

Time Series Forecasting

Transformers

National University of Singapore (NUS)

Félix Fuentes

Curso 2020/2021 – Edición II

Fecha 27/03/2021

Índice

1. Attention is all you need
2. ¿Qué son los Transformers?
3. ¿Cómo funcionan los Transformers?

Transformers

Attention is all you need

- A mediados de 2017 aparece un artículo científico ("*paper*", para los amigos) que demuestra que existe una **alternativa** a las redes recurrentes para trabajar con problemas secuenciales.
- En concreto, habla del concepto de "**atención**", que básicamente significa "ponderar" los elementos de nuestras secuencias según afecten más o menos a las predicciones.
- Habla sobre todo de NLP (Natural Language Processing), pero también es aplicable a series temporales.
- Elimina el concepto de recursión, con lo que el entrenamiento puede ser mucho más rápido

The screenshot shows the arXiv page for the paper "Attention Is All You Need" (arXiv:1706.03762). The page is from Cornell University and includes a search bar, navigation links, and a list of download options (PDF, Other formats). The abstract is visible, discussing the Transformer model. The page also features a sidebar with references and citations, and a submission history section.

Cornell University

We gratefully acknowledge support from the Simons Foundation and member institutions.

arXiv.org > cs > arXiv:1706.03762

Search... All fields Search

Help | Advanced Search

Computer Science > Computation and Language

[Submitted on 12 Jun 2017 (v1), last revised 6 Dec 2017 (this version, v5)]

Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin

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 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.

Comments: 15 pages, 5 figures

Subjects: **Computation and Language (cs.CL)**; Machine Learning (cs.LG)

Cite as: arXiv:1706.03762 [cs.CL]
(or arXiv:1706.03762v5 [cs.CL] for this version)

Submission history

From: Ashish Vaswani [view email]

[v1] Mon, 12 Jun 2017 17:57:34 UTC (1,102 KB)

[v2] Mon, 19 Jun 2017 16:49:45 UTC (1,125 KB)

[v3] Tue, 20 Jun 2017 05:20:02 UTC (1,125 KB)

[v4] Fri, 30 Jun 2017 17:29:30 UTC (1,124 KB)

[v5] Wed, 6 Dec 2017 03:30:32 UTC (1,124 KB)

Download:

- PDF
- Other formats (license)

Current browse context: cs.CL

< prev | next >

new | recent | 1706

Change to browse by: cs

cs.LG

References & Citations

- NASA ADS
- Google Scholar
- Semantic Scholar

50 blog links (what is this?)

DBLP - CS Bibliography

listing | bibtex

Ashish Vaswani
Noam Shazeer
Niki Parmar
Jakob Uszkoreit
Llion Jones

...

Export BibTeX Citation

Bookmark

PDF icon, BibTeX icon, HTML icon, etc.

Transformers

¿Qué son los Transformers?

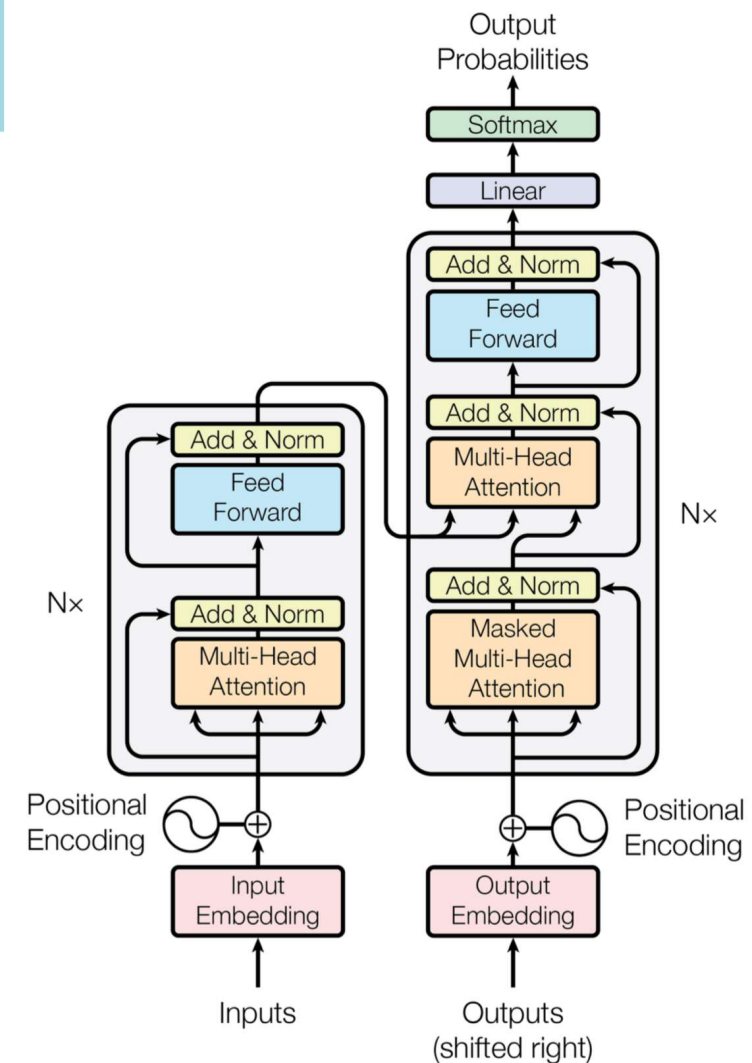
Una arquitectura **encoder-decoder** con **atención**.

La atención mira a la secuencia de entrada y decide para cada instante qué otras partes de la misma son importantes (podéis entenderlo como el "**contexto**").

Consiste en bloques encoder y decoder apilables (Nx bloques)

Capas **densas** y bloques Multi-Head Attention (MHA).

Positional embedding (para introducir la información "temporal" en el caso de las series temporales o "situacional" en el caso de NLP)



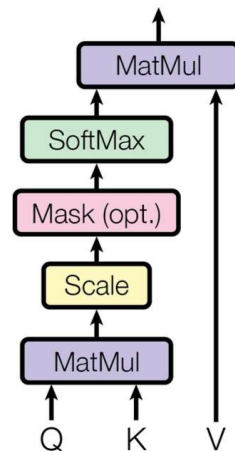
Transformers

¿Qué son los Transformers?

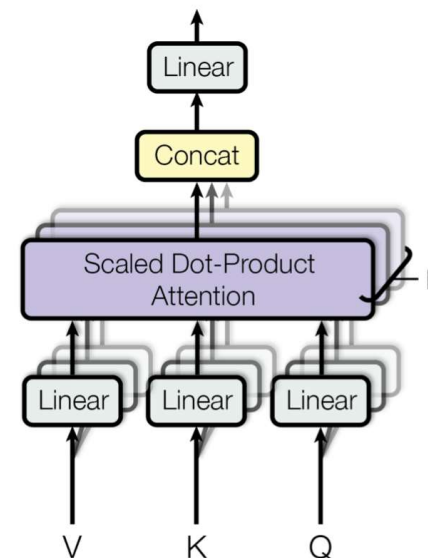
El bloque **Multi-Head Attention** es el que permite al modelo “**fijarse**” en las cosas más importantes.

Funciona **creando varios mapas de activación** (Q: query, K: key, V: value) y **comprobando** sus **relaciones**.

Scaled Dot-Product Attention



Multi-Head Attention



Transformers

¿Cómo funcionan los Transformers?

Vamos a ver la arquitectura transformer explicada en el campo del procesamiento del lenguaje natural, ya que es el caso de uso más típico.

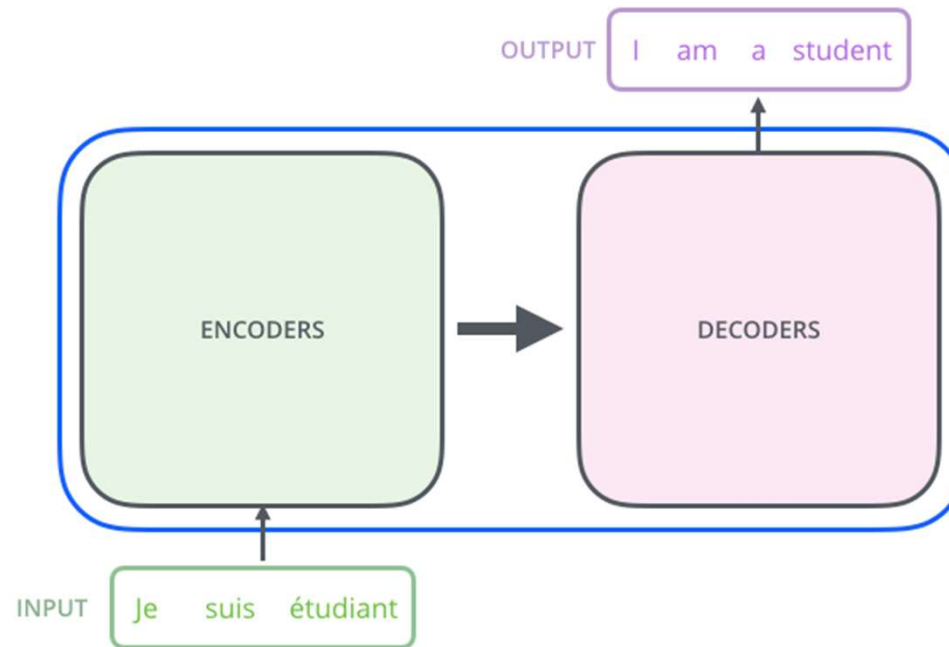
Para el caso de series temporales es muy similar, lo único que cambia es la codificación de los datos de entrada: ya no son palabras, sino números.



Transformers

¿Cómo funcionan los Transformers?

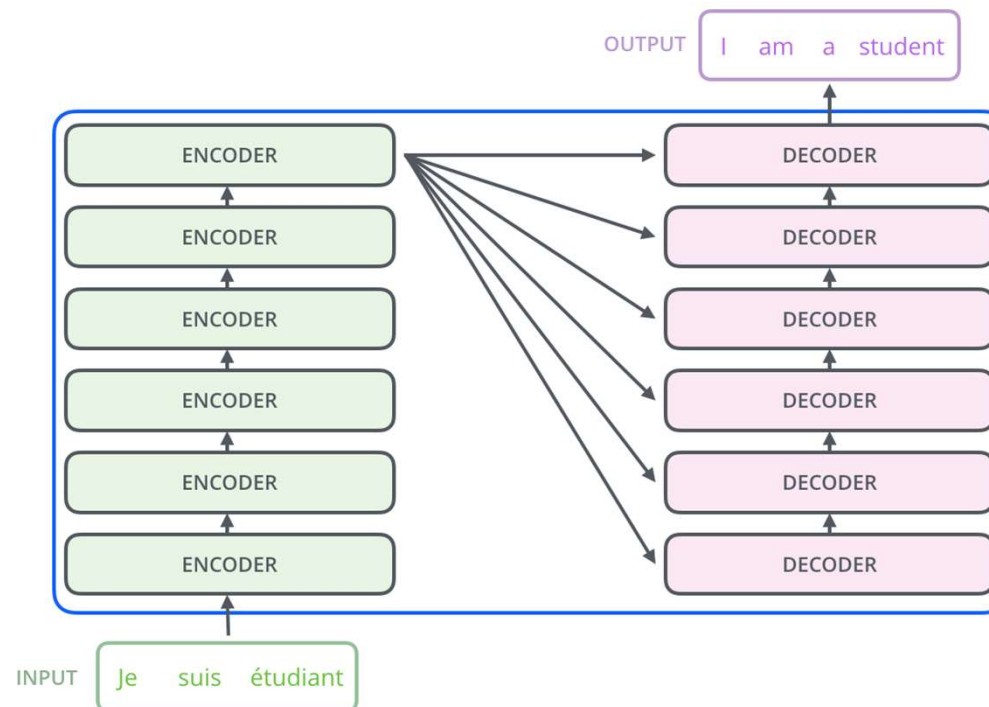
Están compuestos por un encoder y un decoder.



Transformers

¿Cómo funcionan los Transformers?

O para ser más exactos, de N bloques de encoders apilados seguidos de N bloques de decoders apilados.

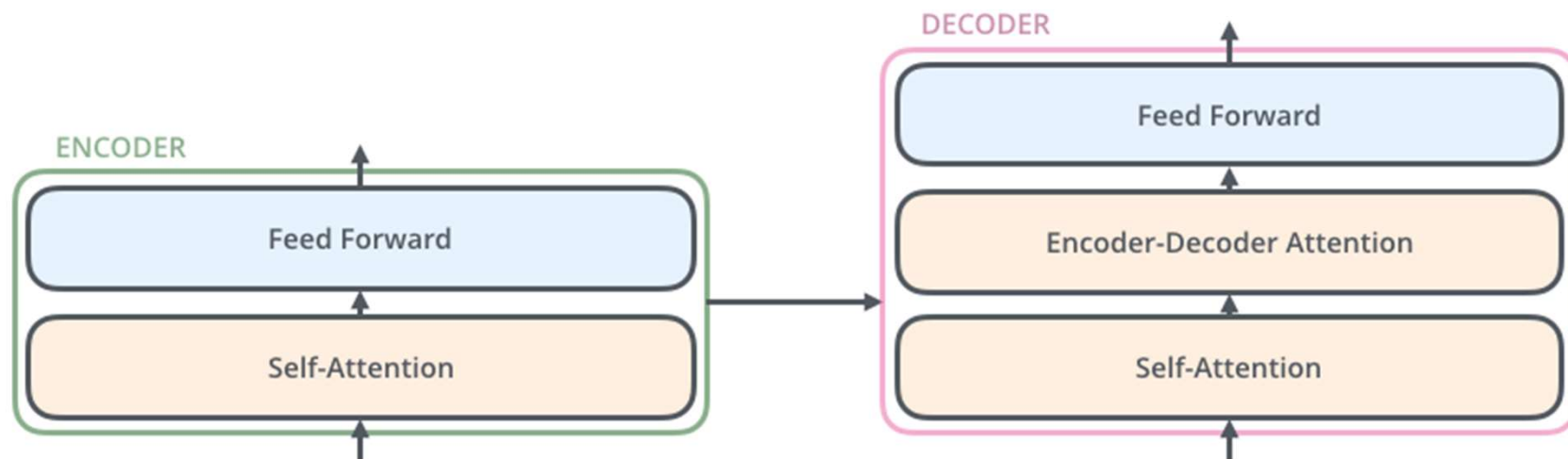


Transformers

¿Cómo funcionan los Transformers?

Dentro de **cada encoder** tendremos un bloque de **capas feed-forward** (las densas de toda la vida) y otro bloque de **Self-Attention**, el cual ayuda al encoder a **fijarse** no solo en la palabra que está procesando en ese momento, sino también en las **demás** palabras de la **secuencia** de entrada.

En los **decoders**, tendremos además de eso, un bloque de **Encoder-Decoder attention** que le permita decirle qué partes de la secuencia de entrada son importantes.

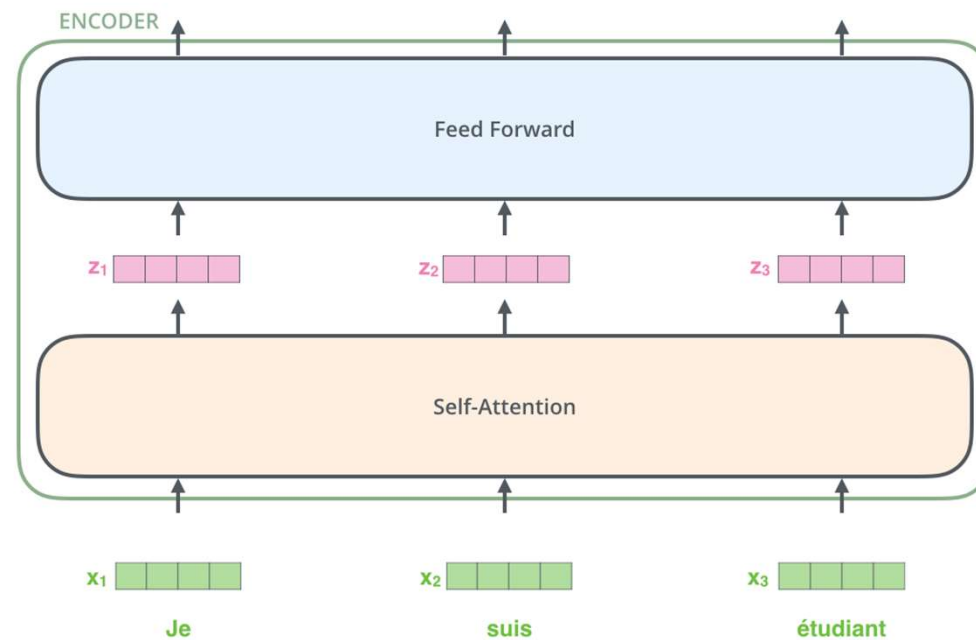


Transformers

¿Cómo funcionan los Transformers?

Los vectores de entrada **fluyen** a través de los bloques cada uno en una **posición fija**. El encoder crea **dependencias** entre estos “**caminos**” en los bloques de “**self-attention**”.

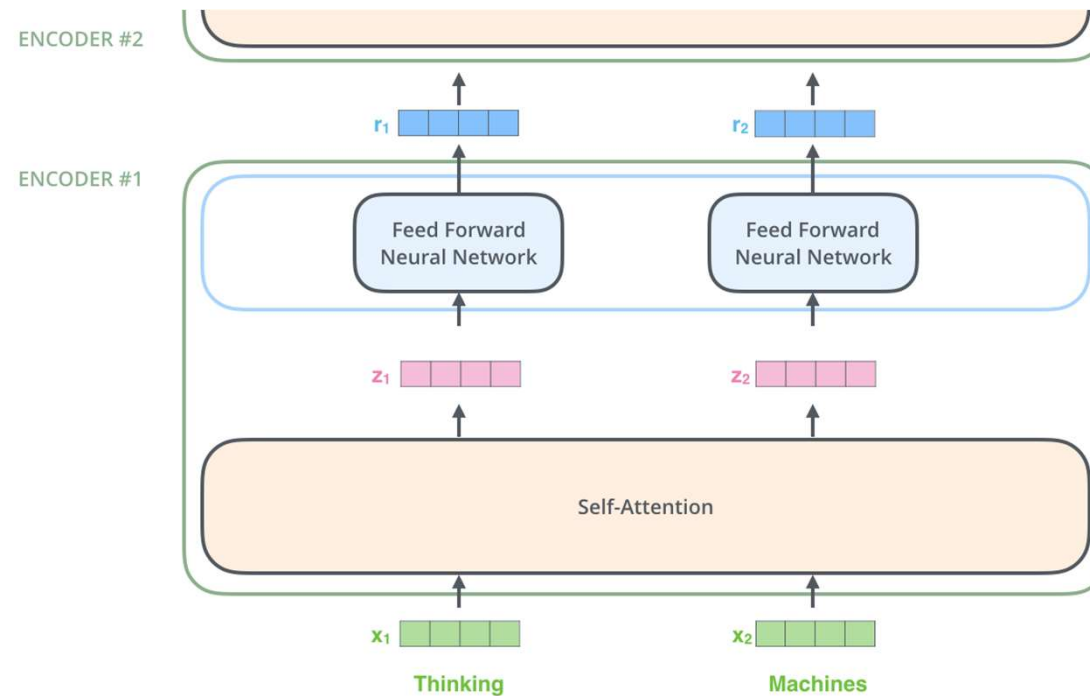
¡Esto se puede **paralelizar**!



Transformers

¿Cómo funcionan los Transformers?

Este proceso se repite para cada bloque “encoder”.



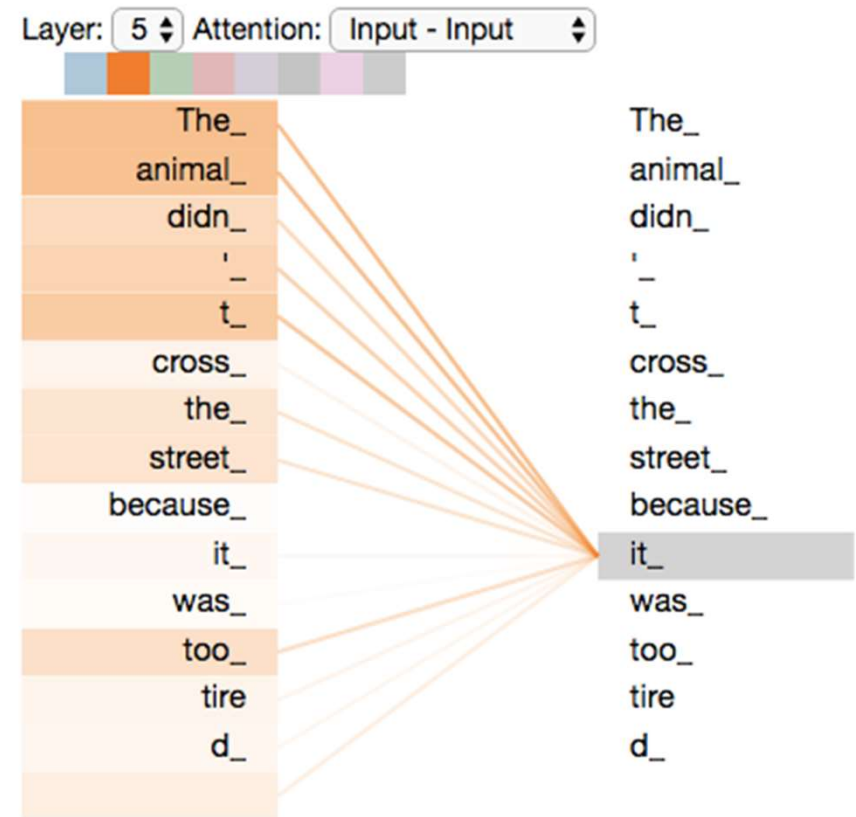
Transformers

¿Cómo funcionan los Transformers?

¿Cómo funciona la atención?

En la frase "The **animal** didn't cross the street because **it** was too tired", el modelo no sabe a priori que "it" se refiere a "The animal" → ¡Esto es lo que hace el bloque de **self-attention**!

Conforme el modelo procesa **cada palabra** (o cada posición en la secuencia de entrada), el módulo de self-attention le permite **mirar al resto de las posiciones** en la secuencia de entrada para **pistas** que le permitan **codificar mejor** el elemento **actual**.

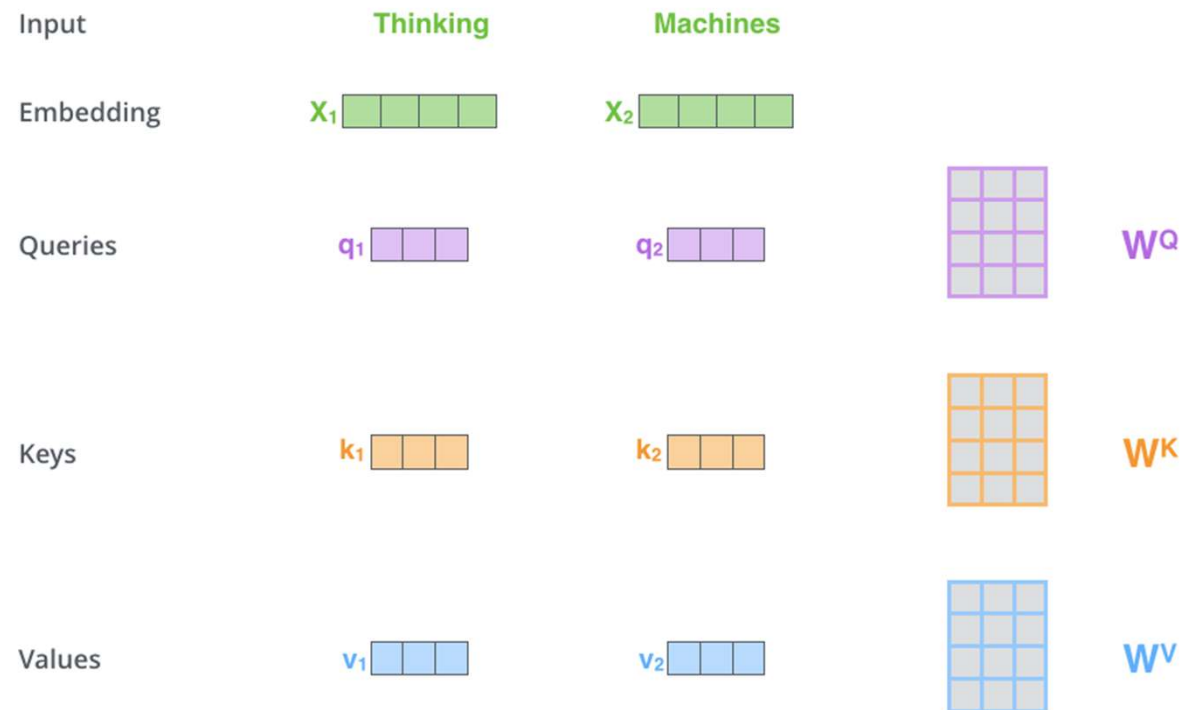


Transformers

¿Cómo funcionan los Transformers?

¿Cómo funciona el bloque de self-attention exactamente?

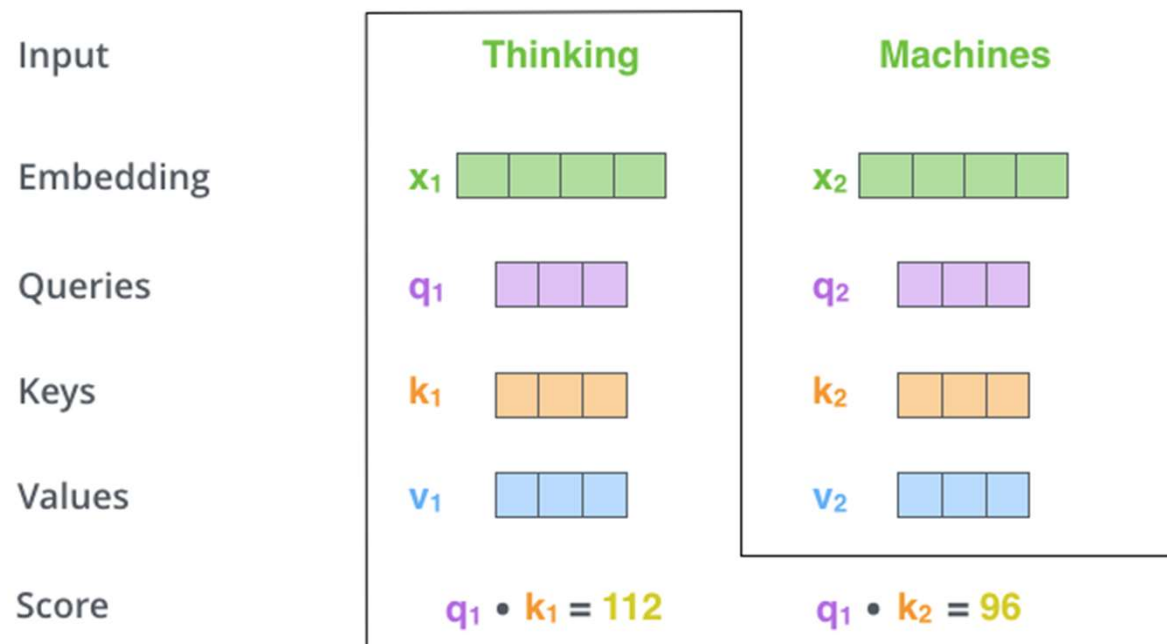
1. Creamos 3 matrices entrenables (W^Q , W^K , W^V)
2. Multiplicamos cada vector de entrada por cada una de las 3 matrices entrenables (W^Q , W^K , W^V)
3. Obtenemos 3 nuevos vectores (q_n , k_n , v_n), que además son de menor dimensionalidad que los de entrada, ya que las matrices se eligen así para reducir el coste computacional



Transformers

¿Cómo funcionan los Transformers?

4. Calculamos la self-attention como el producto escalar de $q_n \cdot k_n$.



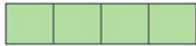
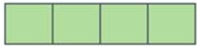


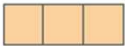
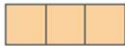
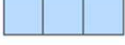
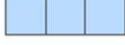
Transformers

¿Cómo funcionan los Transformers?

- Dividimos entre la raíz cuadrada del número de dimensiones escogidas para los vectores **key** (para estabilizar los gradientes durante el entrenamiento)
- Aplicamos la función **softmax** para normalizar los valores y que sumen 1.

¡Ya tenemos el valor de la atención!

Pero no hemos acabado...

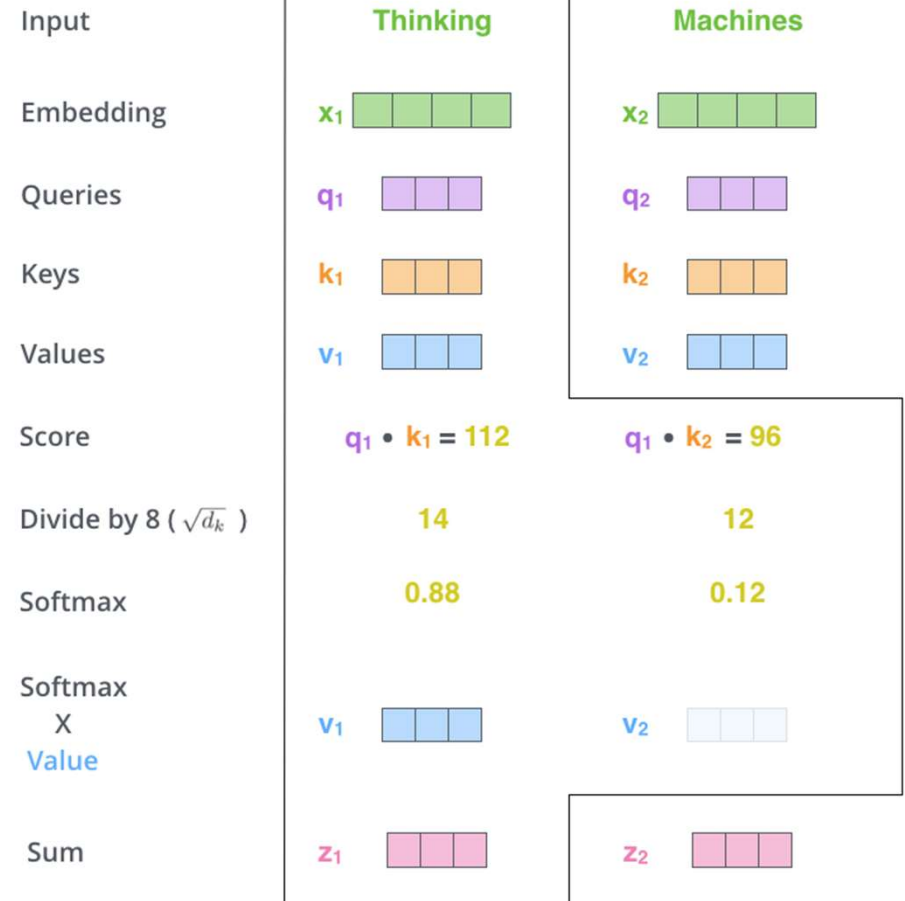
Input	Thinking	Machines
Embedding	x_1 	x_2 
Queries	q_1 	q_2 
Keys	k_1 	k_2 
Values	v_1 	v_2 
Score	$q_1 \cdot k_1 = 112$	$q_1 \cdot k_2 = 96$
Divide by 8 ($\sqrt{d_k}$)	14	12
Softmax	0.88	0.12

Transformers

¿Cómo funcionan los Transformers?

8. Multiplicamos el valor de la **atención** de cada vector de entrada (a_n) por el vector **value** v_1 (porque estamos calculando el valor de la atención para el vector de entrada 1)
9. Sumamos todos los resultados $a_n \cdot v_n$, obteniendo z_1

¡Y ahora sí que hemos acabado!

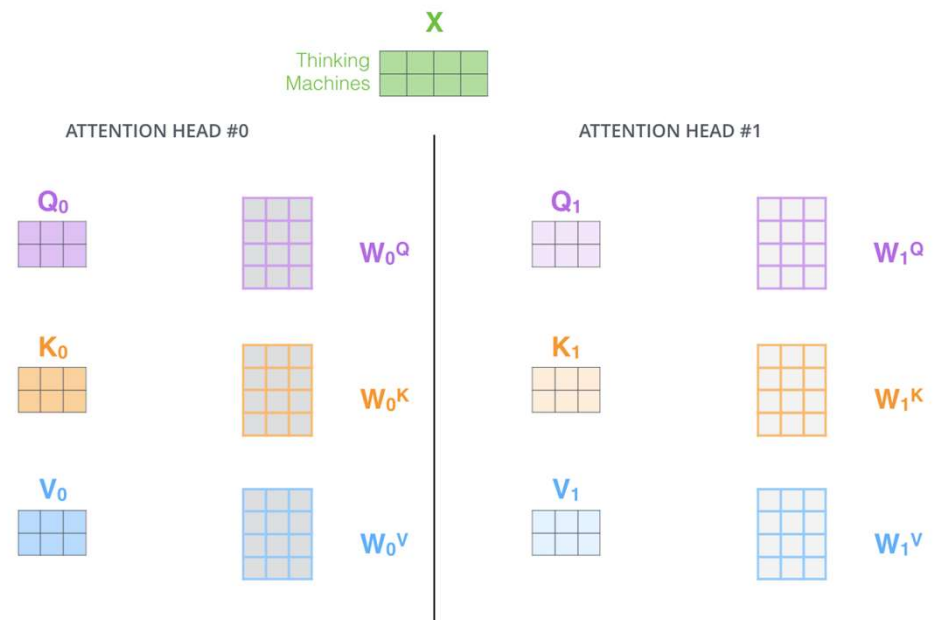
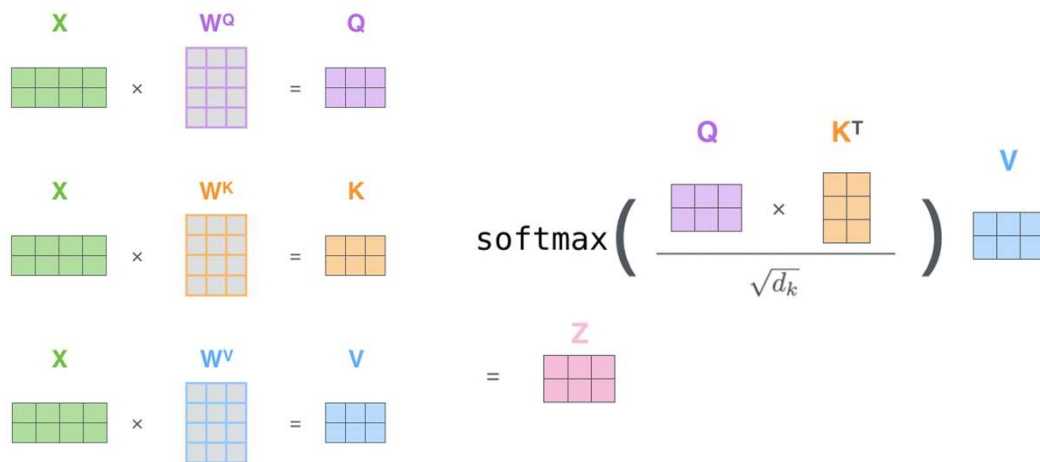


Transformers

¿Cómo funcionan los Transformers?

Solamente indicar que esto, en realidad, se hace de forma **matricial** para ir más rápido.

Y con más de un conjunto de W^Q, W^K, W^V (**multi-head**).



Transformers

¿Cómo funcionan los Transformers?

Resumen:

1) This is our input sentence*

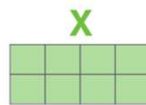
2) We embed each word*

3) Split into 8 heads. We multiply X or R with weight matrices

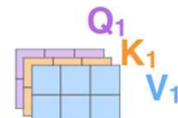
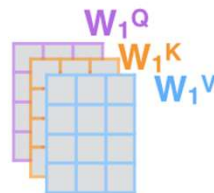
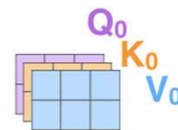
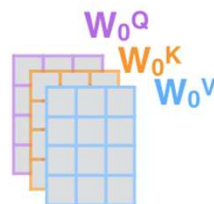
4) Calculate attention using the resulting $Q/K/V$ matrices

5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

Thinking
Machines



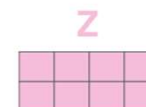
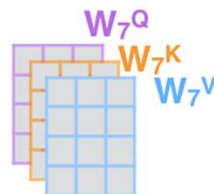
* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one



...

...

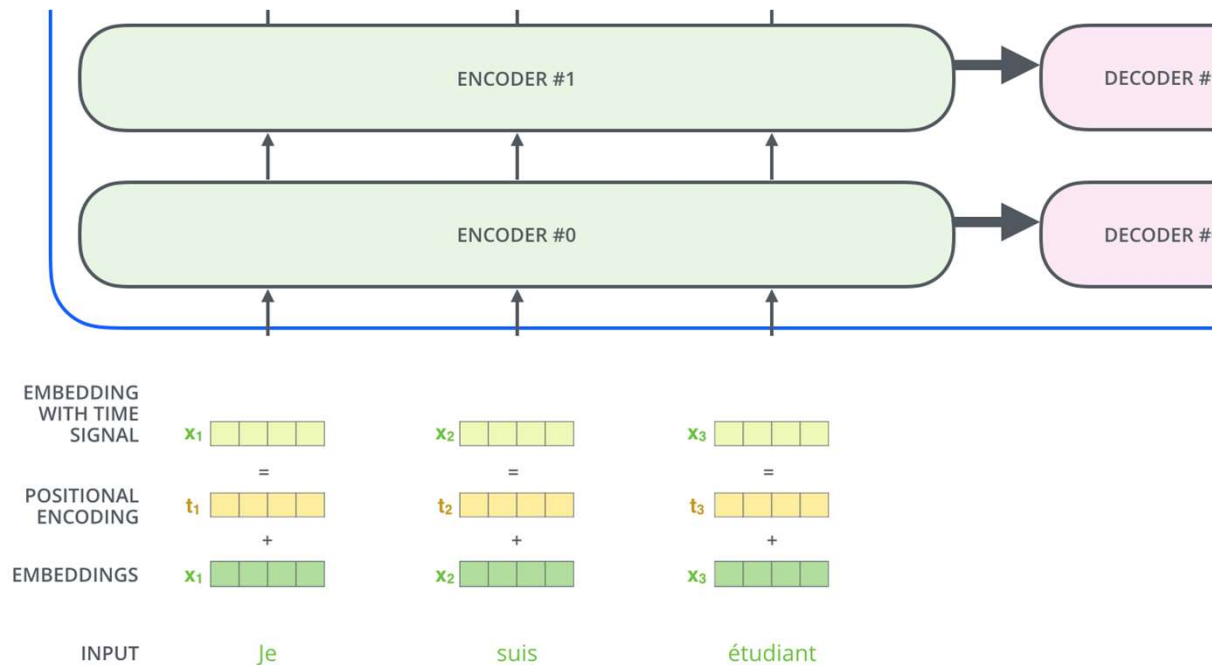
...



Transformers

¿Cómo funcionan los Transformers?

Y nos quedaría el concepto del positional embedding, que es lo que le indica a la red la posición de cada elemento de entrada en la secuencia completa.



Transformers

Ejemplo

[Transformer 1. Single step](#)

[Transformer 2. Multi-step](#)

[Ejemplo avanzado](#)

<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

Video recomendado explicando la intuición: <https://www.youtube.com/watch?v=UNmqTiOnRfg>

Para ampliar: <http://karpathy.github.io/2015/05/21/rnn-effectiveness/> (Video)