NLP, Transformers y Clasificación Zero-Shot

Inteligencia Artificial

Universidad EAFIT

September 9, 2025

Agenda de la Sesión

Parte 1: Del NLP Clásico a los Transformers

Parte 2: Modelos Pre-entrenados y Aprendizaje Zero-Shot

Parte 3: Demo Práctica con Python

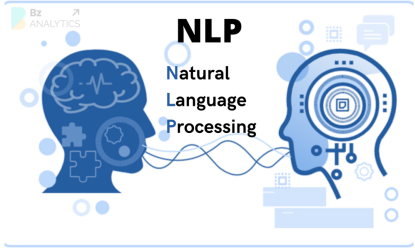
Conclusiones y Recursos

Parte 1

Del NLP Clásico a los Transformers

¿Qué es el Procesamiento del Lenguaje Natural (NLP)?

Es una rama de la Inteligencia Artificial que permite a las máquinas **comprender, interpretar y generar** lenguaje humano.

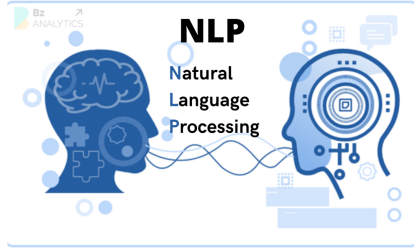


Concepto NLP.

¿Qué es el Procesamiento del Lenguaje Natural (NLP)?

Es una rama de la Inteligencia Artificial que permite a las máquinas **comprender, interpretar y generar** lenguaje humano.

- **Objetivo**: Cerrar la brecha entre la comunicación humana y la comprensión de las computadoras.

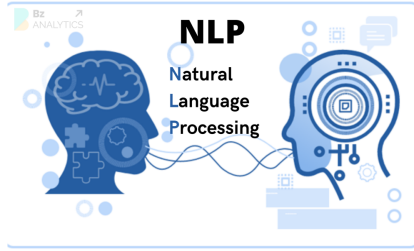


Concepto NLP.

¿Qué es el Procesamiento del Lenguaje Natural (NLP)?

Es una rama de la Inteligencia Artificial que permite a las máquinas **comprender, interpretar y generar** lenguaje humano.

- **Objetivo**: Cerrar la brecha entre la comunicación humana y la comprensión de las computadoras.
- **Tareas Clave**:
  - Análisis de Sentimientos
  - Traducción Automática
  - Resumen de Texto
  - Clasificación de Tópicos



Concepto NLP.

Limitaciones del NLP Clásico: Bolsa de Palabras (BoW)

El Modelo de Bolsa de Palabras (Bag-of-Words)

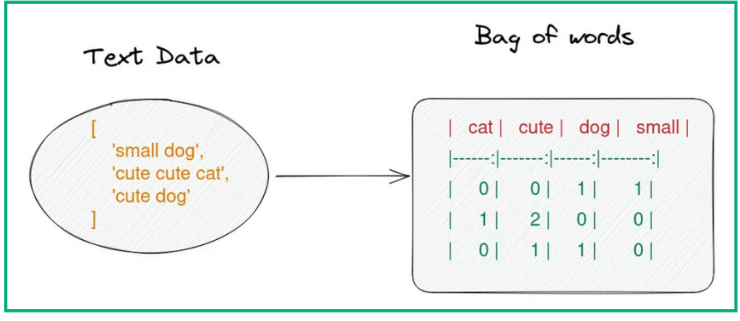
Representa el texto como un multiconjunto (una bolsa) de sus palabras, sin tener en cuenta el orden ni la gramática.

Ejemplo: "El perro persigue al gato" y "El gato persigue al perro" tienen la misma representación BoW.

Limitaciones del NLP Clásico: Bolsa de Palabras (BoW)

El Modelo de Bolsa de Palabras (Bag-of-Words)

Problema Fundamental: Pérdida total del contexto y la secuencia.



Analogía: Una bolsa de fichas de Scrabble. Tenemos las letras, pero no las palabras ni las frases.

La Era Secuencial: Redes Neuronales Recurrentes (RNNs)

Procesando palabras en orden

Las RNNs fueron diseñadas para manejar datos secuenciales, procesando la información un elemento a la vez y manteniendo un "recuerdo" o estado oculto.

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t + b_h) \tag{1}$$

Donde h_t es el estado oculto en el tiempo t, x_t es la entrada y W son los pesos.

La Era Secuencial: Redes Neuronales Recurrentes (RNNs)

Procesando palabras en orden

Limitación Principal: El Problema del Gradiente Evanescente

La información de los primeros elementos de la secuencia se diluye, dificultando la captura de dependencias a largo plazo.

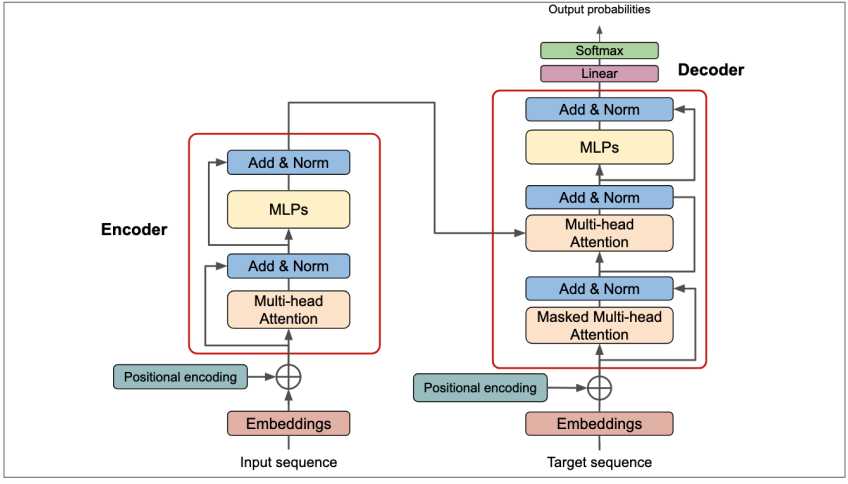
La Revolución Transformer: "Attention Is All You Need"

Presentado por Google en 2017, el Transformer se deshizo por completo de la recurrencia.

Idea Clave: Mecanismo de Auto-Atención (Self-Attention)

Permite al modelo sopesar la importancia de diferentes palabras en la misma secuencia, sin importar su distancia, y procesarlas todas en paralelo.

La Revolución Transformer: "Attention Is All You Need"



La atención permite conexiones directas entre palabras, superando las limitaciones de las RNNs.

Auto-Atención: El Concepto Clave

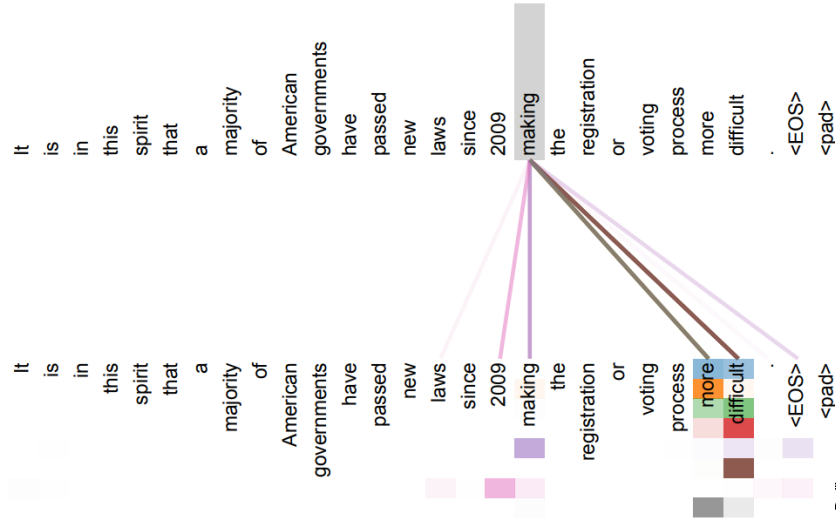
¿En qué otras palabras debería fijarme para entender esta palabra?

Consideremos la frase: "El robot no pudo levantar el trofeo porque **él** era demasiado pesado."

- ¿A qué se refiere **"él"**? ¿Al robot o al trofeo?
- El mecanismo de auto-atención aprende a asociar "él" con "trofeo" asignándole un **peso de atención** más alto.

Auto-Atención: El Concepto Clave

¿En qué otras palabras debería fijarme para entender esta palabra?



La Matemática de la Auto-Atención

Queries, Keys y Values

Para cada palabra (embedding de entrada x), creamos tres vectores:

- **Query (q)**: Lo que estoy buscando.
- **Key (k)**: Lo que esta palabra "ofrece" o representa.
- **Value (v)**: El contenido real de la palabra.

Estos se calculan multiplicando la entrada por matrices de pesos entrenables: $q = W_Q x$, $k = W_K x$, $v = W_V x$.

La Matemática de la Auto-Atención

Queries, Keys y Values

Fórmula de la Atención Escalada

La salida es un promedio ponderado de los Values, donde los pesos se calculan a partir de la compatibilidad entre Queries y Keys.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

El factor $\sqrt{d_k}$ estabiliza los gradientes durante el entrenamiento.

Atención Multiple (Multi-Head Attention)

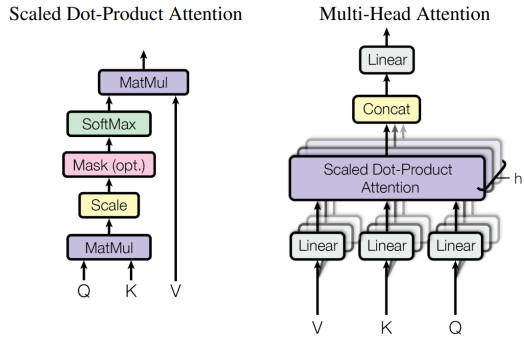
En lugar de calcular la atención una sola vez, el mecanismo de "Multi-Cabeza" lo hace en paralelo varias veces.

Analogía

Es como tener un comité de expertos. Cada "cabeza" (experto) se especializa en detectar diferentes tipos de relaciones:

- Una cabeza para relaciones sintácticas.
- Otra para relaciones semánticas.
- Otra para identificar correferencias.

Los resultados de cada cabeza se

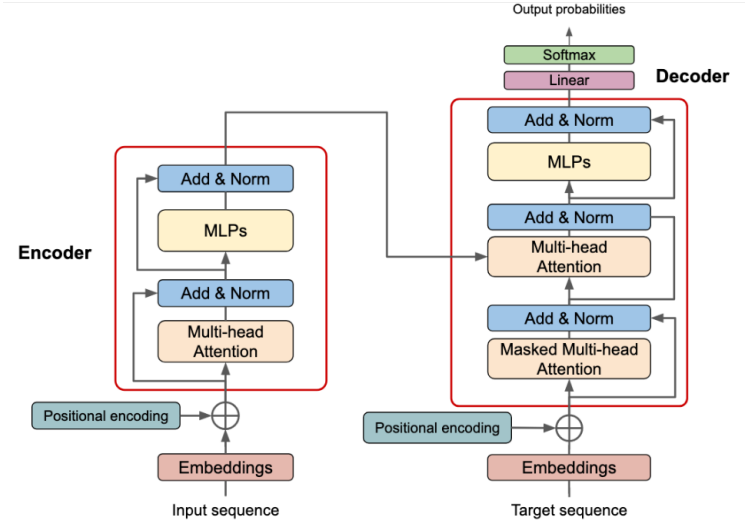


Múltiples capas de atención (cabezas) procesan la entrada en paralelo.

La Arquitectura Completa del Transformer

- **Codificación Posicional (Positional Encoding)**: Ya que no hay recurrencia, se inyecta información sobre la posición de las palabras en la secuencia.
- **Bloque Encoder**: Compuesto por Auto-Atención Multi-Cabeza y una Red Neuronal Feed-Forward.
- **Bloque Decoder**: Similar al encoder, pero con una capa de atención adicional que se enfoca en la salida del encoder.
- **Normalización de Capa y Conexiones Residuales**: Claves para entrenar redes profundas de manera estable.

La Arquitectura Completa del Transformer



Parte 2

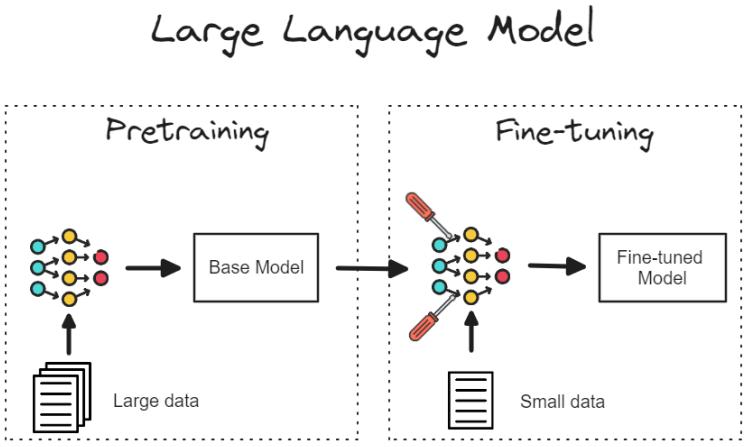
Modelos Pre-entrenados y Aprendizaje Zero-Shot

La Era del Pre-entrenamiento y Fine-Tuning

El Paradigma Dominante

1. **Pre-entrenamiento (Costoso)**: Se entrena un modelo Transformer masivo (como BERT o GPT) en un corpus de texto gigantesco (ej. toda Wikipedia). El objetivo es que el modelo aprenda gramática, semántica, hechos del mundo y razonamiento.
2. **Ajuste Fino (Fine-Tuning, Rápido y Barato)**: Se toma el modelo pre-entrenado y se sigue entrenando en un conjunto de datos mucho más pequeño y específico para una tarea concreta (ej. clasificar emails de soporte).

La Era del Pre-entrenamiento y Fine-Tuning



Dos Familias de Modelos: BERT y GPT

BERT (Bidirectional)

Objetivo: Comprensión.
Optimizado para entender el contexto de una palabra usando **ambas direcciones** (izquierda y derecha).
Tarea de Pre-entrenamiento: Masked Language Model (MLM). Predice palabras enmascaradas.
Ej: "Fui al [MASK] a comprar pan."

GPT (Generative)

Objetivo: Generación.
Optimizado para generar texto coherente prediciendo la **siguiente palabra** basándose solo en el contexto previo (izquierda).
Tarea de Pre-entrenamiento: Causal Language Model (CLM).
Ej: "El gato se sentó en la..."

¿Qué es el Aprendizaje Zero-Shot?

Es la capacidad de un modelo para realizar una tarea para la cual **no ha sido explícitamente entrenado**.

Analogía Humana

Si has aprendido a identificar manzanas, peras y naranjas, y te muestran un mango por primera vez, puedes inferir que es una "fruta" sin que nadie te lo enseñe. Usas tu conocimiento general para clasificar un objeto nuevo.

En NLP, esto significa clasificar texto con etiquetas que el modelo **nunca vio** durante su entrenamiento.

¿Cómo Funciona la Clasificación Zero-Shot?

Reformulando el problema como Inferencia de Lenguaje Natural (NLI)

La tarea NLI consiste en determinar si una "premisa" y una "hipótesis" se relacionan de una de estas tres formas: **entailment** (implica), **contradiction** (contradice) o **neutral**.

Nuestra Tarea: Clasificar ‘"Apple anuncia el nuevo iPhone 17"‘ con las etiquetas ‘[Tecnología, Deporte, Política]‘.

- **Premisa**: ‘"Apple anuncia el nuevo iPhone 17"‘
- **Hipótesis 1**: ‘"Este texto es sobre tecnología."‘ $\rightarrow$ **ALTA PROBABILIDAD DE ENTAILMENT**
- **Hipótesis 2**: ‘"Este texto es sobre deporte."‘ $\rightarrow$ **ALTA PROBABILIDAD DE CONTRADICTION**

La etiqueta cuya hipótesis genera la mayor probabilidad de ‘entailment‘ es la ganadora.

La Intuición Matemática detrás de Zero-Shot

Sea x la secuencia de texto de entrada y $L = \{l_1, l_2, ..., l_k\}$ el conjunto de etiquetas candidatas.

1. Para cada etiqueta $l_j \in L$, construimos una hipótesis h_j usando una plantilla.
   Ejemplo de plantilla: "This text is about l_j."
2. El modelo, pre-entrenado en NLI, calcula la probabilidad de que la premisa x implique la hipótesis h_j.

$$P(\text{entailment} | \text{premise} = x, \text{hypothesis} = h_j)$$

3. La etiqueta predicha $\hat{l}$ es aquella que maximiza esta probabilidad.

$$\hat{l} = \arg\max_{l_j \in L} P(\text{entailment} | x, h_j)$$

Ventajas y Limitaciones del Enfoque Zero-Shot

Ventajas

- **Flexibilidad Extrema**: No se necesita un dataset etiquetado para tus clases específicas.
- **Velocidad**: No requiere re-entrenamiento. Ideal para prototipado rápido.
- **Costo-Efectividad**: Evita la costosa anotación de datos.

Limitaciones

- **Precisión**: Generalmente inferior a un modelo ‘fine-tuned‘ supervisado.
- **Sensibilidad a la Plantilla**: El resultado puede cambiar drásticamente según cómo se formule la hipótesis.
- **Dominios Especializados**: Puede fallar en temas muy técnicos o de nicho no presentes en sus datos de entrenamiento.

Parte 3

Demo Práctica con Python y Transformers

El Ecosistema de Hugging Face

Hugging Face se ha convertido en el centro neurálgico para la comunidad de IA, especialmente en NLP.

- **Model Hub**: Un repositorio con miles de modelos pre-entrenados listos para usar.
- **Librería ‘transformers‘**: Proporciona una API de alto nivel y unificada para cargar y usar estos modelos con unas pocas líneas de código.
- **Librería ‘datasets‘**: Facilita el acceso y procesamiento de grandes conjuntos de datos.

El Ecosistema de Hugging Face

Hugging Face se ha convertido en el centro neurálgico para la comunidad de IA, especialmente en NLP.

- **Librería ‘datasets‘**: Facilita el acceso y procesamiento de grandes conjuntos de datos.

Su misión es "democratizar el buen Machine Learning".

Código 1: Preparando el Entorno

Instalación de dependencias en Python

```
# Instalamos la libreria de transformers y un backend de ML
# Usaremos PyTorch, pero tambien es compatible con TensorFlow
pip install transformers torch
```

Con solo estas dos librerías, tenemos acceso a toda la potencia del Model Hub de Hugging Face.

Código 2: La Abstracción ‘pipeline‘

La forma más sencilla de usar un modelo

La función ‘pipeline‘ de ‘transformers‘ se encarga de todo el pre-procesamiento, inferencia y post-procesamiento.

Inicializando un pipeline de clasificación Zero-Shot

```python
from transformers import pipeline

# Cargamos el pipeline especificando la tarea
# Por detras, descarga un modelo por defecto para esta tarea
# 'facebook/bart-large-mnli' es un modelo excelente para NLI
classifier = pipeline(
    "zero-shot-classification",
    model="facebook/bart-large-mnli"
)
```

La primera vez que se ejecuta, el modelo se descarga y se guarda en caché.

Resultado Esperado

```
{
  'sequence': 'Who are you voting for in the next election?',
  'labels': ['politics', 'business', 'technology'],
  'scores': [0.98, 0.01, 0.01]
}
```

El modelo asigna una confianza del 98% a la etiqueta "politics".

```python
# 1. La secuencia que queremos clasificar
sequence = "Who are you voting for in the next election?"

# 2. Las etiquetas candidatas que queremos probar
candidate_labels = ["politics", "business", "technology"]

# 3. Pasamos todo al pipeline
result = classifier(sequence, candidate_labels)
print(result)
```

=

Código 4: Clasificación Multi-Etiqueta

Podemos permitir que el modelo asigne altas puntuaciones a varias etiquetas si son relevantes.

Ejemplo

```
sequence = "The new Avengers movie is breaking all box office records."
candidate_labels = ["cinema", "art", "finance", "pop culture"]

# El parametro multi_label=True cambia el softmax final
# por un sigmoide en cada etiqueta.
result = classifier(
    sequence,
    candidate_labels,
    multi_label=True
)

print(result)
```

Resultado Esperado (aproximado)

Conclusiones

- El mecanismo de **auto-atención** es el pilar de la arquitectura Transformer y ha supuesto un cambio de paradigma en NLP, permitiendo una mejor captura del contexto.
- El pre-entrenamiento en corpus masivos ha dado lugar a modelos fundacionales como **BERT** y **GPT** que sirven como base para una multitud de tareas.
- El aprendizaje **Zero-Shot** es una técnica increíblemente poderosa y flexible que aprovecha los modelos pre-entrenados en NLI para clasificar texto con etiquetas nuevas y dinámicas sin necesidad de re-entrenamiento.
- El ecosistema de **Hugging Face** ha democratizado el acceso a estas tecnologías, haciendo posible implementar soluciones de IA de vanguardia con muy poco código.

Recursos y Siguientes Pasos

- **Paper Original del Transformer**: "Attention Is All You Need"
- **Hugging Face Hub**: huggingface.co/models - Explora miles de modelos para cualquier tarea.
- **Curso de Hugging Face**: huggingface.co/learn/nlp-course - Un recurso gratuito y completo para aprender sobre Transformers.
- **Blog de Jay Alammar**: "TheIllustratedTransformer" - Una explicación visual e intuitiva excelente.

¿Preguntas?

Gracias por su atención