Attention and Transformers Networks summary

E. Patsiouras, I. Pitas
Aristotle University of Thessaloniki
pitas@csd.auth.gr
www.aiia.csd.auth.gr
Version 1.5

Artificial Intelligence &
Information Analysis Lab

Introduction

- ***Transformers*** (“Attention is all you need”) [VAS2017] were originally introduced to tackle natural language processing (NLP) tasks:
  - Machine translation (BERT [DEV2018])
  - Text summarization (ROBERTA [LIU 2019])
  - Question/answering systems (DISTILBERT [SANH2019])
  - Document generation (GPT v3 [BRO2020])

Introduction

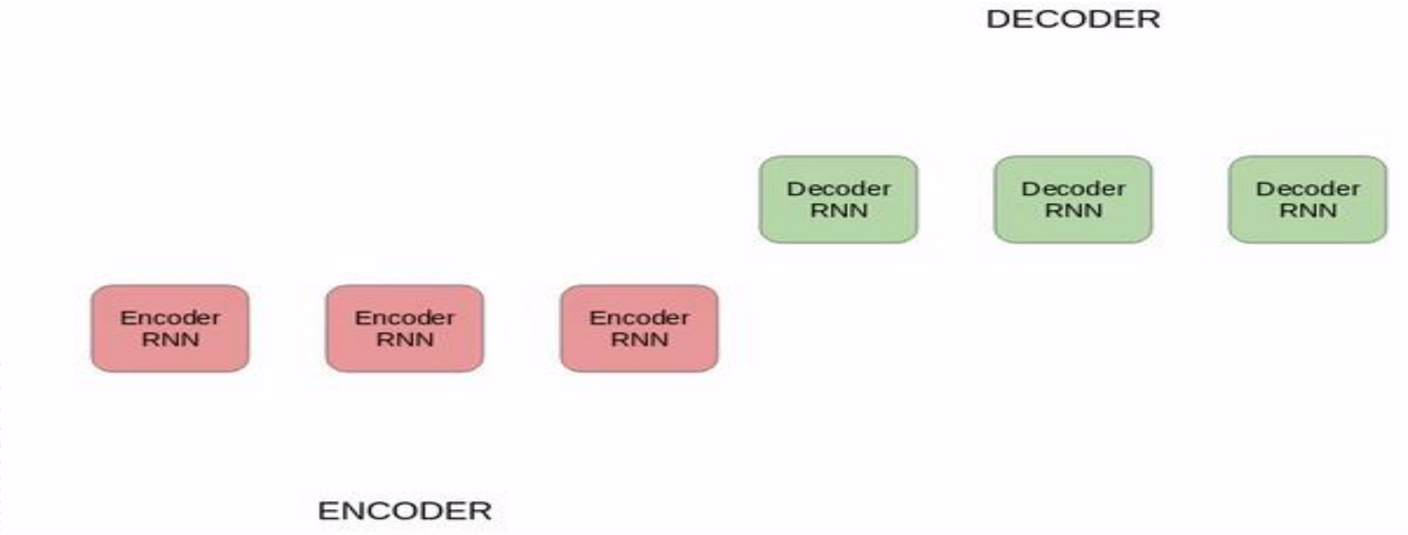
- Recently, they have been applied in standard computer vision tasks achieving state-of-the-art results, e.g., :
  - Image recognition ([DOS2020])
  - Object detection ([CAR2020])
  - Segmentation ([YE2019])

- Over 150 papers were released in 2021.

Introduction

[1/4] ***Transformers vs RNNs***:

- Typically, RNNs (such as LSTMs and GRUs) work in a **sequential** manner, processing one element at a time while keeping a "**memory**" of all the previously seen elements.



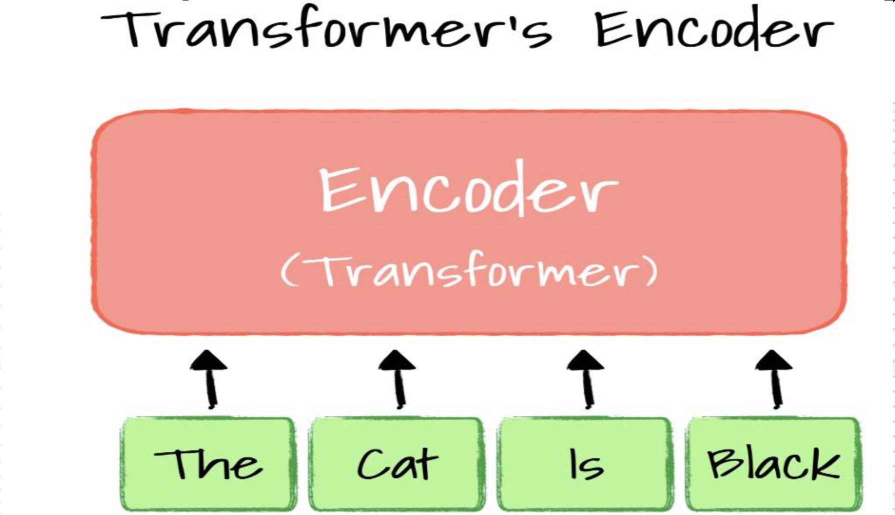
- These models suffer from **exploding gradients** when an input sequence is too long, and dependencies are really distant.
- This sequential nature also makes them **difficult to scale or parallelize**.

Introduction

[2/4] ***Transformers vs RNNs***:

- In Transformers, there is **no concept of time step** regarding the input, hence, they do not require the sequential data be processed in order.

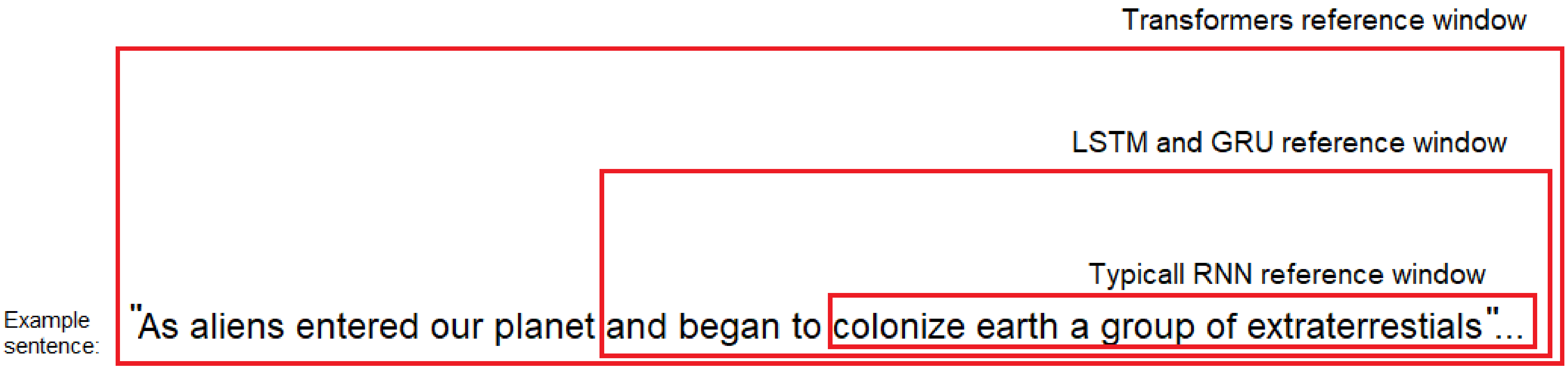
- The entire sequence is processed simultaneously!



- Allows for much more **parallelization** than RNNs and therefore **reduced training times**.

Introduction

[3/4] ***Transformers vs RNNs***:



Hypothetical reference window of RNNs, LSTMs and Transformers.

- Transformers, in theory, have an infinite window to reference from.

Introduction

[4/4] ***Transformers vs RNNs***:

Challenges with **RNNS**:

- Struggles with Long range dependencies
- Gradient explosion
- Large number of training cycles
- Recurrence prevents parallel computation

Transformer Networks:

- Facilitate Long range dependencies
- No gradient explosion
- Fewer number of training cycles
- No recurrence that facilitate parallel computation

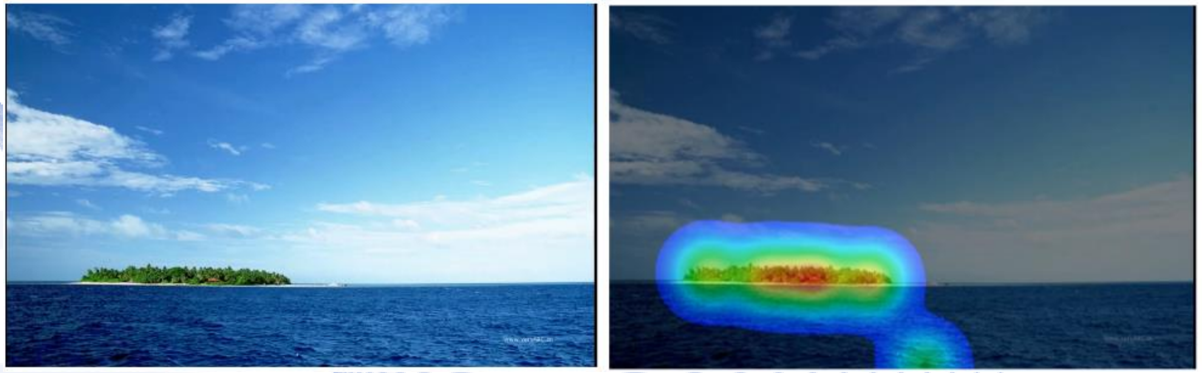
Attention Mechanism in NLP

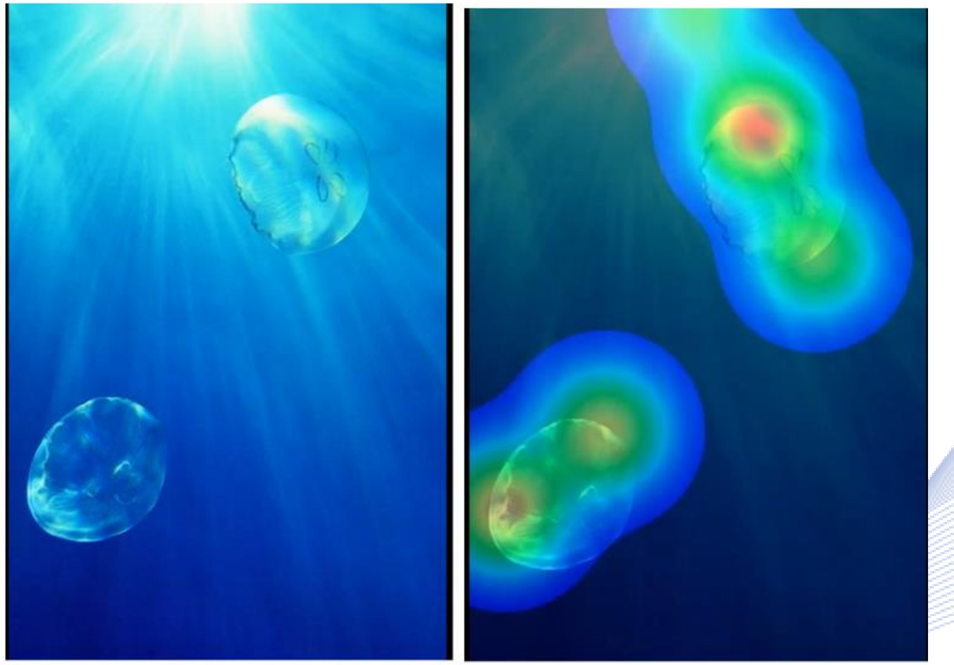
- Attention mechanisms let a model directly **look at**, and **draw from**, the state at **any earlier point in the sequence**.

- Such a mechanism can access all previous states and weight them according to some learned measure of relevancy to the current element, providing sharper information about far-away relevant tokens.

- RNNs combined with attention mechanisms led to large gains in performance.

Attention Mechanism in CV

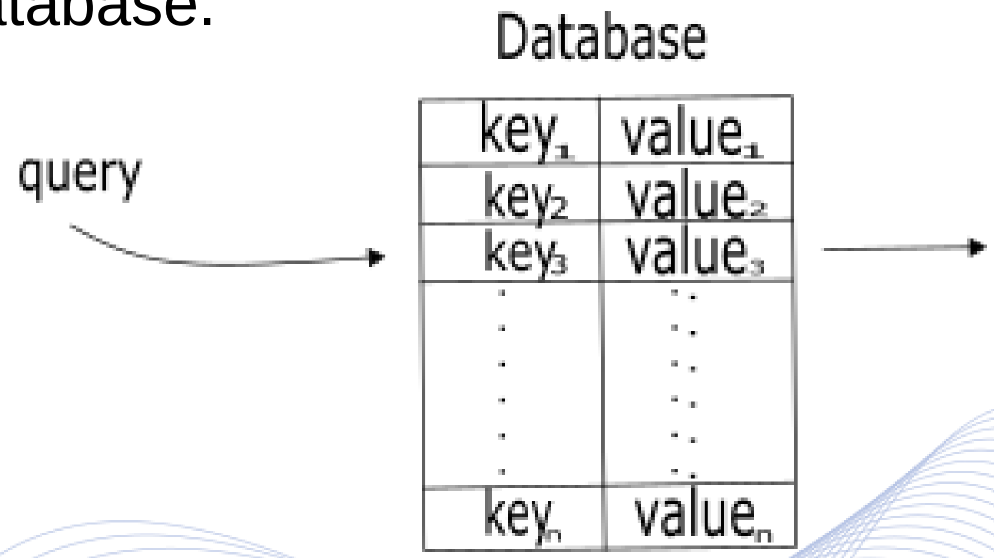




[ITTI1998]

Attention Mechanism

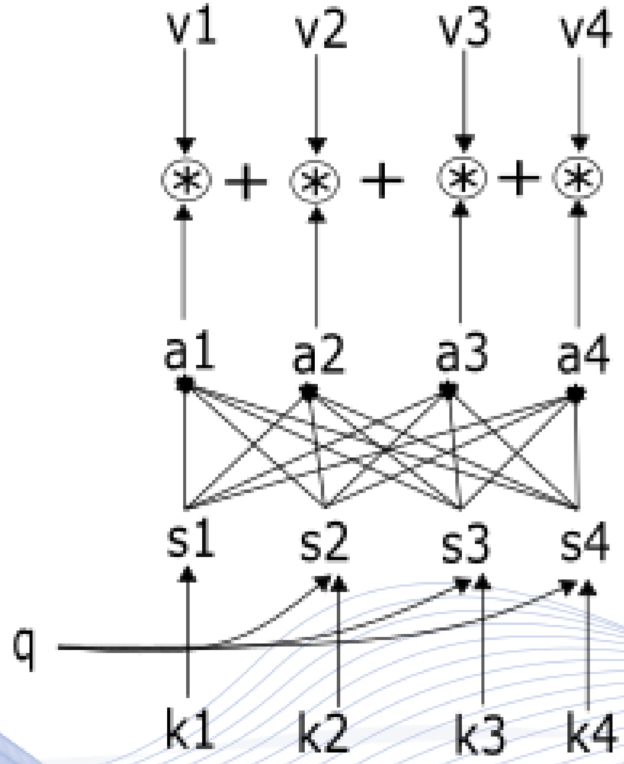
- Mimics the retrieval of a value $\mathbf{v}_i \in \mathbb{R}^d$ for a query $\mathbf{q} \in \mathbb{R}^d$ based on a key $\mathbf{k}_i \in \mathbb{R}^d$ in a database.



- $\text{Attention}(\mathbf{q}, \mathbf{k}_i, \mathbf{v}_i) = \sum_i \phi(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$,

where : $\phi(\mathbf{q}, \mathbf{k}_i) = \begin{cases} \mathbf{q}\mathbf{k}_i^\top & \text{dot product} \\ \frac{\mathbf{q}\mathbf{k}_i^\top}{\sqrt{d}} & \text{scaled dot product} \\ \mathbf{q}\mathbf{W}\mathbf{k}_i^\top & \text{general dot product} \\ \mathbf{q}\mathbf{w}_q^\top + \mathbf{k}_i\mathbf{w}_k^\top & \text{additive similarity.} \end{cases}$

Attention Mechanism



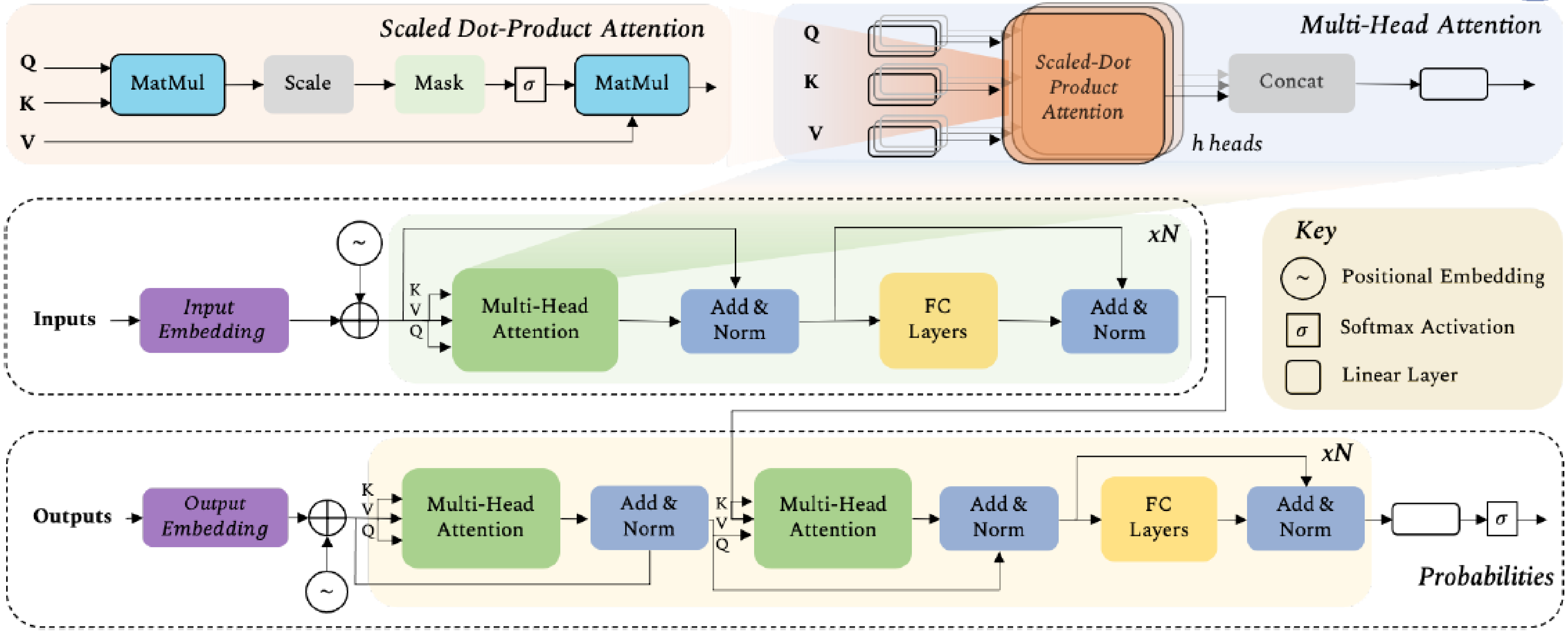
$$\text{Attention} = \sum_i a_i \mathbf{v}_i \, .$$

$$\text{Softmax}(s_j) = a_j = \frac{e^{s_j}}{\sum_i e^{s_j}} \, .$$

- The output is a linear combination of the values $\mathbf{v}_i$ and the “weights” a_i which are generated as a notion of similarity between the query $\mathbf{q}$ and the keys $\mathbf{k}_i$.

Artificial Intelligence & Information Analysis Lab

Transformer architecture



Typical architecture of a Transformer model [KHAN2020].

Scaled dot-product attention

- **Inputs**: $\mathbf{X} \in \mathbb{R}^{n\times d}, \mathbf{Y} \in \mathbb{R}^{m\times d}$

- **Goal**: Enrich the representation of $\mathbf{Y}$ by integrating ("attending") information from $\mathbf{X}$.
- Matrices:
  - Query: $\mathbf{Q} = \mathbf{X}\mathbf{W}_\mathbf{Q}$, $\mathbf{W}_\mathbf{Q} \in \mathbb{R}^{d\times d}$
  - Key: $\mathbf{K} = \mathbf{Y}\mathbf{W}_\mathbf{K}$, $\mathbf{W}_\mathbf{K} \in \mathbb{R}^{d\times d}$
  - Value: $\mathbf{V} = \mathbf{Y}\mathbf{W}_\mathbf{V}$, $\mathbf{W}_\mathbf{V} \in \mathbb{R}^{d\times d}$,

  where $\mathbf{W}_\mathbf{Q}$, $\mathbf{W}_\mathbf{K}$, $\mathbf{W}_\mathbf{V}$ are linear transformations applied on the temporal dimensions of the input sequence.

- Attention: $\mathbf{A} = \underbrace{\mathrm{Softmax}\left(\frac{1}{\sqrt{d}}\mathbf{Q}\mathbf{K}^\top\right)}_{\mathbf{S}}\mathbf{V}.$

Multihead Scaled dot-product attention

- ***Self-attention*** is defined when $\mathbf{X} = \mathbf{Y}$ (common case in the transformer-encoder architecture), where $\mathbf{QK}^T$ is now a square matrix of dimensions $n \times n$.

- In the case of **multi-head attention**, we have N_h number of attention heads and we split the $\mathbf{W_Q}, \mathbf{W_K}, \mathbf{W_V}$ matrices into N_h matrices of dimensions $d \times \frac{d}{N_h}$ (d should be divisible by N_h).

Multihead Scaled dot-product attention

- The attention of every head $\mathbf{A}_h, h = 1, \dots, N_h$ is defined as:

$$\mathbf{A}_h = \underbrace{\mathrm{Softmax}\left(\frac{1}{\sqrt{D}}\mathbf{Q}_h\mathbf{K}_h^\top\right)}_{\mathbf{S}_h}\mathbf{V}_h,$$

where $\mathbf{Q}_h = \mathbf{X}\mathbf{W}_{\mathbf{Q}_h}, \mathbf{K}_h = \mathbf{Y}\mathbf{W}_{\mathbf{K}_h}, \mathbf{V}_h = \mathbf{Y}\mathbf{W}_{\mathbf{V}_h}$ for $h = 1, \dots, N_h$.

- And the overall attention is defined as:

$$\mathbf{A} = \mathrm{Concat}(\mathbf{A}_1, \mathbf{A}_2, \dots, \mathbf{A}_\mathrm{h})\mathbf{W}_\mathrm{O}\ ,$$

where $\mathbf{W}_\mathrm{O} \in \mathbb{R}^{d\times d}$ is a linear projection matrix.

Artificial Intelligence & Information Analysis Lab

Bottlenecks of Transformers

- ***Self-attention*** constitutes a major efficiency bottleneck.

- Memory and time complexity to compute the attention matrix $\mathbf{A}_h$ is quadratic w.r.t the length of the sequence n.

- In particular, the computation of $\boldsymbol{S}_h = \mathrm{Softmax}\left(\frac{1}{\sqrt{d}}\mathbf{Q}\mathbf{K}^\top\right)$ requires multiplying two $n \times \frac{d}{N_h}$ matrices, leading to an overall complexity of $\mathcal{O}(n^2)$.

- Prohibitive to train Transformer models with long sequences (e.g., $n = 2048$).

Efficient Transformers

- Huge surge of proposed efficient Transformer variants [KATH2020, BELT2020, KITAEV2020, WANG2020, XIONG2021]

- Efficiency could refer to reducing either the memory footprint or the computational cost, e.g., number of FLOPS.

- The goal of such models is to propose a way to approximate the quadratic coast of the similarity matrix $\mathbf{S}_h$, by assuming low-rank structure in the $n \times n$ matrix.

Efficient Transformers

- ***Linformer*** [WANG2020] is an efficient transformer model that reduces complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$.

- The similarity matrix $\boldsymbol{S}_h$ can be approximated by a low-rank matrix $\bar{\mathbf{S}}_h$, by introducing two linear projection matrices $\mathbf{E}_h, \mathbf{F}_h \in \mathbb{R}^{k \times n}$ that serve to reduce the dimension of key and value matrices from n to a lower dimension k.

Efficient Transformers

- The new attention is defined as:

$$\mathbf{A}_h = \text{Softmax}\left(\frac{\overbrace{\mathbf{Q}_h \overbrace{(\mathbf{E}_h \mathbf{K}_h)^\top}^{\bar{\mathbf{K}}_h}}^{\bar{\mathbf{S}}_h}}{\sqrt{d}}\right)\left(\overbrace{\mathbf{F}_h \mathbf{V}_h}^{\bar{\mathbf{V}}_h}\right).$$

- The $n \times n$ matrix $\mathbf{S}_h$ has decomposed to the $n \times k$ matrix $\bar{\mathbf{S}}_h$. Hence for small values of $k (k \ll n)$ time and memory consumption are reduced.

References

[VAS2017] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N.Gomez, Ł. Kaiser, and I.Polosukhin, “Attention is all you need,” in Advances in neural information processing systems, pp. 5998–6008, 2017.

[DEV2018] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pretraining of deep bidirectional transformers for language understanding,” arXiv preprint arXiv:1810.04805, 2018.

[LIU 2019] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy,M. Lewis, L. Zettlemoyer, and V. Stoyanov, “Roberta: A robustly optimized bert pretraining approach,” arXiv preprint arXiv:1907.11692, 2019.

[BRO2020] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan,P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell,et al., “Language models are few-shot learners,” arXiv preprint arXiv:2005.14165, 2020.

[DOS2020] L. Dosovitskiy, Al.and Beyer, A. Kolesnikov, D. Weis-senborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Min-derer, G. Heigold, S. Gelly, U. Jakob, and H. Neil, “A nimage is worth 16x16 words: Transformers for image recognition at scale,” arXiv preprint arXiv:2010.11929,2020.

References

[CAR2020] N. Carion, F. Massa, G.l Synnaeve, N. Usunier, A. Kir-illov, and S. Zagoruyko, “End-to-end object detectionwith transformers,”arXiv preprint arXiv:2005.12872,2020.

[YE2019] L. Ye, M. Rochan, Z. Liu, and Y. Wang, “Cross-modal self attention network for referring image segmentation,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019.

[KHAN2020] S. Khan, M. Naseer, M. Hayat, S. Waqas Zamir, F. Shahbaz Khan, and M. Shah, “Transformers in Vision: A Survey,” arXiv:2101.01169, 2021.

[CHOR2020] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J.d Davis, A. Mohiuddin, L. Kaiser, D. Belanger, L. Colwell, and A. Weller, “Rethinking attention with performers,” arXiv preprint arXiv:2009.14794, 2020

[KATH2020] A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret, “Transformers are rnns: Fast autoregressive transformers with linear attention,” in International Conference on Machine Learning (ICML). 2020.

References

[BELT2020] I. Beltagy, M. E. Peters, and A. Cohan, “Longformer: The long-document transformer,” arXiv preprint arXiv:2004.05150, 2020.

[KITAEV2020]. N. Kitaev, L. Kaiser, and A. Levskaya, “Reformer: The efficient transformer,” arXiv preprint arXiv:2001.04451, 2020.

[WANG2020] S. Wang, B. Li, M.n Khabsa, H. Fang, and H. Ma, “Linformer: Self-attention with linear complexity,” arXiv preprint arXiv:2006.04768, 2020.

[XIONG2021] Y. Xiong, Z. Zeng, R. Chakraborty, M. Tan, G. Fung, Y. Li, V. Singh, “Nystromformer: A Nystrom-based Algorithm for Approximating Self-Attention”, arXiv preprint arXiv:2102.03902, 2021.

[SANH2019] V. Sanh, L. Debut, J. Chaumond, T. Wolf, “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”, arXiv preprint arXiv:1910.01108, 2019.

[ITTI1998] L. Itti, C. Koch, E. Niebur, "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis," IEEE Transactions on PAMI, 1998.