Aprendizaje profundo

Autoatención y arquitectura Transformer

Gibran Fuentes-Pineda

Noviembre 2023

Autoatención (1)

- Cada salida $\mathbf{y}^{[i]}$ es simplemente la suma ponderada de todas las entradas $\left(\mathbf{x}^{[1]}, \mathbf{x}^{[2]}, \ldots, \mathbf{x}^{[T]}\right)$ en la secuencia:

$$\mathbf{y}^{[i]} = \sum_{j=1}^{T} \alpha_{i,j} \cdot \mathbf{x}^{[j]}, \text{ donde } \sum_{j=1}^{T} \alpha_{i,j} = 1$$

- Cada valor de atención $\alpha_{i,j}$ se obtiene a partir de una función de la entrada $\mathbf{x}^{[j]}$ correspondiente a la salida $\mathbf{y}^{[i]}$ y cada entrada $\mathbf{x}^{[j]}$.

- Una función comúnmente utilizada es

$$\alpha_{i,j} = softmax(\mathbf{x}^{[i]\top} \mathbf{x}^{[j]})$$

Autoatención (2)

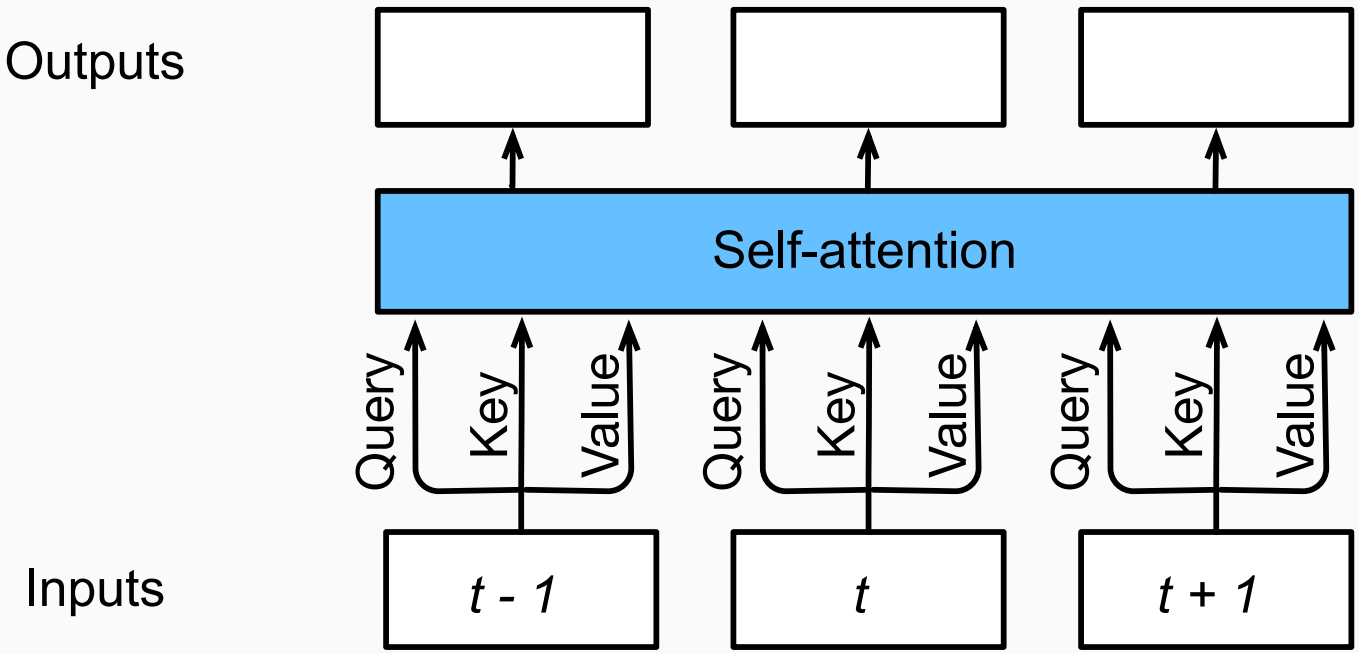

Figura tomada de Zhang et al. Dive into Deep Learning, 2022

Autoatención (3)

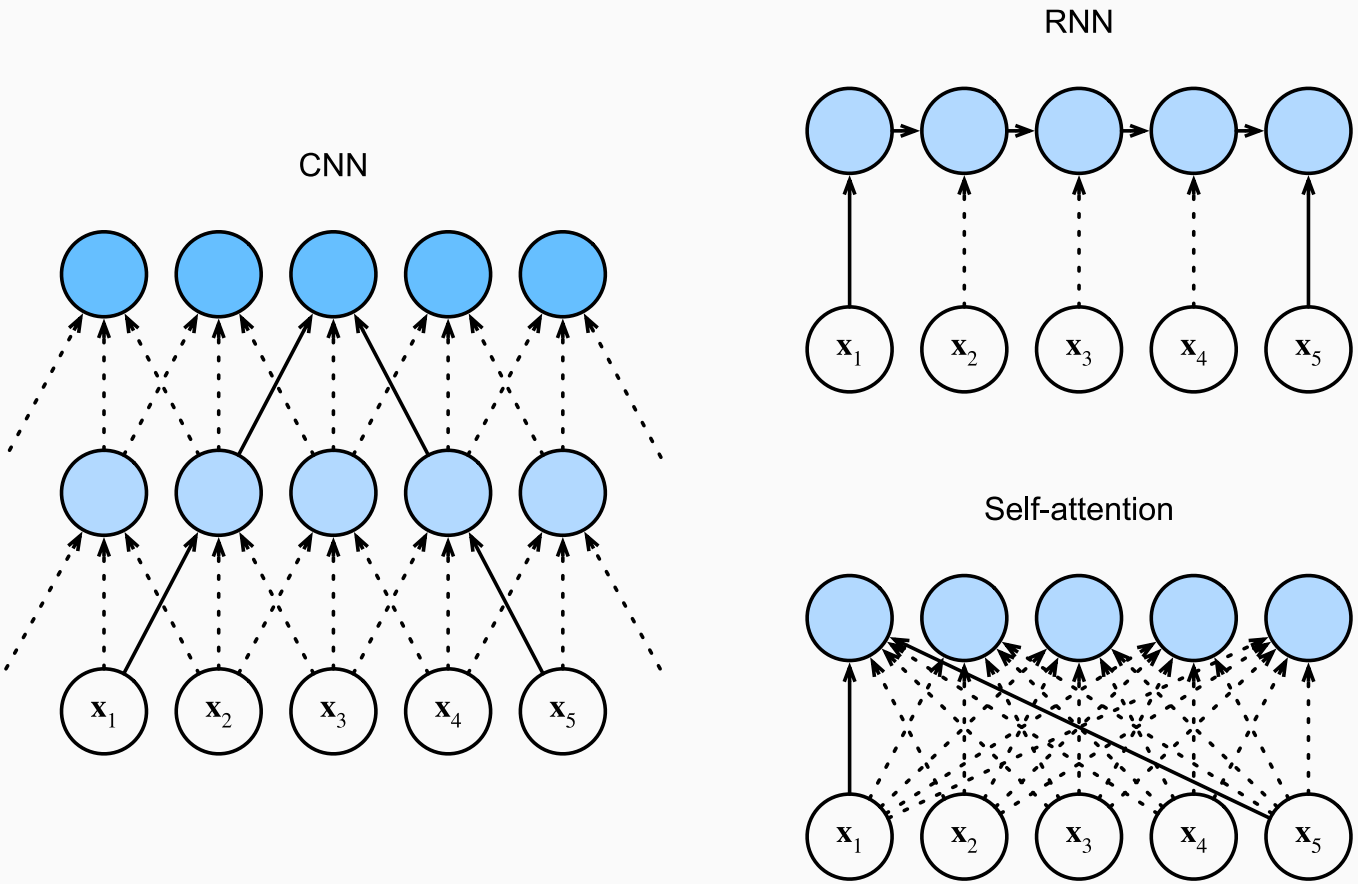

Figura tomada de Zhang et al. Dive into Deep Learning, 2022

Arquitectura *Transformer*

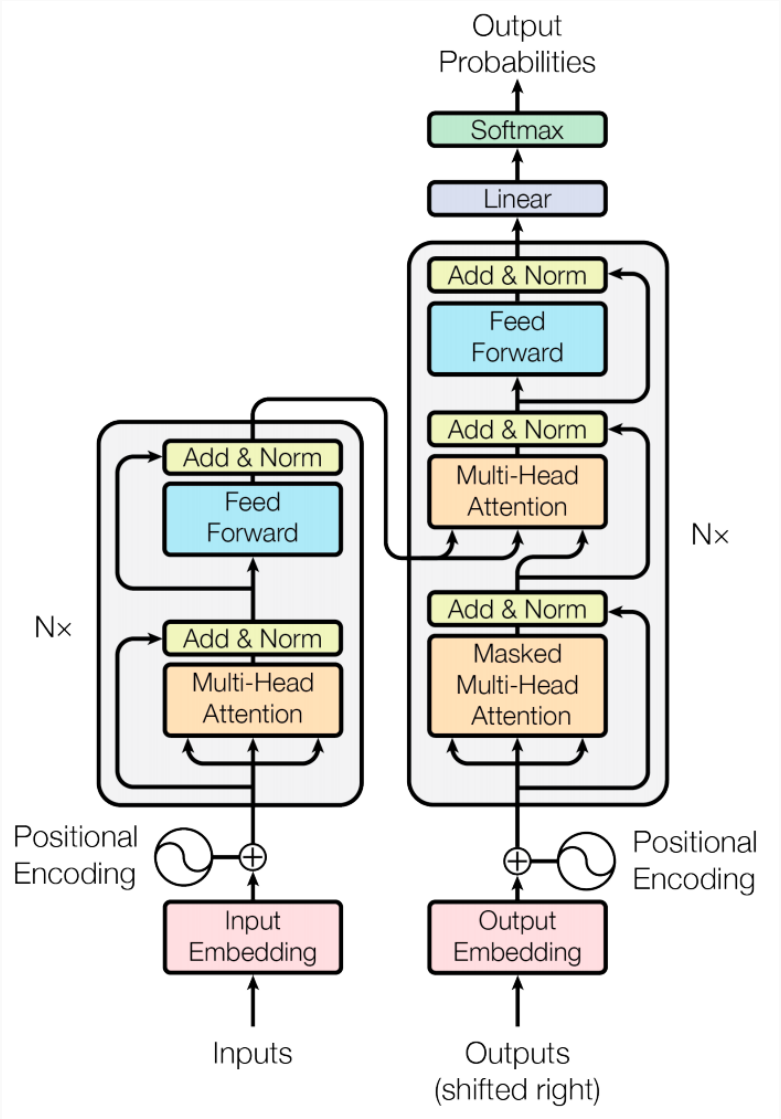

Imagen tomada de Vaswani et al. Attention Is All You Need, NIPS, 2017

Transformer: autoatención

- Se transforma linealmente cada entrada $\mathbf{x}^{[i]}$ a los vectores consulta ($\mathbf{q}^{[i]}$), llave ($\mathbf{k}^{[i]}$) y valor ($\mathbf{v}^{[i]}$)

$$\mathbf{q}^{[i]} = \mathbf{W}_q \mathbf{x}^{[i]}$$
$$\mathbf{k}^{[i]} = \mathbf{W}_k \mathbf{x}^{[i]}$$
$$\mathbf{v}^{[i]} = \mathbf{W}_v \mathbf{x}^{[i]}$$

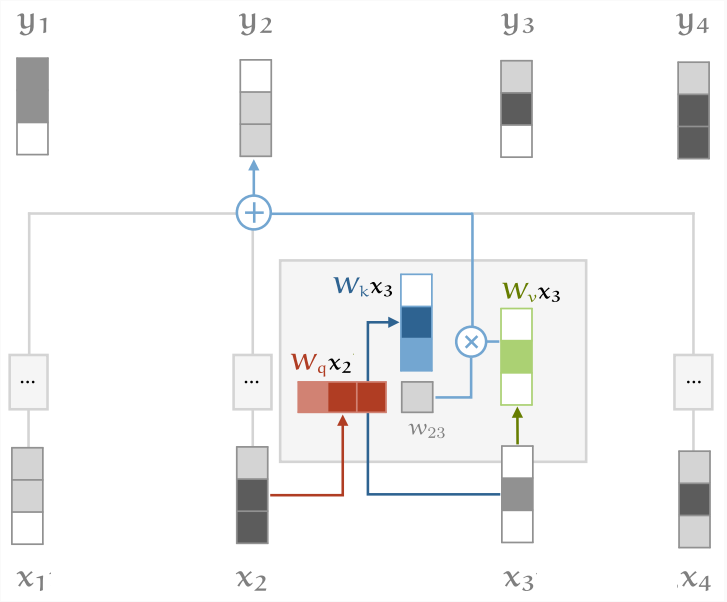

Imagen tomada de `http://www.peterbloem.nl/blog/transformers`

- Cada salida $\mathbf{y}^{[i]}$ es la suma de cada valor $\mathbf{v}^{[j]}$ ponderado por su valor de atención $\alpha_{i,j}$, esto es, $\mathbf{y}^{[i]} = \sum_{j=1}^{T} \alpha_{i,j} \cdot \mathbf{v}^{[j]}$

Transformer: producto punto normalizado

- Considerando que $\mathbf{q}^{[i]}, \mathbf{k}^{[i]} \in \mathbb{R}^{d_k}$ y $\mathbf{v}^{[i]} \in \mathbb{R}^{d_v}$, la función de puntaje está dada por el producto punto normalizado

$$puntaje(\mathbf{x}^{[i]}, \mathbf{x}_j) = \frac{\mathbf{q}^{[i]\top}\mathbf{k}^{[i]}}{\sqrt{d_k}}$$

- Por lo tanto, cada valor de atención $\alpha_{i,j}$ estaría dado por

$$\alpha_{i,j} = softmax\left(\frac{\mathbf{q}^{[i]\top}\mathbf{k}^{[i]}}{\sqrt{d_k}}\right)$$

Transformer: autoatención multicabeza

- Se transforma cada entrada con h distintos $\mathbf{W}_q$, $\mathbf{W}_k$ y $\mathbf{W}_v$ (cabezas) y se calcula la autoatención para cada una

- La concatenación de todas salidas se multiplica por la matriz de pesos $\mathbf{W}_o$ para producir la secuencia de salida

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = [\mathbf{Y}^1; \ldots; \mathbf{Y}^h] \cdot \mathbf{W}_o$$

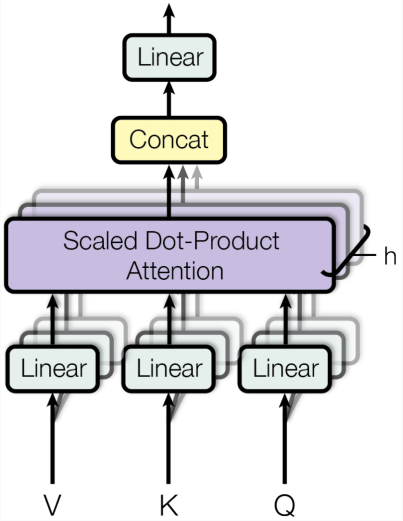

Imagen tomada de Vaswani et al. Attention Is All You Need, NIPS, 2017

Transformer: codificación posicional

- Para tomar en cuenta el orden en una secuencia, se codifica la posición de cada entrada.
- Vaswani et al. proponen funciones sinusoidales[1]

$$PE(pos, 2i) = sin\left[\frac{pos}{10000^{\left(\frac{2i}{d_{model}}\right)}}\right]$$

$$PE(pos, 2i+1) = cos\left[\frac{pos}{10000^{\left(\frac{2i}{d_{model}}\right)}}\right]$$

donde *pos* es la posición en la secuencia, i la dimensión del vector (*embedding*) de entrada y d_{model} su tamaño.

[1] También es posible aprender la codificación. Por ej. Gehring et al. *Convolutional Sequence to Sequence Learning*, arxiv:1705.03122, 2017.

Transformer: red hacia adelante por posición

- Las salidas de los bloques de autoatención se conectan a 2 capas densas, la primera con función de activación ReLU

$$FFN(\mathbf{X}^{(i)}) = \mathrm{máx}(0, \mathbf{x}_j^{(i)} \cdot \mathbf{W}^{\{1\}} + b^{\{1\}}) \cdot \mathbf{W}^{\{2\}} + b^{\{2\}}$$

- Esto se realiza de forma separada por cada entrada en la secuencia[2]

- Llamada red hacia adelante por posición o *Position-wise Feed-Forward Networks*

[2] Esto se puede ver como una convolución 1D con un filtro de tamaño 1.

Bloque tipo Transformer

- Generalmente compuestos de:
  1. Autoatención multicabeza
  2. Red hacia adelante por posición
- Con conexiones residuales y normalización por capa (*Layer Normalization*) en ambos

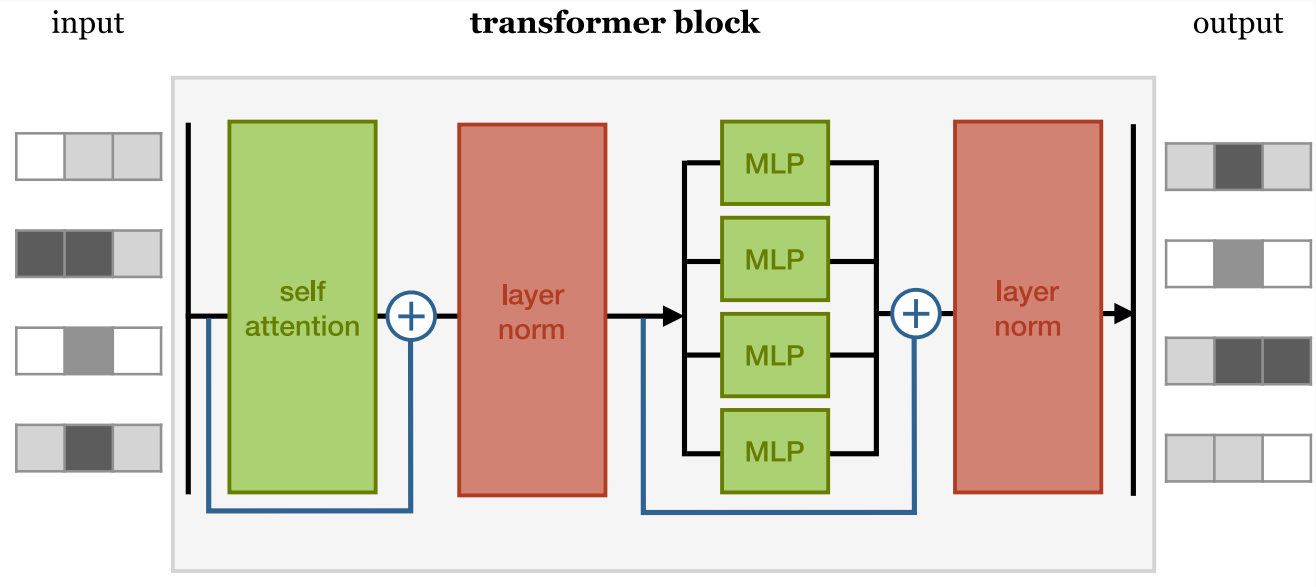

Imagen tomada de Peter Bloem. Transformers from scratch, 2019.

Arquitecturas Transformer: tarea de clasificación

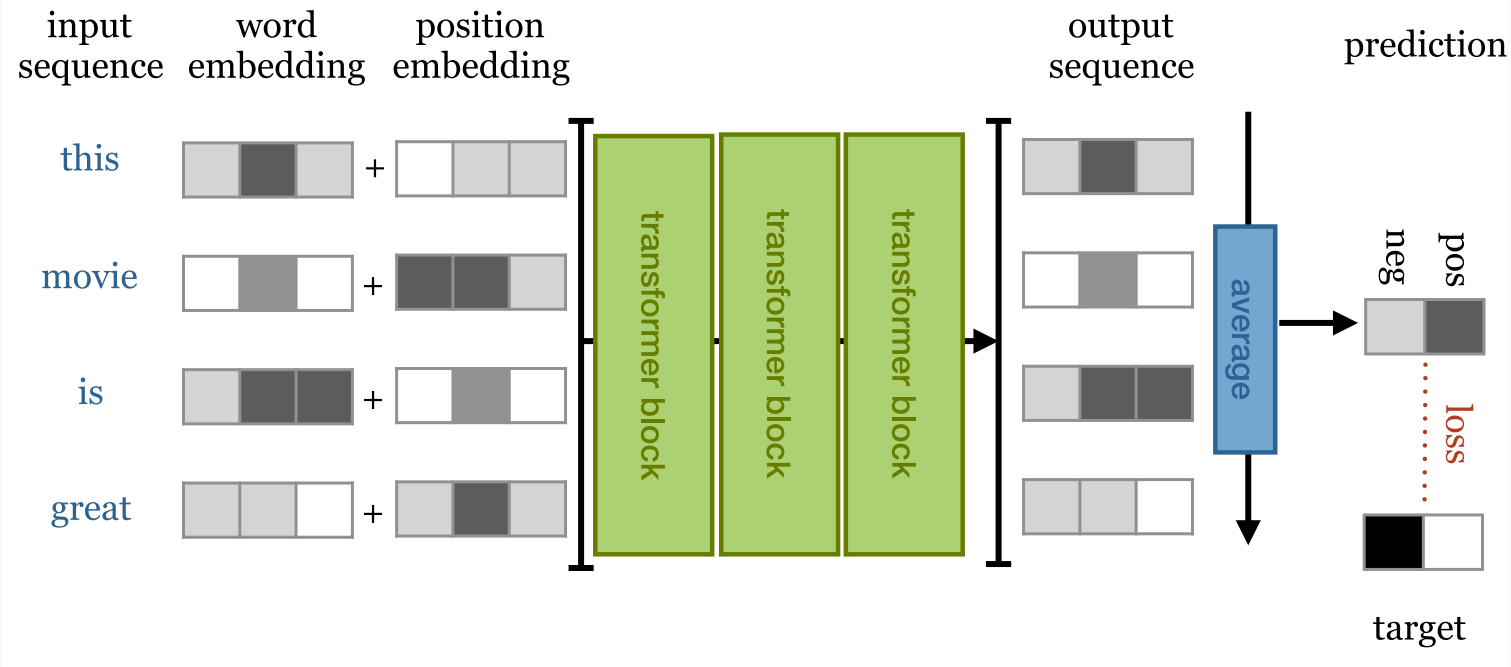

Imagen tomada de Peter Bloem. Transformers from scratch, 2019.

Arquitecturas Transformer: tarea de generación

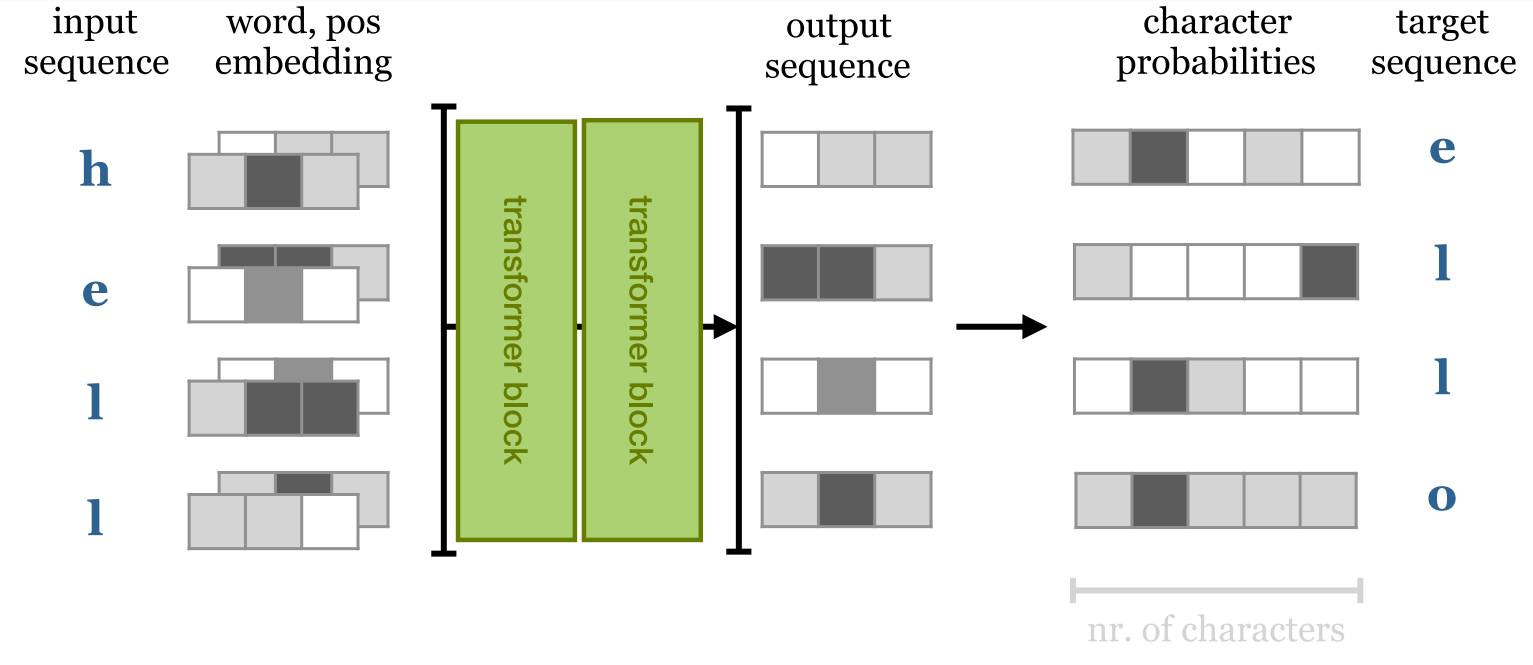

Imagen tomada de Peter Bloem. Transformers from scratch, 2019.

Transformer para generación: enmascaramiento

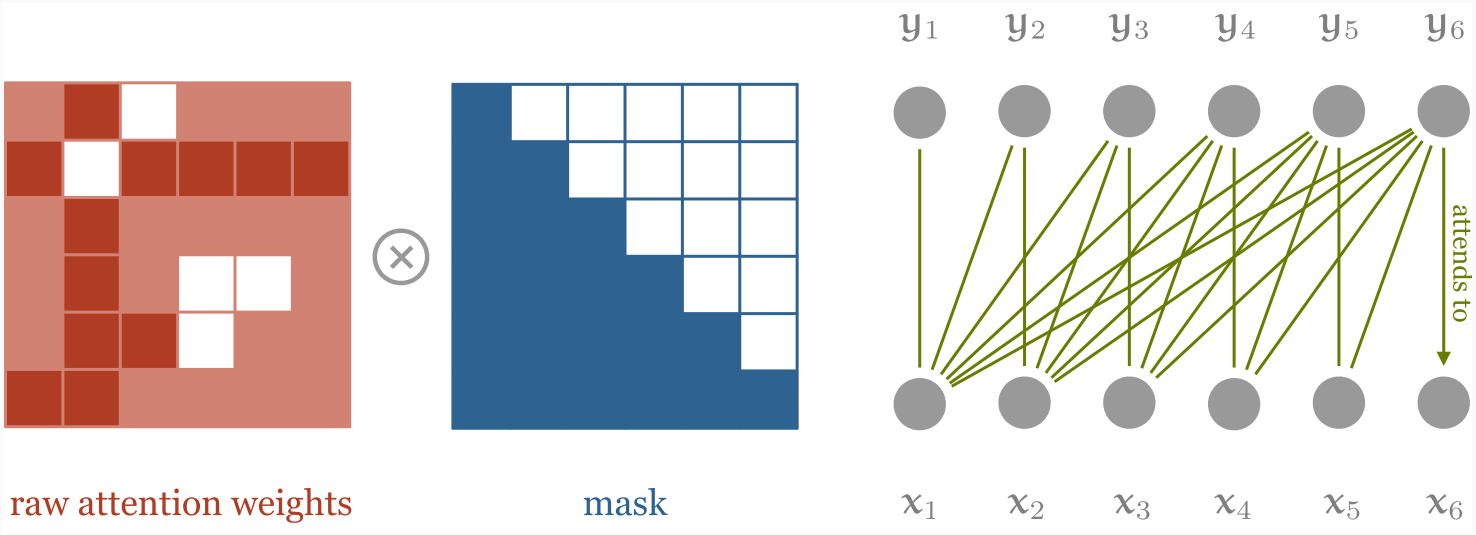

Imagen tomada de Peter Bloem. Transformers from scratch, 2019.