Attentions Mechanisms and Transformers

E. A. León-Gómez [1]

ealeon@unal.edu.co[1]
Universidad Nacional de Colombia

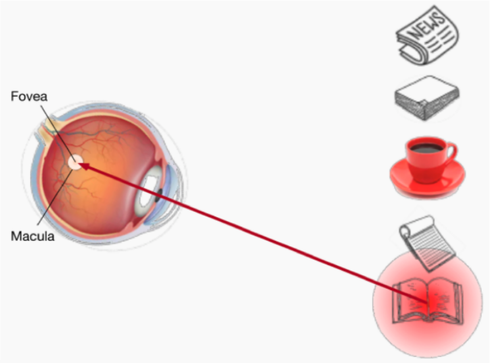
May 30, 2023

Overview

Introduction

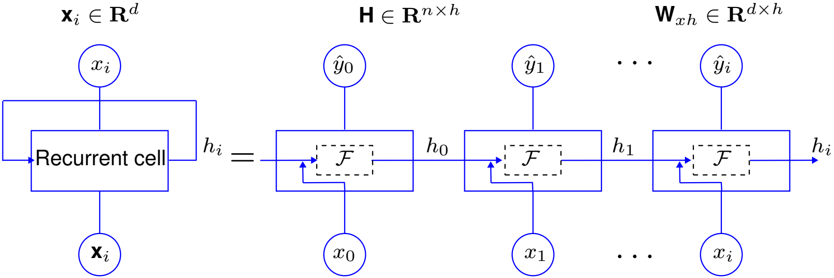
What is Attention?

Attention is a critical mechanism in deep learning that enables models to concentrate on the **most relevant parts** of input data for the given task at hand



Credit:`http://d2l.ai/`

Recurrent Neural Networks (RNN)



$$\hat{y}_i = \mathcal{F}(x_i, h_{i-1}) \tag{1}$$

Problems

- Distant positions in the sequence can be disregarded
- Parallelizing the work is challenging because it processes variables sequentially

Attentions mechanism

Learning task

$$\{\mathbf{x}_i\}_{i=1}^{n} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_d\} \ ; \ \mathbf{x}_i \in \mathcal{R}^d$$

$$\mathbf{X} \in \mathcal{R}^{n \times d}$$

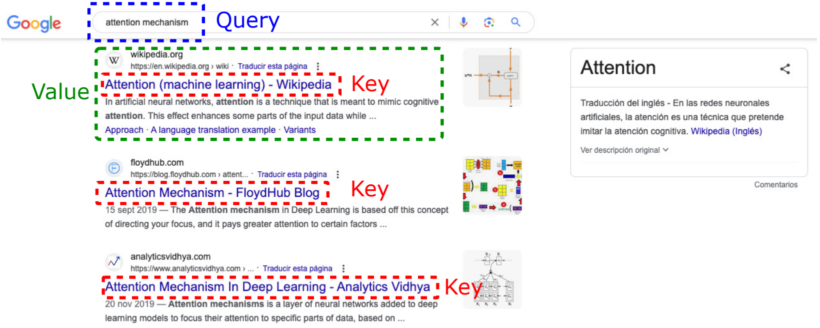
$$\mathbf{h} = \mathbf{x}_1\alpha_1 + \mathbf{x}_2\alpha_2 + \cdots + \mathbf{x}_d\alpha_d \ ; \mathbf{h} \in \mathcal{R}^n$$

$$\mathbf{H} \in \mathcal{R}^{n \times d}$$

Self-attention computes the output sequence **H** from **X** as follows:

- Projection the input into diferrent subspaces
- Computing the output as a weighted average
- Multi-Head Attention

Attention as search

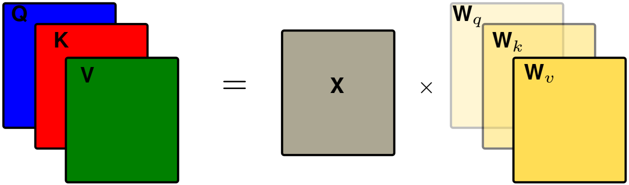


Projection the input into diferrent subspaces

The input **X** is transformed into the query matrix **Q**, the key matrix **K**, and the value matrix **V** via three linear transformations:

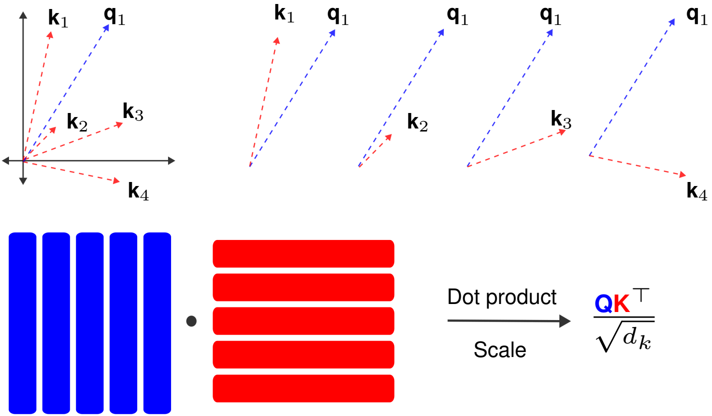
$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q^\top \qquad \mathbf{K} = \mathbf{X}\mathbf{W}_K^\top \qquad \mathbf{V} = \mathbf{X}\mathbf{W}_V{}^\top$$

$$\mathbf{Q} \in \mathcal{R}^{n\times d_k}, \mathbf{W}_Q \in \mathcal{R}^{d\times d_k} \qquad \mathbf{K} \in \mathcal{R}^{n\times d_k}, \mathbf{W}_K \in \mathcal{R}^{d\times d_k} \qquad \mathbf{V} \in \mathcal{R}^{n\times d_v}, \mathbf{W}_V \in \mathcal{R}^{d\times d_v}$$

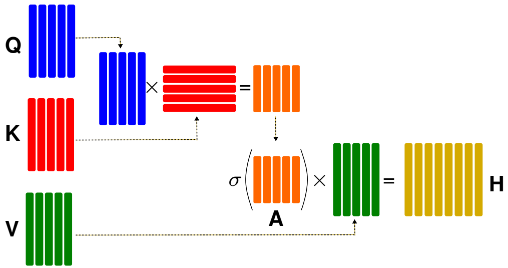


Interpretability of query arrays, keys, and values

Query arrays, keys, and values can be considered an "information retrieval" system



Computing the output as a weighted average



$$\mathbf{H} = \underbrace{softmax\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)}_{\mathbf{A}}\mathbf{V}$$

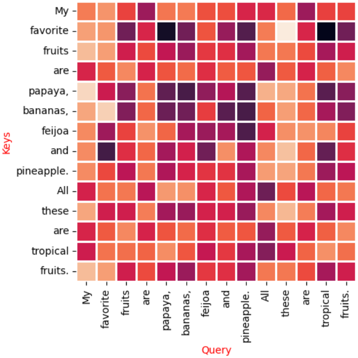
- The self-attention capture the intra-correlation of a given input **X**
- Where $\mathbf{A} \in \mathcal{R}^{n\times n}$ is a probability distribution over the element of **K**

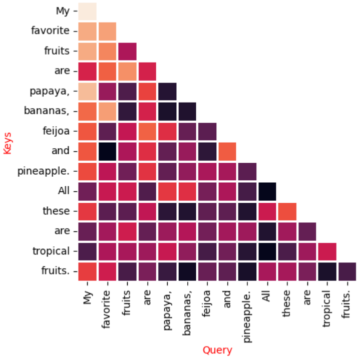
Masked attention

$$\mathbf{H} = softmax\left(\frac{\mathbf{QK}^\top + \mathbf{M}}{\sqrt{d_k}}\right)\mathbf{V}$$

$$\mathbf{M} \in R^{n\times n}\ ; M_{i,j} = \{0\ if\ i \leq j, -\infty if i > j\}$$

$$\mathbf{M} = \begin{bmatrix} 0 & -\infty & -\infty & \dots & -\infty \\ 0 & 0 & -\infty & \dots & -\infty \\ 0 & 0 & 0 & \dots & -\infty \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

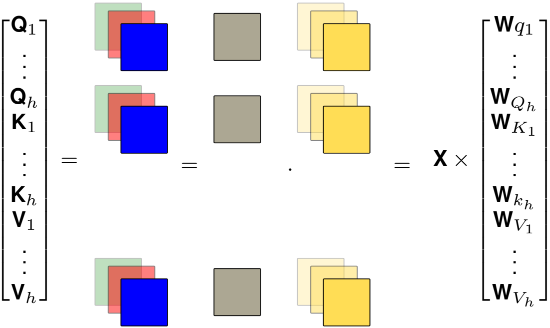




Multi-Head Attention

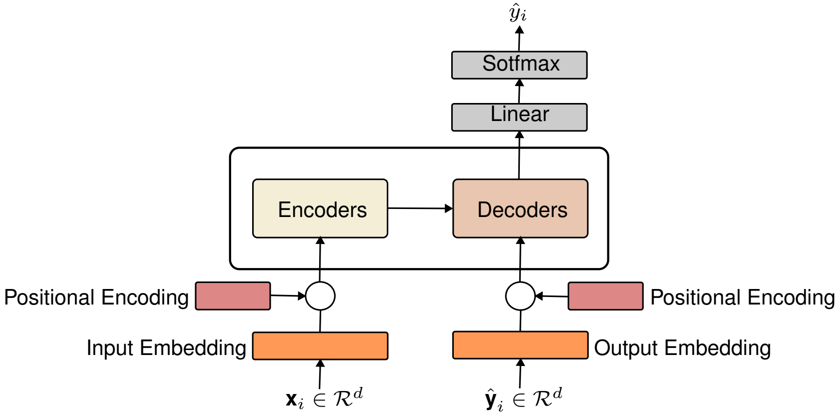
Each output sequence **H** forms an attention head. Multi-head attention concatenates multiple heads to compute the final output.

$$MultiHead\left(\{\mathbf{Q},\mathbf{K},\mathbf{V}\}_{i=1}^{H}\right) = Concat\left(\mathbf{H}_1,\mathbf{H}_2,\ldots,\mathbf{H}_h\right)\mathbf{W}^O \tag{2}$$



Transformers

How deep learning can have attention?



Why?

- Distinct parts of the input convey unique information
- Retain memories of specific, interconnected events from the past

Word Embeddings for NLP

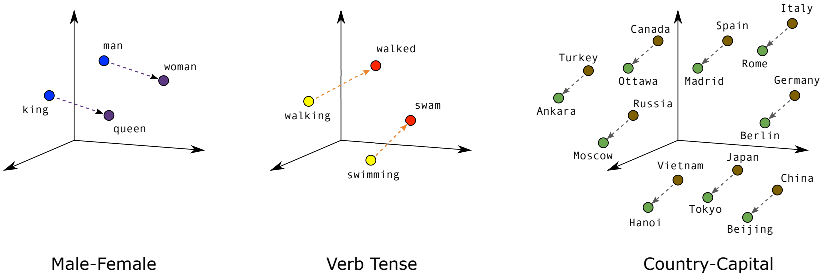


Figure: Embeddings representation [1]

Word embeddings models

- Bag of words(BOW)
- Word2Vec
- GloVe: Global Vector for word representation

1 https://developers.google.com/machine-learning/crash-course/embeddings/translating-to-a-lower-dimensional-space

Intuition

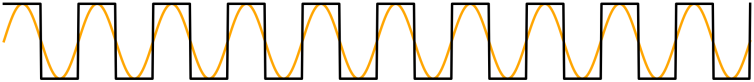
Transformers use wave frequency to find positional information

$\mathbf{X} \in \mathcal{R}^{n \times d}$

Po	Hello,	0.65	0.14	0.23	• • •	0.94	0.81	0.78
P1	how	0.56	0.42	0.73	• • •	0.11	0.32	0.23
P2	are	0.45	0.91	0.82	• • •	0.23	0.99	0.18
P3	you?	0.76	0.66	0.21	• • •	0.98	0.11	0.98

$\mathbf{P} \in \mathcal{R}^{n \times d}$

0	0	0	• • •	0	0	0
0	0	0	• • •	0	1	0
0	0	0	• • •	0	1	1
0	0	0	• • •	1	0	0



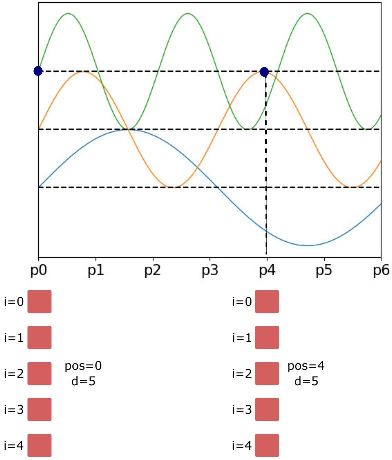
$$\mathbf{p}_{pos,t} = \begin{cases} \sin(w_k t), & \text{if } pos = 2k \\ \cos(w_k t), & \text{if } pos = 2k+1 \end{cases}$$

$$w_k = \frac{pos}{10000^{\frac{2k}{d}}}$$

- w_k:
- pos: Position of input in the sequence
- d: size of input/word/token
- i: Individual dimension of the embedding

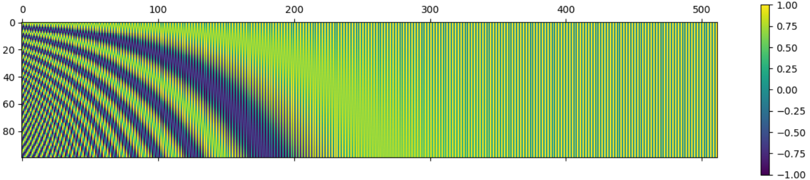
Example

$$\mathbf{p}_t = \begin{bmatrix} \sin(w_1 t) \\ \cos(w_1 t) \\ \\ \sin(w_2 t) \\ \cos(w_2 t) \\ \\ \vdots \\ \\ \sin\left(w_{\frac{d}{2}} t\right) \\ \cos\left(w_{\frac{d}{2}} t\right) \end{bmatrix}_{d \times 1}$$



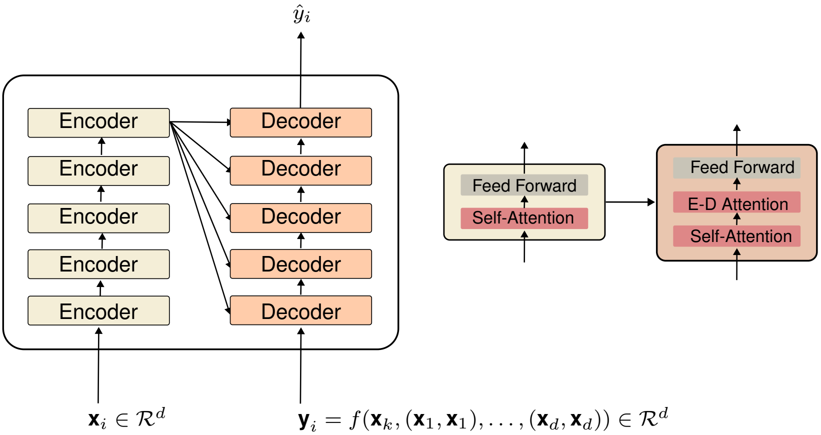
Positional encoding [2]

Positional encoding for a sentence of 100 words and dimension embedding of 500



2 https://github.com/ealeongomez/Machine-Learning/blob/master/Attention-mechanisms/3-Pytorch_PopsitionEncoding.ipynb

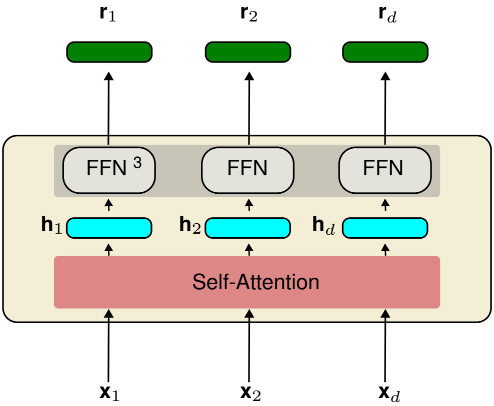
Learning Self-Attention with Neural Network



Features

- The encoders are equal but with different weights
- **E-D Attention**: Allows you to focus on relevant parts of the input

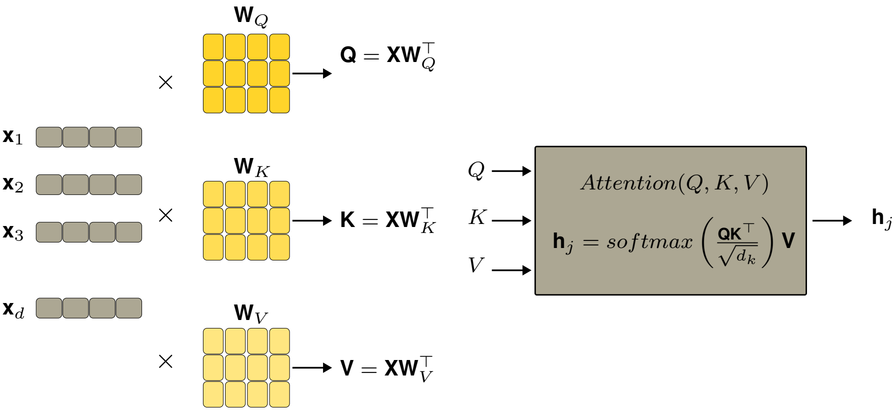
The Encoder Block



[3] Feed-Forward Networks

Self-Attention: Query, keys, and values

The dot-product attention fail to capture the correlations between these features

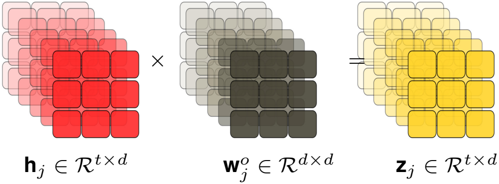


Where $\mathbf{W}_Q, \mathbf{W}_K \in \mathcal{R}^{d_k \times n}$ and $\mathbf{W}_Q \in \mathcal{R}^{d_v \times n}$ are represent learnable weight matrices.

Multi-Head Self-Attention

The multi-head attention Self-Attention consists of multiple attention operators with different similarity function determined by different groups of weight matrices.

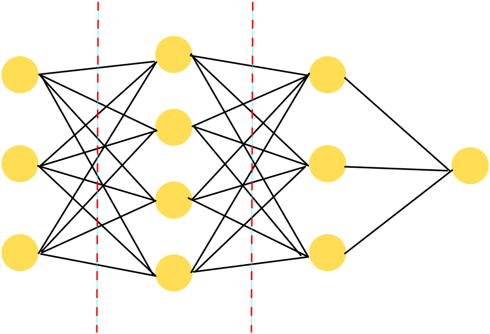
$$\mathbf{z}_j = \mathbf{h}_j \mathbf{w}_j^o \qquad (3)$$



The multi-head attention allows each head to attend different locations based on the similarity in different representation subspaces.

Layer normalization

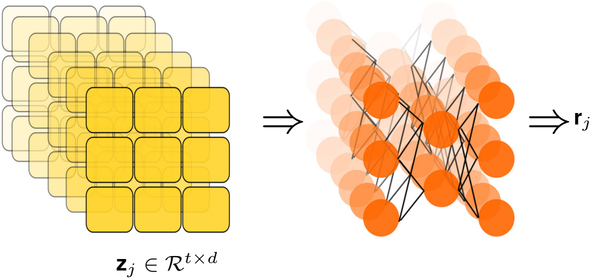
$$\mathbf{z}'_j = LayerNorm(\mathbf{z}_j + \mathbf{x}_j) \qquad \mathbf{x}' = \delta(W_1^\top + b_1)$$



$$\mathbf{z}'_j = \gamma_1 \left(\frac{\mathbf{x} - \mu_1}{\sigma} \right) + \beta_1$$

Full connected network

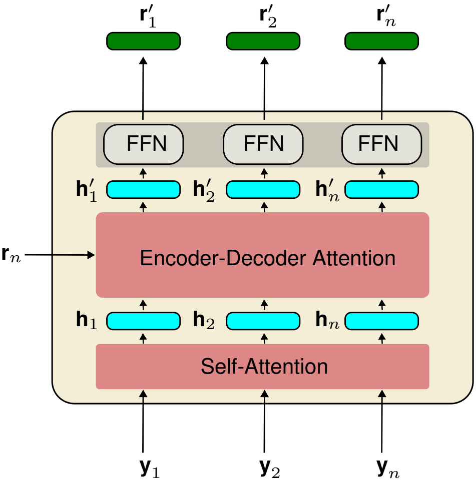
$$FFN(\mathbf{h}_j) = \mathbf{r}_j = ReLU(\mathbf{h}_j\mathbf{W}^1 + \mathbf{b}^1)\mathbf{W}^2 + \mathbf{b}^2$$



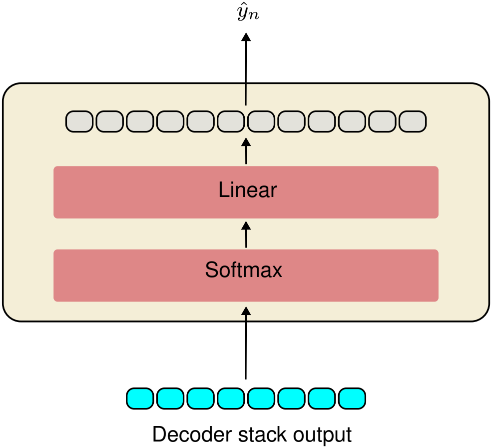
- Where $\mathbf{W}^1$, $\mathbf{W}^2$, $\mathbf{b}^1$ and $\mathbf{b}^2$ are parameters.
- The feed-forward layer is connected back-to-back with residual connections and normalization layers.
- The output of an encoder block is used as an input for the next encoder block.

The Decoder Block

Each decoder block consists of similar layers and operations as the encoder block.

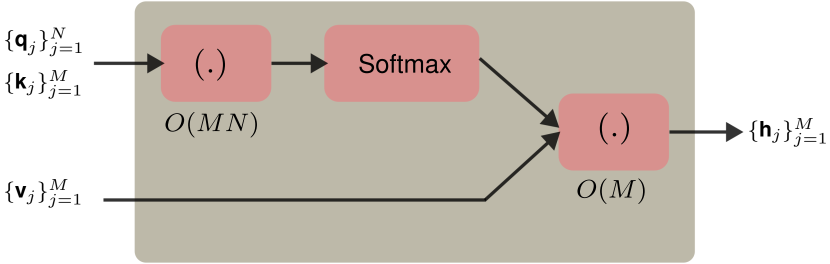


The Output



Random Feature Attention

Complexity computation for softmax attention

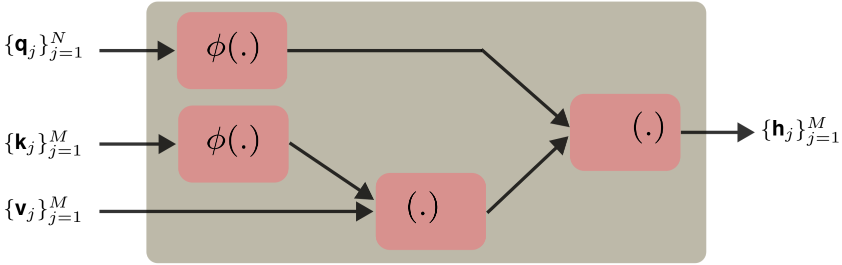


$$\mathbf{h} = \mathbf{D}^{-1} \underbrace{\exp\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)}_{\hat{\mathbf{A}} \in \mathcal{R}^{n\times n}} \mathbf{V} = \mathbf{D}^{-1}\hat{\mathbf{A}}\mathbf{V} \; ; \mathbf{D} = diag\left(\hat{\mathbf{A}}\mathbf{1}\right)$$

Linearized attention

Objetive

$$\exp\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right) \approx \phi(\mathbf{Q})\phi(\mathbf{K})^\top$$



Attention in kernel machine I

Theorem: Random Fourier Features [4]

Let $\phi : \mathcal{R}^d \to \mathcal{R}^{2D}$ be transformation:

$$\phi(\mathbf{x}) = \sqrt{\frac{1}{D}}\left[\sin(\mathbf{w}_1\mathbf{x}), \ldots, \sin(\mathbf{w}_D\mathbf{x}), \ldots, \cos(\mathbf{w}_1\mathbf{x}), \ldots, \cos(\mathbf{w}_1\mathbf{x})\right]^\top$$

When d-dimensional random vectors $\mathbf{w}_i$ are independently sampled from $\mathcal{N}(0, \sigma^2\mathbf{I}_d)$

$$\mathbf{h} = softmax\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V};\ softmax(\mathbf{a}) = \frac{\exp(\mathbf{a}_i)}{\sum_k \exp(\mathbf{a}_k)}$$

$$\mathbf{h} = \frac{\exp\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)}{\sum_{j=1}^{n} \exp\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)}\mathbf{V}$$

[4] https://proceedings.neurips.cc/paper/2007/hash/013a006f03dbc5392effeb8f18fda755-Abstract.html

Attention in kernel machine II

$$\mathbf{D} = \sum_{j=1}^{n} \underbrace{\exp\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)}_{\hat{\mathbf{A}}} = \sum_{j=1}^{n} \hat{\mathbf{A}} = diag(\hat{\mathbf{A}}\mathbf{1}^\top)$$

$$\mathbf{h} = \frac{\exp\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right)}{\mathbf{D}}\mathbf{V}$$

$$\exp\left(\frac{\mathbf{QK}^\top}{\sqrt{d_k}}\right) \approx \phi(\mathbf{Q})\phi^\top(\mathbf{K})$$

$$\boxed{\mathbf{h} = \mathbf{D}^{-1}\phi(\mathbf{Q})\phi^\top(\mathbf{K})\mathbf{V}}$$

Illustrated transformer/
https://jalammar.github.io/illustrated-transformer/.

Rahimi, Ali and Recht, Benjamin
Attention is All You Need
Advances in neural information processing systems (2007).

Vuckovic, James and Baratin, Aristide and Combes, Remi Tachet des
A mathematical theory of attention
arXiv preprint arXiv (2020).

Ji, Shuiwang and Xie, Yaochen and Gao, Hongyang
A mathematical view of attention models in deep learning
Texas A&M University: College Station, TX, USA (2019).

Nguyen, Tan and Pham, Minh and Nguyen, Tam and Nguyen, Khai and Osher, Stanley J and Ho, Nhat
Transformer with Fourier Integral Attentions
rXiv preprint arXiv (2022).

AAhmed, Sabeen and Nielsen, Ian E and Tripathi, Aakash and Siddiqui, Shamoon and Rasool, Ghulam and Ramachandran, Ravi P
Transformers in time-series analysis: a tutorial
rXiv preprint arXiv (2022).

Rahimi, Ali and Recht, Benjamin
Random features for large-scale kernel machines
Advances in neural information processing systems.

Transformers

- Self-attention https://github.com/ealeongomez/Machine-Learning/blob/master/Attention-mechanisms/1_selfAttention.ipynb
- Multi Head Attention: https://github.com/ealeongomez/Machine-Learning/blob/master/Attention-mechanisms/2-TensorFlow_MultiHeadAttention.ipynb

Thank you!