Universidad Nacional de Educación a Distancia (UNED)

E.T.S. de Ingeniería Informática — Máster Universitario en Tecnologías del Lenguaje

Minería de Textos

Reconocimiento de Entidades Nombradas

Tarea 3 Obligatoria

Autor: Franly Iris Urbina Franco
DNI: 71.997.418-N
Correo: furbina7@alumno.uned.es
Fecha: 11 de febrero de 2026

Curso 2025-2026 — Tema 3

Comunidad 0

Id	Label	Grado	Grado con pesos
267	vote	51	74.1021
4	election	41	61.9433
135	taxpayer	38	30.4639
100	federal	35	31.6903
64	government	33	39.0482
692	method	32	18.0729
352	tax	30	34.9038
33	voter	29	39.3848
266	force	28	33.8163
22	report	26	30.0436

Top 10 por Grado

Id	Label	Grado	Grado con pesos
267	vote	51	74.1021
4	election	41	61.9433
33	voter	29	39.3848
64	government	33	39.0482
352	tax	30	34.9038
242	candidate	18	34.3173
266	force	28	33.8163
66	propose	26	32.3521
100	federal	35	31.6903
116	state	22	31.1115

Top 10 por Grado con pesos

Índice

Introducción

La evolución humana ha marcado una aceleración del desarrollo cognitivo, tecnológico y social que puede definirse en términos culturales y tecnológicos como un crecimiento exponencial que ha permitido generar, acumular y transmitir conocimiento a un ritmo muy superior al que, biológicamente, podemos asimilar. La necesidad de procesar nuestra información convertida en historia ha creado un paradigma que, tras digitalizar nuestra realidad, nos permite acceder a ella y procesarla al mismo ritmo que lo hace nuestra necesidad de evolución. Sistemáticamente, debemos identificar patrones, hechos y relaciones de manera consistente, mediante procesos que reduzcan la carga cognitiva del análisis humano. Para ello, la transformación del texto en datos estructurados que puedan ser comparados, consultados y analizados automáticamente facilita la gestión de nuestro legado y, con ello, la toma de decisiones y el acceso efectivo al conocimiento, que puede ser extraído, gestionado e interpretado en órdenes inconmensurables de texto.

El lenguaje natural es ambiguo pero funcional, dinámico pero heredado, impreciso pero inferencial: es una arquitectura simbólica finita en sus medios e inagotable en sus combinaciones, moldeada por el uso social, dependiente del contexto y capaz de trascenderlo mediante la abstracción, la escritura y la memoria cultural; optimizado no para la coherencia formal, sino para la comunicación, la cognición compartida y la continuidad civilizatoria

La extracción de información responde a la necesidad de transformar el lenguaje natural de los textos en representaciones estructuradas que permitan su tratamiento computacional. La presente memoria se centrará concretamente en dicha extracción; en la identificación de entidades y en cómo diferentes Reconocedores de Entidades Nombradas (NER, por sus siglas en inglés *Named Entity Recognition*) son capaces de asignar las etiquetas a esos objetos del mundo real o conceptos claramente identificables, como personas, organizaciones, lugares u otras categorías relevantes para su procesamiento.

El esquema de etiquetado utilizado como convención para representar las entidades será IOB, y de igual forma IOBES. Ambos se emplean en tareas de reconocimiento de entidades nombradas para representar, a nivel de token, la extensión y los límites de las entidades dentro de un texto. En el esquema IOB, cada palabra se etiqueta como comienzo de una entidad (B), interior de una entidad (I) o fuera de cualquier entidad (O), mientras que IOBES amplía este esquema añadiendo etiquetas específicas para entidades de un solo token (S) y para el final de una entidad (E). Estas diferencias hacen interesante analizar ambos esquemas, ya que distintas formas de codificar los límites de las entidades influyen en la representación del problema de etiquetado secuencial y en la evaluación de los sistemas NER. Qué diferencias observables en precisión, recall o F1 aportan, o cómo distintos esquemas de anotación pueden condicionar el comportamiento de los modelos, son cuestiones planteadas en el desarrollo de la práctica que se han querido plasmar ampliando la perspectiva del enunciado.

El evaluador utilizado para estimar las etiquetas mediante métricas de validación será el proporcionado por la asignatura, `conlleval.py`, aplicado sobre el corpus `CoNLL-2003`, un

corpus ampliamente utilizado como referencia en tareas de reconocimiento de entidades nombradas.

En primer lugar se empleará NLTK, una librería de carácter académico y didáctico iniciada en 2001, cuyo reconocedor de entidades se apoya en enfoques clásicos basados en *chunking* y modelos menos complejos. Su uso permitirá disponer de un sistema representativo de las primeras aproximaciones al reconocimiento automático de entidades.

Como segundo sistema se utilizará spaCy, una librería moderna de Python publicada inicialmente en 2015, diseñada para aplicaciones prácticas de procesamiento del lenguaje natural. spaCy integra modelos estadísticos entrenados a gran escala y se utiliza de forma habitual en entornos de producción, lo que permitirá analizar el comportamiento de un sistema actual orientado a uso real.

La comparación entre NLTK y spaCy permitirá contrastar dos aproximaciones representativas de distintas etapas en la evolución de las técnicas de procesamiento del lenguaje natural. No obstante, para ampliar esta comparativa hacia modelos de etiquetado estadísticos modernos, se incorporará un tercer reconocedor basado en Transformers, concretamente BERT, propuesto originalmente en 2018. El modelo utilizado, *dbmdz/bert-large-cased-finetuned-conll03-english*, se obtiene de la plataforma *HuggingFace* y se basa en un preentrenamiento sobre grandes corpus textuales como *Wikipedia* y *BooksCorpus*, lo que sitúa el análisis en el contexto de las técnicas más recientes y permite observar de forma empírica la evolución de los enfoques tratados.

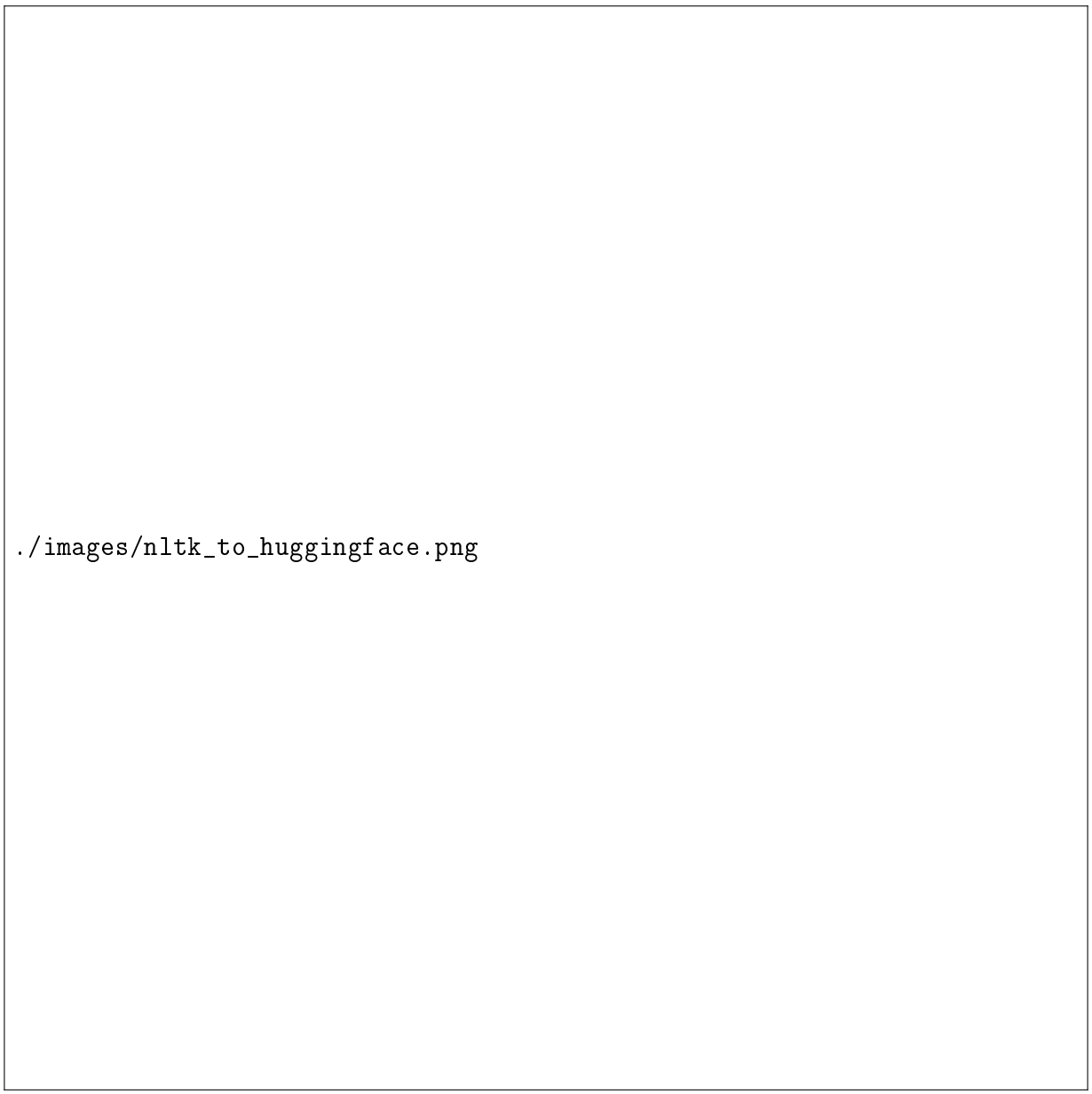


Figura 1: Natural Language Processing (NLTK to Huggingface)

El ejercicio de análisis que se presenta pretende una comparación que busca no solo obtener resultados numéricos y concluir que un sistema supera a otro, sino comprender las decisiones metodológicas implicadas, las limitaciones de cada enfoque y su relación directa con los conceptos teóricos estudiados a lo largo del tema.

1 NLTK - Natural Language Toolkit

NLTK (*Natural Language Toolkit*) es una librería de software libre para procesamiento del lenguaje natural desarrollada en Python. Su origen se sitúa en el ámbito académico, concretamente en la Universidad de Pensilvania, donde fue iniciada en 2001 por Steven Bird y Edward Loper, y posteriormente mantenida y ampliada por Steven Bird y la comunidad académica [**?**]. Desde su creación, NLTK ha tenido como objetivo principal servir como plataforma educativa y de investigación para el estudio del PLN. La librería se concibe como un conjunto modular de herramientas que cubren tareas fundamentales del procesamiento del lenguaje natural. Entre ellas se incluyen la tokenización, el etiquetado morfosintáctico, el análisis sintáctico, el reconocimiento de entidades nombradas, el manejo de corpus y la evaluación de sistemas. NLTK no está orientada a ofrecer modelos de alto rendimiento listos para producción, sino a proporcionar implementaciones claras y accesibles de técnicas clásicas y modernas [**?**].

Se apoya en estructuras de datos explícitas y algoritmos bien documentados, donde muchas de sus funcionalidades se basan en modelos estadísticos tradicionales, reglas lingüísticas y métodos de aprendizaje supervisado sencillos. En el caso del reconocimiento de entidades nombradas, NLTK utiliza un enfoque basado en *chunking*, donde primero se realiza un etiquetado gramatical y posteriormente se agrupan secuencias de palabras en función de patrones aprendidos. El resultado se representa internamente mediante árboles sintácticos que pueden transformarse a esquemas de etiquetado secuencial como IOB.

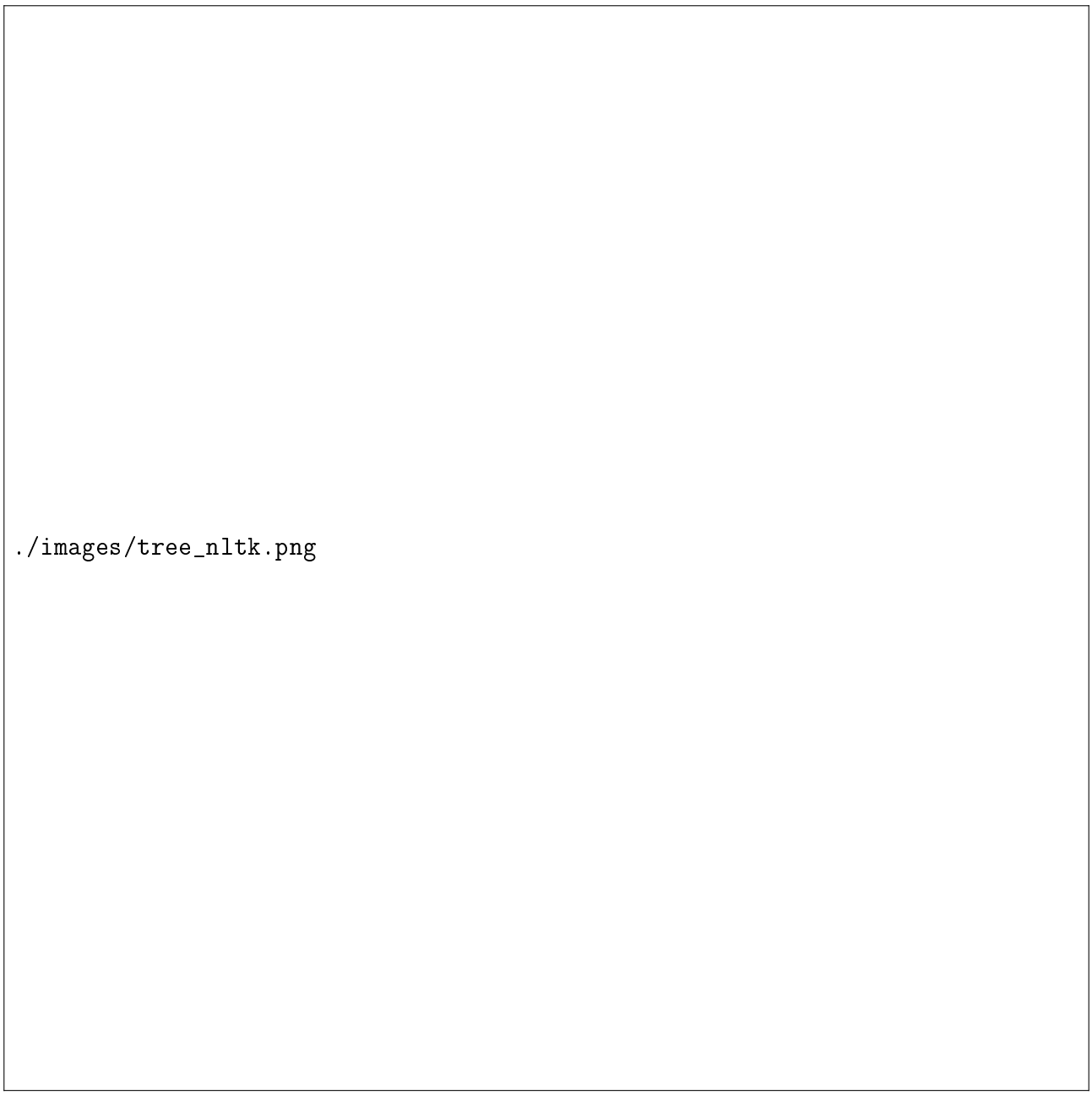


Figura 2: Árbol de análisis generado con NLTK

Una característica central de NLTK es su estrecha integración con corpus lingüísticos. La librería incluye utilidades para descargar, explorar y manipular conjuntos de datos ampliamente utilizados en investigación, lo que facilita la experimentación controlada y la evaluación reproducible. Esta orientación la hace especialmente adecuada para contextos donde la transparencia del proceso y la comprensión de cada etapa son prioritarias. Entre las principales fortalezas de NLTK destaca su valor pedagógico; el diseño de la librería permite inspeccionar fácilmente los pasos intermedios de los algoritmos, comprender las decisiones del modelo y modificar el comportamiento de los componentes. Además, su documentación y ejemplos están pensados para introducir conceptos de PLN de forma progresiva y estructurada [**?**].

Como contrapartida, NLTK presenta limitaciones claras en entornos aplicados; suele ser menos precisa que las librerías modernas, ya que su rendimiento no está optimizado para pro-

cesar grandes volúmenes de texto y su enfoque no se adapta bien a escenarios productivos en tiempo real. Estas limitaciones no son un defecto de diseño, sino una consecuencia directa de su orientación académica, por lo que su inclusión permite observar cómo se aborda el problema desde una perspectiva más clásica y contrastar sus resultados con los obtenidos por herramientas más recientes.

2 spaCy

spaCy es una librería de procesamiento del lenguaje natural desarrollada en Python y Cython, orientada a la construcción de aplicaciones prácticas y sistemas de PLN en entornos de producción. Fue creada por Matthew Honnibal y publicada por primera vez en 2015, con el objetivo explícito de ofrecer herramientas eficientes, robustas y fáciles de integrar en flujos de trabajo reales, en contraste con librerías de carácter más académico [**?**].

Desde su diseño inicial, spaCy se ha centrado en proporcionar componentes de alto rendimiento para tareas del procesamiento del lenguaje natural, como la tokenización, el etiquetado morfosintáctico, el análisis de dependencias y el reconocimiento de entidades nombradas. La librería prioriza la velocidad, la estabilidad y la coherencia de los resultados, lo que la ha convertido en una opción habitual en aplicaciones industriales.

Figura 3: Library architecture

A nivel interno, spaCy se basa en una arquitectura de procesamiento por *pipelines*. Cada documento pasa secuencialmente por una serie de componentes independientes que añaden anotaciones lingüísticas al texto. Estos componentes están implementados en Cython y optimizados para reducir la latencia y el consumo de memoria. El núcleo de spaCy se apoya en estructuras de datos compactas y en modelos estadísticos preentrenados.

En el caso del reconocimiento de entidades nombradas, spaCy emplea modelos supervisados entrenados sobre grandes corpus anotados. En versiones recientes, estos modelos se basan en arquitecturas neuronales que integran representaciones distribuidas de palabras y mecanismos de contexto. El sistema produce etiquetas de entidad directamente alineadas con los tokens generados por su propio tokenizador, lo que garantiza consistencia interna a lo largo del proceso [?].

En este contexto, el modelo de spaCy utilizado será *en_ core_ web_ sm*, un modelo preentrenado para inglés que incluye componentes de tokenización, etiquetado morfosintáctico, análisis de dependencias y reconocimiento de entidades nombradas. Se trata de un modelo neuronal ligero que se apoya en una arquitectura basada en redes neuronales convolucionales (CNN). Emplea un componente denominado `tok2vec`, que genera representaciones vectoriales de los tokens combinando embeddings subléxicos, que se componen de información de caracteres y hashes de palabras, con capas convolucionales de ventana deslizante. Estas capas permiten capturar información contextual local de manera eficiente, agregando información de los tokens vecinos sin recurrir a mecanismos recurrentes ni de atención global. El modelo está diseñado para ofrecer un equilibrio entre velocidad y calidad, y ha sido entrenado sobre corpus anotados de dominio general, orientándose a su uso en aplicaciones prácticas.

Entre las principales fortalezas de spaCy destacan su rendimiento, su diseño orientado a producción y su facilidad de uso, al ofrecer modelos preentrenados listos para usar, una API clara y una documentación enfocada en casos prácticos. Como limitación, spaCy puede funcionar como una caja negra, ya que los detalles del entrenamiento y de la estructura interna del modelo no siempre son accesibles, lo que puede dificultar la comprensión del sistema y la realización de ajustes personalizados. Aun así, destaca por el equilibrio entre calidad, eficiencia y facilidad de integración que ofrece, resultando especialmente adecuado para sistemas orientados a uso real.

3 BERT: Bidirectional Encoder Representations from Transformers

BERT (*Bidirectional Encoder Representations from Transformers*) es un modelo de lenguaje basado en redes neuronales profundas propuesto en 2018 por Jacob Devlin y otros investigadores de Google AI. Su aparición marca un punto de inflexión en el procesamiento del lenguaje natural, al introducir un enfoque de preentrenamiento contextual bidireccional que permite aprender representaciones lingüísticas generales reutilizables en múltiples tareas [?].

El origen de BERT se sitúa en la necesidad de superar las limitaciones de modelos previos basados en representaciones estáticas de palabras y en arquitecturas recurrentes. Hasta su aparición, muchos sistemas procesaban el texto de forma unidireccional o dependían de ventanas de contexto limitadas. BERT adopta la arquitectura Transformer, propuesta en 2017, y se centra exclusivamente en el bloque codificador (*encoder*), prescindiendo de componentes recurrentes o convolucionales.

Desde el punto de vista interno, BERT se compone de múltiples capas de atención de autoatención multi-cabeza y capas *feed-forward* totalmente conectadas. Cada token de entrada se representa como la suma de tres embeddings: el embedding de palabra, el embedding de posición y el embedding de segmento. Esta representación permite al modelo capturar información léxica, posicional y de estructura de frases. El mecanismo de autoatención permite que cada token tenga acceso directo al resto de la secuencia, lo que facilita la modelización de dependencias a largo alcance.

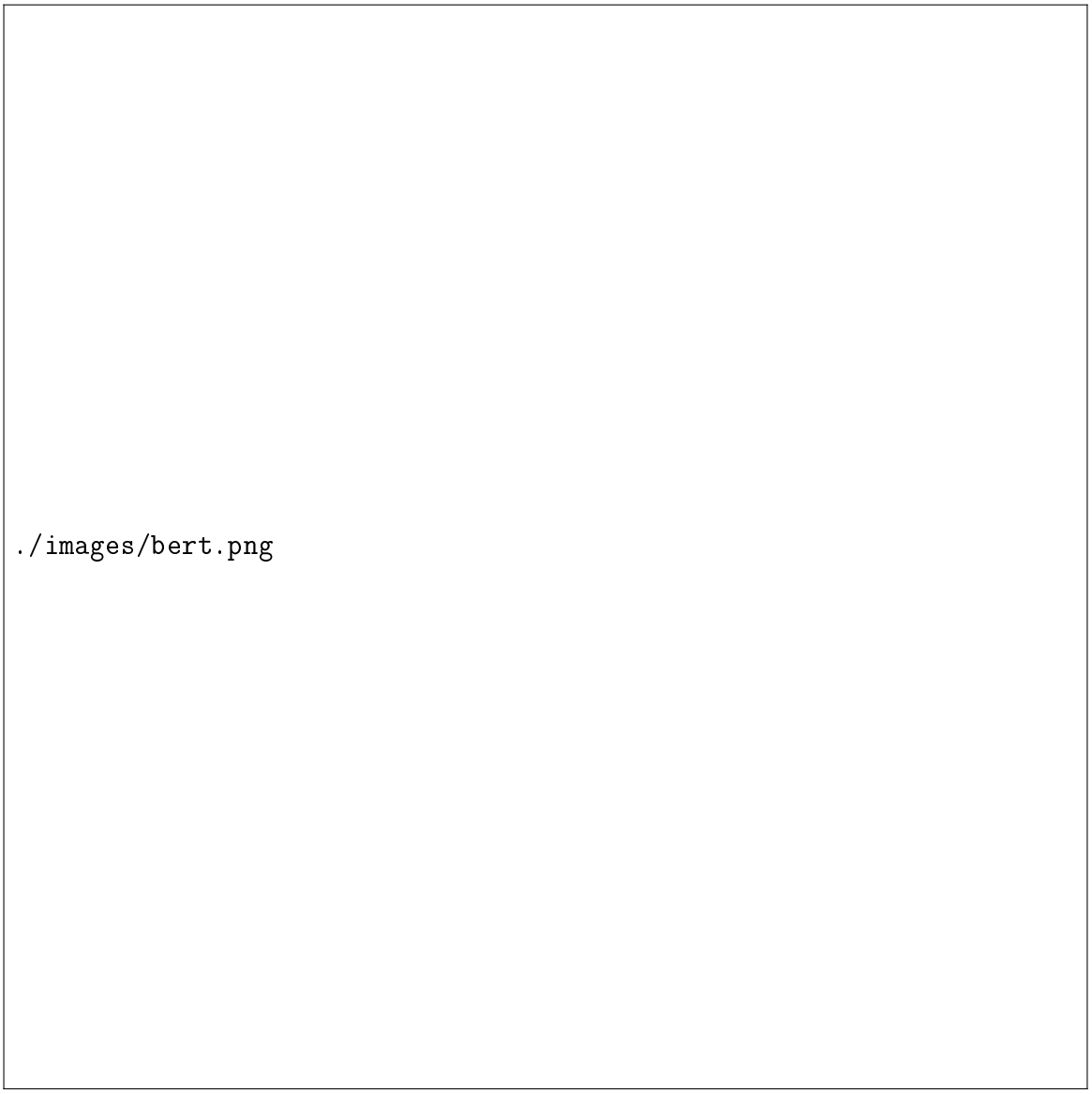


Figura 4: Transformer architecture for NER

BERT, una vez preentrenado, puede ajustarse a tareas concretas mediante un proceso de *fine-tuning*. En tareas de reconocimiento de entidades nombradas, se añade una capa de salida específica y se ajustan todos los parámetros del modelo utilizando un corpus anotado, donde el problema se formula como una tarea de etiquetado secuencial[?].

Entre las principales fortalezas de BERT destaca su capacidad para capturar contexto bidireccional real, que mejora notablemente el rendimiento en tareas que dependen del significado global de la secuencia. Su disponibilidad en plataformas como HuggingFace permite un acceso casi inmediato a modelos potentes que han marcado un punto de inflexión en el rendimiento de los sistemas de PLN, ya que permiten aplicar reconocimiento de entidades sobre grandes volúmenes de texto con un esfuerzo reducido.

Como contrapartida, BERT presenta limitaciones claras en términos de tamaño y complejidad. A cambio de su rendimiento, el coste computacional asociado a su entrenamiento y ajuste genera dependencia de recursos externos con grandes capacidades de cómputo y memoria. No está pensado para entrenarse en entornos con recursos limitados y, además, el modelo actúa como una caja negra, lo que dificulta la interpretación directa de sus decisiones. En el contexto de esta práctica se utilizará un modelo BERT ajustado específicamente para reconocimiento de entidades nombradas, que parte de *BERT-large cased*, preentrenado sobre grandes colecciones de texto general como *Wikipedia* y *BooksCorpus*.

4 Corpus CoNLL-2003

El corpus seleccionado para el desarrollo de la práctica es *CoNLL-2003*, un corpus de referencia ampliamente utilizado en tareas de reconocimiento de entidades nombradas, ya que proporciona un marco de evaluación bien establecido, para la comparativa en contextos académicos bajo distintos reconocedores.

El corpus se distribuye públicamente como parte de los objetivos de la CoNLL[1] y se ha convertido en un punto de referencia para la validación empírica de modelos de reconocimiento de entidades. CoNLL-2003 está compuesto por textos periodísticos en inglés procedentes de la agencia Reuters y anotados siguiendo un esquema de entidades bien definido. Las entidades incluidas pertenecen a las categorías *PER*, *ORG*, *LOC* y *MISC*, y se representan mediante un esquema de etiquetado secuencial IOB. Este formato permite comparar directamente las salidas de NLTK, spaCy y BERT, con el Gold Standard[?].

Una de las principales ventajas de `CoNLL-2003` es la calidad y consistencia de sus anotaciones, al haber sido revisadas manualmente y su formato estructurado facilita el procesamiento automático y la integración con herramientas de evaluación como `conlleval.py`.

Para el desarrollo de la práctica se ha trabajado con un subconjunto del corpus, seleccionando las primeras 30 frases del conjunto de datos. Esta reducción permite limitar el volumen

[1]Una conferencia de primer nivel centrada en enfoques teóricos, cognitivos y científicos motivados por la lingüística computacional y el PNL

de datos a un tamaño manejable y facilita la identificación de las fronteras de frase, porque se preserva la alineación entre tokens y etiquetas de referencia.

1. SOCCER - JAPAN GET LUCKY WIN , CHINA IN SURPRISE DEFEAT
2. Nadim Ladki
3. AL-AIN , United Arab Emirates 1996-12-06
4. Japan began the defence of their Asian Cup title with a lucky 2-1 win against Syria in a Group C championship match on Friday
5. But China saw their luck desert them in the second match of the group , crashing to a surprise 2-0 defeat to newcomers Uzbekistan
6. China controlled most of the match and saw several chances missed until the 78th minute when Uzbek striker Igor Shkvyrin took advantage of a misdirected defensive header to lob the ball over the advancing Chinese keeper and into an empty net
7. Oleg Shatskiku made sure of the win in injury time , hitting an unstoppable left foot shot from just outside the area
8. The former Soviet republic was playing in an Asian Cup finals tie for the first time
9. Despite winning the Asian Games title two years ago , Uzbekistan are in the finals as outsiders
10. Two goals from defensive errors in the last six minutes allowed Japan to come from behind and collect all three points from their opening meeting against Syria
11. Takuya Takagi scored the winner in the 88th minute , rising to head a Hiroshige Yanagimoto cross towards the Syrian goal which goalkeeper Salem Bitar appeared to have covered but then allowed to slip into the net
12. It was the second costly blunder by Syria in four minutes
13. Defender Hassan Abbas rose to intercept a long ball into the area in the 84th minute but only managed to divert it into the top corner of Bitar 's goal
14. Nader Jokhadar had given Syria the lead with a well-struck header in the seventh minute
15. Japan then laid siege to the Syrian penalty area for most of the game but rarely breached the Syrian defence
16. Bitar pulled off fine saves whenever they did
17. Japan coach Shu Kamo said : ' ' The Syrian own goal proved lucky for us
18. The Syrians scored early and then played defensively and adopted long balls which made it hard for us . '
19. Japan , co-hosts of the World Cup in 2002 and ranked 20th in the world by FIFA , are favourites to regain their title here
20. Hosts UAE play Kuwait and South Korea take on Indonesia on Saturday in Group A matches
21. All four teams are level with one point each from one game
22. RUGBY UNION - CUTTITTA BACK FOR ITALY AFTER A YEAR
23. ROME 1996-12-06
24. Italy recalled Marcello Cuttitta

25. on Friday for their friendly against Scotland at Murrayfield more than a year after the 30-year-old wing announced he was retiring following differences over selection
26. Cuttitta , who trainer George Coste said was certain to play on Saturday week , was named in a 21-man squad lacking only two of the team beaten 54-21 by England at Twickenham last month
27. Stefano Bordon is out through illness and Coste said he had dropped back row Corrado Covi , who had been recalled for the England game after five years out of the national team
28. Cuttitta announced his retirement after the 1995 World Cup , where he took issue with being dropped from the Italy side that faced England in the pool stages
29. Coste said he had approached the player two months ago about a comeback
30. " He ended the World Cup on the wrong note , " Coste said

4.1 Análisis del Corpus

El corpus utiliza el formato de anotación CoNLL, empleado en tareas de reconocimiento de entidades nombradas donde se representa cada token colocando una palabra por línea con las distintas anotaciones lingüísticas. Las frases se delimitan mediante líneas en blanco, y cada documento comienza con una marca `-DOCSTART- -X- -X- O` que no forma parte del contenido textual.

```
-DOCSTART- -X- -X- O

EU NNP B-NP B-ORG
rejects VBZ B-VP O
German JJ B-NP B-MISC
call NN I-NP O
to TO B-VP O
boycott VB I-VP O
British JJ B-NP B-MISC
lamb NN I-NP O
. . O O
```

Cada línea del corpus contiene cuatro columnas definidas; la primera columna corresponde al token original del texto; la segunda a la etiqueta morfosintáctica (*Part-of-Speech*) que describe la categoría gramatical del token. En este ejemplo, `EU` tiene la etiqueta `NNP`, que significa *Proper Noun, Singular*, lo que indica que se trata de un nombre propio en singular. La tercera columna representa el sintagma gramatical al que pertenece el token, como sintagma nominal, verbal o preposicional y la cuarta columna contiene la etiqueta de reconocimiento de entidades nombradas, que constituye el *gold standard* del corpus bajo el esquema IOB.

Por ejemplo, la anotación `EU NNP B-NP B-ORG` indica que el token `EU` es nombre propio, sintagma nominal y corresponde al inicio de una entidad nombrada de tipo organización. En

Token	POS	Sintagma gramatical	Etiqueta IOB
EU	NNP	B-NP	B-ORG
rejects	VBZ	B-VP	O
German	JJ	B-NP	B-MISC
call	NN	I-NP	O
to	TO	B-VP	O
boycott	VB	I-VP	O
British	JJ	B-NP	B-MISC
lamb	NN	I-NP	O
.	.	O	O

Cuadro 1: Ejemplo de anotación del corpus CoNLL-2003

cambio, el token `rejects` es un verbo, sintagma verbal y está etiquetado como `O` (Outside), lo que indica que no forma parte de ninguna entidad nombrada.

Las etiquetas IOBES se obtienen a partir del esquema IOB mediante una transformación de las etiquetas. En este proceso, la etiqueta *O* se mantiene sin cambios para los tokens que no pertenecen a ninguna entidad, la etiqueta *B-X* se mantiene como inicio de entidad si el token siguiente está etiquetado como *I-X*; en caso contrario, se transforma en *S-X*, indicando que la entidad está formada por un único token. Las etiquetas *I-X* se mantienen como interior de entidad si el siguiente token también es *I-X*, pero se convierten en *E-X* cuando marcan el final de la entidad. De este modo, IOBES marca de forma más precisa el comienzo, el interior, el final y las entidades unitarias. Así el esquema delimita con mayor claridad los límites de las entidades y facilita el análisis comparativo del rendimiento de los modelos en tareas de reconocimiento de entidades nombradas.

5 Preparación del entorno

Una vez establecidos los modelos que se van a evaluar y el corpus con el que se trabajará, se genera un entorno que permita realizar el experimento en condiciones reproducibles. Para ello, se configura un entorno virtual con Python 3.10 que aísla las dependencias del proyecto e instala las librerías necesarias para el reconocimiento de entidades en inglés mediante el gestor de paquetes `pip`, incluyendo `spacy` v3.7, `nltk` v3.8.1, `numpy` v1.26.4 y `pandas` v2.1.4.

El corpus `CoNLL-2003` se descargó localmente para facilitar su procesamiento directo[2] y, a partir del conjunto de test del corpus, se generó un subconjunto reducido de treinta frases que sirve como base común para los reconocedores. En este contexto se preparó un script que procesa el corpus y genera los ficheros de salida en los formatos IOB e IOBES.

En el caso de spaCy y NLTK, cada herramienta emplea su propio inventario interno de etiquetas, distinto al definido en el corpus `CoNLL-2003`. NLTK utiliza categorías como *PERSON*, *ORGANIZATION* o *FACILITY*, que se pueden observar en el Cuadro **??**, mientras que spaCy

[2]La descarga se realiza directamente desde el *CoNLL-2003 Named Entity Recognition Dataset* [?]

distingue etiquetas como *GPE*, *NORP*, *LAW* o *EVENT*, tal y como se aprecia en el Cuadro **??**. Estas categorías no coinciden directamente con las etiquetas gold del corpus (*PER*, *ORG*, *LOC*, *MISC*), por lo que es necesario realizar una adaptación de las predicciones para que puedan compararse de forma válida con las anotaciones de referencia.

```
$ python ./materiales-tarea3/conlleval.py < eval_nltk.txt
processed 619 tokens with 68 phrases; found: 73 phrases; correct: 0.
accuracy:   0.00%; (non-O)
accuracy:  84.68%; precision:   0.00%; recall:   0.00%; FB1:   0.00
              GPE: precision:   0.00%; recall:   0.00%; FB1:   0.00  33
              GSP: precision:   0.00%; recall:   0.00%; FB1:   0.00  5
              LOC: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
             MISC: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
              ORG: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
     ORGANIZATION: precision:   0.00%; recall:   0.00%; FB1:   0.00  12
              PER: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
           PERSON: precision:   0.00%; recall:   0.00%; FB1:   0.00  21
```

Como se puede observar, sin esta adaptación la evaluación mide diferencias de esquema y no la calidad real del reconocimiento. Los resultados indican que se procesaron 619 tokens y que el corpus contenía 68 entidades anotadas, mientras que el sistema detectó 73. Sin embargo, ninguna de las entidades predichas fue considerada correcta, lo que se refleja en el valor `correct: 0` y en las métricas nulas de precisión, recall y FB1. Este comportamiento evidencia la incompatibilidad entre los esquemas de etiquetado utilizados y justifica la necesidad del mapeo previo.

NLTK	**CoNLL**
PERSON	PER
ORGANIZATION	ORG
GPE	LOC
GSP	LOC
FACILITY	LOC

Cuadro 2: Mapeo de etiquetas entre NLTK y CoNLL-2003

spaCy	**CoNLL**
PERSON	PER
ORG	ORG
GPE	LOC
LOC	LOC
FAC	LOC
NORP	MISC
EVENT	MISC
PRODUCT	MISC
WORK_OF_ART	MISC
LAW	MISC
LANGUAGE	MISC
DATE	MISC
TIME	MISC
PERCENT	MISC
MONEY	MISC
QUANTITY	MISC
ORDINAL	MISC
CARDINAL	MISC

Cuadro 3: Mapeo de etiquetas entre spaCy y CoNLL-2003

Por el contrario, el modelo BERT basado en Transformers ya está entrenado y configurado con el mismo esquema de etiquetas del corpus, por lo que no requiere este paso adicional. Al haber sido ajustado específicamente sobre `CoNLL-2003` y utilizando el mismo criterio de

anotación que el gold standard, su salida es compatible de forma directa con el proceso de evaluación y puede compararse sin necesidad de mapeos adicionales, lo que hace que el modelo Transformer esté mejor alineado con el corpus de referencia.

6 Resultados

La interpretación de los resultados se abordará desde la perspectiva de un problema de clasificación multiclase. Esto se debe a que la valoración se realiza con base en el conjunto de entidades correctamente etiquetadas.

Las tablas de resultados muestran cuatro métricas fundamentales para evaluar los reconocedores de entidades. El *accuracy* mide la proporción total de tokens correctamente etiquetados; como veremos, .estará dominada por los aciertos en la etiqueta `O`, al ser la clase mayoritaria. Por ello, aunque se realizarán observaciones al respecto, se tomará en cuenta el *accuracy non-O* para centrar el análisis en las etiquetas correctamente clasificadas. La *precision* evalúa la fiabilidad de las predicciones: de todas las veces que el sistema asignó una etiqueta, qué porcentaje fue correcto (evitando falsos positivos). Por otro lado, el *recall* mide la cobertura del sistema: de todas las etiquetas que realmente existían en el texto, qué porcentaje fue capaz de predecir (evitando falsos negativos); y la métrica *FB1* (F1-score) que combina *precision* y *recall* en una única medida, ofreciendo una estimación equilibrada del rendimiento global en la que centraremos el análisis.

6.1 NLTK

```
$ python ./materiales-tarea3/conlleval.py < eval_nltk.txt
processed 619 tokens with 68 phrases; found: 73 phrases; correct: 37.
accuracy:  56.52%; (non-O)
accuracy:  93.05%; precision:  50.68%; recall:  54.41%; FB1:  52.48
             LOC: precision:  63.16%; recall:  80.00%; FB1:  70.59  38
            MISC: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
             ORG: precision:   8.33%; recall:  50.00%; FB1:  14.29  12
             PER: precision:  52.17%; recall:  54.55%; FB1:  53.33  23
```

Se observa cómo el *accuracy* global se ve incrementado por la alta frecuencia de la etiqueta `O`. Sin embargo, el *accuracy (non-O)* demuestra que solo el 56.52 % de los tokens pertenecientes a entidades fueron clasificados correctamente.

NLTK presenta una capacidad de detección de entidades cercana al 50 %. En rasgos generales, la precisión más alta se obtuvo en las etiquetas de Localización (`LOC`) con un 63.16 %, contrastando con la gran dificultad para identificar correctamente organizaciones (`ORG`), con apenas un 8.33 %. El *recall* confirma que las etiquetas de localización fueron las mejor recuperadas con un 80.00 %, mientras que el *FB1* refleja un rendimiento equilibrado global del

52.48 %. De las 68 entidades reales, el modelo identificó 73 candidatas, de las cuales solo 37 fueron correctas. Esto resulta en una precisión del 50.68 %, evidenciando que casi la mitad de las etiquetas propuestas por el modelo son incorrectas. Este desempeño limitado es característico de los etiquetadores clásicos de inicios de los 2000, y sirve como punto de referencia histórico para observar la evolución hacia los clasificadores actuales.

6.1.1. NLTK IOB vs IOBES

El script `conlleval.py` también se aplicó tras adaptar el corpus al esquema `IOBES`, con el fin de analizar el impacto de este etiquetado en el rendimiento de los modelos y comprender sus variaciones en distintos escenarios.

```
$ python materiales-tarea3/conlleval.py < eval_nltk_iobes.txt
processed 619 tokens with 68 phrases; found: 73 phrases; correct: 37.
accuracy:  48.91%; (non-O)
accuracy:  91.92%; precision:  50.68%; recall:  54.41%; FB1:  52.48
             LOC: precision:  63.16%; recall:  80.00%; FB1:  70.59  38
            MISC: precision:   0.00%; recall:   0.00%; FB1:   0.00  0
             ORG: precision:   8.33%; recall:  50.00%; FB1:  14.29  12
             PER: precision:  52.17%; recall:  54.55%; FB1:  53.33  23
```

A primera vista vemos que el rendimiento se mantiene en niveles muy similares a los obtenidos con `IOB`. Teóricamente, `IOBES` debería facilitar la detección de fronteras `S-` y `E-`; pero en su lugar, la transición a IOBES en NLTK se ve penalizada en los valores marcados en rojo pero sin existir un cambio en las predicciones reales del modelo como se observa en los resultados en verde, los cuales corresponden a los aciertos de las etiquetas que permanecen invariables. Si nos basamos en estos últimos, el resultado de NLTK es exactamente el mismo, pero el evaluador nos devuelve peores métricas de rendimiento[3].

6.2 spaCy

Con spaCy se observa una evolución interesante en la detección de entidades. Aunque el *accuracy (non-O)* solo aumenta un punto porcentual, el modelo demuestra una especialización notable en categorías clave. A pesar de que las métricas globales como el *precision* y el *FB1* son ligeramente inferiores a las de NLTK debido al alto volumen de falsos positivos (91 entidades propuestas frente a las 68 reales), el sistema introduce con éxito la clase `MISC`, diversificando el reconocimiento.

[3]En este apartado mostraremos los resultados obtenidos y en la Sección **??**, se detallará el motivo que explica esta aparente inconsistencia

```
$ python ./materiales-tarea3/conlleval.py < eval_spacy.txt
processed 619 tokens with 68 phrases; found: 91 phrases; correct: 39.
accuracy:  57.61%; (non-O)
accuracy:  82.71%; precision:  42.86%; recall:  57.35%; FB1:  49.06
              LOC: precision:  86.36%; recall:  63.33%; FB1:  73.08  22
             MISC: precision:  19.15%; recall:  64.29%; FB1:  29.51  47
              ORG: precision:   0.00%; recall:   0.00%; FB1:   0.00  10
              PER: precision:  91.67%; recall:  50.00%; FB1:  64.71  12
```

El valor diferencial de spaCy en este experimento reside en su alta fiabilidad para dos categorías en concreto: alcanza una precisión excelente del 91.67 % en personas (`PER`) y del 86.36 % en localizaciones (`LOC`). Esto supone una mejora sustancial en la calidad de los aciertos; cuando el modelo identifica una persona o un lugar, es altamente probable que sea correcto. Aunque el rendimiento global se ve opacado por el ruido en la clase `MISC` y la nula detección de organizaciones (`ORG`), estos resultados son positivos para aplicaciones donde la prioridad sea la identificación precisa de individuos y ubicaciones, marcando un paso adelante hacia modelos con mayor capacidad de discriminación entre tipos de entidades.

6.2.1. spaCy IOB vs IOBES

Las métricas han resultado idénticas a las obtenidas mediante el esquema `IOB`.

```
$ python materiales-tarea3/conlleval.py < eval_spacy_iobes.txt
processed 619 tokens with 68 phrases; found: 91 phrases; correct: 39.
accuracy:  57.61%; (non-O)
accuracy:  82.71%; precision:  42.86%; recall:  57.35%; FB1:  49.06
              LOC: precision:  86.36%; recall:  63.33%; FB1:  73.08  22
             MISC: precision:  19.15%; recall:  64.29%; FB1:  29.51  47
              ORG: precision:   0.00%; recall:   0.00%; FB1:   0.00  10
              PER: precision:  91.67%; recall:  50.00%; FB1:  64.71  12
```

Esto sugiere que, al igual que en NLTK, el tipo de esquema no afecta a los etiquetadores; las capacidades de predicción se mantienen intactas y el incremento de la complejidad no genera un beneficio demostrable en estos experimentos.

6.3 Transformer

Con el uso de Transformers, el reconocimiento de entidades alcanza un nivel de rendimiento superior, demostrando la potencia de los modelos basados en atención. La precisión en la categoría de personas (`PER`) llega al 100 %, lo que significa que cada vez que el modelo predijo una persona, acertó sin margen de error. Es destacable también el *recall* del 100 % en organizaciones

(ORG), indicando que el sistema fue capaz de localizar todas las entidades de este tipo presentes en el texto, aunque con una precisión del 50 % que sugiere cierta sensibilidad a los falsos positivos en esta clase (que, aun así, sigue siendo muy superior a la de los anteriores etiquetadores).

```
$ python ./materiales-tarea3/conlleval.py < eval_transformer.txt
processed 619 tokens with 68 phrases; found: 67 phrases; correct: 61.
accuracy:  91.30%; (non-O)
accuracy:  98.71%; precision:  91.04%; recall:  89.71%; FB1:  90.37
              LOC: precision:  93.33%; recall:  93.33%; FB1:  93.33  30
             MISC: precision:  86.67%; recall:  92.86%; FB1:  89.66  15
              ORG: precision:  50.00%; recall: 100.00%; FB1:  66.67  4
              PER: precision: 100.00%; recall:  81.82%; FB1:  90.00  18
```

De las 68 entidades, 67 fueron detectadas y de esas, 61 fueron correctas, lo que se traduce en un *FB1* global del 90.37 %. Este valor se ha visto incrementado gracias a que la detección de falsos positivos es drásticamente inferior a la de los etiquetadores anteriores. Un detalle fundamental es el *accuracy (non-O)* del 91.30 %, que refleja una precisión casi total a nivel de token dentro de las entidades. Estos resultados dejan de manifiesto el salto evolutivo del paradigma: mientras que los modelos anteriores sufrían para mantener el equilibrio entre detectar y clasificar erróneamente, el Transformer logra una cobertura (*recall*) y una fiabilidad (*precision*) muy altas. Este experimento confirma que la arquitectura de aprendizaje profundo ha minimizado la brecha de error, permitiendo automatizaciones de alta fidelidad en el procesamiento del lenguaje natural actual.

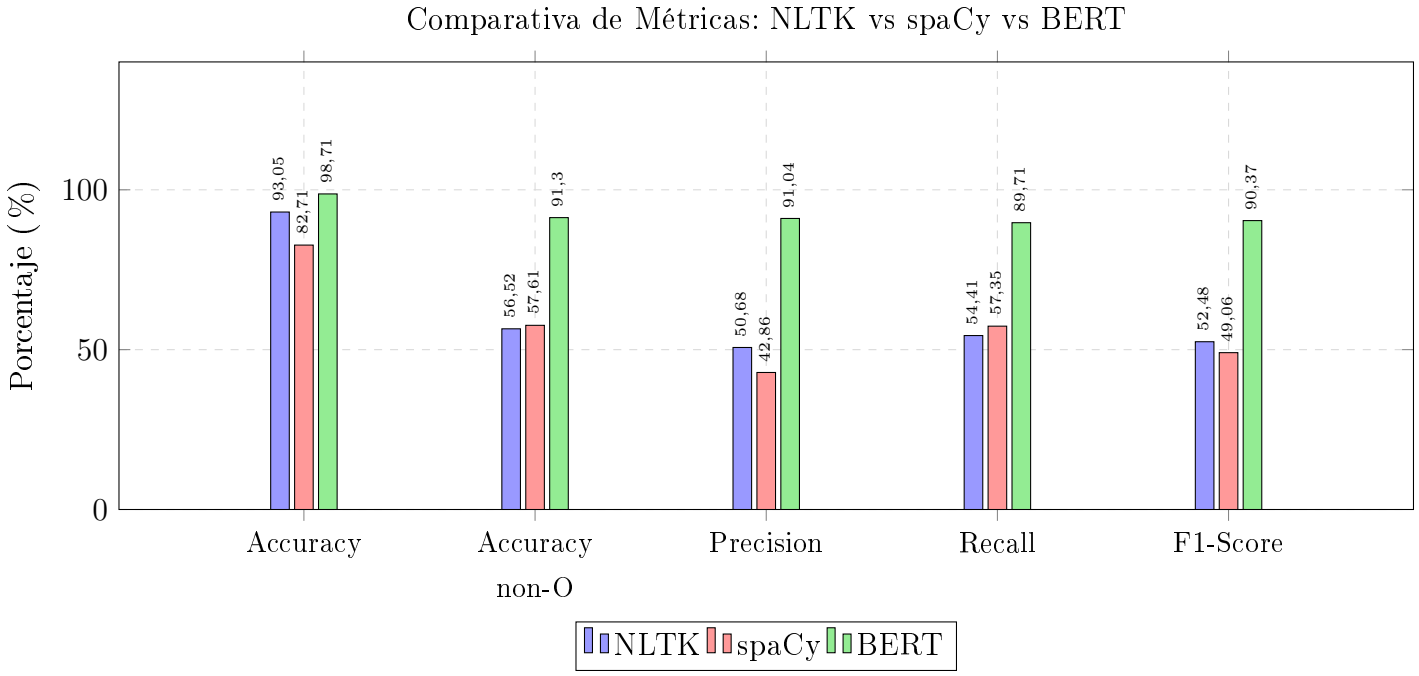
6.3.1. Transformer IOB vs IOBES

```
$ python ./materiales-tarea3/conlleval.py < eval_transformer_iobes.txt
processed 619 tokens with 68 phrases; found: 67 phrases; correct: 61.
accuracy:  91.30%; (non-O)
accuracy:  98.71%; precision:  91.04%; recall:  89.71%; FB1:  90.37
              LOC: precision:  93.33%; recall:  93.33%; FB1:  93.33  30
             MISC: precision:  86.67%; recall:  92.86%; FB1:  89.66  15
              ORG: precision:  50.00%; recall: 100.00%; FB1:  66.67  4
              PER: precision: 100.00%; recall:  81.82%; FB1:  90.00  18
```

BERT mantiene su consistencia al igual que spaCy; las métricas demuestran que el modelo identifica y respeta estrictamente sus fronteras gramaticales, validando que los modelos de aprendizaje profundo generan estructuras de etiquetado altamente robustas.

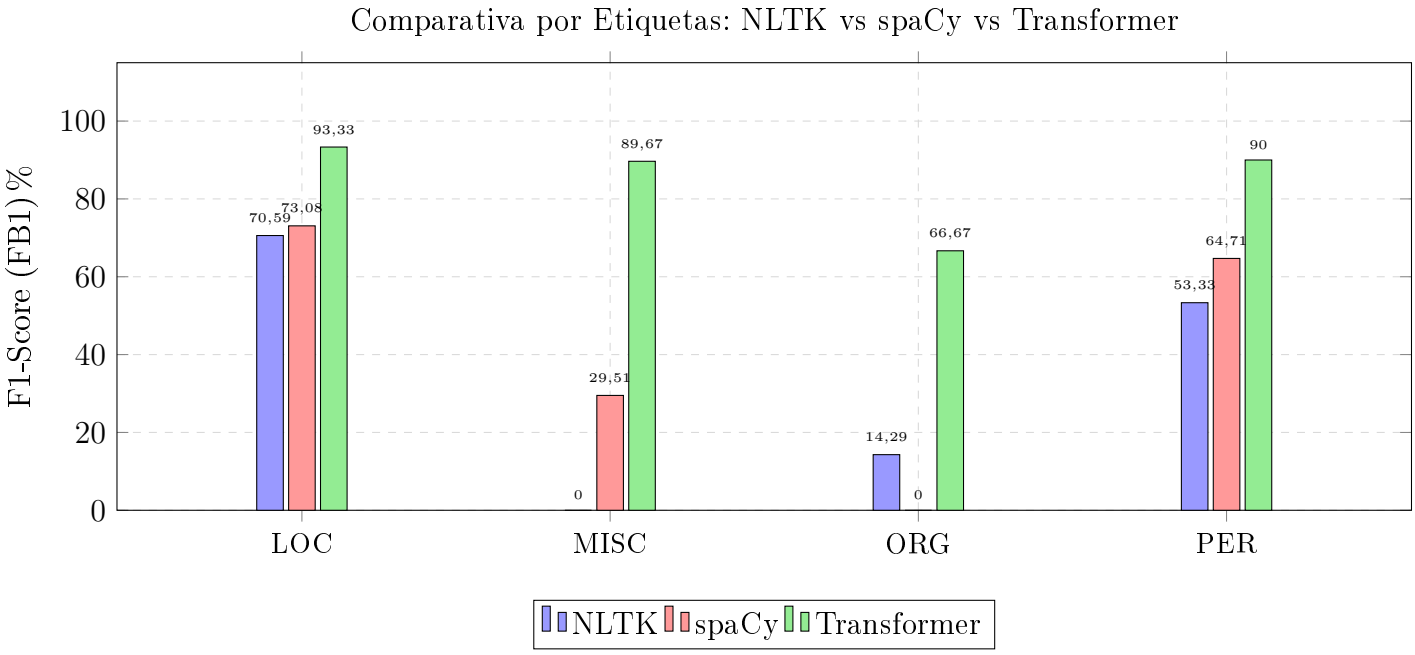
6.4 Análisis de resultados

Para comparar los tres etiquetadores en la detección de entidades mediante el esquema IOB, se presenta la siguiente gráfica comparativa. En ella se evidencia el "sesgo del accuracy global": mientras que NLTK y spaCy muestran valores altos en esta métrica debido al peso de la clase mayoritaria `O`, su rendimiento real en la detección (*Accuracy non-O*, *Precision* y *Recall*) se sitúa en niveles mucho más bajos. Esto confirma la necesidad de evaluar el desempeño mediante métricas que prioricen los aciertos sobre las etiquetas de contenido, como el *Accuracy non-O*.



Se observa que spaCy, pese a tener un *accuracy* global inferior al de NLTK, mantiene una capacidad de cobertura (*recall*) ligeramente superior. Esto indica que spaCy es más .ªrriesgado.ªl identificar entidades; sin embargo, ese aumento de la sensibilidad provoca un incremento de falsos positivos, lo que resulta en información menos fiable y penaliza su *precision*.

Finalmente, el gráfico ilustra el salto cualitativo definitivo de BERT, cuyas métricas mantienen una estabilidad superior al 90 %. Esta consistencia visual confirma que el modelo basado en Transformers no solo detecta más etiquetas, sino que lo hace con una fiabilidad que los métodos tradicionales no pueden alcanzar, minimizando drásticamente la brecha entre detección y error.



Complementando el análisis global, la comparativa por etiquetas permite identificar las fortalezas de cada modelo según la categoría. Se observa que `LOC` es la etiqueta más estable en los tres sistemas, mientras que `ORG` representa el mayor desafío, con un rendimiento nulo en spaCy y muy reducido en NLTK. Es notable cómo spaCy, a pesar de sus limitaciones globales, logra superar a NLTK en las categorías `MISC` y `PER`, lo que confirma su mayor capacidad de especialización. Nuevamente, el modelo Transformer demuestra su hegemonía al mantener un *F1-Score* robusto en todas las clases, logrando rescatar incluso categorías críticas como `ORG` y `MISC` donde los modelos clásicos muestran una eficacia mínima o inexistente.

7 Anexo: Inconsistencia en NLTK. IOB vs IOBES

Tras varios análisis, se ha observado que la caída del *accuracy (non-O)* en NLTK responde a la incapacidad del modelo para generar secuencias estructuralmente coherentes con la jerarquía de etiquetas de IOBES.

En el esquema IOB, el modelo puede obtener un acierto parcial por token si identifica correctamente el inicio de una entidad (`B-`), incluso si falla en la segmentación de los tokens posteriores. Sin embargo, en IOBES esta permisividad desaparece si el modelo no detecta la continuidad de una entidad y la etiqueta erróneamente como palabra única (`S-`).

```
Nadim Ladki. Inconsistencia en ‘‘Nadim’’
  IOB   -> Gold: B-PER  | Pred: B-PER  => ACIERTO
  IOBES -> Gold: B-PER  | Pred: S-PER  => FALLO

United Arab Emirates. Inconsistencia en ‘‘Arab’’
  IOB   -> Gold: I-LOC  | Pred: I-LOC  => ACIERTO
  IOBES -> Gold: I-LOC  | Pred: E-LOC  => FALLO
```

```
Oleg Shatskiku. Inconsistencia en ‘‘Oleg’’
  IOB   -> Gold: B-PER  | Pred: B-PER  => ACIERTO
  IOBES -> Gold: B-PER  | Pred: S-PER  => FALLO

Takuya Takagi. Inconsistencia en ‘‘Takuya’’
  IOB   -> Gold: B-PER  | Pred: B-PER  => ACIERTO
  IOBES -> Gold: B-PER  | Pred: S-PER  => FALLO

Nader Jokhadar. Inconsistencia en ‘‘Nader’’
  IOB   -> Gold: B-PER  | Pred: B-PER  => ACIERTO
  IOBES -> Gold: B-PER  | Pred: S-PER  => FALLO

RUGBY UNION. Inconsistencia en ‘‘RUGBY’’
  IOB   -> Gold: B-ORG  | Pred: B-ORG  => ACIERTO
  IOBES -> Gold: B-ORG  | Pred: S-ORG  => FALLO

Stefano Bordon. Inconsistencia en ‘‘Stefano’’
  IOB   -> Gold: B-PER  | Pred: B-PER  => ACIERTO
  IOBES -> Gold: B-PER  | Pred: S-PER  => FALLO
```

En estos escenarios se produce un error de coincidencia frente al *Gold Standard*, que espera un prefijo `B-`. Por tanto, la degradación en el accuracy de NLTK no se debe a la aparición de nuevos errores, sino a que el esquema IOBES es más estricto. Aunque el modelo falla en las mismas entidades en ambos esquemas, IOB contabiliza el primer token de la secuencia fallida como correcto, mientras que IOBES penaliza la falta de exactitud en la estructura de las entidades compuestas si, tras acertar una primera etiqueta, el modelo no mantiene o no continua con los elementos que completan la secuencia `B-I-E`.

Esta discrepancia explica por qué el esquema IOBES devuelve un porcentaje de acierto por token inferior en NLTK. A diferencia de las redes neuronales de spaCy o de la arquitectura Transformer de BERT, NLTK carece de mecanismos de modelado secuencial capaces de aprender y mantener de forma consistente la estructura interna de las entidades. Mientras que en modelos modernos la predicción de etiquetas se realiza teniendo en cuenta el contexto completo de la secuencia, lo que favorece transiciones coherentes entre etiquetas `B-`, `I-` y `E-` alineadas con el *gold standard*, NLTK puede generar secuencias válidas en IOB pero inconsistentes al transformarse a IOBES. Como resultado, aunque el número de entidades detectadas se mantiene, la normalización penaliza estas incoherencias estructurales, afectando a las métricas finales.

Conclusiones

El análisis comparativo entre NLTK, spaCy y BERT sobre el corpus CoNLL-2003 muestra una clara evolución en la detección automatizada de entidades. NLTK, basado en métodos clásicos de *chunking*, presenta dificultades estructurales que los modelos actuales han superado. Mientras que NLTK destaca por su transparencia y utilidad didáctica al permitir explorar su funcionamiento interno, spaCy y BERT ofrecen resultados mucho más sólidos para un uso real gracias a sus arquitecturas de redes neuronales y mecanismos de atención.

Un hallazgo clave es la especialización de los modelos; spaCy ofrece una gran precisión en categorías específicas como personas y localizaciones, aunque genera un mayor volumen de falsos positivos en otras etiquetas. Por su parte, BERT es el etiquetador más robusto, manteniendo métricas de calidad superiores al 90 % en casi todas las clases. Esto confirma que el aprendizaje profundo ha reducido drásticamente el margen de error, siendo una herramienta crucial para procesar y clasificar volúmenes masivos de texto con alta fidelidad.

La comparativa entre los esquemas IOB e IOBES deja claro que para modelos modernos esta diferencia es irrelevante, ya que su consistencia interna es total. Sin embargo, en el caso de NLTK, el formato IOBES sirve para exponer errores de segmentación que el esquema IOB pasaba por alto al ser más permisivo. A pesar de estas limitaciones, NLTK sigue siendo una primera aproximación válida para explorar modelos sencillos y transparentes, algo que la naturaleza de "caja negra" de spaCy y BERT dificulta.

Sin duda, se ha visto reflejado el éxito de los Transformers en entornos reales, no solo por su facilidad de uso mediante plataformas como HuggingFace, sino que al estar preentrenados, permiten una implementación rápida y resultados excelentes de forma inmediata. Esta ventaja se refuerza si además, el modelo ha sido ajustado bajo el estándar del corpus, lo que evita adaptaciones técnicas o mapeos manuales y ofrece una alta fiabilidad con un esfuerzo mínimo. Dado que actualmente ya se alcanza casi el 100 % de precisión en algunas entidades, no es de extrañar que en pocos años estos sistemas logren un rendimiento prácticamente perfecto, posicionándose como una de las necesidades más importantes en problemas de detección y clasificación para el Reconocimiento de Entidades Nombradas, para satisfacer la necesidad crítica de estructurar información en grandes volúmenes de texto con una precisión extraordinaria.

Referencias

[1] Elavarasan P. *Natural Language processing (Basics to SOTA models) Part-1*. Disponible en https://digitaldata.science.blog/2022/01/18/natural-language-processing-basics-to-sota-models-part-1/.

[2] Wikipedia. *Natural Language Toolkit*. Disponible en https://en.wikipedia.org/wiki/Natural_Language_Toolkit.

[3] NLTK Team. *Natural Language Toolkit (NLTK) Documentation*. Disponible en https://www.nltk.org/. 2026.

[4] Luis Merino Ulizarna. *NLTK: tus primeros pasos con Procesamiento del Lenguaje Natural*. Disponible en https://adictosaltrabajo.com/2023/07/27/nltk-python/. 2026.

[5] Wikipedia. *spaCy*. Disponible en https://en.wikipedia.org/wiki/SpaCy.

[6] spaCy Developers (Explosion AI). *spaCy: Library Architecture*. Disponible en https://spacy.io/api. 2025.

[7] Explosion AI. *spaCy: Model Architectures*. Disponible en https://spacy.io/api/architectures.

[8] Wikipedia. *BERT (modelo de lenguaje)*. Disponible en https://es.wikipedia.org/wiki/BERT_(modelo_de_lenguaje).

[9] Marcello Politi (2022). *Custom Named Entity Recognition with BERT*. Disponible en https://towardsdatascience.com/custom-named-entity-recognition-with-bert-cf1fd4510804/.

[10] Vikas Yadav and Steven Bethard (2019). *A Survey on Recent Advances in Named Entity Recognition*. Disponible en https://arxiv.org/abs/1910.11470.

[11] DeepAI. (s.f.). *CoNLL-2003 Named Entity Recognition Dataset*. Disponible en https://data.deepai.org/conll2003.zip.