Cursos Extraordinarios

Verano 2024

“Inteligencia Artificial y Grandes Modelos de Lenguaje: Funcionamiento, Componentes Clave y Aplicaciones”

Zaragoza, del 3 al 5 de julio

Transformer

- **Arquitectura**
  - Residual layers
  - Layer normalization
  - Multi-head attention
  - Feed forward
  - Positional embedding
  - Embedding de entrada
  - Arquitectura general

Transformer

- **Attention is all you need (**Vaswani **2017)** > 100k cites

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N. Kaiser L, Polosukhin, I. (2017 Attention is all you need. Advances in neural information processing systems, 30, 5998-6008..

- A new type of sequence models
- The basis model for most state of the art results nowadays:
  - Translation
  - Language modeling
  - NLP: question answering, summary
  - Language understanding,
  - Speech recognition (ASR), synthesis
  - Image recognition, segmentation, detection
  - Multimodal image+text
  - DNA folding
  - Drug discovery
  - …

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

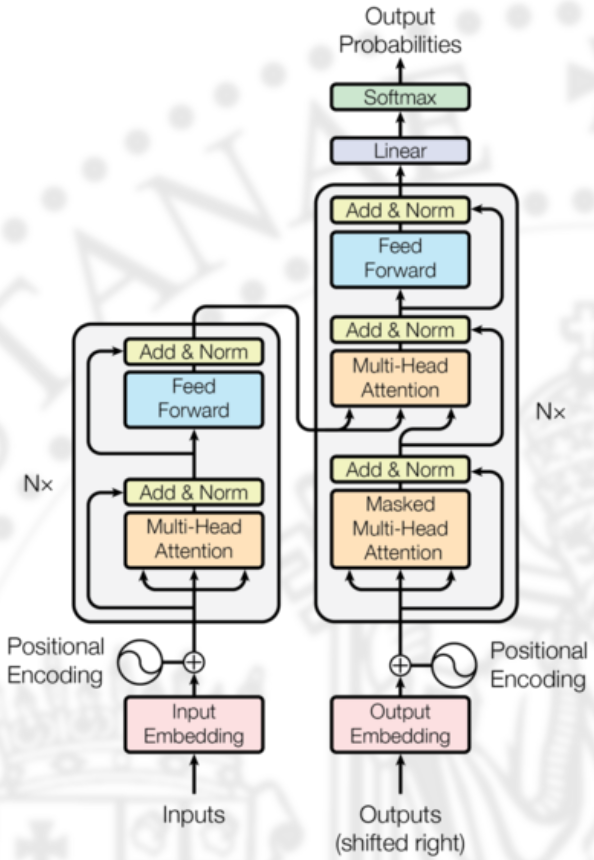
Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Transformer

Transformer

- Architecture:
  - **Encoder / Decoder**
  - Similar to **seq2seq** architecture
  - Some application only use Encoder
- Encoder creates a processed representation for the input sentence
- Decoder predicts next symbol/word given previously generated
- No LSTMs or convolutions (origin of the title)
- **Self attention** (multihead) is the main computational unit:



Transformer

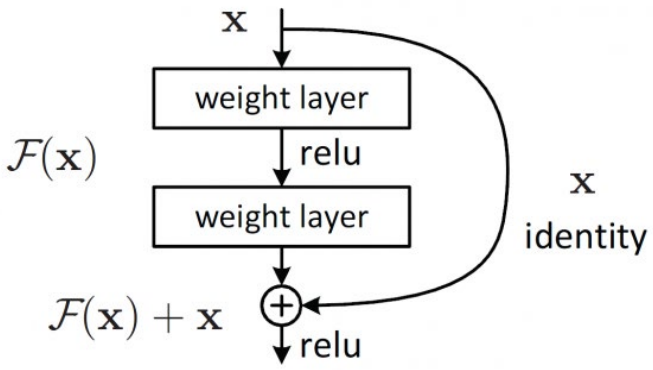
- **Arquitectura**
  - **Residual layers**
  - Layer normalization
  - Multi-head attention
  - Feed forward
  - Positional embedding
  - Arquitectura general

Residual layers

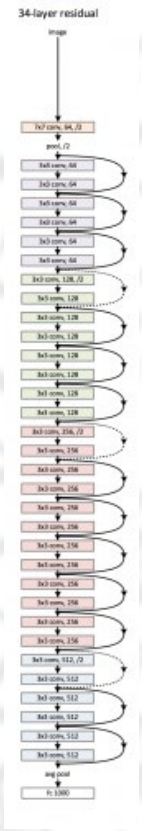
- **Residual layers (121k citas)**

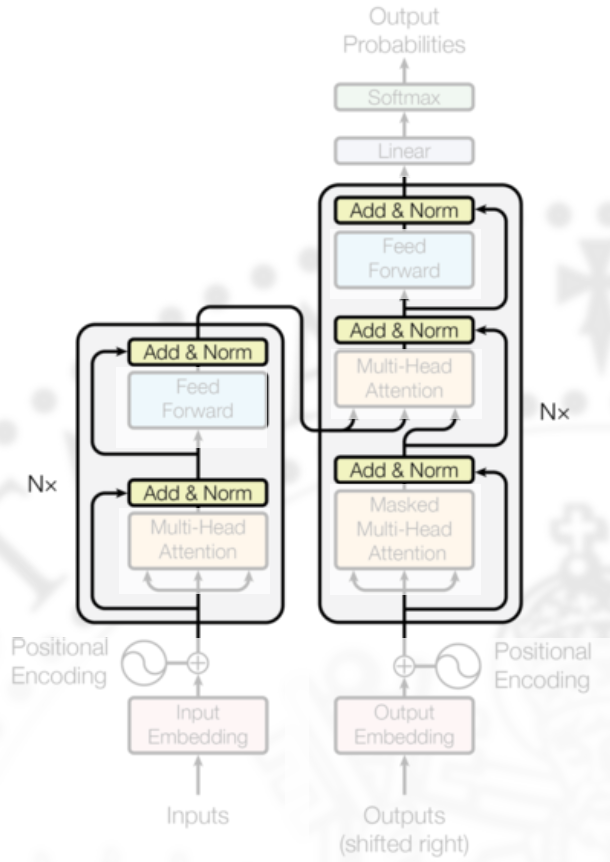
He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

- The core idea is to modify the representation incrementally



- The vanishing gradient problem prevented deeper networks:
  - In the paper they present networks from 28 to 152 layers





Transformer

- **Arquitectura**
  - Residual layers
  - **Layer normalization**
  - Multi-head attention
  - Feed forward
  - Positional embedding
  - Embedding de entrada
  - Arquitectura general

Layer normalization

- **Normalization**
  - **Layer normalization (**Ba **2016)**

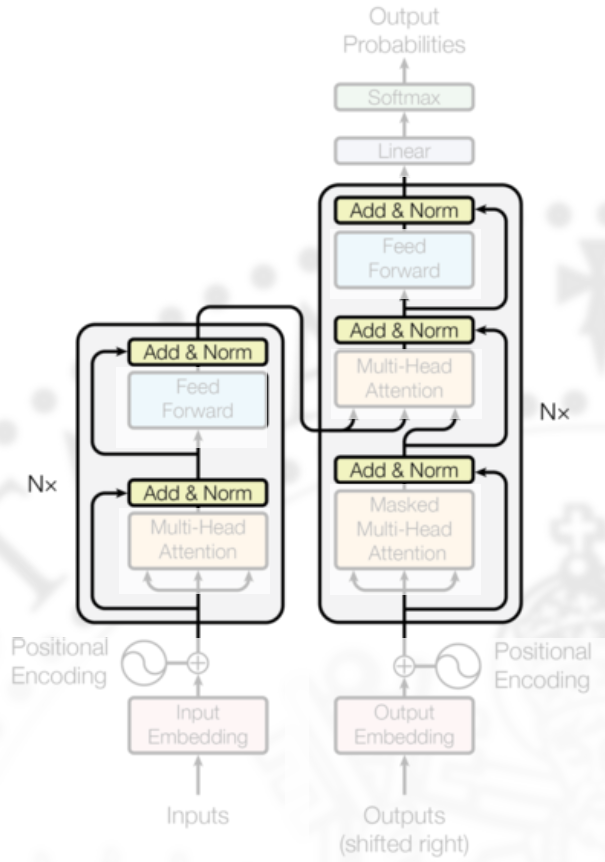
Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. arXiv preprint arXiv:1607.06450.

  - **Instance Normalization(**Wu **2016)**

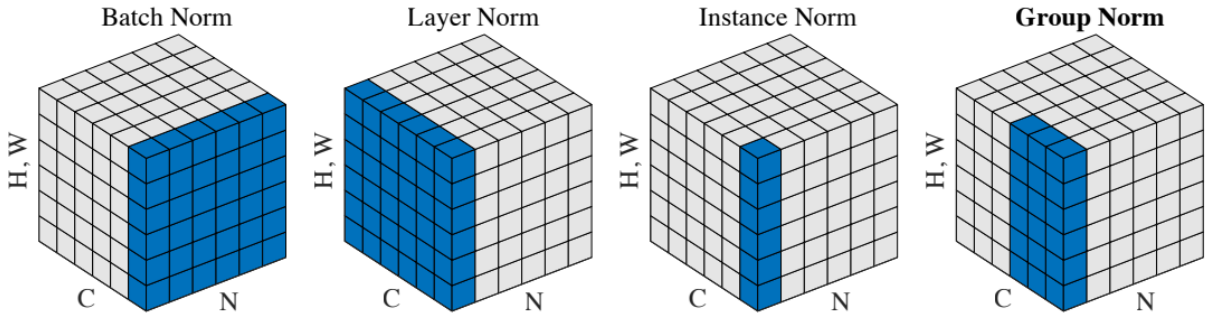
Ulyanov, D., Vedaldi, A., & Lempitsky, V. (2016). Instance normalization: The missing ingredient for fast stylization. arXiv preprint arXiv:1607.08022.

  - **Group Normalization(**Wu **2018)**

Wu, Y., & He, K. (2018). Group normalization. In Proceedings of the European conference on computer vision (ECCV) (pp. 3-19).



- **Layer normalization**
  - The layer normalization is an alternative to the batch normalization
  - The objective and mechanism is similar to BN
  - The mean and std is calculated over the signal **channel dimensions** instead



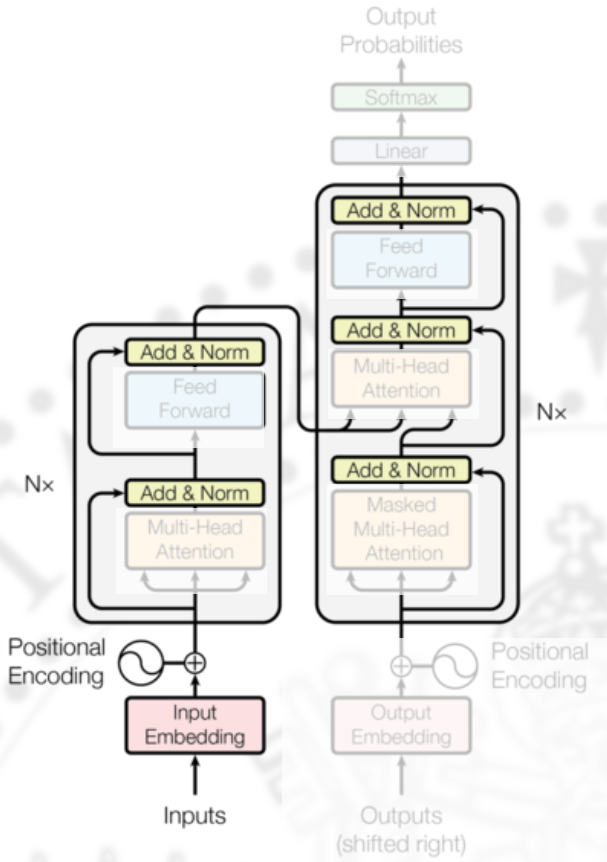
Layer normalization

- **Layer normalization**

- **Layer normalization**
- The mean and std is calculated over the signal **channel dimensions** instead of minibatch

- Steps
  - 1. **Normalize** signal channel dimension
  - 2. Aplly **linear transformation** with learned parameters

- Special case: RMSNorm, no bias is taken into account




```python
class RMSNorm(torch.nn.Module):
    def __init__(self, d_model=512, eps=1e-8, **kwargs):
        super().__init__()
        self.eps = eps
        self.scale = torch.nn.Parameter(torch.ones(1, 1, d_model))

    def forward(self, x):
        x = x / (torch.sqrt(torch.mean(x**2, dim=-1, keepdim=True)) + self.eps)
        return x * self.scale
```

Transformer

- **Arquitectura**
  - Residual layers
  - Layer normalization
  - **Multi-head attention**
  - Feed forward
  - Positional embedding
  - Embedding de entrada
  - Arquitectura general

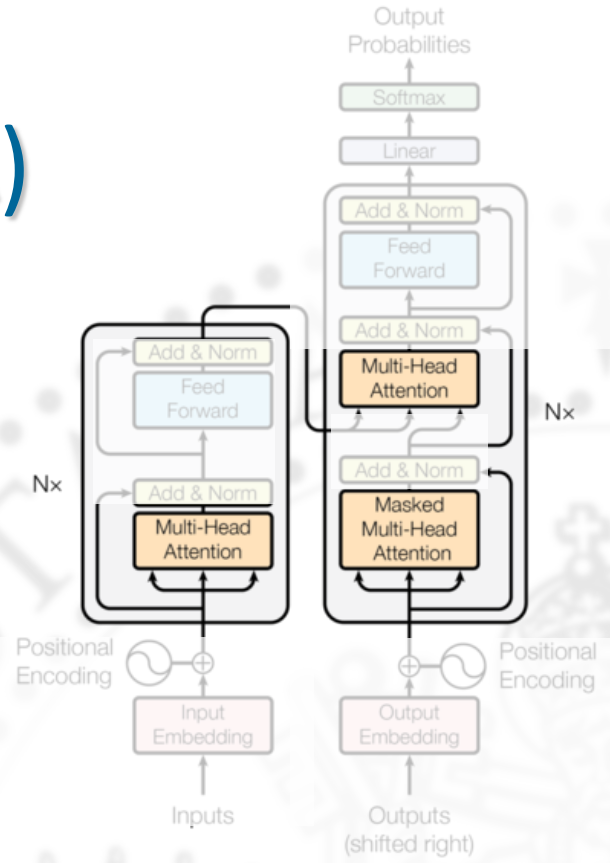
Multi-head Attention (MHA)

- **Multi-head Attention**
- MHA mechanism
  - Attention mechanism
    - Self attention
      - Input: sequence size (n x d)
      - Output: sequence size (n x d)
    - Cross attention
      - Input: sequence size (n x d)
        external signal (m x d)
      - Output: sequence size (n x d)

    Output is a linear combination of inputs

    Similar to a convolution (dynamic filter)

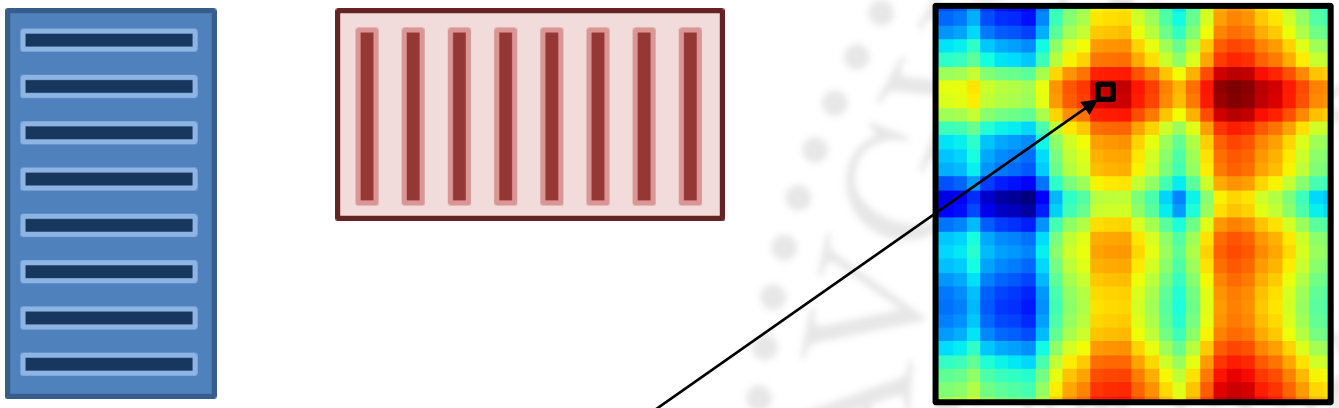
    The decoder has a causal restriction
  - Heads -> h (from 1 to 96)
    - parallel Attention modules
    - Operation steps:
      - **Split** the vector in h parts
      - **Apply** Attention to each part
      - **Concatenate**



Multi-head Attention (MHA)

- **Self attention**
- The core of the transformer is the self-attention operation
  - Given two sequences of vectors, k, q, of length T and dimension D
  - The product of the two matrices is used to find **similarities** between the sequences
    - In signal processing, similar to the idea of **Cross-correlation**

$$\underset{(T \times D)}{\boldsymbol{q}} \cdot \underset{(D \times T)}{\boldsymbol{k}^T} = \boldsymbol{C} \quad \text{(CrossSimilarity)}$$



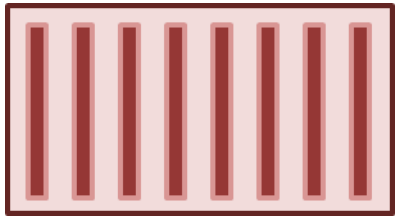
Element C_{ij}

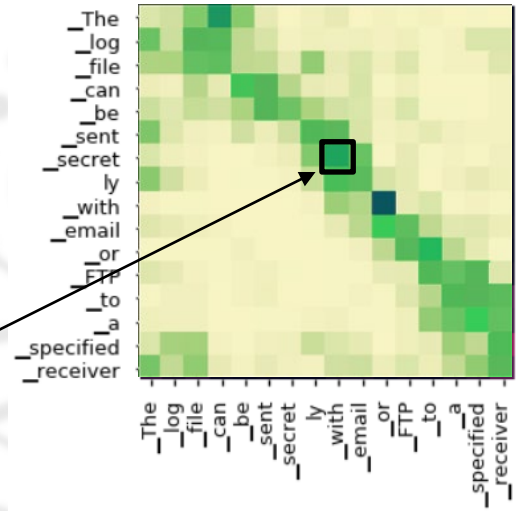
similarity, vectors: $\boldsymbol{k}_i \cdot \boldsymbol{q}_j^T$

Multi-head Attention (MHA)

- **Self attention**
- The core of the transformer is the self-attention operation
  - Given two sequences of vectors, k, q, of length T and dimension D
  - The product of the two matrices is used to find similarities between the sequences
  - A softmax operation is used to produce **mappings** of input frames to outputs (alignment matrix)

$$softmax(\underset{(T \times D)}{\boldsymbol{q}} \cdot \underset{(D \times T)}{\boldsymbol{k}^T}) = \boldsymbol{A} \quad \text{(Self attention)}$$





$$\sum_j A_{ij} = 1$$
(sum 1 cols.)

Element A_{ij}

Degree of influence of input sequence item j over output i

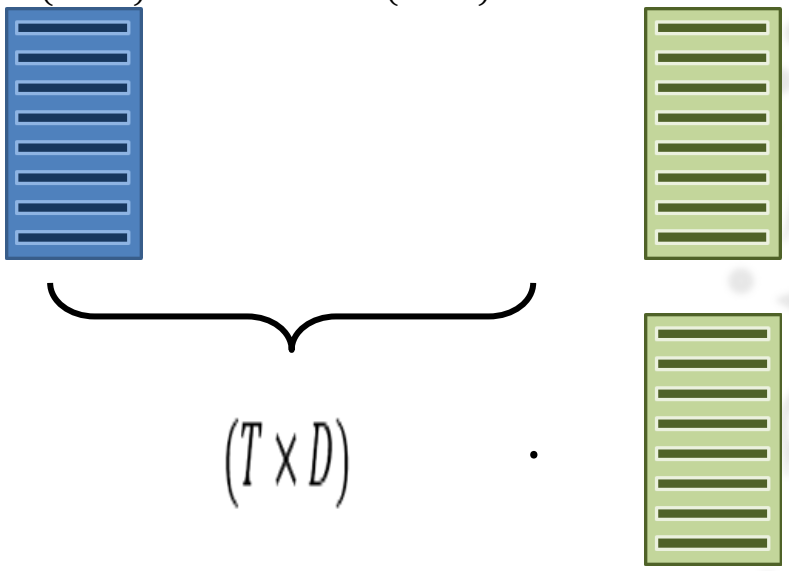
https://nlp.seas.harvard.edu/2018/04/03/attention.html

Multi-head Attention (MHA)

Self attention

- The core of the transformer is the self-attention operation
  - Given two sequences of vectors, $\boldsymbol{k}, \boldsymbol{q}$, of length T and dimension D
  - The product of the two matrices is used to find similarities between the sequences
  - A softmax operation is used to produce **mappings** of input frames to outputs (alignment matrix)
  - The mapping is used to produce the output of the layer after multiplying by the values matrix
    - The process can be summarized as: the application of a dynamical mapping to the matrix $\boldsymbol{v}$

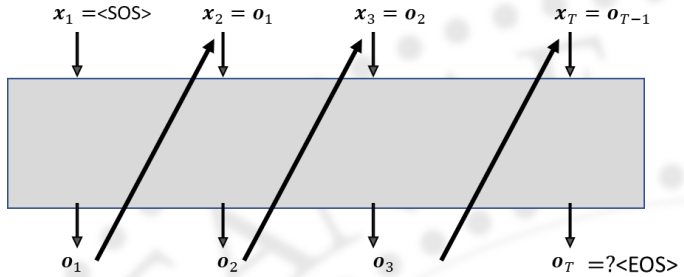
$$softmax(\underset{(T \times D)}{\boldsymbol{q}} \cdot \underset{(D \times T)}{\boldsymbol{k}^T}) \cdot \boldsymbol{v} = \boldsymbol{o} \quad \text{(output)}$$



- The output sequence can be **reordered**
- Each output item can be dependent of **any part of the input** sequence

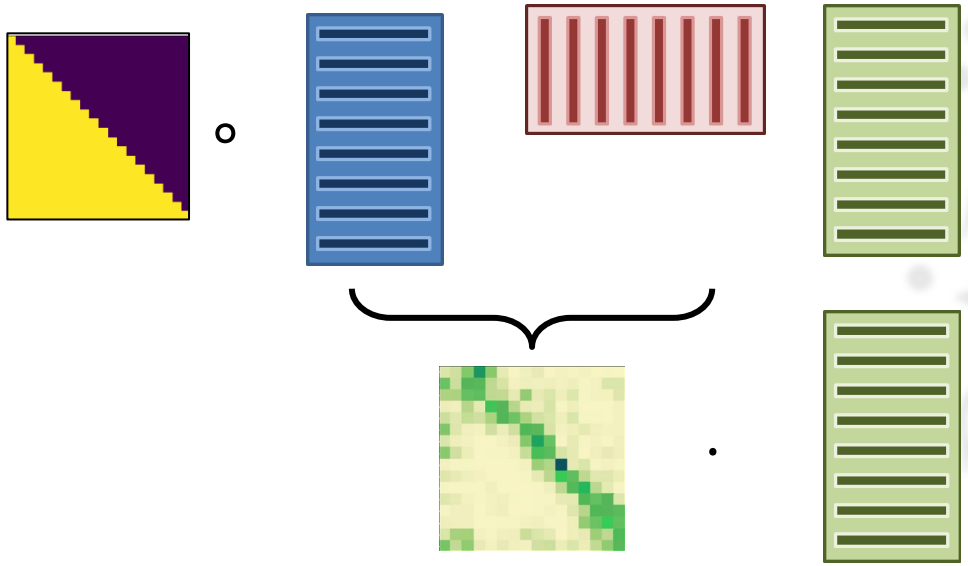
- The final output is the mapping/alignment multiplied by the sequence
- The mapping/alignment process can be interpreted as an **attention mechanism**

Multi-head Attention (MHA)



- **Causal self attention**
- To operate in generators of seq2seq architectures
  - The objective is to predict next item given previous
  - The problem if the model is not modified is that self attention allows a trivial shortcut from future token present in the intput to the Desired output
  - This is solved by multiplying the attention alignment with a mask matrix (upper diagonal values = 0)

$$softmax(M \circ (\underset{(T \times D)}{\boldsymbol{q}} \cdot \underset{(D \times T)}{\boldsymbol{k}^T})) . \boldsymbol{v} = \boldsymbol{o} \quad \text{(output)}$$



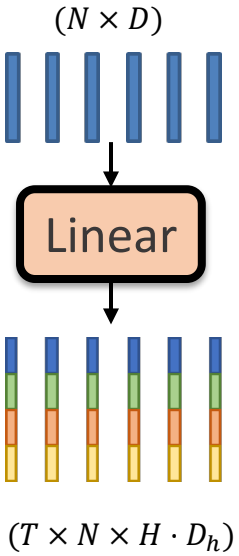
Multi-head Attention (MHA)

- **Multi-Head attention**
  - The representation dimension: split in h subvectors: $\boldsymbol{k}_h, \boldsymbol{q}_h, \boldsymbol{v}_h$
  - The self-attention is then used with each subvector

$$Attention(\boldsymbol{q}_h, \boldsymbol{k}_h, \boldsymbol{v}_h) = softmax\left(\frac{\boldsymbol{q}_h \cdot \boldsymbol{k}_h^T}{scale}\right) \cdot \boldsymbol{v}_h$$

scale constant (scale = 1 / $\sqrt{D_h}$) plays the role of temperature

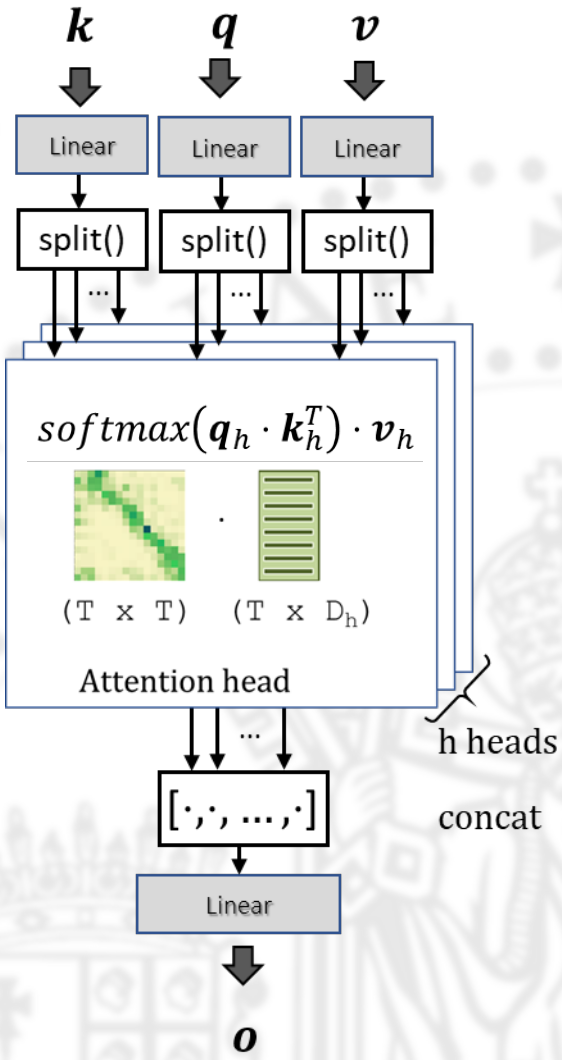
  - The resulting sequences are **concatenated**
  - Finally a linear layer is applied to obtain the output




```
self.linear = nn.Linear(d, h*dh)
```

```
q = self.linear(q).reshape(b, n, h, dh)
```

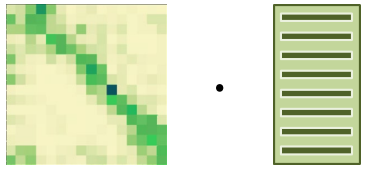
split the ouput in **h subvectors** of dimension dh

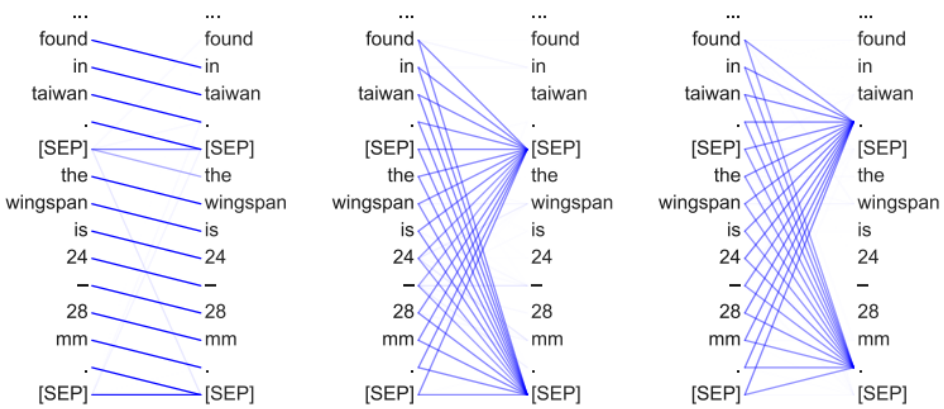


Multi-head Attention (MHA)

- **Self attention**
- There have been many studies trying to analyze the meaning of the mappings
  - One of the classic problems of NLP has been to determine automatically the relationship of a pronoun with previous text
    - e. g. ***they*** in the example on the left

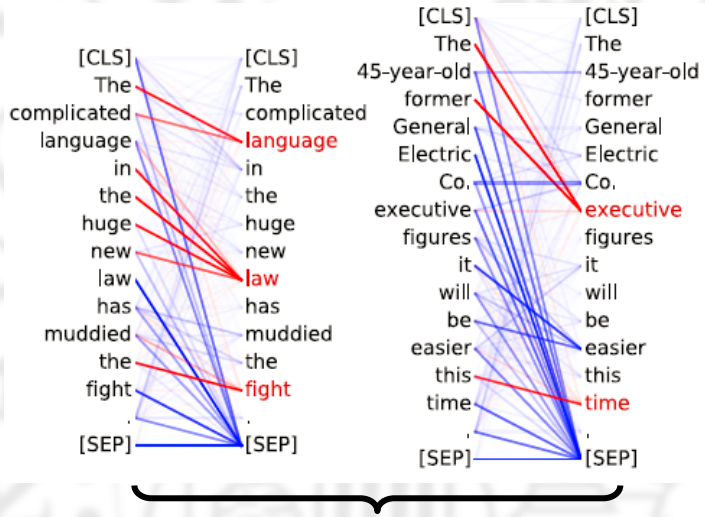
$$softmax(\boldsymbol{q} \cdot \boldsymbol{k}^T) \cdot \boldsymbol{v}$$





Attention to previous item

Attention to end of sentence

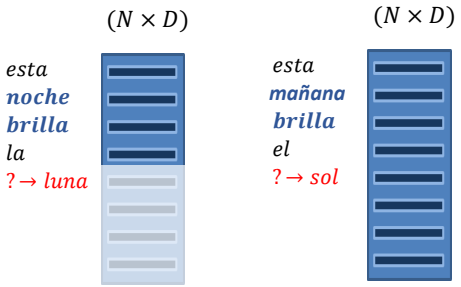


Noun modifiers, e.g. determiners attend to their noun

https://medium.com/synapse-dev/understanding-bert-transformer-attention-isnt-all-you-need-5839ebd396db
https://medium.com/dair-ai/aspects-of-language-captured-by-bert-32bc3c54016f

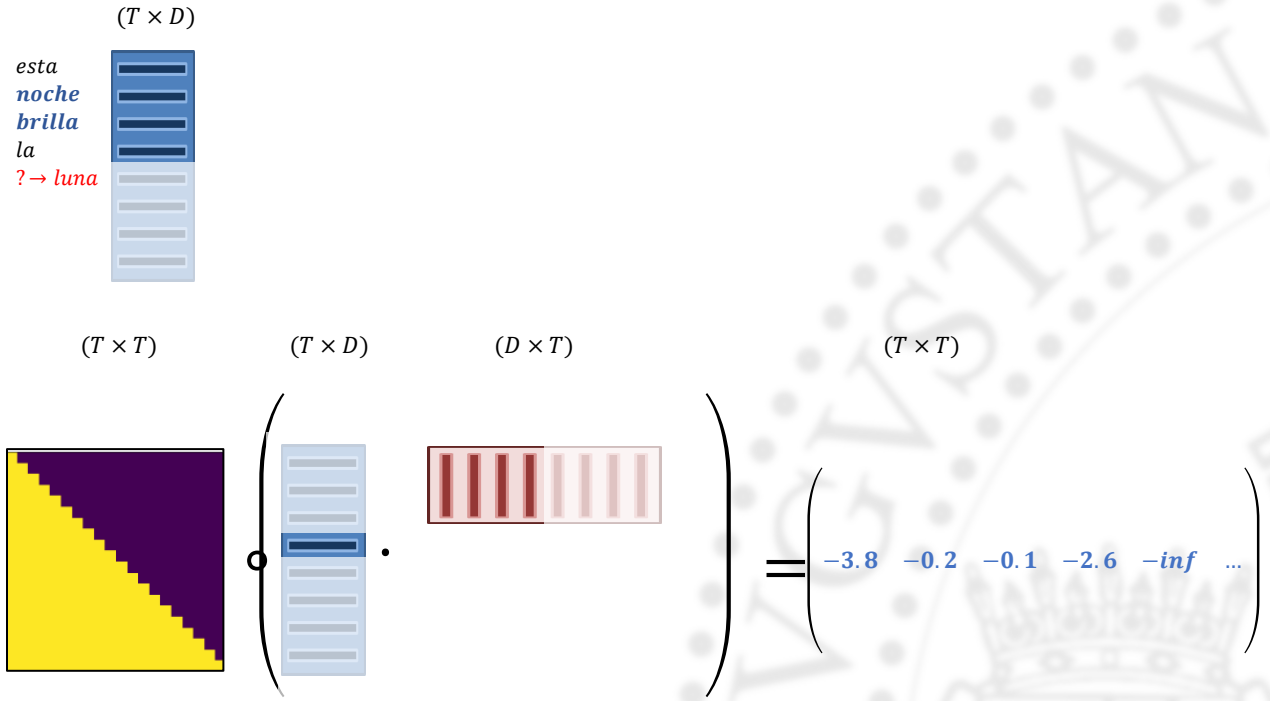
Multi-head Attention (MHA)

- **Self attention: example**



Multi-head Attention (MHA)

- **Self attention: example**



Multi-head Attention (MHA)

- **Self attention: example**

$(N \times D)$ $(N \times D)$

$$softmax\left(\begin{array}{ccccc} -2.8 & -0.2 & -0.1 & -2.6 & -inf \quad \ldots \end{array}\right) \cdot$$

$=$

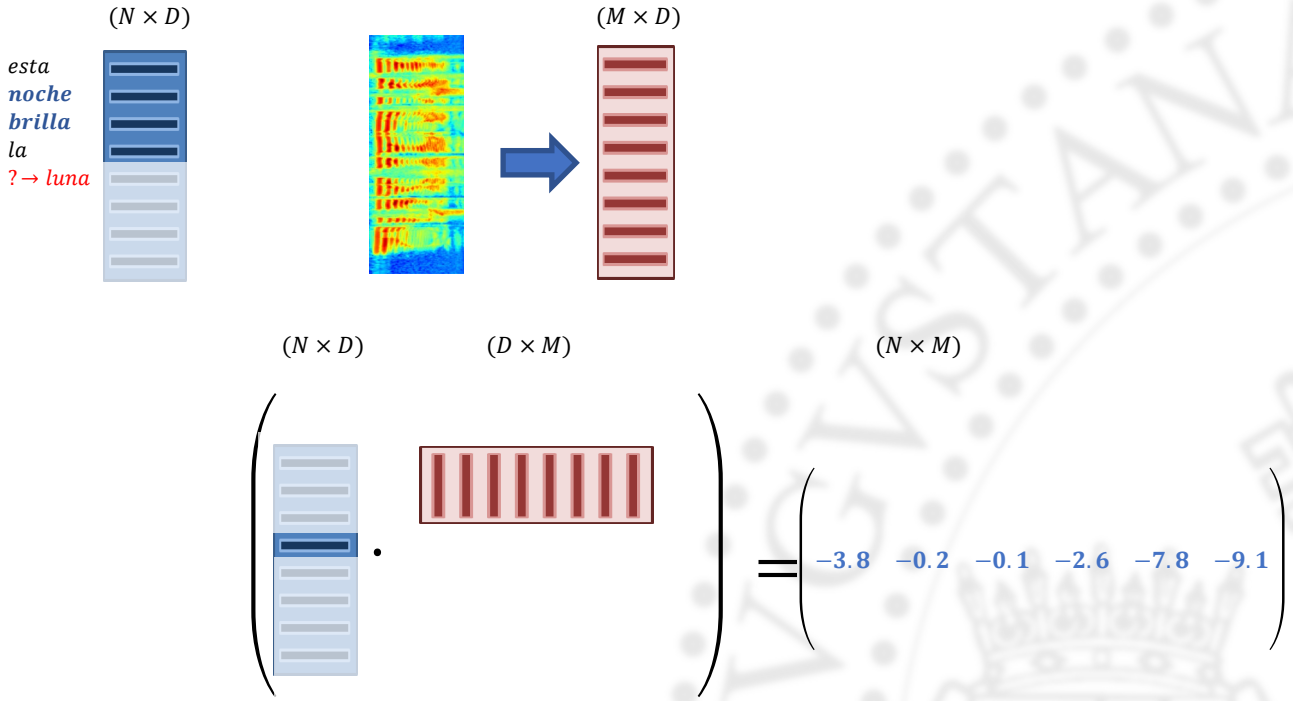
$(N \times N)$ $(N \times D)$ $(N \times D)$

$$\left(\begin{array}{ccccc} 0.03 & \mathbf{0.45} & \mathbf{0.48} & 0.04 & 0 \quad \ldots \end{array}\right) \cdot$$

$=$

Multi-head Attention (MHA)

- **Cross attention: example**



Multi-head Attention (MHA)

- **Cross attention: example**