¿Qué es PLN?

- El PLN es un campo de la lingüística y el machine learning enfocado en entender todo lo relacionado con el lenguaje humano.
- El objetivo de las tareas de PLN no sólo es entender palabras de manera individual, sino también entender su contexto.

- **Clasificar oraciones enteras**: Obtener el sentimiento de una reseña, detectar si un correo electrónico es spam, determinar si una oración es gramaticalmente correcta o si dos oraciones están lógicamente relacionadas entre si
- **Clasificar cada palabra en una oración**: Identificar los componentes gramaticales de una oración (sustantivo, verbo, adjetivo) o las entidades nombradas (persona, ubicación, organización)
- **Generar texto**: Completar una indicación con texto generado automáticamente, completar los espacios en blanco en un texto con palabras ocultas
- **Extraer una respuesta de un texto**: Dada una pregunta y un contexto, extraer la respuesta a la pregunta en función de la información proporcionada en el contexto
- **Generar una nueva oración de un texto de entrada**: Traducir un texto a otro idioma, resumir un texto

Transformadores, ¿qué pueden hacer?

- La librería Transformers provee la funcionalidad de crear y usar estos modelos compartidos.
- El Hub de Modelos contiene miles de modelos preentrenados que cualquiera puede descargar y usar.

- El objeto más básico en la librería Transformers es la función **pipeline**().
- Esta función conecta un modelo con los pasos necesarios para su preprocesamiento y posprocesamiento, permitiéndonos introducir de manera directa cualquier texto y obtener una respuesta.

¿Lo probamos?

```
from transformers import pipeline

classifier = pipeline("sentiment-analysis")
classifier("Hoy estoy feliz.")
```

- Por defecto, este pipeline selecciona un modelo particular preentrenado que ha sido ajustado para el análisis de sentimientos en Inglés.
- El modelo se descarga y se almacena en el caché cuando creas el objeto classifier. Si vuelves a ejecutar el comando, se usará el modelo almacenado en caché y no habrá necesidad de descargarlo de nuevo.
- Hay tres pasos principales que ocurren cuando pasas un texto a un pipeline:
  - El texto es preprocesado en un formato que el modelo puede entender.
  - La entrada preprocesada se pasa al modelo.
  - Las predicciones del modelo son posprocesadas, de tal manera que las puedas entender.

Algunos de los pipelines disponibles son

- feature-extraction (obtener la representación vectorial de un texto)
- fill-mask
- ner (reconocimiento de entidades nombradas)
- question-answering
- sentiment-analysis
- summarization
- text-generation
- translation
- zero-shot-classification

Clasificación zero-shot

- Este es un escenario común en proyectos de la vida real porque anotar texto usualmente requiere mucho tiempo y dominio del tema. Para este caso de uso, el pipeline zero-shot-classification es muy poderoso: permite que especifiques qué etiquetas usar para la clasificación, para que no dependas de las etiquetas del modelo preentrenado.

```
from transformers import pipeline
 classifier = pipeline("zero-shot-classification")
classifier( "This is a course about the
   Transformers library",
   candidate_labels=["education", "politics",
   "business"], )
```

Generación de texto

La idea es que proporciones una indicación (*prompt*) y el modelo la va a completar automáticamente al generar el texto restante. Esto es parecido a la función de texto predictivo que está presente en muchos teléfonos. La generación de texto involucra aleatoriedad, por lo que es normal que no obtengas el mismo resultado que se muestra abajo.

```
from transformers import pipeline
generator = pipeline("text-generation")

generator("In this course, we will teach you how
    to")
```

1-¡Pruébalo!

Usa los argumentos num_return_sequences y max_length para generar dos oraciones de 15 palabras cada una.

Usa cualquier modelo del Hub en un pipeline

- Los ejemplos anteriores usaban el modelo por defecto para cada tarea, pero también puedes escoger un modelo particular del Hub y usarlo en un pipeline para una tarea específica - por ejemplo, la generación de texto.

- ¡Intentemos con el modelo distilgpt2!

```
from transformers import pipeline
generator = pipeline("text-generation",
    model="distilgpt2")
generator( "In this course, we will teach you
    how to", max_length=30,
    num_return_sequences=2, )
```

Llenado de ocultos (Mask filling)

```
from transformers import pipeline

unmasker = pipeline("fill-mask")

unmasker("This course will teach you all about
   <mask> models.", top_k=2)
```

2 - ¡Pruébalo!

- Busca el modelo bert-base-cased en el Hub e identifica su *mask token* en el widget de la API de Inferencia. ¿Qué predice este modelo para la oración que está en el ejemplo de pipeline anterior?

Reconocimiento de entidades nombradas

- El reconocimiento de entidades nombradas (REN) es una tarea en la que el modelo tiene que encontrar cuáles partes del texto introducido corresponden a entidades como personas, ubicaciones u organizaciones.

```
from transformers import pipeline
ner = pipeline("ner", grouped_entities=True)

ner("Mi nombre es Chelo y trabajo como
    profesora en IES Abastos")
```

3 –¡Pruébalo!

Busca en el Model Hub un modelo capaz de hacer etiquetado *part-of-speech* (que se abrevia usualmente como POS) en Inglés. ¿Qué predice este modelo para la oración en el ejemplo de arriba?

Responder preguntas

- El pipeline question-answering responde preguntas usando información de un contexto dado.

```
from transformers import pipeline
    question_answerer = pipeline("question-
    answering")
question_answerer( question="Where do I
    work?", context="My name is Sylvain and I
    work at Hugging Face in Brooklyn", )
```

Resumir (Summarization)

- Resumir es la tarea de reducir un texto en uno más corto, conservando todos (o la mayor parte de) los aspectos importantes mencionados.

```
summarizer = pipeline("summarization")
summarizer( """ America has changed dramatically during recent years. Not only has
    the number of graduates in traditional engineering disciplines such as mechanical,
    civil, electrical, chemical, and aeronautical engineering declined, but in most of the
    premier American universities engineering curricula now concentrate on and
    encourage largely the study of engineering science. As a result, there are declining
    offerings in engineering subjects dealing with infrastructure, the environment, and
    related issues, and greater concentration on high technology subjects, largely
    supporting increasingly complex scientific developments. While the latter is
    important, it should not be at the expense of more traditional engineering. Rapidly
    developing economies such as China and India, as well as other industrial
    countries in Europe and Asia, continue to encourage and advance the teaching of
    engineering. Both China and India, respectively, graduate six and eight times as
    many traditional engineers as does the United States. Other industrial countries at
    minimum maintain their output, while America suffers an increasingly serious
    decline in the number of engineering graduates and a lack of well-educated
    engineers. """ )
```

Traducción

- Para la traducción, puedes usar el modelo por defecto si indicas una pareja de idiomas en el nombre de la tarea (como "translation_en_to_fr"), pero la forma más sencilla es escoger el modelo que quieres usar en el Hub de Modelos

```
translator = pipeline("translation",
    model="Helsinki-NLP/opus-mt-fr-en")
translator("Ce cours est produit par Hugging
    Face.")
```

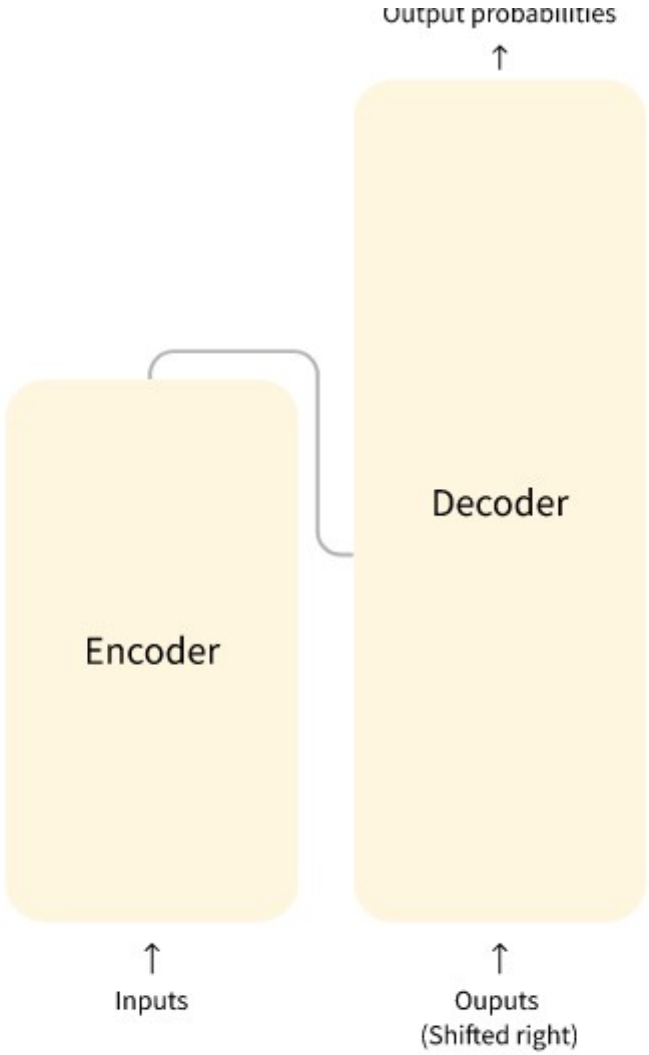
4 -¡Pruébalo!

Busca modelos de traducción en otros idiomas e intenta traducir la oración anterior en varios de ellos.

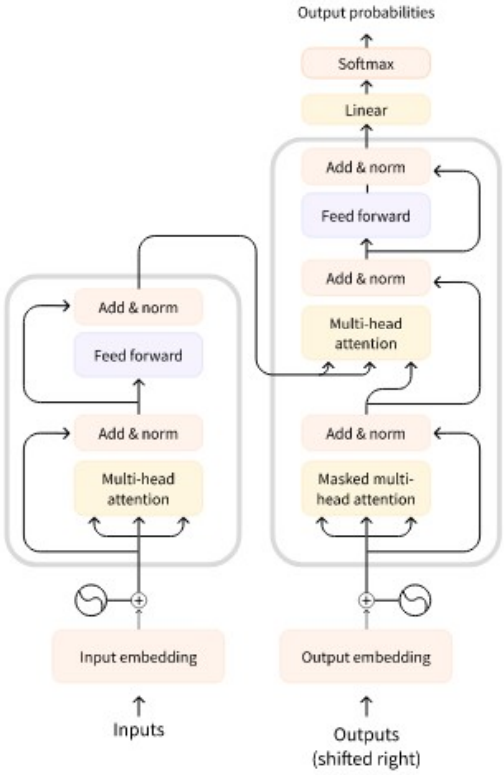
¿Cómo funcionan los Transformadores?

- El modelo está compuesto por dos bloques:
- **Codificador (izquierda)**: El codificador recibe una entrada y construye una representación de ésta (sus características). Esto significa que el modelo está optimizado para conseguir un entendimiento a partir de la entrada.
- **Decodificador (derecha)**: El decodificador usa la representación del codificador (características) junto con otras entradas para generar una secuencia objetivo. Esto significa que el modelo está optimizado para generar salidas.

- Todos los modelos de Transformadores d (GPT, BERT, BART, T5, etc.) han sido entrenados como *modelos de lenguaje*.
- Esto significa que han sido entrenados con grandes cantidades de texto crudo de una manera auto-supervisada.
- El aprendizaje auto-supervisado es un tipo de entrenamiento en el que el objetivo se computa automáticamente de las entradas del modelo.


Output probabilities
Decoder
Encoder
Inputs
Ouputs
(Shifted right)

- Una característica clave de los Transformadores es que están construidos con capas especiales llamadas *capas de atención*. De hecho, el título del trabajo que introdujo la arquitectura de los Transformadores fue "Attention Is All You Need".


Output probabilities
Softmax
Linear
Add & norm
Feed forward
Add & norm
Multi-head attention
Add & norm
Masked multi-head attention
Add & norm
Feed forward
Add & norm
Multi-head attention
Input embedding
Output embedding
Inputs
Outputs (shifted right)

- La arquitectura del Transformador fue diseñada originalmente para traducción.
- Durante el entrenamiento, el codificador recibe entradas (oraciones) en un idioma dado, mientras que el decodificador recibe las mismas oraciones en el idioma objetivo.
- En el codificador, las capas de atención pueden usar todas las palabras en una oración
- Por su parte, el decodificador trabaja de manera secuencial y sólo le puede prestar atención a las palabras en la oración que ya ha traducido (es decir, sólo las palabras antes de que la palabra se ha generado).
- Por ejemplo, cuando hemos predicho las primeras tres palabras del objetivo de traducción se las damos al decodificador, que luego usa todas las entradas del codificador para intentar predecir la cuarta palabra.

- Para poner esto en contexto, piensa en la tarea de traducir texto de Inglés a Francés.

- Dada la entrada “You like this course”, un modelo de traducción necesitará tener en cuenta la palabra adyacente “You” para obtener la traducción correcta de la palabra “like”, porque en Francés el verbo “like” se conjuga de manera distinta dependiendo del sujeto.

- Sin embargo, el resto de la oración no es útil para la traducción de esa palabra.

- En la misma línea, al traducir “this”, el modelo también deberá prestar atención a la palabra “course”, porque “this” se traduce de manera distinta dependiendo de si el nombre asociado es masculino o femenino.

- De nuevo, las otras palabras en la oración no van a importar para la traducción de “this”. Con oraciones (y reglas gramaticales) más complejas, el modelo deberá prestar especial atención a palabras que pueden aparecer más lejos en la oración para traducir correctamente cada palabra.

Modelos de codificadores

- Los modelos de codificadores usan únicamente el codificador del Transformador. En cada etapa, las capas de atención pueden acceder a todas las palabras de la oración inicial. Estos modelos se caracterizan generalmente por tener atención “bidireccional” y se suelen llamar modelos *auto-encoding*.
- Los miembros de esta familia de modelos incluyen:
- ALBERT
- BERT
- DistilBERT
- ELECTRA
- RoBERTa

Modelos de decodificadores

- Los modelos de decodificadores usan únicamente el decodificador del Transformador. En cada etapa, para una palabra dada las capas de atención pueden acceder solamente a las palabras que se ubican antes en la oración. Estos modelos se suelen llamar modelos *auto-regressive*.
- El preentrenamiento de los modelos de decodificadores generalmente gira en torno a la predicción de la siguiente palabra en la oración.
- Estos modelos son más adecuados para tareas que implican la generación de texto.

- Los miembros de esta familia de modelos incluyen:
- CTRL
- GPT
- GPT-2
- Transformer XL

Modelo	Ejemplos	Tareas
Codificador	ALBERT, BERT, DistilBERT, ELECTRA, RoBERTa	Clasificación de oraciones, reconocimiento de entidades nombradas, respuesta extractiva a preguntas
Decodificador	CTRL, GPT, GPT-2, Transformer XL	Generación de texto
Codificador-decodificador	BART, T5, Marian, mBART	Resumen, traducción, respuesta generativa a preguntas

- En los transformers, como en el modelo original de atención (BERT, GPT, etc.), tanto los codificadores como los decodificadores son componentes clave en la arquitectura del modelo.
- **Codificadores (Encoder):**
  - En BERT, por ejemplo, el codificador es responsable de procesar la secuencia de entrada y capturar la información contextual en cada una de las palabras.
  - En GPT, el codificador también se utiliza para procesar la secuencia de entrada y capturar información contextual, pero en un formato autoregresivo (de izquierda a derecha).
- **Decodificadores (Decoder):**
  - Los decodificadores son comunes en modelos como el Transformer original utilizado en traducción automática. Toman como entrada la representación vectorial generada por el codificador y generan una secuencia de salida palabra por palabra.
  - En modelos como GPT-2 y GPT-3, no hay un decodificador separado. La generación de texto se realiza autoregresivamente utilizando el mismo mecanismo que en el codificador, pero con una cabecera de atención especial para prevenir la atención a tokens futuros.
- En resumen, en los transformers, los codificadores se encargan de procesar la entrada y capturar la información contextual, mientras que los decodificadores generan secuencias de salida basadas en la información codificada.