

Introducción a Transformers

Aprendizaje automático

Docente: Juan David Martínez Vargas

jdmartinev@eafit.edu.co

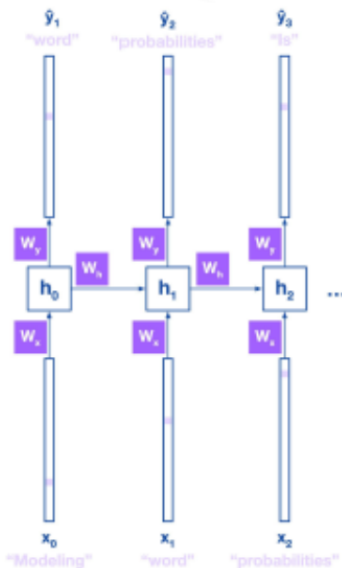
2023

Redes recurrentes

Weights are shared
over time steps

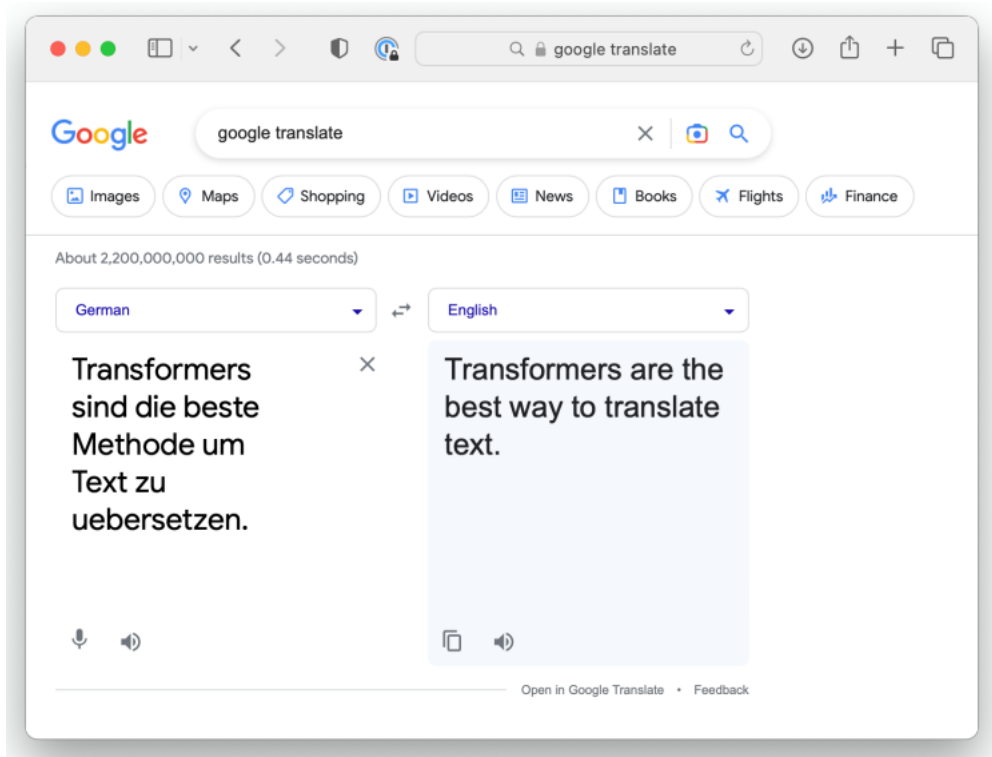


RNN

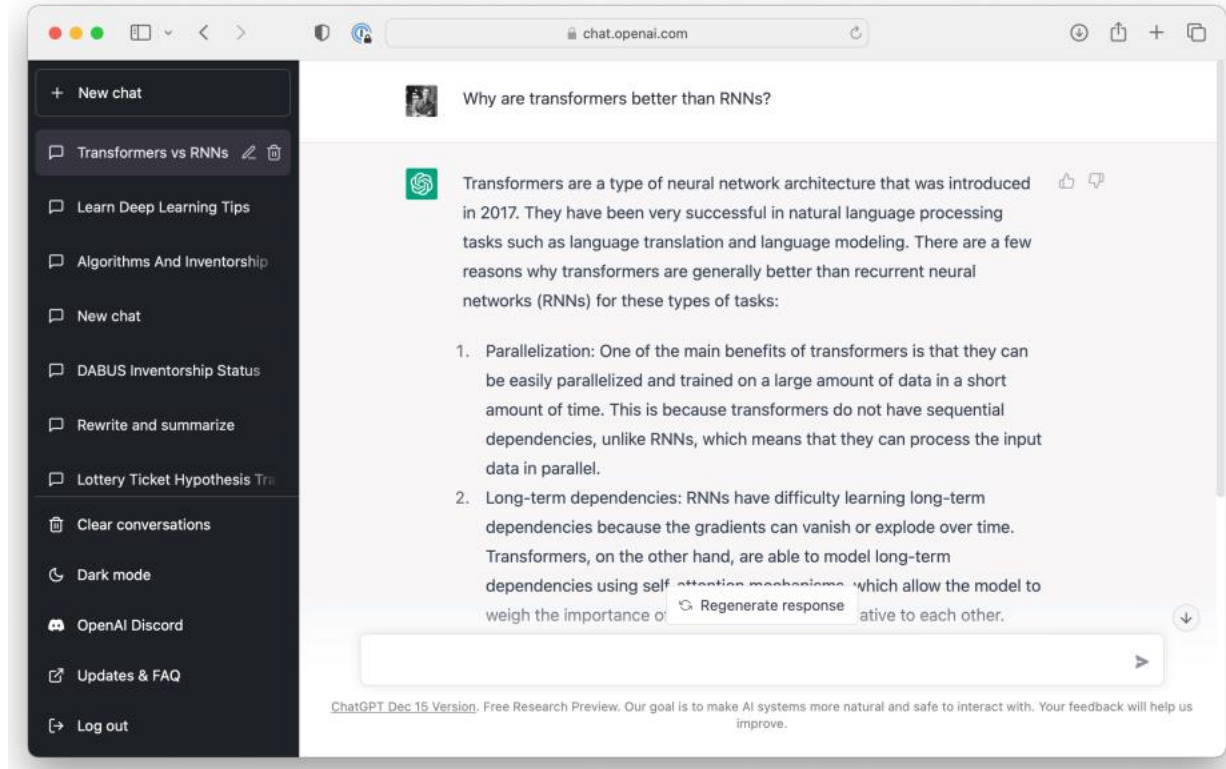


RNN rolled out over time

Introducción a transformers



Introducción a transformers



Introducción a transformers

nature

[View all journals](#) [Search](#) [Log in](#)

[Explore content](#) [About the journal](#) [Publish with us](#)

[nature](#) > [articles](#) > [article](#)

[Download PDF](#)

Article | [Open Access](#) | [Published: 15 July 2021](#)

Highly accurate protein structure prediction with AlphaFold

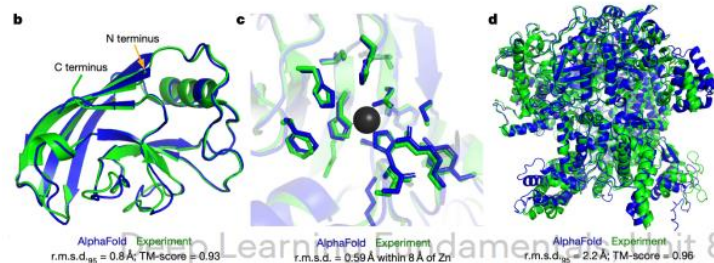
John Jumper , Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Židek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislaw Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, ... Demis Hassabis  [+ Show authors](#)

Nature **596**, 583–589 (2021) | [Cite this article](#)

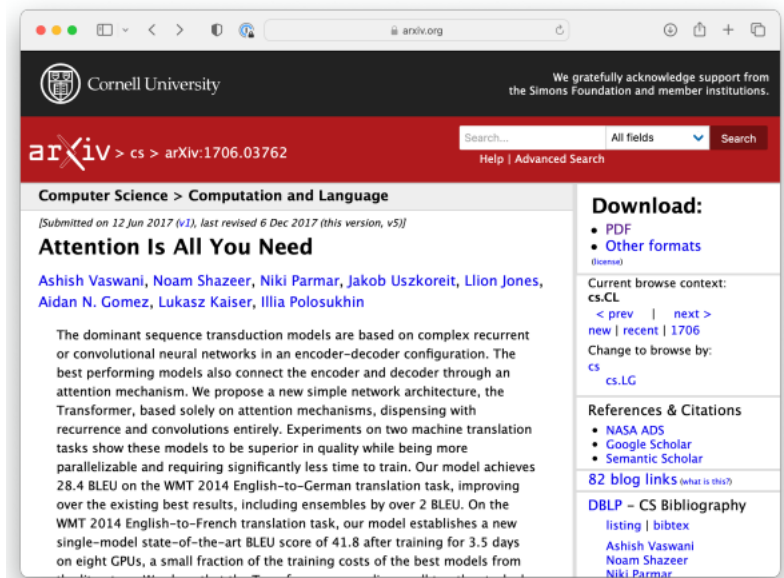
923k Accesses | **4944** Citations | **3393** Altmetric | [Metrics](#)

Abstract

Proteins are essential to life, and understanding their structure can facilitate a mechanistic understanding of their function. Through an enormous experimental effort^{1,2,3,4}, the structures of around 100,000



Artículo original



The screenshot shows the arXiv page for the paper "Attention Is All You Need" (arXiv:1706.03762) by Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. The paper is categorized under Computer Science > Computation and Language. The abstract states: "The dominant sequence transduction models are based on complex recurrent or convolutional neural networks in an encoder-decoder configuration. 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..."

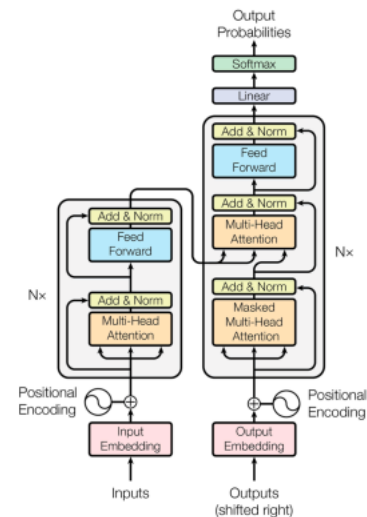


Figure 1: The Transformer - model architecture.

Atención

Nosotros no traducimos documentos palabra por palabra

Can you me help this sentence to translate
↑ ↑ ↑ ↑ ↑ ↑ ↑ ↑
Kannst du mir helfen diesen Satz zu uebersetzen ?

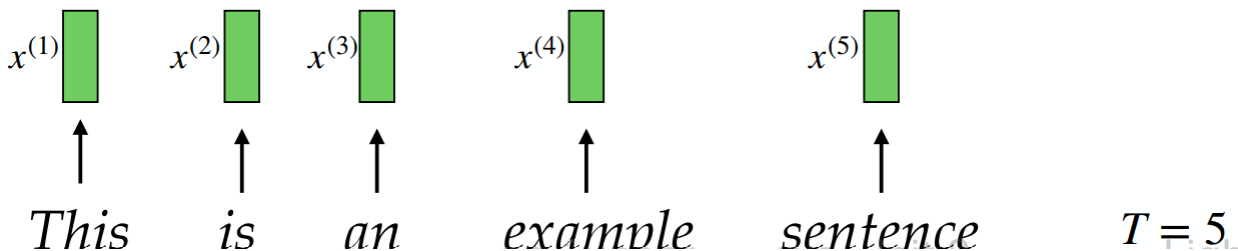
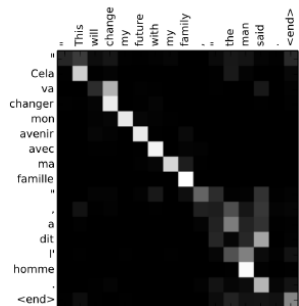
Can you help me to translate this sentence
↑ ↑ ↗ ↘ ↗ ↘ ↗ ↘
Kannst du mir helfen diesen Satz zu uebersetzen ?

Atención

Idea: Crear vectores de contexto que contengan información acerca de la secuencia completa. Utilizar **scores de atención** para ponderar la importancia de cada palabra

$$z^{(i)} = \sum_{j=1}^T \alpha_{ij} \cdot x^{(j)}$$

Ejemplo: $i = 2$



Self-attention

$$z^{(i)} = \sum_{j=1}^T \alpha_{ij} \cdot x^{(j)}$$

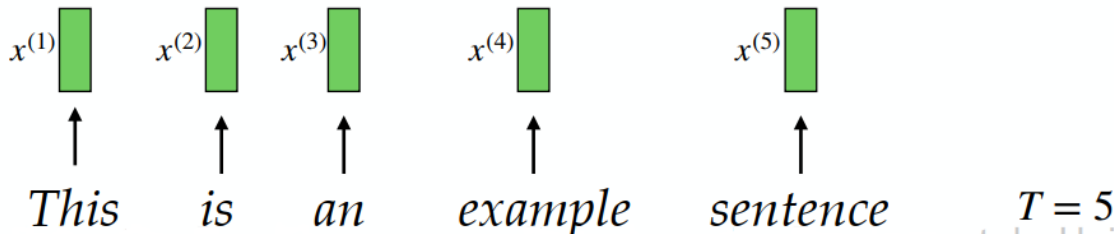
So that attention value
sum to 1 $\sum_{j=1}^T \alpha_{ij} = 1$

1. **Similarity** between i -th element all inputs $j = 1 \dots T$

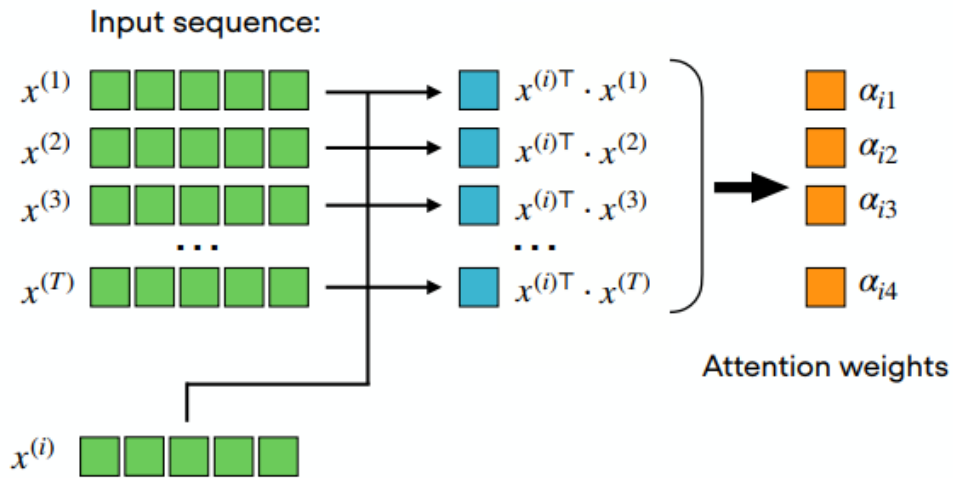
$$\omega_{ij} = x^{(i)\top} \cdot x^{(j)}$$

2. **Normalize** ω values to obtain attention scores α

$$\alpha_{ij} = \frac{\exp(\omega_{ij})}{\sum_{j=1}^T \exp(\omega_{ij})} = \text{softmax}\left(\left[\omega_{ij}\right]_{j=1 \dots T}\right)$$



Self-attention



$$z^{(i)} = \sum_{j=1}^T \alpha_{ij} \cdot x^{(j)}$$

$$\begin{aligned}
 & x^{(1)} \times \alpha_{i1} \\
 + & x^{(2)} \times \alpha_{i2} \\
 + & x^{(3)} \times \alpha_{i3} \\
 & \dots \\
 + & x^{(T)} \times \alpha_{i4} \\
 \\
 = & \text{Context vector } z^{(i)}
 \end{aligned}$$

Self-attention with learnable weights

Idea: El modelo aprende qué tan importante es cada palabra. Modelo de atención más utilizado y propuesto en el paper original

query sequence: $q^{(i)} = U_q x^{(i)}$ for $i \in [1, \dots, T]$

key sequence: $k^{(i)} = U_k x^{(i)}$ for $i \in [1, \dots, T]$

value sequence: $v^{(i)} = U_v x^{(i)}$ for $i \in [1, \dots, T]$

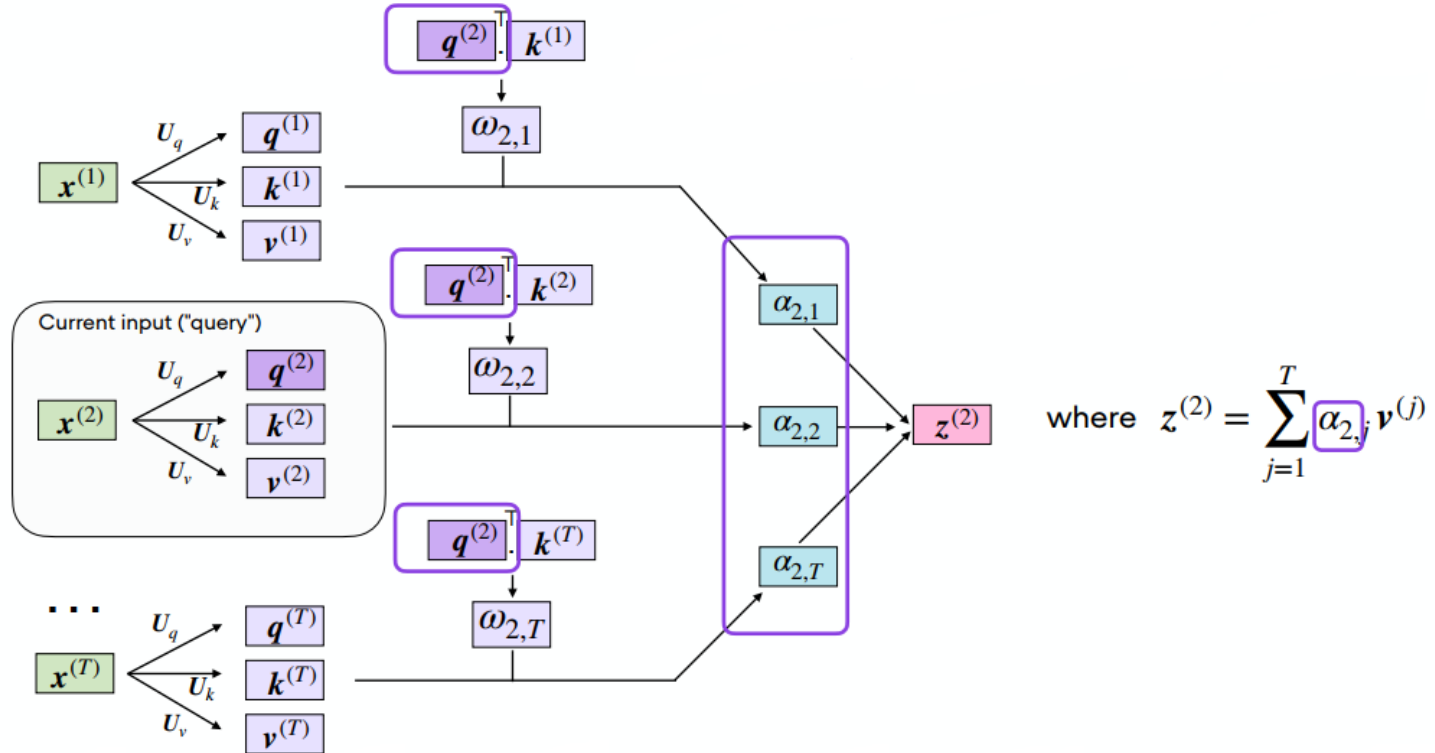
Self-attention with learnable weights

Query, key y value se inspiran en sistemas de recuperación de información.

query se compara con **key** para devolver un **value**.

Al igual al módulo básico de self-attention, debemos calcular un **vector de contexto**

Self-attention with learnable weights

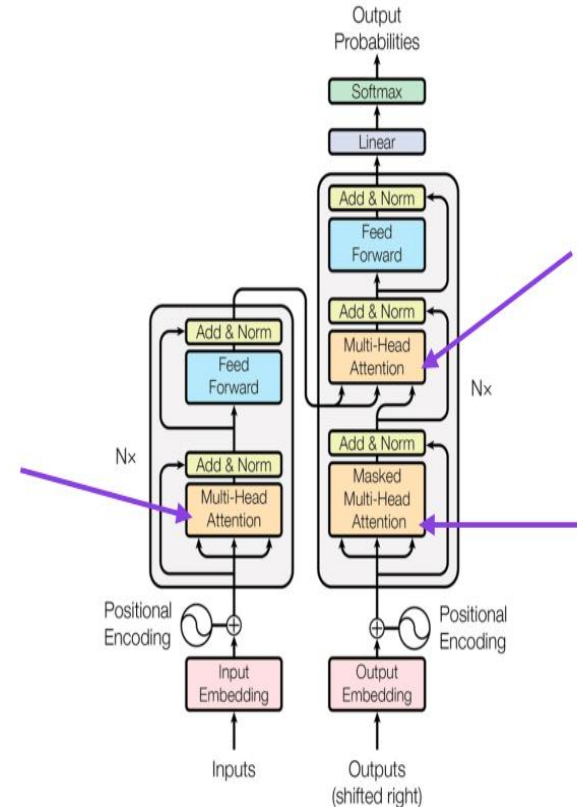


Self-attention with learnable weights

Resumen

Para cada token, el mecanismo de self-attention:

- Compara el query de cada palabra con los keys de todas las palabras en la secuencia de entrada
- Calcula el score de atención a partir de los valores obtenidos en la comparación
- Calcula el promedio ponderado de todas las entradas
- Metodología sequence-to-sequence
 - Toma T entradas
 - Devuelve T salidas



Multi-head attention

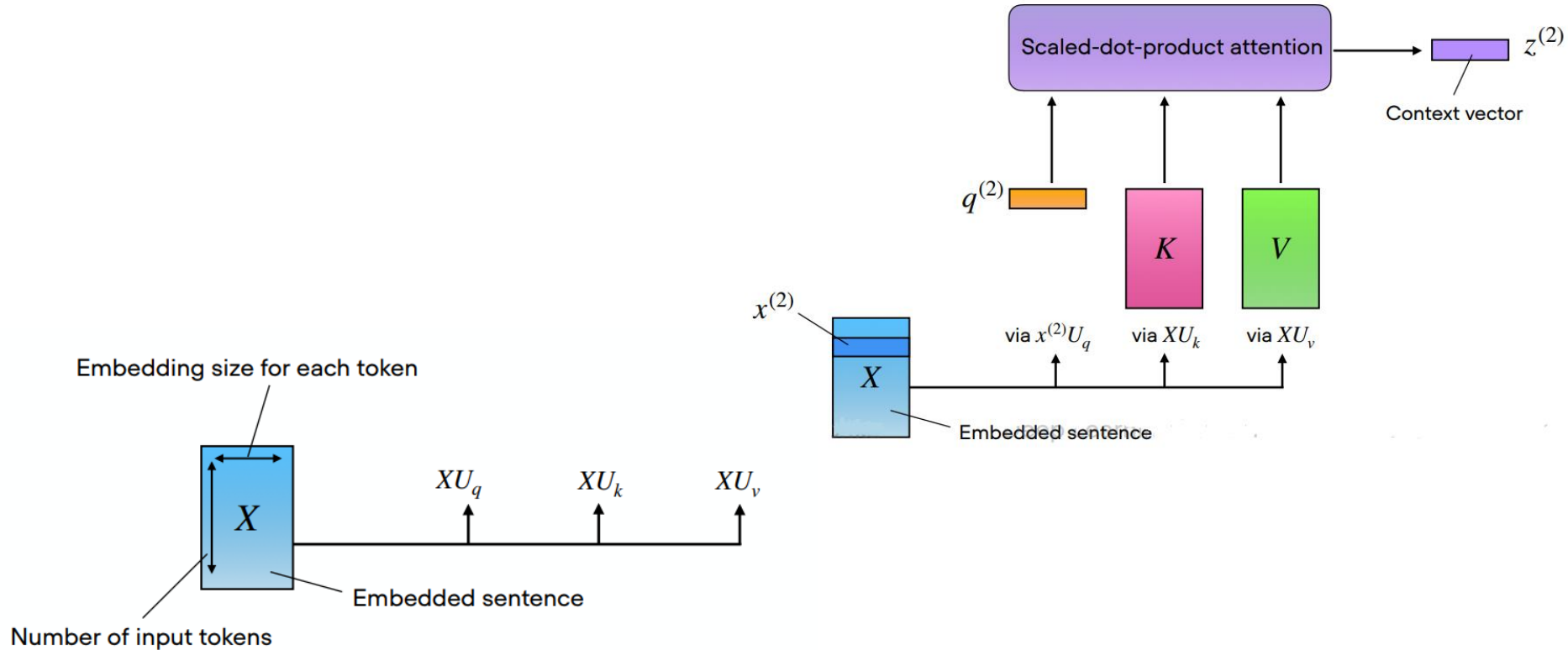
Anteriormente se usaron tres matrices de parámetros. Agreguemos un índice adicional

$$U_{q_1} \quad U_{k_1} \quad U_{v_1}$$

En multi-head attention tenemos un conjunto de matrices

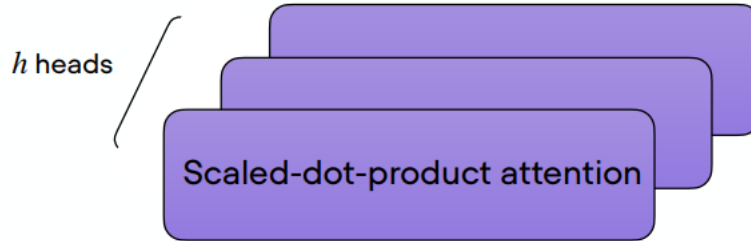
$$\begin{array}{ccc} U_{q_2} & U_{k_2} & U_{v_2} \\ U_{q_3} & U_{k_3} & U_{v_3} \\ \dots & & \end{array}$$

Multi-head attention

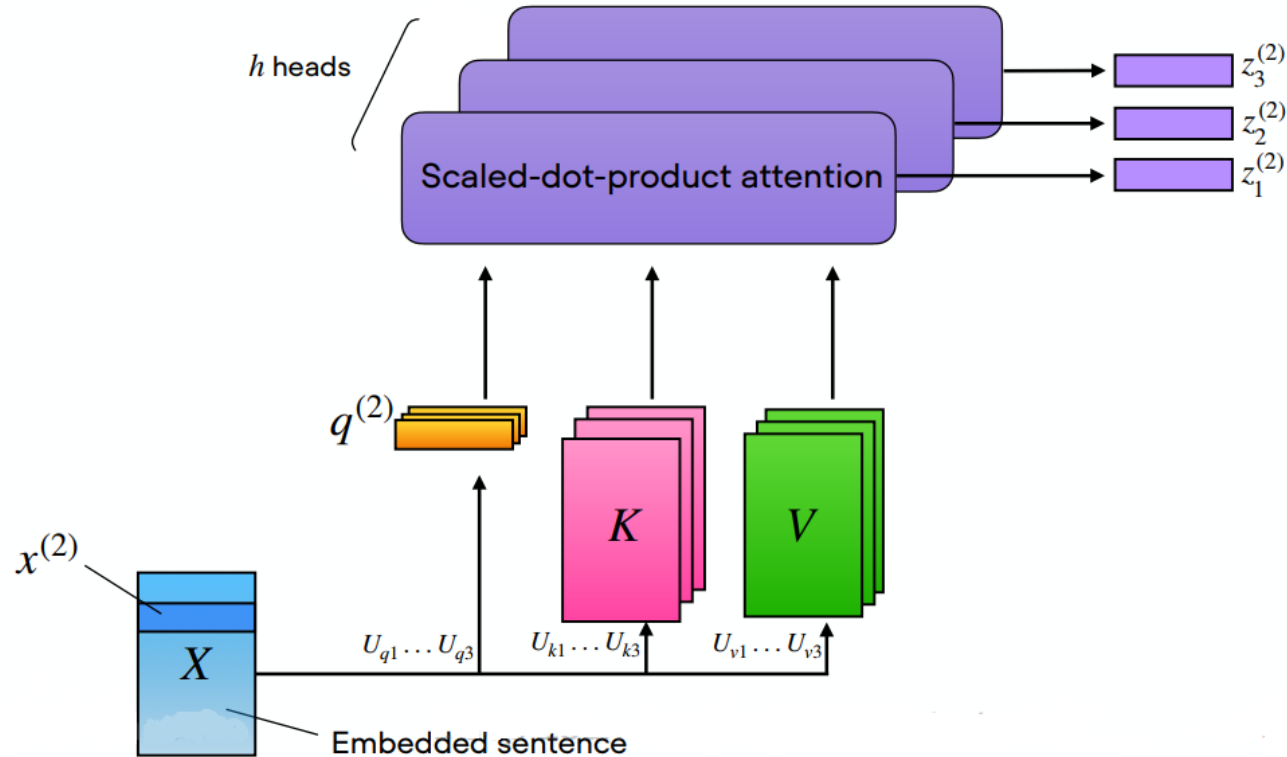


Multi-head attention

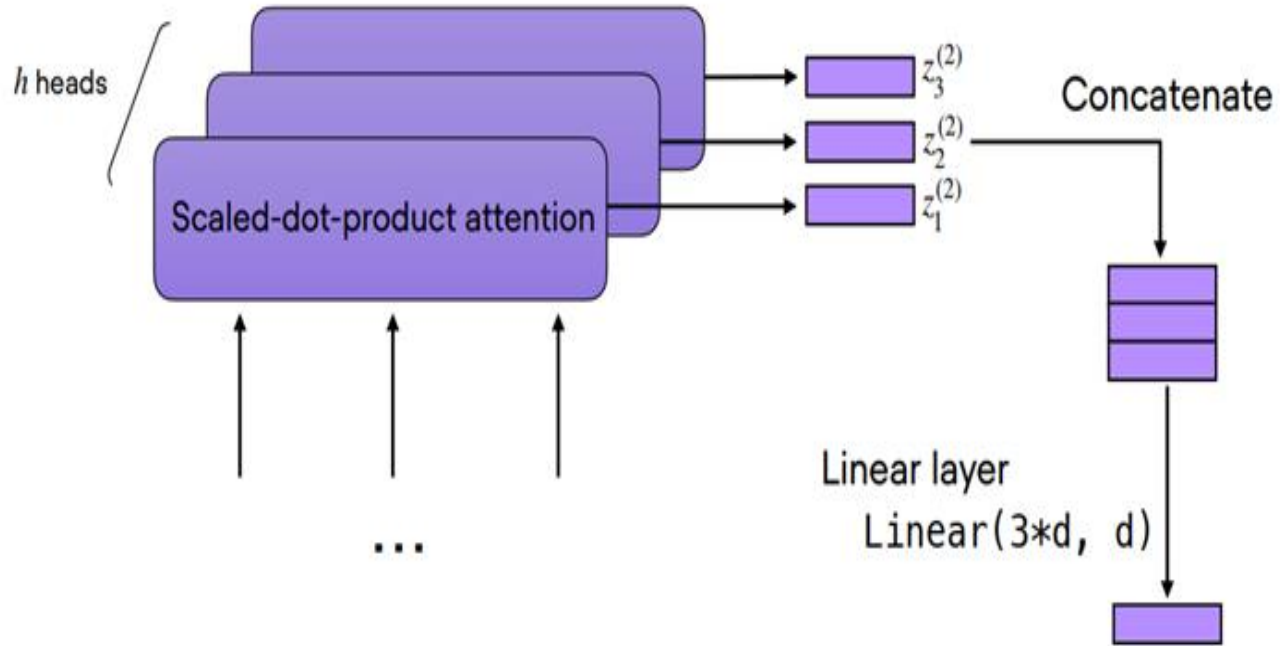
Similar a tener varios filtros convolucionales



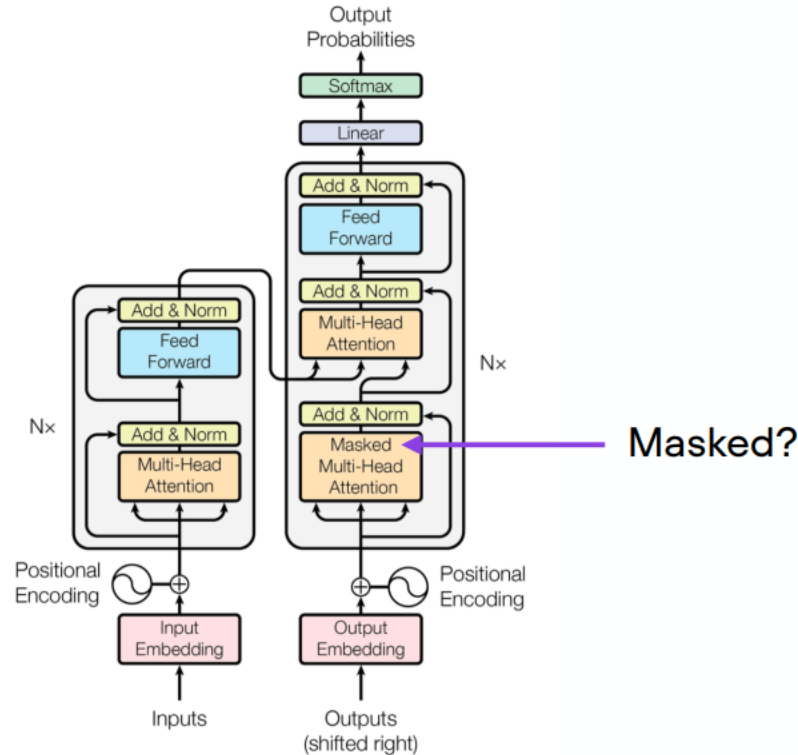
Multi-head attention



Multi-head attention



Masked Multi-head attention



Masked Multi-head attention

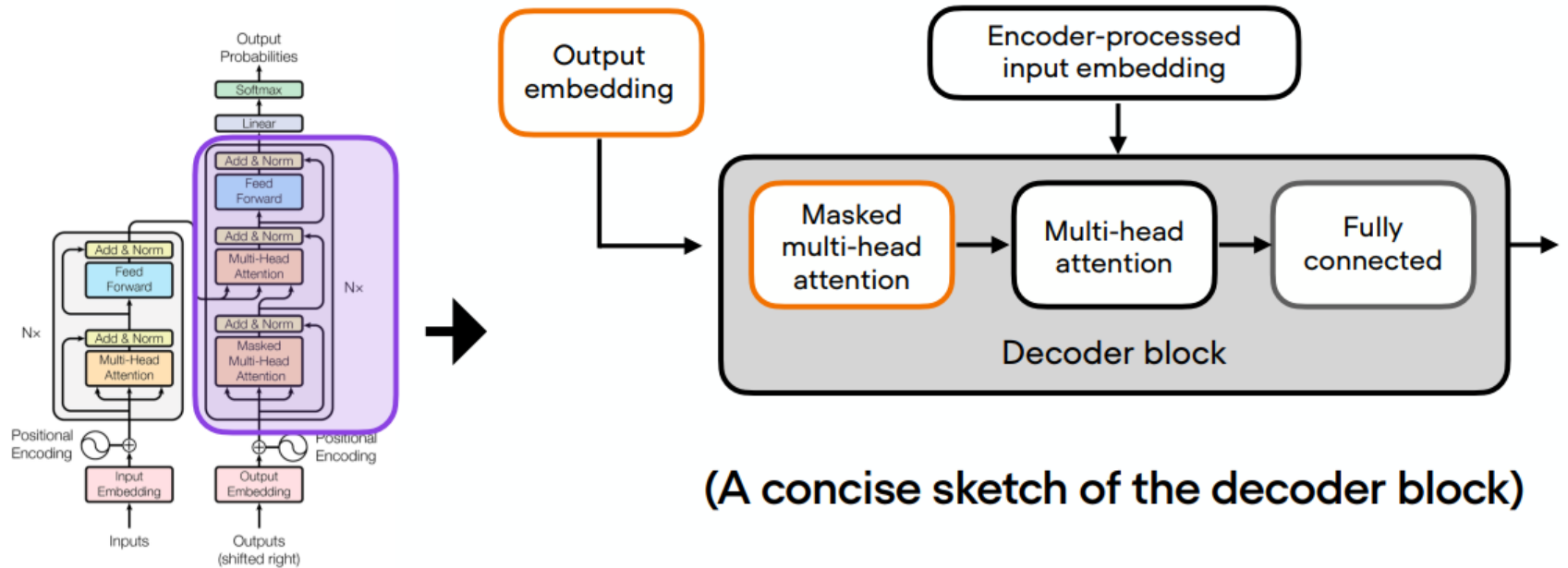


Figure 1: The Transformer - model architecture.

(A concise sketch of the decoder block)

Masked Multi-head attention

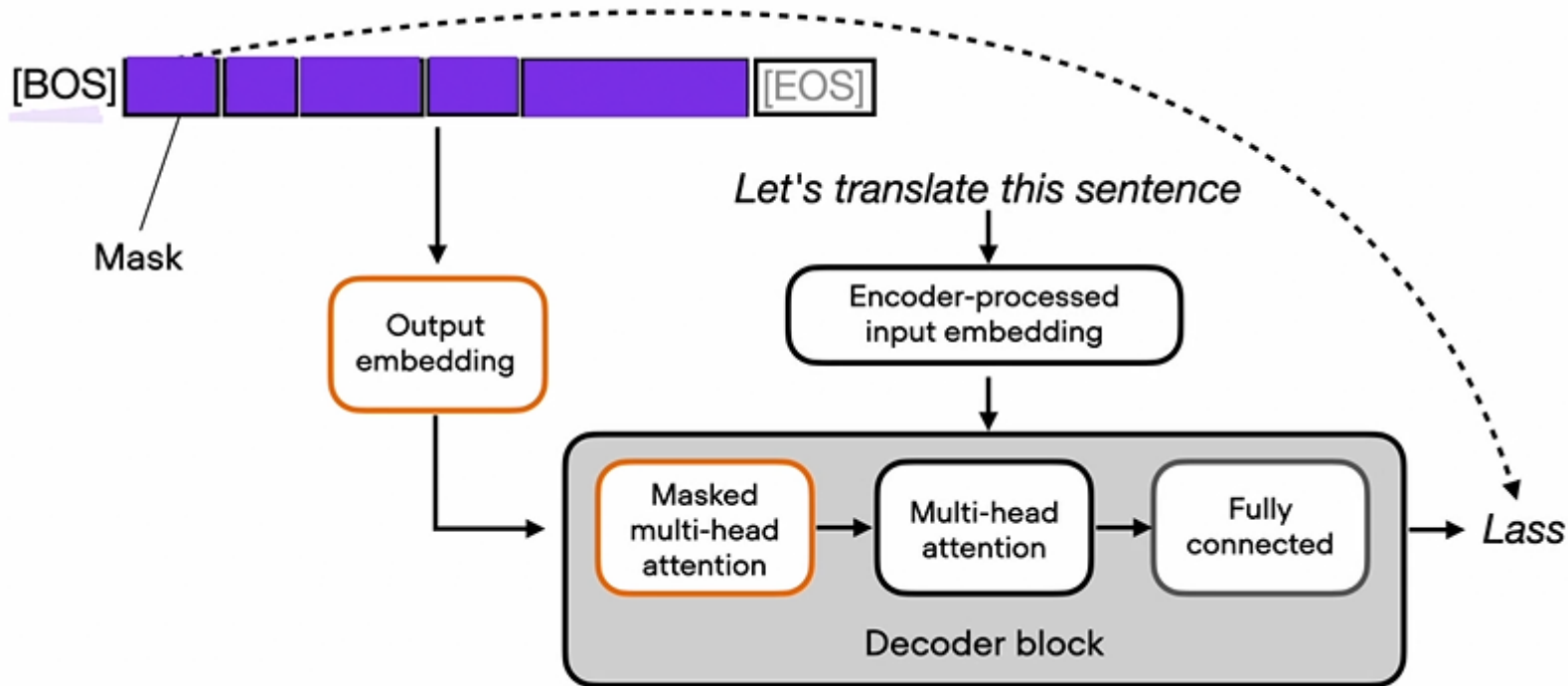
English → German Translation

Input: *Let's translate this sentence*

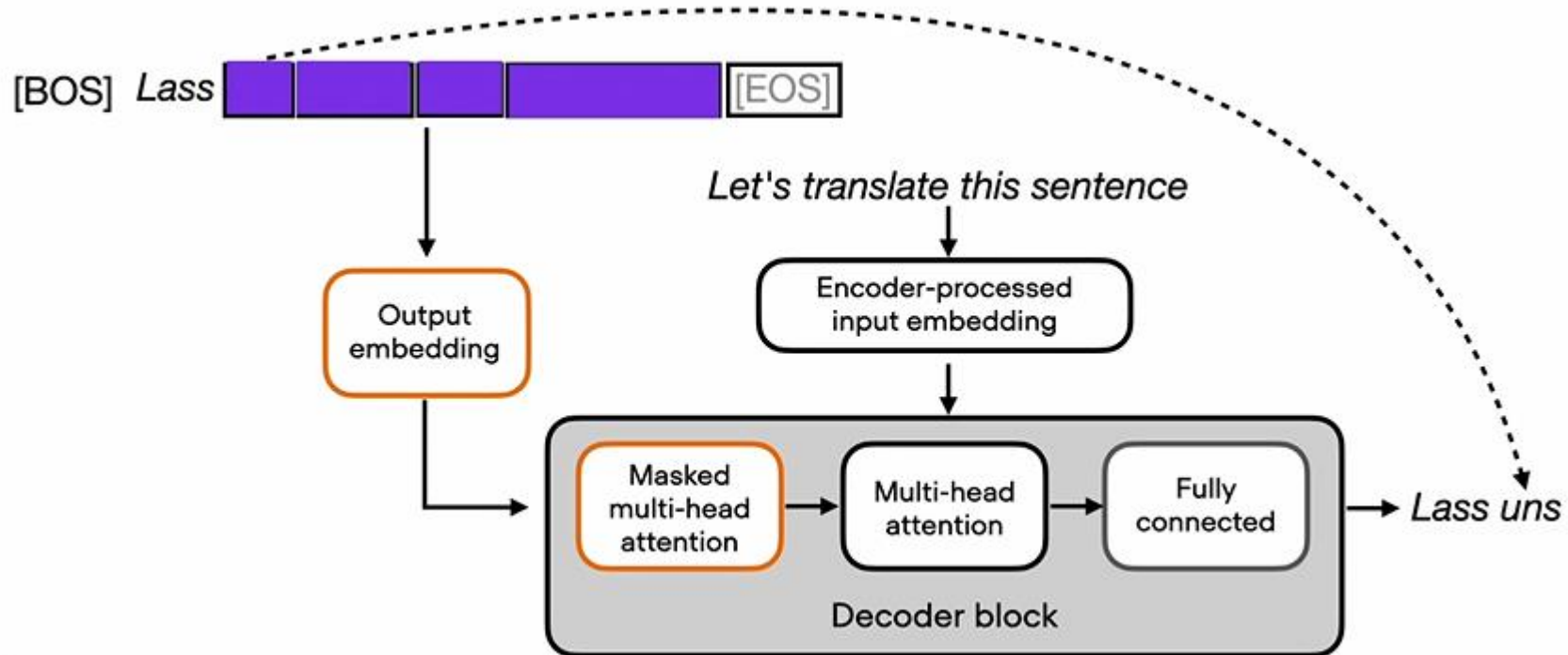
Target: *Lass uns diesen Satz uebersetzen*

Masked Multi-head attention

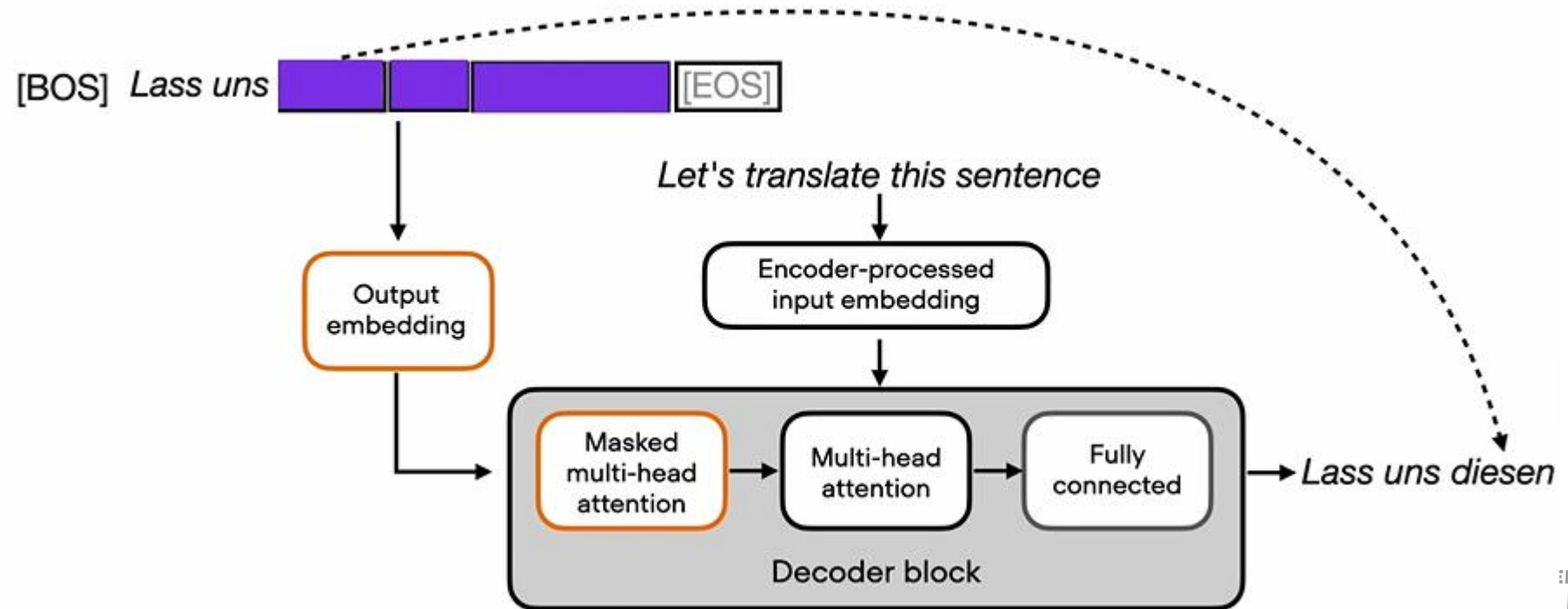
Masked attention enmascara los tokens que el modelo no ha visto.
Acá visualizamos en las entradas en lugar de los tokens por facilidad



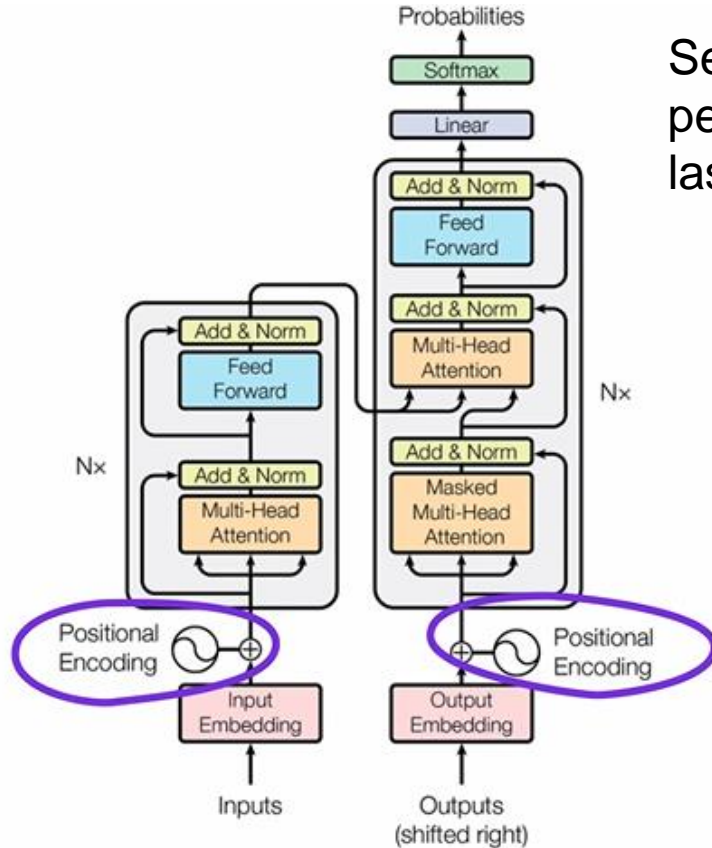
Masked Multi-head attention



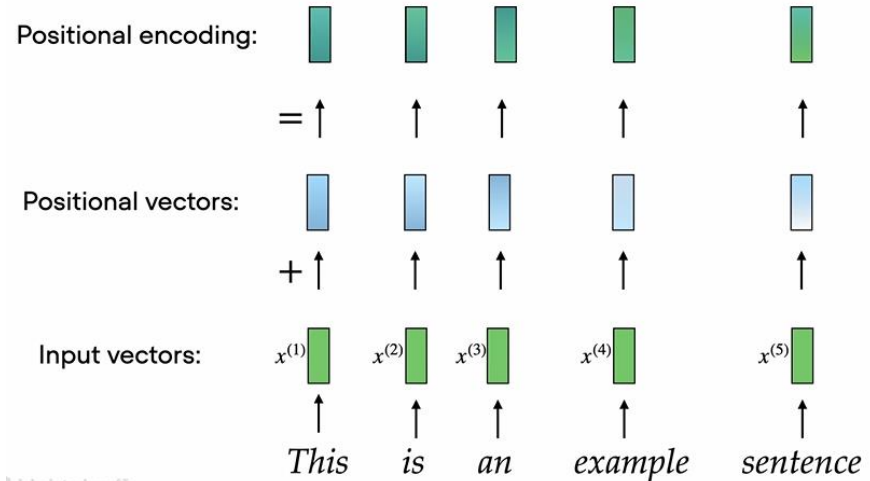
Masked Multi-head attention



Positional encoding



Self-attention y Fully Connected son invariantes a la permutación, i.e., no tienen en cuenta el orden de las palabras en la entrada



El inicio de la red

Generative Pretrained Transformers - GPT

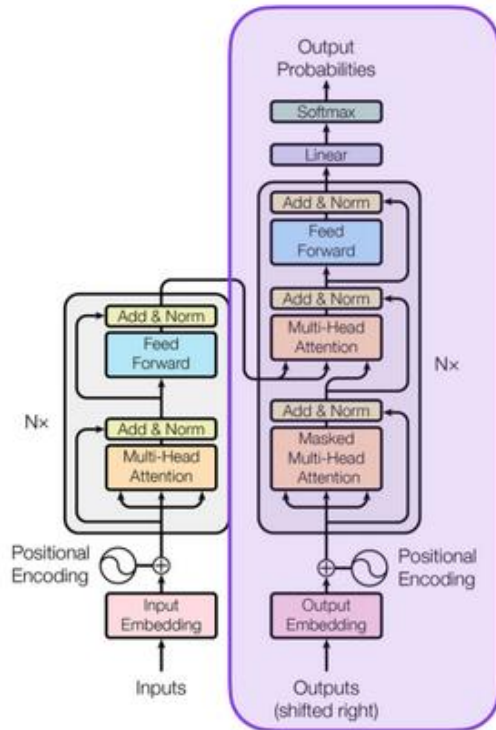
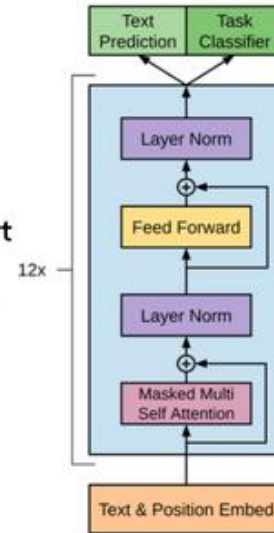


Figure 1: The Transformer - model architecture.

GPT Recap

GPT is essentially the **decoder** part of the original transformer



https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf

Generative Pretrained Transformers - GPT

GPT ingresa el texto de izquierda a derecha de forma que el modelo aprende a predecir la palabra siguiente

Self-supervised pre-training

1. Pre-entrenar: Predecir la palabra siguiente (self-attention unidireccional)
2. Fine-tune

Self-supervised pre-training

1. Pre-entrenar:

- a. Predecir palabras aleatoriamente enmascaradas (bi-direccional/no direccional)
- b. Predecir el orden de las oraciones

2. Fine-tune

BERT

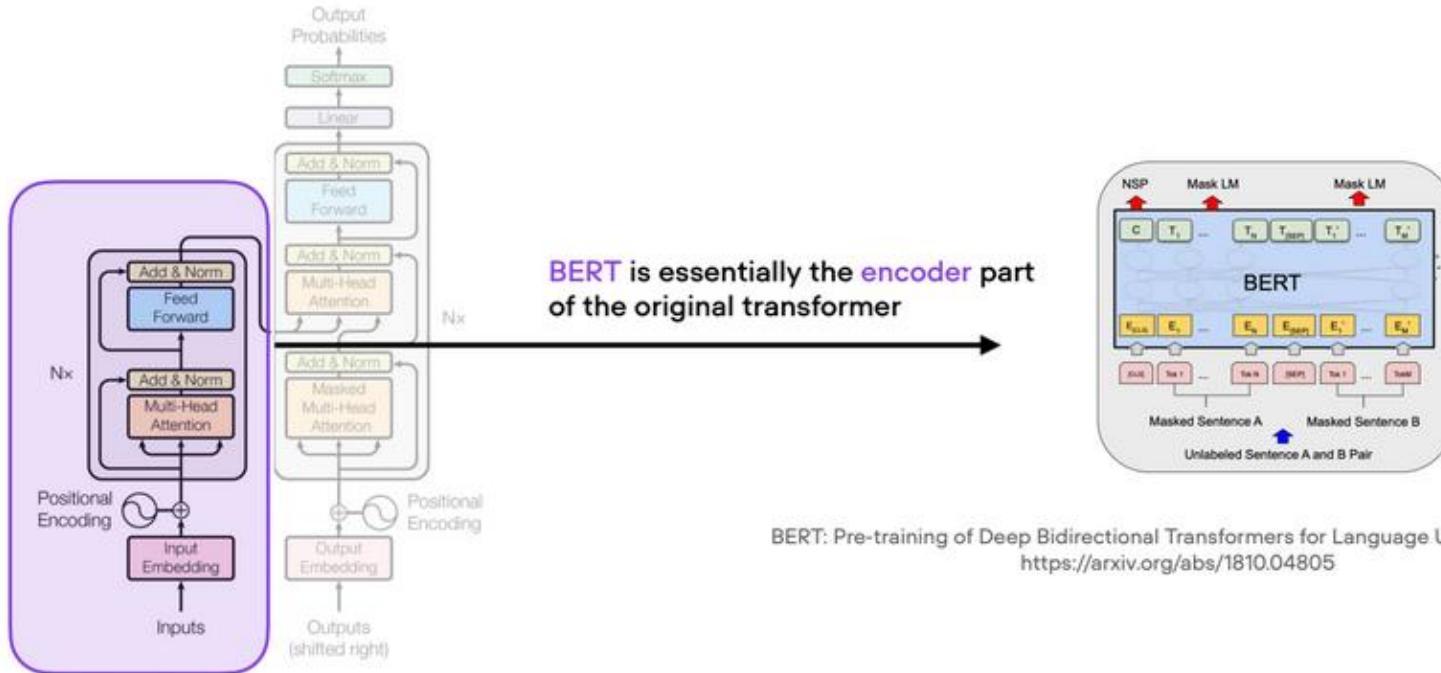


Figure 1: The Transformer - model architecture.

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,
<https://arxiv.org/abs/1810.04805>

BERT

1. Pre-entrenar en un conjunto de datos no etiquetado (aprender un modelo de lenguaje general)
 - a. Predecir palabras aleatoriamente enmascaradas (bi-direccional/no direccional)

Input sentence: *The curious kitten deftly climbed the bookshelf*



Pick 15% of the words randomly

The curious kitten deftly climbed the bookshelf



- 80% of the time, replace with [MASK] token
- 10% of the time, replace with random token (e.g. **ate**)
- 10% of the time, keep unchanged

BERT

1. Pre-entrenar en un conjunto de datos no etiquetado (aprender un modelo de lenguaje general)
 - b. Predecir el orden de las oraciones

[CLS] Sentence A [SEP] Sentence B

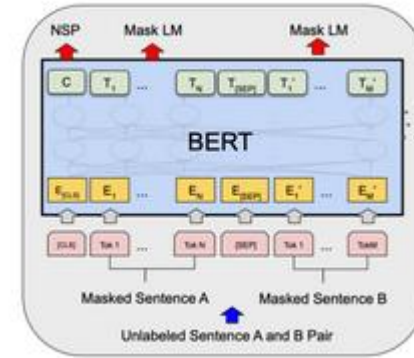
Placeholder for the `IsNext=True / False` label in the decoder output

[CLS] Toast is a simple yet delicious food [SEP] It's often served with butter, jam, or honey.

IsNext = True

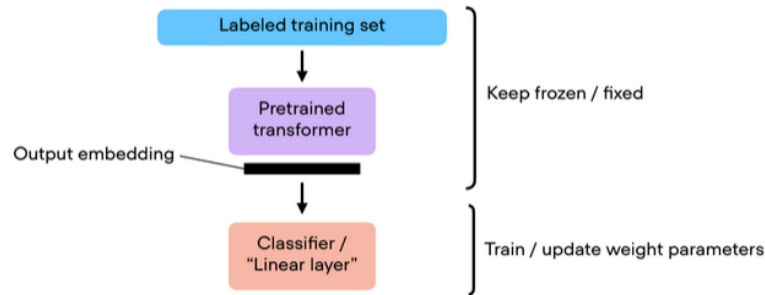
[CLS] It's often served with butter, jam, or honey. [SEP] Toast is a simple yet delicious food.

IsNext = False

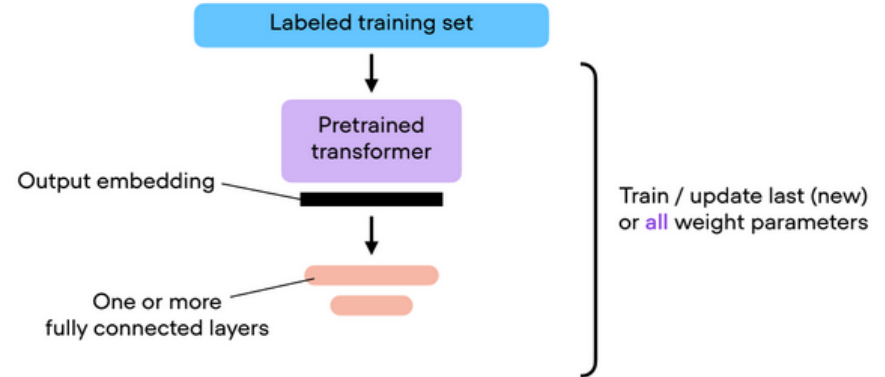


Utilización de Transformers pre-entrenados

1. Feature-based approach

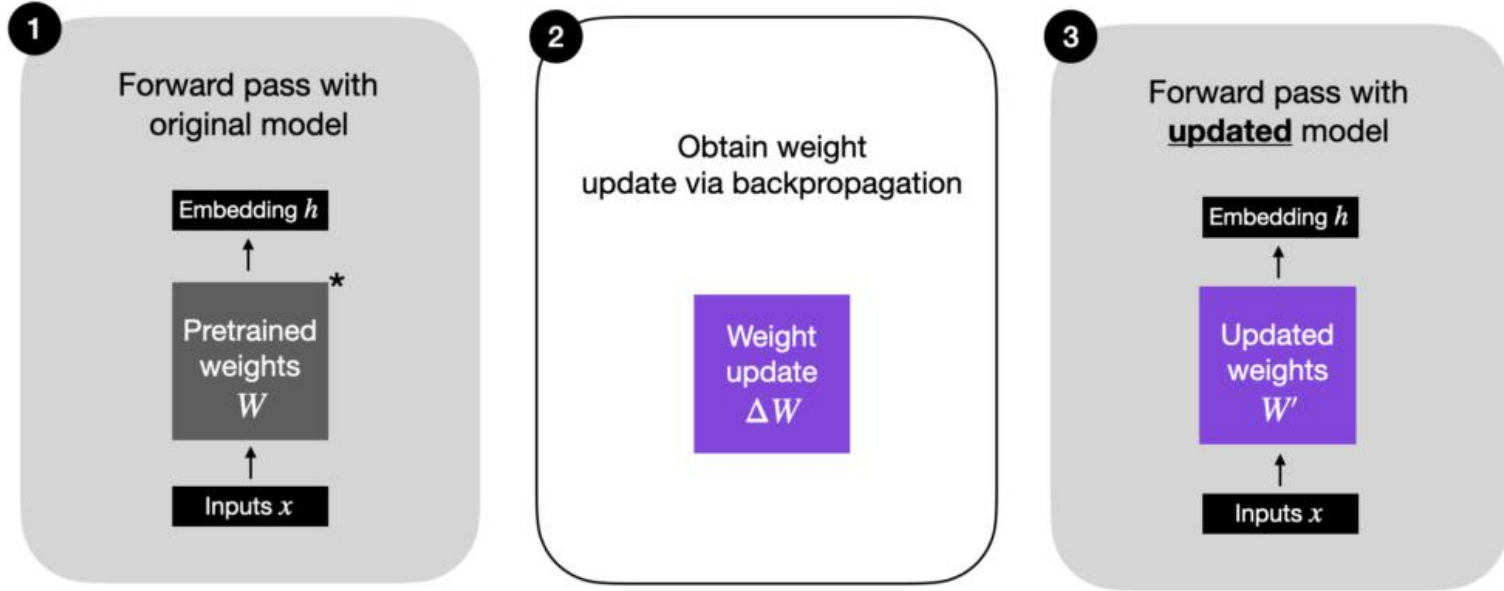


2. Fine-tuning approach



Parameter Efficient Fine Tuning - PEFT

Regular Finetuning



* The pretrained model could be any LLM, e.g., an encoder-style LLM (like BERT) or a generative decoder-style LLM (like GPT)

LoRA: Low-Rank Adaptation of LLMs

Computer Science > Computation and Language

[Submitted on 17 Jun 2021 (v1), last revised 16 Oct 2021 (this version, v2)]

LoRA: Low-Rank Adaptation of Large Language Models

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen

An important paradigm of natural language processing consists of large-scale pre-training on general domain data and adaptation to particular tasks or domains. As we pre-train larger models, full fine-tuning, which retraining all model parameters, becomes less feasible. Using GPT-3 175B as an example -- deploying independent instances of fine-tuned models, each with 175B parameters, is prohibitively expensive. We propose Low-Rank Adaptation, or LoRA, which freezes the pre-trained model weights and injects trainable rank decomposition matrices into each layer of the Transformer architecture, greatly reducing the number of trainable parameters for downstream tasks. Compared to GPT-3 175B fine-tuned with Adam, LoRA can reduce the number of trainable parameters by 10,000 times and the GPU memory requirement by 3 times. LoRA performs on-par or better than fine-tuning in model quality on RoBERTa, DeBERTa, GPT-2, and GPT-3, despite having fewer trainable parameters, a higher training throughput, and, unlike adapters, no additional inference latency. We also provide an empirical investigation into rank-deficiency in language model adaptation, which sheds light on the efficacy of LoRA. We release a package that facilitates the integration of LoRA with PyTorch models and provide our implementations and model checkpoints for RoBERTa, DeBERTa, and GPT-2 at [this https URL](https://github.com/microsoft/LoRA).

Descomponer ΔW en una representación de menor rango

Regular fine-tuning



Actualización:

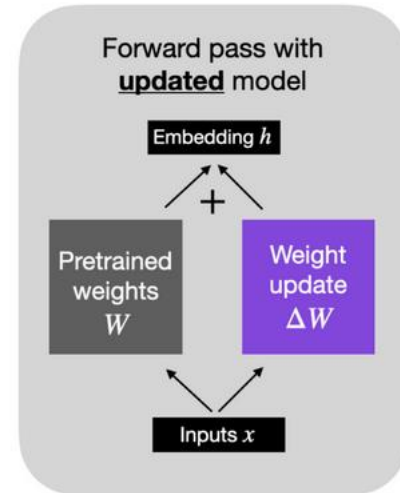
$$\Delta \mathbf{W} = -\alpha \frac{\partial \mathcal{L}}{\partial \mathbf{W}}$$

$$\mathbf{W} = \mathbf{W} + \Delta \mathbf{W}$$

$$\mathbf{h} = \mathbf{W}\mathbf{x} + \mathbf{b}$$

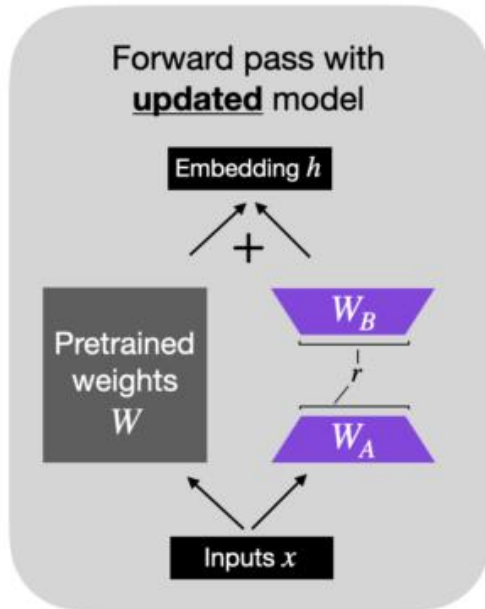
Alternativa:

$$\mathbf{h} = \mathbf{W}\mathbf{x} + \Delta \mathbf{W}\mathbf{x} + \mathbf{b}$$



LoRA: Low-Rank Adaptation of LLMs

LoRA weights, W_A and W_B , represent ΔW



$$\Delta W = W_A W_B$$

$$W_A \in \mathbb{R}^{A \times r}$$

$$W_B \in \mathbb{R}^{r \times B}$$

$$W \in \mathbb{R}^{A \times B}$$

Mantener "congelado" W y actualizar W_{AB}

Si $r \ll \{A, B\}$ se actualizan muchos menos parámetros