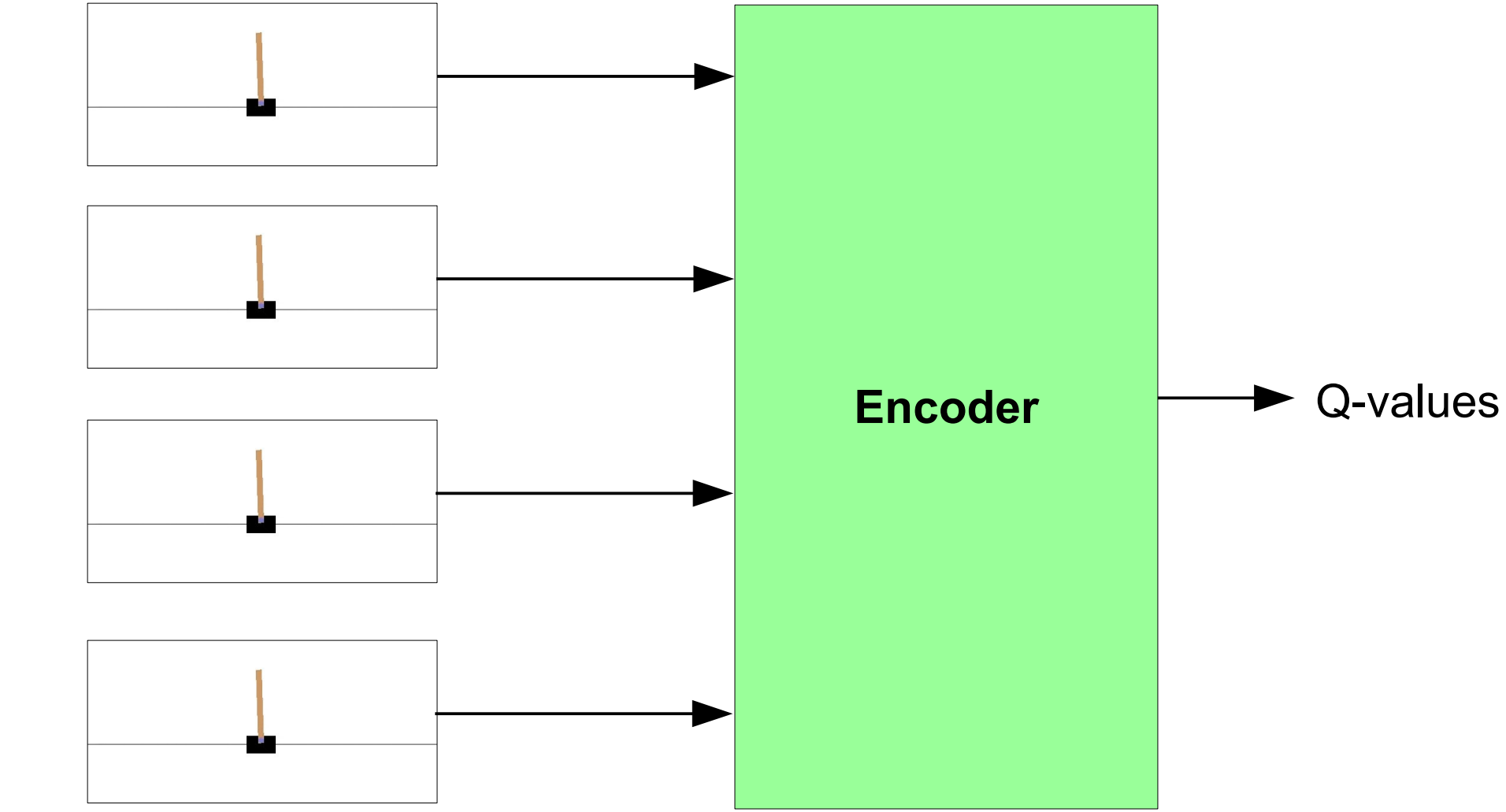


Figure 27: DTQN architecture [6].

Image source:
https://miro.medium.com/freeze/max/600/1*C6T-7ytta7KRbejnCfOtXQ.gif (call date: 20.07.22)

7.1.2 DTQN: Evaluation

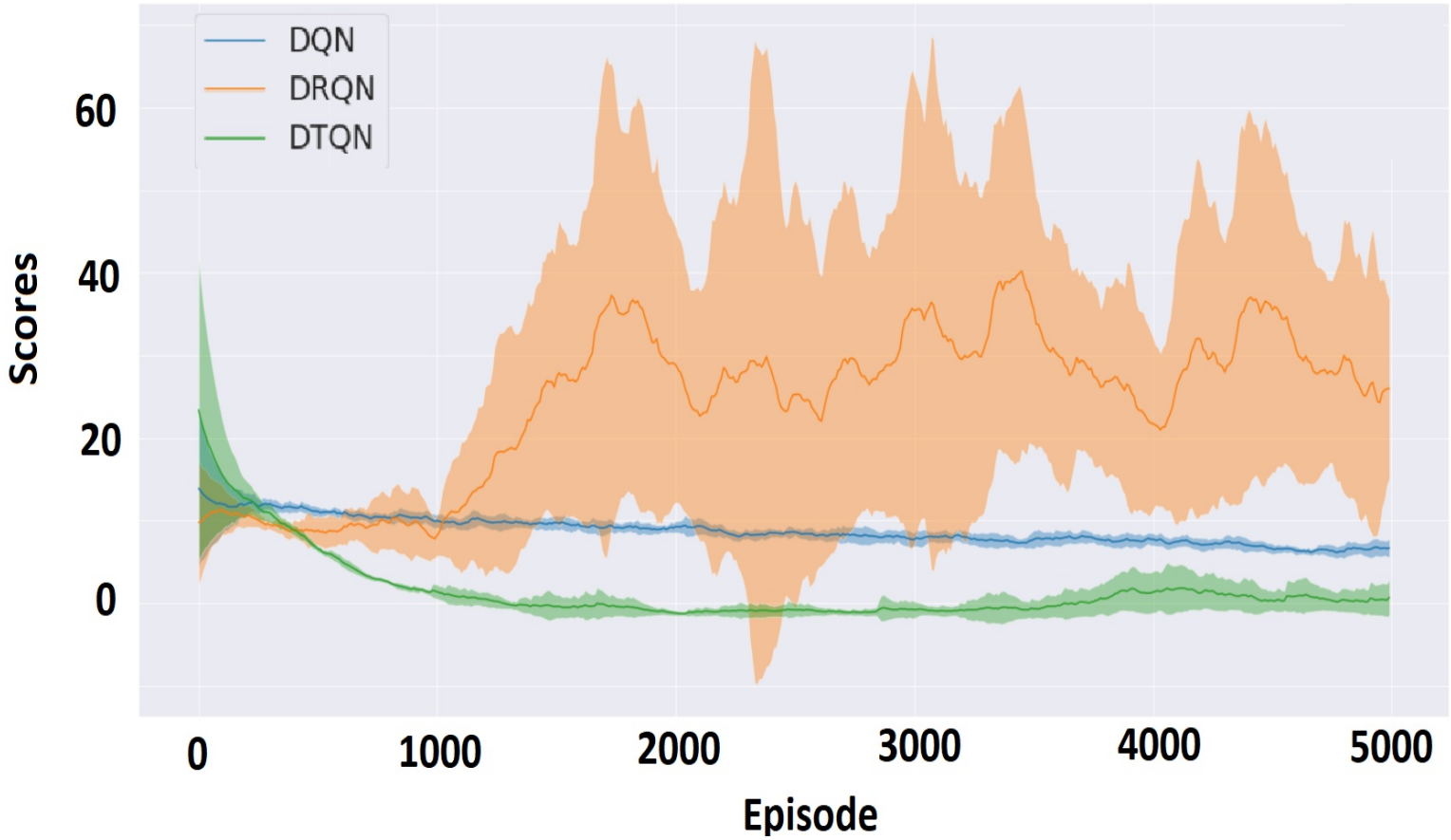


Figure 28: DTQN performance [6].

7.2.1 Decision Transformer: Architecture

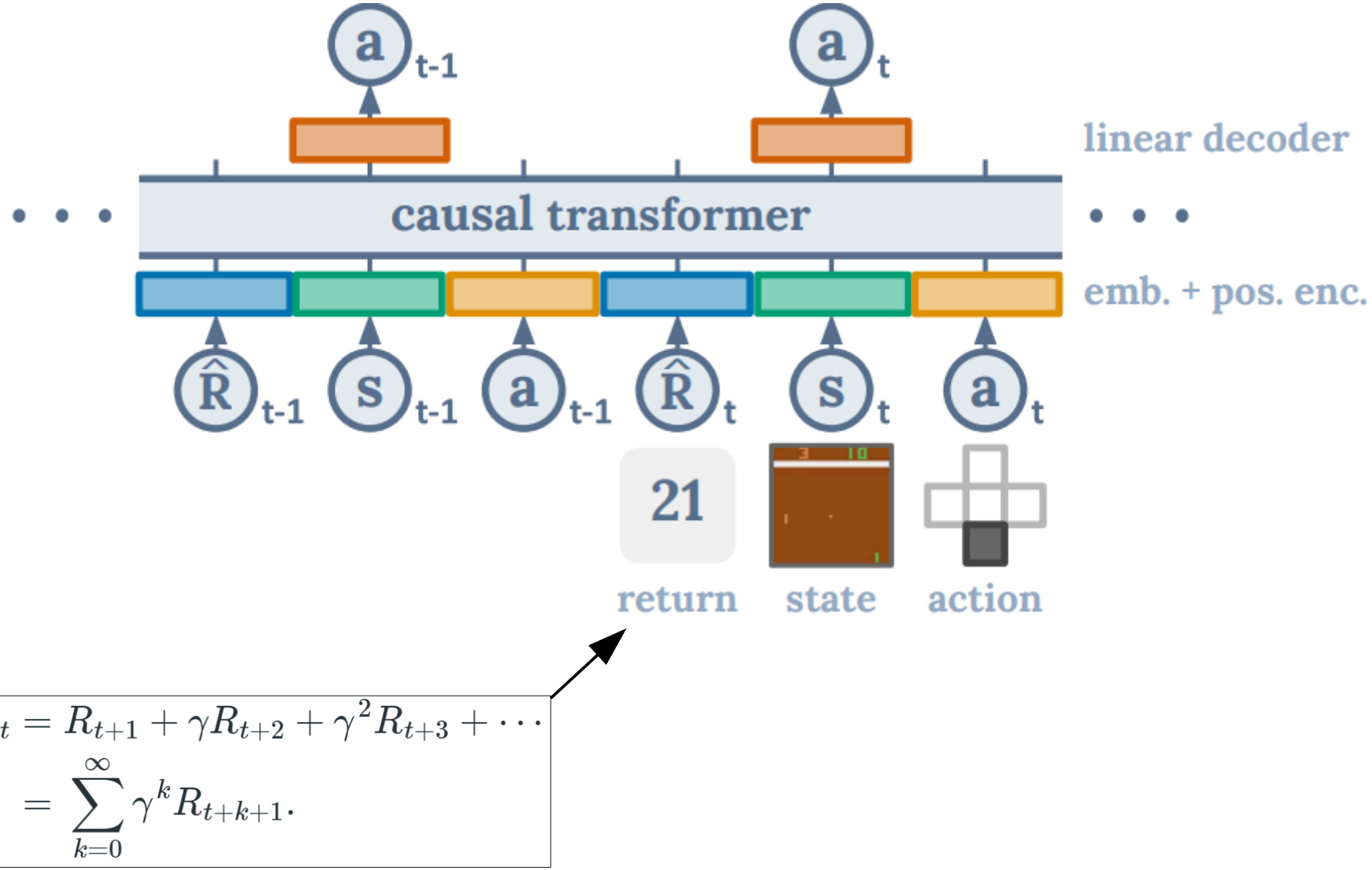
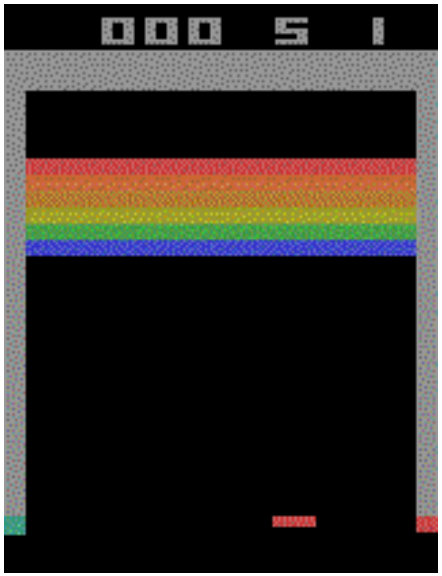
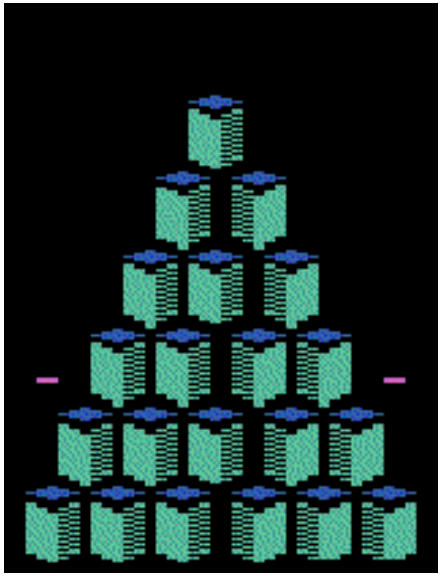


Figure 29: Decision Transformer architecture [7].

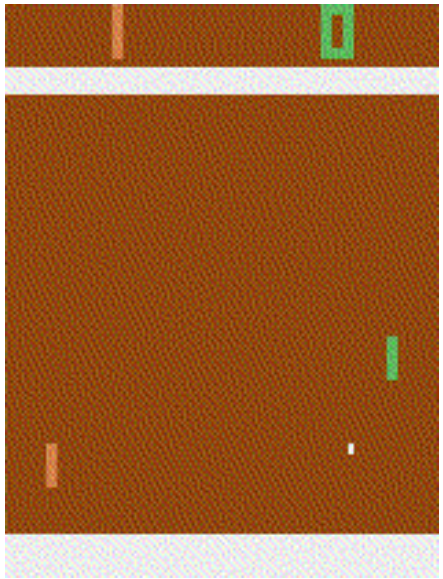
7.2.2 Decision Transformer: Evaluation



Breakout



Qbert



Pong

Seaquest

Figure 30: Evaluated environments.

Image sources:
https://www.gymlibrary.ml/_images/breakout.gif
https://www.gymlibrary.ml/_images/qbert.gif
https://www.gymlibrary.ml/_images/pong.gif
https://www.gymlibrary.ml/_images/seaquest.gif (call dates: 20.07.22)

7.2.3 Decision Transformer: Evaluation

CQL: conservative Q-Learning
QR-DQN: distributional RL
REM: random ensemble mixture

Game	DT	CQL	QR-DQN	REM
Breakout	**267.5 ± 97.5**	211.1	21.1	32.1
Qbert	25.1 ± 18.1	**104.2**	1.7	1.4
Pong	106.1 ± 8.1	**111.9**	20.0	39.1
Seaquest	**2.4 ± 0.7**	1.7	1.4	1.0

Table 4: Performance of the Decision Transformer compared with state-of-the-art DRL methods [7].

7.2.4 Decision Transformer: Why attention?

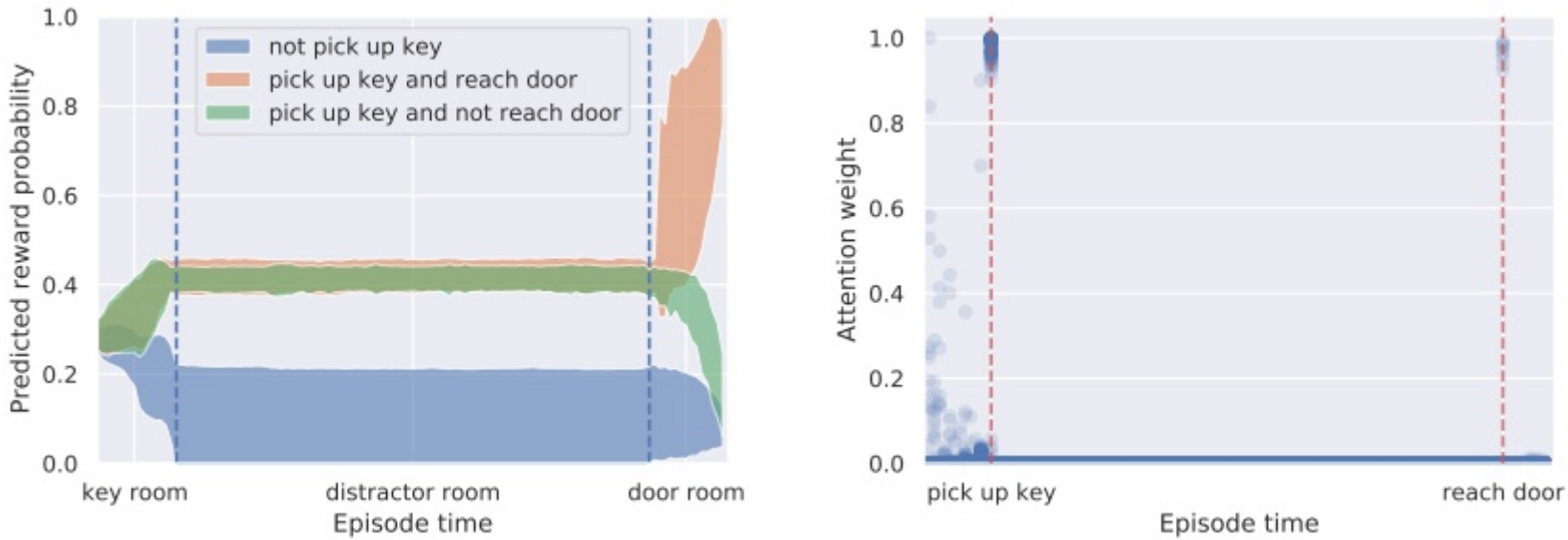
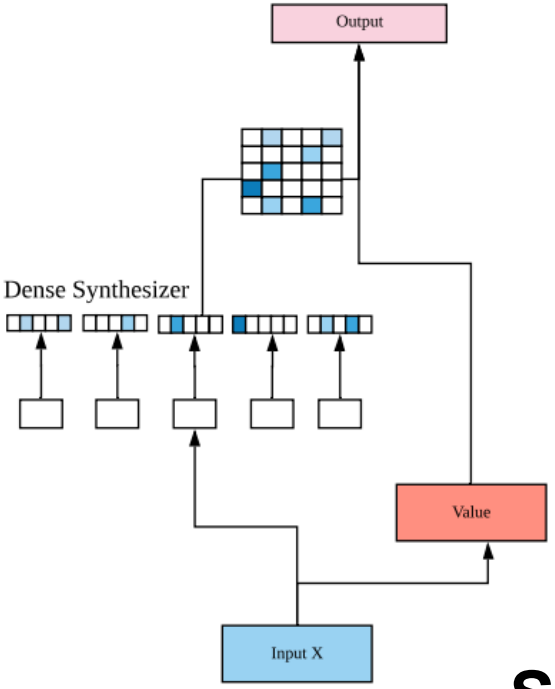
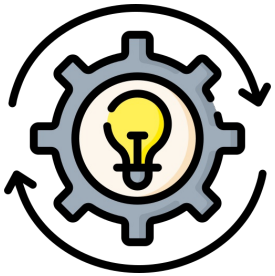
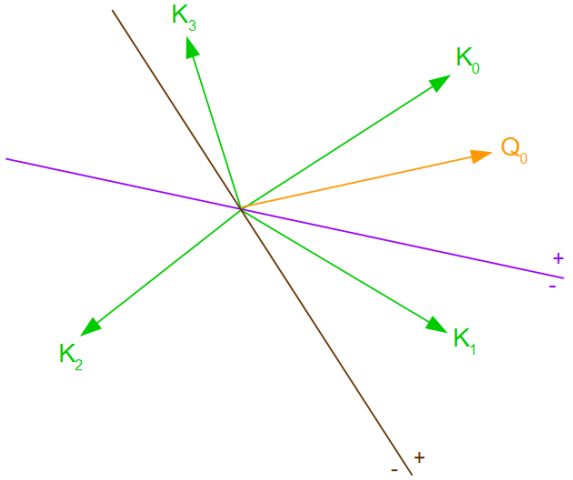


Figure 31: Decision Transformer attend to pivotal events [7].







Reformer

7. Modifications of attention: Synthesizer, Linformer, Reformer

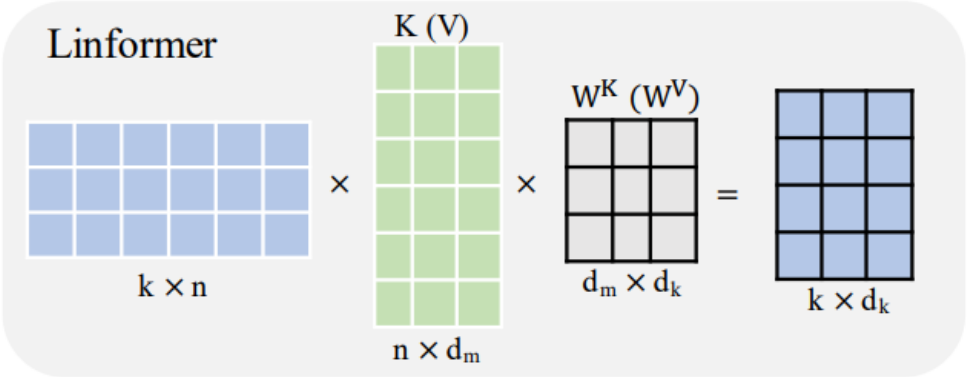
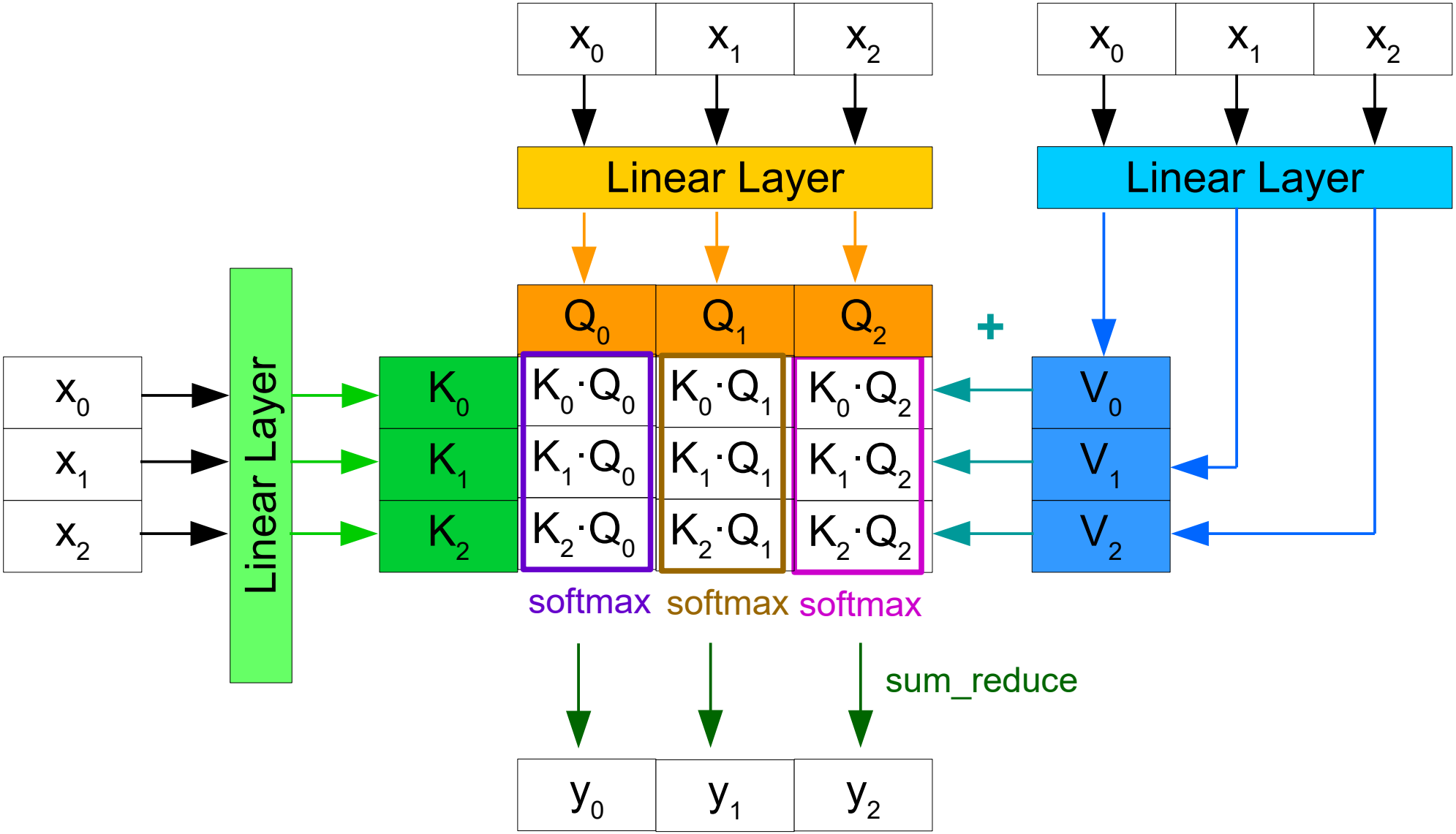
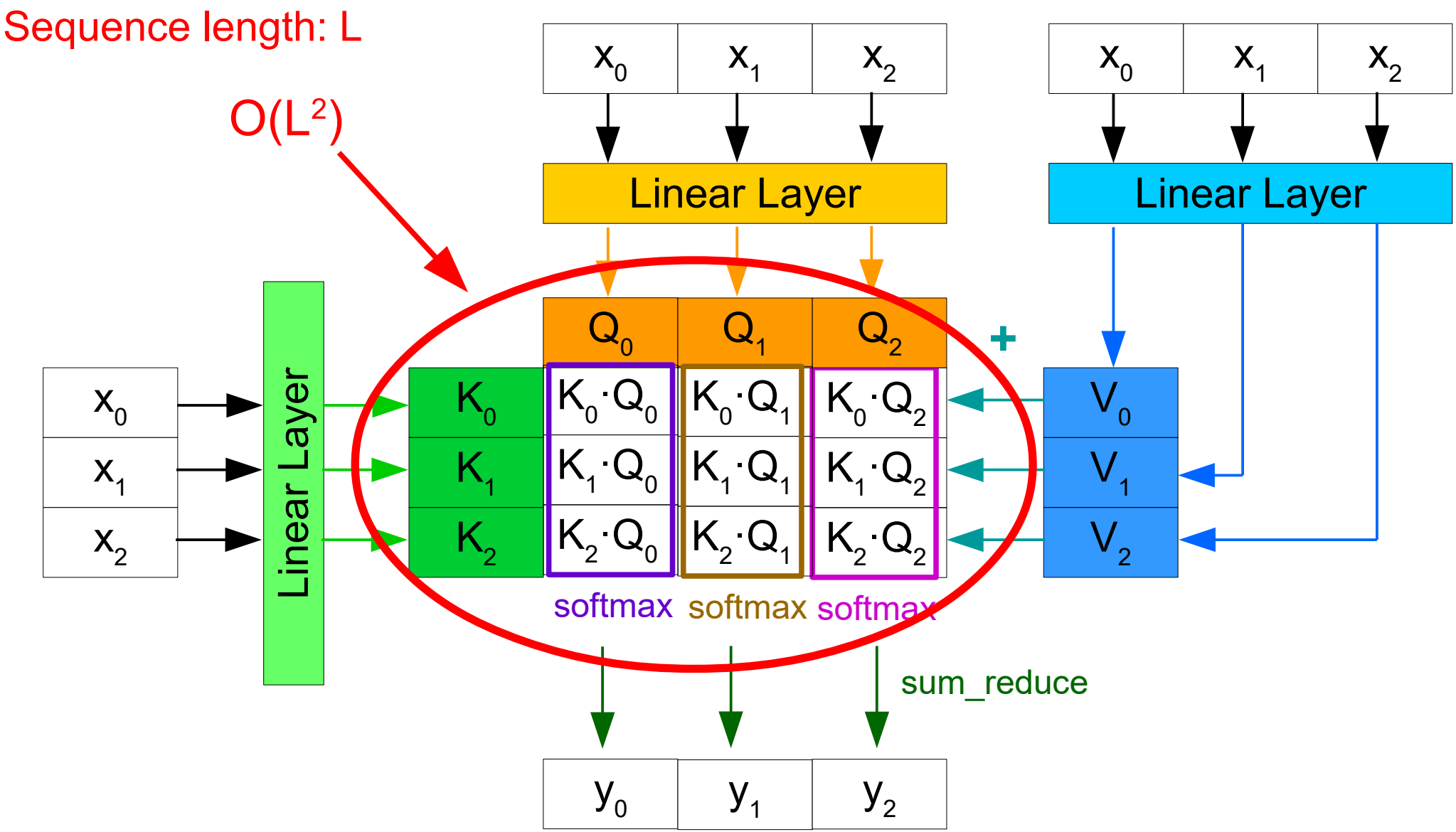


Image sources:
https://cdn-icons-png.flaticon.com/512/5116/5116423.png
(call dates: 21.01.23)
Tay, Yi et al. "Synthesizer: Rethinking Self-Attention in Transformer Models." International Conference on Machine Learning (2020).
Wang, Sinong et al. "Linformer: Self-Attention with Linear Complexity." ArXiv abs/2006.04768 (2020): n. pag.

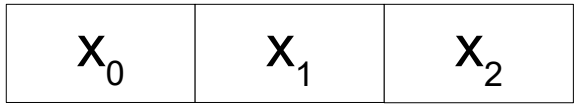
7. Self-attention: Time and Space Complexity



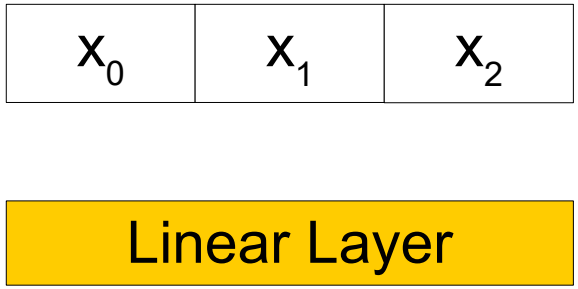
7. Self-attention: Time and Space Complexity



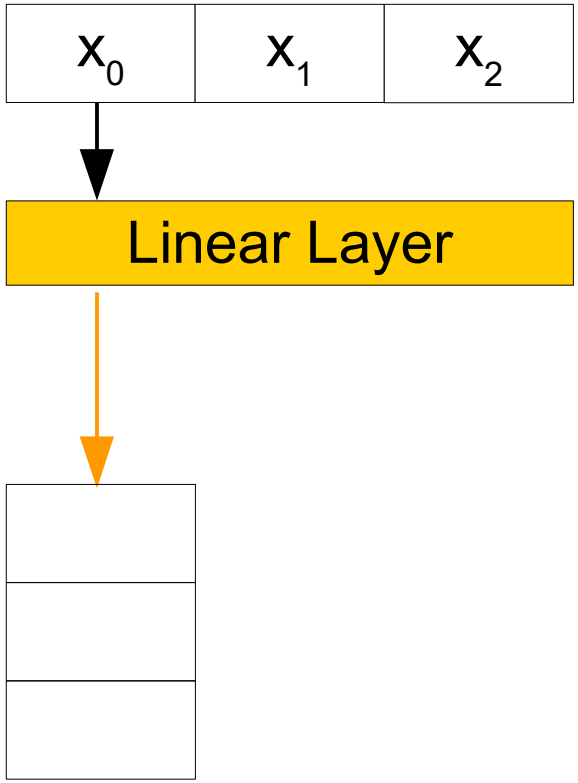
7.1 Synthesizer: *Dense* variant



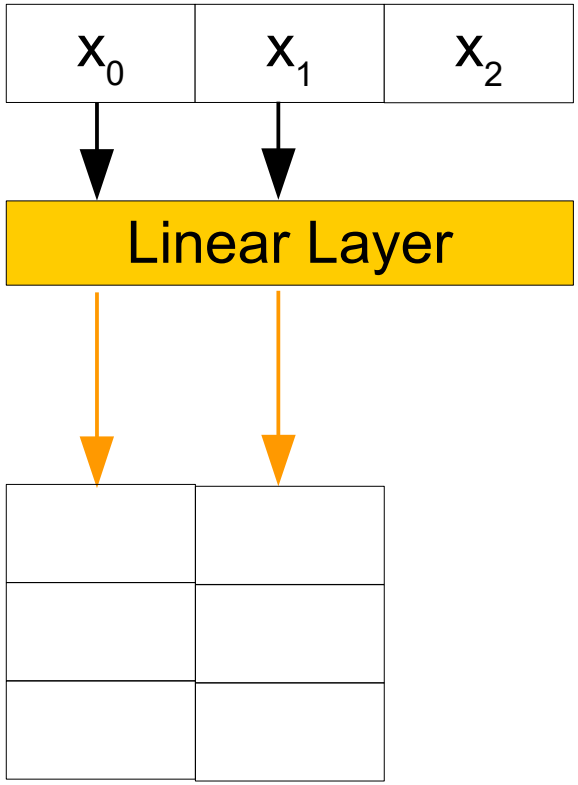
7.1 Synthesizer: *Dense* variant



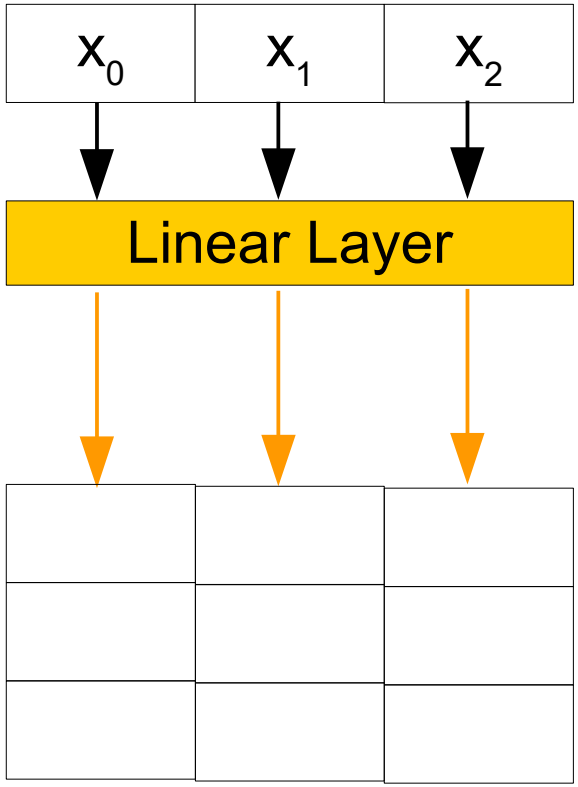
7.1 Synthesizer: *Dense* variant



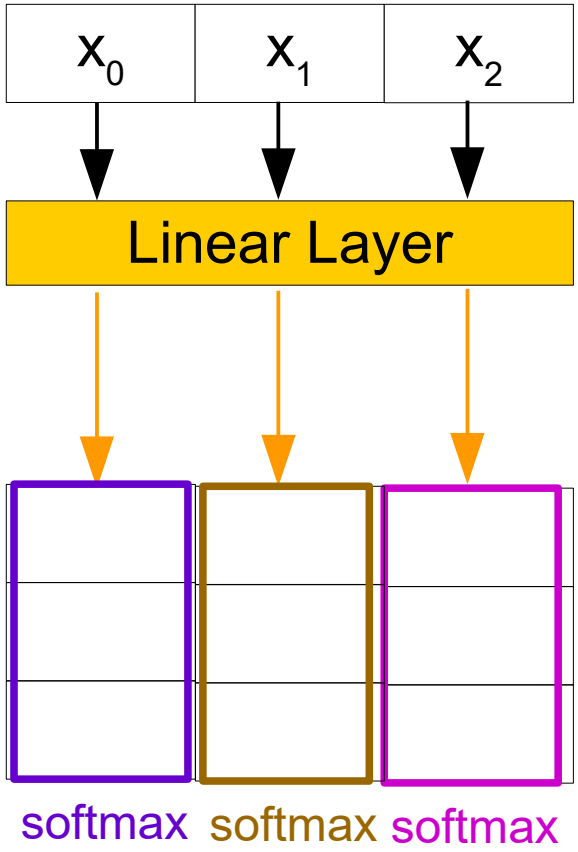
7.1 Synthesizer: *Dense* variant



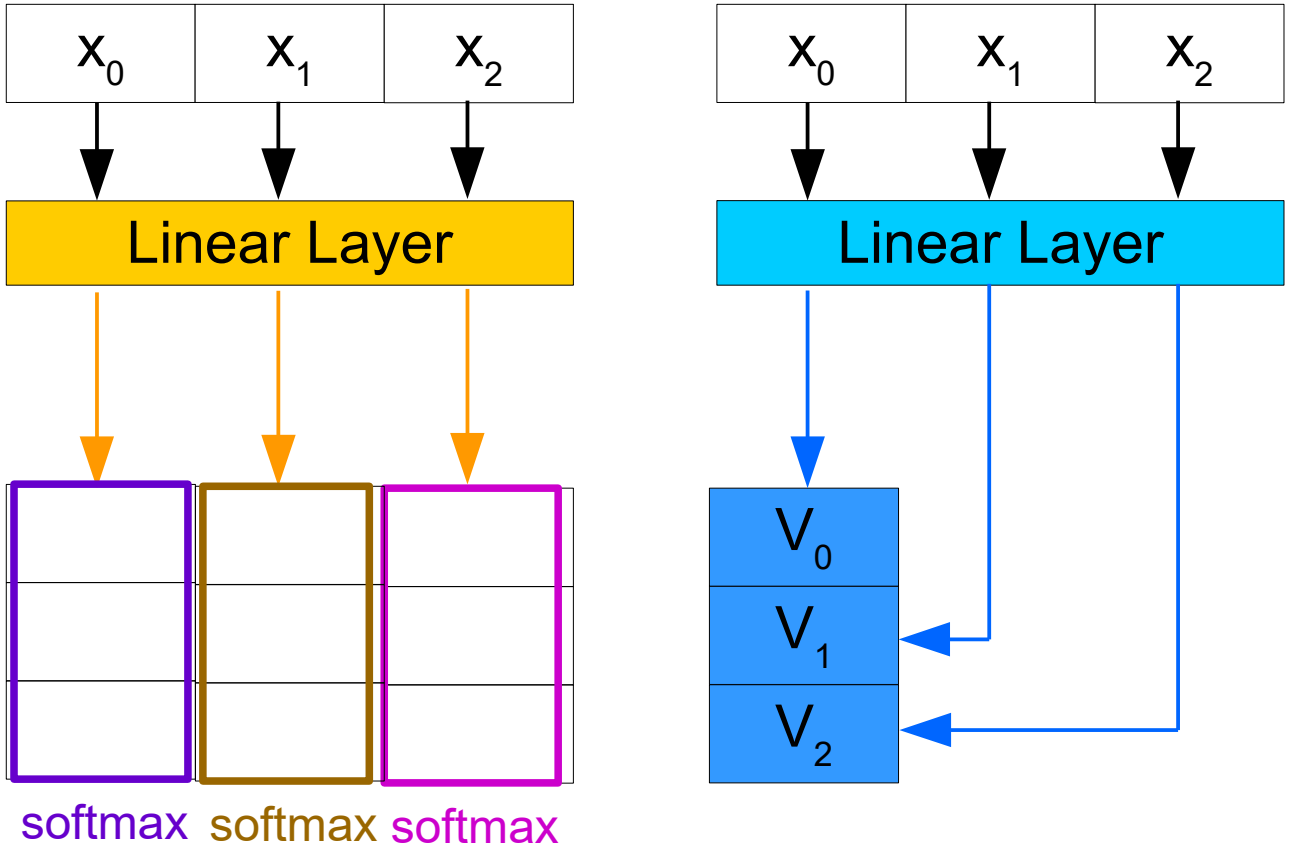
7.1 Synthesizer: *Dense* variant



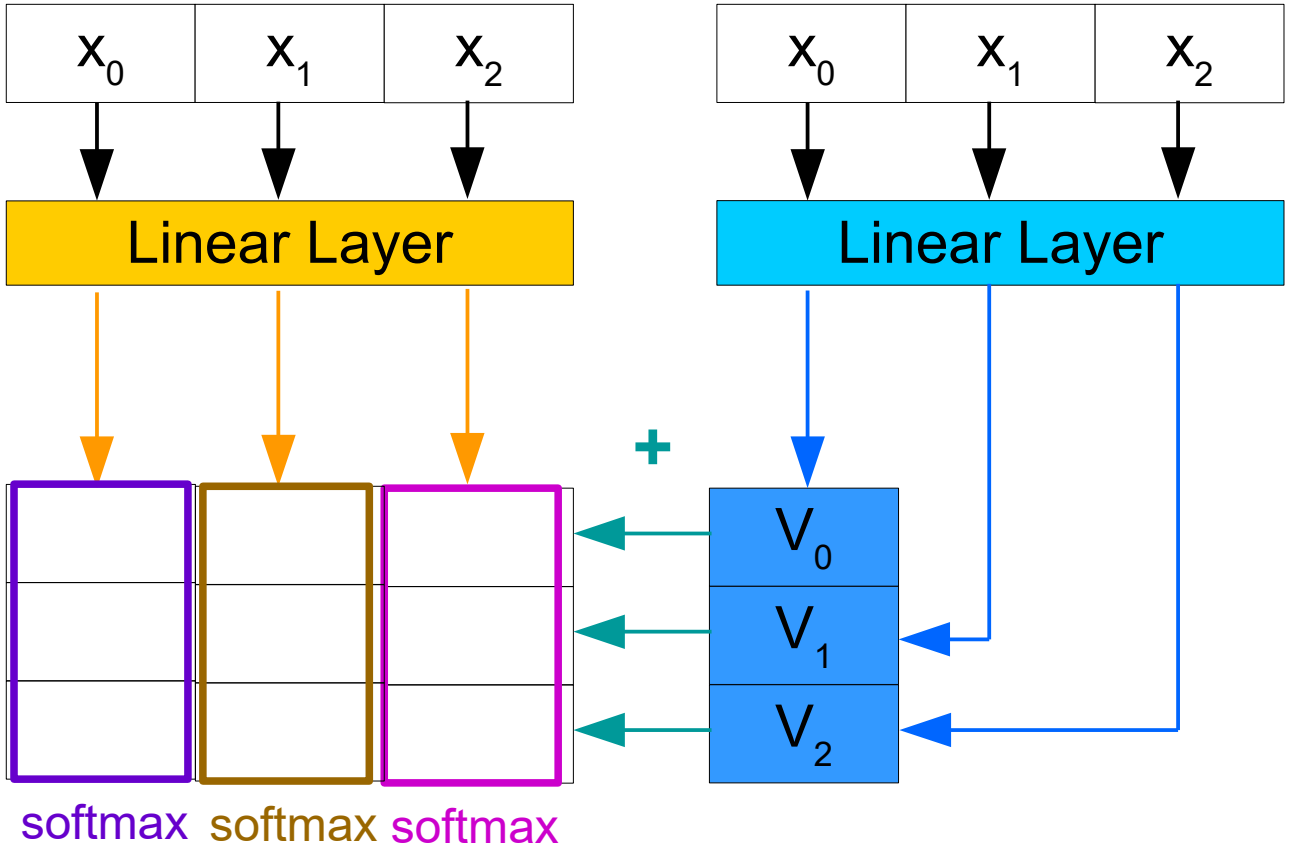
7.1 Synthesizer: *Dense* variant



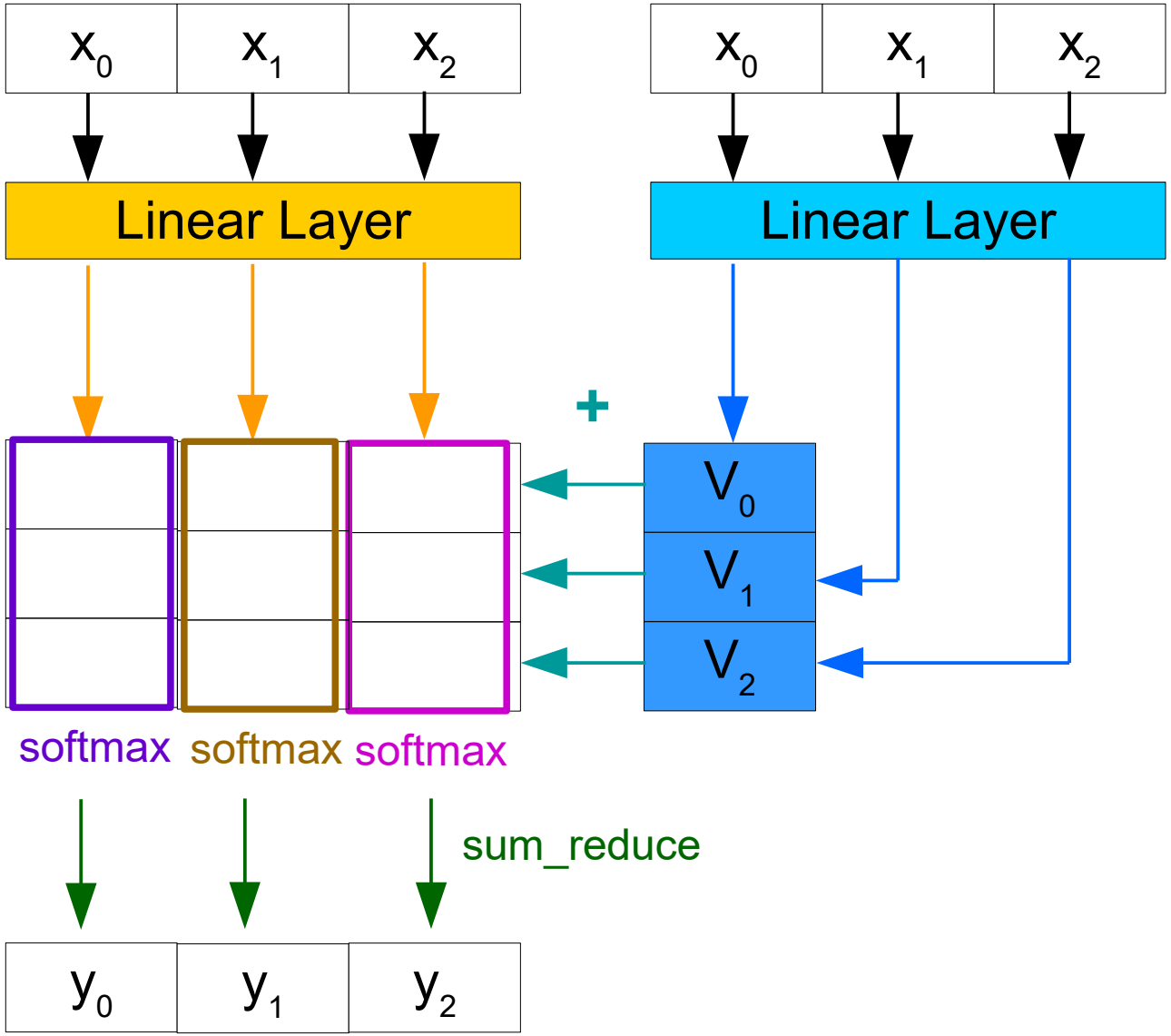
7.1 Synthesizer: *Dense* variant



7.1 Synthesizer: *Dense* variant



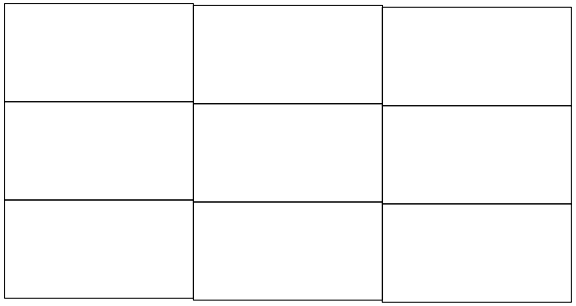
7.1 Synthesizer: *Dense* variant



7.1 Synthesizer: *Random* variant

x_0	x_1	x_2

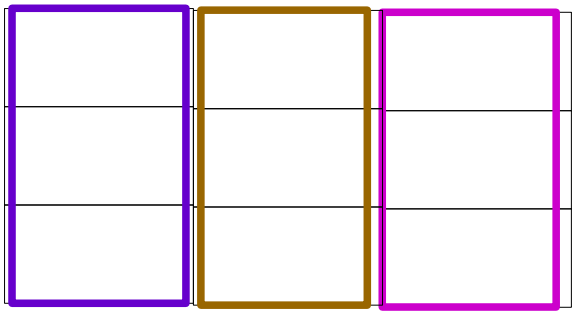
Fixed



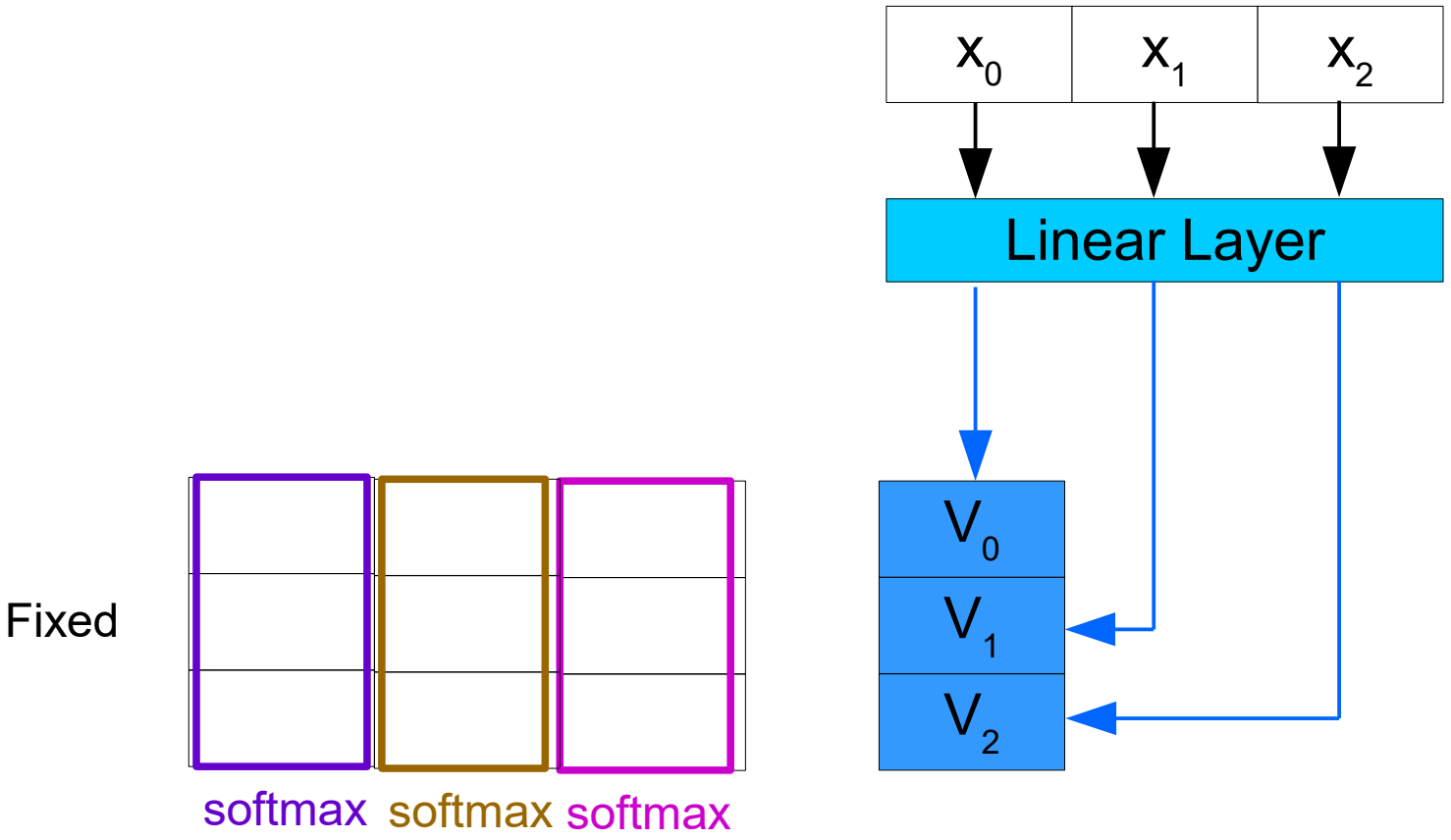
7.1 Synthesizer: *Random* variant

x_0	x_1	x_2

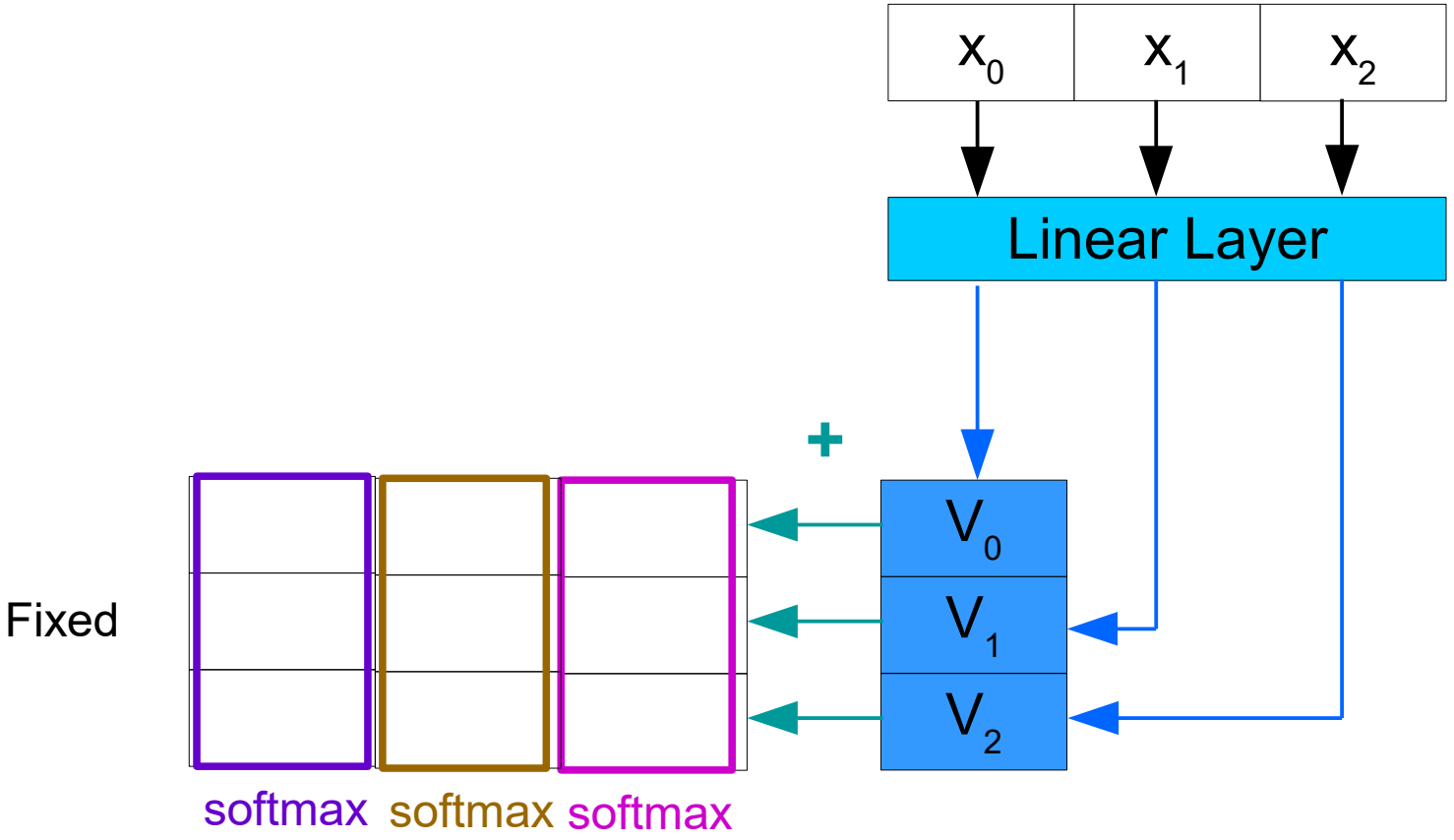
Fixed



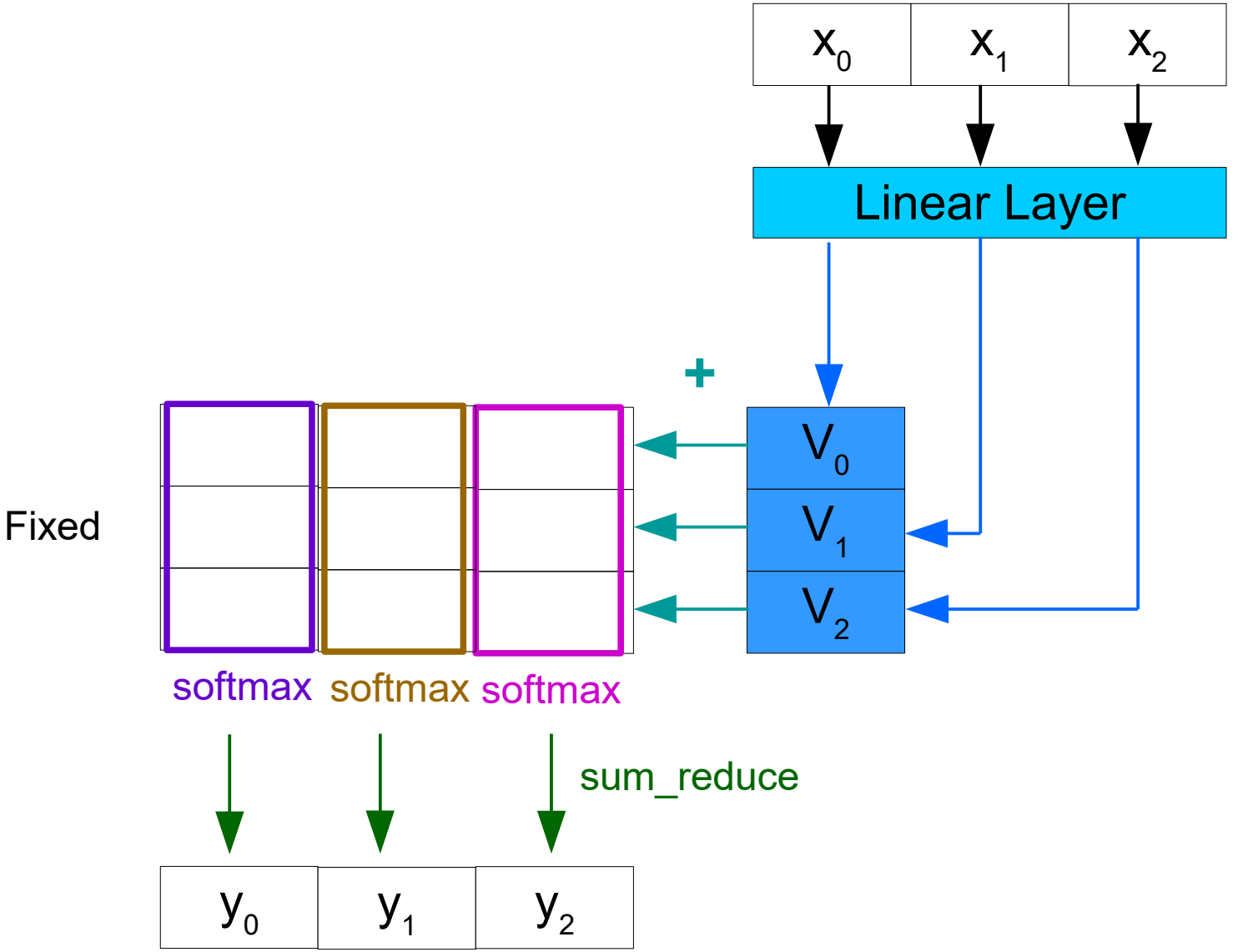
7.1 Synthesizer: *Random* variant



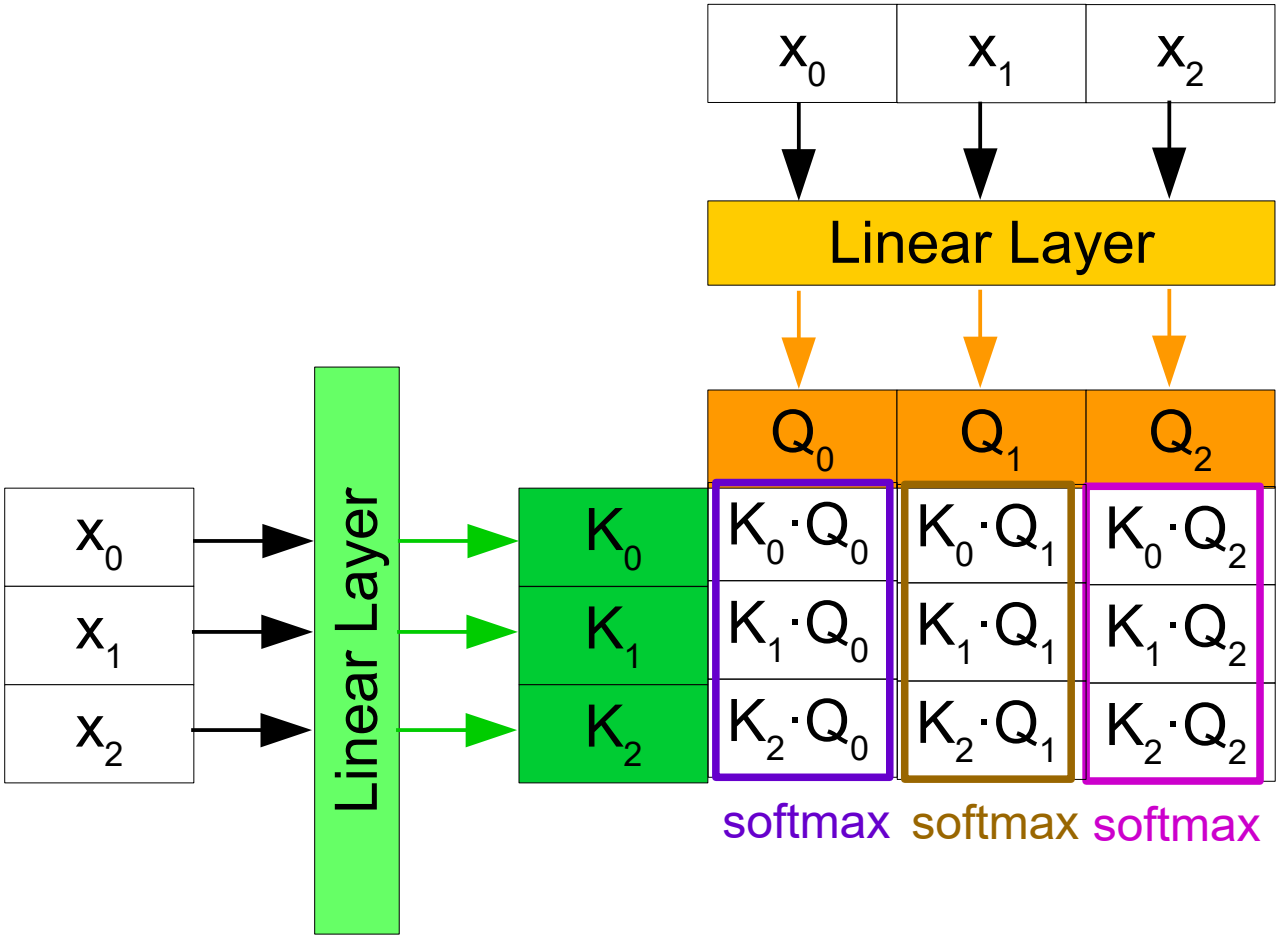
7.1 Synthesizer: *Random* variant



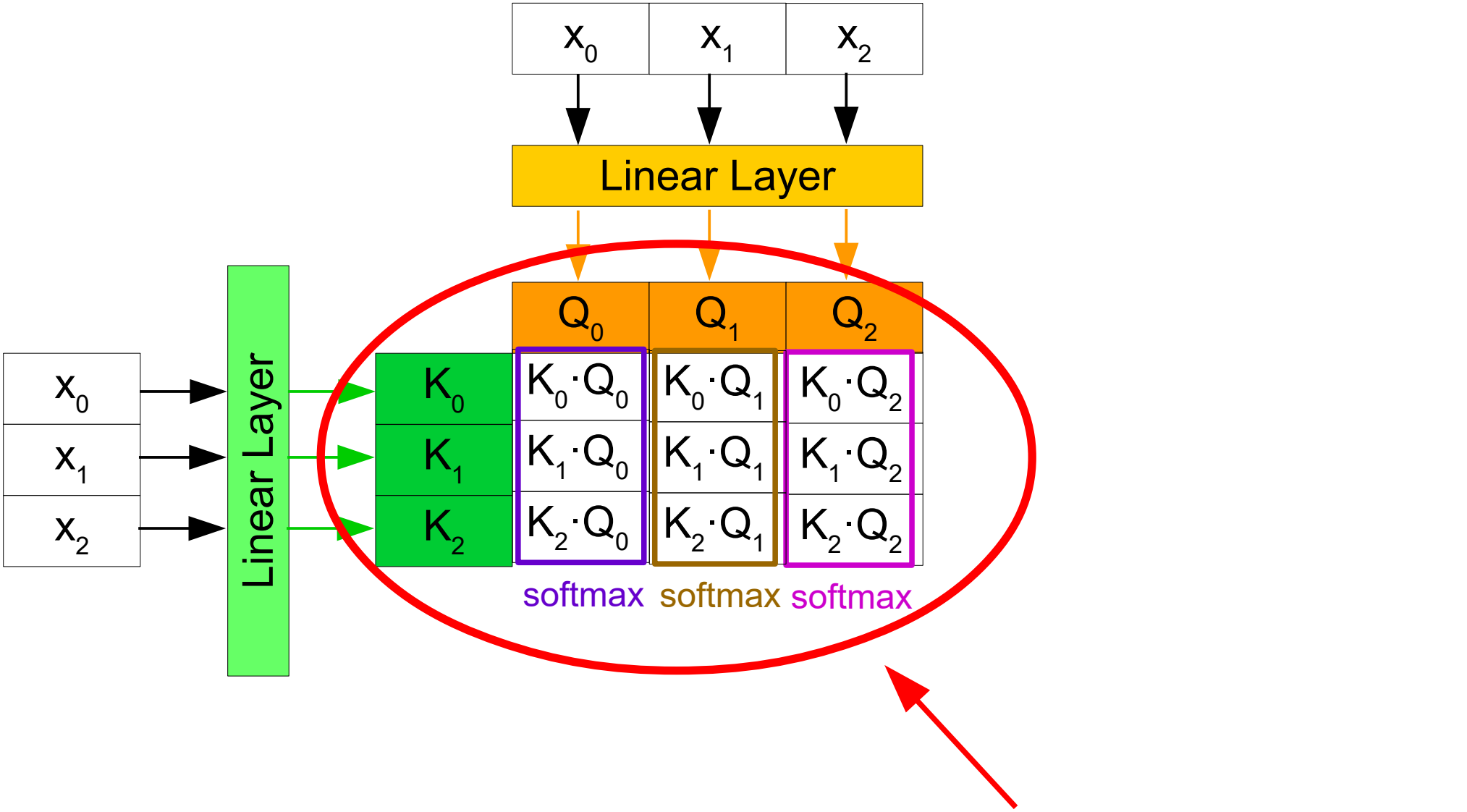
7.1 Synthesizer: *Random* variant



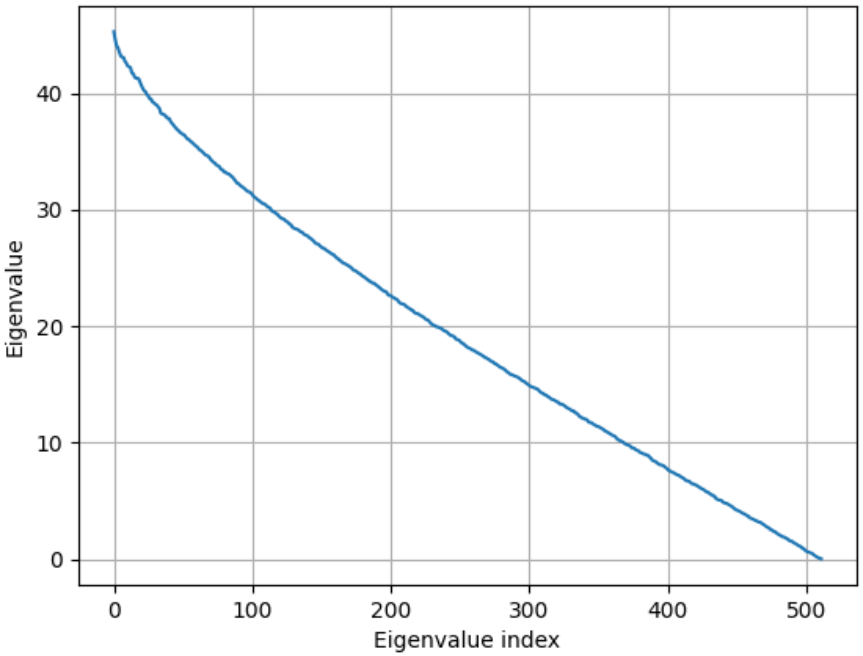
7.2 Linformer: Self-Attention with Linear Complexity



7.2 Linformer: Self-Attention with Linear Complexity



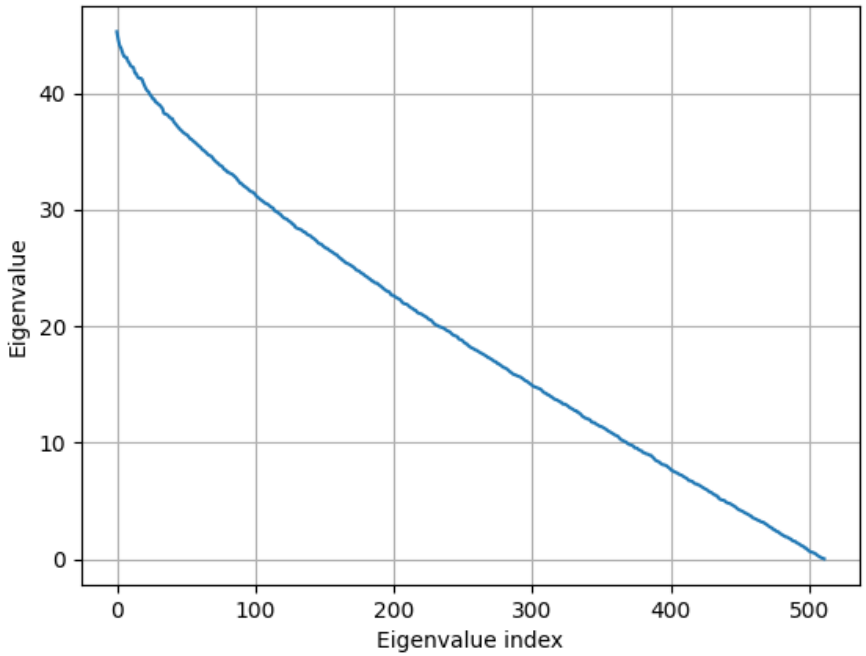
7.2 Linformer: Self-Attention with Linear Complexity



Eigenvalue spectrum of a 256x256 matrix (full rank).

7.2 Linformer: Self-Attention with Linear Complexity

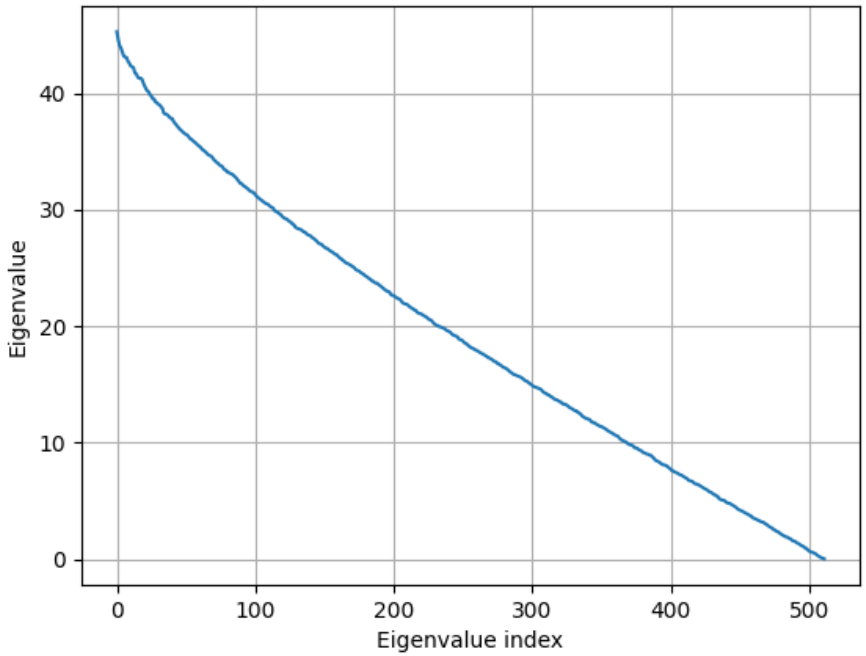
“Order a set of uniform distributed numbers”
→ All dimensions have the same amount of information

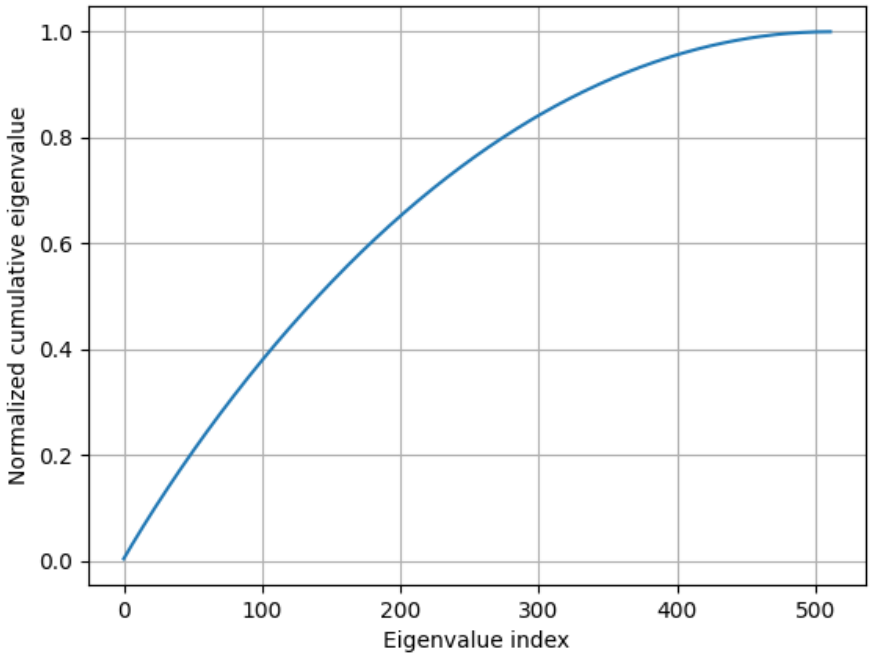


Eigenvalue spectrum of a 256x256 matrix (full rank).

7.2 Linformer: Self-Attention with Linear Complexity

“Order a set of uniform distributed numbers”
→ All dimensions have the same amount of information

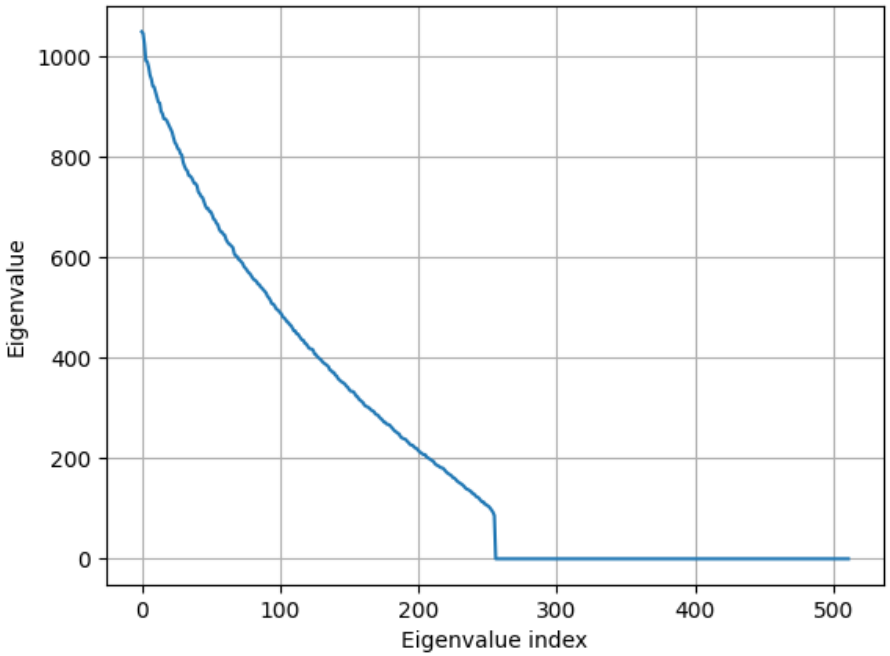




Eigenvalue spectrum of a 256x256 matrix (full rank).

7.2 Linformer: Self-Attention with Linear Complexity

A = 256 x 128
B = 128 x 256
M = A · B

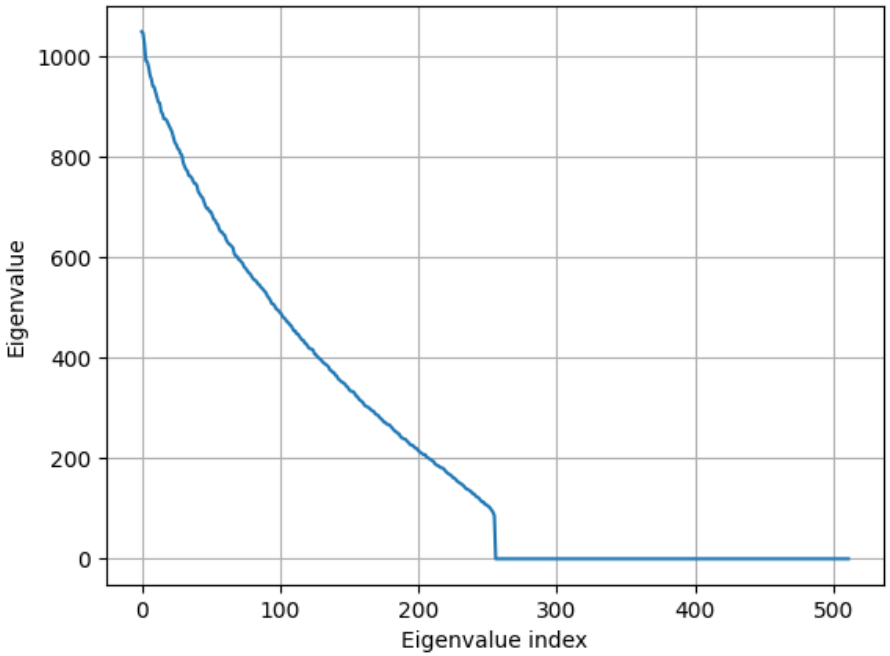


Eigenvalue spectrum of a 256x256 matrix (low rank).

7.2 Linformer: Self-Attention with Linear Complexity

A = 256 x 128
B = 128 x 256
M = A · B

Information is concentrated along few dimensions

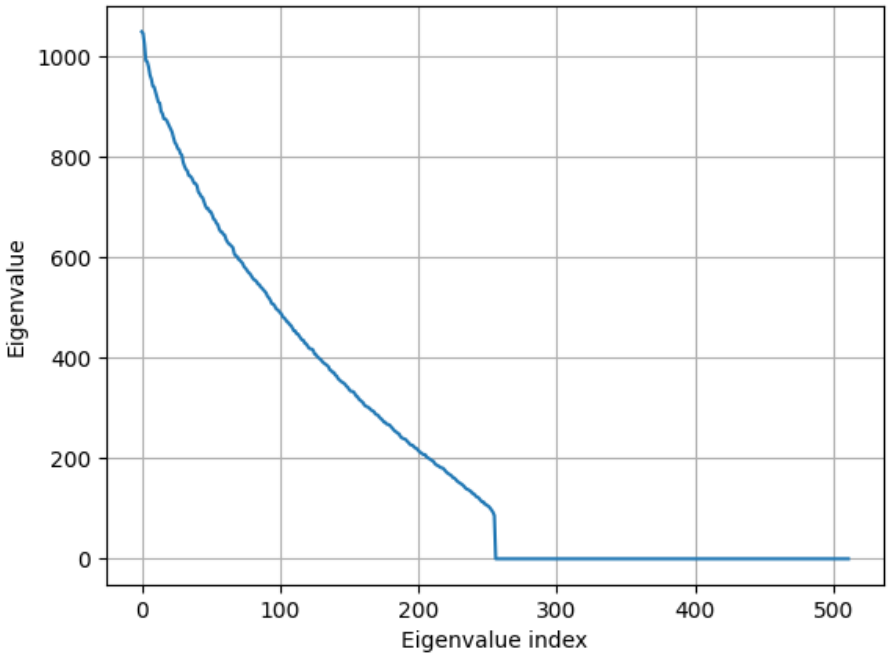


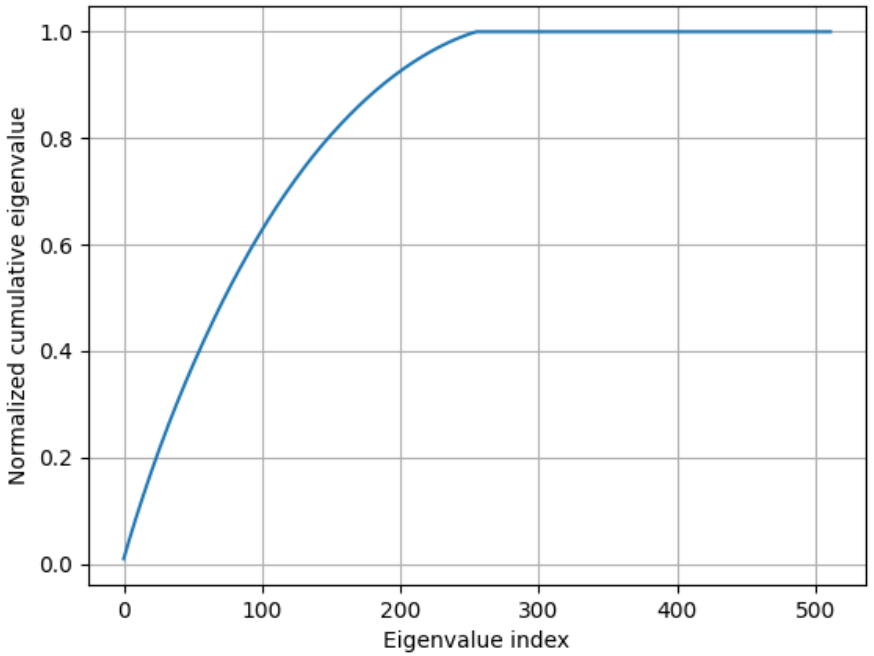
Eigenvalue spectrum of a 256x256 matrix (low rank).

7.2 Linformer: Self-Attention with Linear Complexity

A = 256 x 128
B = 128 x 256
M = A · B

Information is concentrated along few dimensions





Eigenvalue spectrum of a 256x256 matrix (low rank).

7.2 Linformer: Self-Attention is Low Rank

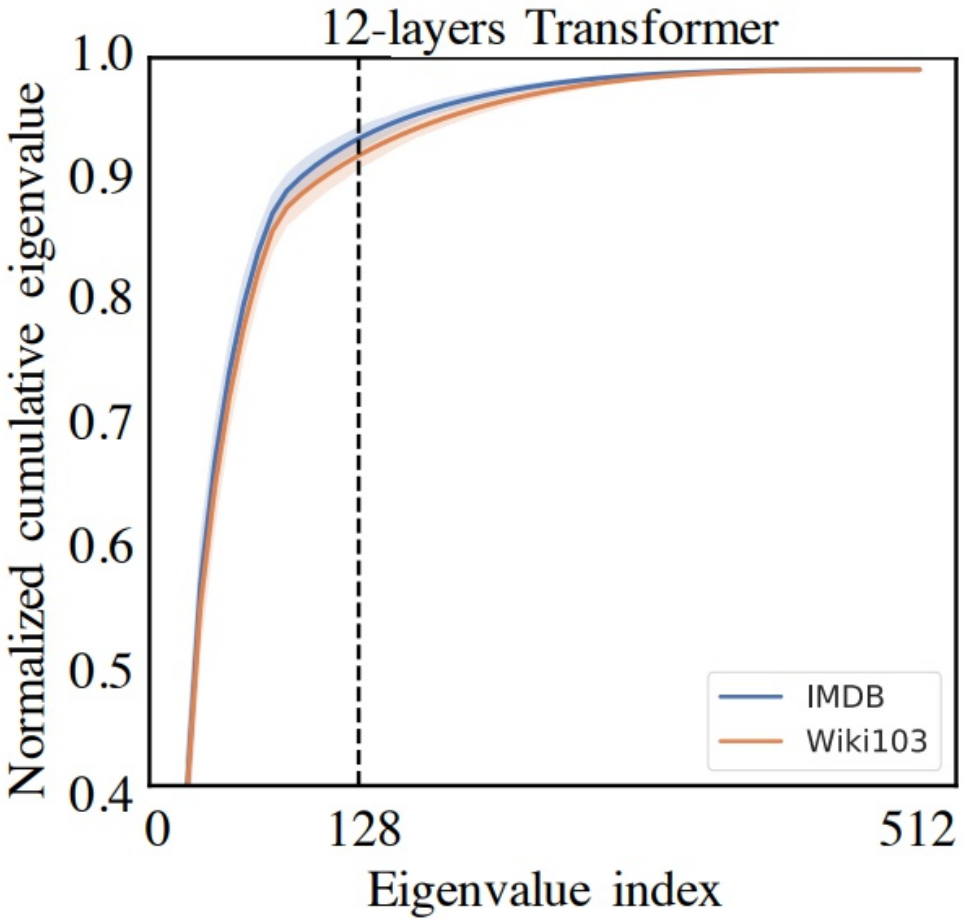
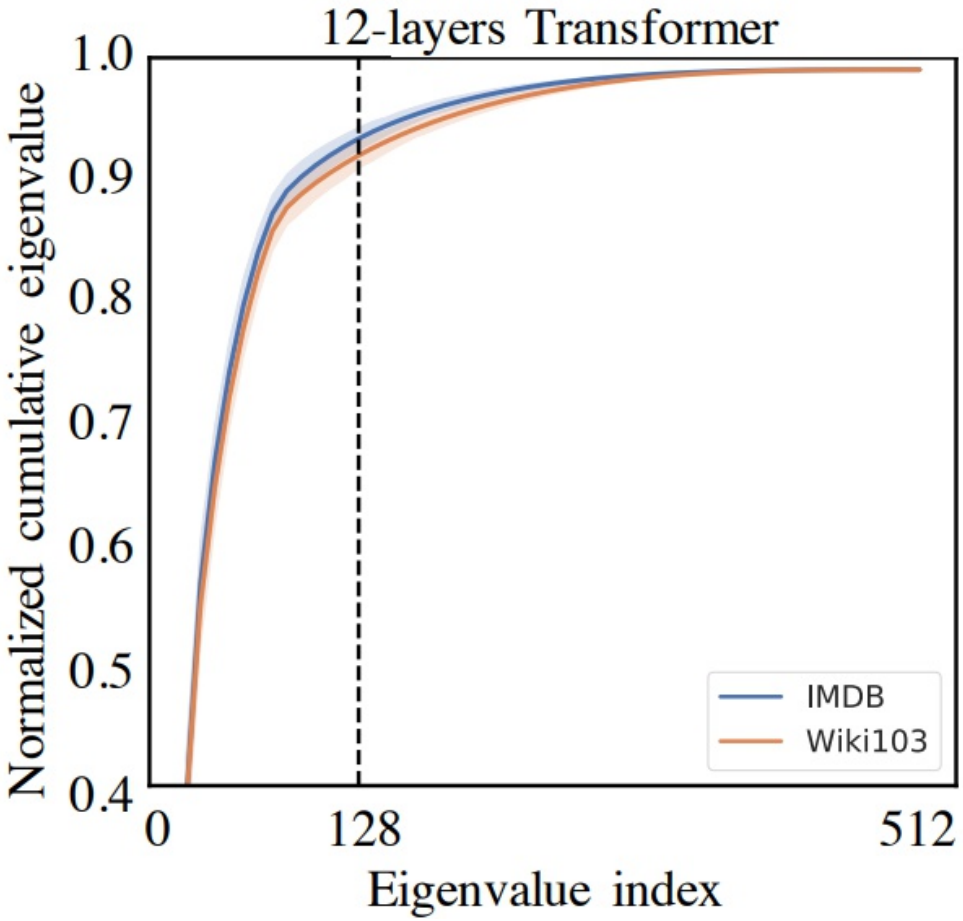
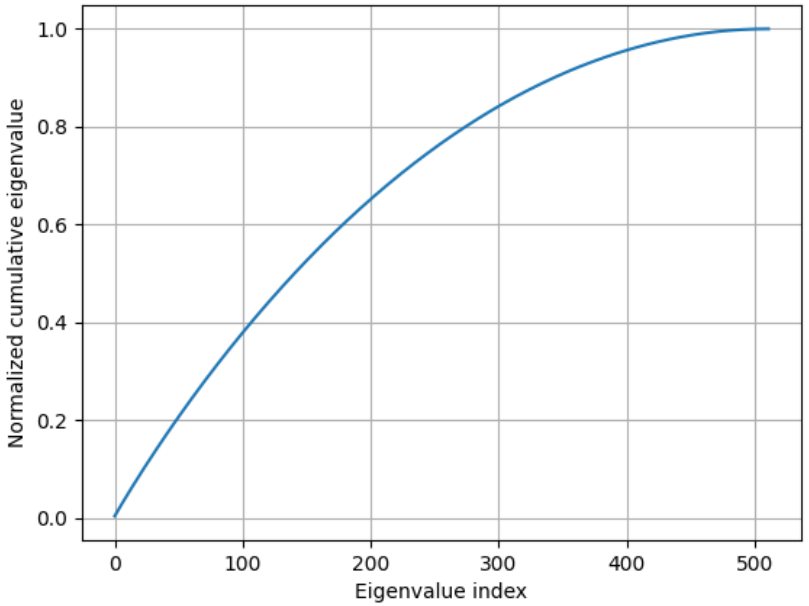


Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

7.2 Linformer: Self-Attention is Low Rank



Full rank



Low rank

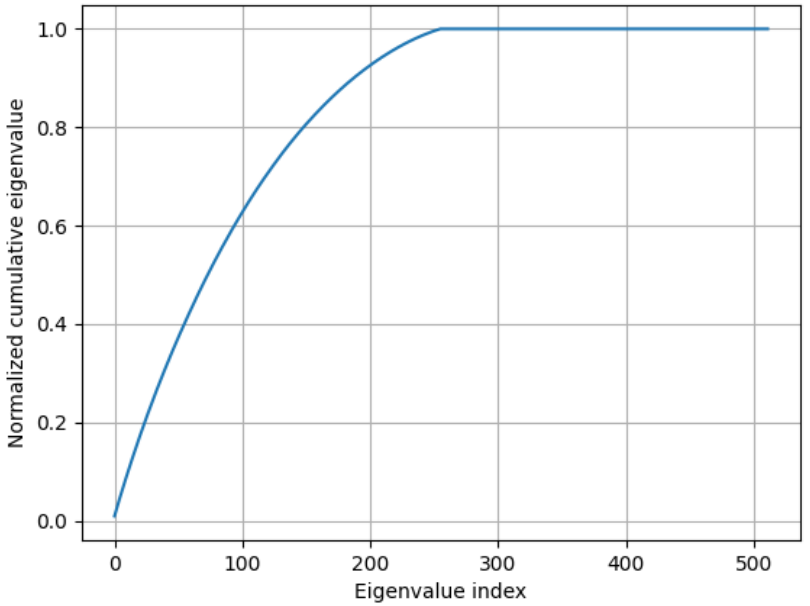
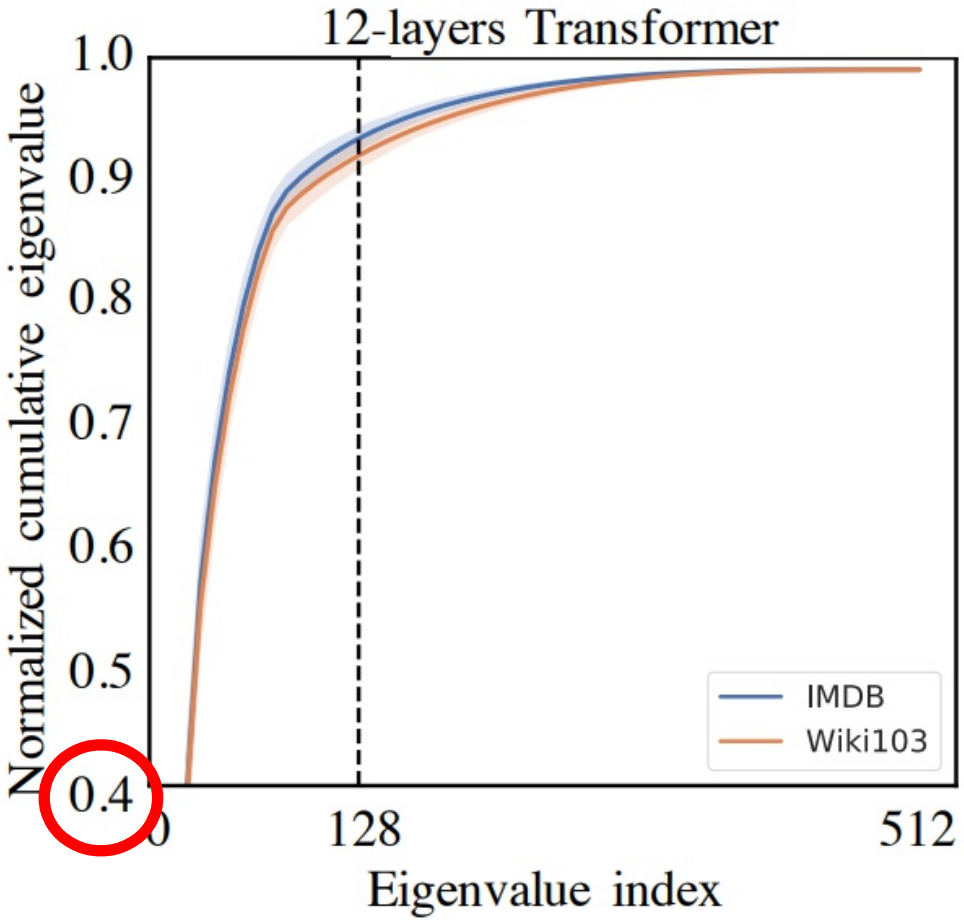
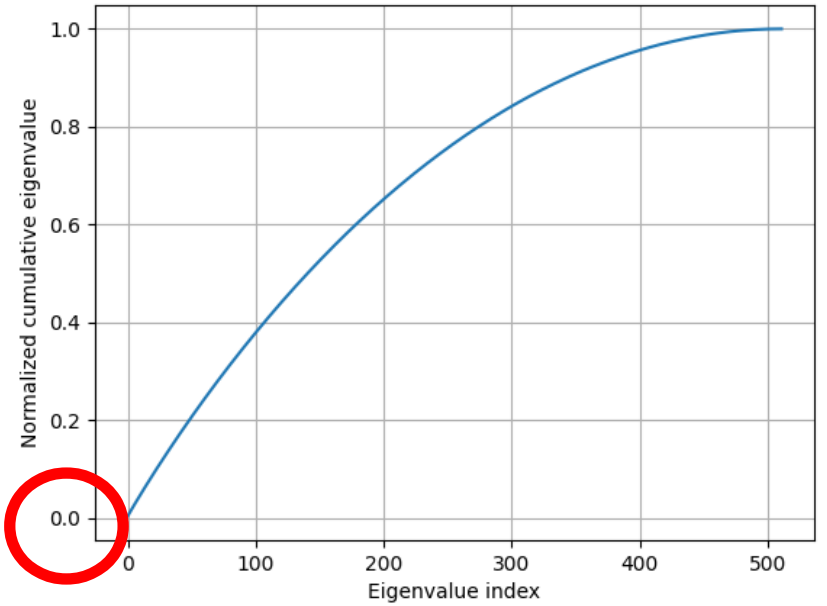


Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

7.2 Linformer: Self-Attention is Low Rank



Full rank



Low rank

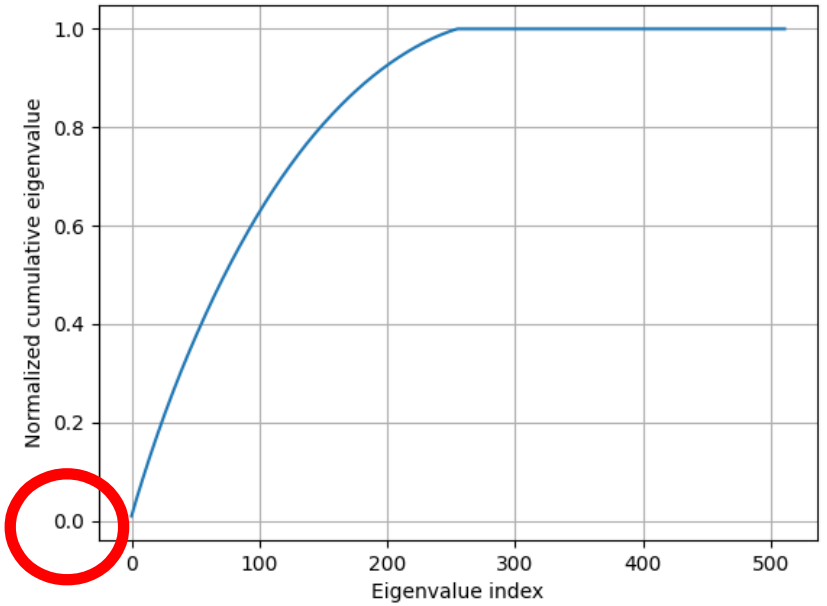


Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

7.2 Linformer: Self-Attention with Linear Complexity

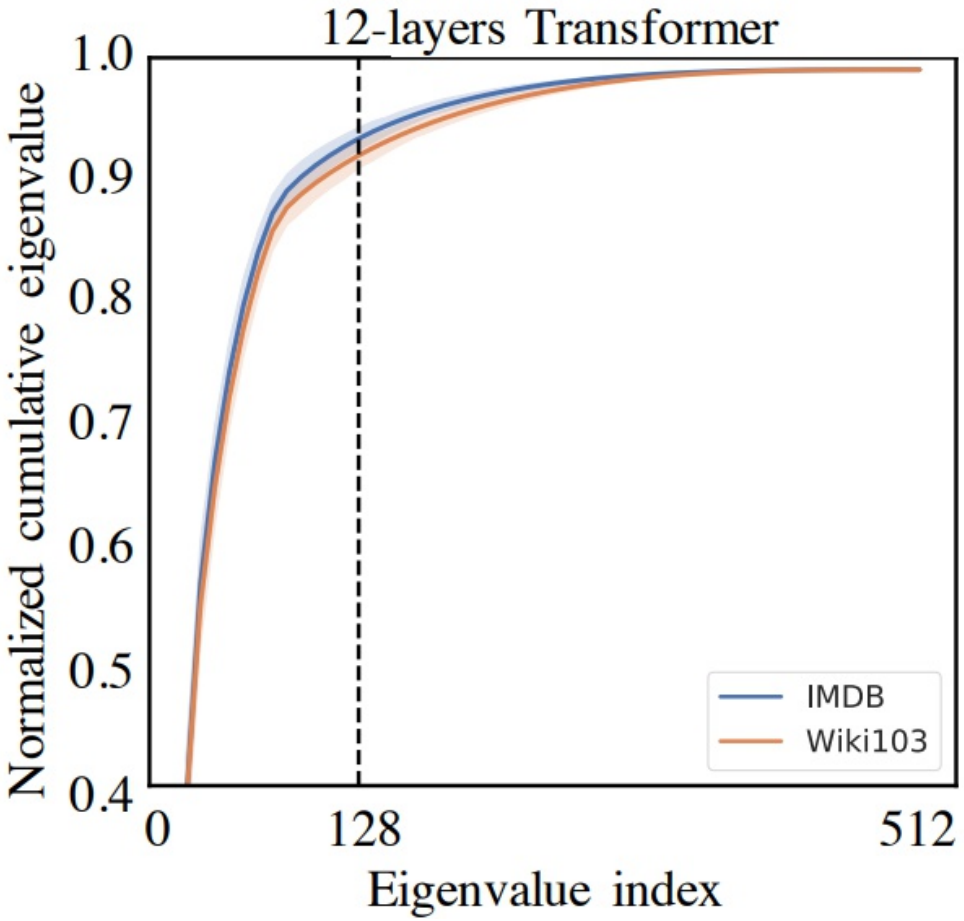
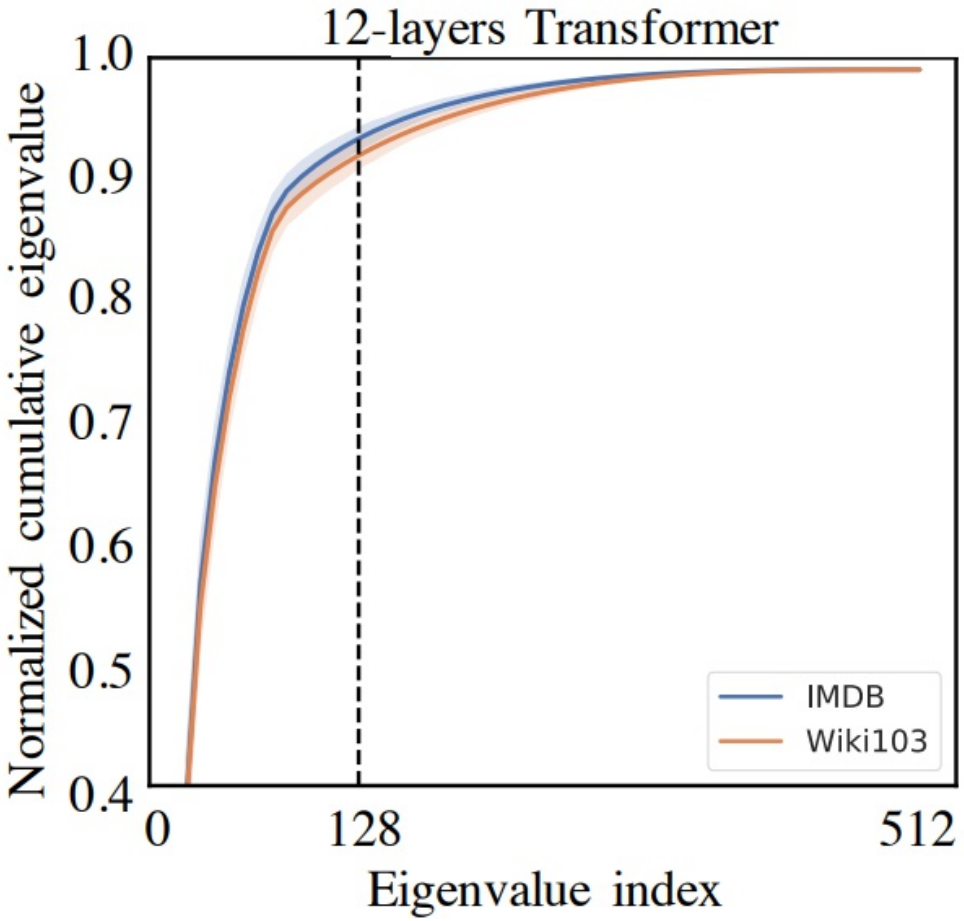


Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

7.2 Linformer: Self-Attention with Linear Complexity



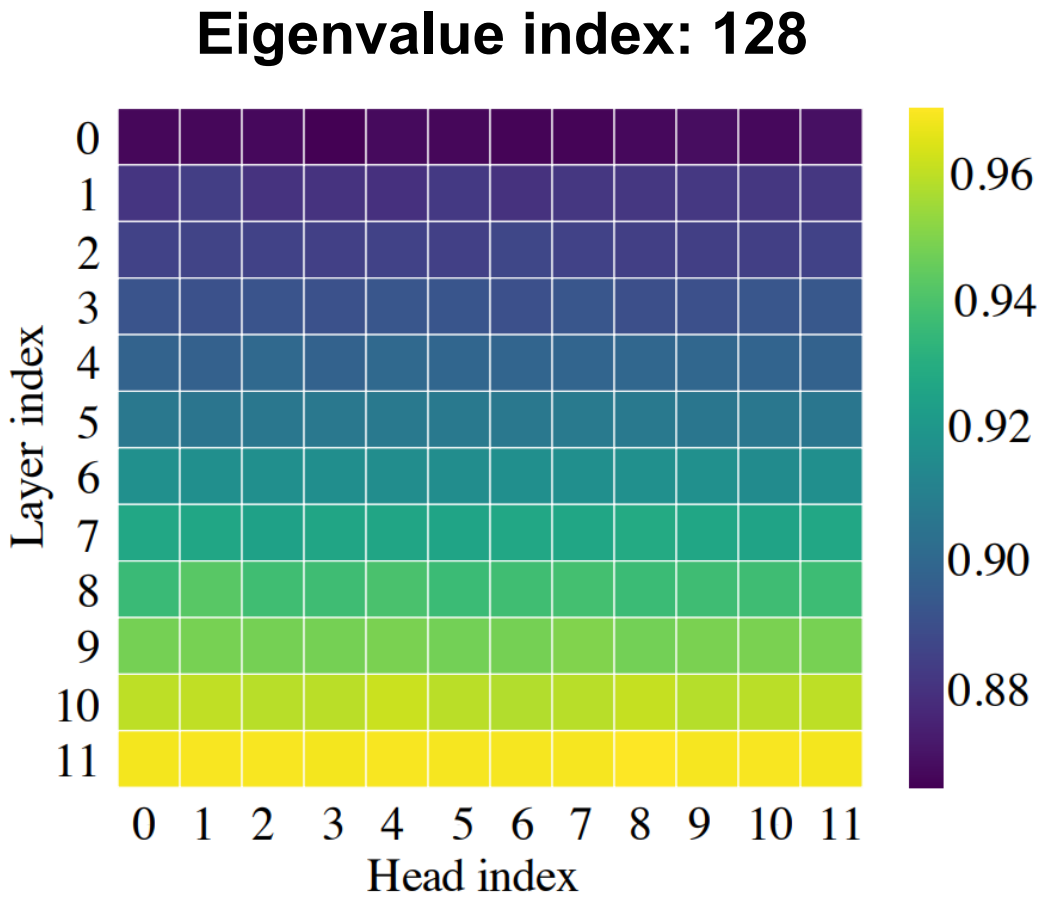
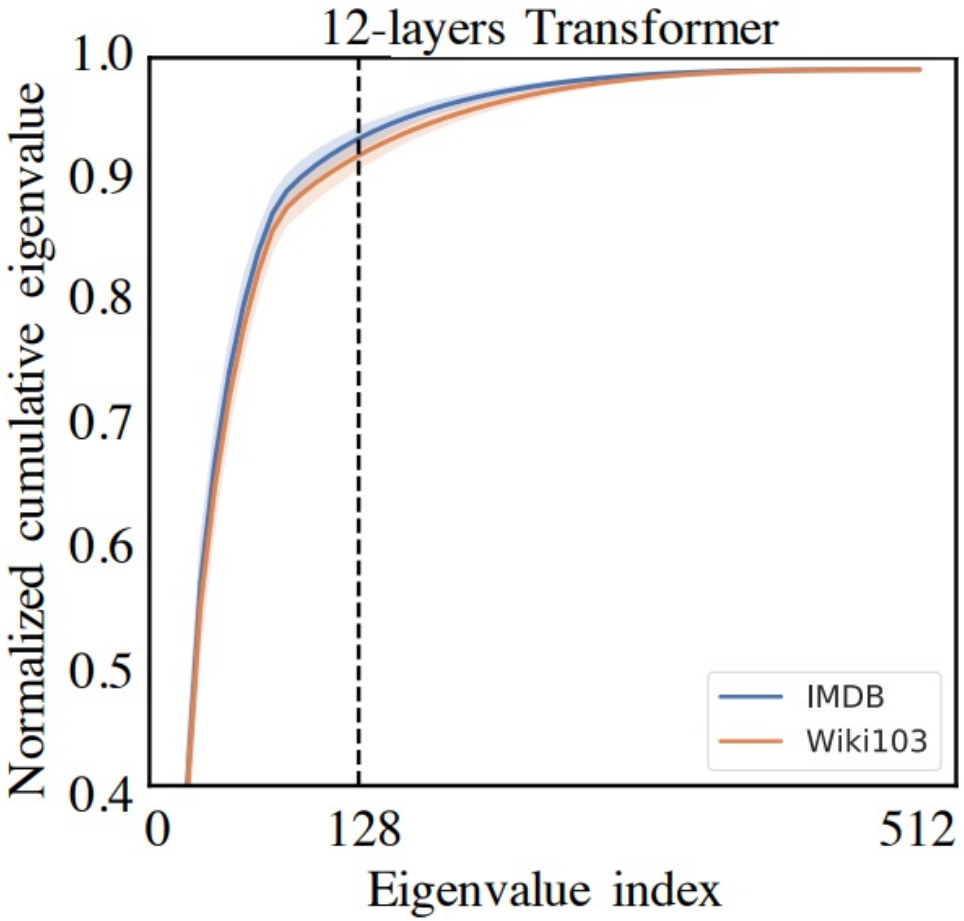
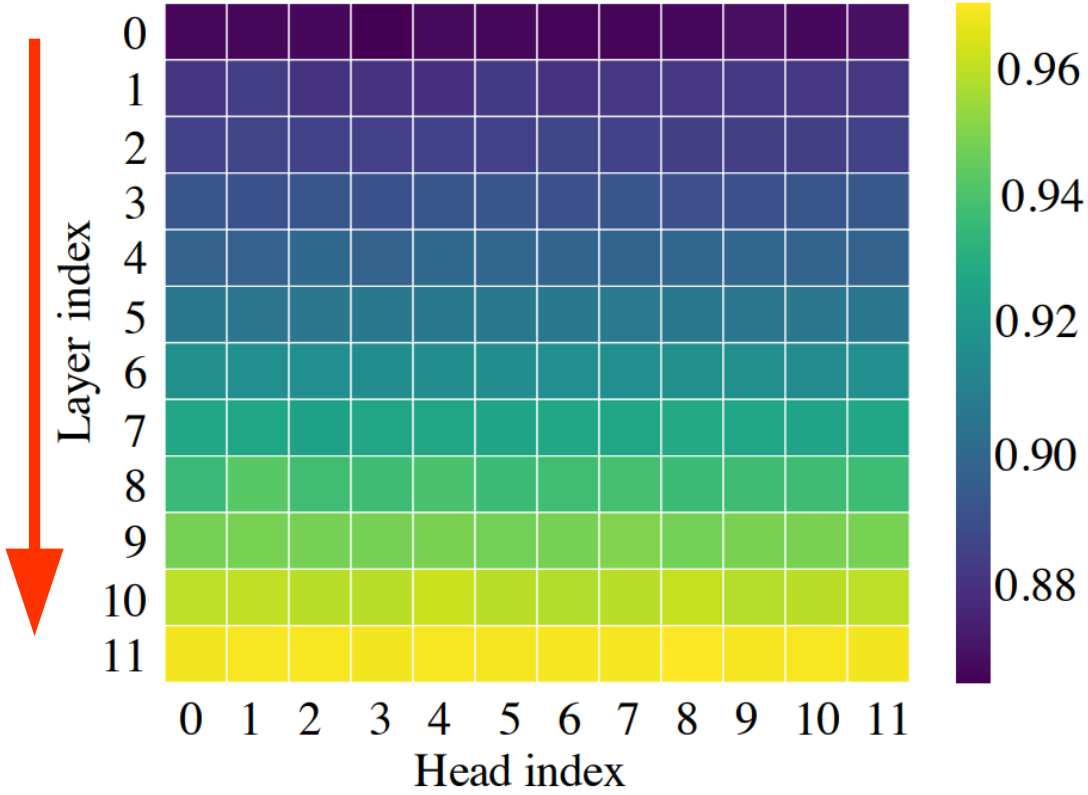


Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

7.2 Linformer: Self-Attention with Linear Complexity



Eigenvalue index: 128

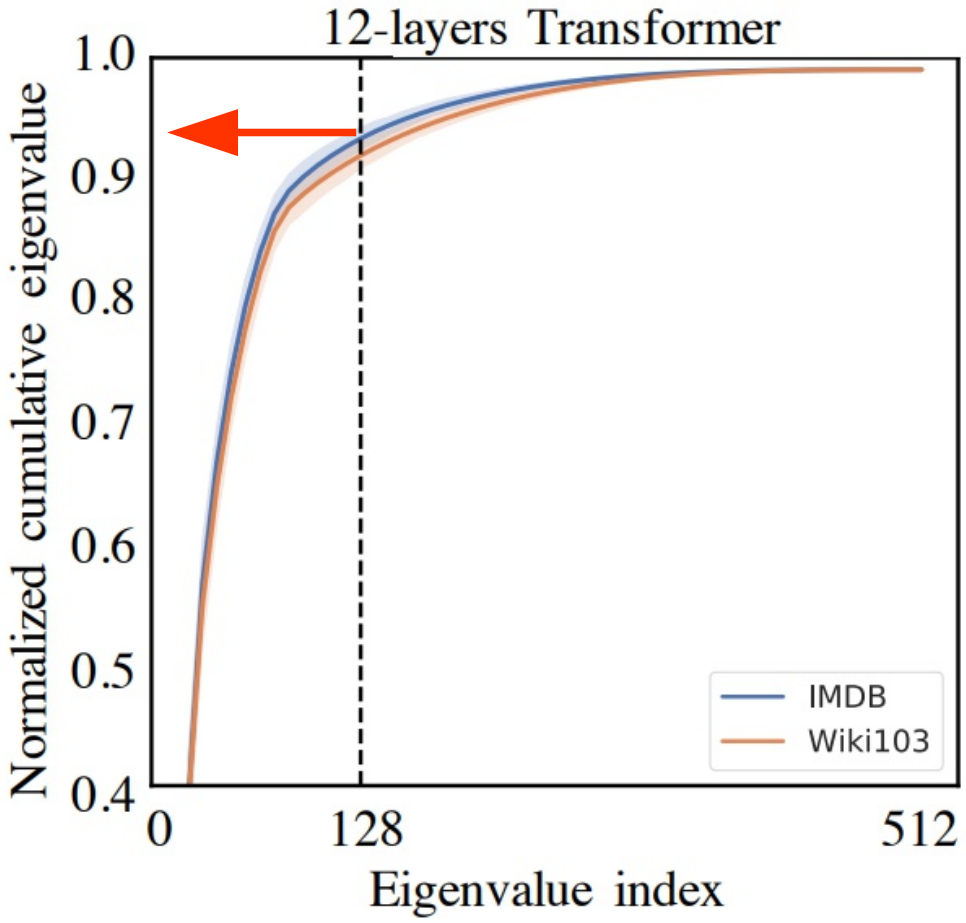


With increasing deep:
Put more and more information into fewer and fewer dimensions

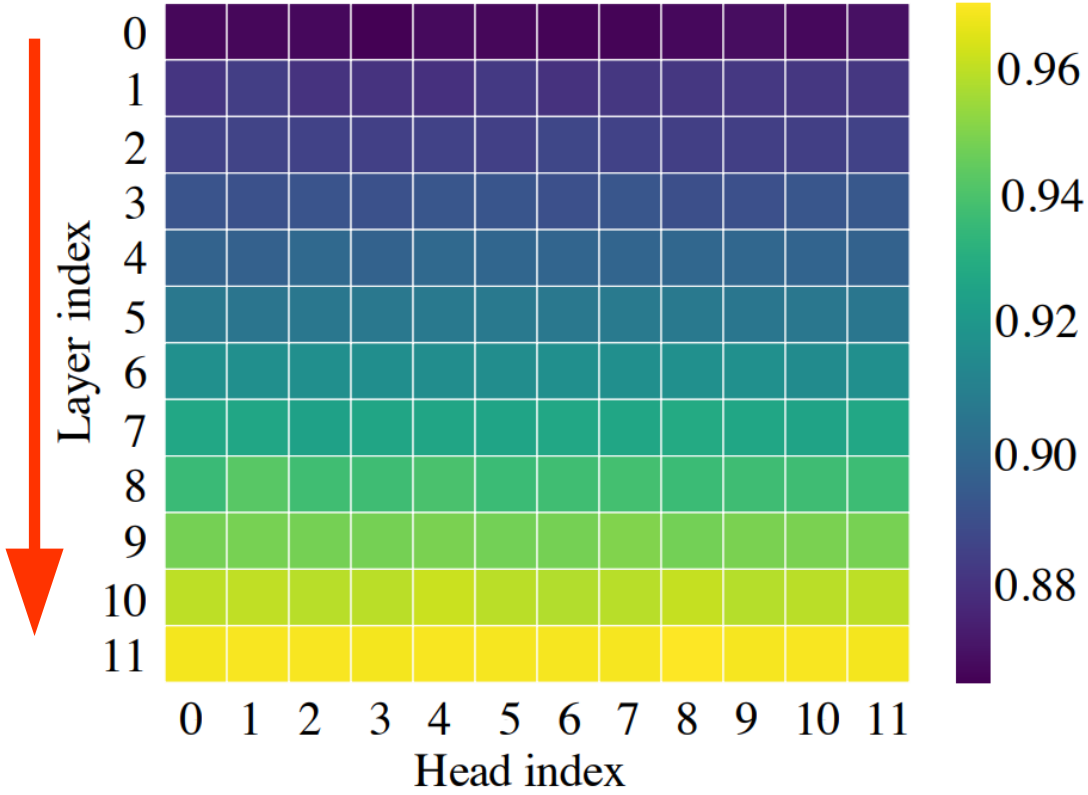
Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

7.2 Linformer: Self-Attention with Linear Complexity

More and more skewed



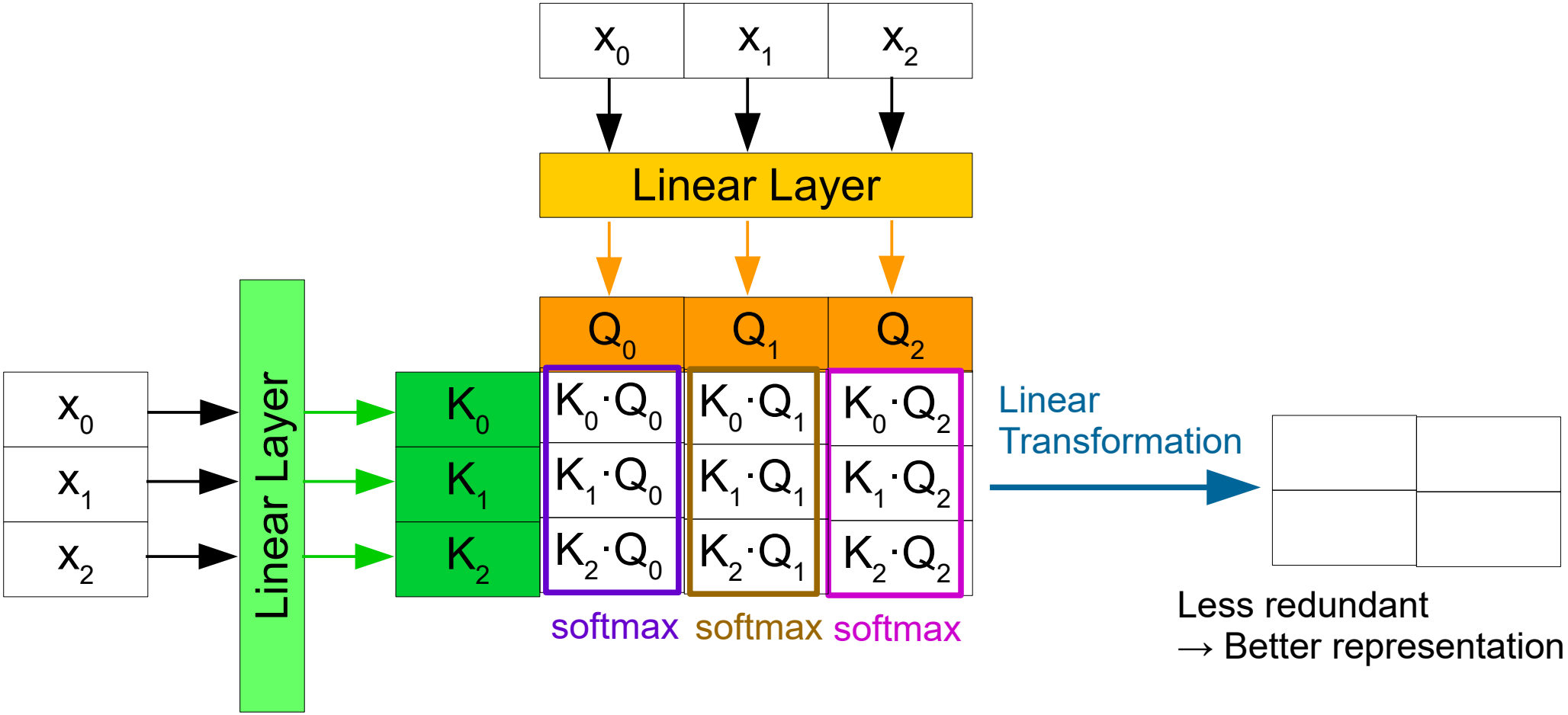
Eigenvalue index: 128



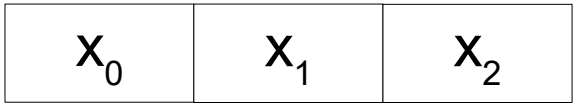
With increasing deep:
Put more and more information into fewer and fewer dimensions

Image sources:
Wang, S., Li, B.Z., Khabsa, M., Fang, H., & Ma, H. (2020). Linformer: Self-Attention with Linear Complexity. ArXiv, abs/2006.04768.

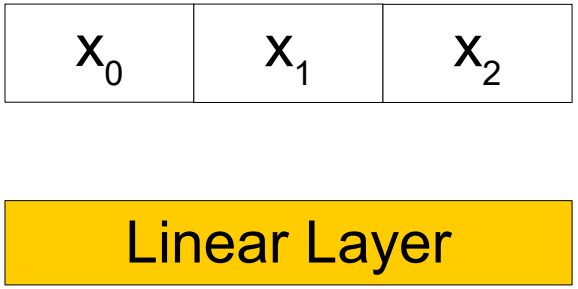
7.2 Linformer: Self-Attention with Linear Complexity



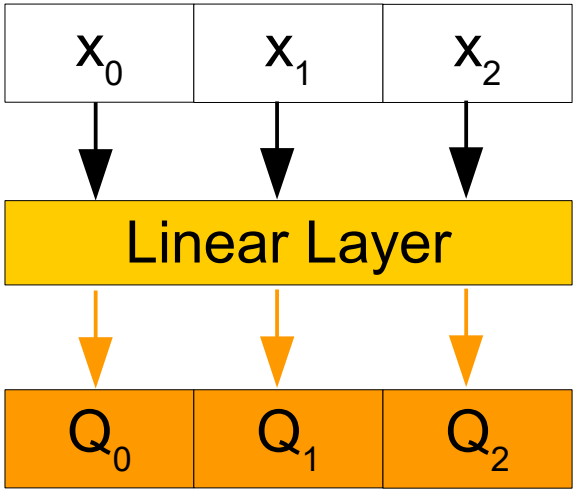
7.2 Linformer: Self-Attention with Linear Complexity



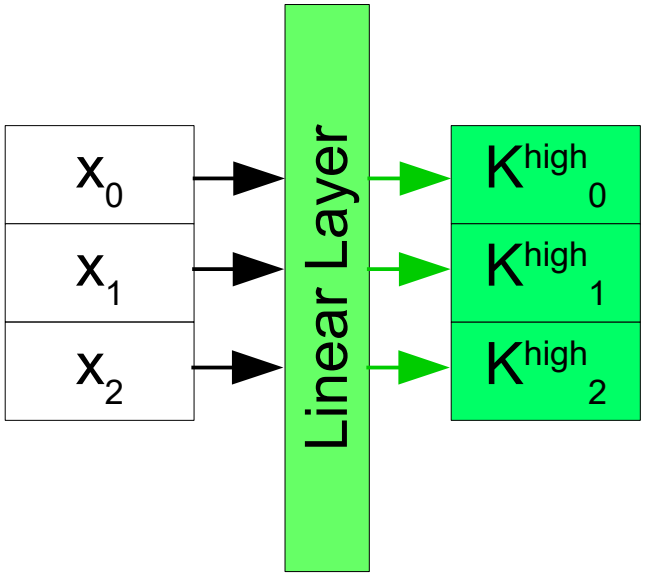
7.2 Linformer: Self-Attention with Linear Complexity

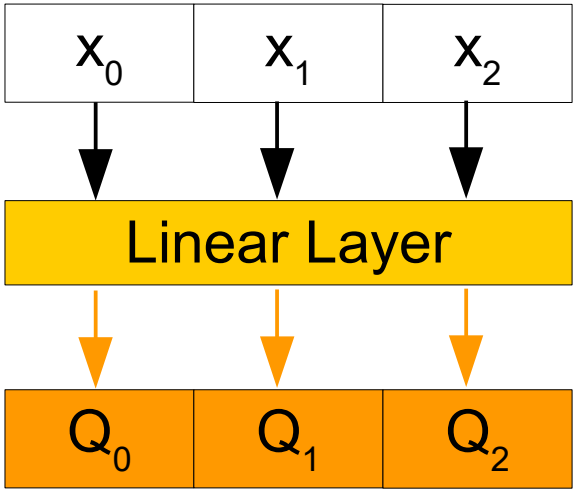


7.2 Linformer: Self-Attention with Linear Complexity

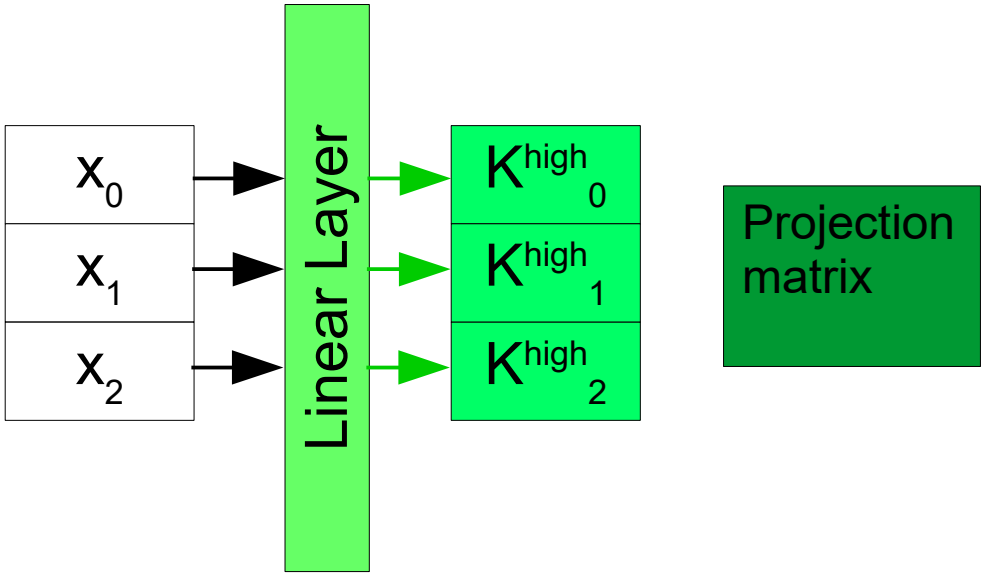


7.2 Linformer: Self-Attention with Linear Complexity

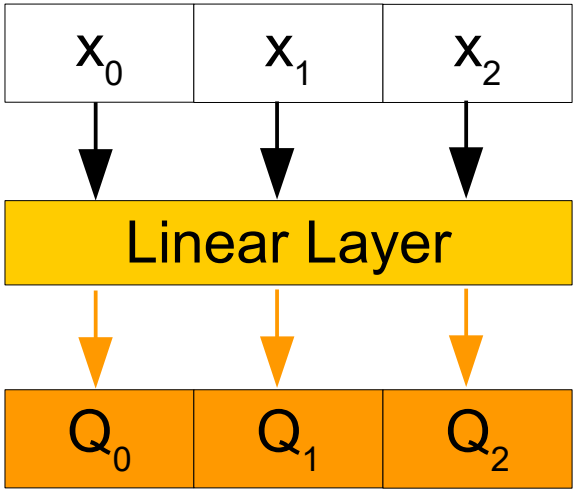




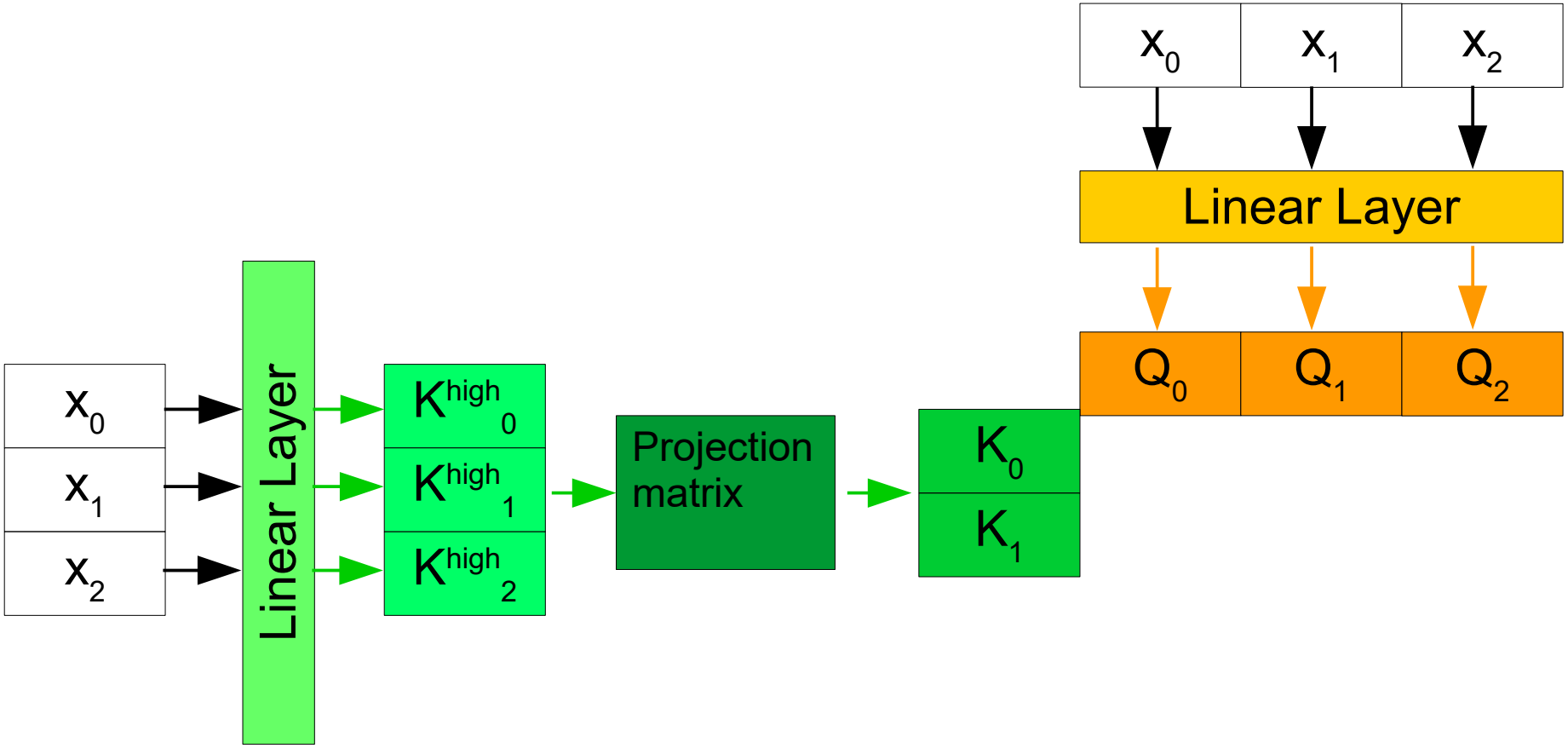
7.2 Linformer: Self-Attention with Linear Complexity



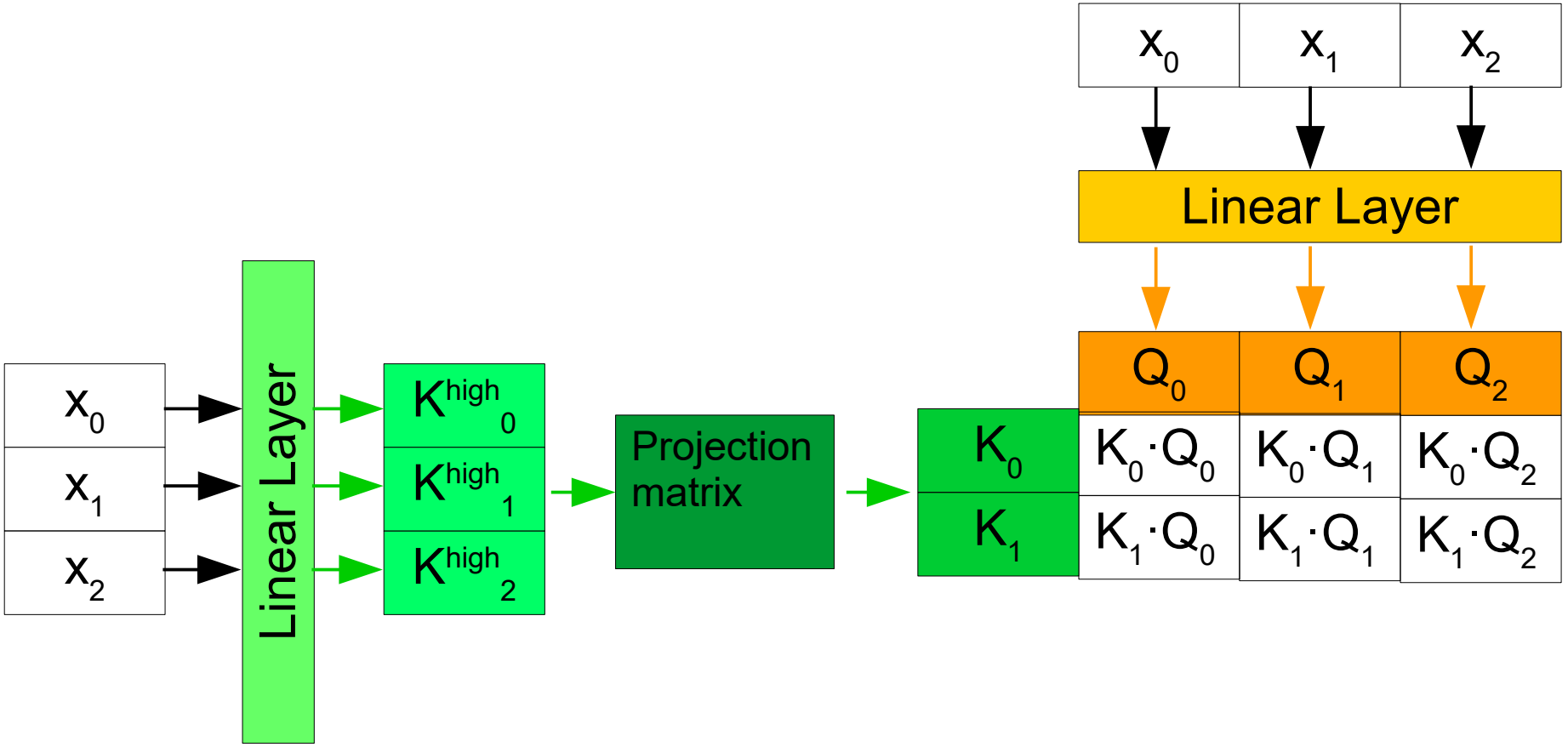
Projection matrix



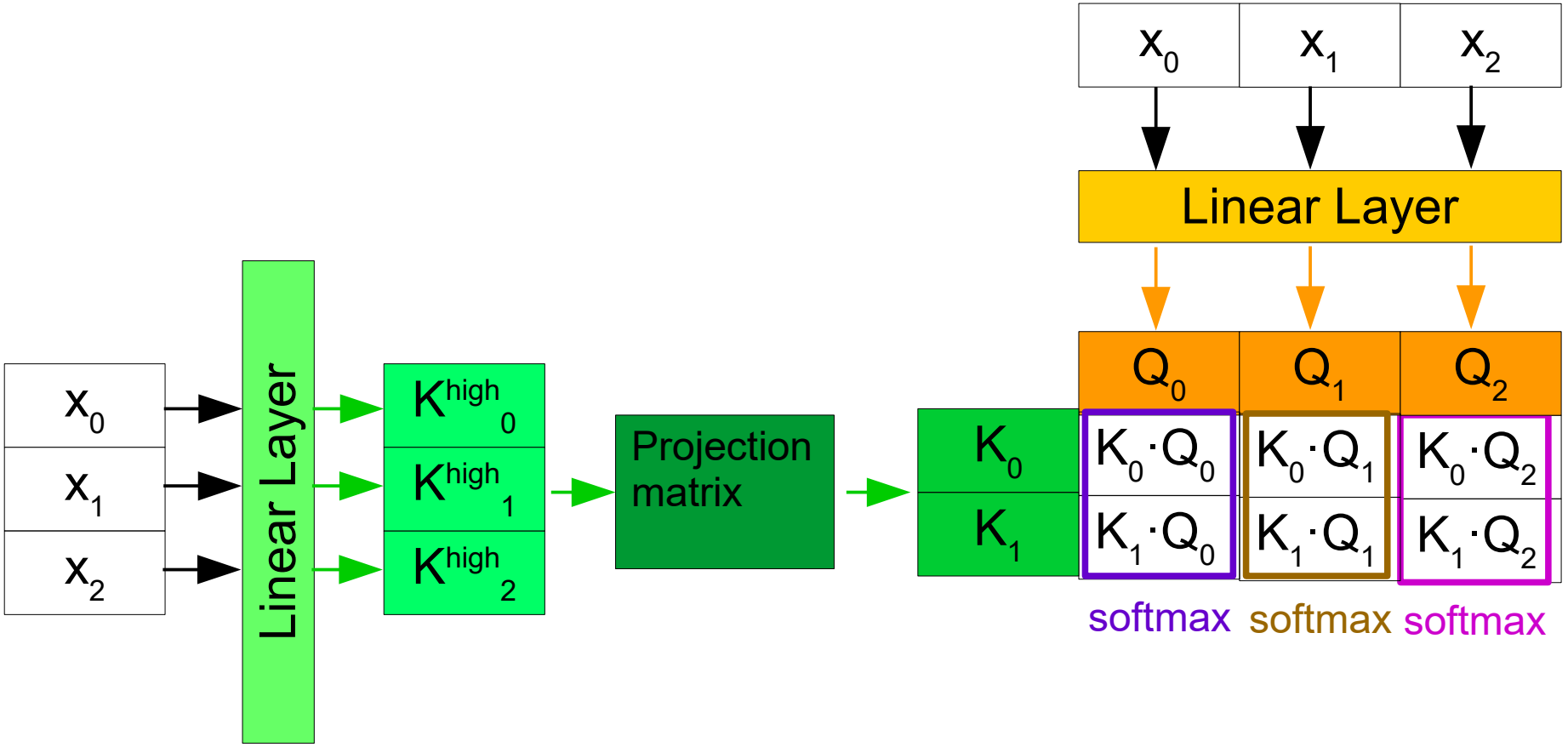
7.2 Linformer: Self-Attention with Linear Complexity



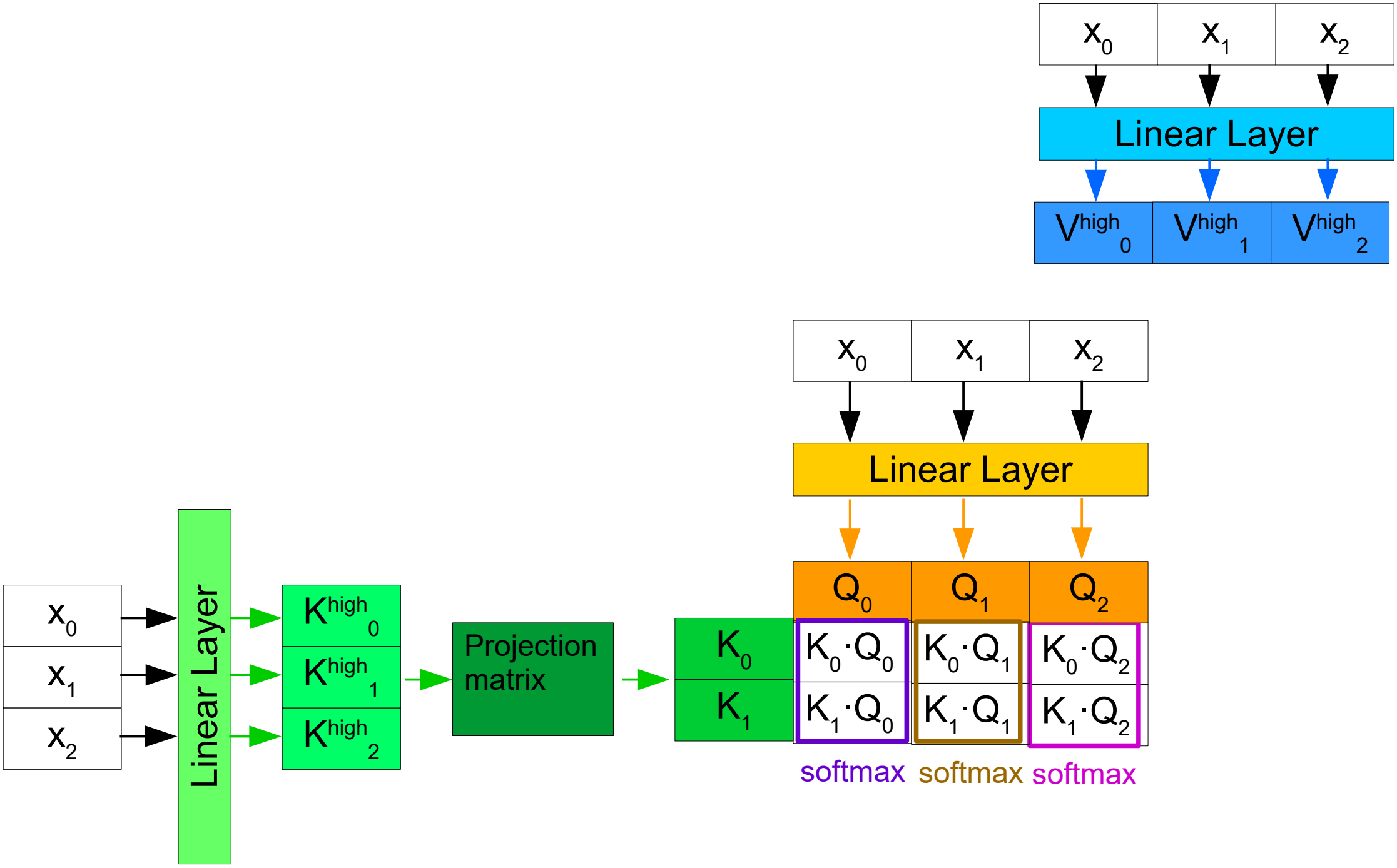
7.2 Linformer: Self-Attention with Linear Complexity



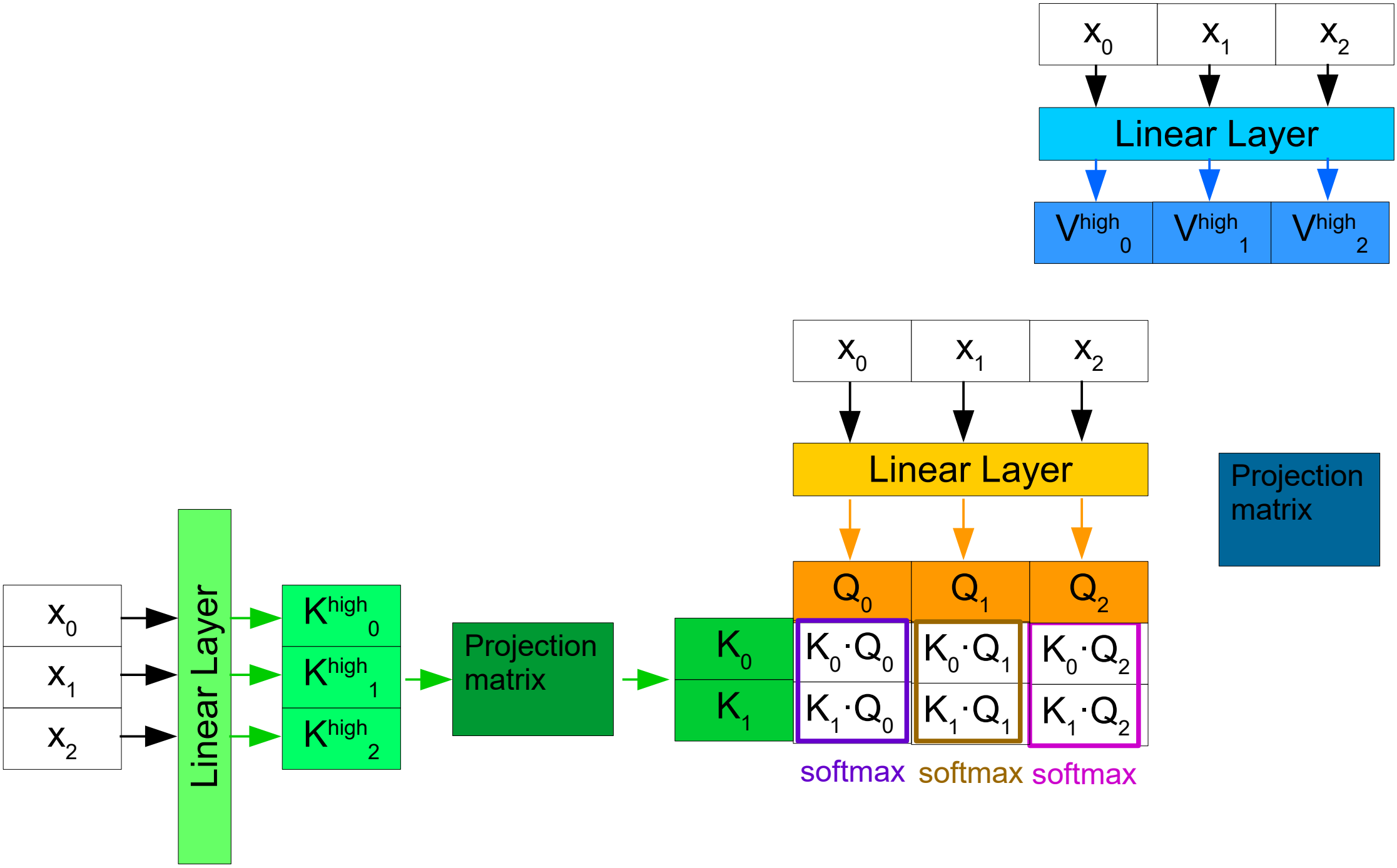
7.2 Linformer: Self-Attention with Linear Complexity



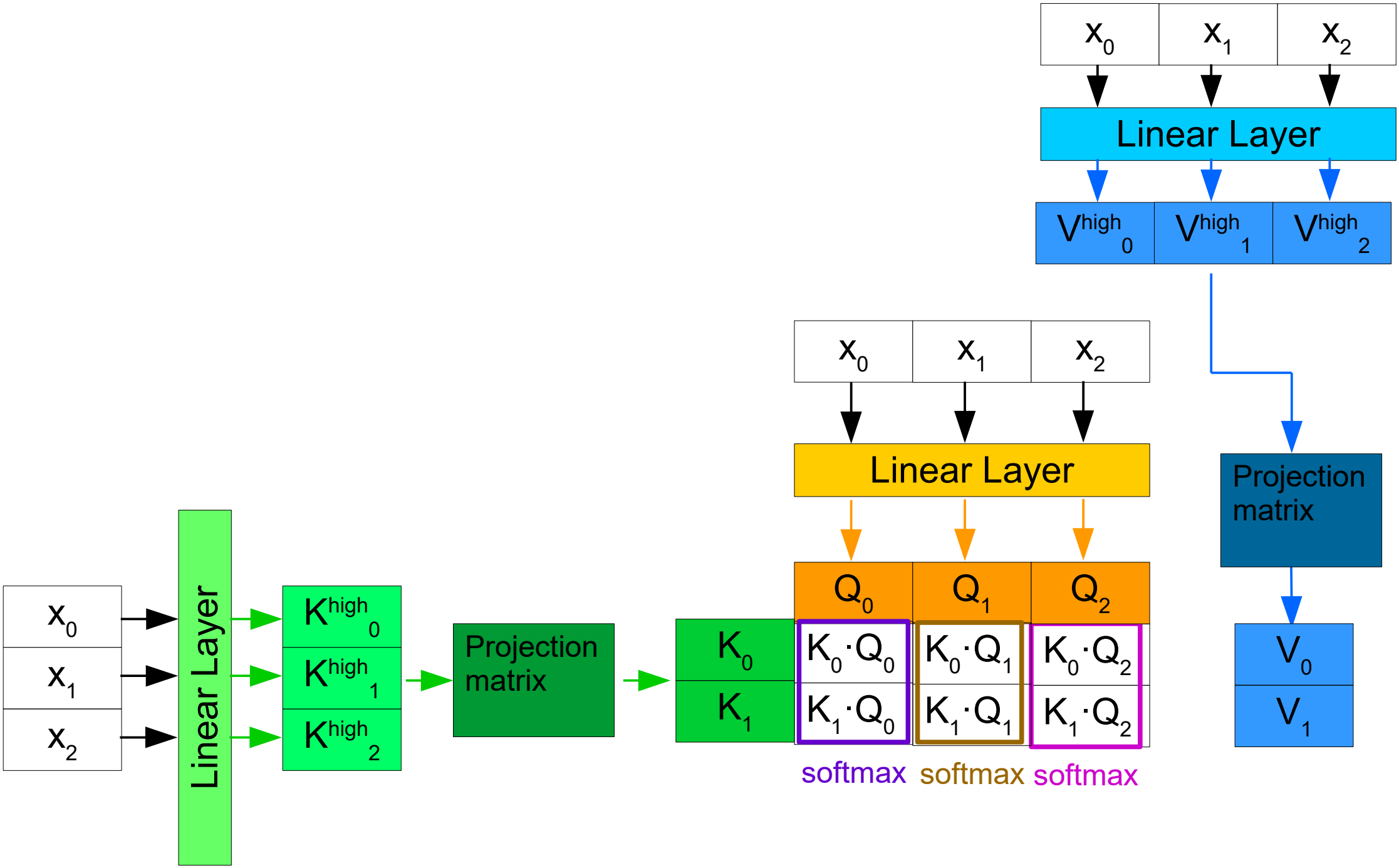
7.2 Linformer: Self-Attention with Linear Complexity



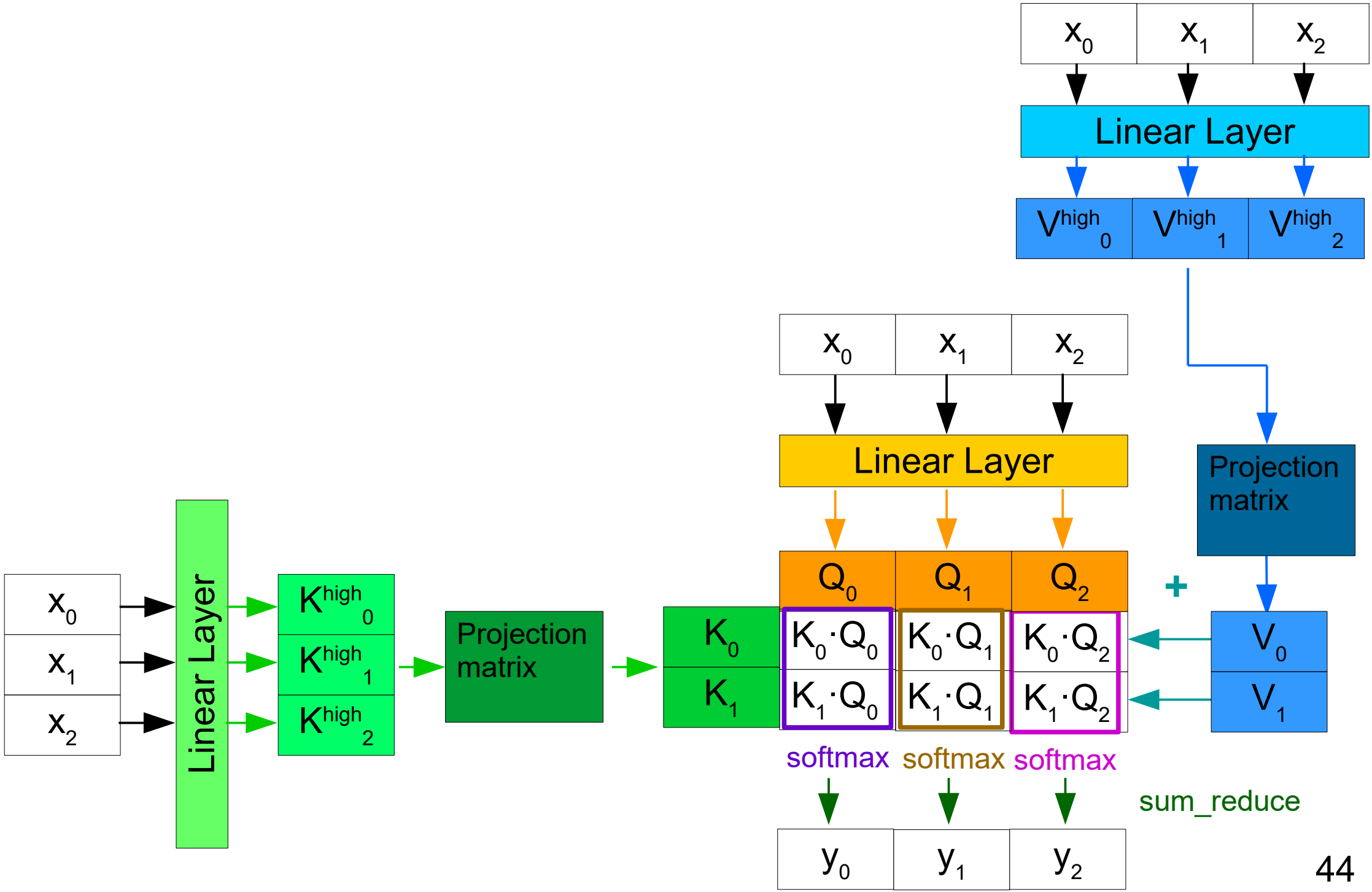
7.2 Linformer: Self-Attention with Linear Complexity



7.2 Linformer: Self-Attention with Linear Complexity



7.2 Linformer: Self-Attention with Linear Complexity



7.3 Reformer: The Efficient Transformer

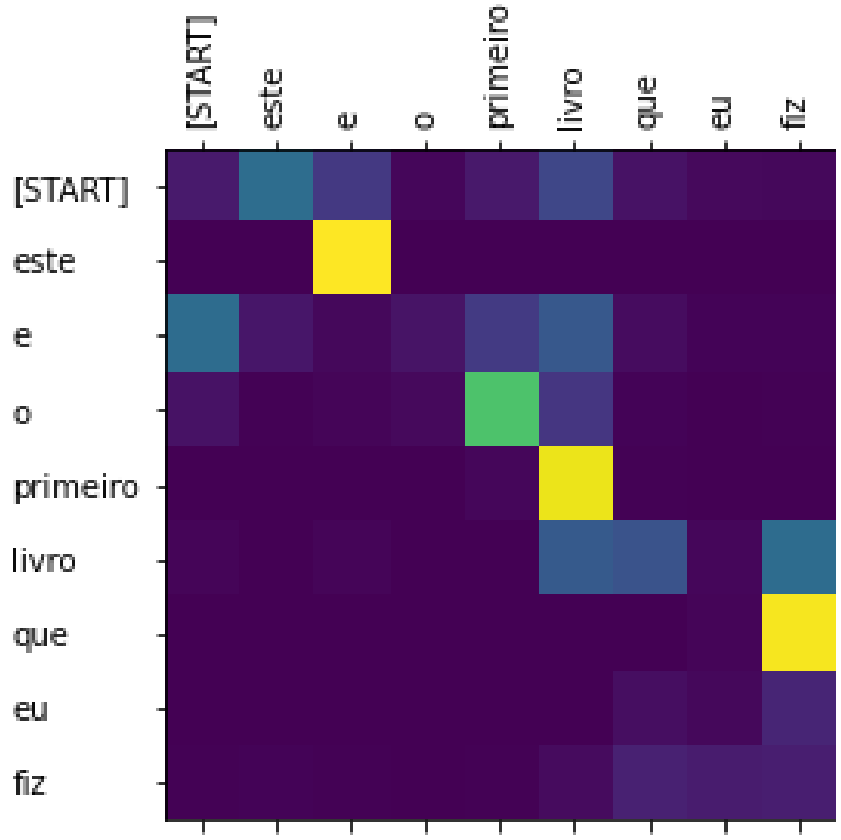


Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

7.3 Reformer: The Efficient Transformer

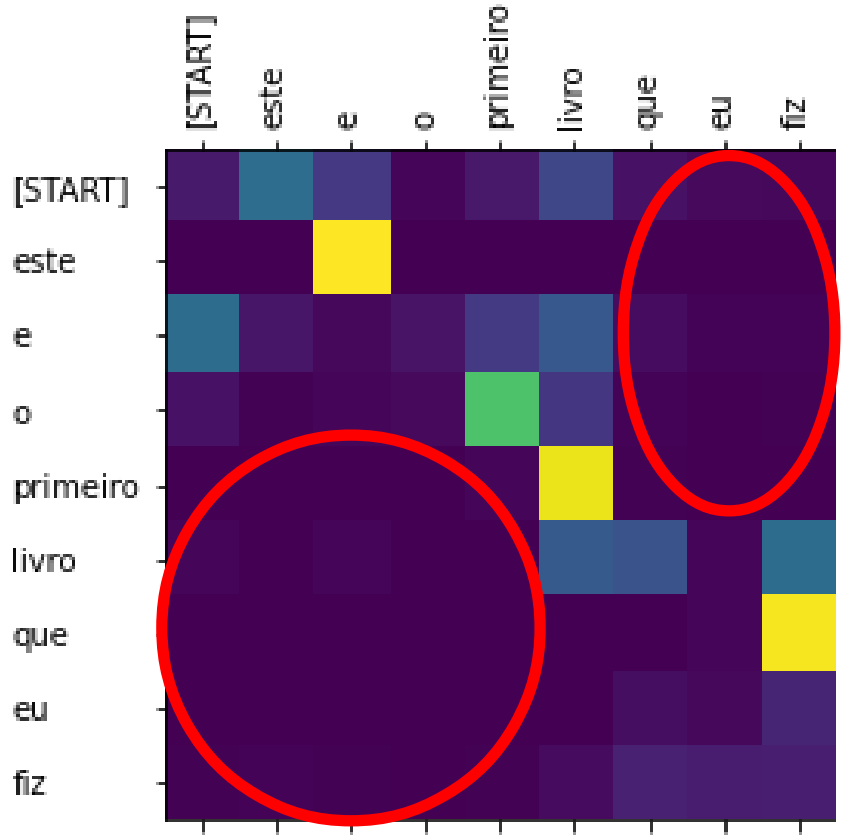
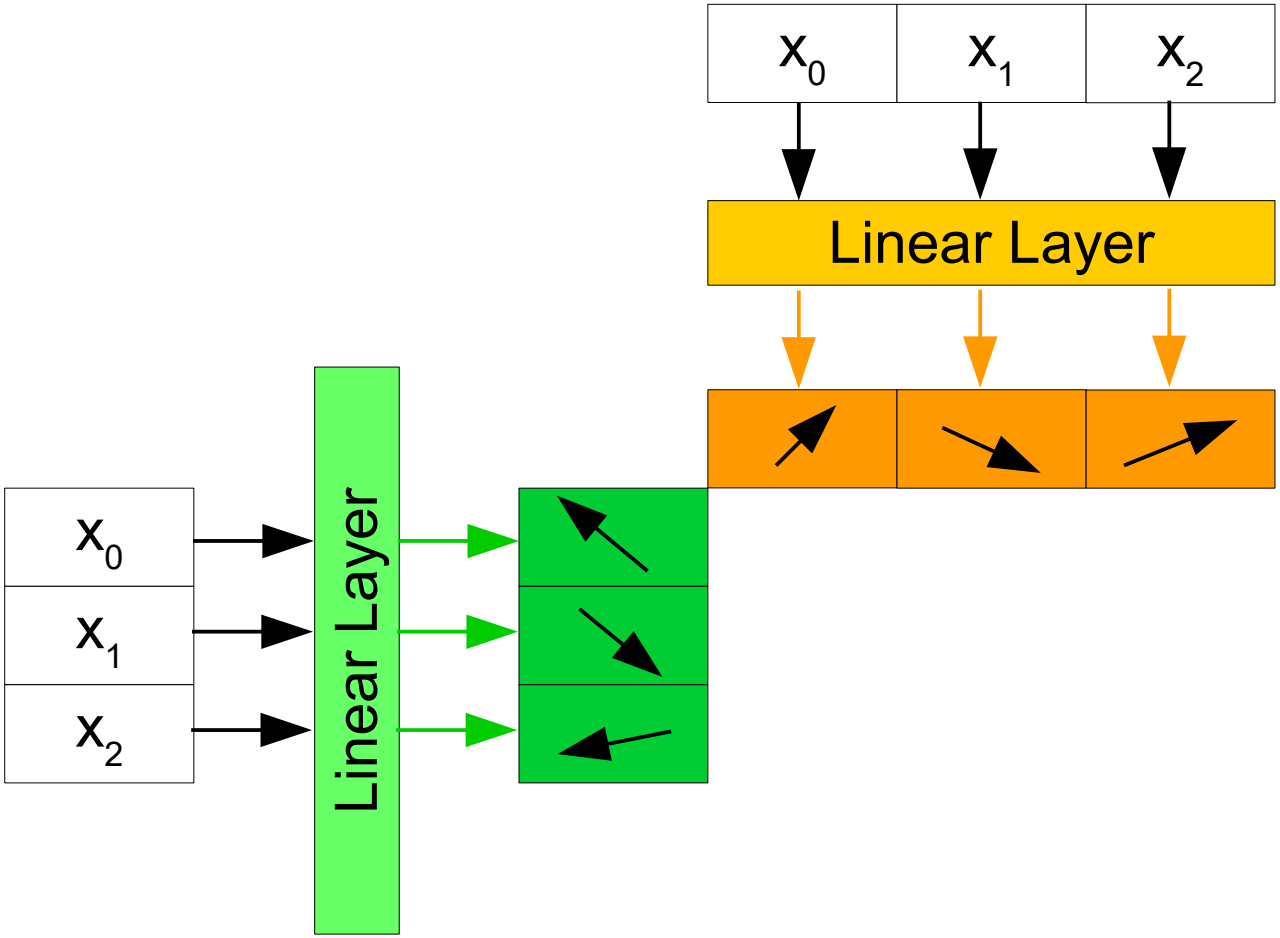
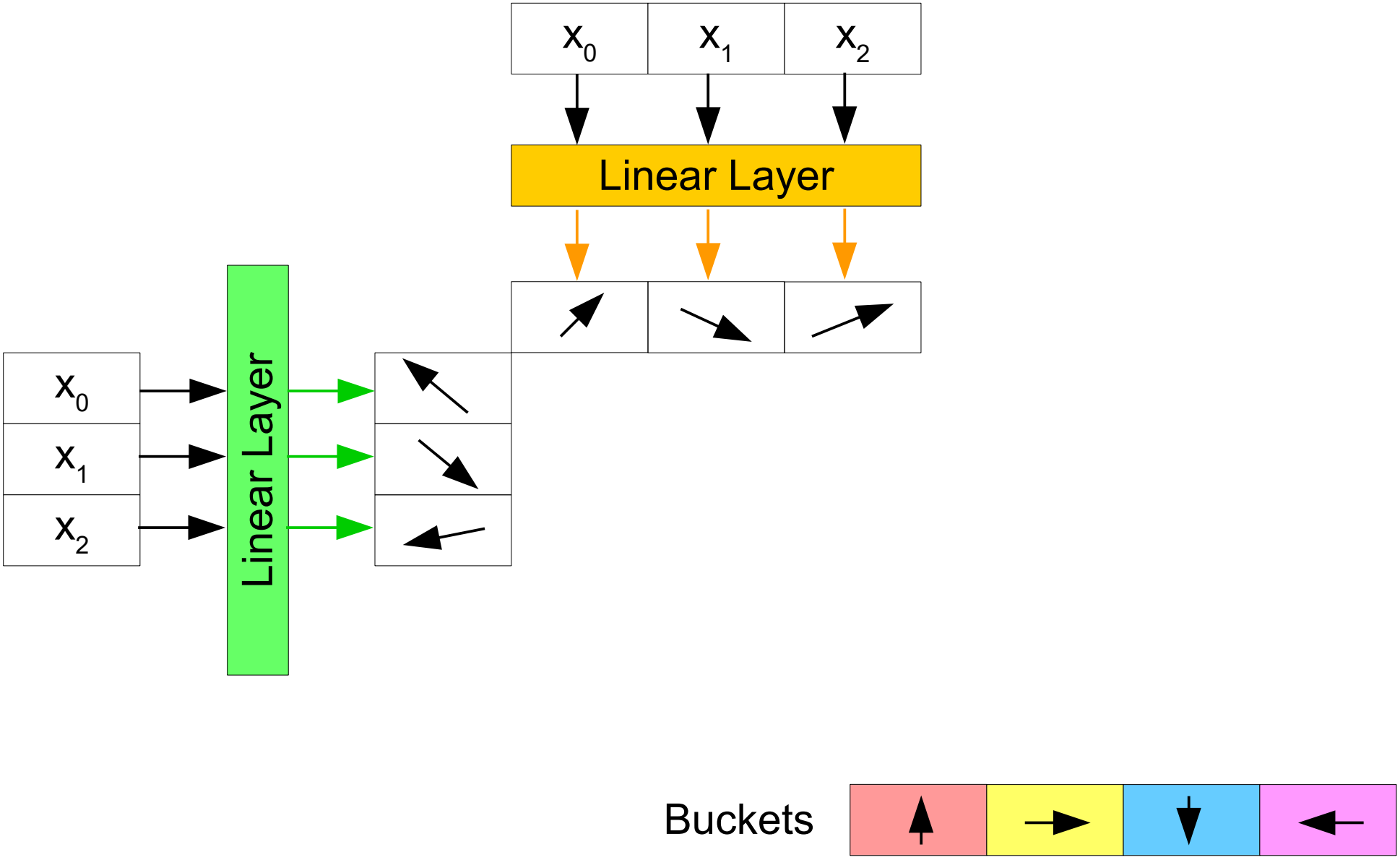


Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

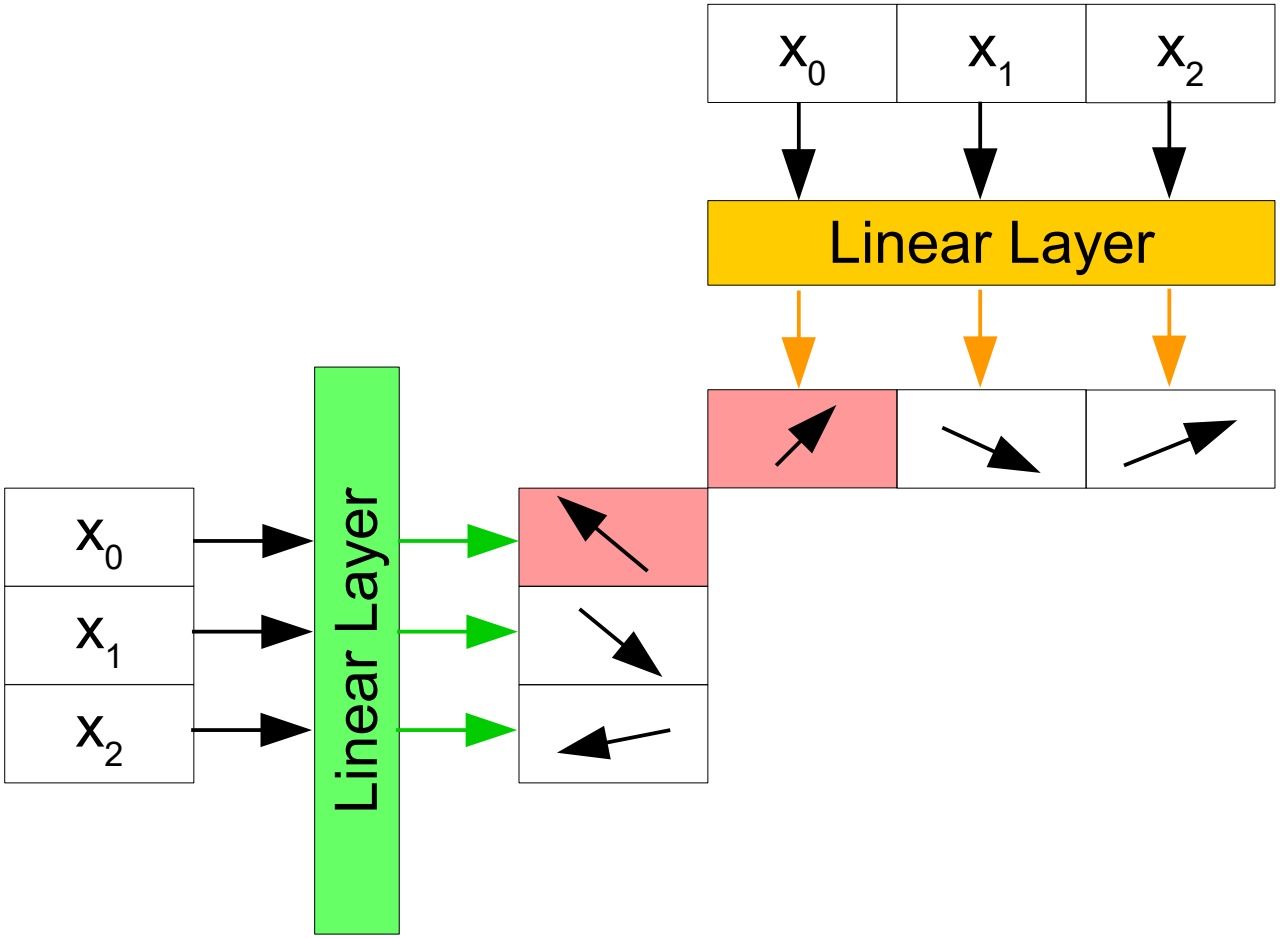
7.3 Reformer: The Efficient Transformer

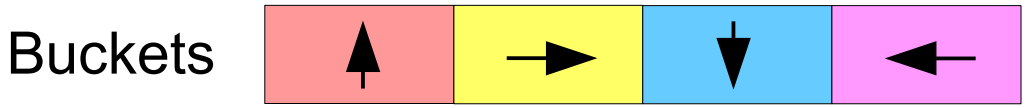


7.3 Reformer: The Efficient Transformer

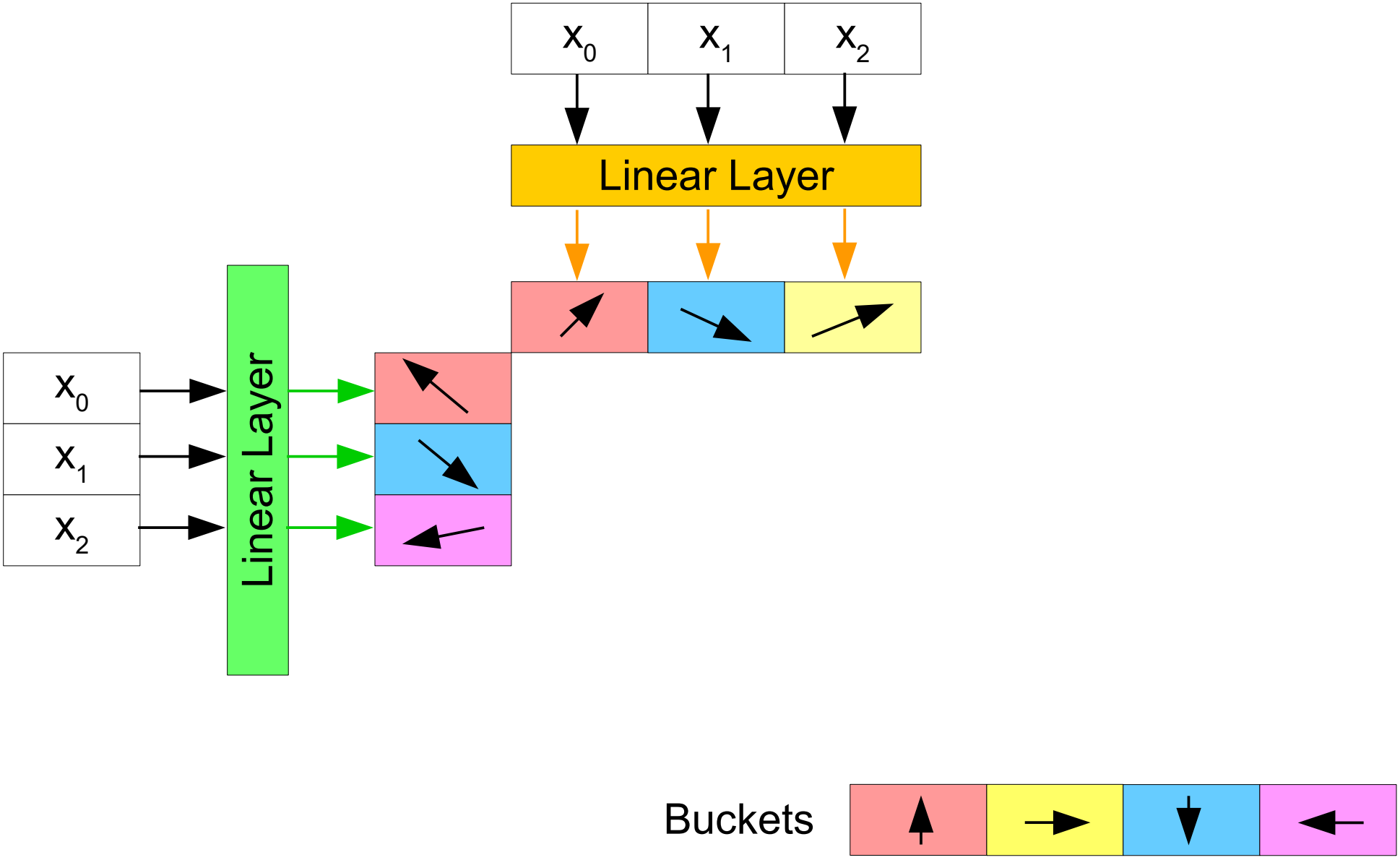


7.3 Reformer: The Efficient Transformer

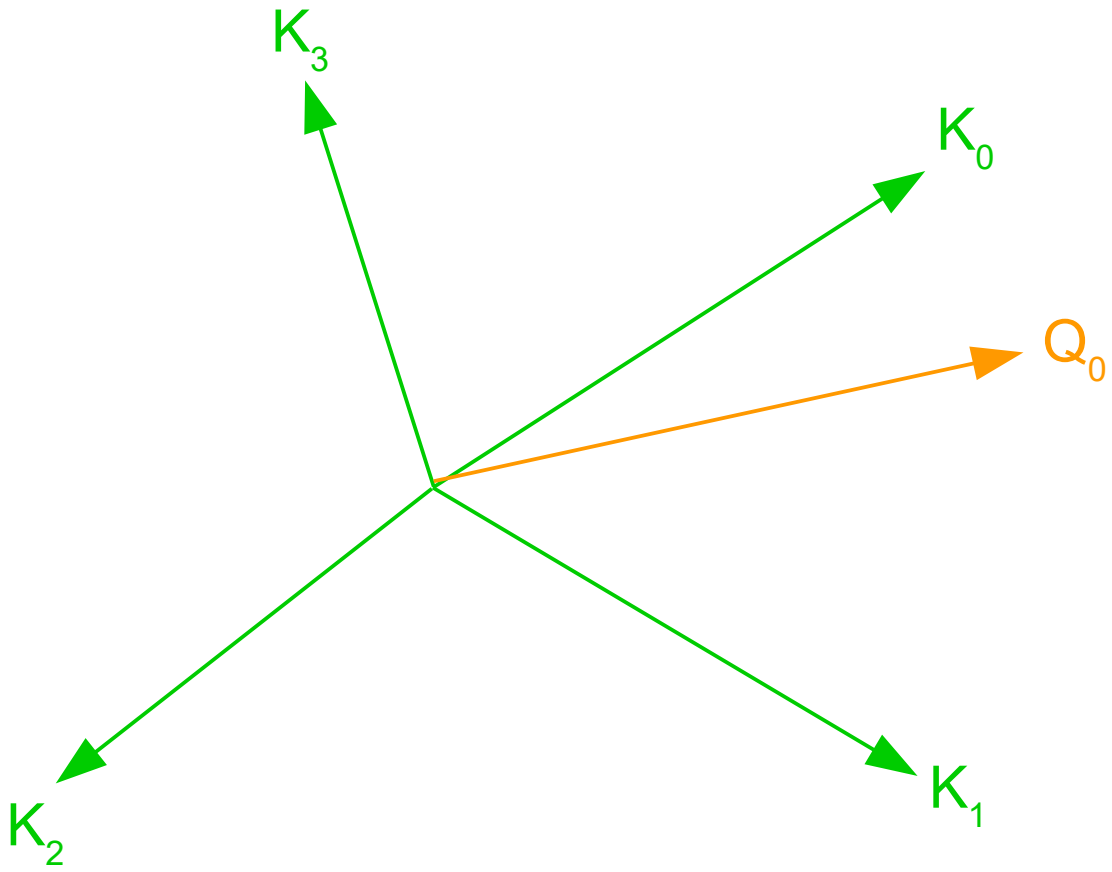




7.3 Reformer: The Efficient Transformer

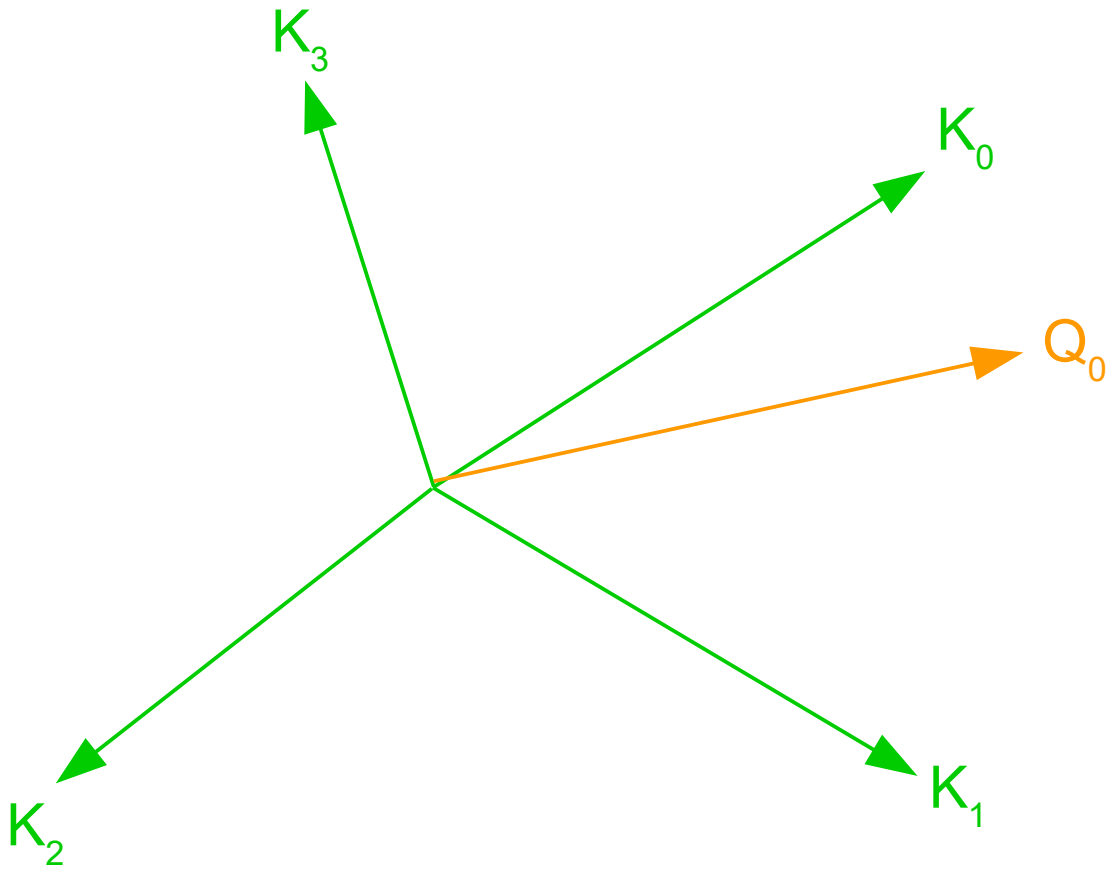


7.3.1 Random Plane Projections



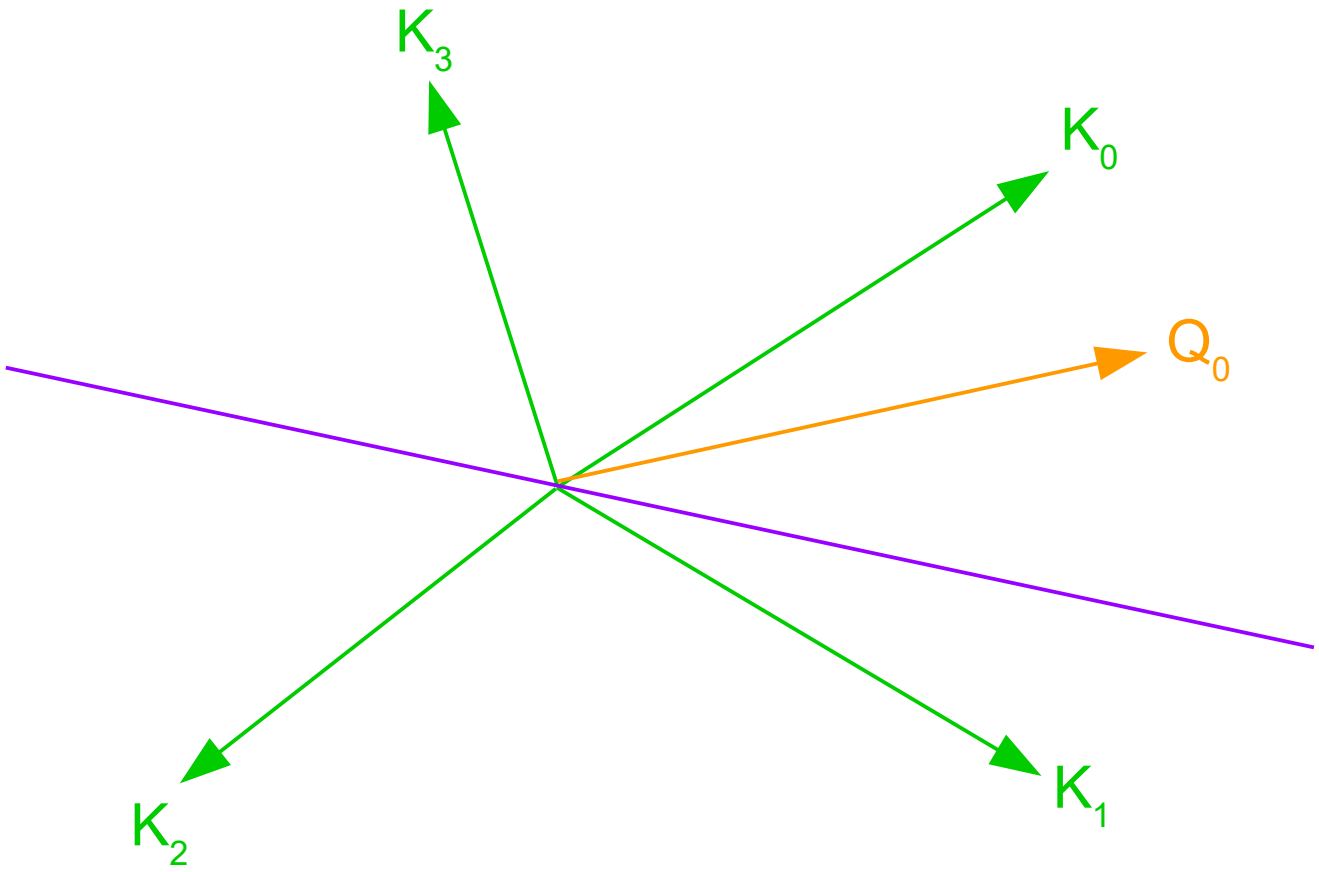
7.3.1 Random Plane Projections

Four buckets
→ Two hyperplanes



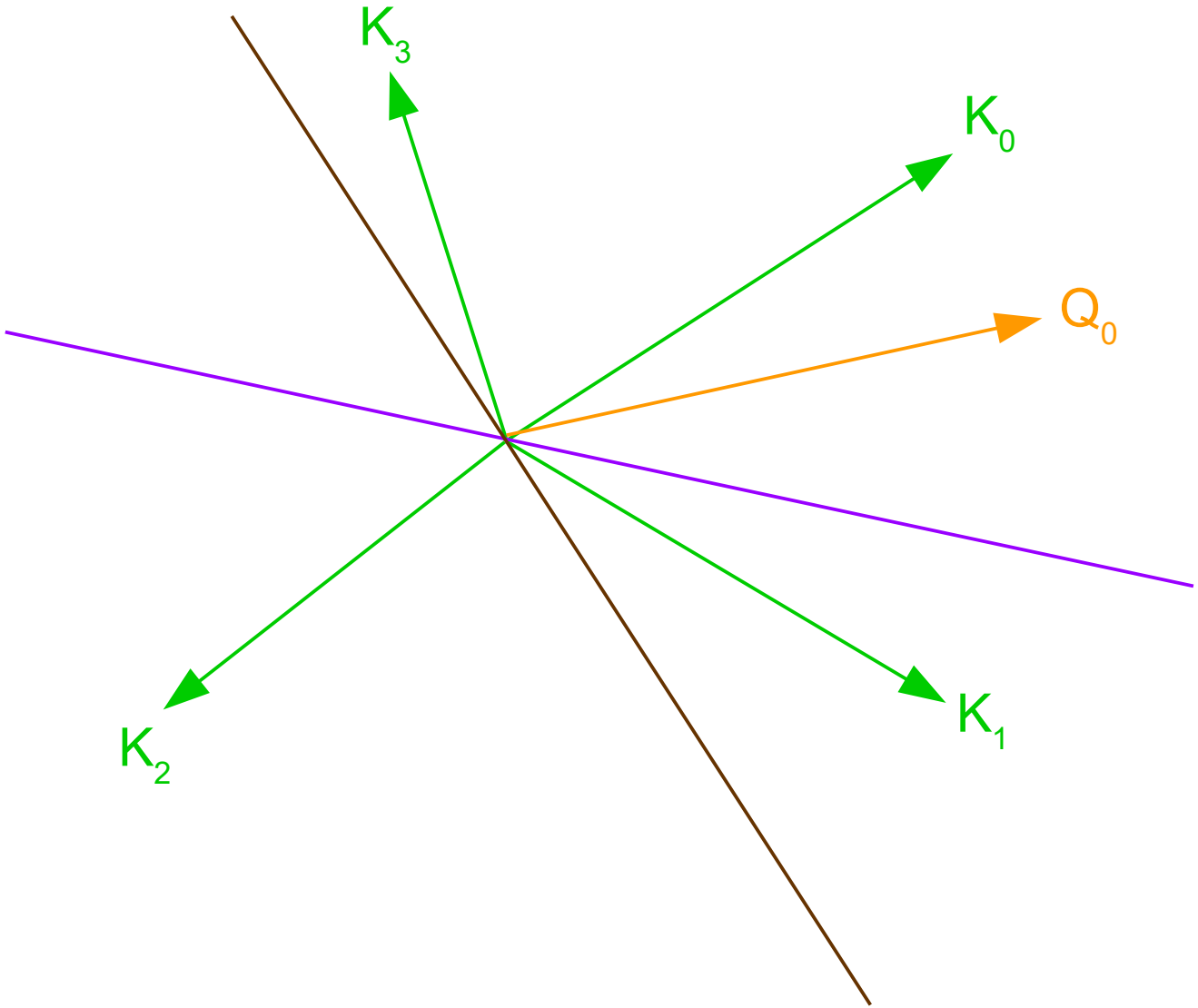
7.3.1 Random Plane Projections

Four buckets
→ Two hyperplanes



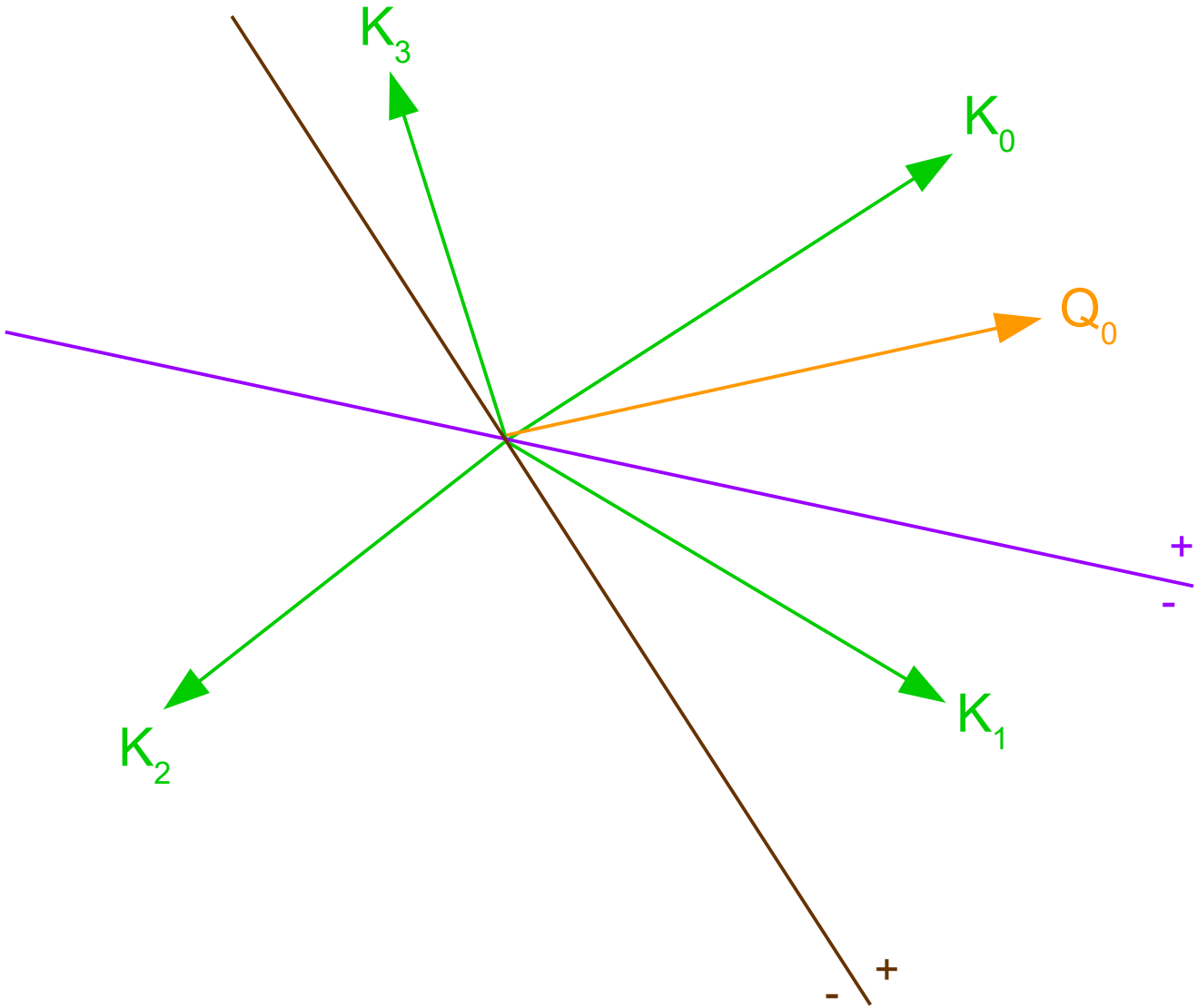
7.3.1 Random Plane Projections

Four buckets
→ Two hyperplanes

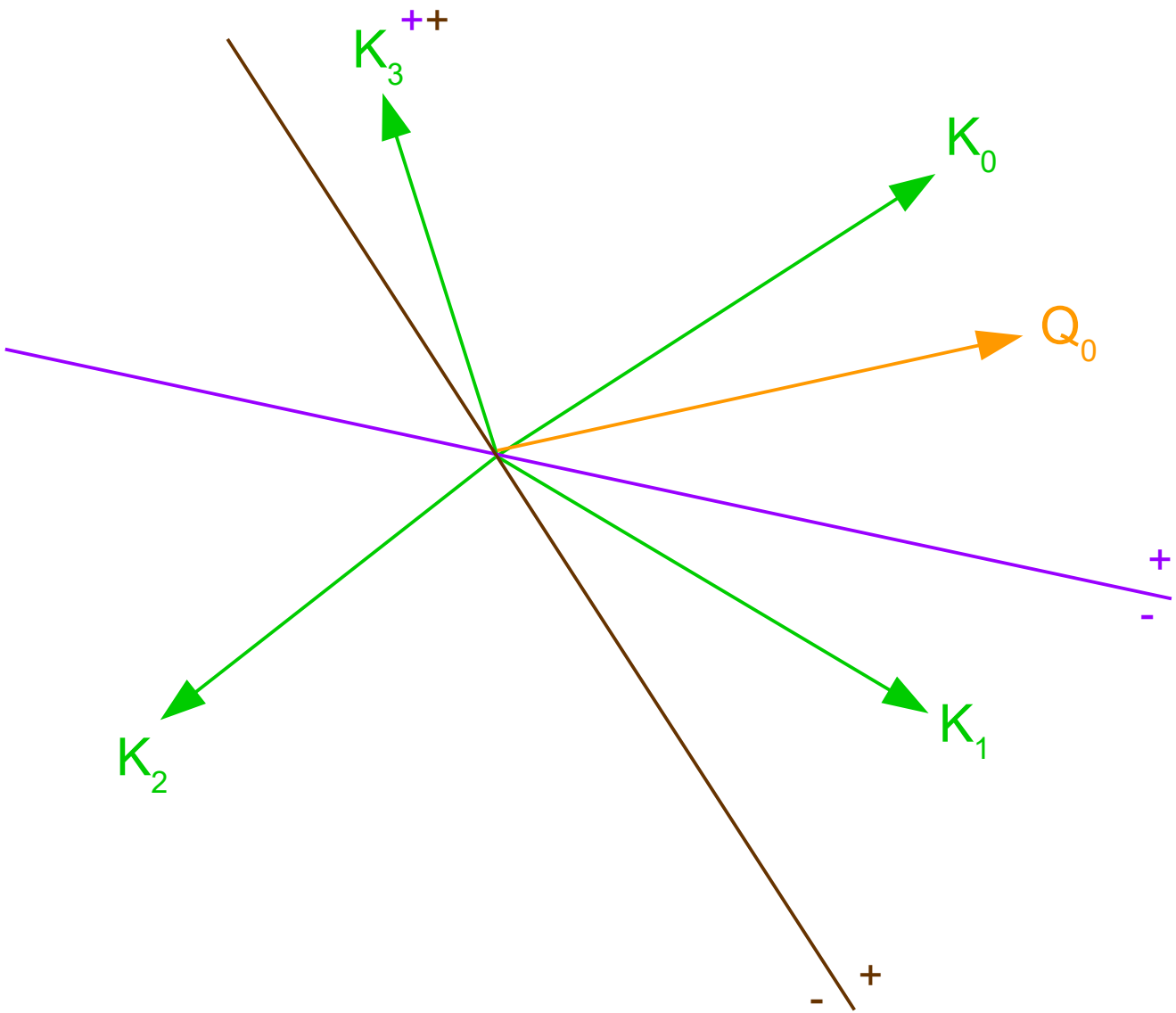


7.3.1 Random Plane Projections

Four buckets
→ Two hyperplanes

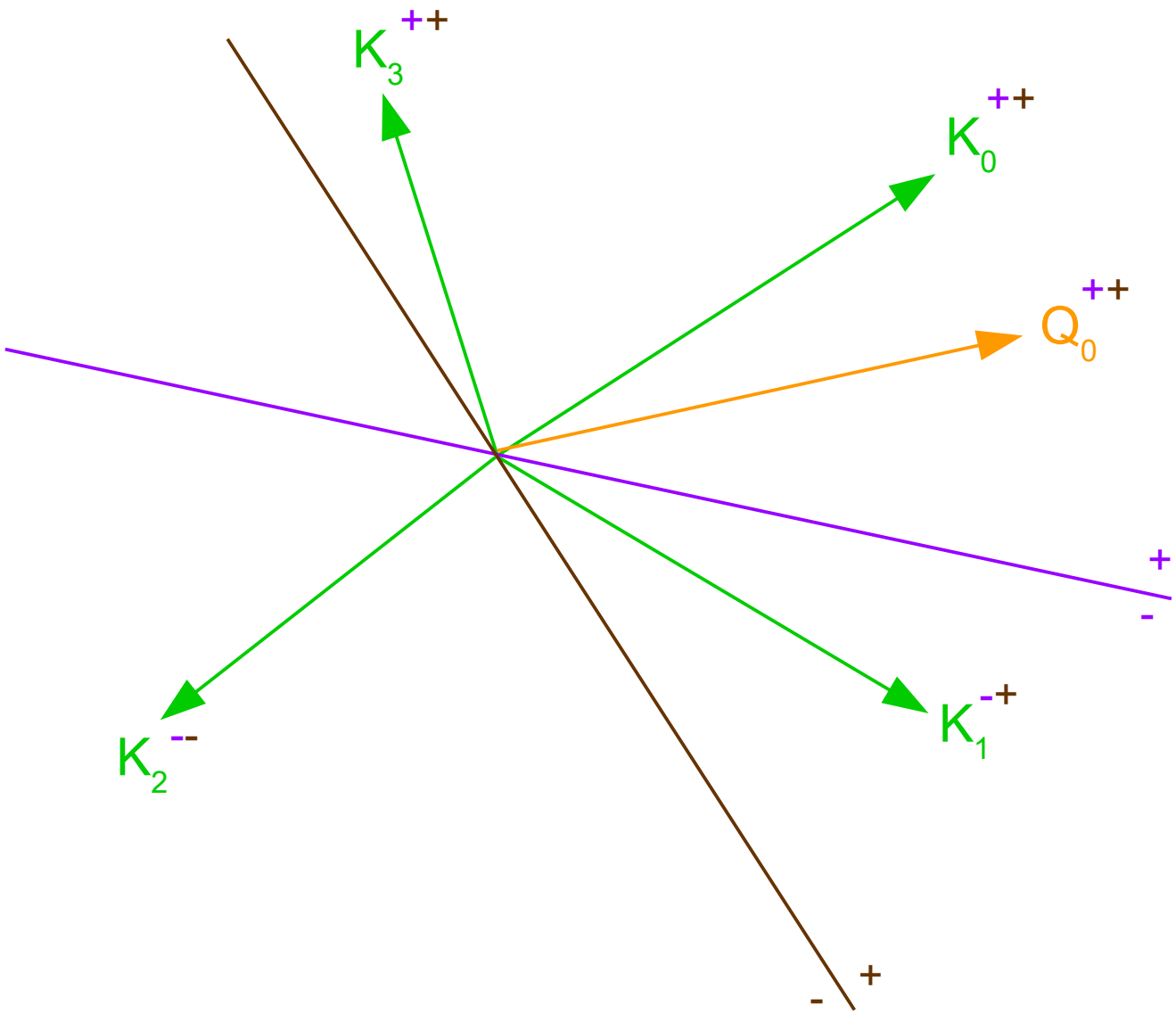


7.3.1 Random Plane Projections



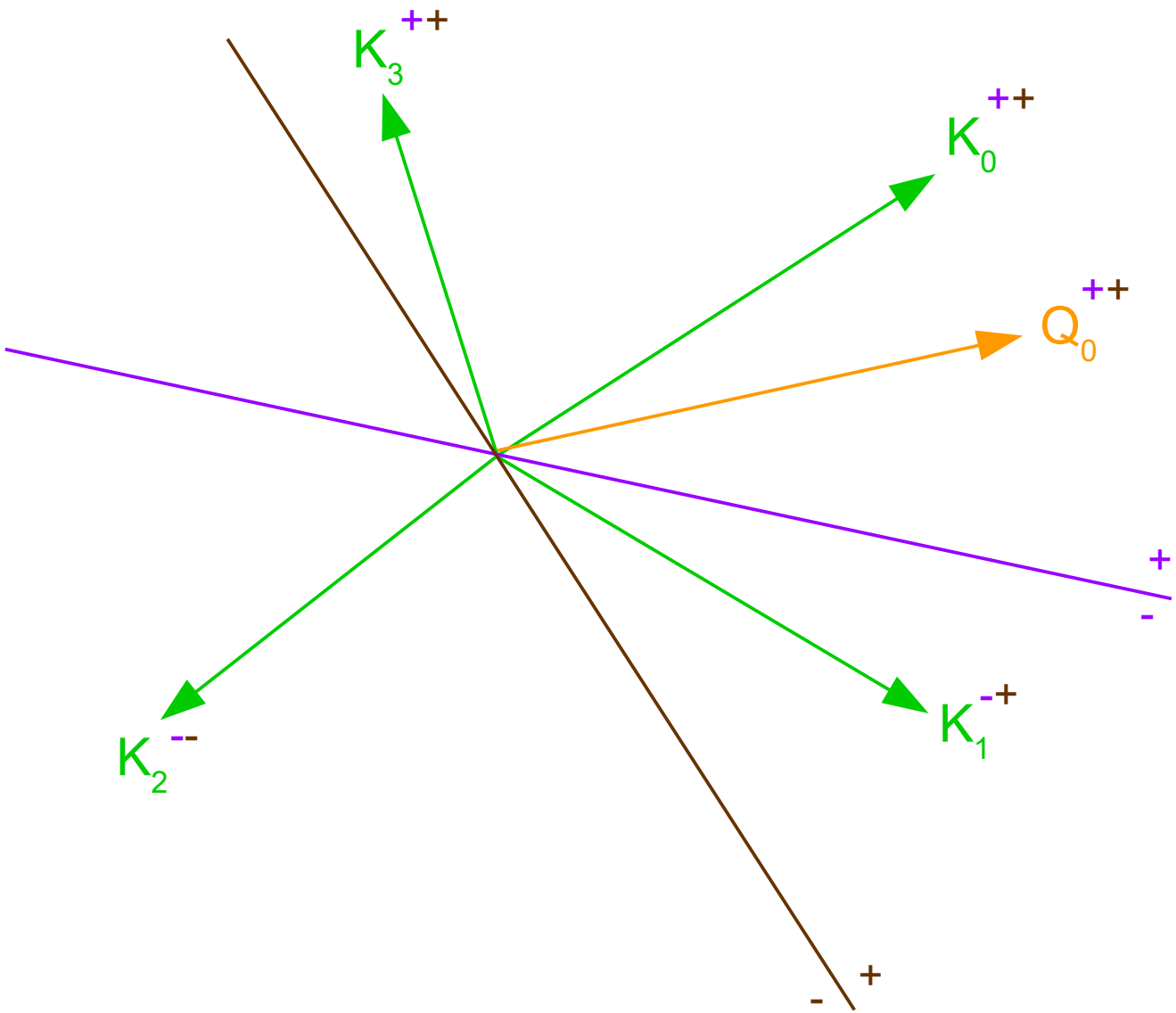
Four buckets
→ Two hyperplanes

7.3.1 Random Plane Projections



Four buckets
→ Two hyperplanes

7.3.1 Random Plane Projections



Four buckets
→ Two hyperplanes

Buckets

++	+-	-+	--
K_0		K_1	K_2
K_3			
Q_0			

7.3 Reformer: The Efficient Transformer

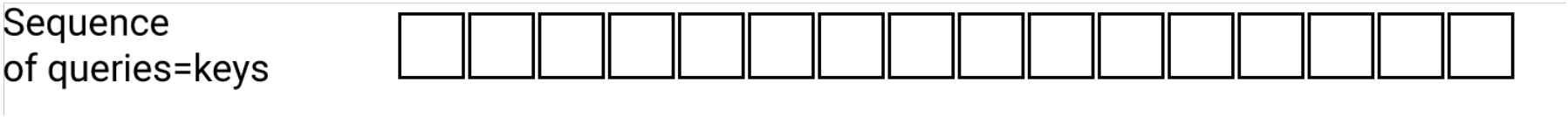


Image source:
Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The Efficient Transformer.
ArXiv, abs/2001.04451.

7.3 Reformer: The Efficient Transformer

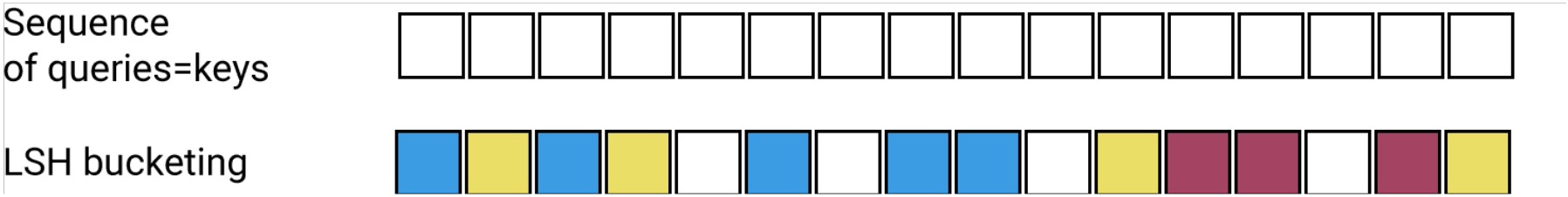


Image source:
Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The Efficient Transformer.
ArXiv, abs/2001.04451.

7.3 Reformer: The Efficient Transformer

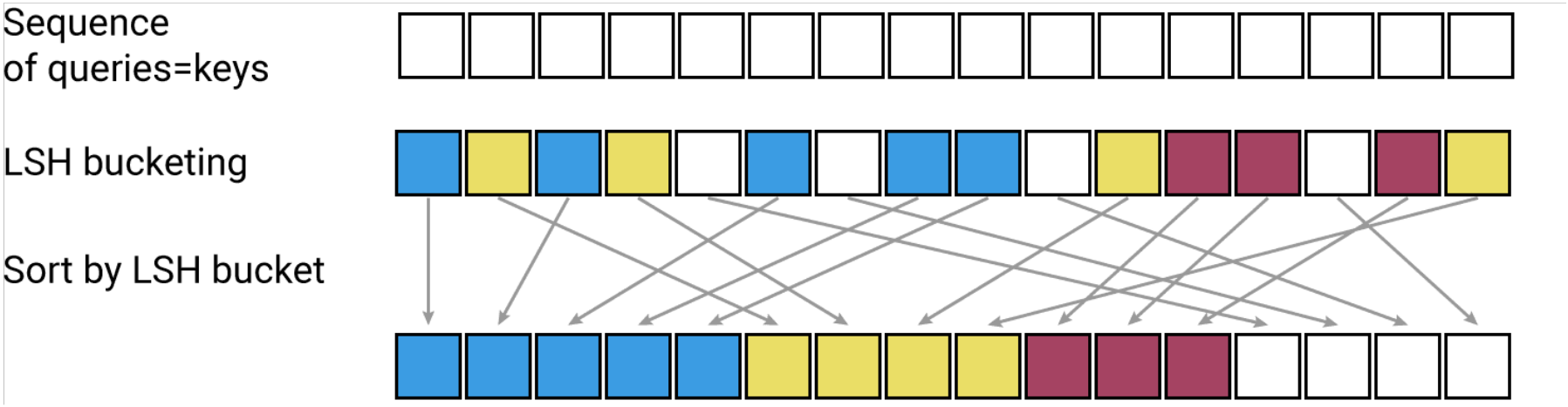


Image source:
Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The Efficient Transformer.
ArXiv, abs/2001.04451.

7.3 Reformer: The Efficient Transformer

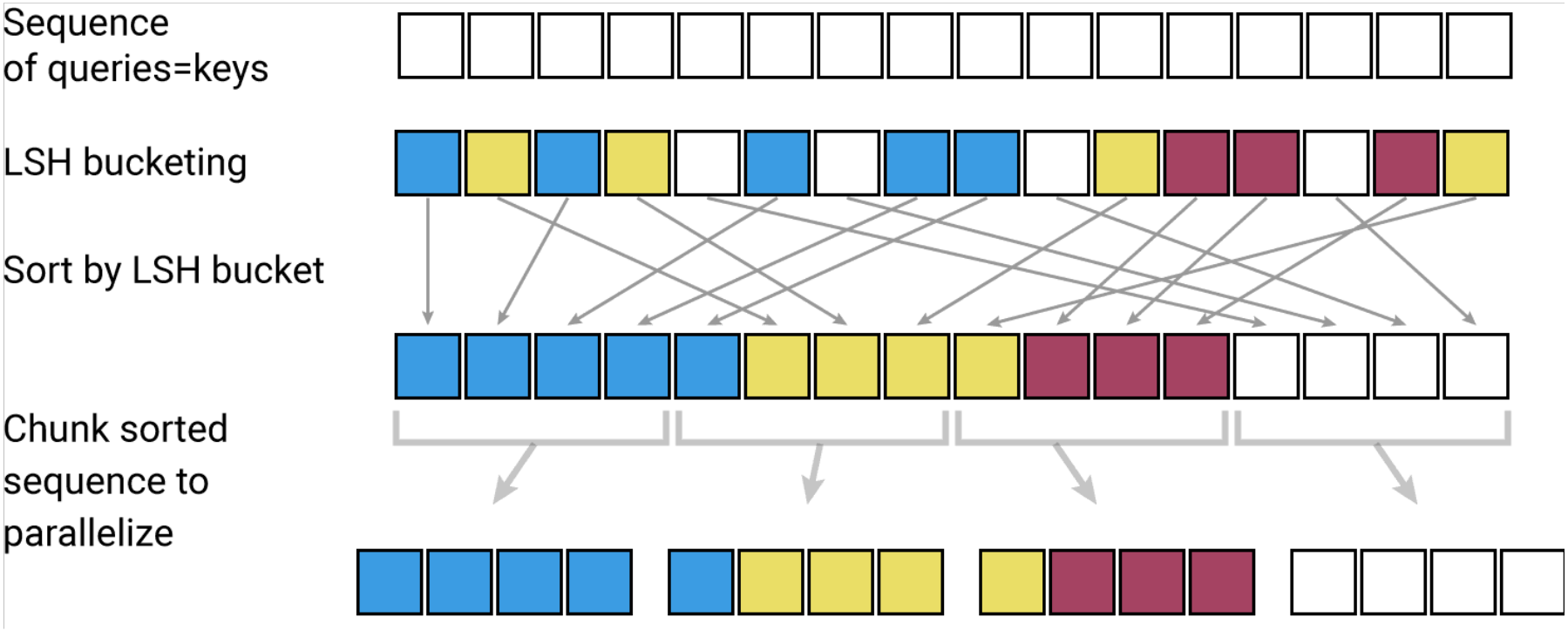


Image source:
Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The Efficient Transformer.
ArXiv, abs/2001.04451.

7.3 Reformer: The Efficient Transformer

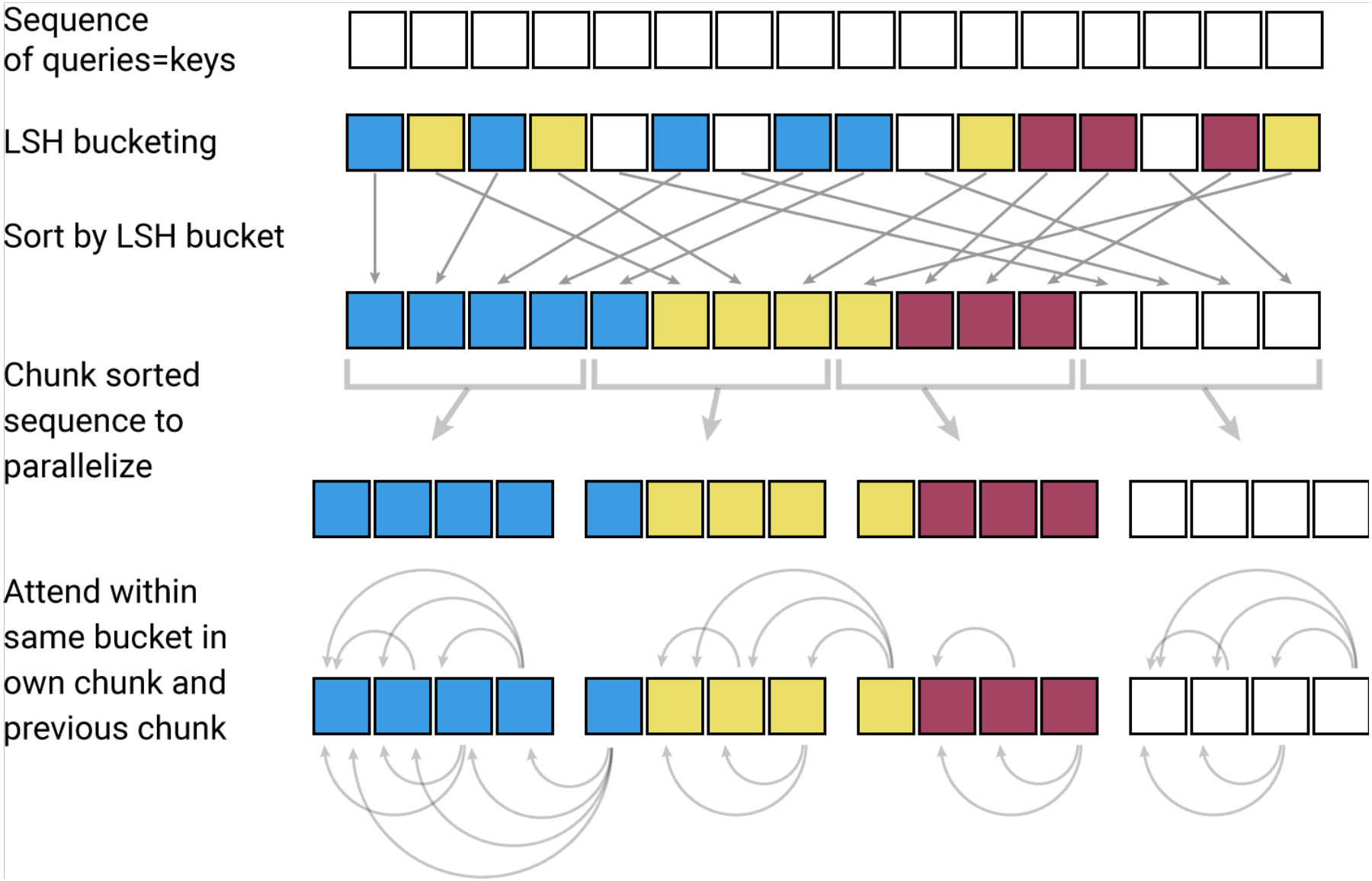
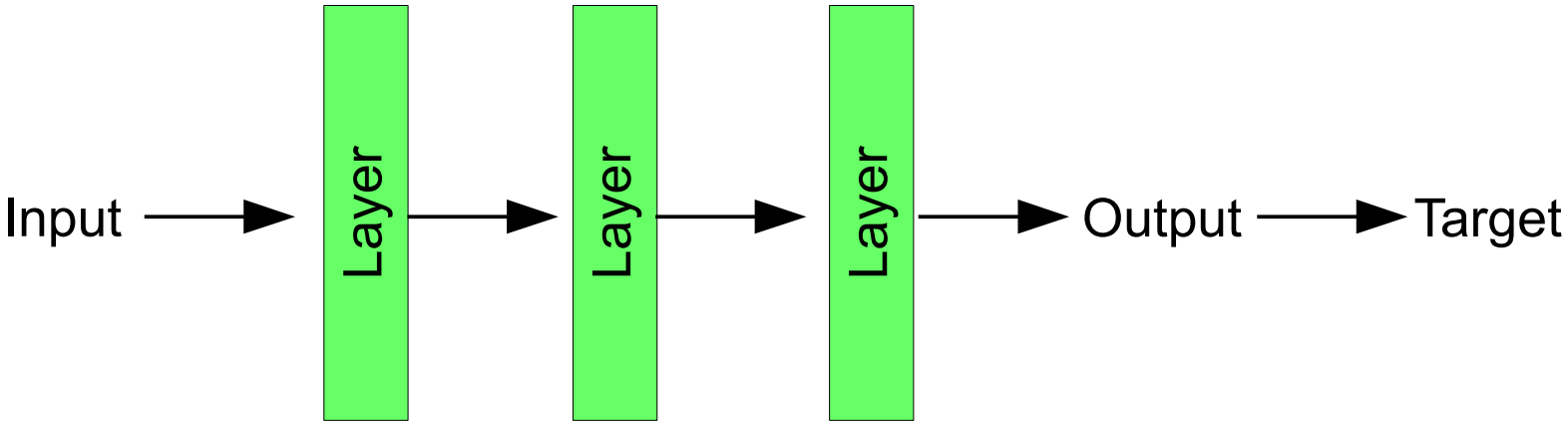
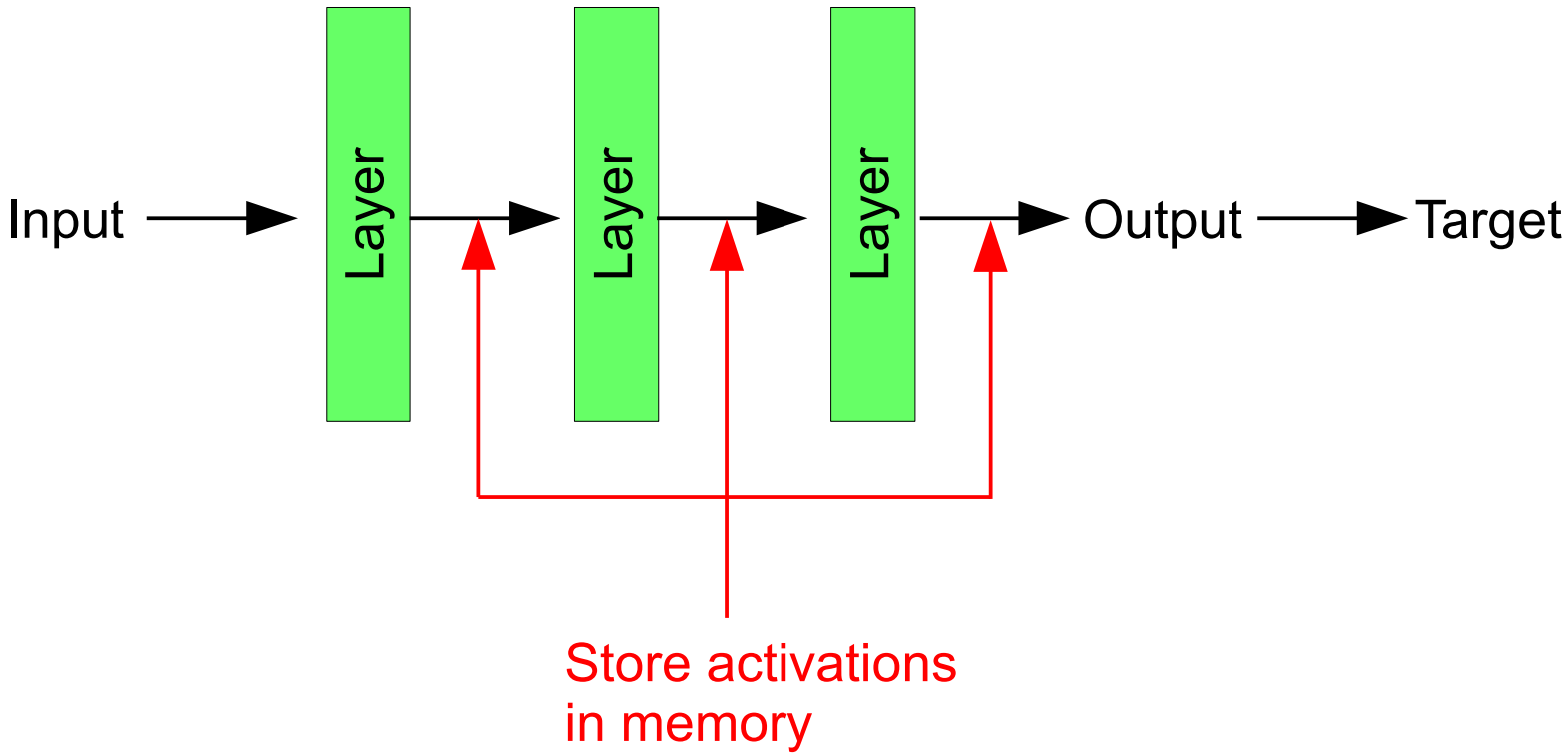


Image source:
Kitaev, N., Kaiser, L., & Levskaya, A. (2020). Reformer: The Efficient Transformer. ArXiv, abs/2001.04451.

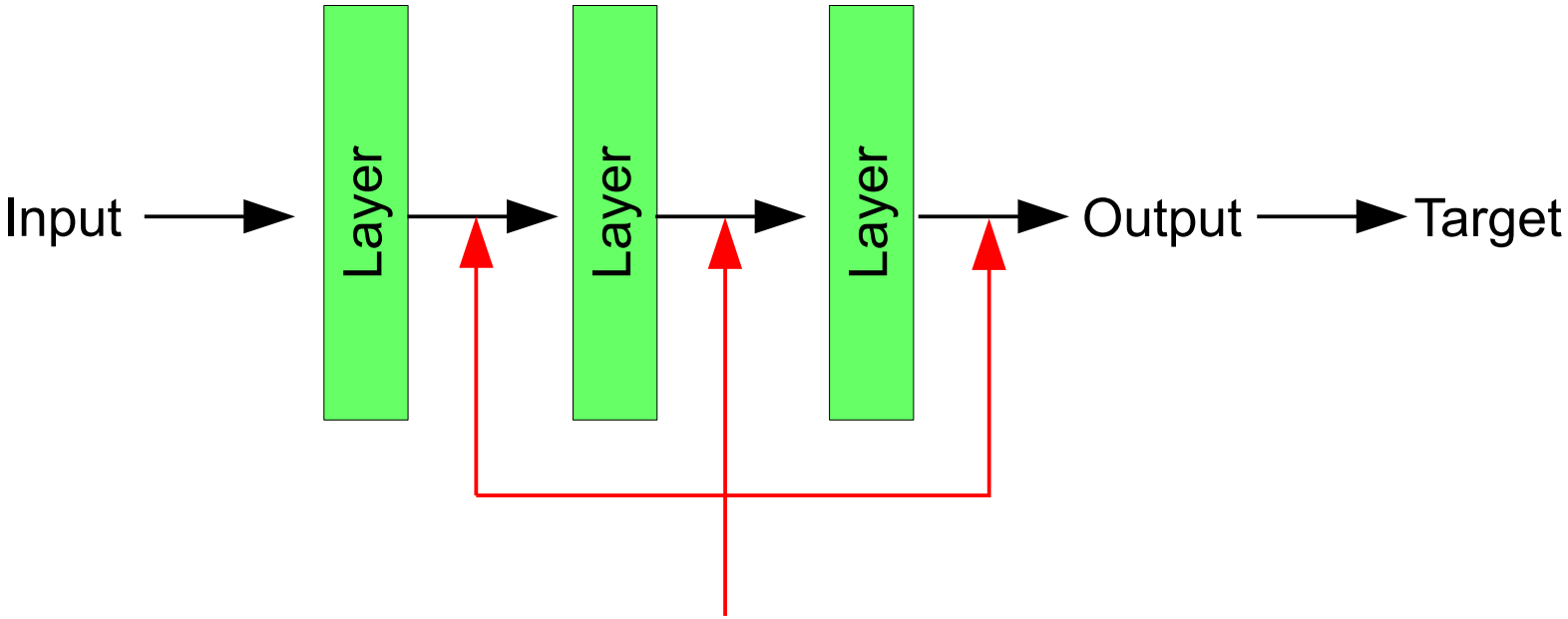
7.3 Memory problem – Backpropagation



7.3 Memory problem – Backpropagation



7.3 Memory problem – Backpropagation



Store activations in memory

including:

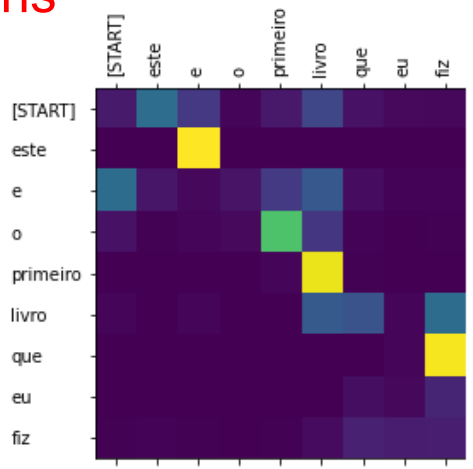


Image source:
https://www.tensorflow.org/images/tutorials/transformer/attention_map_portuguese.png
(call date: 20.07.22)

7.3 Reformer: The Efficient Transformer

Residual connection

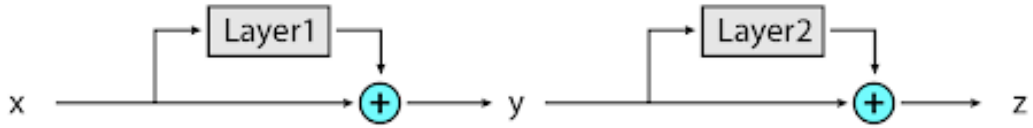
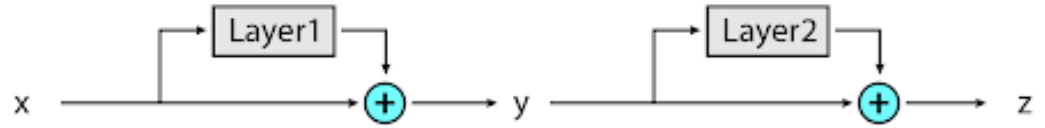


Image source:
https://1.bp.blogspot.com/-l3GO6OKfPPQ/Xh-9qQA6HvI/AAAAAAAAFNM/ioxl9LL_GoghLIjTVGLu2Pm8ghclT22egCEwYBhgL/s1600/image4.png (call date: 29.01.22)

7.3 Reformer: The Efficient Transformer

Residual connection



Reversible layer
(forward pass)

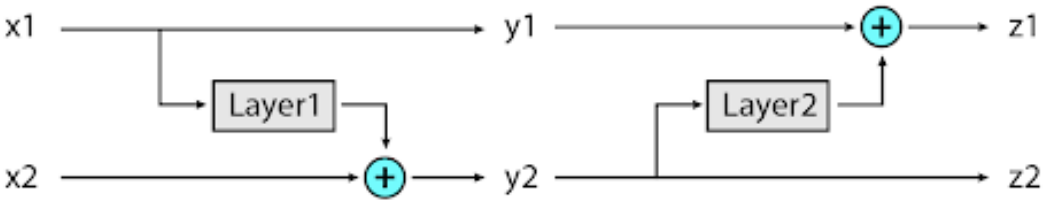
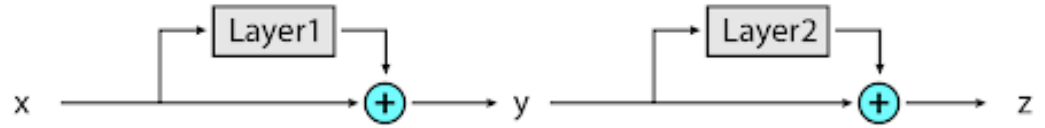


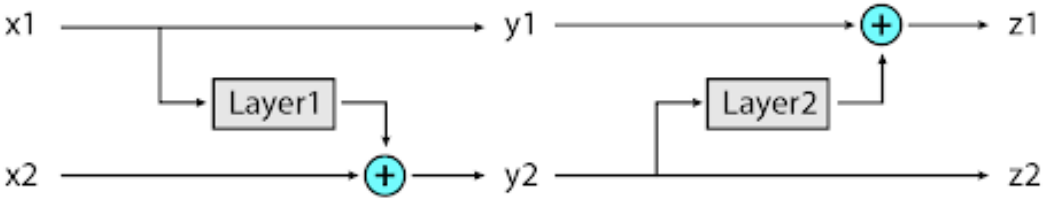
Image source:
https://1.bp.blogspot.com/-l3GO6OKfPPQ/Xh-9qQA6HvI/AAAAAAAAFNM/ioxl9LL_GoghLIjTVGLu2Pm8ghclT22egCEwYBhgL/s1600/image4.png (call date: 29.01.22)

7.3 Reformer: The Efficient Transformer

Residual connection



Reversible layer (forward pass)



Reversible layer (backward pass)

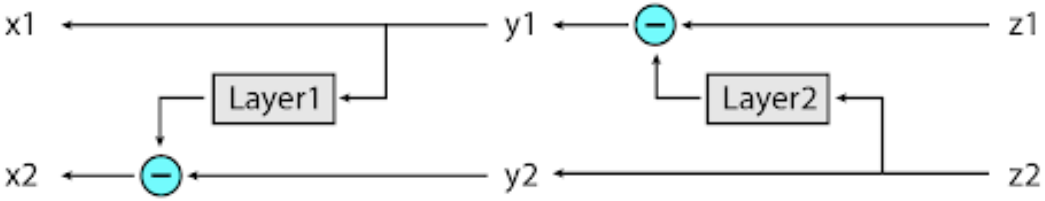
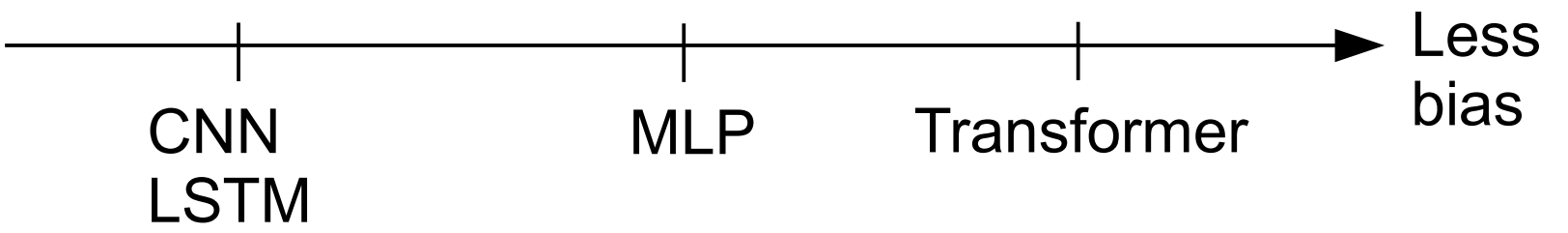


Image source:
https://1.bp.blogspot.com/-l3GO6OKfPPQ/Xh-9qQA6HvI/AAAAAAAAFNM/ioxl9LL_GoghLIjTVGLu2Pm8ghclT22egCEwYBhgL/s1600/image4.png (call date: 29.01.22)

8. Conclusions

- Transform data into a sequence
- More general



- Data hungry 
- Quadratic scale of self-attention
- Outside of context window
- Performance depends on the situation → RNN better? 🤔

Image sources:
https://www.flaticon.com/free-icon/fangs_2068587?term=fangs&page=1&position=3&page=1&position=3&related_id=2068587&origin=search
https://www.flaticon.com/free-icon/file-storage_3616558?term=data&page=1&position=37&page=1&position=37&related_id=3616558&origin=search
https://emojipedia-us.s3.dualstack.us-west-1.amazonaws.com/thumbs/120/whatsapp/326/thinking-face_1f914.png
(call dates: 20.07.22)

AlphaCandy

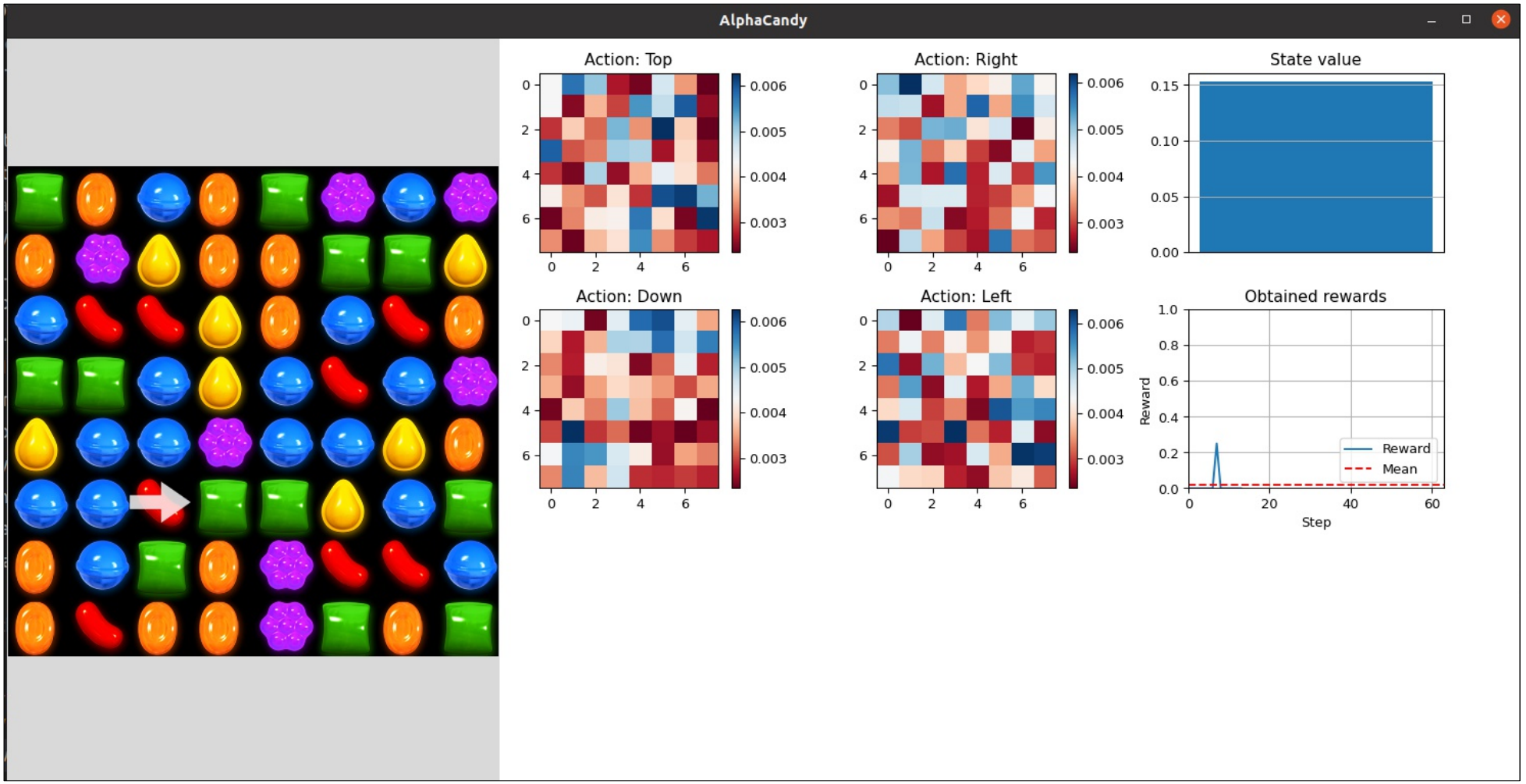


Figure 35: AlphaCandy UI.

Image sources:
https://www.flaticon.com/free-icon/candy-jar_10751352?related_id=10751352&origin=tag
https://emojipedia.org/de/toss-face/march-2022/gesicht-mit-umarmenden-h%C3%A4nden/
(call dates: 20.07.22)

References - Papers

[1] BAHDANAU, D., CHO, K., AND BENGIO, Y. Neural machine translation by jointly learning to align and translate. In 3rd ICLR, Conference Track Proceedings (2015), Y. Bengio and Y. LeCun, Eds.

[2] VASWANI, A., SHAZEER, N., PARMAR, N., USZKOREIT, J., JONES, L., GOMEZ, A. N., KAISER, L. U., AND POLOSUKHIN, I. Attention is all you need. In Advances in Neural Information Processing Systems (2017), I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc.

[3] DOSOVITSKIY, A., BEYER, L., KOLESNIKOV, A., WEISSENBORN, D., ZHAI, X., UNTERTHINER, T., DEHGHANI, M. MINDERER, M., HEIGOLD, G., GELLY, S., USZKOREIT, J., AND HOULSBY, N. An image is worth 16x16 words: Transformers for image recognition at scale. In 9th ICLR (2021), OpenReview.net.

[4] CARION, N., MASSA, F., SYNNAEVE, G., USUNIER, N., KIRILLOV, A., AND ZAGORUYKO, S. End-to-end object detection with transformers. In Computer Vision - ECCV - 16th European Conference, Proceedings, Part I (2020), A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, Eds., vol. 12346 of Lecture Notes in Computer Science, Springer, pp. 213–229.

[5] JIANG, Y., CHANG, S., AND WANG, Z. Transgan: Two pure transformers can make one strong gan, and that can scale up. In Advances in Neural Information Processing Systems 34: NeurIPS 2021 (2021), M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., pp. 14745–14758.

[6] UPADHYAY, U., SHAH, N., RAVIKANTI, S., AND MEDHE, M. Transformer based reinforcement learning for games. vol. Abs/1912.03918.

References - Papers

[7] CHEN, L., LU, K., RAJESWARAN, A., LEE, K., GROVER, A., LASKIN, M., ABBEEL, P., SRINIVAS, A., AND MORDATCH, I. Decision transformer: Reinforcement learning via sequence modeling. In Advances in Neural Information Processing Systems 34: NeurIPS 2020 (2021), M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., pp. 15084–15097.

[8] DEVLIN, J., CHANG, M., LEE, K., AND TOUTANOVA, K. BERT: pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 Conference of the NAACL-HLT, Volume 1 (Long and Short Papers) (2019), J. Burstein, C. Doran, and T. Solorio, Eds., Association for Computational Linguistics, pp. 4171–4186.

[9] BROWN, T. B., MANN, B., RYDER, N., SUBBIAH, M., KAPLAN, J., DHARIWAL, P., NEELAKANTAN, A., SHYAM, P., SASTRY, G., ASKELL, A., AGARWAL, S., HERBERT-VOSS, A., KRUEGER, G., HENIGHAN, T., CHILD, R., RAMESH, A., ZIEGLER, D. M., WU, J., WINTER, C., HESSE, C., CHEN, M., SIGLER, E., LITWIN, M., GRAY, S., CHESS, B., CLARK, J., BERNER, C., MCCANDLISH, S., RADFORD, A., SUTSKEVER, I., AND AMODEI, D. Language models are few-shot learners. In Advances in Neural Information Processing Systems 33: NeurIPS 2020 (2020), H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds.