Aplicaciones de la lingüística computacional

Proyecto NERC

Federico Ortega Riba

22 de mayo del 2024

Índice

1. Introducción

Este proyecto de clasificación y reconocimiento de entidades nombradas (NERC) se enmarca dentro de la asignatura Aplicaciones de la lingüística computacional. A lo largo del primer bloque, se han realizado actividades tanto teóricas como prácticas que serán la base fundamental del presente informe, de manera que se reutilizará toda la información posible con el fin de maximizar la utilidad de las actividades que se han entregado, lo que incluye tanto información sobre esta tarea de reconocimiento como el código utilizado para el ejemplo real que tratamos.

Una entidad nombrada es un término cuya importancia para la extracción de información se destacó en el congreso MUC-6. Es un tipo de palabra que reconoce elementos que tienen propiedades similares a una colección de elementos. También se le denomina un designador rígido, un elemento atómico o un miembro de una clase semántica que puede variar según el campo de interés. Por ejemplo, en el ámbito biomédico, las entidades de interés son genes y productos genéticos. Por otra parte, en el ámbito general, personas, organizaciones, números, fechas, horas, entre otros, son entidades importantes.

Las entidades nombradas pueden clasificarse en varios tipos, que incluyen, pero no se limitan a: personas, lugares, organizaciones, fechas, cantidades, números, productos, eventos, términos médicos, términos legales, u otros según el dominio. La tarea de NERC se utiliza en diversas aplicaciones de procesamiento del lenguaje natural, como la extracción de información, sistemas Respuesta-Pregunta, la traducción automática, simplificación automática de texto, agrupamiento de texto, recuperación de información, población de ontologías o bases de datos del conocimiento, minería de datos o búsqueda semántica. La NERC de dominio general se realiza en textos que abarcan una amplia variedad de temas, mientras que la NERC de dominios específicos se centra en documentos de áreas particulares, como textos biomédicos, legales, financieros, etc. En el caso de dominios específicos, se requiere un modelo más especializado y entrenado en vocabulario y patrones específicos de ese dominio.

Teniendo esta información en cuenta, la tarea que vamos a realizar se enmarca dentro del dominio jurídico y se basa en extraer este tipo de entidades nombradas en dos campos concretos, cuyo proceso se explica más adelante.

2. Objetivos

El objetivo de esta tarea es entrenar un modelo de NERC usando la herramienta spaCy. Para ello, se deben extraer dos tipos de entidades, en este caso concreto: entidades relativas a las leyes y entidades sobre organizaciones. Una vez extraídas dichas entidades, se deben preprocesar, tokenizar y normalizar, así como adaptarlas al formato de spaCy para poder entrenar nuestro modelo. El entrenamiento seguirá un *gold standard*, por lo que se revisará el mayor número de entidades manualmente con el fin de obtener unos parámetros de validación óptimos.

3. Metodología

La tarea que se pretende realizar se puede dividir en cuatro pasos:

1. Implementación de la anotación.
2. Validación del *silver standard*.
3. Entrenamiento.
4. Evaluación.

Durante todo el proyecto, se ha trabajado desde Google Collaboratory, disponible con este enlace. Las tareas descritas se presentan con comentarios en un solo cuaderno y el conjunto de datos usado para cada paso se puede consultar en el archivo *OrtegaRiba_Federico_ProyectoNERC* anexo al informe.

El código seguirá Python 3 y, al margen de las funciones propias, entre las librerías y módulos utilizados, encontramos los siguientes:

- OS (Operating System): un módulo que nos permite interactuar con el sistema operativo. A través de sus funciones, manipulamos rutas, creamos y eliminamos carpetas de Drive. En esta tarea de NERC, es útil para administrar grandes cantidades de documentos en distintos directorios y se ha usado, en concreto, para poder abrir los archivos en XML y HTML y poder convertirlos a TXT u otros formatos.

- RE (Regular Expressions): un módulo para poder crear patrones de búsqueda de texto según un formato establecido. Una vez se realiza un estudio lingüístico del texto, se definen las expresiones regulares que nos permiten extraer únicamente los datos que necesitamos. Esta función se ha utilizado para localizar y devolver entidades legales.

- spaCy: una biblioteca de NLP que nos permite normalizar, tokenizar, cargar modelos, entrenarlos y aplicar funciones propias de NERC. Proporciona una amplia e interesante gama de funciones para tareas del procesamiento del lenguaje natural, incluido el reconocimiento de entidades nombradas.

- ElementTree: un módulo utilizado para analizar y manipular documentos XML. Con él, se ha filtrado el texto de etiquetas concretas.

- BeautifulSoup: una biblioteca de Python utilizada fundamentalmente para *web scraping* de documentos HTML. Al igual que ElementTree, se ha utilizado para recuperar texto a partir de etiquetas, en este caso HTML.

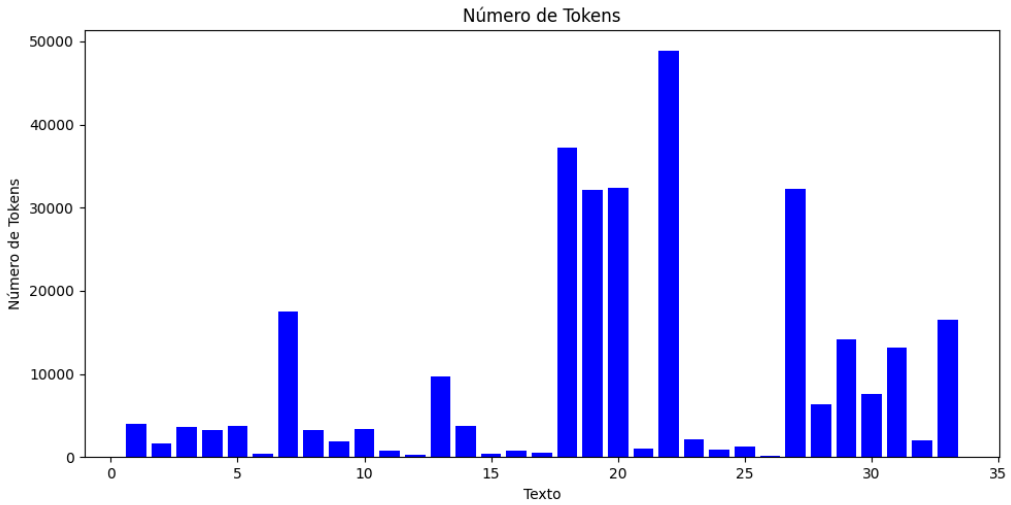
4. Datos

En este proyecto se ha trabajado con 3 tipos de documentos legislativos distintos que conforman un corpus de 42 textos:

- BOE (Boletín Oficial del Estado): diario oficial de España donde se publican textos legislativos de carácter público. En total, se han recogido 22 textos en formato XML.
- Eur-Lex: publicaciones legislativas disponibles en el portal online de la Unión Europea. En total, se han recogido 10 textos en formato HTML.
- JRC (Joint Research Center): dirección general de la Comisión Europea que proporciona un repositorio científico y legal en línea. En total, se han recogido 10 textos en formato XML.

Estos documentos se han dividido en un 80 % para el conjunto de entrenamiento y un 20 % para el conjunto de test. Para la extracción de los datos de los ficheros con extensión XML, se creó una función con ElementTree que recogía el texto en las etiquetas <p>, , <head> y <title>, mientras que para los archivos con extensión HTML, solo se recogió el texto contenido en las etiquetas <p> a través de una función con BeautifulSoup.

Una vez extraído el texto y almacenado en una carpeta de Drive, la siguiente fase es la normalización, para la que también se ha creado una función que elimine espacios y caracteres mal codificados. Cuando los datos normalizados se han reescrito en los archivos originales, se pasa a cargar los modelos de spaCy (en este caso hemos trabajado con el modelo grande en español) para tokenizar los textos normalizados. Mediante una función estadística, conseguimos los siguientes resultados:



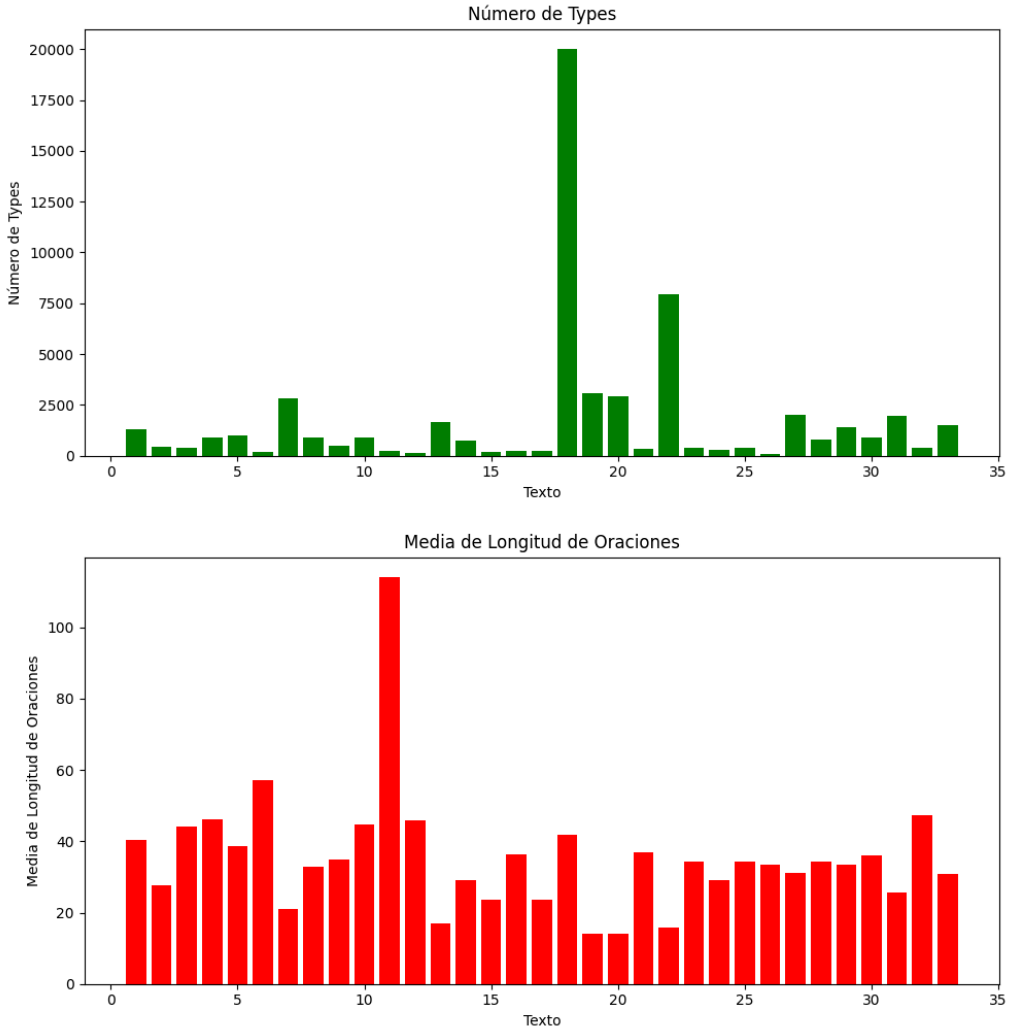


Figura 1. Resultados estadísticos del número de tokens, número de *types* y media de longitud de oraciones de cada fichero.

5. Anotación: tipo de entidades y método de anotación

a. Tipos de entidades

Como se definía en los objetivos de la tarea, se han anotado dos tipos de entidades:

1. Organizaciones (ORG): según el DLE, «asociación de personas regulada por un conjunto de normas en función de determinados fines»[1]. Dentro de esta categoría, es importante subrayar la importancia de la correferencia a organizaciones de textos legales, ya que es una práctica común definir las partes de un acuerdo o que aparecen en un documento

[1] Real Academia Española. (s.f.). Organización. En *Diccionario de la lengua española*. Recuperado el 14 de mayo de 2024, de https://dle.rae.es/corpus

legal para más tarde referirse a alguna de las partes mediante un nombre propio abreviado. Para ilustrar esto, tomemos como ejemplo el texto 21979A0412(07)_ES, donde aparece en primera instancia *Comité de Prácticas «Antidumping»*, más tarde referido solo como *Comité*.

2. Entidades relacionadas con la legislación (LEGAL): no solamente se incluyen leyes, sino también reales decretos, decisiones, reglamentos, tratados y órdenes.

b. Método de anotación

Se han diseñado dos vías de anotación de entidades diferentes. En cuanto a la primera vía, se ha optado por utilizar la anotación disponible en spaCy para las organizaciones, se ha definido una función *anotar_spacy_ORG,* donde se implementan las funciones para extraer etiquetas *ORG*. Sin embargo, como se ha revisado que la herramienta también devuelve como resultado *Real Decreto*, se indica que si la reconoce como entidad, no la anote. La función también devuelve las entidades en el formato que requiere spaCy, es decir, una lista de tuplas donde la primera es la oración al completo y la segunda es un diccionario con *entities* como clave y cuyos valores son una lista de tuplas donde se separan por comas la entidad, la posición de inicio, la posición de final y el tipo de entidad. Aunque en el formato definitivo que tendremos que utilizar para entrenar el modelo no se permite añadir el nombre de la entidad reconocida, sino simplemente sus límites y el tipo, en este caso se ha incluido para revisar si se está ejecutando correctamente la tarea.

Para las entidades legales, se han utilizado expresiones regulares, en concreto, las siguientes:

- Leyes y Reales Decretos: expresión que devuelve varios tipos de formato:
    - Ley 3/2017: la palabra *Ley* hasta una coma y deja de devolver información si no hay un *de* detrás.
    - Ley 39/2015, de 1 de octubre: si hay un *de* detrás, busca hasta la siguiente coma si no hay un *de* detrás de esta coma.
    - Ley 40/2015, de 1 de octubre, de Régimen Jurídico del Sector Público o Ley 13/1998, de 4 de mayo, de Ordenación del Mercado de Tabacos y Normativa Tributaría se publican los precios de venta al público de determinadas labores de tabaco en Expendedurías de Tabaco y Timbre del área del Monopolio: si hay un *de* detrás de la coma, busca hasta el siguiente signo de puntuación o para cuando encuentra una de las siguientes frases: *se publican*, *se publica*, *se anuncian*, *se anuncia*, *se notifican*, *se notifica*, *se comunican*, *se comunica*, *se informan* o *se informa*.
- Decisión: la palabra *decisión* hasta que se encuentre una apertura de paréntesis seguida de "DO" o hasta el final de la línea. Ejemplo: Decisión 2000/750/CE del Consejo, de 27 de

noviembre de 2000, por la que se establece un programa de acción comunitario para luchar contra la discriminación (2001-2006) (DO L 303 de 2.12.2000, p. 23).

- Reglamento: la palabra *reglamento* hasta el siguiente punto. Ejemplo: Reglamento (CE) no 1655/2000 del Parlamento Europeo y del Consejo, de 17 de julio de 2000, relativo al instrumento financiero para el medio ambiente (LIFE).
- Tratado: la palabra *tratado* hasta la siguiente coma. Ejemplo: Tratado constitutivo de la Comunidad Europea
- Orden: dos posibilidades:
    - Orden APU/2245/2005: combinación de letras mayúsculas, dígitos y barras inclinadas (/) que puede opcionalmente terminar con una barra inclinada seguida de cuatro dígitos.
    - Orden de 4 de diciembre de 1986: órdenes que incluyen una fecha. Este formato puede empezar opcionalmente con la palabra *de* seguida de un espacio, luego uno o dos dígitos que representan un día del mes, seguido de *de* y el nombre de un mes en letras, y finalmente *de* seguido de cuatro dígitos que indican un año. Ejemplos de este formato incluyen *de 15 de marzo de 2023* o *12 de abril de 2022*.

Tras hacer la anotación automática de ambas entidades, conseguimos un *silver standard* con 3126 entidades de organizaciones y 281 entidades legales. Un ejemplo de los resultados lo observamos en la Figura 2:

Texto de la oración	Texto de la entidad detectada	Tipo de la entidad	Posición_inicio	Posición_fin
Principalmente, esta orden determina que los	Orden ITC/3292/2008	LEGAL	460	479
El apartado octavo de la mencionada Orden d	Orden de 16 de julio de 1998	LEGAL	36	64
Contra la presente resolución, que no pone fir	Ley 39/2015, de 1 de octubre, de Procedimiento Administrativo C	LEGAL	288	389
La Orden IET/389/2015, de 5 de marzo, actual	Orden IET/389/2015	LEGAL	3	21
Las autoridades competentes de la Comunida	Orden IET/389/2015	LEGAL	237	255
Esta resolución no será de aplicación a los gas	Ley 34/1998, de 7 de octubre, del sector de hidrocarburos.	LEGAL	400	458
El coste de comercialización sin impuestos, co	Orden IET/389/2015	LEGAL	156	174
En el precio máximo de venta indicado en el a	Orden IET/389/2015	LEGAL	167	185

Figura 2. Ejemplo de hoja de cálculo con las entidades legales.

Dichas entidades las combinamos con una función y creamos los archivos en TXT con las entidades de cada documento. Al disponer de los archivos en formato de texto plano, es sencillo crear un excel para poder anotar y corregir los fallos que haya habido en el *silver standard*. Se ha revisado la totalidad de las entidades legales y la mitad de las entidades sobre organizaciones y se han realizado los siguientes cambios:

- Para las entidades ORG: Como hay demasiadas entidades incorrectas, la manera más rápida que encontramos de solucionar este problema es pasar una lista de entidades que no queremos que reconozca como ORG a la función que creamos en el paso previo. Esta lista comprende un total de 258 elementos únicos.
- Para las entidades LEGAL: Comprobamos que hay 2 entidades que no se reconocen bien:

- Reglamentos: solo había establecido una expresión regular para que abarcase desde *Reglamento* hasta el punto siguiente; sin embargo, en muchas oraciones vemos que sigue la misma estructura que *Decisión* e incluye un *(DO L 308...)*. Para solucionarlo, añadimos esta opción como posible parada.

- Tratados: solamente hay un tipo de tratado y es el *Tratado constitutivo de la Comunidad Europea* o *Tratado constitutivo de la Comunidad Económica Europea*. Limitamos la expresión regular a estas dos opciones.

Tras estos cambios, el número de entidades legales se mantiene, pero el de organizaciones experimenta un descenso de la mitad y llega a 1582. En última instancia, para finalizar la fase de anotación, creamos una función que combina ambas entidades y esta vez sí se adapte al formato requerido por spaCy. Teniendo en cuenta que las entidades ya han sido anotadas, no necesitamos conocer cuáles ha reconocido, así que eliminamos esta información de la tupla, que nos facilitará el proceso de entrenamiento.

6. Entrenamiento

Para el entrenamiento seguiremos el código proporcionado en las semana 6 de la asignatura. No obstante, debido a problemas de solapamiento entre entidades, se han tenido que hacer algunos cambios y así impedir que haya conflicto al entrenar. Esto sucede en casos donde entidades, como organizaciones, recogen a otras más pequeñas. Si nos fijamos en el ejemplo *Consejo de Cooperación Aduanera de la Unión Europea*, veremos que puede tener dos tipos de segmentación: la frase como una entidad completa o dos entidades separadas como *Consejo de Cooperación Aduanera* y *Unión Europea.* Con el fin de suplir las carencias de segmentación se modifica la función de entrenamiento y se configura otra que evita estos solapamientos.

```
def filter_overlapping_spans(entities):
    filtered_entities = []
    entities = sorted(entities, key=lambda x: x[0])
    i = 0
    while i < len(entities):
        start, end, label = entities[i]
        j = i + 1
        while j < len(entities) and entities[j][0] < end:
            if entities[j][1] > end:
                end = entities[j][1]
            j += 1
        filtered_entities.append((start, end, label))
        i = j
    return filtered_entities

def entrenar(datos, iteraciones):
    datos_gold = datos
    nlp = spacy.blank('es')
    if 'ner' not in nlp.pipe_names:
```

```
            ner = nlp.create_pipe('ner')
            nlp.add_pipe("ner", last=True)

        for _, anns in datos_gold:
            for ent in anns.get('entities'):
                ner.add_label(ent[2])

        other_pipes = [pipe for pipe in nlp.pipe_names if pipe != 'ner']
        with nlp.select_pipes(disable=other_pipes):
            optimizer = nlp.begin_training()
            for n in range(iteraciones):
                print("Iteración número " + str(n))
                random.shuffle(datos_gold)
                losses = {}
                for texto, anns in datos_gold:
                    doc = nlp.make_doc(texto)
                    golds = []
                    for ent in anns.get('entities'):
                        start, end, label = ent[0:3]
                        golds.append((start, end, label))
                    golds = filter_overlapping_spans(golds)
                    ejemplo = Example.from_dict(doc, {"entities": golds})
                    nlp.update(
                        [ejemplo],
                        drop=0.2,
                        sgd=optimizer,
                        losses=losses
                    )
                print(losses)
        return nlp
```

En la función entrenar, se crea un modelo spaCy vacío para el idioma español. Se agrega un componente de NER al *pipeline* del modelo si no está presente. Luego, se recorren las anotaciones en los datos de entrenamiento para agregar las etiquetas de las entidades al componente de NER del modelo. Este proceso asegura que el modelo sepa qué etiquetas debe aprender a reconocer. El siguiente paso en entrenar es desactivar todos los componentes del *pipeline* excepto el de NER durante el entrenamiento. Se inicia el entrenamiento del modelo con un optimizador y se ejecuta un ciclo de entrenamiento por el número de iteraciones especificado. En cada iteración, se mezclan aleatoriamente los datos de entrenamiento, y se calculan las pérdidas en el proceso de actualización del modelo con los ejemplos anotados, los cuales se crean a partir de los textos y sus entidades correspondientes. Se imprime la pérdida en cada iteración para controlar el progreso del entrenamiento. Por otra parte, la función *filter_overlapping_spans* se encarga de ordenar las entidades por su posición de inicio y eliminar las superposiciones. Para cada entidad, se verifica si se solapa con las siguientes y se ajustan las posiciones de inicio y fin en consecuencia. Esta función devuelve una lista de entidades no superpuestas, lo que es crucial para asegurar que las anotaciones utilizadas para entrenar el modelo sean válidas y no generen conflictos durante el proceso de actualización del modelo.

Se han hecho un total de 30 iteraciones en el entrenamiento con un *drop* de 0,2, que ha durado alrededor de una hora.

7. Evaluación de resultados

Una vez entrenado el modelo, se ha almacenado en una carpeta de Drive para poder evaluarlo. Para la evaluación se usan cuatro métricas:

- *Token accuracy*: mide la proporción de tokens que el modelo ha clasificado correctamente entre todos los tokens del conjunto de datos.
- *Token precision*: proporción de tokens etiquetados como entidades por el modelo entre el número de predicciones totales del modelo.
- *Token recall*: proporción de tokens etiquetados como entidades por el modelo entre todas las entidades del *gold standard*.
- *Token F-Score*: media armónica de la precisión y la cobertura.

Los resultados son excelentes, ya que se ha conseguido un 100 % en todas las métricas de evaluación y el modelo es capaz de seguir todos los patrones con los que lo hemos entrenado para evaluar anotaciones nuevas.

8. Retos y conclusiones

La tarea NERC, a pesar de haber nacido hace más de medio siglo, sigue presentando complicaciones tanto lingüísticas como informáticas. Entre las que hemos tenido que resolver durante este proyecto podemos encontrar: la anidación de entidades, que ha sido el problema principal del entrenamiento, no a gran escala pero sí en ciertos ejemplos en los que se solapaban entidades dentro de otras; la ambigüedad, ya que hemos encontrado las mismas entidades con nombres distintos dentro de una misma expresión lingüística, como es el caso de *Consejo de Cooperación Aduanera de la Unión Europea* que hemos analizado y la anotación de los datos de entrenamiento, puesto que es una tarea tan minuciosa como tediosa que requiere tiempo y recursos.

Aun así, a lo largo del proyecto se han podido superar las dificultades que entraña y hemos observado cómo es el flujo de trabajo de un proyecto real y la tarea de orfebrería que supone la anotación de entidades. Las métricas de evaluación utilizadas nos dan resultados óptimos y comprobamos la consistencia del modelo para nuevos textos.

9. Referencias bibliográficas

European language equality. (2023b). En A. Way & G. Rehm (Eds.), *Cognitive technologies* (1.ª ed.). https://doi.org/10.1007/978-3-031-28819-7

Ghosh, S., & Gunning, D. (2019). *Natural Language Processing Fundamentals: Build Intelligent Applications That Can Interpret the Human Language to Deliver Impactful Results*.

Goyal, A., Gupta, V. y Kumar, M. (2018). Recent Named Entity Recognition and Classification techniques: A systematic review. In *Computer Science Review*, Vol. 29, p. 21-43. DOI: https://doi.org/10.1016/j.cosrev.2018.06.001

Grishman, B. Sundheim. (1996). Message understanding conference-6: A brief history. In *COLING Proceedings*, pp. 466–471. Disponible en: https://aclanthology.org/C96-1079/

ICC. (2022). *Anotación de corpus lingüísticos: metodología utilizada en el IIC*. Instituto de Ingeniería del Conocimiento. https://www.iic.uam.es/whitepapers/anotacion-corpus-linguisticos-metodologia-utilizada-iic/

Li, J., Sun, A., Han, J. and Li, C. (2022). "A Survey on Deep Learning for Named Entity Recognition," in *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50-70, 1 Jan. 2022, doi: 10.1109/TKDE.2020.2981314.

M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd. (2020). *spaCy: Industrial-strength Natural Language Processing in Python*. DOI: 10.5281/zenodo.1212303

Real Academia Española. (s.f.). Organización. En *Diccionario de la lengua española*. Recuperado el 14 de mayo de 2024, de https://dle.rae.es/corpus

Samy, D. (2021). Reconocimiento y clasificación de entidades nombradas en textos legales en español. *Procesamiento del Lenguaje Natural*, *67*(67), 103-114. https://doi.org/10.26342/2021-67-9.