Procesamiento de Lenguaje Natural Avanzado

Aprendizaje por transferencia y modelado del lenguaje

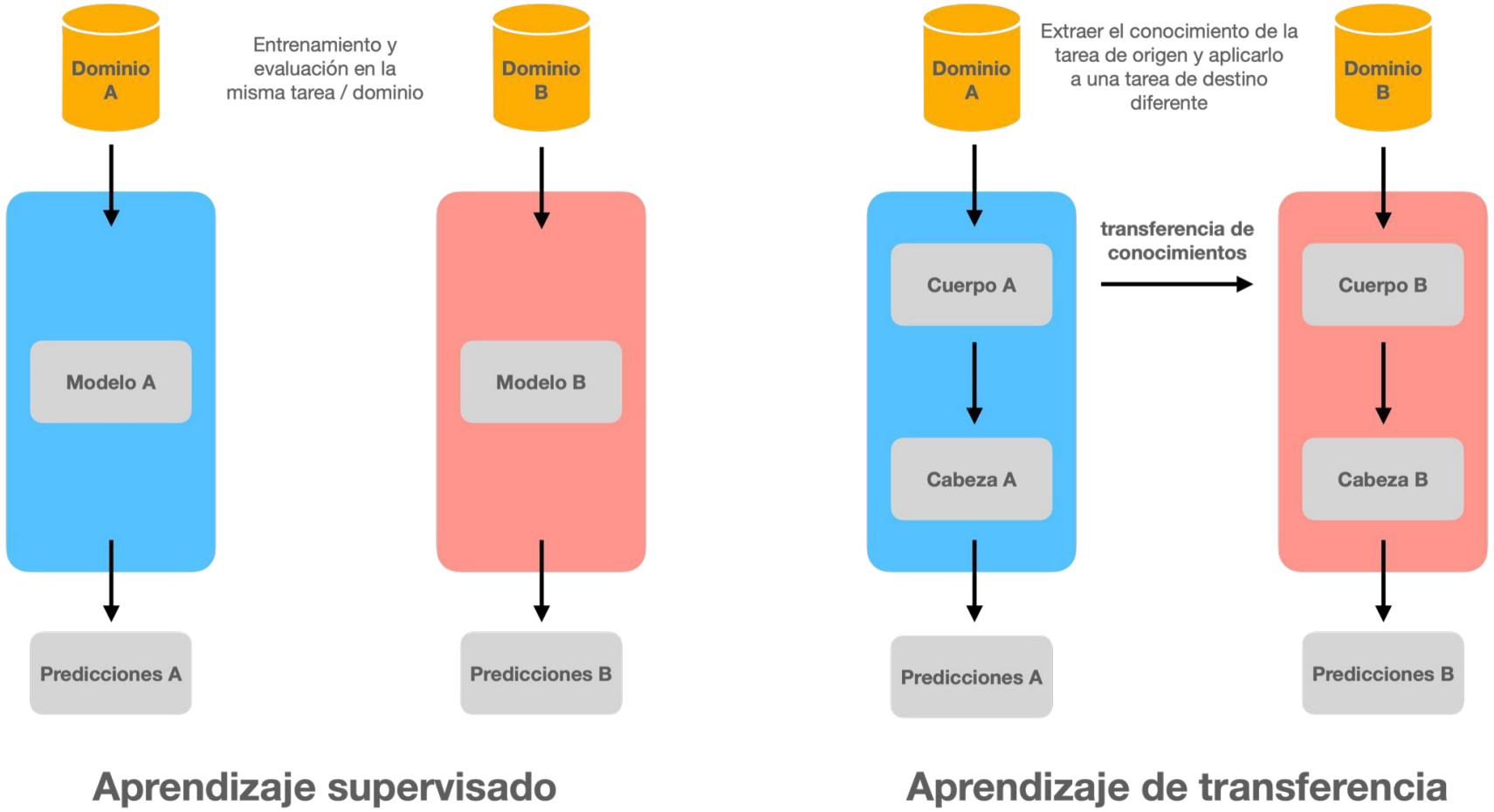
Dra. Helena Gómez Adorno
helena.gomez@iimas.unam.mx

Dr. Fazlourrahman Balouchzahi
fbalouc@iimas.unam.mx

Correo del curso:
pln.cienciadedatos@gmail.com


3:51
Gatorade

Es una forma de aprovechar un modelo entrenado para una tarea, en otra tarea que probablemente no tenga tantos datos disponibles.

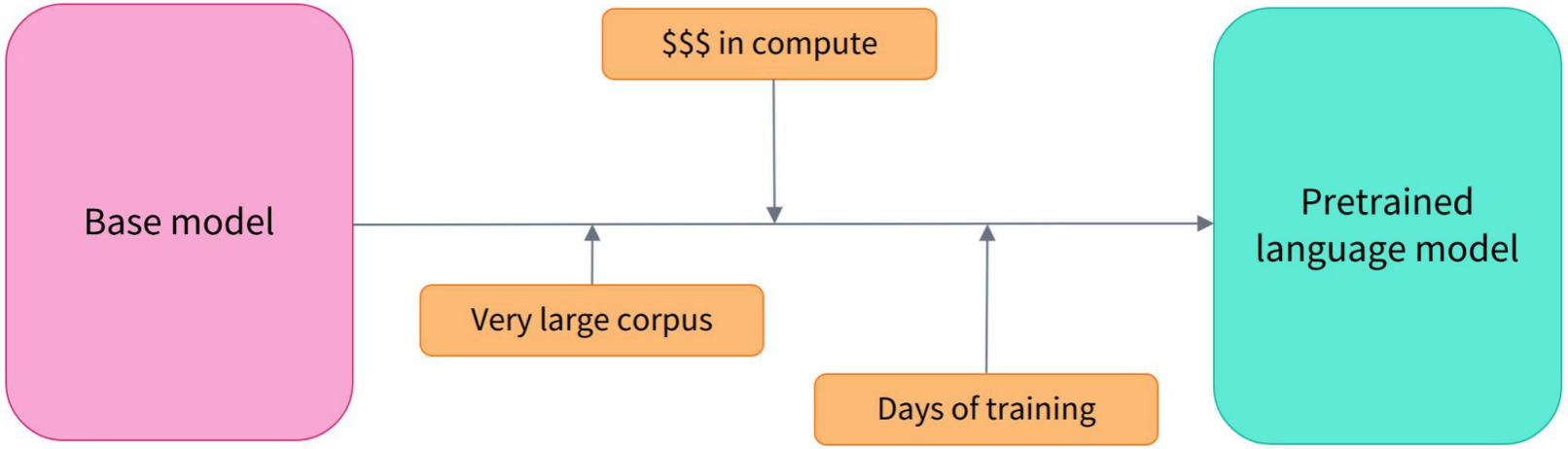


Un modelo **preentrenado** se puede aprovechar de las siguientes formas:

- **Extracción de características.** Se aprovecha el modelo para obtener una representación de los datos y estos se utilizan en un modelo para resolver una tarea específica.

- **Ajuste fino.** Se ajustan los pesos del modelo preentrenado para adaptarse a la tarea específica. Normalmente se utiliza una tasa de aprendizaje muy pequeña y apenas unas pocas épocas.

- En este paso entrenamos un modelo desde cero
- Requiere muchos datos y mucha computación
- Tras el entrenamiento, los “pesos” se reutilizan en diversas tareas (clasificación de textos etc.)



Aprendizaje Autosupervisado

- Algunos modelos de lenguaje preentrenados no fueron creados para solucionar una tarea particular, si no que su intención es crear un modelo para ser reutilizado en otras tareas.

- En PLN se utilizan comúnmente tareas pretexto para entrenar modelos de este tipo.

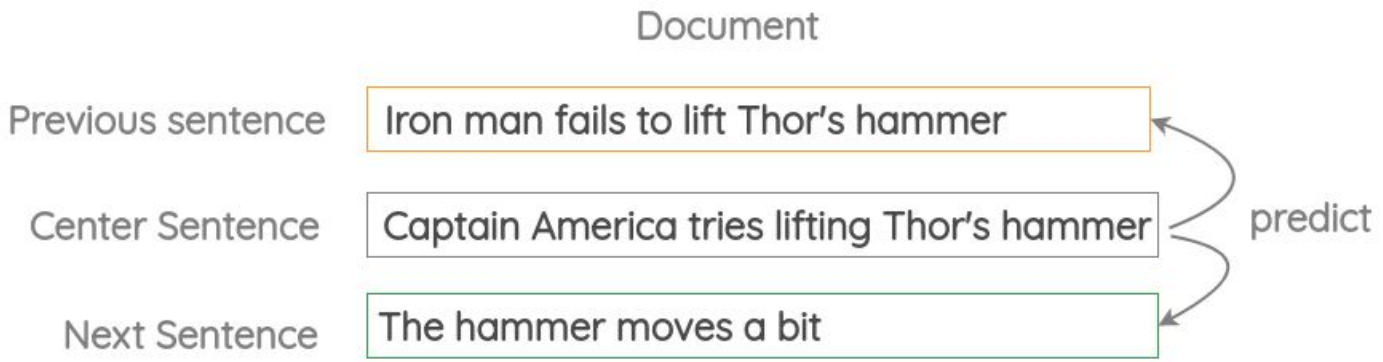
- Estas tareas son de tipo autosupervisado, ya que se busca aprovechar la gran cantidad de datos textuales existentes sin la necesidad de realizar un etiquetado manual.

Tareas Pretexto en PLN

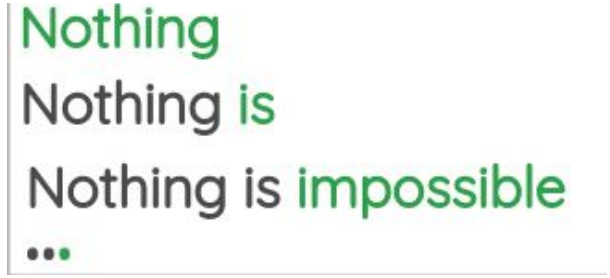
- Predicción de una palabra central

- Predicción del contexto

- Predicción de oraciones vecinas

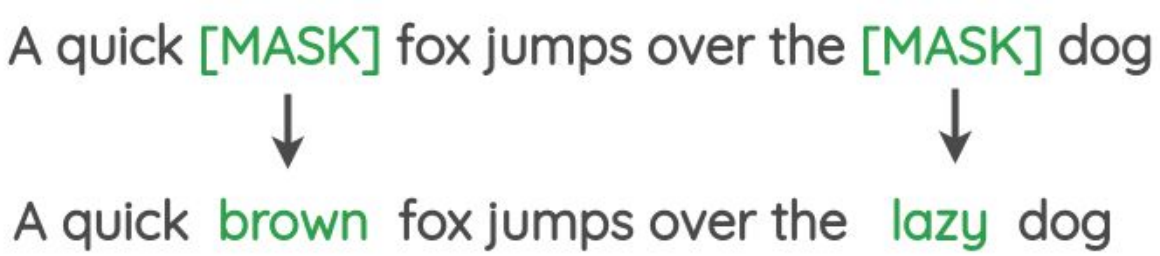


- Modelado de lenguaje



Tareas Pretexto en PLN

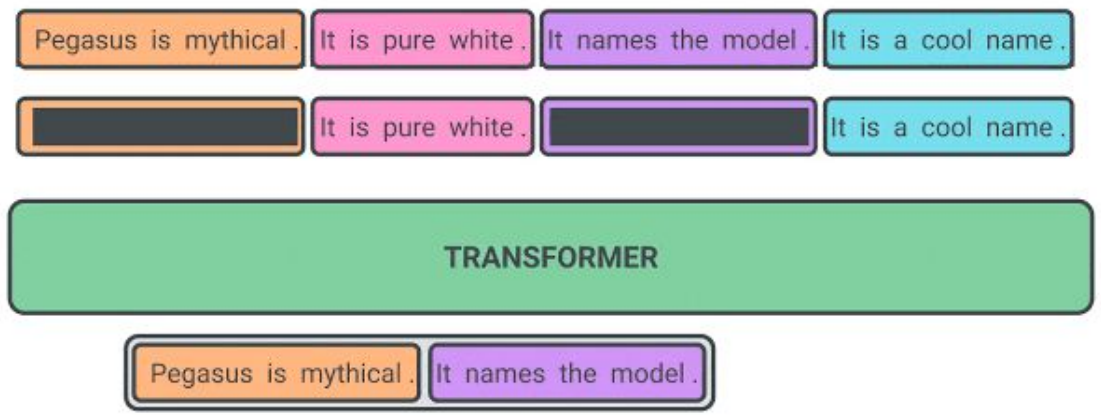
- Modelado del lenguaje enmascarado



- Predicción de próxima oración

- Predicción del orden de oraciones

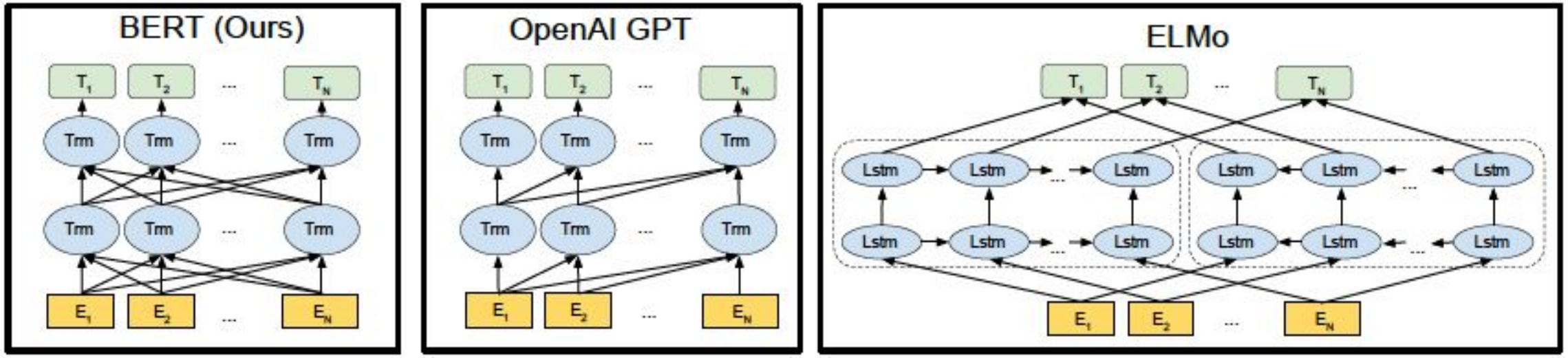
- Predicción de oraciones



BERT

Bidirectional Encoder Representations from Transformers
Devlin et al 2019

https://www.aclweb.org/anthology/N19-1423/



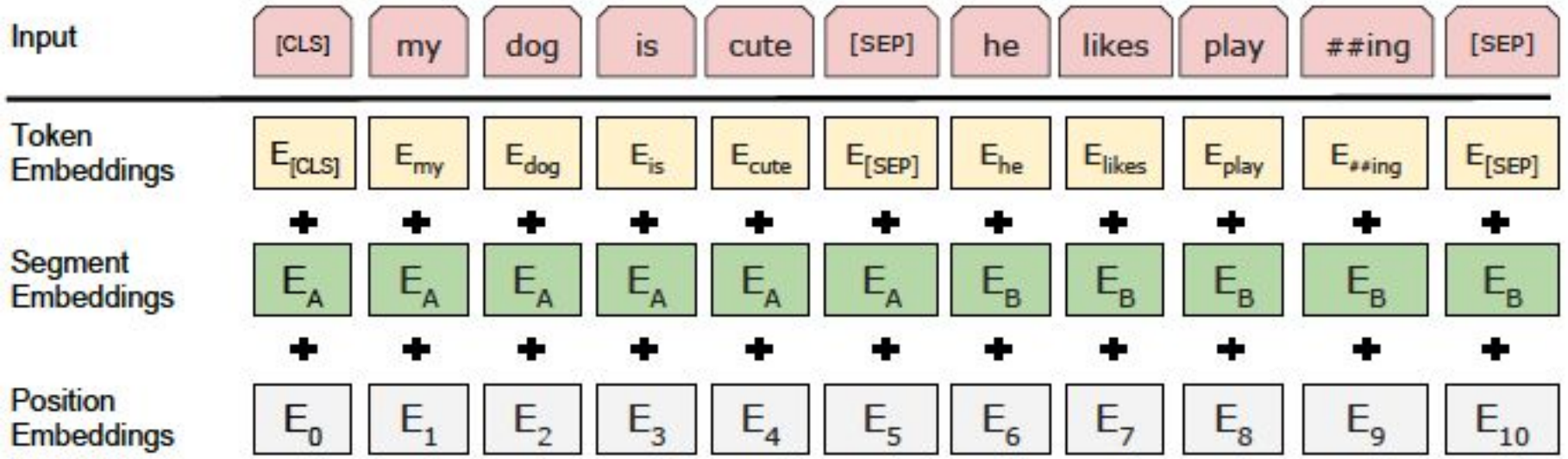
BERT - Arquitectura

BERT consiste en el codificador de la arquitectura Transformer.
Cuenta con bloques codificadores apilados.

El artículo propone dos versiones del modelo:

- Base
  - 12 capas
  - 12 cabezas
  - Dimensión de estados ocultos de 768
  - 110 Millones de parámetros
- Grande
  - 24 capas
  - 16 cabezas
  - Dimensión de estados ocultos de 1024
  - 340 Millones de parámetros

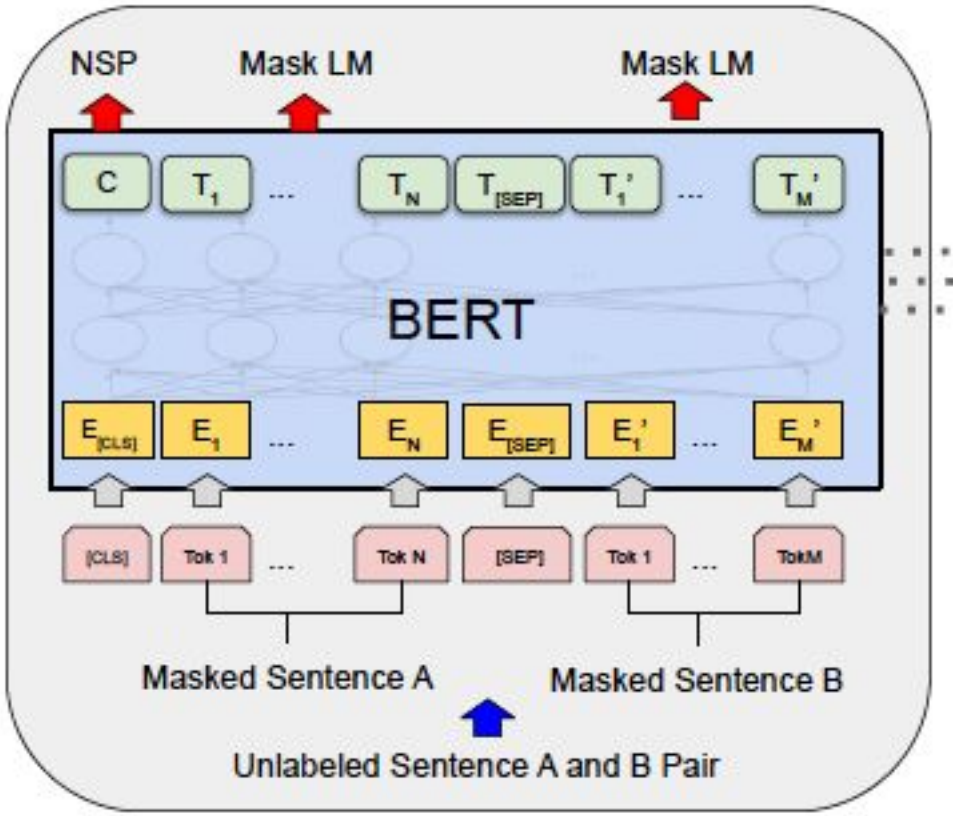
BERT - Entradas



BERT - Preentrenamiento

BERT utilizó dos tareas pretexto para su preentrenamiento.

- Modelado del lenguaje enmascarado.
- Predicción de próxima oración.



Datos

- Wikipedia 2.5 mil millones de palabras.
- BookCorpus 800 millones de palabras.

Entrenamiento por 4 días.

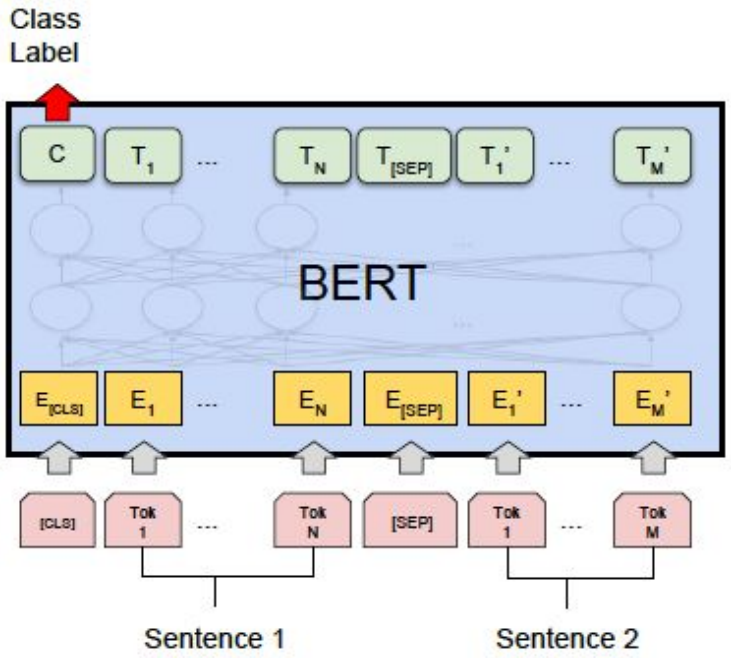
BERT - Tokenización

BERT utiliza la tokenización de **WordPiece**. Cada token está en algún lugar entre las secuencias a nivel de palabra y a nivel de carácter. Cualquier palabra que no aparezca en el vocabulario de **WordPiece** se descompone en subpalabras de manera codiciosa. Por ejemplo, «RTX» se descompone en «R», «##T» y «##X», donde ## indica que es un subtoken.

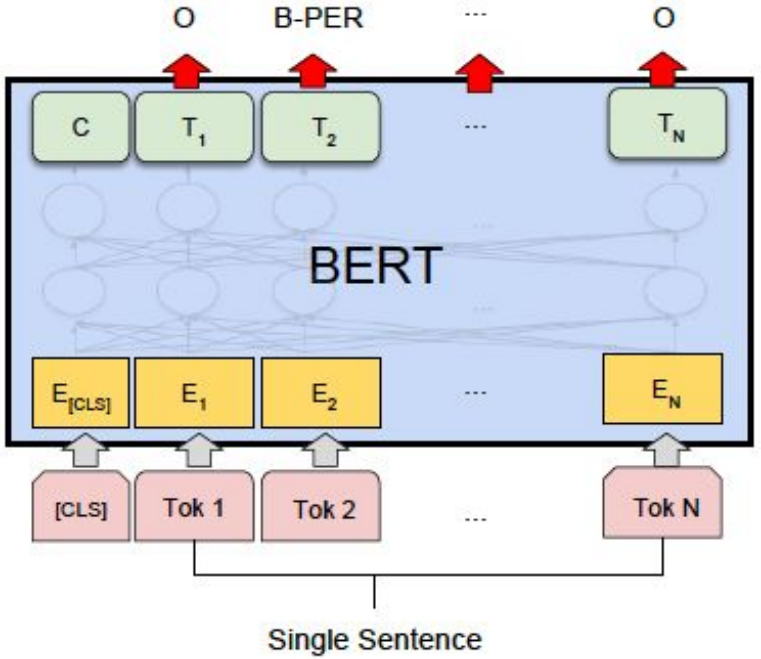
Tokens	[CLS]	don	‘	t	like	it	!	[SEP]	[PAD]	[PAD]
Token IDs	101	2123	1005	1056	2066	2099	999	102	0	0
Token Type IDs	0	0	0	0	0	0	0	0	0	0
Attention Mask	1	1	1	1	1	1	1	1	0	0

https://alvinntnu.github.io/NTNU_ENC2045_LECTURES/nlp/dl-attention-transformer-intuition.html

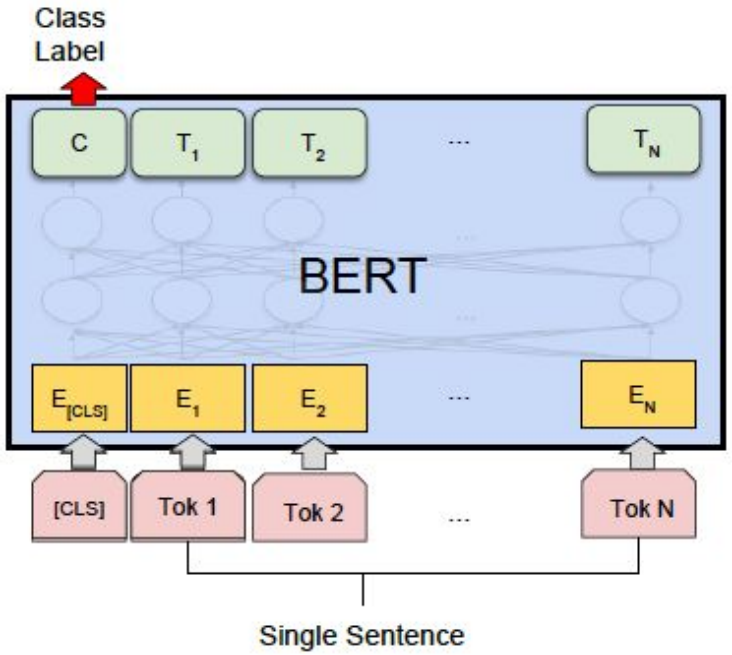
BERT – Ajuste Fino



Clasificación de un par de oraciones



Etiquetado de secuencias



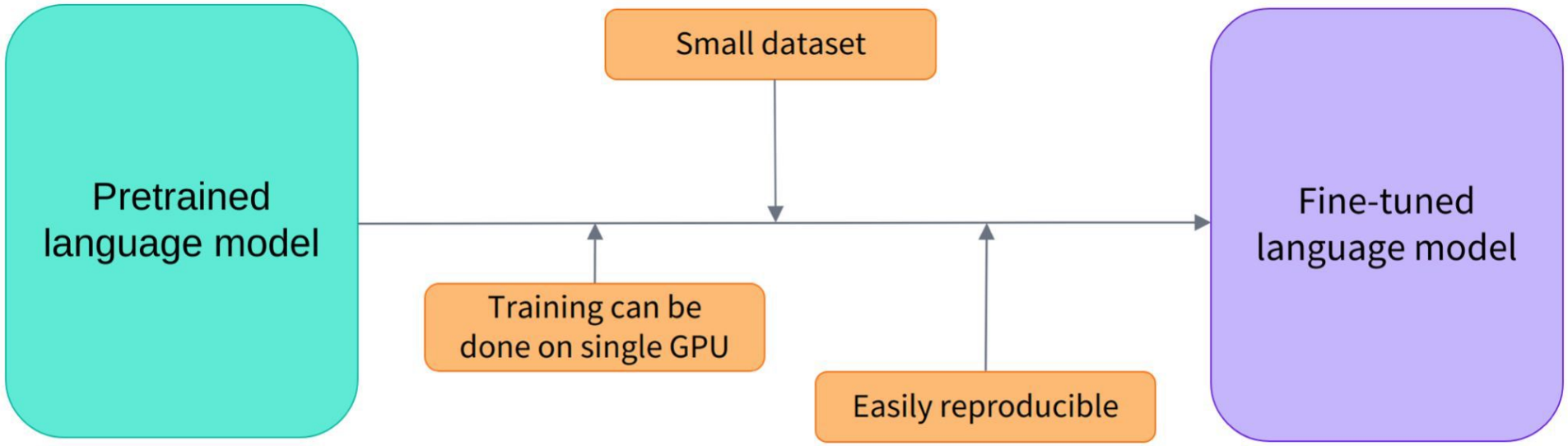
Clasificación de una oración

GPT-2 (Generative Pre-training 2)

- GPT-2 se hizo famoso (notorio) en los medios de comunicación como la "IA de escritura maliciosa", 2019.
- A diferencia de BERT, GPT-2 es fundamentalmente un modelo de generación de lenguaje (es decir, un **modelo generativo** orientado al decodificador).
- En comparación con BERT, GPT-2 presenta mayor diversidad lingüística de sus datos de entrenamiento (por ejemplo, publicaciones y enlaces a través del sitio de red social *Reddit* con un nivel mínimo de apoyo social, es decir, "Likes").
- Estadísticas de GPT-2:
  - Bloques de transformador: 48
  - Tamaño de la secuencia: 1024
  - Dimensión de Embedding: 1600
  - Cabezas de Atención: 36
  - Número de parámetros: 1.5B

Ajuste Fino

- Utilizar los pesos del modelo preentrenado como “cuerpo”
- Añadir una “cabeza” específica para la tarea (por ej. fully connected network + softmax)
- Afinar los pesos de la cabeza con un entrenamiento supervisado

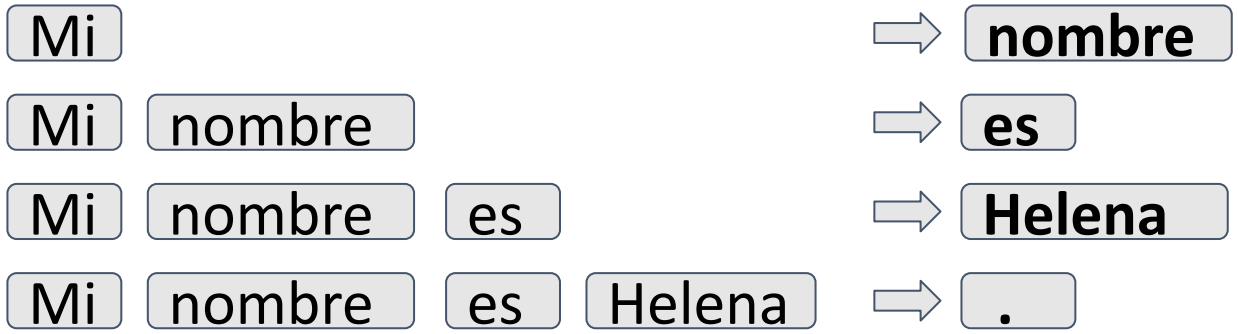


Youtube videos
BooksCorpus
SWAG,IMDB,Twitter
Knowledge Graph
Monolingual Corpora (104 languages)
Biomedical corpus
Corpora/Data
Clinical notes/EHR
Hierarchical diagnostic codes
Scientific publications
English Wikipedia
Graph Neural Net
SciBERT
ERNIE(1)
M-BERT
Pre-trained models
ClinicalBERT
ERNIE(2)
G-BERT
BioBERT
VideoBERT
BERT
TransBERT
Generic & Domain specific NLP tasks (e.g. NER)
DocBERT
Relation classification
PatentBERT
Code Switching (e.g. English/Hindi mix sentences)
Prediction tasks (e.g. Hospital readmission)
Fine-tuned models
Video captioning (classification)
Classification tasks (e.g. medication recommendation)

https://towardsdatascience.com/a-review-of-bert-based-models-4ffdc0f15d58

Modelado autorregresivo

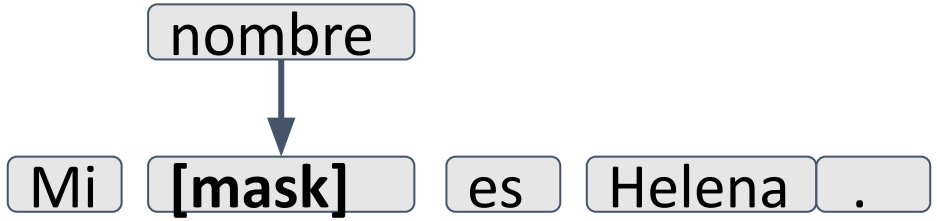
El modelado de lenguaje autorregresivo o causal predice el siguiente token en una secuencia de tokens y el modelo solo puede considerar los tokens a la izquierda.



https://somosnlp.org/nlp-de-cero-a-cien/sesi

Modelado enmascarado

El modelado de lenguaje por enmascaramiento predice un token enmascarado en una secuencia y el modelo puede considerar los tokens bidireccionalmente.



https://somosnlp.org/nlp-de-cero-a-cien/sesion-04

Modelos basados en BERT

Ortiz Barajas, Jesús Germán. Detección automática de paráfrasis mediante el modelo sentence-crobi, una arquitectura de red neuronal simple basada en codificadores cruzados y bi-codificadores. Tesis UNAM 2023.

Existen muchas variantes que intentan mejorar algunos aspectos como el rendimiento y el consumo de recursos.

- **XLNet**. Mejora la metodología de preentrenamiento, más datos y más poder computacional. Reportan mejores resultados que BERT.
- **RoBERTa**. Los creadores de este modelo consideraron que BERT estaba "under-trained". Ajustaron los hiperparámetros, eliminaron la tarea de predicción de próxima oración. Utilizaron más texto para el entrenamiento
- **DistilBERT**. Aprende una versión aproximada de BERT, mantiene el 97% de su rendimiento utilizando la mitad de parámetros. Para el entrenamiento utiliza una técnica conocida como destilación.
- **DeBERTA**. Se incorpora la atención desenredada: Las palabras son codificadas usando 2 vectores, uno en el que se captura el contenido o significado y otro en el que se codifica la posición relativa dentro del texto. Resuelve la limitante de BERT en el que los vectores de contenido y posición se suman para obtener la representación vectorial de un token.

Tokenización

En el procesamiento de lenguaje natural (NLP), la **tokenización** es el paso fundamental que convierte texto crudo en secuencias numéricas o **tensores** .

Para modelos basados en Transformers, la arquitectura no procesa letras ni palabras, procesa **Tokens**. Durante el **ajuste fino**, es imperativo reutilizar el exacto del pre-entrenamiento para no invalidar los *Embeddings* aprendidos.

Primeros Enfoques y sus Limitaciones

Word-based Tokenization

Divide el texto basándose en espacios y signos de puntuación (ej. "I love AI" → ["I", "love", "AI"]).

- Pro: Conserva el significado semántico completo de cada palabra.
- Con: Genera vocabularios inmensos y sufre drásticamente del problema de palabras fuera del vocabulario (OOV) al encontrar palabras nuevas o mal escritas.

Character-based Tokenization

Divide el texto en letras individuales o caracteres (ej. "AI" → ["A", "I"]).

- Pro: Vocabulario extremadamente reducido y elimina el problema por completo.
- Con: Destruye los límites semánticos y genera secuencias excesivamente largas, saturando el *Context Window* del Transformer.

Tokenización basada en subpalabras

El estándar actual en Transformers. Divide palabras raras en sub-unidades lógicas (ej. "tokenization" → "token", "##ization") manteniendo las palabras comunes intactas. Balancea el tamaño del vocabulario y la longitud de la secuencia.

Byte-Pair Encoding (BPE)

Algoritmo iterativo que comienza con caracteres individuales y fusiona los pares más frecuentes en el corpus de entrenamiento secuencialmente. Es la base de la familia de modelos **GPT**.

WordPiece

Similar a BPE, pero en lugar de elegir el par más frecuente, elige la fusión que maximice la probabilidad (*likelihood*) de los datos de entrenamiento. Es el motor principal detrás de **BERT**.

SentencePiece / Unigram

Trata los espacios como caracteres regulares (ej. `_`), procesando el texto crudo directamente. Es completamente agnóstico del idioma, utilizado ampliamente en modelos como **T5** y **LLaMA**.

El Pipeline de Tokenización

El proceso estandarizado para transformar una cadena de texto (*string*) en los *inputs* requeridos por el modelo pre-entrenado.

1. Normalization
Limpieza del texto crudo. Incluye pasar a minúsculas (*lowercasing*), eliminar acentos o estandarizar espacios.

2. Pre-Tokenization
División inicial del texto en unidades básicas o palabras, generalmente usando espacios o reglas lingüísticas.

3. Model Tokenization
Aplicación estricta del algoritmo entrenado (WordPiece, BPE) para dividir en subpalabras basados en el **vocabulario**.

4. Post-Processing
Inyección de **Tokens Espaciales** estructurales (`[CLS]`, `[SEP]`, ` `) exigidos por la arquitectura del Transformer.

Utilizaremos la librería de Huggingface para implementar un modelo de análisis de sentimientos: https://huggingface.co/docs/transformers/index