Mecanismo de atención; transformers; embeddings posicionales

Fernando Schiaffino
schiaffinofernando@gmail.com

Clase 2 Parte 1
Martes 22/04/2025

- Clase 1: Nociones básicas de redes neuronales para el procesamiento del lenguaje natural
  - Parte 1: Tokenización, embeddings estáticos
  - Parte 2: el perceptrón; arquitectura básica de redes neuronales; funciones de activación; backpropagation.
- **Clase 2: Grandes y pequeños modelos de lenguaje**.
  - **Parte 1: Mecanismo de atención; transformers; embeddings posicionales**
  - Parte 2: Similitudes y diferencias entre modelos de lenguaje
- Clase 3: *Prompt engineering*
  - Parte 1: Hiperparámetros de los modelos de lenguaje; prompting: Zero-Shot; Few-Shot
  - Parte 3: Prompting: RAG, Chain of Thought y otros

Presentación

Estructura y temas de la clase de hoy:

Un modelo de lenguaje es un modelo que predice la probabilidad de una secuencia de palabras o tokens. Es decir que es un modelo que va a predecir la probabilidad del próximo token dados los anteriores.

- Modelos simples basado en una arquitectura *feedforward*
- Modelos seq2seq u otros basados en RNN
- Grandes modelos de lenguaje basados en arquitecturas de transformers

Estos modelos tienen la ventaja de ser autosupervisados, ya que no es necesario anotar un valor de salida. El corpus de texto es al mismo tiempo el input y el output esperado, porque a cada paso t ya sabemos cuál es la siguiente palabra.

Modelos tradicionales

- **Cuello de botella**: La información de palabras lejanas se "diluía" o se perdía. Dificultad para manejar dependencias a larga distancia

Modelos tradicionales

- **Cuello de botella**: La información de palabras lejanas se "diluía" o se perdía. Dificultad para manejar dependencias a larga distancia
- El perro que perseguía al gato que vive en la otra cuadra [...] ladró. – *¿Quién ladró?*.

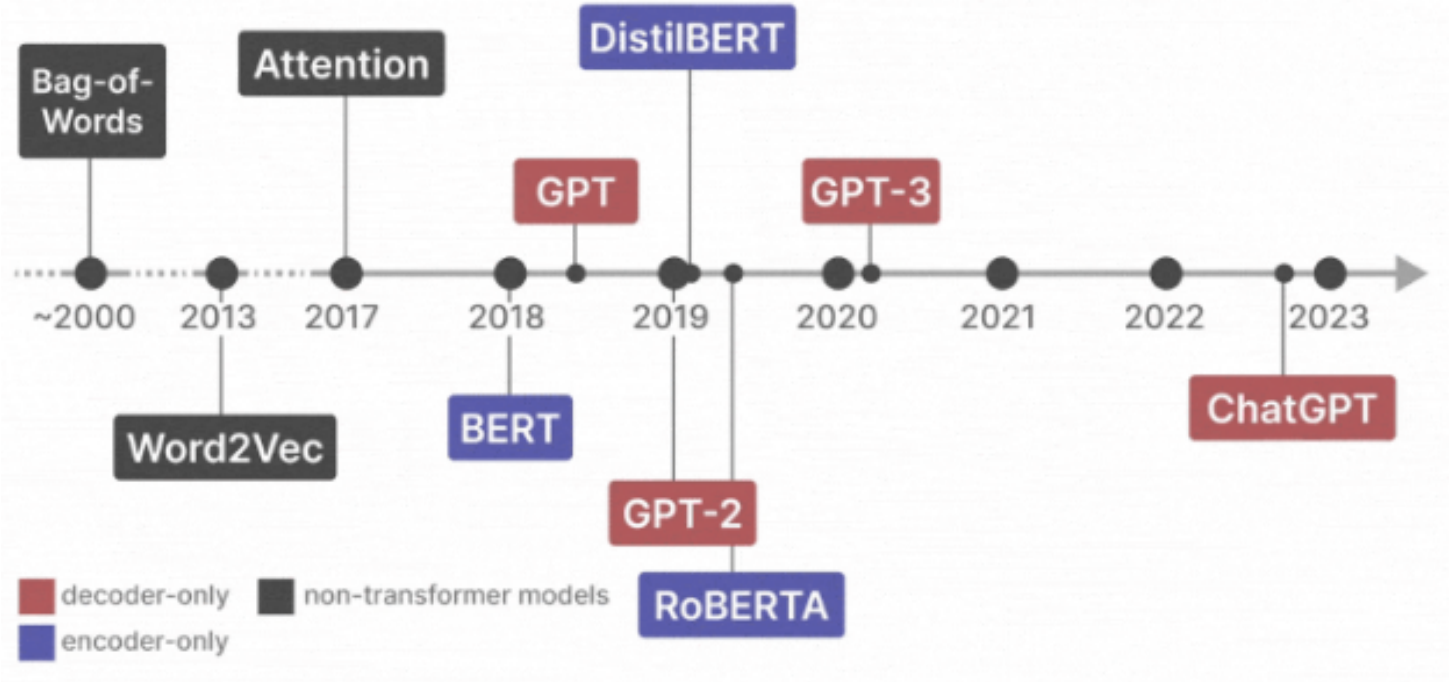
Modelos tradicionales

- **Cuello de botella**: La información de palabras lejanas se "diluía" o se perdía. Dificultad para manejar dependencias a larga distancia
- El perro que perseguía al gato que vive en la otra cuadra [...] ladró. – *¿Quién ladró?*.
- **Procesamiento secuencial**: lento y difícil de acelerar en hardware moderno (GPUs).

Modelos tradicionales

- **Cuello de botella**: La información de palabras lejanas se "diluía" o se perdía. Dificultad para manejar dependencias a larga distancia
- El perro que perseguía al gato que vive en la otra cuadra [...] ladró. – *¿Quién ladró?*.
- **Procesamiento secuencial**: lento y difícil de acelerar en hardware moderno (GPUs).
- **Problema de los embeddings fijos**: en modelos tradicionales, cada palabra tiene un único embedding sin importar el contexto.

Modelos de Lenguaje



El Transformer

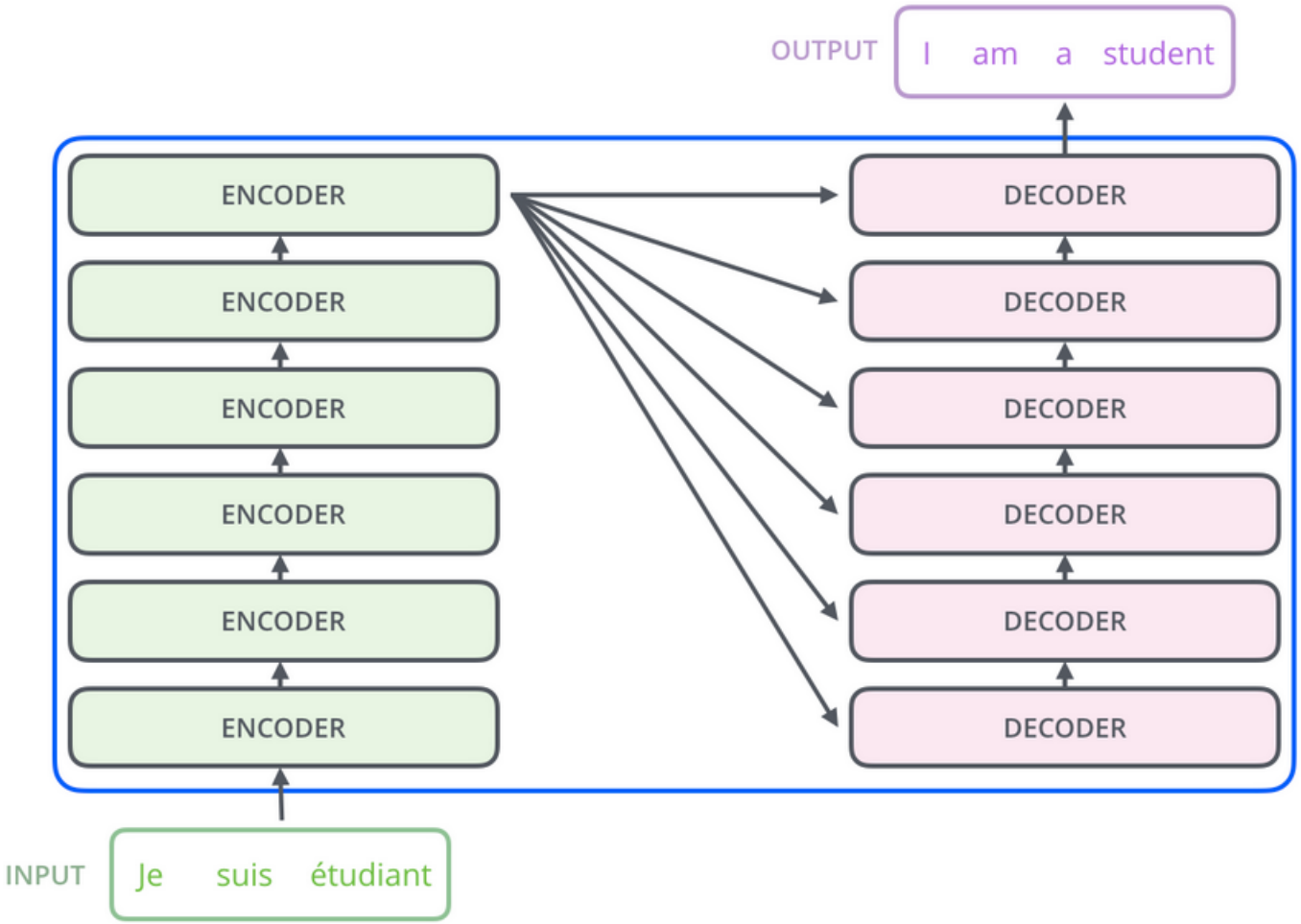
> *The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely.*

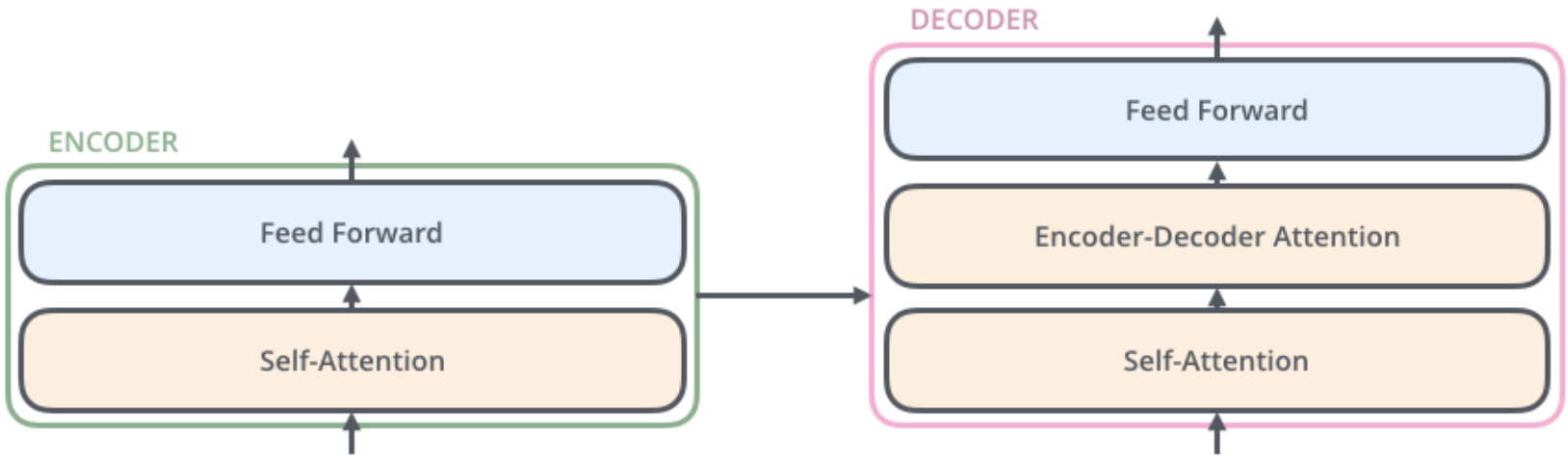
Vaswani *et al.* (2017) proponen una arquitectura de aprendizaje basada en modelos Transformers y mecanismos de atención.

Esta innovación sentó las bases para toda una nueva generación de Modelos de Lenguaje.

La arquitectura de transformers puede separarse en:

- Bloques Encoder/Decoder apilados
- Mecanismos de atención
- Encoding posicional


OUTPUT
I am a student
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
ENCODER
DECODER
DECODER
DECODER
DECODER
DECODER
DECODER
INPUT
Je suis étudiant


ENCODER
Feed Forward
Self-Attention
DECODER
Feed Forward
Encoder-Decoder Attention
Self-Attention

¿Qué es la Atención?

- Cuando leemos, no damos la misma importancia a cada palabra. Nuestros ojos (y cerebro) saltan, se enfocan en palabras clave, conectan pronombres con sus referentes, etc.
- En 'María vio a Juan. Ella lo saludó.', para entender 'Ella' y 'lo', atendemos específicamente a 'María' y 'Juan'.

¿Qué es la Atención?

- Cuando leemos, no damos la misma importancia a cada palabra. Nuestros ojos (y cerebro) saltan, se enfocan en palabras clave, conectan pronombres con sus referentes, etc.
- En 'María vio a Juan. Ella lo saludó.', para entender 'Ella' y 'lo', atendemos específicamente a 'María' y 'Juan'.

El mecanismo de atención permite al modelo, para cada palabra que está procesando, ponderar la importancia de todas las demás palabras en la secuencia (incluida ella misma) y extraer información de las más relevantes.

Mecanismo de Atención

- Para cada palabra, se calcula cuánto debe prestar atención a las demás (incluyéndose a sí misma).

Mecanismo de Atención

- Para cada palabra, se calcula cuánto debe prestar atención a las demás (incluyéndose a sí misma).
- Esto se realiza mediante el mecanismo de *self-attention*:
  - **Consulta (Query)**: ¿Qué información necesito para esta palabra?
  - **Claves (Keys)**: ¿Qué información ofrecen las demás palabras?
  - **Valores (Values)**: ¿Qué información voy a usar finalmente?

Mecanismo de Atención

- Para cada palabra, se calcula cuánto debe prestar atención a las demás (incluyéndose a sí misma).
- Esto se realiza mediante el mecanismo de *self-attention*:
  - **Consulta (Query)**: ¿Qué información necesito para esta palabra?
  - **Claves (Keys)**: ¿Qué información ofrecen las demás palabras?
  - **Valores (Values)**: ¿Qué información voy a usar finalmente?
- Para la palabra en la posición i, se compara su consulta con las claves de todas las posiciones j (incluida i) para obtener un peso de atención α_{ij}.

Mecanismo de Atención

- Para cada palabra, se calcula cuánto debe prestar atención a las demás (incluyéndose a sí misma).
- Esto se realiza mediante el mecanismo de *self-attention*:
  - **Consulta (Query)**: ¿Qué información necesito para esta palabra?
  - **Claves (Keys)**: ¿Qué información ofrecen las demás palabras?
  - **Valores (Values)**: ¿Qué información voy a usar finalmente?
- Para la palabra en la posición i, se compara su consulta con las claves de todas las posiciones j (incluida i) para obtener un peso de atención α_{ij}.
- Luego, se suman los valores de todas las posiciones ponderados por dichos pesos:

$$\text{output}_i = \sum_{j=1}^{T} \alpha_{ij} \cdot \text{value}_j$$

Esto permite capturar dependencias internas, contexto, relaciones sintácticas y semánticas dentro de una misma frase u oración.

¿Cómo Funciona la Atención?

La Pregunta (Query - Q): La palabra actual *Ella* tiene una pregunta: *¿A quién me refiero?*

Las Etiquetas (Keys - K): Cada palabra en la secuencia *María*, *vio*, *a*, *Juan*, . tiene una *etiqueta* que describe su contenido o rol potencial (p.ej., *Sustantivo femenino*, *Verbo*, *Preposición*, *Sustantivo masculino*, *Puntuación*).

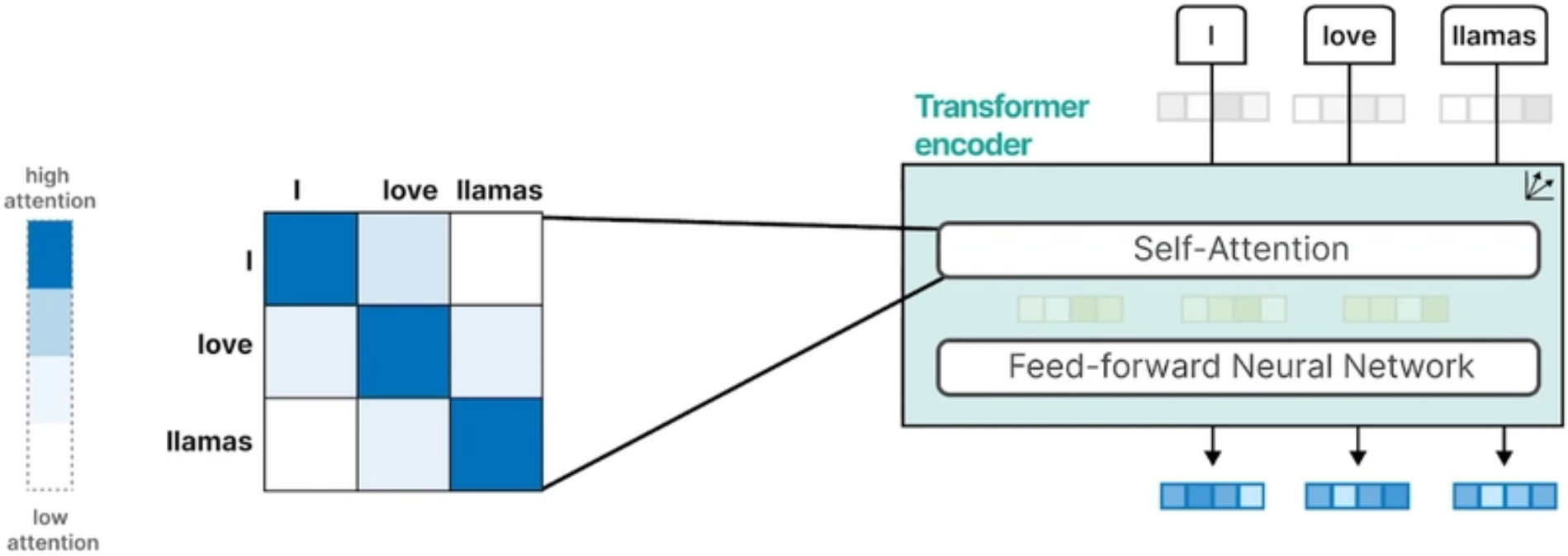
La Comparación: La *pregunta* de *Ella* (Query) se compara con todas las *etiquetas* (Keys).

La etiqueta que mejor *encaja* con la pregunta (la de *María*) obtiene una puntuación alta.

¿Cómo Funciona la Atención? (Parte 2)

Las Respuestas (Values - V): Cada palabra también tiene un *valor* o *contenido* (una representación de su significado en contexto).

El Resultado: Las puntuaciones de la comparación se usan para hacer un promedio ponderado de los *valores* (Values). La palabra *Ella* obtiene una nueva representación que está fuertemente influenciada por el *valor* de *María* (porque su Key coincidió bien con la Query de *Ella*), y menos influenciada por las otras palabras.


high attention
low attention
I
love
llamas
I
love
llamas
I
love
llamas
Transformer encoder
Self-Attention
Feed-forward Neural Network

BERT

BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks.

(Devlin et al., 2019)

BERT

> *BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks.*
>
> *(Devlin et al., 2019)*

- BERT: ***B**idirectional **E**ncoder **R**epresentation **T**ransformer*

BERT

> *BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks.*
>
> *(Devlin et al., 2019)*

- BERT: ***B**idirectional **E**ncoder **R**epresentation **T**ransformer*
- Modelos **preentrenados** en grandes corpus de datos con el fin de aprender a relacionar secuencias y abstraer nociones generales sobre el funcionamiento del lenguaje.

BERT

> *BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks.*
>
> *(Devlin et al., 2019)*

- BERT: ***B**idirectional **E**ncoder **R**epresentation **T**ransformer*
- Modelos **preentrenados** en grandes corpus de datos con el fin de aprender a relacionar secuencias y abstraer nociones generales sobre el funcionamiento del lenguaje.
- Modelos **preentrenados** que podríamos luego *ajustar* o reentrenar en tareas específcas.

BERT

> *BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be finetuned with just one additional output layer to create state-of-the-art models for a wide range of tasks.*
>
> *(Devlin et al., 2019)*

- BERT: ***B**idirectional **E**ncoder **R**epresentation **T**ransformer*
- Modelos **preentrenados** en grandes corpus de datos con el fin de aprender a relacionar secuencias y abstraer nociones generales sobre el funcionamiento del lenguaje.
- Modelos **preentrenados** que podríamos luego *ajustar* o reentrenar en tareas específcas.
  - *Fine-tuning*
  - Clasificación de textos, Análisis de Sentimiento, etc

BERT

Esta familia de modelos, fue entrenada con el objetivo de resolver dos tipos de problemas o tareas:

- Next Sentence Prediction
  - Dadas dos secuencias, ¿la segunda es continuación de la primera?

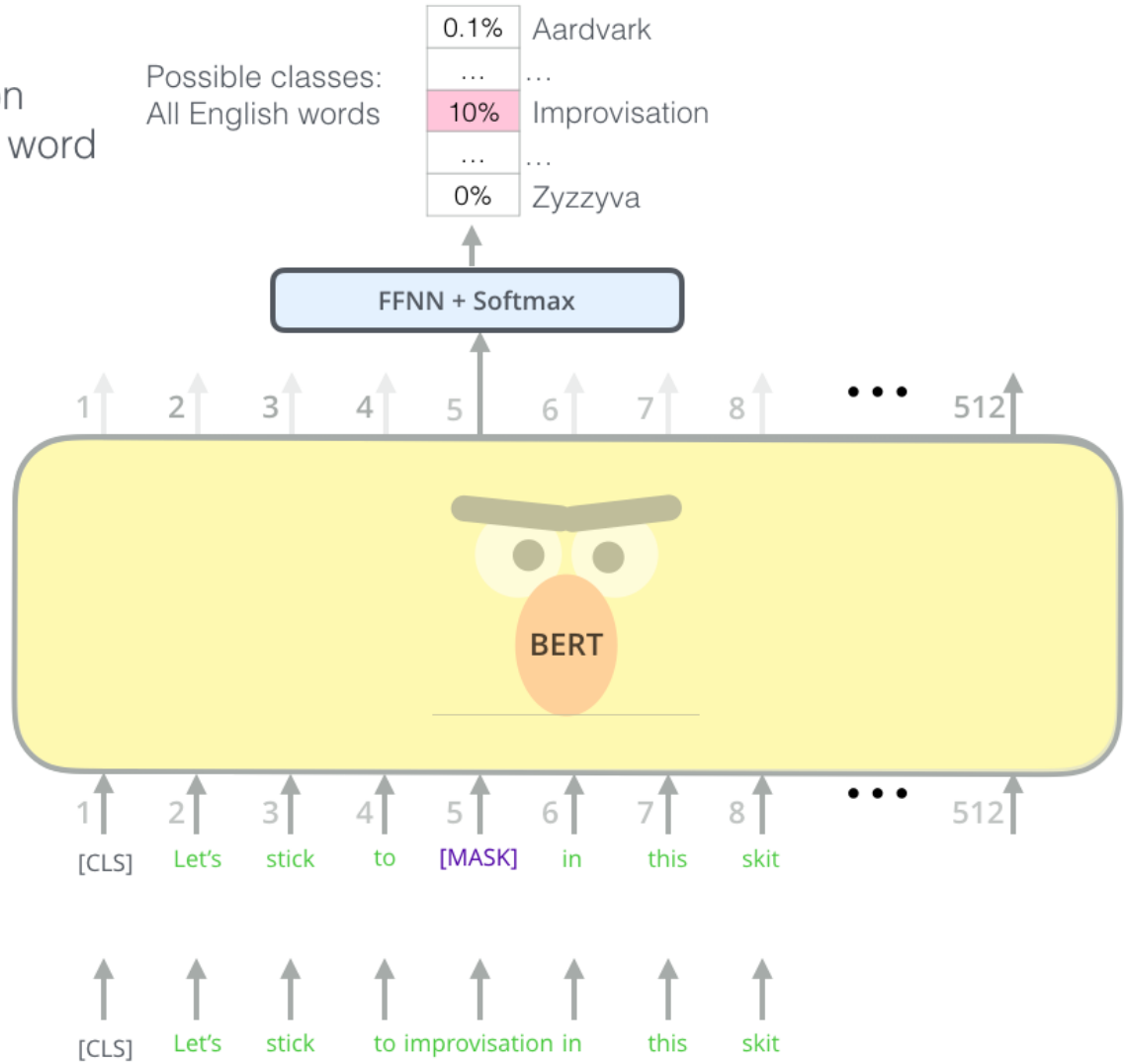
BERT

Esta familia de modelos, fue entrenada con el objetivo de resolver dos tipos de problemas o tareas:

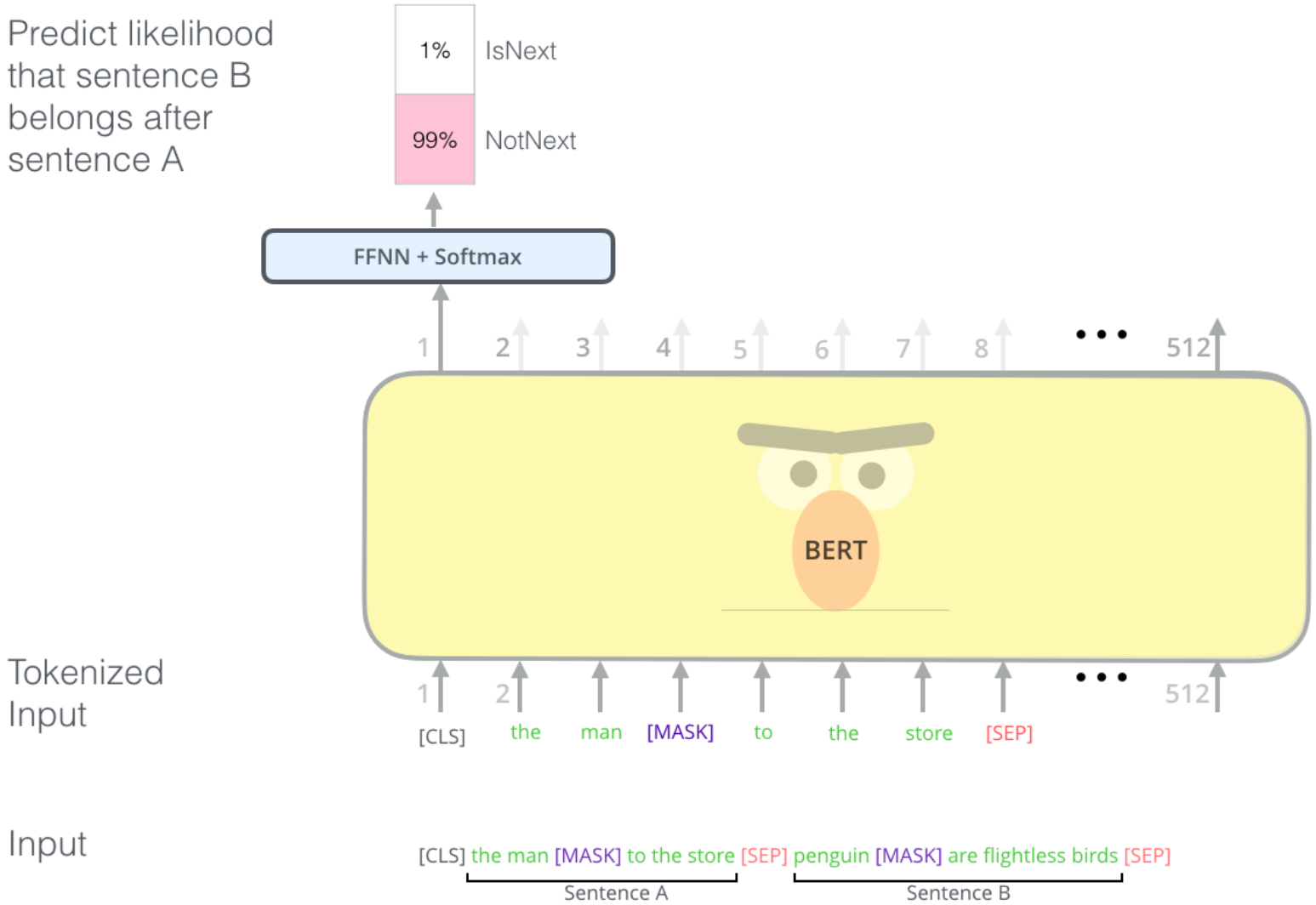
- Next Sentence Prediction
  - Dadas dos secuencias, ¿la segunda es continuación de la primera?
- Masked Language Modeling
  - Dado un contexto de palabras, ¿cuál es la palabra que falta?

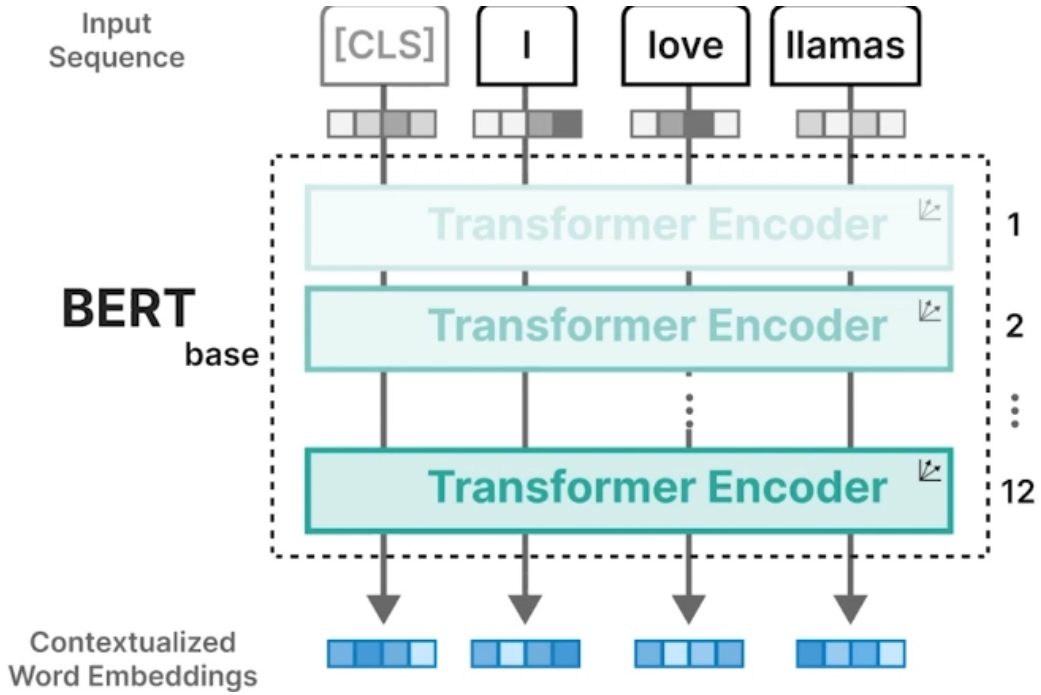
BERT: MLM

Use the output of the masked word's position to predict the masked word



BERT: NSP


Predict likelihood that sentence B belongs after sentence A
1%
IsNext
99%
NotNext
FFNN + Softmax
1
2
3
4
5
6
7
8
512
BERT
Tokenized Input
[CLS]
the
man
[MASK]
to
the
store
[SEP]
Input
[CLS] the man [MASK] to the store [SEP] penguin [MASK] are flightless birds [SEP]
Sentence A
Sentence B


Input Sequence
[CLS]
I
love
llamas
BERT
base
Transformer Encoder
Transformer Encoder
Transformer Encoder
1
2
12
Contextualized Word Embeddings

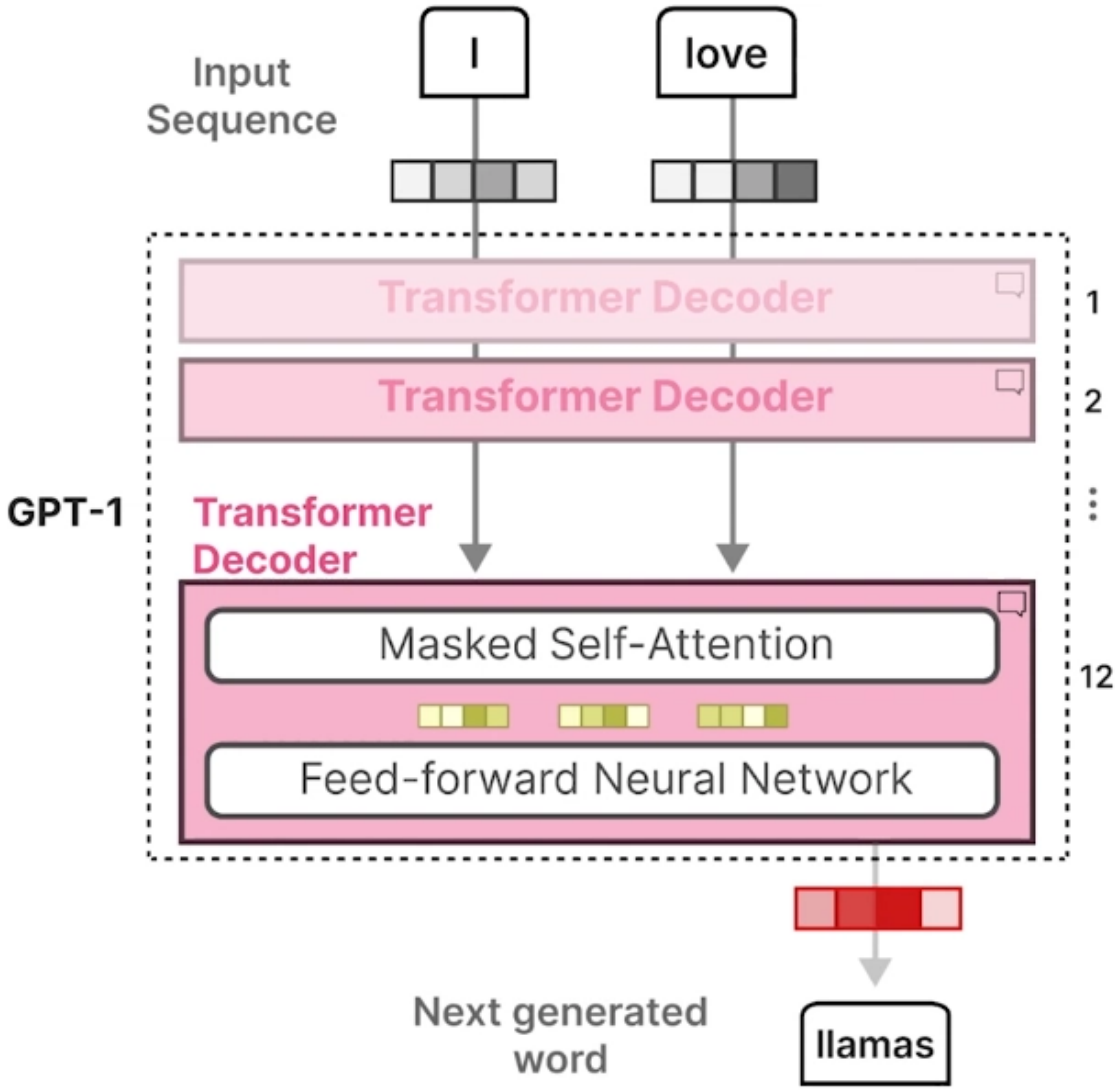
Generative Pre-trained Transformer

Estos modelos son Only-Decoder models.
GPT-2 se entrena con el objetivo de **modelado de lenguaje autoregresivo**: dado un prefijo de tokens $x_{1:t-1}$, el modelo maximiza la probabilidad del siguiente token x_t.

Formalmente, la función de pérdida se define como:

$$\mathcal{L} = -\sum_{t=1}^{T} \log P(x_t \mid x_{1:t-1}; \theta)$$

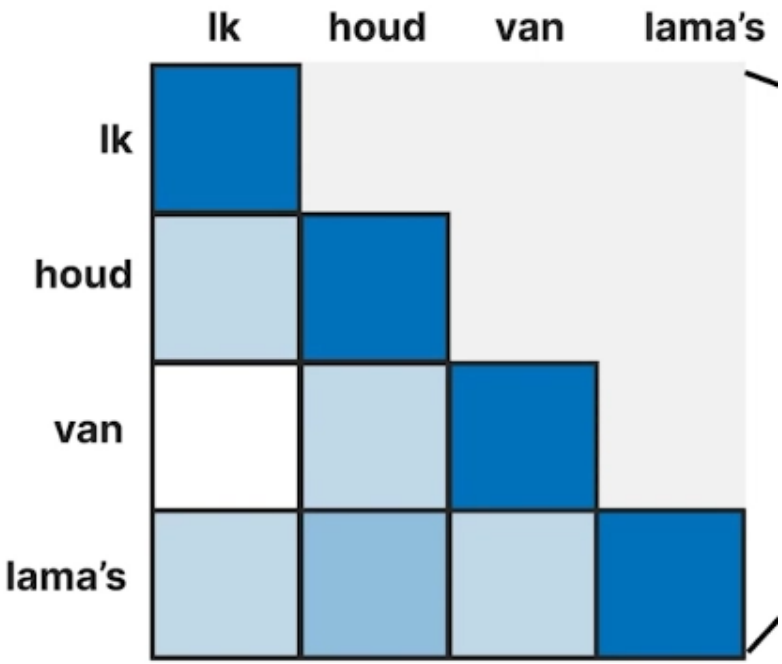
Esto corresponde a la **pérdida de entropía cruzada** entre la distribución predicha por el modelo y el token real.



Los embeddings de input se inicializan random y van pasando por una pila de decoders.

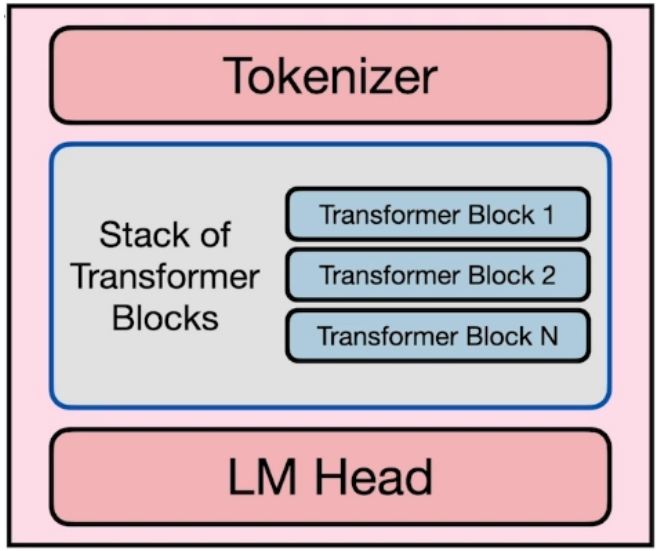
Masked Self-Attention

Durante el entrenamiento, o pre-entrenamiento, de este modelo only-decoder, se utiliza masked-self-Attention. Así la palabra solo 'atiende' a aquellas que la preceden en la secuencia y se entrena para predecir la siguiente.



Large Language Models

- Modelos generalistas, preentrenados para producir embeddings contextuales y/o abstracciones
- Modelos generadores, preentrenados para predecir las probabilidades del siguiente token en la secuencia
- Surge la idea de apilar además de encoders y decoders, bloques transformers enteros uno sobre otro. Donde cada uno se encargará de algun aspecto relevante para todo el proceso.



Bibliografía I

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